


2011

# Combining Natural Language Processing and Statistical Text Mining: A Study of Specialized Versus Common Languages

Jay Jarman

*University of South Florida, [jay@jayjarman.com](mailto:jay@jayjarman.com)*

Follow this and additional works at: <http://scholarcommons.usf.edu/etd>

 Part of the [American Studies Commons](#), and the [Databases and Information Systems Commons](#)

---

## Scholar Commons Citation

Jarman, Jay, "Combining Natural Language Processing and Statistical Text Mining: A Study of Specialized Versus Common Languages" (2011). *Graduate Theses and Dissertations*.  
<http://scholarcommons.usf.edu/etd/3166>

This Dissertation is brought to you for free and open access by the Graduate School at Scholar Commons. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact [scholarcommons@usf.edu](mailto:scholarcommons@usf.edu).

Combining Natural Language Processing and Statistical Text Mining: A Study of Specialized  
Versus Common Languages

by

Jay Jarman

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
Department of Information Systems and Decision Sciences  
College of Business  
University of South Florida

Major Professor: Donald J. Berndt, Ph.D.  
Stephen L. Luther, Ph.D.  
Balaji Padmanabhan, Ph.D.  
Rosann W. Collins, Ph.D.

Date of Approval:  
September 30, 2011

Keywords: data mining, machine learning, computational linguistics, decision tree, rule mining

Copyright © 2011, Jay Jarman

## **Dedication**

I would like to dedicate this work to my wife, Cari Jarman. I don't know how anyone could do this without a support system. She is the most important person in my support system and has been there for me while I followed my dream. I am eternally grateful to her and will never be able to express my gratitude. Cari, I appreciate everything you do and I love you with all my heart.

## **Acknowledgments**

Funding for this research came from the Consortium for Healthcare Informatics Research, [HIR-09-002] HSR&D Center of Excellence, James A. Haley Veterans Hospital, Tampa, FL. This dissertation presents the findings and conclusions of the authors and does not necessarily represent the Department of Veterans Affairs (VA).

I would also like to acknowledge both the VA and specifically Steve Luther. I was given an opportunity to be on a research team and work with some of the best informaticians in this country and for that I will always be grateful.

I would like to acknowledge the help and collaboration of James McCart and Varol Kayhan.

Finally, I would like to acknowledge Don Berndt, my chair and mentor. When he came to me and asked if I'd like to do some mining research with the VA I had no idea that it would turn into a career and that I would gain so many new friends and collaborators. I'm truly appreciative of this opportunity.

## Table of Contents

List of Tables . . . . .	vii
List of Figures . . . . .	ix
Abstract . . . . .	xi
Chapter 1 Introduction . . . . .	1
1.1 Why the medical domain? . . . . .	2
1.2 Research Questions . . . . .	4
1.2.1 Research Question One - Specialized Language vs Common Language . . . . .	6
1.2.2 Research Question Two - Rule-based Classifiers . . . . .	7
Approach One - Classification Rules . . . . .	8
Approach Two - Decision Tree Induction (DTI) . . . . .	8
Chapter 2 Research Methodology . . . . .	10
2.1 Computational Linguistics . . . . .	10
2.1.1 Terminology . . . . .	12
2.1.2 Form . . . . .	13
Morphology . . . . .	13
Roots and Stems . . . . .	13
Tense, Plurality, and Gender . . . . .	14
Derivation and Compounding . . . . .	14
Affix . . . . .	14
Applications . . . . .	14
Syntax . . . . .	14
Syntactic Rules . . . . .	16
Applications . . . . .	17
Phonology . . . . .	17
2.1.3 Meaning . . . . .	18
Semantics . . . . .	18
Applications . . . . .	18
Pragmatics . . . . .	19
Applications . . . . .	19
2.2 Natural Language Processing (NLP) . . . . .	20
2.2.1 Vocabularies and Language Systems or Terminologies and Ontologies . . . . .	20
Controlled Vocabulary . . . . .	21
Ontology . . . . .	21
Unified Medical Language System (UMLS) . . . . .	24
Metathesaurus . . . . .	25

	Semantic Network . . . . .	25
	SPECIALIST Lexicon . . . . .	26
2.3	Computational Learning Theory . . . . .	26
	Probably Approximately Correct . . . . .	27
	Vapnik-Chervonenkis Theory . . . . .	27
	Algorithmic Learning Theory . . . . .	27
2.3.1	Machine Learning . . . . .	27
	Types of Machine Learning . . . . .	28
	Machine Learning Algorithms . . . . .	29
2.3.2	Information Theory . . . . .	29
	Entropy . . . . .	31
	Mutual Information . . . . .	31
	Information Gain . . . . .	31
	Information Gain Ratio . . . . .	32
2.3.3	Deductive vs Inductive Reasoning . . . . .	32
2.4	Statistical Text Mining (STM) . . . . .	33
2.4.1	Structure . . . . .	34
2.4.2	Identifying Patterns . . . . .	36
2.4.3	Evaluation and Interpretation . . . . .	36
	Confusion Matrix . . . . .	36
	Evaluation Metrics . . . . .	37
Chapter 3	Data . . . . .	39
3.1	Cohort . . . . .	40
3.2	Collection . . . . .	41
3.3	De-Identification and PHI . . . . .	41
3.4	Annotation . . . . .	42
3.5	Dataset . . . . .	43
Chapter 4	Medically Relevant Terminology vs Common Language Study . . . . .	45
4.1	Abstract . . . . .	45
4.2	Introduction . . . . .	45
4.3	Background . . . . .	47
4.4	Methods . . . . .	48
4.4.1	Data Set . . . . .	48
	Cohort . . . . .	48
	Collection . . . . .	49
	Annotation and Partitioning . . . . .	49
4.4.2	Process . . . . .	50
	NLP . . . . .	51
	Text Mining . . . . .	52
4.5	Results . . . . .	54
4.6	Discussion . . . . .	55
4.6.1	Error Analysis . . . . .	56
	Misspellings . . . . .	57
	Clinician Judgments . . . . .	57

	Fall History . . . . .	57
	Templates/Grammar . . . . .	58
4.6.2	Limitations . . . . .	59
4.6.3	Future Work . . . . .	59
4.6.4	Conclusion . . . . .	60
Chapter 5	Rule Mining Study . . . . .	61
5.1	Abstract . . . . .	61
5.2	Introduction . . . . .	61
5.3	Background . . . . .	62
5.4	Methods . . . . .	65
5.4.1	Data Set . . . . .	65
	Cohort . . . . .	65
	Collection . . . . .	66
	Annotation and Partitioning . . . . .	66
5.4.2	Process . . . . .	67
	1. Concept Extraction . . . . .	67
	2. Rule Mining . . . . .	67
	3. Contrast Sets . . . . .	68
	4. Rule Creation . . . . .	68
	5. Rule Evaluation . . . . .	68
5.5	Results . . . . .	69
5.6	Discussion . . . . .	71
5.6.1	Error Analysis . . . . .	72
5.6.2	Limitations and Future Work . . . . .	73
5.6.3	Conclusion . . . . .	74
Chapter 6	Decision Tree Induction Study . . . . .	75
6.1	Abstract . . . . .	75
6.2	Introduction . . . . .	75
6.3	Background . . . . .	77
6.3.1	Decision Tree Induction . . . . .	77
	Decision Selection . . . . .	79
6.3.2	Pruning . . . . .	79
6.3.3	Production Rules . . . . .	81
6.4	Methods . . . . .	81
6.4.1	Data Set . . . . .	81
	Cohort . . . . .	81
	Collection . . . . .	82
	Annotation and Partitioning . . . . .	82
6.4.2	Process . . . . .	83
	Step 1. Concept Extraction . . . . .	83
	Step 2. Statistical Text Mining (STM) . . . . .	85
	Step 3. DTI . . . . .	86
	Step 3a. DTI - Criterion . . . . .	86
	Step 3b. DTI - Reduce Terms . . . . .	86

Step 3c. DTI - Adjust Minimum Leaf Size . . . . .	87
Step 3d. DTI - Prune . . . . .	88
DTI - Production Rules . . . . .	90
6.4.3 Results . . . . .	90
6.5 Conclusions . . . . .	92
6.5.1 Error Analysis . . . . .	92
6.5.2 Future Work . . . . .	93
6.5.3 Concluding Remarks . . . . .	94
Chapter 7 Conclusion . . . . .	95
7.1 Research Question 1 . . . . .	95
7.2 Research Question 2 . . . . .	97
7.2.1 Approach 1: Classification Rule Mining . . . . .	98
7.2.2 Approach 2: Decision Tree Induction . . . . .	98
7.3 Other Contributions . . . . .	99
7.4 Future Work . . . . .	100
7.4.1 Ontology Development . . . . .	101
7.4.2 NLP . . . . .	101
WSD . . . . .	101
Acronym Detection . . . . .	102
Templates . . . . .	102
Cascading Errors . . . . .	103
7.5 Final Discussion . . . . .	103
References . . . . .	104
Appendix A Terms and Definitions . . . . .	116
Appendix B Tools . . . . .	117
B.1 Negation . . . . .	117
B.2 MetaMap/MMtx . . . . .	118
B.3 UIMA/cTAKES . . . . .	118
B.4 GATE . . . . .	119
Appendix C SAS Enterprise Miner . . . . .	120
C.1 Graphical User Interface . . . . .	120
C.1.1 1 - Project . . . . .	120
C.1.2 2 - Properties . . . . .	121
C.1.3 3 - Tab Bar . . . . .	121
C.1.4 4 - Diagram Workspace . . . . .	122
C.2 Pipeline . . . . .	122
C.2.1 TMPTRAINCUI . . . . .	123
C.3 Data Partition . . . . .	123
TMPTESTCUI . . . . .	124
C.3.1 Text Miner . . . . .	124
Parse . . . . .	125
Transform . . . . .	125



Cluster . . . . .	125
C.3.2 Regression . . . . .	125
C.3.3 Model Comparison . . . . .	125
C.3.4 SAS Code . . . . .	125
Appendix D General Architecture for Text Engineering (GATE) . . . . .	131
D.1 Graphical User Interface . . . . .	131
D.1.1 1 - Navigation . . . . .	131
D.1.2 2 - Loaded Processing Resources . . . . .	131
D.1.3 3 - Selected Processing Resources . . . . .	133
D.1.4 4 - Corpus . . . . .	133
D.1.5 5 - Parameters . . . . .	133
D.2 Pipeline Processing Resources . . . . .	133
D.2.1 Document Reset PR . . . . .	133
D.2.2 ANNIE English Tokeniser . . . . .	133
D.2.3 JAPE Transducer . . . . .	134
D.2.4 RegEx Sentence Splitter . . . . .	134
D.2.5 JAHVAOpenNLPOSTagger . . . . .	134
D.2.6 JAHVAOpenNLPChunker and JAHVAPhraseTagger . . . . .	134
D.2.7 JAHVAUMLSConceptFinder . . . . .	134
D.2.8 JAHVAConceptNegator . . . . .	135
D.2.9 JAHVAConceptWriter . . . . .	135
D.2.10 Nonconcept Counter (JAPE Tranducer) . . . . .	136
D.2.11 JAHVAAnnotationWriter . . . . .	136
D.2.12 JAHVAAnnotationCountWriter . . . . .	137
D.2.13 JAHVATokenFrequencyWriter . . . . .	137
Appendix E RapidMiner . . . . .	138
E.1 Graphical User Interface . . . . .	138
E.1.1 1 - Overview . . . . .	139
E.1.2 2 - Operators and Repository . . . . .	139
E.1.3 3 - Process . . . . .	139
E.1.4 4 - Parameters . . . . .	140
E.1.5 5 - Monitoring and Help . . . . .	140
E.2 DTI Process . . . . .	140
E.2.1 TrainCUIs . . . . .	141
E.2.2 Process Document . . . . .	141
E.2.3 Validation . . . . .	142
Term Weightings . . . . .	142
Numerical to Binomial . . . . .	143
Remember . . . . .	143
Decision Tree . . . . .	144
Apply Model (Term Weights) . . . . .	144
Numerical to Binomial . . . . .	145
Apply Model (Decision Tree) . . . . .	145
Performance . . . . .	145

Log . . . . .	146
E.2.4 Test CUIs . . . . .	146
E.2.5 Preprocess Document . . . . .	147
E.2.6 Recall . . . . .	147
E.2.7 Apply Model (Term Weightings) . . . . .	147
E.2.8 Numerical to Binomial . . . . .	148
E.2.9 Apply Model (DT) . . . . .	148
E.2.10 Performance . . . . .	148
Appendix F Regulation . . . . .	150
F.1 HIPAA . . . . .	150
F.2 HITECH . . . . .	151
F.3 Effect on Research . . . . .	151
Appendix G Mind Map . . . . .	153
Appendix H Information Systems Research Framework . . . . .	155
H.1 Design Science . . . . .	156

## List of Tables

Table 1	Computational Linguistic Terminology . . . . .	12
Table 2	Language Nomenclature . . . . .	20
Table 3	Vocabulary Relationships . . . . .	22
Table 4	Vocabulary Types . . . . .	23
Table 5	Ontology Definitions . . . . .	25
Table 6	Machine Learning Algorithms . . . . .	30
Table 7	Term-By-Document Matrix . . . . .	35
Table 8	Evaluation Metrics . . . . .	38
Table 9	Inter-rater Reliability Kappa Scores (%) . . . . .	43
Table 10	Intra-rater Reliability Kappa Scores (%) . . . . .	43
Table 11	Cohort description . . . . .	49
Table 12	Term-by-Document Matrix . . . . .	53
Table 13	Token Make-up of TEXT Data Set . . . . .	54
Table 14	Token Make-up of CMN and CUI Data Sets . . . . .	54
Table 15	Text mining and regression settings . . . . .	55
Table 16	LR Comparison (%) . . . . .	55
Table 17	Top 5 Predictive Terms from Text Mining TEXT and CUIS . . . . .	56
Table 18	Example Item Sets . . . . .	64
Table 19	Example Rules . . . . .	65
Table 20	Cohort description . . . . .	66
Table 21	Concepts in Itemsets . . . . .	69
Table 22	Highest Predictive Itemsets . . . . .	70
Table 23	Top Predictive Item Set Combinations on TRAIN (%) . . . . .	70
Table 24	Top Predictive Item Set Combinations on TEST (%) . . . . .	71
Table 25	Evaluation Comparison of Different Data Sets . . . . .	72
Table 26	CDSS Decision Tree Studies . . . . .	76

Table 27	C4.5 Requirements . . . . .	78
Table 28	Cohort description . . . . .	82
Table 29	Term-by-Document Matrix . . . . .	85
Table 30	Evaluation Metrics for Best Model All Terms (%) . . . . .	86
Table 31	Evaluation Metrics for Best Model Minimal Leaf (%) . . . . .	88
Table 32	Pruned Trees Eval Metrics (%) . . . . .	90
Table 33	Evaluation Comparison of Different Data Sets (%) . . . . .	91
Table 34	Results from all studies ( $*p < 0.001$ )( $**p < 0.05$ ) . . . . .	100
Table H.35	Dissertation IT Artifacts . . . . .	156

## List of Figures

Figure 1	Research Question Approaches . . . . .	5
Figure 2	Dissertation Map . . . . .	6
Figure 3	Controlled Vocabularies . . . . .	22
Figure 4	Model of Learning from Examples . . . . .	29
Figure 5	Inductive/Deductive Reasoning . . . . .	33
Figure 6	Confusion Matrix and Evaluation Metrics . . . . .	37
Figure 7	Episode of Care Hierarchy . . . . .	40
Figure 8	Breakdown of Terms . . . . .	46
Figure 9	Study Process . . . . .	50
Figure 10	CONSORT Flow Diagram . . . . .	51
Figure 11	Flow of Rule Mining/Classification Process . . . . .	63
Figure 12	Rule Accuracy . . . . .	71
Figure 13	Example Decision Tree . . . . .	78
Figure 14	Flow of Decision Tree/Classification Process . . . . .	84
Figure 15	Model Accuracy Based on Terms Used . . . . .	87
Figure 16	Model Accuracy Based on Minimal Leaf Size . . . . .	88
Figure 17	4 Level Decision Tree . . . . .	89
Figure 18	% Difference Between Decision Tree and STM Classifiers . . . . .	91
Figure 19	Breakdown of Terms . . . . .	96
Figure 20	Accuracy of Text, Medically Relevant Terms, and Common Language . . . . .	97
Figure 21	Graph of accuracy for all models . . . . .	99
Figure C.22	SAS Enterprise Miner Graphical User Interface . . . . .	121
Figure C.23	Sample Tab . . . . .	122
Figure C.24	Model Tab . . . . .	122
Figure C.25	Text Miner Tab . . . . .	122
Figure C.26	Training Data Set Properties . . . . .	123

Figure C.27	TRAINING Data Partition Properties . . . . .	124
Figure C.28	TEST Data Partition Properties . . . . .	124
Figure C.29	Text Miner Properties . . . . .	126
Figure C.30	Regression Properties . . . . .	127
Figure C.31	SAS Code Component . . . . .	128
Figure C.32	Actual Classification Variable . . . . .	129
Figure C.33	Predicted Classification Variable . . . . .	130
Figure C.34	Data Set Variables Properties . . . . .	130
Figure D.35	GATE Graphical User Interface . . . . .	132
Figure D.36	JAHVA UMLS Concept Finder Parameters . . . . .	135
Figure D.37	JAHVA Concept Negation Parameters . . . . .	135
Figure D.38	JAHVA Concept Writer Parameters . . . . .	136
Figure D.39	JAHVA Annotation Writer Parameters . . . . .	136
Figure D.40	JAHVA Annotation Count Writer Parameters . . . . .	137
Figure D.41	JAHVA Token Frequency Writer Parameters . . . . .	137
Figure E.42	RapidMiner Graphical User Interface . . . . .	138
Figure E.43	Main DTI Process . . . . .	140
Figure E.44	Process Document Parameters . . . . .	141
Figure E.45	Validation Operator . . . . .	142
Figure E.46	Term Weights Parameters . . . . .	143
Figure E.47	Decision Tree Parameters . . . . .	144
Figure G.48	Dissertation Mind Map . . . . .	154
Figure H.49	Information Systems Research Framework . . . . .	155

## ABSTRACT

This dissertation focuses on developing and evaluating hybrid approaches for analyzing free-form text in the medical domain. This research draws on natural language processing (NLP) techniques that are used to parse and extract concepts based on a controlled vocabulary. Once important concepts are extracted, additional machine learning algorithms, such as association rule mining and decision tree induction, are used to discover classification rules for specific targets. This multi-stage pipeline approach is contrasted with traditional statistical text mining (STM) methods based on term counts and term-by-document frequencies. The aim is to create effective text analytic processes by adapting and combining individual methods. The methods are evaluated on an extensive set of real clinical notes annotated by experts to provide benchmark results.

There are two main research questions for this dissertation. First, can information (specialized language) be extracted from clinical progress notes that will represent the notes without loss of predictive information? Secondly, can classifiers be built for clinical progress notes that are represented by specialized language? Three experiments were conducted to answer these questions by investigating some specific challenges with regard to extracting information from the unstructured clinical notes and classifying documents that are so important in the medical domain.

The first experiment addresses the first research question by focusing on whether relevant patterns within clinical notes reside more in the highly technical medically-relevant terminology or in the passages expressed by common language. The results from this experiment informed the subsequent experiments. It also shows that predictive patterns are preserved by preprocessing text documents with a grammatical NLP system that separates specialized language from common language and it is an acceptable method of data reduction for the purpose of STM.

Experiments two and three address the second research question. Experiment two focuses on applying rule-mining techniques to the output of the information extraction effort from experi-

ment one, with the ultimate goal of creating rule-based classifiers. There are several contributions of this experiment. First, it uses a novel approach to create classification rules from specialized language and to build a classifier. The data is split by classification and then rules are generated. Secondly, several toolkits were assembled to create the automated process by which the rules were created. Third, this automated process created interpretable rules and finally, the resulting model provided good accuracy. The resulting performance was slightly lower than from the classifier from experiment one but had the benefit of having interpretable rules.

Experiment three focuses on using decision tree induction (DTI) for a rule discovery approach to classification, which also addresses research question three. DTI is another rule centric method for creating a classifier. The contributions of this experiment are that DTI can be used to create an accurate and interpretable classifier using specialized language. Additionally, the resulting rule sets are simple and easily interpretable, as well as created using a highly automated process.



## **Chapter 1**

### **Introduction**

A central challenge in knowledge discovery is handling both structured and unstructured data. Structured data are the typical elements one might find stored in a database, carefully selected attributes and their associated data types such as numbers, dates, or limited-length character strings. These data are typically part of planned data collection efforts with associated data quality concerns expressed through integrity constraints and business rules. Unstructured data are the inherently ambiguous free-form passages of text or semi-structured lists and tables of data that are natural human language artifacts. With the advent of the Internet and World Wide Web protocols, there are vast amounts of unstructured data available on-line with no end in sight. Of course, this immense collection of information lacks any central planning, data quality standards, or easily interpreted meta-data, making its use a challenging endeavor. Data analytic techniques including traditional statistics, machine learning, and data mining have been successfully applied to structured data collections. Modern technology has made it possible not only to search for but to also collect and store massive amounts of unstructured textual documents. There is a need to be able to analyze and draw conclusions concerning these data. Text mining, while not as mature as data mining, is being applied to many disciplines in the hope of uncovering unexpected and perhaps counter-intuitive new knowledge.

A series of classic studies in medicine provide motivation for further research in this area. Swanson and Smallheiser pursued a stream of research that looked at connecting important, but bibliographically unrelated topics, by uncovering linkages through intermediate topics. Swanson's first effort in this regard connected fish oil with the treatment of Raynaud's syndrome (Swanson, 1986a). While this first research was somewhat serendipitous and largely conducted by hand, it provided promising results and a framework for a more automated discovery process. His original discovery was prompted by reading two unrelated literature collections, leading to the hypothesis that fish oil might be useful in the treatment of Raynaud's syndrome. Patients suffering from

Raynaud's disease have problems related to blood flow in the extremities. Swanson then found the literature on fish oil and its beneficial effects on blood viscosity, making the connection and formulating a hypothesis. His second discovery followed a similar logical process, connecting magnesium with migraine headaches (Swanson, 1988). These discoveries led to work on more automated hypothesis generation approaches based on text mining the medical literature (Swanson, 1987, 1989, 1990; Smallheiser and Swanson, 2007a, b, 1998).

This dissertation focuses on developing and evaluating hybrid approaches for analyzing free-form text in the medical domain. This research draws on natural language processing (NLP) techniques that are used to parse and extract concepts based on a controlled vocabulary. Once important concepts are extracted, additional machine learning algorithms, such as association rule mining and decision tree induction, are used to discover classification rules for specific targets. This multi-stage pipeline approach is contrasted with traditional statistical text mining (STM) methods based on term counts and term-by-document frequencies. The goal is to create effective text analytic processes by adapting and combining individual methods. The methods are evaluated on an extensive set of real clinical notes annotated by experts to provide benchmark results.

Traditional "bag of words" methods create classifiers with acceptable performance and rules can be constructed using these methods as well. So why combine NLP with STM? Sizes of databases are constantly growing. Currently, there are over one billion progress notes available in VA databases. By combining these techniques, they can be used as a method of data reduction. Initial results from this research show it is possible to reduce the amount of data used to create the classifiers by as much as 75%. The terms used to create the classifiers can be reduced by as much as 50%. Rules created from specialized language are more likely to be interesting and more easily interpretable because much of the noise is removed.

## **1.1 Why the medical domain?**

Imagine your 85 year old grandfather goes to the hospital for a routine out patient procedure. Grandpa has fallen a couple times in the past year and received a few bruises but he's embarrassed so he does not mention this to the admissions nurse. His procedure is completed without any complications and he is in recovery waiting for you to come pick him up. He decides he needs to use the restroom and gets out of his bed without requesting assistance and falls on the floor and breaks

his hip along with some other injuries. Grandpa now becomes an inpatient and his quality of life is altered for his remaining years. There are two issues with this scenario. First, and most importantly, Grandpa now has a very painful and lengthy recovery ahead of him. Also, the older a person is when they experience a fall-related injury, the less likely they are to have a meaningful recovery. Secondly, had everything gone as expected, Medicare and his coinsurance would have covered the cost of his procedure. Now, because the injury happened inside the hospital, the hospital is financially responsible for his medical bills for the treatment of his fall-related injuries. Had the hospital staff known that your Grandfather was at risk for a fall, simple precautions could have been in place to prevent the fall.

Falls are an important health care issue especially among aging veterans. A history of a previous fall is one of the most important clinical indicators that identifies an elderly patient as high risk for additional falls and targets them for fall prevention programs (Ganz et al., 2007). However, information about fall-related injuries (FRIs) in administrative databases has been found to be significantly under-coded, thereby limiting a clinicians access to information about a history of falls (Luther et al., 2005).

With the evolution of the electronic health record (EHR), ever increasing amounts of structured and unstructured data are being made available for research purposes. An EHR is made up of both structured and unstructured data. More and more of this information is aggregated at the point of patient contact by smart devices or captured as clinicians use mobile devices to create clinical or pharmaceutical orders. Data such as vitals, demographic data, lab results and medical codes are stored in structured data fields. In addition to the more structured data, the art of careful observation and documentation has a long tradition in medicine. Clinicians are trained to use the SOAP (Subjective, Objective, Assess, Plan) system for documenting the more subtle on-going issues that are not easily captured through formal medical coding systems or the EHR structured data fields. Much of the information stored in unstructured data is information provided by patients to clinicians. In other words, when a patient expresses how they feel or where a specific pain has been occurring, this type of data is mainly stored in unstructured data. The most obvious artifact of this documentation process is the large number of unstructured clinical notes that are also an important component of the EHR. Specifically, the Veterans Health Administration, the largest health care system in the US, has imported all veterans' health records into its EHR system, which

at this time exceed one billion progress notes.

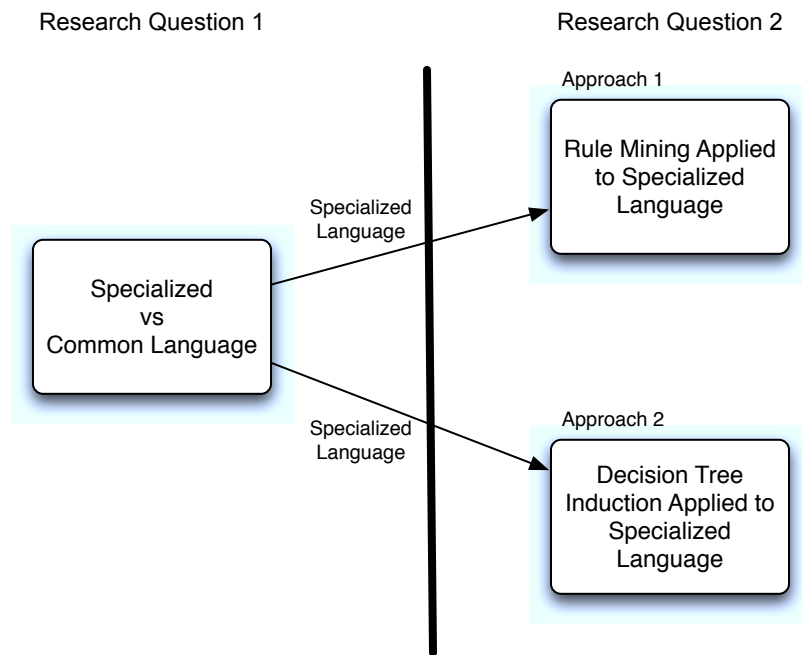
Components necessary for this research are the controlled vocabularies of medical terms. Creating such vocabularies is a very labor intensive and time consuming activity, and even though controlled vocabularies have been created for numerous fields of research in many different disciplines, the development of controlled vocabularies in medicine are especially well-developed and well-funded. Foremost is the Unified Medical Language System (UMLS) Metathesaurus created by the National Library of Medicine (NLM). The Metathesaurus is a vocabulary database containing biomedical and health related concepts as well as relationships among the concepts. It is comprised of more than 100 source vocabularies, each representing hundreds, thousands, and sometimes millions of health related terms. Because well-developed controlled vocabularies are a necessary component of this research stream, this makes medicine an attractive area on which to perform data/text mining research.

## **1.2 Research Questions**

There are two main research questions this dissertation addresses. The first, can information (specialized language) be extracted from clinical progress notes that will represent the notes without loss of predictive information? This is addressed by the experiment in Chapter 4. With this question emerge some interesting issues to be addressed. Specifically, is the locus of predictive information in specialized language, common language, or a combination and does the quality (i.e., coverage) of the controlled vocabulary affect the specialized language? In other words, is the quality of a controlled vocabulary covering a mature area better than one covering an emerging area? The second research question is, can classifiers be built for clinical progress notes that are represented by specialized language? Three criteria were used to decide on the approaches to be used to answer this question. First, the approaches had to allow for automated rule extraction. Secondly, the resulting classifier had to be interpretable and finally, the classifier needed to be accurate. Two approaches allowed this question to be answered while still adhering to these criteria. The first approach uses modified association rule techniques to create classification rules. This approach provided a classifier that supplied rules and was interpretable, however, even though it had acceptable accuracy we felt another approach might provide a more accurate model. The second approach uses a decision tree induction (DTI) process to create a decision tree and production rules.

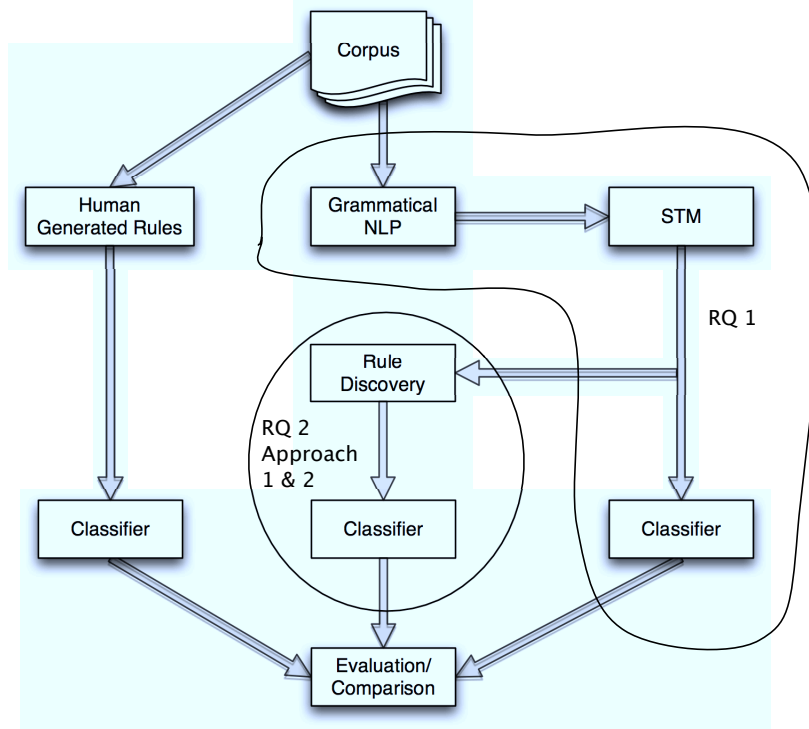
This approach created rules and was interpretable just as the first method, however, the accuracy achieved with this classifier was much better.

Figure 1 shows the two research questions as well as the experiments used to answer them. The results from the first research question are used to inform the next two experiments used to answer the second research question.



**Figure 1.:** Research Question Approaches

Figure 2 provides an overview of the research landscape and interrelationships between the three experiments in this dissertation. In order to classify documents in health care, decision-making rules are often handcrafted by clinical experts (Hayes and Weinstein, 1990; Hayes et al., 1990). The left side of figure 2 shows this method of classification. This dissertation concentrates on more automated, machine learning methods of classification. Research question one uses a hybrid methodology utilizing grammatical NLP in conjunction with machine learning. The results of this are used to inform the two approaches used to answer research question two. The next sections describe each of the research questions in more detail.



**Figure 2.:** Dissertation Map

### 1.2.1 Research Question One - Specialized Language vs Common Language

Medical progress notes are comprised of specialized language (medically-relevant terminology) and common language. The first experiment focuses on whether relevant patterns within clinical notes reside more in the highly technical medically-relevant terminology or in the passages expressed by common language. Of course, it is likely that a combination of both medically-relevant and common language is used to express many thoughts regarding the progress of care. Yet, there is still an interesting question to be resolved in determining the locus of clinically relevant patterns. There is a natural experiment in separating the two languages and assessing the predictive power of the language subsets. This experiment addressed the first main goal of this dissertation and the results from this experiment informed the subsequent approaches that address research question two. The research question this experiment sought to answer is *using statistical text mining (STM) for the purpose of classification, which set of terms from progress notes will provide better accuracy: (1) all terms (2) only medically-relevant terms (MRTs) or (3) terms based on common language?* The contribution of this experiment is that it shows that predictive patterns

are preserved by preprocessing text documents with a grammatical NLP system that separates specialized language from common language and it is an acceptable method of data reduction for the purpose of STM.

This research described a way to reduce EHR text notes to only medically-relevant terms by using a grammatical NLP system to extract medical concepts from progress notes. These medically-relevant terms were used in a statistical text mining process in order to classify the notes as either FALL or nFALL. Evaluation metrics were collected and all were slightly better when compared to those for text mining the raw progress notes. This shows that important patterns are not lost by removing common language, effectively reducing the information to be processed and in turn maximizing efficiency. Since no loss occurred, this extracted data can serve as a foundation for other data mining research such as targeted information extraction and even clinical discovery.

### **1.2.2 Research Question Two - Rule-based Classifiers**

An accurate classifier was created in the first experiment to answer research question one. Since logistic regression was used to create the model, it is difficult for a human to interpret in spite of its accuracy. For research question two, the goal was not only to see if classifiers can be built for clinical progress notes represented by specialized language, but also can those classifiers be rule based in order to improve their human interpretability. Three criteria were used to select the machine learning methodology for creating the classifiers. The first is criterion is automated. Human can create rules but this is time consuming and expensive. One goal is to create an automated process for generating rules. The second criterion is interpretable. An accurate classifier can be created using many different machine learning methodologies, some of which, result in models that are not easily interpretable by humans. We wanted the rules generated for the classifier to be interpretable. The third criterion is accurate. This should be self explanatory because an inaccurate model is obviously useless. We also wanted this model to be more accurate than simply assigning a single classification to every case. Two approaches were used to answer this research question taking the criteria into consideration. The main contribution of the experiments performed to answer research question two is that predictive patterns are preserved by preprocessing text documents with a grammatical NLP system that separates specialized language from common language.

### **Approach One - Classification Rules**

The first approach focuses on applying rule-mining techniques to the output of the information extraction effort resulting from the first research question, with the ultimate goal of creating rule-based classifiers. Association rule mining is one of the most widely used data mining techniques. In particular, market basket analysis relies on using rule mining on the often very large collections of shopping basket data collected at point-of-sale terminals. Most of the time, rule mining is applied to structured data. In addition to research question two, this approach addresses a couple of research questions specific to it. First, *Can traditional association rule mining procedures be used to generate rules (frequent item sets) using specialized language?* Secondly, *can these rules be used with acceptable accuracy for classification purposes?* There are several contributions from this approach. First, it uses a novel approach to create classification rules from specialized language and to build a classifier. The data is split by classification and then rules are generated. Secondly, several toolkits were assembled to create the automated process by which the rules were created. Third, this automated process created interpretable rules and finally, the resulting model provided good accuracy.

There are other methods for creating a classifier that might have better performance such as support vector machines (SVMs), which are especially useful where large variable sets exist. A dilemma arises using other methods in that the resultant predictive model cannot always be interpreted by a human. One of the benefits of creating a classifier with rules is that the rules themselves are interpretable. This approach used the data set from the first research question experiment based on the extracted medically-relevant terms. A rule-based classifier was used to classify this data set. The resulting performance was slightly lower than from the classifier from the first experiment but had the benefit of having interpretable rules. Another dilemma that exists within association rule mining is extracting meaningful and interesting rules. By mining using only medically-relevant terms, most terms that create uninteresting or not meaningful rules are eliminated before the mining process begins.

### **Approach Two - Decision Tree Induction (DTI)**

Approach two focuses on using DTI for a rule discovery approach to classification. DTI is another rule centric method for creating a classifier. It is a method whereby the process iteratively



selects a set of attributes that most effectively splits the sample data into subsets to create a classifier. The tree is made of nodes which are either leaf nodes indicating the classification or decision nodes that specify a test to be carried out on a single attribute value. Of course, these pathways through the decision nodes are basically rules and a decision tree classifier is human interpretable. The research question this approach seeks to answer is *can decision tree induction be used to classify clinical progress notes represented by specialized language?* There are several contributions, which are the result of this approach. First, DTI can be used to create an accurate and interpretable classifier using specialized language. Secondly, the resulting rule sets are simple and easily interpretable. Third, an automated process is used to create rule sets.

There was no significance in the difference in the performance of the STM classifier from the first experiment versus the DTI model from this approach. However, the resultant classifier from the DTI process was much more interpretable, where as the model from the STM process would be nearly impossible for a human to interpret.

## **Chapter 2**

### **Research Methodology**

This chapter is organized in four sections, two are theoretically based and the other two are application based. The first section is a theoretical discussion of computational linguistics and how these theories lay the ground work for NLP. The second section is a discussion on how computational linguistics are applied, specifically, an explanation of NLP and how it is used in this dissertation. The third section is a theoretical discussion of computational learning theory and information theory and how they lay the ground work for STM. The fourth section is a discussion of the STM techniques used to complete the three studies.

#### **2.1 Computational Linguistics**

The Association for Computational Linguistics defines computational linguistics as “...the scientific study of language from a computational perspective.” Computational linguistics is based on linguistic theories and how mathematics can be applied. Computational linguists are interested in providing computational models of various kinds of linguistic phenomena. There are two schools of thought where processing language is concerned. One is statistically based where probabilities are associated with possible meanings and the most probable outcome is the correct one. The second is based on linguistics and the only way to gain understanding of text is to learn the rules of the language. STM is based on the first approach. Computational linguistics is based on the second approach.

When computers began to be thought of as useful for interpreting text there were two groups of scientists interested in this: cryptographers and linguists. Cryptographers were generally statisticians and saw the computer as being used for translation. Probabilities are associated with possible meanings and the most probable outcome is the correct one. Warren Weaver, a pioneer in machine translation, published along with Shannon landmark work on communications (Shan-

non and Weaver, 1949). Linguists, not being mathematicians, saw the computer as being used to implement linguistic theories. Computational linguists develop formal models simulating aspects of the human language faculty and implement them as computer programs. The creation of CYK (Cocke, Younger, Kasami) is generally thought of as the birth of computational linguistics (Cocke, 1969; Kasami, 1965; Younger, 1967). Chomsky, one of the most famous linguists, published this sentence in *Syntactic Structures*, “Colorless green ideas sleep furiously” Chomsky (1957). It is grammatically correct but semantically nonsensical. He used it to show the inadequacy of probabilistic models of grammar and the need for more structured models.

Real world applications based on computational linguistics include applications in natural language understanding such as machine translation, spoken dialog systems, and question answering as well as NLP which include information retrieval (IR), information extraction (IE), and concept/term extraction. For the purposes of this dissertation, this section will only concentrate on computational linguistics from an NLP perspective. Because of the use of body language and other non-verbal cues, people are able to relax the rules of language and still effectively communicate verbally. Written communications can many times be written using relaxed language rules but without the verbal cues, effective communication can be hampered. This creates a need for computational linguistics because not all written utterances are grammatically correct. IE, translation, and grammar checking also create a need for computational linguistics.

Linguistics can be broken down into two categories: form and meaning.

- Form
  - Morphology - Study of internal structures of words and how they can be modified
  - Syntax - Study of how words combine to form grammatical sentences
  - Phonology - Study of sounds (signs) as discrete, abstract elements in the speaker’s mind that distinguish meaning
  
- Meaning
  - Semantics - Study of the meaning of words (lexical semantics) and fixed word combinations (phraseology) and how these combine to form the meanings of sentences
  - Pragmatics - Study of how utterances are used in communicative acts, and role played by context and nonlinguistic knowledge in the transmission of meaning

The technologies used in this dissertation are mainly concerned with form, however, meaning has a secondary consideration. This section discusses the theoretical aspects of computational linguistics. The following subsections will provide definitions for terminology used within the section and an explanation of the two categories. Unless otherwise stated, the information provided in this section applies to the American English language. Most of the basics of computation linguistics applies to languages in general. However, this dissertation is concerned with American English, therefore, that is the concentration of this section.

### 2.1.1 Terminology

This sections provides definitions of terms, ideas, and theories used throughout this chapter.

**Table 1:** Computational Linguistic Terminology

<b>Terminology</b>	<b>Explanation</b>
Affix	A morpheme that is attached to a word stem to form a new word.
Morpheme	The smallest unit in a language, which can be assigned a meaning.
NP	Noun Phrase - A phrase containing a noun and other modifying words such as adjectives, adverbs, and other nouns.
VP	Verb Phrase - A phrase containing a verb and modifying adverbs.
Context Free Grammar (CFG)	A formal grammar in which every production rule is of the form $V \rightarrow w$ where $V$ is a single nonterminal symbol and $w$ is a string of terminals and or nonterminals. Used in linguistics to describe the structure of sentences and words in natural language.
Bottom-up Parsing	A strategy for analyzing unknown information that attempts to identify the most fundamental units first, and then to infer higher-order structures from them.
CYK	Cocke, Younger, and Kasami. Sometimes referred to as CKY, is a parsing algorithm for context free grammars that uses bottom-up parsing and dynamic programming.

## 2.1.2 Form

### Morphology

Morphology deals with strings that make up words and these strings are combined to make sentences. It also attempts to uncover the rules that govern the creation of words (Trost, 2003). Regardless the language, the smallest unit in a language, which can be assigned a meaning is a morpheme. Each language contains hundreds of thousands or even millions of words. In contrast, languages contain some 10,000 morphemes, an order of magnitude smaller than the number of words. There are semantic morphemes called roots such as *dog*, *red*, and *take*. There are also abstract morphemes that imply plurality and tense such as *s*. A free morpheme forms a word on its own such as *dog*. Bound morphemes occur only with other morphemes such as *s* in *dogs*. Each language has grammatical rules that control how these morphemes are connected to create words.

Much of the parsing technology in an NLP pipeline has its background in morphology. The parser finds tokens, then words. The part of speech for each word is computed and used to create phrases. This being done correctly starts with determining word boundaries correctly. If the words cannot be determined correctly, all subsequent steps will also be incorrect. These technologies were used in the first experiment to extract medically-relevant terms from the progress notes. More detail on this can be found in Chapter 4 and in Appendix D. Vocabularies and ontologies as well have underpinnings in morphology. The basic unit for both is words. From there words are combined to create terms. These are explained in more detail in a subsequent section. The following sections discuss the core theoretical foundations that underpin critical components of NLP pipelines.

**Roots and Stems** The root of a word is the base morpheme for the word with no affixes. For example, the root of *degrade* is the morpheme *grade* with the affix morpheme *de* removed. A similar concept is stemming. A stem is the base form of a word. One example is the stem for *gave*, *gives*, and *giving* is *give*. *Give* is the base form for each of those words but not the root. Another example is the stem for *degrades* and *degrading* is *degrade* while the root for *degrades* is *grade*. A stemming NLP component was used in the NLP pipeline in experiment one, which focused on assessing the predicative power of specialized versus common language. More detail on this can be found in Chapter 4.

**Tense, Plurality, and Gender** Some languages mark words with tense, plurality, and gender while others do not (Trost, 2003). For example, Latin based languages mark words with gender such as in Italian pomodoro (tomato - masculine), and cipolla (onion - feminine). On the other hand, Japanese does not mark plurality on nouns.

**Derivation and Compounding** New words can be created using derivation and compounding (Trost, 2003). By using derivation, affix morphemes are added to stems to create a new word that is also a different part of speech from the original word. For example, the noun *hospitalize* can be derived from *hospital* by adding the affix morpheme *ize*. By compounding, a new word is created by joining two base morphemes. For example, by compounding *door* and *bell*, a new word, *doorbell* is created.

**Affix** There are different types of affix morphemes that can be joined to stem morphemes. A prefix affix is joined to the beginning of a stem (Trost, 2003). For example, by adding the prefix *un* to *common* creates *uncommon*. A suffix affix is a morpheme joined to the end of a stem. A common suffix is to add an *s* to a stem to make it plural.

There are also rules for affixes. Multiple suffixes can be added to stems to create new words, however, there are rules that control the order for affixes. For instance, the suffixes *ize* and *ation* can be added to *hospital* to create *hospitalization*. Those same suffixes cannot be added in a different order. The string *hospitalationize* is not a word. *ize* must be connected to a noun and creates a verb. *ation* must be connected to verbs and creates nouns.

**Applications** There are several practical applications of the morphological theories in NLP. Word processors apply morphology theories in hyphenation, spell checking, and grammar checking. Grammar rules are used in NLP parses to determine parts of speech. Stemming is another NLP component used to convert words into their stem forms.

## **Syntax**

Syntax is the study of the rules used to construct phrases and sentences in natural languages (Kaplan, 2003). Just as there are rules that govern how words are made by combining letters, there are grammatical rules that govern how words are combined to make phrases and sentences. In order

for a grammatical NLP parser to make sense of a phrase, it must first be parsed into its grammatical components. There are several syntactic theories that are behind this technology. A major goal of syntactic theories is to define a notation that can be used to create rules that define a grammar. Theories that were once popular are now considered to be out dated but they were instrumental in the development of newer, more explanatory theories. A few syntactic theories are explained below. What these theories have in common is that they map a sentence into a tree structure. The root is the sentence, next are the phrases such as NPs and VPs, then the parts of speech, and finally the words that comprise the sentence. How these theories differ is how the tree is built and what can further be done with the tree.

- In 1957, Chomsky published *Syntactic Structures* where he laid out the original generative theory, Transformational Grammar (Chomsky, 1957). In this theory, a sentence can be represented by a deep structure and a surface structure. The deep structure is the starting point and maps a sentence in a tree of phrases representing the sentence. This tree goes through a series of transformations where one tree is the input for a subsequent transformation until the final tree, the surface structure, is created. The sentences of the language are the strings that appear at the bottom of the surface structures. One example is to transform an active declarative sentence into the passive equivalent. It is possible to go from a deep structure to a surface structure but it is next to impossible go from a surface structure to a deep structure.
- Generalized Phrase Structure Grammar (GPSG) - Developed in the late 70s by Gerald Gazdar. It is a framework for describing the syntax and semantics of natural languages (Gazdar et al., 1985). One of its goals is to show that syntax of natural language can be described by CFG, however, Gazdar has since argued that this is not true. Much of the innovations from GPSG were later incorporated into Head Driven Phrase Structure.
- Head-Driven Phrase Structure Grammar (HPSG) - Developed by Carl Pollard and Ivan Sag. It is used in NLP because it is organized in a modular way. It uses the concept of a sign which has a type hierarchy and its features describe its properties (Phonological, Syntactic, and Semantic) (Pollard and Sag, 1988). Grammatical rules are expressed through the constraints signs place on each other.
- Lexical Functional Grammar (LFG) - Initiated by Joan Bresnan and Ronald Kaplan in the 70s.

LFG views language as being made up of structures that are grammatical functions and syntax (Bresnan, 1976). A sentence is broken into phrases such as verb phrases (VP) and noun phrases (NP) and then further broken into parts of speech until finally are the words of the sentence.

**Syntactic Rules** In English, words comprise a sentence if there is a noun and verb, as in:

*Bob visited.*

One grammatical rule from this could be:

A sentence can consist of noun-verb sequence.

Take this one step further and examine the sentence:

*Bob visited Mary*

Now another rule can be created.

A sentence can consist of a noun-verb-noun sequence.

So far these have been simple sentences. Do the rules hold when the sentence becomes more complex? Take for example the sentence:

*The man from city visited old woman.*

There are more words in this sentence than previous examples but the previous rule still applies: *man visited woman*. More rules need to be created to compensate for the additional words. The concept of a *phrase* becomes important now. A phrase is a contiguous sequence of related words. The first phrase in the sentence above is *The man from the city*. The second phrase is *old woman*. Now an additional rule can be created that describes this sentence.

A sentence can consist of NP - VP - NP.

In addition, rules can be made that qualify what comprises a NP or a VP. Syntactic rules describe how sentences are constructed but are not concerned with meaning. In other words, a sentence can be grammatically correct and not make any sense. A popular game, Mad Libs, allows



players to create nonsensical sentences that are syntactically correct. Leonard Stern and Roger Price created the game and published the first book of the game in 1958. Players are given sentences with words missing. They are told to give a word based only on the part of speech. Those words are entered along with the original words provided and a grammatically correct sentence is formed, however, this sentence most likely is nonsensical because the player supplying the words is unaware of the other words supplied by the game. The game is good for a laugh but it is also a good exercise in syntax by looking at how sentences are formed by parts of speech and phrases.

**Applications** NLP parsers are grounded in syntactic theories which enable them to correctly parse text into its correct parts of speech. This is one of the beginning steps in most NLP processes. For example, to perform a lookup, the application must have the correct word to lookup and its part of speech. The pipeline used in experiment one parses text eventually into words and determines whether each word is a medically-relevant term or not. The details on how this is accomplished are in Chapter 4.

## **Phonology**

This dissertation concentrates only on written language due to the fact that the progress notes are written. However, much of this same background applies to spoken language as well. There are language recognition systems that are used to convert natural spoken language into a form that a computer can understand then NLP is performed on that. The IBM Watson computer application is one example of that (Ferrucci et al., 2010). Phonology is briefly mentioned here even though no spoken technology is used in this dissertation.

Phonology is the study of sounds as they are used in language. Computational phonology is the study of computational techniques used to represent and process phonological information (Bird, 2003). Humans have a vocal apparatus capable of creating an infinite variety of sounds. Human language takes this infinite number of sounds and reduces them into a sound system consisting of a few dozen categories of sounds called phonemes.

It should be noted that phonemes do not match one to one to the number of letters in an alphabet. The letter *t* has multiple phonemes associated with it and many vowel phonemes are associated with multiple letters such as *ou*. The number of phonemes in the English language also

varies from dictionary to dictionary. For instance, the *American Heritage Dictionary* uses 25 consonant and 18 vowel phonemes on the other hand, the *Longman Pronunciation Dictionary* uses 24 consonants and 23 vowels (Ladefoged, 2001).

### 2.1.3 Meaning

#### Semantics

Semantics is the study of the meaning of language. Computational semantics is the area of computational linguistics that uses a computational approach to natural language to acquire the meaning (Lappin, 2003). Semantics is concerned with how the meanings of individual words contribute to the meaning of the phrase or sentence containing those words.

There are two approaches to computational semantics. First, is static where each sentence meaning is self-contained. Secondly, is dynamic where the semantic interpretation of a sentence is dependent on previous sentences in the discourse. In using dynamic approaches, a dialog record is constructed that allows the meanings of previous sentences to be used in the interpretation of subsequent sentences.

Syntax and semantics are tied together. Once a sentence has been analyzed and parsed into its syntactic components, computational semantics computes systematically the sentence's meaning from the words (Lappin, 2003). In many models, a syntactic structure is two dimensional in that it contains both syntactic and semantic information. The Parallel Correspondence Model (PCM) is an approach that encodes both syntactic and semantic information. There are several theories that utilize this approach. GPSG, HPSG, and LFG discussed in the previous section use the PCM model. In Fernandez et al., they propose a HPSG approach that applies semantic meaning to phrases and sentences based on questions previously listed in the discourse (Fernandez et al., 2011).

**Applications** The biggest application of semantics within NLP is Word Sense Disambiguation (WSD). Homonyms exist in the English language, which are words that have different meanings but are spelled and pronounced the same. Take for instance, the word *cold*. Cold can be used to describe how one feels, the temperature outside, an illness, and many other meanings. NLP parsers used to extract meanings from words are grounded in semantic theories. The pipeline used in ex-

periment one does not have a WSD module. At the time, a WSD module for medical terminology could not be located. Currently, this is a growing area of research within medical informatics.

## **Pragmatics**

Closely related to semantics is pragmatics. Pragmatics is the study of the meaning of linguistic messages in terms of context (Leech and Weisser, 2003). Mostly, pragmatics is related to speech rather than the written word, however, it is discussed here because it has some practical applications in the field of NLP.

Illocutionary Acts, also known as Speech Acts, is one of the philosophical foundations of pragmatics (Austin, 1962; Searle, 1969). A speech act conveys meaning through action. Up until this point most approaches to language had been treated as a statement that can be treated as either true or false (Leech and Weisser, 2003). Statements such as *I promise* are a verbal action rather than stating something that is true or false. Speech acts characterizes verbal actions as one of three categories of acts: locution, illocution, and perlocution.

Locution is the actual utterance and its apparent meaning (Austin, 1962). The utterance, *Don't run in front of cars* in its locutionary meaning is an utterance stating not to run in front of cars. It does not convey whether the cars are moving or parked or where the cars are located. However, given the same utterance, from a speaker to a hearer next to a busy highway and the meaning is taken not to run out into the traffic in front of the moving cars. This is the illocutionary meaning. Now given the same statement in the same situation and it prevents the hearer of the utterance from running out in front of cars and getting hurt, that is the perlocutionary meaning. The perlocutionary meaning is dependent on two things: whether the hearer understands the utterance and whether the hearer is willing to comply with the utterance.

**Applications** There are many applications for pragmatics in speech recognition and yet not as many with written text. One of the areas in NLP where pragmatic theories are applied are the use of pronouns. Pronouns are used extensively in writing and many utterances contain multiple pronouns, some referring to people and others referring to objects. The challenge is to determine who or what the noun is that the pronoun is referring. Pragmatic theories underpin the technology used by NLP processes to determine these references.

## 2.2 Natural Language Processing (NLP)

NLP is concerned with the interaction between natural language, human language as opposed to computer language, and computers. This section discusses NLP and associated technologies that are grounded in computational linguistic theories.

### 2.2.1 Vocabularies and Language Systems or Terminologies and Ontologies

The process of classifying information is as old as language itself. There is a nomenclature used to represent language and the different tools used to classify language. A great deal of disagreement exists in the literature concerning the meanings of some of these terms. What follows is the nomenclature that will be used throughout this dissertation as well as the definitions. Since there is disagreement, this should alleviate confusion caused by any preconceived definitions. Table 2 lists some common terms used throughout this dissertation.

**Table 2:** Language Nomenclature

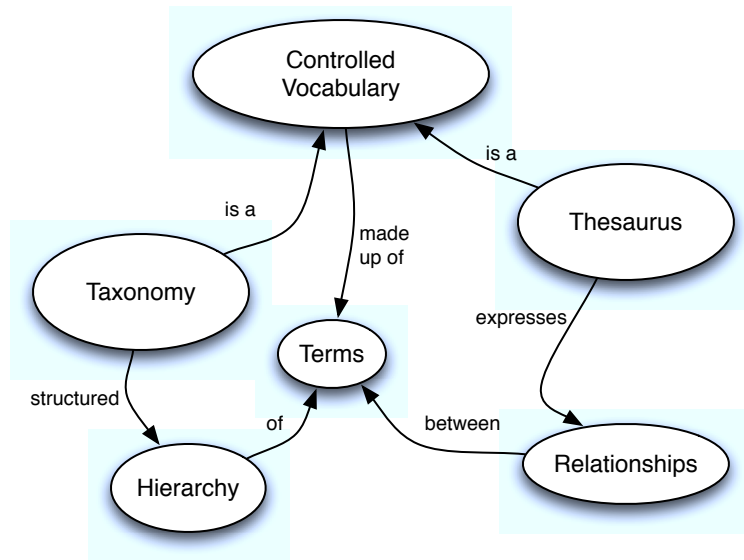
<b>Terminology</b>	<b>Explanation</b>
Document	Used to describe a unit of text. Has nothing to do with the way it is physically stored. It is a logical unit.
Word (Graphic)	A string of contiguous alphanumeric characters with space on either side: may include hyphens and apostrophes, but no other punctuation marks (Kucera and Francis, 1967).
Lexeme	Corresponds to a set of words taken by a single word (Kucera and Francis, 1967). The result of stemming. For example, “give”, “gave”, “gives”, “giving”, and “given” all stem to the lexeme give.
Token	The basic unit in a document. Can represent a word, number, symbol, or letter. e.g. “Heart attack” contains two tokens; “Heart” and “attack”. “B/P: 110/70” contains the other types of tokens. Both “B” and “P” are letter tokens. “/” and “:” are symbol tokens and “110” and “70” are number tokens. For the purposes of this dissertation, a token is surrounded by spaces.
Term/Concept	A single word or multiple contiguous words that represent a concept. The two words in “heart attack” represent a single concept, therefore, those two words can be a single term. “Heart” by itself is also a term. A term can represent one or many concepts. e.g. “cold” can represent a descriptive concept for temperature or it can represent a common illness. A concept can also be represented by one or many terms. e.g. “heart attack” and “myocardial infarction” represent a single concept.

## **Controlled Vocabulary**

In the field of computational linguistics, terms are brought together in collections to be used for processing. A controlled vocabulary is the basic collection used. There are different types of controlled vocabularies and the ANSI/NISO standard Z39.19-2005 (Ano, 2005), which covers vocabularies and the development of them, explains in detail how controlled vocabularies are constructed, formatted, and managed. A controlled vocabulary in its basic form is a list of terms to be used for processing. This is also the foundation for all other vocabulary collections. The goal of a controlled vocabulary is to achieve consistency. Without consistency, the other vocabulary collections would be ineffective. The standard defines relationship indicators and how they are used. Figure 3 shows the relationship between the different types of controlled vocabularies. Table 3 list the different relationships that can be expressed by terms in the different types of controlled vocabularies which are listed in table 4. Disciplines other than medicine use controlled vocabularies as well. The Art and Architecture Thesaurus (AAT), the Getty Thesaurus of Geographic Names (TGN), the Union List of Artist Names (ULAN), and the Cultural Objects Name Authority (CONA) are all vocabularies that contain information on art, architecture, and material culture and are all vocabularies that are created, maintained, and copyrighted by the The Getty Research Institute (The Getty Research Institute, 2011). The Library of Congress Subject Headings (LCSH) has been maintained since 1898 to catalog materials held by the US Library of Congress (Library of Congress, 2011) . LCSH is also used internationally. The Labourline Thesaurus is published by Labour Canada Library Services and consists of industrial relations/human resource management terms (Chaplan, 1995). Another vocabulary resource is the Taxonomy Warehouse (TW:, 2011). This is a repository of vocabularies and contains over 670 vocabularies covering 73 subject domains. Vocabularies covering practically anything from Academic Link Thesaurus for Universities to Zoological Record Thesaurus can be found at the Taxonomy Warehouse.

## **Ontology**

Early students of Aristotle used the term metaphysics to refer to what Aristotle referred to as “first philosophy.” Ontology has also been used as an alternative for metaphysics. The term ontology was first published in 1613 independently by two different philosophers, Goclenius and Lohard (Goclenius, 1613; Lorhard, 1613). Goclenius implies that ontology is concerned with abstract enti-



**Figure 3.:** Controlled Vocabularies

**Table 3:** Vocabulary Relationships

<b>Relationship</b>	<b>Explanation</b>	<b>Expressed By</b>	<b>Example</b>
Equivalency	Used when more than one term can be used to express a single concept or lexical variants.	USE & USED FOR	“MI” & “myocardial infarction” and “orthopedic” & “orthopaedic”
Hierarchy	Express broad to narrow or narrow to broad relationships. Also known as “isa” “hasa”	BT, NT	“broken femur” is a “fracture” and “central nervous system” has a “spinal cord.”
Association	Sibling and derivational relationships.	RT	“broken femur” and “broken radius” are siblings.

**Table 4:** Vocabulary Types

<b>Vocabulary</b>	<b>Explanation</b>	<b>Example</b>
Synonym Ring	Used exclusively in information retrieval. Ensures when a concept is used in a search, any document containing that concept or any synonymous concept can be retrieved.	Searching for all documents pertaining to heart attack, a synonym ring is used to also retrieve documents pertaining to MI, Myocardial Infarction, and any of the other many concepts that represent heart attack.
Taxonomy	Controlled vocabulary with hierarchical structure to it.	
Thesaurus	Controlled vocabulary with order and structure added so relationships between terms are represented by the standardized relationship indicators.	UMLS Metathesaurus, a thesaurus of medical concepts and relationships made up of controlled vocabularies such as ICD <sup>1</sup> and MeSH <sup>2</sup> (Cimino et al., 2008; Tuttle et al., 2008).
Lexicon	The vocabulary including its words and expressions for a language. An inventory of a language's lexemes. It is also synonymous with thesaurus.	

ties and formal structures (Munn and Smith, 2008). The German philosopher, Christian Wolff used the term when referring to Aristotle's "first philosophy" (Wolff, 1736). The computer and information science disciplines have adopted the ontology to be a formal representation of knowledge. Today, some refer to taxonomies, thesauri, terminology, and ontologies interchangeably. There is much discussion as to what an ontology is as can be seen by the many definitions in table 5 (Alberts, 1993; Wielinga and Schreiber, 2010; van Heijst et al., 1997). Even though these definitions differ, they all have the concept of expressing knowledge.

One of the most widely referenced definitions of ontology is from Gruber.

Ontologies: vocabularies of representational terms - classes, relations, functions, objects constants - with agreed-upon definitions, in the form of human readable text and machine-enforceable, declarative constraints on their well-formed use. Definitions may include restrictions on domains and ranges, placement in subsumption hierarchies, class-wide facts inherited to instances, and other axioms.

The Penman Upper Model and the Generalized Upper Model, a descendant of the Penman Upper Model, are examples of linguistic ontologies that exist (Bateman, 1990). The General Upper Model "provides a domain- and task-independent classification system that supports sophisticated natural language processing while significantly simplifying the interface between domain-specific knowledge and general linguistic resources."

### **Unified Medical Language System (UMLS)**

In 1986, The National Library of Medicine (NLM) formed a multidisciplinary multi-site team to create the UMLS (Humphreys et al., 1998). "The purpose of NLM's Unified Medical Language System (UMLS) is to facilitate the development of computer systems that behave as if they 'understand' the meaning of the language of biomedicine and health (National Library of Medicine, 2010)." The UMLS consists of three knowledge sources: Metathesaurus, Semantic Network, and SPECIALIST Lexicon. These tools are provided by the National Library of Medicine and are freely available to anyone possessing a license which is also free.



**Table 5: Ontology Definitions**

<b>Author</b>	<b>Definition</b>
Wielinga and Schreiber	An ontology is a theory of what entities can exist in the mind of a knowledgeable agent
Alberts	An ontology for a body of knowledge concerning a particular task or domain describes a taxonomy of concepts for that task or domain that define the semantic interpretation of the knowledge
van Heijst	An ontology is an explicit knowledge level specification of a conceptualization, which may be affected by the particular domain and task it is intended for
Gruber	A specification of a representational vocabulary for a shared domain of discourse - definitions of classes, relations, functions and other objects - is called an ontology.

### **Metathesaurus**

The Metathesaurus is a large vocabulary database. The contents of this database are provided by source vocabularies such as SNOMED-CT, DSM, and MeSH. The Metathesaurus provides concepts which allow terms from different source vocabularies to be map to concepts. These concepts allow users to link these heterogeneous source vocabularies together. The Metathesaurus also provides relationships between concepts, both parent/child and sibling. For example, the concept ‘myocardial infarction’ has a parent concept ‘structural disorder of the heart’, which has a parent ‘heart disease’ and a sibling ‘infarction.’ These concepts can be represented by terms from different source vocabularies. Of the three knowledge sources, the Metathesaurus is the most widely used (Chen et al., 2007). For the purposes of this dissertation, the UMLS Metathesaurus is considered a thesaurus and not an ontology.

### **Semantic Network**

The Semantic Network currently contains 135 semantic types and 54 relationships. These semantic types are categories in which to group the concepts in the Metathesaurus. There are also semantic grouping such as organisms, anatomical structures, biologic function, chemicals, and

events, which categorize the semantic types. These semantic types can be used to focus NLP efforts toward a specific topic.

### **SPECIALIST Lexicon**

Lastly is the SPECIALIST Lexicon and Lexical Programs. A set of lexical NLP tools have been created and provided by the NLM. The purpose of these tools is help researchers “investigate the contributions that natural language processing techniques can make to the task of mediating between the language of users and the language of online biomedical information resources (National Library of Medicine, 2010).” A part of speech tagger, a spell checker (GSpell), and visual tagging tool are just some of the tools that make up the lexical programs. The SPECIALIST Lexicon provides the general English lexical information which also includes biomedical terms needed by the SPECIALIST Lexical Programs.

## **2.3 Computational Learning Theory**

Where as the first section in this chapter discusses the theories used in NLP, this section discusses the theories that are the underpinnings for Statistical Text Mining (STM) and associated technologies.

Computational Learning Theory also known as Statistical Learning Theory is a mathematical theory related to the analysis of ML algorithms. There are different approaches to computational learning theory. ML theories refer to the term “concept” differently from what has been and will be used in this research. In this section, the term “concept” is used to mean the set of all instances that positively exemplify some simple or interesting rule (Kearns and Vazirani, 1994). For example, in the case of this research, the concept fall is represented by the set of all positive fall documents. A “concept class” or “classification” is a collection of concepts over all instances. For example, the documents in the corpus previously mentioned is a concept class made up of FALL and nFALL concepts. This dissertation is not building any new theories or directly testing any theories traditionally used in Information Systems (IS) research. It does however, combine several existing ML technologies in novel ways to preprocess and classify documents. The three theories that follow are some of the theories that have been used in either the building of the these ML algorithms or in the evaluation of them.

## **Probably Approximately Correct**

Probably approximately correct learning (PAC learning) was proposed by Leslie Valiant in 1984 (Valiant, 1983). The premise of PAC learning is that for a learning algorithm to be considered successful (the correct part), learning an unknown target concept entails obtaining, with high probability (the probably part), a hypothesis that is a good approximation (the approximately part) of the target concept (Haussler, 2011).

## **Vapnik-Chervonenkis Theory**

Vapnik-Chervonenkis theory, otherwise known as VC theory, was proposed by Vladimir Vapnik and Alexey Chervonenkis (Vapnik, 1998). It explains learning from a statistical point of view. VC theory introduced the idea of Support Vector Machines (SVMs) which are another popular prediction model building technique, however, it is not used in the dissertation because it is not designed to be interpretable. VC theory also has a core concept called the VC Dimension. The VC Dimension is a way of quantifying the ease of learning categories from small data sets (Vapnik, 2000; Kearns and Vazirani, 1994).

## **Algorithmic Learning Theory**

Algorithmic learning theory also known as algorithmic inductive inference is another ML framework and was introduced in Gold's seminal paper Language identification in the limit (Gold, 2011). He created this framework when he was investigating language learnability. In his research, information is presented to the learner about an unknown language as well as a class of possible languages. The research question asked is "is the information sufficient for the learner to determine which of the possible languages is the unknown language." The learner receives a unit of information and guesses which language the unknown language is. The language is considered learnable if an algorithm exists that the learner can use to make guesses and after a finite period of time, the guesses are correct. The idea of ML algorithms is based on this theory.

### **2.3.1 Machine Learning**

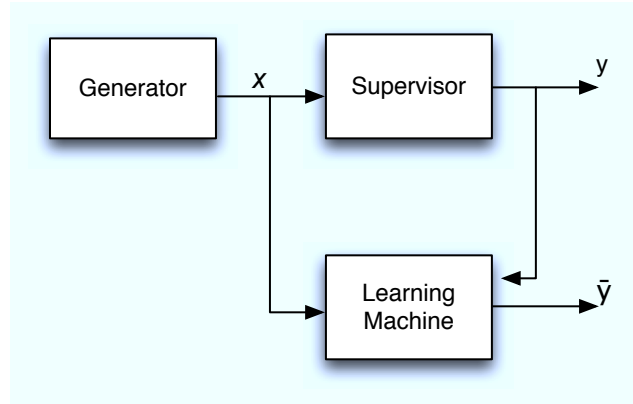
The Oxford Dictionary defines "learning" as the acquisition of knowledge or skill through study, experience, or being taught. From a computer science perspective, this definition fails to include

the conditions that must occur in order for the same results to be achieved from multiple experiences. Repeatable results is the desired effect to be achieved using ML. If a student is taught the same material from professor A or professor B, the testing result of that student will most likely be different for the two professors. However, using ML, given learning sets with the same probability distribution, an algorithm should perform with no statistically significant difference.

### **Types of Machine Learning**

There are two types of ML, supervised and unsupervised. The main difference between the two methods is that in supervised learning, the model has some knowledge of the classification of a subset of the data and learns from this subset. Supervised learning uses a “training” set of data to learn. One of the main uses of supervised learning is classification. The classification of each datum in the training set is known. The ML algorithm then uses this information to then classify data with an unknown classification. Unsupervised learning on the other hand is used for discovery tasks such as clustering. In this case, classifications of the data are not known. The ML algorithm is then used to find patterns or clusters of information within the data. For the purposes of this dissertation, supervised learning techniques are used.

With supervised ML, the machine is given examples chosen randomly and based on those examples it attempts to generate an answer. Figure 4 shows a ML model (Vapnik, 1998). The *Generator* is the source of situations or in this case, documents.  $x \in X$ . For this research,  $X$  is the set of documents or the corpus used in the STM process.  $x$  is fed to both the *Supervisor* and the *Learning Machine*. The *Supervisor* then returns  $y$  for each  $x$ . In other words, for each document  $x$  in the corpus  $X$ , there exists a classification  $y$ . If there are  $l$  documents in the corpus, then the learning machine will have  $l$  pairs,  $(x_1, y_1), \dots, (x_l, y_l)$  to use to learn. The *Supervisor* includes an operator that is used to determine the value of  $y$ . For this research, the *Supervisor* is the group of annotators that established the Gold Standard. The operator used in this case is not a mathematical function but is based on a set of rules that the annotators used to determine the classification for each document. The *Learning Machine* will create its own operator, which approximates the *Supervisor's* operator, to determine the classification,  $\bar{y}$ . Logistic regression, classification rules, and decision tree induction are the ML techniques used in this dissertation.



**Figure 4.:** Model of Learning from Examples

## Machine Learning Algorithms

There are numerous ML algorithms. Decision tree induction is a category of algorithms and there are several DTI algorithms. Table 6 describes many of these algorithms, some of which are used in this dissertation. There are many more but for the purposes of STM, these are some of the more popular.

### 2.3.2 Information Theory

In 1948, Shannon published what became a seminal article on maximizing the amount of information that can be transmitted across an imperfect channel (Shannon, 1948). At the time, it was thought that the faster you transmitted information the more errors occurred. He showed this was not true provided the data were transmitted slower than the capacity of the channel. From this came measurement techniques that are also used in the STM and ML processes. Many of these measurements are used in the decision tree induction process and will be discussed in more detail in a later chapter.

Where as Shannon was interested in reproducing at one point, either exactly or approximately, a message selected at another point, ML is concerned with taking a “message” and predicting whether it is similar to another message. Other concepts such as N-Grams also came out of Shannon’s research (Shannon, 1951).

Information theory is applied in many places in machine learning but specifically two within the technology used within this dissertation. The first place is in the decision selection process

**Table 6:** Machine Learning Algorithms

<b>Algorithm</b>	<b>Description</b>
Association Rule Mining	A method for discovering interesting and meaningful rules between variables or features. Even though association rule mining was not used in its complete form, more information can be found in Chapter 5.
Classification Rule Mining	Similar to association rule mining, however, rather than being concerned with the interestingness of the rule, the rule is judge on how well it is used in a classifier. Specific details on classification rule mining can be found in Chapter 5.
Decision Tree Induction (DTI)	A method in which a decision tree consisting of decision nodes and leaf nodes is created. Decisions are made and can lead to additional decision nodes or leaf nodes. The leaf nodes are the classes for the classifier. There are several DTI algorithms, such as C4.5, ID3, CHAID (CHi-squared Automatic Interaction Detector), and Multivariate Adaptive Regression Splines (MARS). Specific details on DTI can be found in Chapter 6
Support Vector Machine	Given a group of classified data belonging to one of two classifications, an SVM graphs these training cases with as large a gap between the two classifications as possible. It then maps an unclassified case amongst the two classifications and predicts the classification based on which side of the gap it is mapped. SVMs are good for classifications where the cases have large numbers of variables or features.
Clustering	Is an unsupervised methods where cases are grouped together based on similarities.

in DTI. The C4.5 algorithm used in answering the second research question relies on the information gain ration. This is explained in more detail in Chapter 6. Additionally, text mining uses entropy as a possible weighting scheme. It can be used as both a frequency and term weighting scheme. Many models were built to answer the first research question (specialized versus common language) and some used entropy for frequency and term weightings, however, the none of the models selected as having the best performance used entropy for weightings.

### **Entropy**

The entropy of a random variable is the measure of the amount of information in that variable or in other words a quantified measure of uncertainty in a random variable. This indicates how easily message data can be compressed. For example, there is more uncertainty in predicting the number from rolling a 6 sided die as opposed to predicting the flip of a two-sided coin. With the die you have a 1 in 6 chance of predicting it correctly but a 5 in 6 chance of predicting it incorrectly. With the coin, you have a 1 in 2 chance of either predicting it correctly or incorrectly. The uncertainty is less with the coin.

### **Mutual Information**

Mutual information on the other hand is the amount of information one random variable contains about a second random variable. In other words, it is the measure of the information common or mutual between two random variables. Mutual information can be used to measure the information common between two cases.

### **Information Gain**

Related is also information gain also known as Kullback-Leibler divergence (Kullback and Leibler, 1951). Information gain is the change in entropy from one state to another state. Information gain can be used in decision tree induction in evaluating the splitting criterion. An attribute with high information gain is preferable to a lower value. Information gain has a bias toward attributes with many possible distinct outcomes (Quinlan, 1988). For example, an attribute such as patient ID would have a high information gain value than an attribute such as gender but patient ID most likely would not be a good choice for an attribute on which to split.

## **Information Gain Ratio**

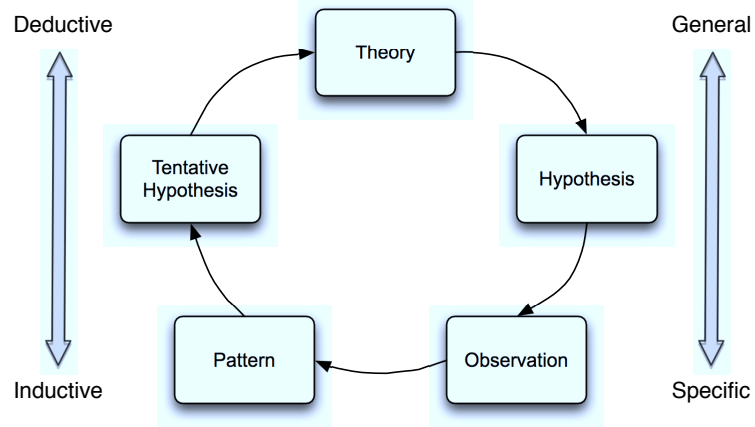
To alleviate the bias issue caused by information gain, information gain ratio or just gain ratio is used. Again, the information gain will be high for an attribute that has many distinct possible outcomes. In many cases, an attribute may have many unique values such as IDs. To calculate gain ratio, information gain is divided by the information learned about the attribute, otherwise known as the split information. For example, if the attribute is an ID, the information gain is going to be a large number because it seemingly gives a lot of information. That is divided by the number of bits learned about the feature. The larger the entropy for the attribute the larger the denominator. In this case, attributes such as IDs will be penalized more than attributes with few possible outcomes. In other words, just as 50 is much larger when compared to 1, however,  $1/2$  is the same as  $50/100$ .

### **2.3.3 Deductive vs Inductive Reasoning**

There are two broad areas of reasoning, deductive and inductive (Trochim, 2000). Using a deductive approach, general truths lead to specific conclusions whereas using an inductive approach, specific observations lead to general conclusions. Darwin applied inductive reasoning in the development of his theories on evolution. Specifically, he observed differences in finches on different islands within the Galapagos islands (Darwin, 1859; Lack, 1940). He observed the finches were isolated on the different islands and reasoned because the finches also had similarities, they all came from a common ancestor but had adapted and evolved to meet the requirements of each individual island. The specific observations he made were of the finches and their unique traits on the different islands. This led him to a general conclusion, evolution and adaptation.

Research can be performed using both types of reasoning (Trochim, 2000). Induction can be used to ultimately create a theory as in the previous example with Darwin. Deductive reasoning can then be used to validate or falsify a theory. Deductive reasoning is a top-down approach. Researchers start with a theory concerning the topic of interest. That is narrowed to a hypothesis that can be tested. Observations are then collected to support the hypothesis which allows specific data to be used to test the hypothesis. Inductive reasoning on the other hand is a bottom-up approach. Specific observations are used to detect patterns which then generalize into tentative hypotheses. These hypotheses can be explored and developed into more general theories. Even though these are two separate methods of reasoning, together they are used in a circular process. See figure 5.





**Figure 5.:** Inductive/Deductive Reasoning

Inductive generalization starts with a premise about a specific sample which leads to a conclusion about a general population (Norton, 2010). For example the premise, 80% of the injury progress notes in the sample having features A, B, and C are fall notes implies the conclusion, 80% of all injury progress notes having features A, B, and C are fall notes. With inductive generalization, the premise is generalized to the conclusion population. Classification tasks or concept learning in general are inductive by nature and classification tasks using unstructured data specifically are performed well using inductive inferencing or ML methods.

Concept learning is an example of inductive learning. Concepts are induced from observations. Bruner defines concept learning as “...the search for and listing of attributes that can be used to distinguish exemplars from non exemplars of various categories (Bruner et al., 1956).” In psychology research, there are many theories that describe concept learning as applied to humans or chimps. Computational Learning Theory and Information Theory provide the foundations for concept learning, also known as supervised ML, where computers are the learners. These theories provide the foundations for the techniques that are used in this research.

## 2.4 Statistical Text Mining (STM)

Data mining allows unknown information to be extracted from structured data. Text mining has the same purpose but it applies to unstructured data. There are three phases to STM. Since STM is performed on unstructured data, adding structure to the data is the first phase. The next phase is identifying patterns and lastly is evaluation and interpretation.

Text mining has been shown to be effective in many domains. For example, text mining is now being applied to social media content. It is one possible way to support Facebook's anti-bullying campaign (Foley, 2011). Documents are collected and manually annotated as bullying or not. These documents are then used to train a model used by Facebook to determine bullying behavior by sites. Another social media use of text mining is a study that was done to determine the most popular subjects posted on the web (Kalafatis, 2010). They determined that "how to" was the most popular subject and the most likely way to improve the popularity of a user's posts and achieve higher click-through rates. Yet another social media use is mining Twitter feeds to determine users' sentiment. Wall Street firms are using this information to inform the buying and selling of stocks (Bollen et al., 2011). One such firm was able to make a 24% gain in a week by buying and selling airline stocks based on Twitter users' sentiment concerning swine flu pandemic outbreak fears (Shell, 2011). Text mining is also used in fraud detection. One study used text mining techniques to process employee emails to detect disgruntled employees, who are a major contributor to occupational fraud (Holton, 2009). Collecting documents has become easier as technology advances, however, evaluating massive numbers of documents has left analysts perplexed. Recent studies have looked at how to apply text mining to this problem. One such study evaluated using text mining to predict new uses for existing technologies, specifically engineering technologies that may have an impact on viral warfare in the future (Smalheiser, 2001). Another use is in intelligence analysis in the context of military, police, and business intelligence. One example is a study that used multilingual text mining of Open Source Intelligence to train intelligence analysts (Baldini et al., 2007). This dissertation will explore statistical text mining techniques as applied to the medical domain.

### **2.4.1 Structure**

Before any processing can take place, the unstructured text must be structured. This is done by first parsing the documents into words or tokens. Next a term-by-document matrix is created. This can be thought of as a spreadsheet where the rows represent the documents and the columns represent each unique word in the corpus. A value is then entered for each cell in the term-by-document matrix this can be anywhere from a 1 or 0 representing presence or it can be some weight determined by one of many weighting schemes. This matrix can be sizable but sparsely populated.

**Table 7: Term-By-Document Matrix**

	<b>D1</b>	<b>D2</b>	<b>D3</b>
cough		X	
day	X		
had		X	
motivated			X
of		X	
packs	X		
patient		X	
per	X		
quit			X
smoking	X		X
to			X
two	X	X	
weeks		X	

Given the following three sample documents, a term-by-document matrix can be created (table 7).

D1: smoking two packs per day

D2: patient had two weeks of cough

D3: motivated to quit smoking

The X's in the matrix represent the presence of the term in the document. One of the things that can be observed is that the matrix is sparsely populated in that two thirds of the matrix has no value because the term does not exist in that document. Perhaps, since a term exists in only one document, that term may not have any predictive power. By reducing the size of the matrix the process can be completed in less time. The term-by-document matrix in (table 7) consists of only three documents and 13 unique words. The processing time would be significantly longer for a corpus of 2,000 documents containing 75,000 unique words. By using dimension reduction techniques, the size of the term-by-document matrix can be reduced thus reducing the processing time. One such method is Latent Semantic Analysis (LSA) using Singular Value Decomposition (SVD) (Deerwester et al., 2010). Factors are created that represent a concept. For example, the matrix made up of 2,000 documents and 75,000 unique terms could be reduced to 200 SVD factors. Each factor contains weights for each term but the matrix is reduced because it is made up of the 2,000 documents and 200 factors. This dissertation introduces another approach to dimension reduction.

By processing the corpus first with an NLP pipeline, the relevant terms can be extracted from the documents, thus reducing the number of unique terms. This technique is discussed in a later chapter.

### **2.4.2 Identifying Patterns**

After the term-by-document matrix is created, a predictive model is created from the terms, or the SVD factors, or a combination of both. Logistic regression (Kurt et al., 2008), decision trees (Wilcox and Hripcsak, 1999), and Support Vector Machines (SVM) (Cortes and Vapnik, 2011; Dumais, 1998) are all popular statistical methods for creating predictive models. A classified subset (training) of the documents is used to train one of the statistical models. The model is then tested on another subset of documents (test). In some cases a third subset (validation) can be used. The model can use the validation subset to improve the performance of the model. Evaluation metrics can be used to evaluate the performance of a specific model.

### **2.4.3 Evaluation and Interpretation**

There are numerous evaluation statistics that can be used to evaluate and interpret the performance of the model. Based on the results of these statistics, the model can be determined as effective or not.

#### **Confusion Matrix**

Before applying one of the statistics, a confusion matrix needs to be created. This matrix will show the numbers of correctly and incorrectly classified documents processed by the model. The center of figure 6 shows the four possible results that can occur from a classification problem. If both the gold standard and model predicted classifications match, then the result is either a True Positive (TP) or True Negative (TN). If, however, the classifications do not match then one of the false results will be applied. A False Positive (FP) is where the gold standard classification is negative yet the model predicted the classification to be positive, or falsely classified as a positive. These are the results of Type I errors. On the other hand, if the gold standard classification is positive and the model predicted the classification to be negative, or falsely classified as a negative, then a False Negative (FN) is the result. These misclassifications are Type II errors.

		Gold Standard Classification		Stats
		Pos	Neg	
Model Classification	Pos	TP	FP (Type I)	<b>Precision (PPV)</b> $TP/(TP + FP)$
	Neg	FN (Type II)	TN	<b>NPV</b> $TN/(TN + FN)$
Stats		<b>Sensitivity (Recall)</b> $TP/(TP + FN)$	<b>Specificity</b> $TN/(TN + FP)$	<b>Accuracy</b> $(TP + TN)/(TP + TN + FP + FN)$

**Figure 6.:** Confusion Matrix and Evaluation Metrics

### Evaluation Metrics

From these four classification results, several statistics can be calculated to help evaluate and interpret the model. Figure 6 shows the statistics and the formulas for calculating them around the outside of the chart. Table 8 lists the evaluation metrics used and a short explanation of each.

Traditional NLP research does not report specificity or NPV (Sokolova and Lapalme, 2009). The reason being is in that type of research, typically there are no cases classified as true negative (TN). A grammatical NLP pipeline with the purpose of extracting terms compared to a human annotated gold standard will find terms that match the human annotation and those will be true positives (TP). It will find terms that were not found by the human annotation and those will be false positives (FP). It will also not find terms that were found by the human annotation and those will be false negatives (FN). The pipeline, however, cannot not find terms that were also not found by the human annotation. Those would be true negatives. Because of this, specificity and NPV are typically excluded from NLP research. Those statistics are however, reported in STM research and will be reported here.

Many different metrics are used to evaluate the performance of a classifying model, among those, accuracy is the most widely used (Sokolova and Lapalme, 2009). Depending on how the model is to be evaluated, some measures will work better than others. A measure is invariant if its value does not change when the confusion matrix changes. For example, if the number of cases making up the (FN) counts were reduced and the cases making up the (TN) counts were increased by the same amount, the metric precision would not change to reflect the change in counts.

**Table 8:** Evaluation Metrics

<b>Statistic</b>	<b>Definition</b>
Sensitivity (Recall)	Proportion of actual positives which are correctly identified as such. This measures the lack of missed classifications
Specificity	Proportion of negatives which are correctly identified.
Accuracy	Proportion of correctly identified of the total identified. It is a balance of Precision and Recall.
Negative Predictive Value	Proportion of actual negatives identified from the total negatives identified.
Precision (PPV)	Proportion of actual positives identified from the total positive identified. This measures the lack of false positives.
F-Measure	Weighted average of the precision and recall.

As mentioned above, traditional NLP does not have a count of TNs. This explains why F-Measure is reported in this type of classification task because if there are zero TNs, the F-Measure calculation would have zero in its denominator. Accuracy on the other hand uses all four counts in its calculation. Where accuracy is not a good evaluator is in a task where one class is very small and the other very large. For example, if a task's data split is 2% for the positive class (class of interest) and 98% for the negative class, the model could simply classify everything as the negative class and achieve 98% accuracy which would be deceiving. Accuracy is an acceptable measure when being compared to baseline. The data split for the data set used in the three studies in this dissertation is approximately 23% for the class of interest and 77% for the negative class. Because the class of interest is not very small, accuracy is the measure that will be used to ultimately compare the models to baseline and decide which model to move forward through the process. Other evaluation metrics will be reported as well but will not be used in the selection of a model. Accuracy is invariant in two situations; first, if the positives and negatives are exchanged, and secondly, when there is a uniform change in the positives and negatives. These two situations are not of concern for this research because the size of the data set is constant and the models are not being compared to models created from other data sets.

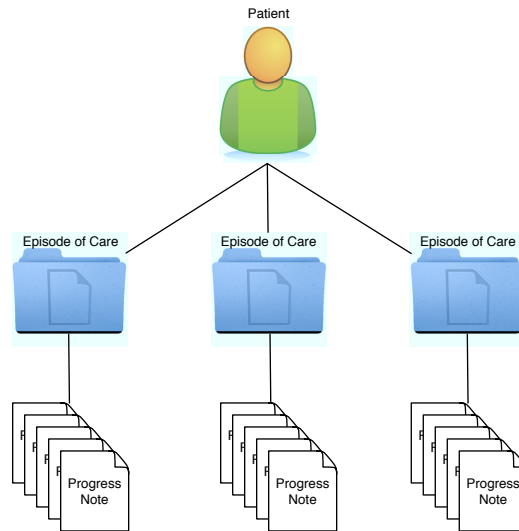
## Chapter 3

### Data

The James A. Haley Veterans Hospital in particular is the largest hospital in the Veterans Health Administration, serving over 115,000 veterans in four counties and is one of five poly-trauma centers in the US. The data set used for this dissertation was originally collected for a pilot study and the goal of this study was to show that fall related injuries treated in out patient clinics within the VA are under coded (Luther et al., 2005).

Within the EHR there is a hierarchical structure. (See figure 7) At the top of this structure is the patient. Below the patient are episodes of care. An episode of care is documented as the main reason the patient is at the facility. It could be anything from an annual physical to a weekly psychiatric care appointment. It also could be a visit to the facility for the treatment of an injury. A patient will have many episodes of care throughout his/her enrollment in the system. Each episode of care has progress notes associated with it. For example, if a patient is at the facility for the treatment of an injury, there may be progress notes from x-ray, nursing, orthopedic, pharmacy, and physical therapy clinicians. Together, these notes document an episode of care and each note, if documented correctly, will have an ICD-9 code for the reason for visit code identifying the treatment of the injury. In some case, this same patient might also have visited, for reasons unrelated to the injury, a facility such as the dermatology clinic while at the facility. The notes associated with treatment at the dermatology clinic in this case would also have a reason for visit code for the treatment of the injury because that is the primary reason for the visit. This hierarchical structure allows inferences to be made at different levels.

Because of this hierarchy, inferences can be made at different levels within the system. Individually, each note can be classified. Based on the classifications of the notes, episodes of care can also be classified, then based on the episode's classifications, patients can be classified. All of these different classifications allow inferences to be made at these different levels. Using this data set, inferences can be made concerning FRI. If an individual note is classified as a "fall" note and



**Figure 7.:** Episode of Care Hierarchy

the reason for visit code is for the treatment of an injury, an inference can be made that it is a FRI. The inference can then be rolled up to the patient level and infer that the patient has experienced a FRI. Based on this, the patient can receive education and services to help prevent future falls.

### 3.1 Cohort

The construction of the research data set started with identifying patients. There are three groups of patients, that make up this cohort. All of the patients in this cohort are fiscal year 2007 patients from a single hospital within the VISN 8 (Veterans Integrated Service Network). First, patients with a reason for visit code for treatment of an injury and an External Cause of Injury code (E880-E888), commonly referred to as an E-code, listed as a secondary diagnostic code were identified. The second group are the matched controls. They were selected from the pool of patients treated for similar injuries but did not receive a fall-related E-code. For example, a patient who fell and broke her arm would have received a reason for visit code for treatment of the broken arm and a fall-related E-code for a secondary diagnostic code, whereas a patient who broke his arm as a result of being hit would not have received a fall-related E-code. These controls were matched based on facility, gender, type of injury, and age. Two matched controls were identified for every patient in the original set. Upon further examination of the patients, it was discovered that some patients had an E-code incorrectly listed as the reason for visit code. These patients made up the



third group and were also added to sample. Because the reason for visit was listed as the E-code, no matched controls were used for these patients. This resulted in a total of 453 patients.

### **3.2 Collection**

All outpatient progress notes were collected for these patients using a 48 hour window around a recorded visit to an outpatient clinic for an injury. This documented an episode of care. A 48 hour window was used for two reasons. First, most clinical notes are recorded immediately after the clinical encounter. Second, if notes are not completed immediately, the Joint Commission on the Accreditation of Healthcare Organizations requires they be completed within 48 hours. All notes that represent an episode of care were collected even if they were not directly related to the care of that injury. This resulted in a total of 5,009 progress notes.

### **3.3 De-Identification and PHI**

In order to comply with Health Insurance Portability and Accountability Act (HIPAA) and the local Institutional Review Board (IRB), the notes were cleansed of protected health information (PHI). A discussion of HIPAA and other health care related regulations can be found in Appendix F.

A program was used to replace names with XXXNAMEXXX. For example,

Mr Johnson saw Dr Smith last week in the ortho clinic.

would be changed to

Mr XXXNAMEXXX saw Dr XXXNAMEXXX last week in the ortho clinic.

Dates were also altered. A random number was generated and added to the original date. All dates concerning a specific patient were modified with the same random number. This way, it could not be determined what day a particular patient received a particular treatment, but the time between visits was preserved so that the data could be used for a temporal study in the future. All other PHI was handled in a similar fashion.

### 3.4 Annotation

In order to use supervised learning techniques, a classified dataset is needed. One way to obtain this is to have humans classify the data set to be used to train and test the classifier. Due to costs, traditionally in NLP a subset of the data set is classified or annotated by humans. In other words, if the data set is comprised of 5,000 records then a subset of the data, perhaps 500 records, would be annotated by multiple annotators. Inter-rater or inter-annotator agreement can then be calculated. If there is disagreement between annotators, an adjudicator is then used to come to an agreement as to the annotation. Using this method, at least two annotators must look at every record. Every record (5,009) in the data set used for this research was annotated by an annotator because the data set was going to be used for something other than pure grammatical NLP. Because of this, having at least two annotators as well as an adjudicator view each record would be very costly. Traditionally in NLP, annotators annotate or mark passages of text that are to then be extracted by the NLP system. This requires multiple annotators view each document in order to gain agreement. The data set for this research simply required a binary document level classification. Since it is a fairly simple task to decide whether a document is pertaining to a FRI or not, a different approach to classifying the documents was used.

Four clinicians in total were used to accomplish this task, one was considered an expert. Three clinicians (A, B, and C) were used to annotate all notes as FRI or not and a clinical expert (D) was used to establish the “gold standard.” All clinicians were trained on the annotation tool and schema then they annotated Training Set 1. Inter-rater reliability was calculated at that point. D was used to create the training standard and the three clinicians training annotations were compared to D’s in order to calculate inter-rater reliability. The clinicians were trained on the differences that appeared and they then annotated Training Set 2. The same process was used as was used on the previous training set. Clinicians were again trained on the differences and began annotating the Final Set. During the annotation process, D was used to rate the reliability of the other clinicians by randomly selecting and annotating 10 notes out of every 1,000 annotated by the other clinicians. Kappa was used to assess inter-rater reliability (Cohen, 2006, 2005). Kappa scores for the three clinicians can be seen in table 9.

Intra-rater reliability was also calculated for the 3 clinicians annotating the dataset. Since the benchmark of 80% inter-rater reliability was not achieved by clinicians A, B, and C on the first

**Table 9: Inter-rater Reliability Kappa Scores (%)**

	<b>D to A</b>	<b>D to B</b>	<b>D to C</b>
Training Set 1	91	84	71
Training Set 2	82	100	80
Final Set	100	100	100

**Table 10: Intra-rater Reliability Kappa Scores (%)**

	<b>A</b>	<b>B</b>	<b>C</b>
Training Set 2	100	100	94

training set that there was no need to compute intra-rater reliability. The benchmark inter-rater reliability was achieved on dataset 2, therefore intra-rater reliability was calculated for clinicians A, B, and C. Table 10 shows the intra-rater reliability Kappa scores for the clinicians.

Two open source tools were used to create the annotations for the dataset; Protege and Knowtator. Protege is a free, open source ontology editor and knowledge base framework (Gennari et al., 2003; Musen et al., 1987). Knowtator is a general purpose annotation tool that was designed as a Protege plug-in (Ogren, 2006). Items annotated were Fall Related Injury (FRI), history of fall, and method of fall. The FRI annotation was used to classify each note as either containing an FRI or not. A tool was used by the clinicians to annotate a fall within the note. The mechanism of the fall, place of the fall, time of the fall, and injury were also annotated.

Patient fell off the stairs in his house yesterday morning and cut his head on a table.

This would be annotated as a fall with stairs being the mechanism, the house being the place, yesterday being the time and cut head being the injury. Because of the fall annotation this note would be classified as a fall. In conjunction with this annotation and the associate ICD-9 code indicating the reason for visit was the treatment of an injury, the note would also be classified as a FRI.

### 3.5 Dataset

The final annotated data set included 1,151 FRI (FALL) and 3,858 not FRI (nFALL) progress notes. The complete data set was then stratified and divided (70%/30%) into a training data set

(TRAIN) and evaluation data set (TEST). This data set will be used for all of the experiments in this dissertation.

## Chapter 4

### Medically Relevant Terminology vs Common Language Study

#### 4.1 Abstract

*The purpose of this research is to answer the question, can medically-relevant terms be extracted from text notes and text mined for the purpose of classification and obtain equal or better results than text mining the original note? A novel method is used to extract medically-relevant terms for the purpose of text mining. A dataset of 5,009 EHR text notes (1,151 related to falls) was obtained from a Veterans Administration Medical Center<sup>1</sup>. The dataset was processed with a natural language processing (NLP) application which extracted concepts based on SNOMED-CT terms from the Unified Medical Language System (UMLS) Metathesaurus. SAS Enterprise Miner was used to text mine both the set of complete text notes and the set represented by the extracted concepts. Logistic regression models were built from the results, with the extracted concept model performing slightly better in the evaluation metrics than the complete note model.*

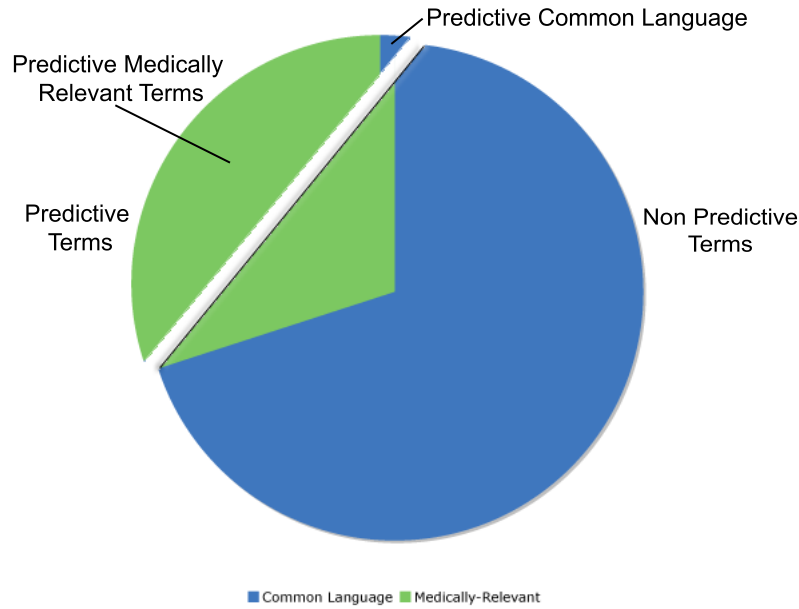
#### 4.2 Introduction

Falls are an important health care issue especially among aging veterans. A history of a previous fall is the single most important clinical indicator that identifies an elderly patient as high risk for additional falls and targets them for fall prevention programs. However, information about fall-related injuries (FRIs) in administrative databases has been found to be significantly under-coded, thereby limiting information about a history of falls to clinicians (Luther et al., 2005).

The purpose of this research is to answer the question can medically-relevant terms be extracted from text notes and text mined for the purpose of classification and obtain equal or better

---

<sup>1</sup>Funding for this research came from the Consortium for Healthcare Informatics Research, [HIR-09-002] HSR&D Center of Excellence, James A. Haley Veterans Hospital, Tampa, FL. This report presents the findings and conclusions of the authors and does not necessarily represent the Department of Veterans Affairs (VA).



**Figure 8.:** Breakdown of Terms

results than text mining the original note? A novel approach was used to extract medically-relevant terms for the purpose of text mining. Figure 8 illustrates the conceptual idea behind this approach. Within each document are two sets of terms: one, medically-relevant and the other, common language terms. Of those terms, it is assumed the medically-relevant terms are more predictive than common language terms (as shown in the separated bottom portion of the figure 8). Thus, this research explores a means of extracting the medically-relevant terms from a document while discarding the common language terms. This research assumes most of the predictive terms will be in the extracted relevant term set. There is probably a small set of predictive terms in the common language but leaving them out of the model is acceptable if the model is at least as accurate as if they were included.

Essentially, we are testing if any important patterns are lost by removing passages of text not annotated via NLP. If no loss occurs, it is quite reasonable to use text or other data mining methods on only the output of the Natural Language Processing (NLP) process. The rest of this paper will discuss the background behind this process, the dataset used, the process used to extract the relevant terms, the model building process, and finally a discussion of the results.

### 4.3 Background

There is disagreement in the literature as to what fields fall under NLP. Most agree that NLP is an area under artificial intelligence (AI) and can also be broken down itself into many areas, some of which are; information retrieval (IR), information extraction (IE), and question answering (Hirschman et al., 2002). IR uses algorithms to retrieve documents in response to a query similar to search engines. Question answering differs from IR in that just the information that answers the question is returned rather than an entire document or search results. It also assumes that the NLP algorithm has some understanding of the question being asked. IE is used to extract names and entities from the text being processed. An intelligent system must be able to form classes of entities united by some principle. It must also be able to recognize membership of an entity in one of these classes (Michalski, 1990). The idea of named entity recognition (NER) is to identify all of the instances of named entities within specified text (Cohen and Hersh, 2005). Some include data and text mining as fields that fall under NLP. For the purposes of this research, we will refer to NLP separately from Statistic Text Mining (STM). This research used NLP to extract the medically-relevant terms or named entities (NEs).

Text mining is the automated discovery of new, previously unknown information from unstructured textual data (Hearst, 1999). Swanson argued that unknown information exists in the medical domain because it would be physically impossible for a human to read all of the available literature and create all of the possible hypotheses (Swanson, 1987). By mining the bibliographies of existing literature, he developed the hypothesis that dietary fish oil improves Raynaud's Syndrome (Swanson, 1986b). Tremblay et al. showed that text mining a set of text notes from electronic health records (EHR) could be helpful in correctly identifying notes pertaining to fall related injuries (Tremblay et al., 2009).

Several tools exist that have been shown to be successful in extracting information from medical text notes. MedLEE has knowledge-based components and processing engines to process text (Friedman et al., 2004). Multi-threaded Clinical Vocabulary Server (MCVS) uses SNOMED-CT codes to process text and represent medical terms (Elkin et al., 2006). The Unstructured Information Management Architecture (UIMA) is an open source project that allows users to build systems that analyze large volumes of unstructured information in order to discover knowledge. UIMA has been used to create NLP applications using a pipeline architecture (Ferrucci and Lally,

2004). The Mayo Clinic built an NLP framework using the UIMA called cTAKES (Clinical Text Analysis and Knowledge Extraction System) (Savova et al., 2008). This system is a UIMA pipeline which processes clinical notes and identifies medical NEs based on the UMLS. Another open source architecture is the General Architecture for Text Engineering (GATE) (Cunningham et al., 2002; Cunningham, 2002). The Health Information Text Extraction (HITEx) tool was created originally to extract smoking status for asthma research (Zeng et al., 2006). In the same way cTAKES is a medically related pipeline built on the UIMA architecture, HITEx is a medically related pipeline built on the GATE architecture.

The UMLS is comprised of the Metathesaurus, Semantic Network, and SPECIALIST Lexicon. The Metathesaurus is a database of medically related terms and is made up of source dictionaries such as SNOMED-CT, ICD, and other sources. Contained in the Metathesaurus are over 1.7 million unique concepts. Each concept is represented by one to many terms. There are also 135 semantic types within the Metathesaurus that allow the concepts to be categorized. The Metathesaurus used for this research was 2009AA.

Medically-relevant terms for each document are obtained from the UMLS Metathesaurus. The sizes of the sets containing the relevant terms and common language are dependent on the source vocabulary used. For example, if the notes are orthopedic notes and an unrelated specialty source vocabulary is used, e.g., Diagnostic and Statistical Manual of Mental Disorders (DSM), the number of the medically-relevant terms will most likely be smaller than if a general source vocabulary was used, e.g., SNOMED-CT.

## **4.4 Methods**

### **4.4.1 Data Set**

#### **Cohort**

The construction of the research data set started with identifying patients. There are three groups of patients that make up this cohort. All of the patients in this cohort are patients from a single hospital within the VISN 8 (Veterans Integrated Service Network) that were treated in fiscal year 2007. Table 11 lists the make up of the three groups. Group 1 consisted of patients with a reason for visit code for treatment of an injury and an External Cause of Injury code (E880-E888),



**Table 11: Cohort description**

Group	Reason for Visit Code	Secondary Dx Code
1	Injury	E880-E888
2	Injury	-
3	E880-E888	-

commonly referred to as an E-code, listed as a secondary diagnostic code. Group 2 made up the matched controls. They were selected from the pool of patients treated for similar injuries but did not receive a fall-related E-code. For example, a patient who fell and broke her arm would have received a reason for visit code for treatment of the broken arm and a fall-related E-code for a secondary diagnostic code, whereas a patient who broke his arm as a result of being hit would not have received a fall-related E-code. These controls were matched based on facility, gender, type of injury, and age. Two matched controls were identified for every patient in the original set.

Upon further examination, it was discovered that some patients had a fall-related E-code as the reason for visit code but had no specific injury listed. These patients made up group 3 and were also added to sample. Because the reason for visit was listed as the E-code, no matched controls were used for these patients. This these groups resulted in a total of 453 patients.

### **Collection**

All outpatient progress notes were collected for these patients using a 48 hour window around a recorded visit to an outpatient clinic for an injury. This constituted an episode of care. A 48 hour window was used for two reasons. First, most clinical notes are recorded immediately after the clinical encounter. Second, if notes are not completed immediately, the Joint Commission on the Accreditation of Healthcare Organizations requires they be completed within 48 hours. All notes that represent an episode of care were collected even if they were not directly related to the care of that injury. This resulted in a total of 5,009 progress notes.

### **Annotation and Partitioning**

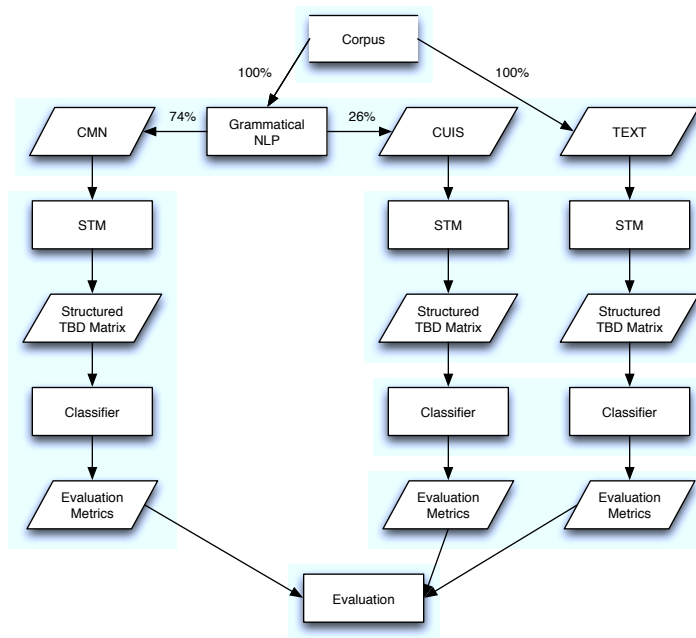
Three clinicians (A, B, and C) were used to annotate all 5,009 notes as fall (FALL) or not fall (nFALL) and a clinical expert (D) was used to rate the reliability of the other clinicians by randomly selecting and annotating 10 notes out of every 1,000 annotated by the other clinicians. After

training sets, the inter-rater and intra-rater reliabilities for the three clinicians had a Kappa score above 95%. This annotated data set became the reference or “gold” standard for the classification task.

In order to comply with Health Insurance Portability and Accountability Act (HIPAA) and the local Institutional Review Board (IRB), the notes were cleansed of protected health information (PHI) using an automated process. The final annotated data set included 1,151 FALL and 3,858 nFALL progress notes and was then divided into two stratified splits, 70% training (TRAIN) and 30% testing (TEST).

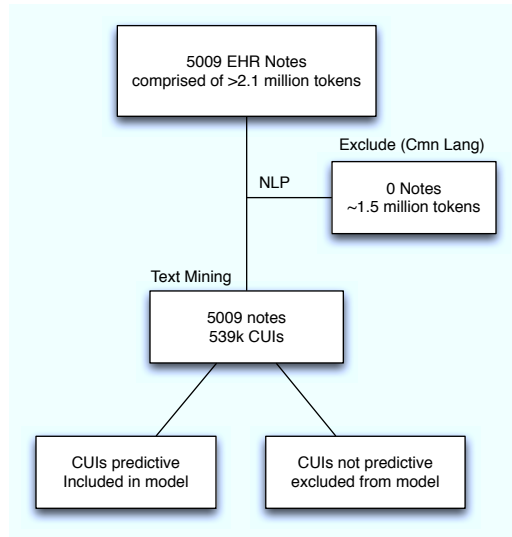
#### 4.4.2 Process

A combination of grammatical NLP and STM are used to create a classifier. This is compared to a classifier created using just STM. Figure 9 shows the flow of data through this process and each step is explained in further detail in the following sections.



**Figure 9.:** Study Process

A modified CONSORT flow diagram is used to show the reduction of the dataset through the process. In this case, no notes were excluded from the dataset, but rather words/terms were excluded throughout the process. See figure 10. These word/terms comprise common language.



**Figure 10.:** CONSORT Flow Diagram

## NLP

For this project, a GATE pipeline was created using openNLP modules for basic parsing and custom components for negation and NE identification. The UMLS concept component was designed to look up noun phrases in the SNOMED-CT source vocabulary of the Metathesaurus in which slightly over 300,000 unique concepts were represented. Because many of the words related to “fall” are verbs, this component was also configured to look up verb phrases as well. The component uses the word(s) in the noun or verb phrase and looks them up in the Metathesaurus. If found, they are annotated as a named entity (NE) and the appropriate properties are set. Specifically, the beginning and ending point in which the NE exists within the document, the source vocabulary, and the negation status are documented. In cases where the NE is comprised of multiple words, the concept that described the most words was the only concept returned. For example, if the phrase is “Regional Medical Center” several concepts can be identified; one for “regional”, “medical”, and “center” as well as one for “medical center” and other combinations of those three words. To avoid inflating word/concept counts, we chose to only use the concept that covered the most words in the phrase. In this case, only the concept for “regional medical center” was used.

The negation component of the pipeline implemented the latest version of NegEx (Chapman et al., 2001). The NegEx algorithm identifies whether a word is negated, possibly negated, or not negated.

A feature extractor component was created to extract the NEs and negation status from the entire set of annotations. Within this component, the list of NEs for a note was parsed, and the CUI for each one was written to a file along with the negation status. The negation status was concatenated to the end of the term. For example, if an NE was annotated as the CUI “C0027051” and was annotated as negated, the feature extractor would convert that CUI to “C0027051\_neg.” The annotations for each note were written to a separate line within a file along with the note ID and classification, fall or not fall.

Word Sense Disambiguation (WSD) also caused concept counts to be inflated. For example, the term “cold” returns seven concepts from the Metathesaurus. Since text mining is not concerned with context, it did not matter what concept was returned for term as long as it was consistent throughout the corpus. The feature extractor component also took all of the CUIs and negation status associated with a term and concatenated them with an underscore. Continuing with the example “cold”, the following was used to represent this concept:

“C0024117nn\_C0719425nn\_C0009264nn...”

By retaining all of the CUIs representing a term, the concepts can be explored as future research.

Another component was created that extracted the common language from each note to a separate file. In essence, if the token was not identified as a concept it was identified as common language. Just as with the concept file, the common language for each note was written to a separate line within a file along with the note ID and classification.

## **Text Mining**

Three different datasets were created from the original set of 5,009 notes. Each dataset contained all 5,009 notes but was comprised of different components from the notes. The first was the original medical text notes (TEXT), the second was the NLP extracted CUIs representing the NEs with negation (CUIIS) and the third was the common language (CMN). In addition, each was also divided into three partitions: training (60%), validation (20%), and test (20%). Using SAS Enterprise Miner 6.1/Text Miner 4.2, models were built from the training partition of each dataset. Then the weights of the terms were adjusted to improve the models using the validation partition. Finally, the models were evaluated against their respective test partition. Based on the evaluation metrics for the validation set, the model with the best accuracy for each dataset was selected.

**Table 12:** Term-by-Document Matrix

	Doc 1	Doc 2	Doc 3
ankle	0	0	1
at	1	0	0
care	0	1	0
fell	1	0	0
health	0	1	0
home	1	1	0
patient	1	0	1
twisted	0	0	1
visited	0	1	0
yesterday	1	1	0

In basic terms, SAS parsed the documents and counted words, then assigned weights to words in order to rank their predictability. A term-by-document frequency matrix is then created. If the following were three documents:

Doc 1 - Patient fell at home yesterday.

Doc 2 - Home health care visited yesterday.

Doc 3 - Patient twisted ankle.

a term-by-document frequency matrix would be created from those three documents (see table 12). Notice that the matrix is sparsely populated in that more than half of the matrix is populated with zeros. With nearly 7,000 unique concepts extracted from the 5,009 documents, the matrix would be quite extensive and presumably just as sparsely populated. Latent Semantic Analysis (LSA) was implemented using Singular Value Decomposition (SVD) and “roll-up terms” were used to reduce the dimensions of the matrix (Albright, 2004). The phrase “roll-up terms” in SAS refers to terms kept for prediction purposes. In this case, using the top 200 roll-up terms and a maximum of 200 SVDs were found to provide the best results.

Several additional things can be done to improve the results of the text mining process. A stop list can be created, which is a list of all of the possibly irrelevant words that might be found in the document. When text mining the TEXT and CMN dataset, the standard SAS stop list was used. This list is comprised of words such as articles, prepositions, and other non-predictive words. Stemming, which is reducing words to their root form, was also performed on these two datasets. For example “giving”, “given”, and “gave” can all be reduced to ‘give’ (Lovins, 1968). Neither a

stop list nor stemming was used on the CUIS dataset because they consisted of conceptual terms and not actual words.

#### 4.5 Results

Table 13 show the number of tokens comprising the data set before the NLP pipeline was executed. Table 14 shows the number of tokens in the data set containing just the common language (CMN) and the data set with just the CUIs (CUIS). Notice that the CMN and CUIS numbers do not sum to the TEXT numbers. This is because some of the concepts represent multiple terms.

**Table 13:** Token Make-up of TEXT Data Set

		<b>TEXT</b>			
		<b>Total</b>	<b>Avg/Note</b>	<b>Min</b>	<b>Max</b>
<b>TEST</b>	<b>FALL</b>	183,666	535	16	3,656
	<b>nFALL</b>	462,865	399	5	3,885
	<b>Total</b>	646,531	467		
<b>TRAIN</b>	<b>FALL</b>	486,518	607	26	3,569
	<b>nFALL</b>	1,021,351	378	3	3,681
	<b>Total</b>	1,507,869	492		
<b>Total</b>		2,154,400	480		

**Table 14:** Token Make-up of CMN and CUI Data Sets

		<b>CMN</b>				<b>CUIS</b>			
		<b>Total</b>	<b>Avg</b>	<b>Min</b>	<b>Max</b>	<b>Total</b>	<b>Avg</b>	<b>Min</b>	<b>Max</b>
<b>TEST</b>	<b>FALL</b>	134,073	391	6	2,590	44,160	129	5	943
	<b>nFALL</b>	330,944	286	3	2,855	119,339	103	1	951
	<b>Total</b>	465,017	338			163,499	116		
<b>TRAIN</b>	<b>FALL</b>	357,889	446	13	3,054	114,441	143	6	663
	<b>nFALL</b>	732,966	271	1	2,657	261,105	97	1	909
	<b>Total</b>	1,090,855	359			375,546	120		
<b>Total</b>		1,555,872	348		539,045	118			

Logistic regression models were created from the results of the text mining. Over 45 models were used testing different weighting schemes, numbers of SVD factors and terms, regression selection model and criteria, and other SAS settings. See Table 15 for the specifics for each dataset.

**Table 15:** Text mining and regression settings

Setting	TEXT	CUIS	CMN
SVD Resolution	Low	Low	Low
Max SVD Dimensions	200	200	200
Freq Weight	Binary	Binary	Log
Term Weight	None	None	None
Num Rollup	200	200	200
Drop Other Terms	No	No	No
Diff POS	No	N/A	No
Select Model	Stepwise	Stepwise	Stepwise
Technique	Default	Congra	Default
Select Criterion	Default	Valid Misclass	Default

**Table 16:** LR Comparison (%)

	TEXT	CUIS	CMN	% Diff
Accuracy	91.68	93.08	81.69	1.01%
F-Measure	81.86	85.10	56.69	2.25%
Precision (PPV)	82.22	86.59	52.48	3.91%
Specificity	94.72	95.99	86.53	1.08%
Recall (Sensitivity)	81.50	83.66	61.64	0.72%
NPV	94.48	95.00	90.34	0.55%

Table 16 shows the evaluation metrics, comparing the logistic regression models for TEXT, CUIS, and CMN. The last column in the table shows the percent difference between the metrics for TEXT and CUIS.

Table 17 shows the top 5 predictive terms from the model. The middle column shows the terms from the model for TEXT. The right column shows the terms generated from the model using for CUIS. Notice that “visible” and “date” exist in both lists, however, they did not receive the same ranking for each model. It was not until term 10 that the lists begin to differ greatly. These results demonstrate that removing common language terms has no negative impact on the results of text mining in order to classify these documents.

## 4.6 Discussion

The text mining process benefits from the NLP output with the addition of negation. If, for example, every note had the phrase “fall,” text mining would not select this as predictive. On the other

**Table 17:** Top 5 Predictive Terms from Text Mining TEXT and CUIS

Rank	TEXT	CUIS
1	fall	fall (C0085639)
2	date	visible (C0205379)
3	visible	date (C2348077)
4	therapy	therapy (C0087111)
5	riskof	riskof (C0035647)

hand, if 80% of the notes had “fall” and 20% of the notes had “didn’t fall,” that might be predictive but since text mining tools do not typically associate negation with terms, it would just identify “fall” in 100% of the notes and not use the term. By creating a single term made up of the NE and the negation status, “C0085639\_neg,” terms can be more precisely determined as predictive or not.

Since the inception of HIPPA, de-identifying health records before medical research can be performed has become paramount. Tools exist for this purpose such as De-ID (website) and HMS Scrubber (Beckwith et al., 2006). One research group has reported recall of 97% and precision of 75% in de-identifying EHRs (Neamatullah et al., 2008). By using NLP to extract only the CUIs from EHRs, a side effect is that PHI is removed from the final dataset. A possible case where the PHI is not removed is if a patient or clinician has a last name that is also a vocation such as Baker. Baker could be identified as a CUI.

The baseline for this set of notes is approximately 77%. In other words, if all of the notes were classified as nFALL, 77% accuracy would be achieved. As can be seen in Table 16, text mining the common language resulted in slightly better accuracy, 81.69%. Our original assertion was that there is a small amount of predictive information in the common language, which this result suggests. However, by leaving this predictive information out, the accuracy of the model using medically relevant terms (CUIS) does not suffer. In fact, it improves.

#### **4.6.1 Error Analysis**

85 documents from the test dataset were incorrectly classified by the model, 55 were Type I errors and 30 were Type II errors. These documents were analyzed by a human and four categories of sources of error surfaced. They included misspellings, clinician judgments in annotating, fall history, and templates/grammar.



## **Misspellings**

Some of documents identified as Fall notes by the annotators used the term “feel”, “feal”, or “fal” for “fell” or “fall.” Clinicians writing the documents can be in a rush and spelling is not their main concern and rightly so. These accounted for six of the false negatives. The term “feel“ is used in documents classified as Fall and NotFall to describe how a patient feels so this term is not a good predictor. The model was able to accurately classify other notes containing these misspelled terms, however, these six notes must not have enough of the other terms used in the model to be accurately classified.

## **Clinician Judgments**

Annotation judgments were made by the clinical annotators when classifying documents. For example, “patient found unconscious on floor” or “wheelchair overturned with patient in it” were judged by annotators that because of what happened, the patient must have fallen. None of the top predictive terms were in these notes. These notes accounted for 12 false positives and 10 false negatives. Because there was not a clear line separating what differentiated a FALL document from a History of a Fall document, judgments were also made to classify these notes. A note containing, “CC: patient fell two months ago and didn’t seek treatment. Complains of pain in rt ankle.” is not clearly either of the classification categories. It could be classified as a History of a Fall and therefore a nFALL document because of the time frame from when the injury occurred. It also could be classified as FALL document because this seems to be the first time the patient is seeking care for the injury but it is not clear as to whether the fall was the cause of the injury.

## **Fall History**

As mentioned previously, documents classified as a History of a Fall were not classified as FALL documents. Of the misclassified documents, 30 of the false positives and seven of the false negatives were classified as History of a Fall. Many of these documents contained the same types of statements as FALL notes. For example, in many notes “follow-up“ was the only thing that differentiated a FALL document from a History of Fall document. In the test dataset, there were a total of 265 documents classified as History of a Fall and therefore nFALL. Of these, the model was able to correctly classify 228 of the History of a Fall documents as nFALL.

## Templates/Grammar

Lastly, is that EHR text notes are rarely grammatically correct which makes it even more challenging to correctly identify sentences and parts of speech. It is not so much that they are grammatically incorrect as it is that they contain clinician's thought written as phrases rather than complete correct grammatical sentences. Clinicians are rightly most concerned with giving veterans the best care as quickly as they can. Because of this, they write notes using phrases and widely accepted codes such as *c/c* for chief complaint. To complicate this is the use of templates and an aging EHR system. The EHR system used by the VA is one of the oldest still being used and it stores the text from notes in 80 character fields. This is a leftover from mainframe days. Because of this sentence and part-of-speech tagging is challenging.

We currently use a regular expression sentence splitter that when all else fails, marks an end of line as the end of the sentence. The reason for this is templates. A template within a note may look like the following:

Stroke? Yes Heart Attack? No

If this template is parsed to end of line, then the appropriate negation assertion will be associated with the concept. In other words, the patient did have a stroke and did not have a heart attack. In contrast, if there is also textual information within the same note and sentences span multiple lines, negation can be detected incorrectly. For example:

Patient experienced loss of consciousness but did not  
suffer a concussion.

This will be parsed into two sentence because there is nothing to denote a sentence on the first line so the end of line will mark the end of the first sentence. This is because there is an end-of-line character after "not." The period on the second line will denote the entire second line as a separate sentence as well. The problem caused by this is that "concussion" should be negated but because the negation is on the previous line and in turn the previous sentence, it will be incorrect. In other words, by marking sentences as the end of line if punctuation does not exist, template information will be correctly processed but textual entries will be incorrectly processed.

Another option for sentence marking is not to use the end of line to denote a sentence. This causes problems of its own in that the template information will now be incorrectly annotated. The "Yes" in the first example will now be associated with heart attack and nothing is associated with

stroke so strokes assertion will be correct but heart attacks assertion will be incorrect. This method does, however, correct the problem with sentences spanning multiple lines. In the second example, “concussion” will be negated because both lines will be marked as a single sentence. Using this method, the opposite results are achieved from marking sentences at ends of lines.

As far as we can determine, this problem is unique to the VAs VistA (Veterans Health Information Systems and Technology Architecture) EHR system. The main reason is that the system was originally developed decades ago and textual information could only be stored as a maximum of 80 characters. More modern systems developed using RDBMS have larger fields for text. That means an end of line does not appear unless someone actually presses return/enter.

#### **4.6.2 Limitations**

One limitation has already been mentioned previously. Templates cause issues with the NLP process and resolving this is a large challenge. At this time, SAS Text Miner does not have cross fold validation available. This or a similar method is necessary in order to test if the results are significant and not a result of the way the data was split. SAS also does not have Support Vector Machines (SVMs) available in Text Miner. Testing other models could provided interesting results. In a concurrent study, regular expressions were able to achieve 91% accuracy with only a single rule that detects “fall” or “fell” with 12 constraints to handle negation. In other words, FRIs are fairly easy to detect. At this point, we are not sure this generalizes to other medical topics.

#### **4.6.3 Future Work**

The subject of the data for this study was falls and the source library, SNOMED-CT, was used in identifying NEs from the text notes. Duplicating this experiment on other datasets with different classifications is needed to generalize the results. A future study is also needed to determine which source libraries to use in identifying NEs. Is it best to use all of the source libraries available in the Metathesaurus or is it better to scale back and use only those relevant to the subject? For example, if the study topic was Post Traumatic Stress Disorder (PTSD) would it be better to use the source library DSM or let the Metathesaurus decide which NEs are relevant and use all of the available source libraries? Additionally, our current results show an increase in all of the statistics listed in table 16, however, future research will undertake cross-fold validation in order to determine sta-

tistical significance between the performance measures from the models. Further analysis is also needed to determine parts of speech for both the CUIS and CMN. As mentioned previously both verb and noun phrases were used to locate concepts. An interesting analysis would be to compare the results of NLP using exclusively verb phrases, exclusively noun phrases and the combination used in this research.

#### **4.6.4 Conclusion**

This research described a way to reduce EHR text notes to only medically-relevant terms by using NLP to extract NEs from text notes. The results were used in text mining to classify the notes as either FALL or nFALL and all of the evaluation metrics were slightly better than those for text mining notes. This shows that important patterns are not lost by removing common language, effectively reducing the information to be processed and in turn maximizing efficiency. Since no loss occurred, this extracted data can serve as a foundation for other data mining research such as targeted information extraction and even clinical discovery.

## Chapter 5

### Rule Mining Study

#### 5.1 Abstract

*This research explores a data/text mining-based approach to classifying textual progress notes within the electronic health record (EHR). A multi-step process that involves natural language processing, statistical text mining, association rule mining, and contrast sets is used to create classifiers that can accurately classify progress notes. A dataset of 5,009 EHR clinical progress notes (1,151 related to falls) was obtained from a Veterans Administration Medical Center<sup>1</sup> and annotated to indicate the presence or absence of fall-related injuries. An automated classification process was developed by using a combination of customized, open source software that creates a classifier comprised of the best combined rule sets. The preliminary results demonstrate that the process does create reasonable classifiers, though not quite as good as our benchmark models based on statistical text mining. However, the resulting rule-based classifiers are easily interpretable and can serve as a base for handcrafted expert refinements.*

#### 5.2 Introduction

Falls are an important health care issue especially among aging veterans. A history of a previous fall is the single most important clinical indicator that identifies an elderly patient as high risk for additional falls and targets them for fall prevention programs. However, information about fall-related injuries (FRIs) in administrative databases has been found to be significantly under-coded, thereby limiting information about a history of falls to clinicians (Luther et al., 2005).

Traditionally, rules have been used for two purposes, classification tasks and information discovery. These rules can either be handcrafted by subject matter experts or created through an

---

<sup>1</sup>Funding for this research came from the Consortium for Healthcare Informatics Research, [HIR-09-007] HSR&D Center of Excellence, James A. Haley Veterans Hospital, Tampa, FL. This report presents the findings and conclusions of the authors and does not necessarily represent the Department of Veterans Affairs (VA).

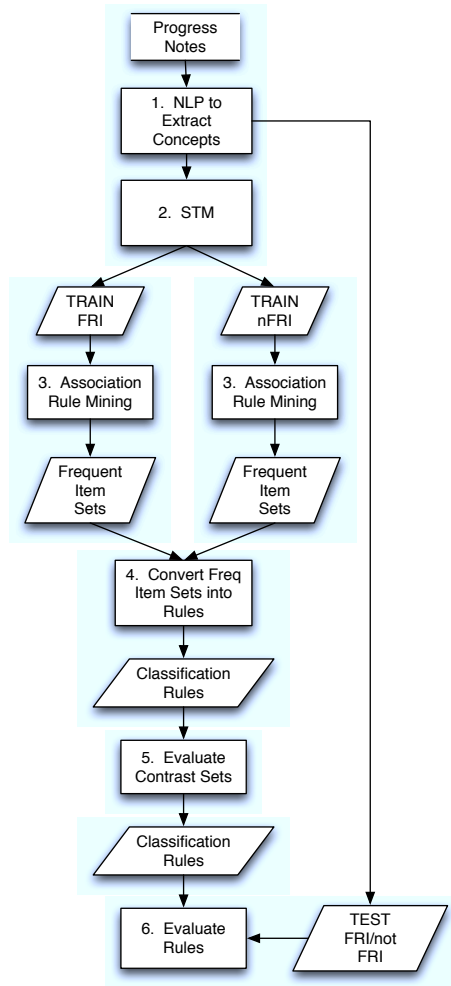
automated process. This research explores a data/text mining-based approach to classifying textual progress notes within the electronic health record (EHR). A multi-step process that involves natural language processing, statistical text mining, association rule mining, and contrast sets is used to create classifiers that can accurately classify progress notes.

Rules were created using an approach that modifies the process of creating association rules. Figure 11 shows a high-level overview of the flow of data throughout this rule mining/classification process. Each of these steps will be described in more detail in subsequent sections of this research. Briefly, the progress notes are processed using a Natural Language Processing (NLP) pipeline that extracts the medically relevant concepts from the documents. Concepts representing the original progress notes are split into training and test data splits and within each, fall (FALL) and not fall (nFALL). The rule mining process is run against the training data and contrast sets are evaluated. The resultant rules are then used to classify the test split of data and the results are analyzed.

### **5.3 Background**

In previous work (Jarman and Berndt, 2010, 2011), we showed that medical progress notes can be separated into two types of language: specialized and common. The specialized language was represented by Concept Unique Identifiers (CUIs) that were extracted from medical progress notes using a grammatical NLP pipeline in conjunction with the Unified Medical Language System's (UMLS) Metathesaurus. Slightly better accuracy in classification prediction using Statistical Text Mining (STM) was achieved using just specialized language (i.e., CUIs) compared to all language in the progress notes. The previous study demonstrated that predictive information was not lost by separating the two types of language from each other within the progress note that make up the data set. This study builds on our prior work by using the same data set and the extracted CUIs.

There are two basic approaches to machine learning: supervised and unsupervised. Supervised is used to classify based on a model built using pre-classified data whereas unsupervised is used to cluster data or discover patterns within the data. There are many methods that can be used to classify documents, some of which use rule based approaches. Decision tree induction is one method whereby the process iteratively selects a set of attributes that most effectively splits the sample data into subsets to create a classifier (Quinlan, 1993). Rule mining is another rule based



**Figure 11.:** Flow of Rule Mining/Classification Process

approach and can be used in both supervised and unsupervised learning. Many use the terms association rules and classification rules interchangeably (Freitas, 2000). However, for the purposes of this research, the aim of association rules mining is unsupervised pattern discovery, while classification rules are used to classify data sets.

Association rule mining can be used to extract rules from a data set which can be used for knowledge discovery (Agrawal et al., 1993; Agrawal and Srikant, 1994). An association rule contains an antecedent and a consequent. The antecedent implies the consequent, where the antecedent is a set of items that appear in the documents/transactions and the consequent also appears in the same transactions/documents but does not exist in the antecedent. For example, customers that buy beer and chips also buy diapers. “Beer and chips” together is the antecedent and diapers the

consequent. One issue with association rule mining is how to limit the number of rules extracted. Without limits, rules can be extracted that exist in a very few documents/transactions. The total number of rules created can also be quite large, leading to information overload. The concepts of support and confidence are used to constrain the rules that are surfaced. Support is used to retain item sets that meet a minimum predefined pervasiveness. If support is set at 20%, only item sets that exist in at least 20% of the documents/transactions are then retained. In other words, if support is set at 20%, then at least 20% of the customers at a store must buy beer, chips, and diapers to qualify. Confidence is also used to constrain the rules retained. If minimum confidence is set at 30%, then 30% of the customers of the store that buy beer and chips, also buy diapers. Using these two statistics, the number of rules can be reduced to a far more manageable number. An example of pruning based on support and confidence follows.

Problem: Retain frequent item sets meeting at least 20% support. Table 18 shows a list of item sets and item set 3 is pruned because it does not meet the minimum support of 20%.

**Table 18:** Example Item Sets

No	Item Set	Support (%)
1	Corned Beef, Cabbage	35
2	Beer, Chips, Diapers	20
	***** Prune Below *****	
3	Lunch Meat, Drier Sheets, Eye Liner	9

Problem: Retain rules meeting a minimum confidence of 30%. Table 19 show a partial list of rules created from the retained items sets in table 18 (Item sets 1 & 2). Rules 4 and 5 are pruned because they do not meet the minimum confidence of 30%.

Researchers have used many approaches to creating classification rules (Li et al., 2001; Liu et al., 1998; Zimmermann and Raedt, 2004). In general, rules are created where the antecedent is the set of information that predicts the consequent or classification. Some automated methods have achieved accuracy levels comparable to human crafted rule classifiers (Apte and Damerau, 1997).

In classification rule mining, the same item sets can be extracted and applied to multiple classifications with differing support. These are known as contrast sets (Bay and Pazzani, 2001).



**Table 19: Example Rules**

No	Antecedent	Consequent	Confidence (%)
1	Corned Beef	Cabbage	35
2	Beer, Diapers	Chips	32
3	Beer, Chips	Diapers	30
	***** Prune Below *****		
4	Chips	Diapers	15
5	Beer	Chips	12

It is necessary to identify the contrast sets in classification rule mining in order to create the most accurate classifier. If an item set occurs in the positive classification data with similar support as in the negative classification data, the rules created from this item set may not be useful in the classifier. On the other hand, a contrast set may have very high support in one classification and very low support in the other classification. The rules created from this item set may be very helpful in the classifier. Therefore, it is necessary to identify and analyze the contrast sets among frequent item sets, looking for asymmetries that are useful in discriminating between classes.

## 5.4 Methods

### 5.4.1 Data Set

#### Cohort

The construction of the research data set started with identifying patients. There are three groups of patients that make up this cohort. All of the patients in this cohort are patients from a single hospital within the VISN 8 (Veterans Integrated Service Network) that were treated in fiscal year 2007. Table 20 lists the make up of the three groups. Group 1 consisted of patients with a reason for visit code for treatment of an injury and an External Cause of Injury code (E880-E888), commonly referred to as an E-code, listed as a secondary diagnostic code. Group 2 made up the matched controls. They were selected from the pool of patients treated for similar injuries but did not receive a fall-related E-code. For example, a patient who fell and broke her arm would have received a reason for visit code for treatment of the broken arm and a fall-related E-code for a secondary diagnostic code, whereas a patient who broke his arm as a result of being hit would not have received a fall-related E-code. These controls were matched based on facility, gender, type of

**Table 20: Cohort description**

Group	Reason for Visit Code	Secondary Dx Code
1	Injury	E880-E888
2	Injury	-
3	E880-E888	-

injury, and age. Two matched controls were identified for every patient in the original set.

Upon further examination, it was discovered that some patients had a fall-related E-code as the reason for visit code but had no specific injury listed. These patients made up group 3 and were also added to sample. Because the reason for visit was listed as the E-code, no matched controls were used for these patients. This these groups resulted in a total of 453 patients.

### **Collection**

All outpatient progress notes were collected for these patients using a 48 hour window around a recorded visit to an outpatient clinic for an injury. This constituted an episode of care. A 48 hour window was used for two reasons. First, most clinical notes are recorded immediately after the clinical encounter. Second, if notes are not completed immediately, the Joint Commission on the Accreditation of Healthcare Organizations requires they be completed within 48 hours. All notes that represent an episode of care were collected even if they were not directly related to the care of that injury. This resulted in a total of 5,009 progress notes.

### **Annotation and Partitioning**

Three clinicians (A, B, and C) were used to annotate all 5,009 notes as fall (FALL) or not fall (nFALL) and a clinical expert (D) was used to rate the reliability of the other clinicians by randomly selecting and annotating 10 notes out of every 1,000 annotated by the other clinicians. After training sets, the inter-rater and intra-rater reliabilities for the three clinicians had a Kappa score above 95%. This annotated data set became the reference or “gold” standard for the classification task.

In order to comply with Health Insurance Portability and Accountability Act (HIPAA) and the local Institutional Review Board (IRB), the notes were cleansed of protected health information (PHI) using an automated process. The final annotated data set included 1,151 FALL and

3,858 nFALL progress notes and was then divided into two stratified splits, 70% training (TRAIN) and 30% testing (TEST).

#### **5.4.2 Process**

Figure 11 shows the flow of data throughout this rule mining/classification process. The following sections are numbered to correspond to the appropriate process in the diagram and provided further explanation of that process.

##### **1. Concept Extraction**

As stated previously, the data set for this research was created and first used in a prior research project. The details of process 1 in figure 11 are presented as part of this prior research and can be reviewed in Jarman et al. (Jarman and Berndt, 2010). Not only were the concepts extracted from the progress notes in this previous research but data splits were also created that were used previously and in this current research. The data set of 5,009 progress notes was divided into two stratified splits, 70% training (TRAIN) and 30% testing (TEST). TRAIN was further separated into FALL and nFALL data splits as indicated by the human annotators. The result was 802 FALL documents ( $TRAIN_{FALL}$ ) and 2,705 nFALL documents ( $TRAIN_{nFALL}$ ).

##### **2. Rule Mining**

The two TRAIN splits were the input into this next process (process 2 in figure 11) for rule mining. The rules for this research were created using traditional association rule mining techniques in that frequent item sets with a specified support were created (Agrawal and Srikant, 1994). These frequent item sets became the antecedent of the classification rules. The consequent became the document's classification. In other words, if the frequent item sets were being created on the  $TRAIN_{FALL}$  documents, then the consequent for those rules became the classification FALL.

RapidMiner 5.1 (Mierswa et al., 2006), a data and text mining application, was used for the rule generation. RapidMiner code modules, called operators, were used to mine the documents and create a term-by-document matrix, where individual CUIs represented terms, for each TRAIN split. From this matrix, frequent item sets were created for  $TRAIN_{FALL}$  with a support of 40% or in other words, the item sets created existed in 40% of the documents. This generated

42 frequent item sets comprised of 10 distinct concepts. The same process was performed against  $\text{TRAIN}_{n\text{FALL}}$  using the same support with no item sets resulting. RapidMiner has a feature that allows the system to reduce the required support until a specified number of item sets are created. This was used until 50 item sets were created. The support used and the number of item sets to be created were both selected to establish a manageable number of item sets for each split in this initial analysis.

### **3. Contrast Sets**

The frequent item sets from both  $\text{TRAIN}_{\text{FALL}}$  and  $\text{TRAIN}_{n\text{FALL}}$  were compared to identify contrast sets in process 3 in figure 11. Of the 42 item sets from  $\text{TRAIN}_{\text{FALL}}$ , 14 of them also existed in  $\text{TRAIN}_{n\text{FALL}}$ . These contrast sets were identified but not removed at this stage of the analysis. Pruning did not occur at this stage but rather after the rules had been built and tested. The reason is that if one of the resultant rules from the contrast sets predicted FALL with high accuracy, perhaps 85%, and predicted nFALL with low accuracy, maybe 15%, it may be beneficial to keep the rule.

### **4. Rule Creation**

In process 4 in figure 11, the classification rules were created from the 42 FALL frequent item sets. The frequent item sets, individually, became the antecedent of each rule and the classification, FALL, became the consequence of each rule. The resultant rules were then turned into regular expressions using GATE's JAPE (Java Annotation Patterns Engine) (Cunningham, 2002). Using these JAPE regular expressions, a GATE pipeline was created that first parsed each document into tokens and sentences. Next, a JAPE transducer was used to execute each JAPE regular expressions against the TRAIN documents. Lastly, a component was used that wrote to a text file, the document ID and the results of the evaluation metrics for each rule for that document ID. This would be used later in the analysis.

### **5. Rule Evaluation**

Even though the GATE application allows for this pipeline to be executed against a defined corpus of documents, GATE also has an API (Application Programming Interface) that allows for other applications to be created accessing the functionality of GATE. An application was used that

accessed the GATE API to automate the process of combining rules and evaluating them. The individual rules were first executed against the TRAIN split. Based on accuracy, the application determined the best set of rules, which became the classifier. This classifier was then tested on the TEST split. This occurred in process 5 in figure 11 and of which the details follow.

First, each rule was evaluated separately against the TRAIN split. Consider that a single rule can be made up of a single CUI or multiple CUIs “anded” together. Next, the computer created rule combinations iteratively based on the accuracy of each possible rule combination. For example, the rule with the highest accuracy was selected and evaluated then the rule that provided the highest accuracy in combination with the previous rule set was added and evaluated. It should be noted that by adding rules and creating combinations the accuracy did not always increase. If there was a tie in the accuracy of multiple rules, the first rule in the list was used.

Lastly, the set of rules was then pruned at the first combination determined the most accurate. The result was a single combination of rules that was then executed against the TEST split and evaluation metrics were collected.

## 5.5 Results

After the rule mining process was run against the TRAIN split, the 42 resultant frequent item sets were comprised of 10 concepts which can be seen in table 21. Each singular CUI was itself a frequent item set within the 42 and the support can be viewed in the table.

**Table 21: Concepts in Itemsets**

<b>CUI</b>	<b>Concept</b>	<b>Support (%)</b>
C0085639	Fall	90.9
C0030193	Pain	79.9
C0030705	Patient	71.8
C0391850	Pulse	55.1
C0013227	Medication	50.4
C0020517	Allergy	47.8
C0439505	Per Day	46.9
C0686904	Patient Needs	45.5
C1261322	Clinical Evaluation	44.5
C0277786	Chief Complaint	44.3

Table 22 shows the top eight predictive item sets from the association rule mining process.

Evaluation metrics were collected for all 42 rules, however, only the top eight are expressed here because they were the only ones used to create the rules with the highest accuracy.

**Table 22:** Highest Predictive Itemsets

<b>Rule</b>	<b>Concepts</b>	<b>Support (%)</b>	<b>Accuracy (%)</b>
2	Fall & Pain	72.6	87.7
19	Fall & Per Day	42.4	82.0
10	Fall & Pain & Pulse	48.1	86.1
14	Fall & Pain & Time	44.8	82.6
23	Fall & Pain & Allergy	41.1	83.0
24	Fall & Pain & Medication	41.0	83.0
25	Fall & Patient & Pulse	40.5	84.4
5	Fall & Pain & Patient	55.2	84.3

The baseline accuracy for this data is 77%. In other words, if every document was classified as a nFALL document, 77% accuracy would be achieved. As can be seen in figure 12, the accuracy does not drop below the baseline until after rule 17 is added. Table 23 shows the top four predictive rule combinations. The accuracy of using just rule 2 is 87.7% on the TRAIN split. By adding the rule that provided the next highest accuracy in combination with rule 2, rule 19, the overall accuracy is increased to 88.0%. Rules can be redundant (Klemettinen et al., 1994). Redundant rules do not add to or increase the predictive power of the classifier. In this case, the highest classification accuracy is achieved with the combination of rules 2 and 19. As can be seen by the last two rule combinations in the table, adding subsequent rules does not change the accuracy. Therefore, the rules below the line in this table were pruned.

**Table 23:** Top Predictive Item Set Combinations on TRAIN (%)

<b>Item Set(s)</b>	<b>Accuracy</b>	<b>F-Measure</b>	<b>Precision</b>	<b>Recall</b>	<b>Specificity</b>
2	87.7	73.0	73.4	73.6	92.2
2 & 19	88.0	74.9	71.6	78.6	90.8
2 & 19 & 10	88.0	74.9	71.6	78.6	90.8
2 & 19 & 10 & 14	88.0	74.9	71.6	78.6	90.8
...					

Based on these metrics, the combination of rule 2 and rule 19 classified with the best performance and table 24 show the evaluation metrics of both rule 2 by itself and the combination of rules 2 and 19 on the TEST data set.

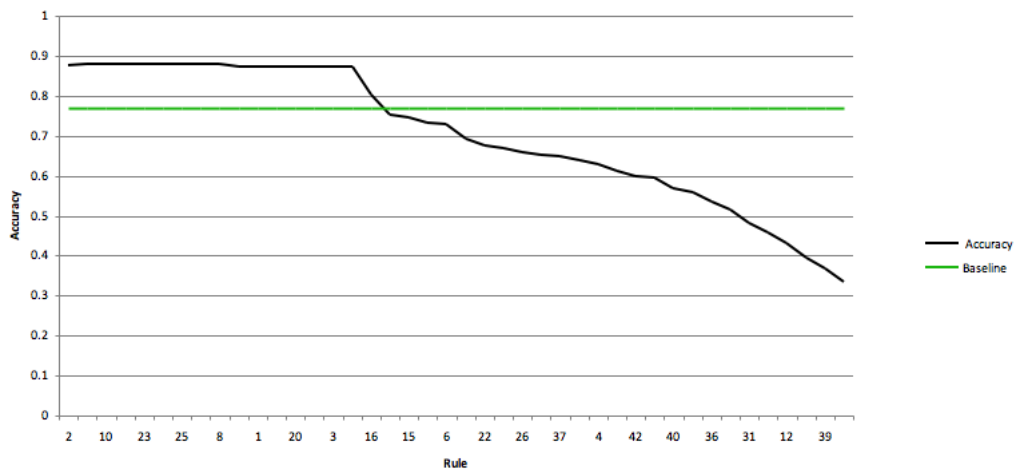
## 5.6 Discussion

One of the challenges of association rule mining is determining interesting or meaningful rules. With classification rule mining this is not necessary because the desired result is the most accurate classification not the most accurate classification based on meaningful rules. However, one of the benefits to using the specialized language (CUIs) in the rule mining process is that much of the language (common) that creates the uninteresting rules is removed. With that said, the two rules providing the most accurate classifier are interesting and can be interpreted. Rule 2 contained the concepts “Fall” and “Pain” and together should be a good predictor of a FRI document. Many of the patients that fall are elderly and have blood pressure issues. If someone with blood pressure issues gets up too quickly, they can become dizzy and in some cases black out. Progress notes that mention “fall” and “pulse” might be referring to a similar issue. The rules containing “medication” and “per day” might also be referring to the same issue.

None of the rules used in the classifier was one of the contrast sets. Therefore, no pruning of contrast sets was needed.

**Table 24:** Top Predictive Item Set Combinations on TEST (%)

Item Set(s)	Accuracy	F-Measure	Precision	Recall	Specificity
2	86.9	70.8	72.0	79.7	92.0
2 & 19	87.5	74.0	70.4	77.8	90.3



**Figure 12.:** Rule Accuracy

As mentioned previously, this research utilizes a data set that was the result of a previous research project. That project used STM to create a classifier to classify the FALL documents and was able to do so with 92.22% accuracy. Even though the accuracy achieved by this current project was not as good as that using STM, this was the first attempt using a rule mining approach and with further efforts might achieve results as good if not better.

Another data set, created the same way, from another medical center was available for both the prior and this research. The rules created on TRAIN and evaluated against TEST were also evaluated against this other data set (TEST2). In the previous research, the final STM model trained on TRAIN and evaluated on TEST was also evaluated on TEST2. A comparison can be seen in table 25. The percent change between the two models between the two data set is similar for accuracy and specificity, however, the difference in percent change for recall between the two data sets is notably different. This shows that the generalizability of the rules is similar to STM where accuracy is concerned, however, the false negative rate is notably greater.

**Table 25:** Evaluation Comparison of Different Data Sets

<b>Data Set</b>	<b>Classifier</b>	<b>Accuracy (%)</b>	<b>Precision (%)</b>	<b>Recall (%)</b>	<b>Specificity (%)</b>
TEST	Rules	87.5	70.4	77.8	90.3
	STM	93.1	82.2	81.5	94.7
	Percent Diff	6.4	16.8	4.8	4.9
TEST 2	Rules	87.3	58.0	50.6	93.6
	STM	93.1	73.8	82.6	94.9
	Percent Diff	6.7	27.2	63.2	1.4

### 5.6.1 Error Analysis

Document were classified by clinicians as FRI if a fall was the mechanism of the patients injury. In other words, if the document said, “patient tripped and injured his ankle yesterday,” that document was classified as a FALL document. On the other hand, if the document said, “patient has fallen several times in the past year and is here today because she bumped her head getting out of the car,” that document was classified as a history of a fall because the mechanism of her injury was not a fall but it clear states she has a history of falling. During the human classification process, most history of a fall documents were classified as nFALL. The goal of this study was to be able to classify current falls. Because of this distinction, this classifier has more Type II errors than



it would have if history of fall and FALL were combined. This is evidenced by the specificity decreasing as the accuracy increases. In the TRAIN split there were 127 history of fall documents classified as nFALL. Of those 127 documents, 107 were classified incorrectly by the classifier as FALL documents.

Upon reading the notes associated with errors, some specific patterns were found. Rule 2 contains the concept “Pain”. In many FALL notes, pain is mentioned frequently. Not only is the pain mentioned but the location of the pain is also mentioned, such as hip pain or back pain, both of which are unique concepts themselves. One improvement to the model would be to be able to roll up all of the different pain related concepts to “pain”.

Another source of error is the misspelling of “fall” and “fell.” “Fall” was misspelled as “feel”, “fal”, and “feal.” A human can read the note and tell by the context that “fall” was the correct term even though the clinician mistyped the term. In many cases a spell checker would not help because even though they misspelled “fall”, the misspelling was a correct word, such as “feel.”

Many of the type II errors were documents that contained fall risk assessments and mentions of pain. The classifier incorrectly classified these documents as FALL because of the assessments.

Some of the human classifications were based on judgment calls. For example, one document does not contain the term “fall” or any of its other forms. The document mentions that the patient slipped while exiting the bus and ruptured her achilles tendon. The document does contain the term “pain.” Even though the note does not mention the patient falling, the annotator determined this injury was caused by a fall.

As with any human task, error is associated with it. A few of the misclassified documents were actually correctly classified and the human annotator had misclassified the document. Out of the 186 documents reviewed that were misclassified by the classifier, six were determined to be incorrectly classified by the human annotators.

## **5.6.2 Limitations and Future Work**

Frequent item sets contain sets of items that occur frequently, hence the name. They do not contain items that do not occur or are omitted from the transactions/documents. One of the limitations of

the method used in this research is that constraints are not added to rules. For instance, to address the problem mentioned above with history of fall documents, perhaps the concepts “History” and “Fall” appear in nFALL documents but not in FALL documents. The combination of rules 2 and 19 might benefit from the addition of “but not History”. Future research will address this issue.

Because this was the first attempt at building a rule based classifier, a manageable number of frequent item sets was used. Future research will use more traditional values for support which in turn will create more frequent item sets and more rules. A tool that will convert the frequent item sets into regular expressions will be created to make the larger number of item sets still manageable.

The accuracy of the classifier developed could be the result of the data split. The same process could be performed against different splits of the data. It might be interesting to compare the resulting rules from multiple data splits. The same process could be performed against multiple different splits of the data (i.e., x-fold cross validation) to provide a more robust measure of accuracy.

### **5.6.3 Conclusion**

There are other methods for creating a classifier that might have better accuracy as well as precision and recall. Support vector machines (SVMs) can be used and are especially useful where large variable sets exist. The problem is that using other methods, the resultant predictive model cannot always be interpreted by a human. One of the benefits of creating a classifier with rules is that the rules themselves are interpretable. Not only do rules as a classifier have the benefit of being easier to understand and interpret but there is also the added benefit of being able to potentially improve the performance of the classifier by using handcrafted expert refinements.

## Chapter 6

### Decision Tree Induction Study

#### 6.1 Abstract

*This research explored a data/text mining-based approach to classifying textual progress notes within the electronic health record (EHR). The primary goal for this research was to create a classifier that accurately identified progress notes of patients being treated for a fall-related injury during that episode of care. The secondary goal was to create a classifier that can be easily interpreted. A multi-step process that involved Natural Language Processing (NLP), Statistical Text Mining (STM), and data mining was used to create rule-based classifiers that accurately classify progress notes. A data set of 5,009 EHR clinical progress notes (1,151 related to falls) was obtained from a Veterans Affairs Medical Center and annotated to indicate the presence or absence of documentation of the patient being treated for a fall-related injury during that episode of care. A process was developed that used a combination of customized, open source software to create a classifier comprised of a decision tree. The preliminary results demonstrate that the process does create reasonable classifiers as well as production rules that are easily interpretable and can serve as the basis for a clinical decision support system (CDSS).*

#### 6.2 Introduction

Falls are an important health care issue especially among aging veterans. A history of a previous fall is one of the most important clinical indicators that identifies an elderly patient as high risk for additional falls and targets them for fall prevention programs (Ganz et al., 2007). However, information about fall-related injuries (FRIs) in administrative databases has been found to be significantly under-coded, thereby limiting a clinicians access to information about a history of falls (Luther et al., 2005).

In health care, there are many uses of Knowledge Discovery and Data Mining (KDD), one of which is to classify documents. Based on those document classifications, inferences can then be made at different levels such as episode of care or patient. For example, if a document can be classified as documenting a fall related injury, an inference can be made that that patient is at risk for a future fall. Surveillance systems as well as Clinical Decision Support Systems (CDSS) are examples of systems within health care that are built using classification methodologies. In general, a “black box” approach for a classifier, in which predictors are not specified to users, is acceptable in a surveillance system. Accuracy of the classification is the most important concern and how the classification was achieved is not as important. In CDSSs, the accuracy of the classifier is important but so is the interpretability of the resultant model. The use of Decision Tree Induction (DTI) in CDSSs has become popular because it can have acceptable levels of accuracy in classification tasks while at the same time creating human understandable rules (Eom et al., 2008). Table 26 shows a sampling of studies that have analyzed the use of decision trees in CDSSs.

**Table 26: CDSS Decision Tree Studies**

<b>Author</b>	<b>Medical Topic</b>
Murphy (Murphy, 2001)	Breast Cancer Diagnostic Errors
Qu et al. (Qu et al., 2002)	Prostate Cancer Detection
Won et al. (Won et al., 2003)	Detection of Renal Cell Carcinoma
Geurts et al. (Geurts et al., 2005)	Classification of Proteomic Fingerprints
Olanow et al. (Olanow and Koller, 2007)	Management of Parkinson’s Disease

The primary goal for this research was to create a classifier that accurately identified progress notes of patients being treated for a fall-related injury during that episode of care. The secondary goal was to create a classifier that can be easily interpreted. A hybridized approach was used to create this classifier. First, a grammatical Natural Language Processing (NLP) pipeline was used to extract the medically relevant terms from progress notes. These medically relevant terms representing the progress notes were then fed into a Statistical Text Mining (STM) process which provided structure to the text. Next, this structured version of the data was fed into a DTI process which built the classifier. Evaluation metrics were gathered to evaluate the model and provided guidance in pruning the tree. The resultant pruned tree model was then used to classify testing data and the results were analyzed. Finally, production rules were created from the resultant tree which can be interpreted by clinicians.

## 6.3 Background

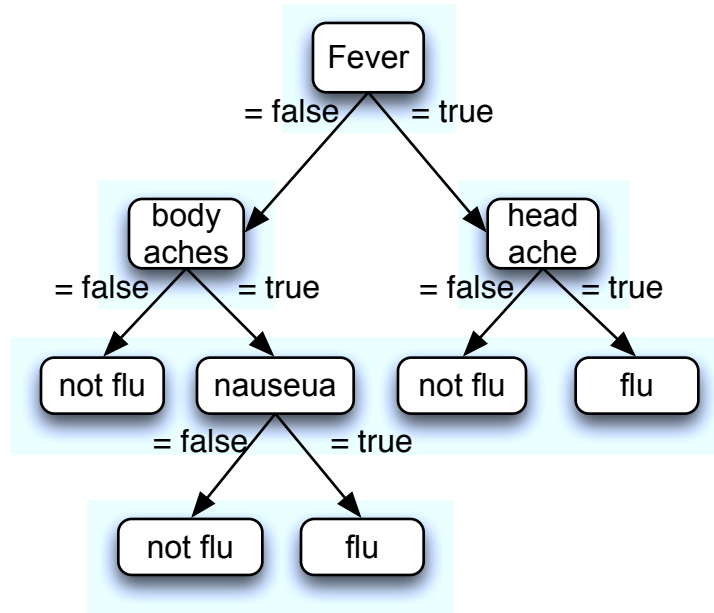
In previous work (Jarman and Berndt, 2010, 2011), we showed that medical progress notes can be separated into two types of language: specialized (medically relevant) and common, without losing predictive information. The specialized language was represented by Concept Unique Identifiers (CUIs) that were extracted from medical progress notes using a grammatical NLP pipeline in conjunction with the Unified Medical Language System’s (UMLS) Metathesaurus. Slightly better accuracy in classification prediction using Statistical Text Mining (STM) was achieved using just specialized language (i.e., CUIs) compared to all language in the progress notes. The previous study demonstrated that predictive information was not lost by separating the two types of language from each other within the progress notes that make up the data set. This study builds on this prior work by using the same data set and the extracted CUIs.

### 6.3.1 Decision Tree Induction

There are many decision tree induction algorithms. They all have the same basic idea in that the tree is made up of decision nodes and leaf nodes. A decision is made at the decision node and the response is either positive or negative, assuming a binary decision tree. Based on that response, the next step in the tree is either another decision node or a leaf node. The leaf node represents a classification. Figure 13 is a graphical representation of an example decision tree. The flu symptoms *fever*, *body aches*, *head ache*, and *nausea* represent decision nodes. The *flu* and *not flu* nodes represent leaf nodes.

The decision tree induction process is traditionally performed in two steps. In the first step, an induction process constructs the decision tree. In the second step, the decision tree is “pruned” or reduced in order to prevent over-fitting and to make the tree more interpretable by a person. One of the earliest works in creating a tree of decisions based on a set of cases is Hunt’s *Experiments in Induction* (Hunt et al., 1966). In this work, they implemented several Concept Learning Systems (CLSs).

Independent of Hunt’s work, other researchers developed similar methods. Specifically, Friedman’s work was the underpinnings of both the CART (Classification and Regression Trees) system and Quinlan’s ID3 algorithm (Friedman, 1977; Breiman et al., 1984; Quinlan, 1986). C4.5, a descendant of ID3, is a set of programs that create classification models based on decision tree



**Figure 13.:** Example Decision Tree

induction and the main program is also referred to as C4.5 (Quinlan, 1993). The decision tree induction algorithm selected for this research is based on the C4.5 algorithm.

To correctly apply C4.5 to a classification task, there are some key requirements concerning the data and the classification task itself. These requirements are described in table 27 (Quinlan, 1993).

**Table 27:** C4.5 Requirements

Requirement	Description
Attribute-Value Description	Attributes must be consistent among the data and must not vary from one case to another.
Predefined Classes	Categories to which cases are assigned must have been previously established.
Discrete Classes	Classes sharply delineated. A particular case either does or doesn't belong to a class. There must be more cases than classes.
Sufficient Data	A detailed classification model usually requires hundreds or even thousands of training cases.
"Logical" Classification Models	Models build decision trees or sets of production rules only.

## Decision Selection

One way DTI algorithms differ is in the way they select the decision nodes. Hunt suggested that Information Theory could be used to determine this (Hunt et al., 1966). CART use the Gini index to derive the splitting criterion at each decision node (Breiman et al., 1984). The Gini index measures inequality or in this case how often a case would be incorrectly classified if it was classified based on the distribution of classes among the training cases. ID3 uses information gain from Information Theory to evaluate a splitting criterion. Using information gain as the criterion has a strong bias in favor of tests with many outcomes. To alleviate the bias, Quinlan uses gain ratio in C4.5, which is robust and gives a better choice of tests than gain criterion (Quinlan, 1988).

Each decision node in the tree has “costs” associated with it. Traversing 10 nodes to classify a case is more “expensive” than traversing five nodes to classify a case. The goal is to select a “cheap” tree but not at the expense of a low classification rate.

Once the tree is constructed, a case is classified by starting at the root and traversing the tree, responding to decision nodes until a leaf is reached. Evaluation metrics are collected in order to make decisions on tuning the tree for better performance.

### 6.3.2 Pruning

A tree which is constructed until a leaf node represents a single class, or in other words no false positives or false negatives, might over fit the data on which it was trained causing a higher error rate when used to classify unseen test cases. This could also potentially be an “expensive” tree. In addition, such an “expensive” tree may be too large to be interpretable by a person. This is where tree pruning comes into play.

The idea in pruning a tree is to remove decision nodes in the tree that do not contribute to the classification accuracy or will reduce the error rate on the unseen data. This creates a less complex tree which in most cases is more easily understood by a human. The problem is how does one base decisions on unseen data? There are two schools of thought on pruning.

- Deciding not to divide any more (stopping). The problem with this method is you may stop dividing while some benefit may still exist in dividing or you may stop too late where the tree is barely simplified (Breiman et al., 1984).

- Removing some of the decision nodes after they have been built. The problem with this method is that you have already incurred the “cost” of building decision nodes that will be discarded. Also, how do you know when to stop pruning?

C4.5 uses the second method and constructs an over fit tree then prunes it back (Quinlan, 1993). The “cost” of building an over fit tree is offset by the benefits of exploring the entire tree before pruning it.

These two methods of pruning do not address the problem of lowering the error rate on unseen data. Decision nodes cannot be removed to reduce the error rate on training data because that is the exact reason they were added to the tree. There are two methods of predicting the error rate.

- Reserve a small set of data from the training data. In other words, if the original data set was split 70%/30% (training/test), split the 70% into two more splits. Use it to calculate error rates and make pruning decisions based on this set. The disadvantage with this method is that data may be scarce and the training data will be even smaller by setting aside a subset. One way around this is to use a cross validation approach.
- Use the entire training set to train the model and calculate predicted error rates (resubstitution error) of the test data set based on the training set to make pruning decisions (Breiman et al., 1984). This is referred to as pessimistic pruning and is the method used in C4.5.

If a specific leaf node covers  $N$  training cases and covers  $E$  of them incorrectly, then the resubstitution error rate is  $E/N$ . Pessimistic pruning uses a confidence interval along with the resubstitution error rate to calculate the upper limit on the probability. This upper limit becomes the predicted error rate and it is this error rate that is used to decide whether to prune a node or not. Quinlan suggests in his book, “...this does violence to statistical notions of sampling and confidence limits, so the reasoning should be taken with a large grain of salt (Quinlan, 1993).” With that said, this method has produced acceptable results.

There is a stream of research in discovering ways to improve existing pruning methods and new ways to implement pruning techniques (Mansour, 1997; Mehta et al., 1995; Rastogi and Shim, 2000). Another study designed a pruning strategy based on minimizing loss rather than minimizing error (Bradford et al., 1998).



### 6.3.3 Production Rules

A graphical depiction of a tree can give a person an at-a-glance basic understanding of the tree. The person can start at the root and traverse the tree by making decisions at each node until a leaf node is reached. Another way to express the tree is using a set of production rules. Production rules can give a person further understanding of tree. For instance, what if a clinician was only concerned with the portion of the tree that classified the positive cases? The clinician can be provided with the production rules that result in positive classifications, which may provide some insight.

In the example decision tree in figure 13, there are two paths through the tree that result in a false classification and three paths that result in a true classification. These individual paths result in production rules (Quinlan, 1987). A rule is comprised of an antecedent and a consequent and can be worded as an “If Then” statement. The “If” portion of the rule is the antecedent and the “Then” portion of the rule is the consequent. Take the rule “**If** fever=true and body aches=true and nausea=true **Then** the class is flu” for example. The antecedent is the conjunction of the decision nodes for a specific path and the consequent is the leaf node for that path. An entire tree can then be represented by a number of rules that matches the number of leaf nodes. The tree in figure 13 can be represented by five rules because there are five leaf nodes.

There is a stream of research that investigates these rules and pruning them based on individual error rates. This is beyond the scope of this project. The rules for this project were used for the purpose of representing the tree and eventually giving clinicians additional insight into the classifier.

## 6.4 Methods

### 6.4.1 Data Set

#### Cohort

The construction of the research data set started with identifying patients. There are three groups of patients that make up this cohort. All of the patients in this cohort are patients from a single hospital within the VISN 8 (Veterans Integrated Service Network) that were treated in fiscal year 2007. Table 28 lists the make up of the three groups. Group 1 consisted of patients with a rea-

**Table 28:** Cohort description

Group	Reason for Visit Code	Secondary Dx Code
1	Injury	E880-E888
2	Injury	-
3	E880-E888	-

son for visit code for treatment of an injury and an External Cause of Injury code (E880-E888), commonly referred to as an E-code, listed as a secondary diagnostic code. Group 2 made up the matched controls. They were selected from the pool of patients treated for similar injuries but did not receive a fall-related E-code. For example, a patient who fell and broke her arm would have received a reason for visit code for treatment of the broken arm and a fall-related E-code for a secondary diagnostic code, whereas a patient who broke his arm as a result of being hit would not have received a fall-related E-code. These controls were matched based on facility, gender, type of injury, and age. Two matched controls were identified for every patient in the original set.

Upon further examination, it was discovered that some patients had a fall-related E-code as the reason for visit code but had no specific injury listed. These patients made up group 3 and were also added to sample. Because the reason for visit was listed as the E-code, no matched controls were used for these patients. This these groups resulted in a total of 453 patients.

### **Collection**

All outpatient progress notes were collected for these patients using a 48 hour window around a recorded visit to an outpatient clinic for an injury. This constituted an episode of care. A 48 hour window was used for two reasons. First, most clinical notes are recorded immediately after the clinical encounter. Second, if notes are not completed immediately, the Joint Commission on the Accreditation of Healthcare Organizations requires they be completed within 48 hours. All notes that represent an episode of care were collected even if they were not directly related to the care of that injury. This resulted in a total of 5,009 progress notes.

### **Annotation and Partitioning**

Three clinicians (A, B, and C) were used to annotate all 5,009 notes as fall (FALL) or not fall (nFALL) and a clinical expert (D) was used to rate the reliability of the other clinicians by ran-

domly selecting and annotating 10 notes out of every 1,000 annotated by the other clinicians. After training sets, the inter-rater and intra-rater reliabilities for the three clinicians had a Kappa score above 95%. This annotated data set became the reference or “gold” standard for the classification task.

In order to comply with Health Insurance Portability and Accountability Act (HIPAA) and the local Institutional Review Board (IRB), the notes were cleansed of protected health information (PHI) using an automated process. The final annotated data set included 1,151 FALL and 3,858 nFALL progress notes and was then divided into two stratified splits, 70% training (TRAIN) and 30% testing (TEST).

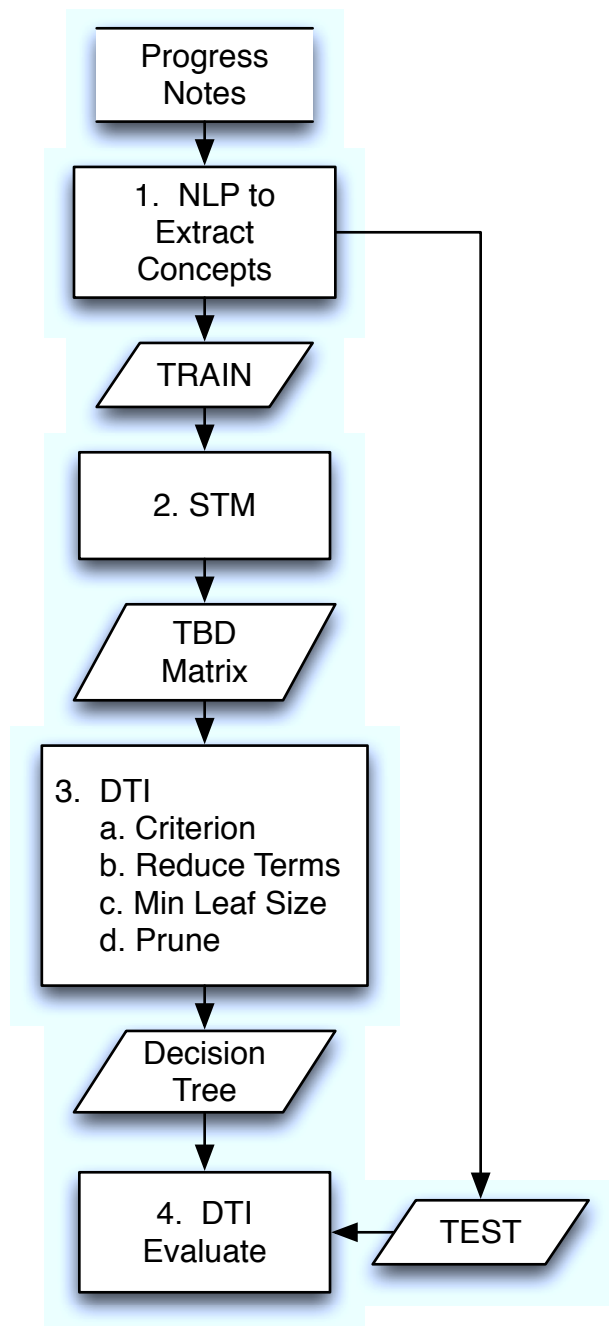
#### **6.4.2 Process**

The process used to create the classifier is a hybrid approach consisting of NLP, STM, and DTI. The NLP and STM were discussed in our previous research mentioned above, therefore, the details of those processes will not be discussed here (Jarman and Berndt, 2010, 2011). Instead, the focus will be on the DTI process. Figure 14 shows the flow through this process. The numbered sections below correspond to the numbered steps in the diagram and further describe the process.

GATE (General Architecture for Text Engineering) was used for the NLP portion of the process (Cunningham, 2002). RapidMiner 5.1 (Mierswa et al., 2006), an open source data and text mining application, was used for the STM and DTI processes. RapidMiner code modules, called operators, were used to mine the documents and create a term-by-document matrix, where individual CUIs represented terms. Other operators were used to implement the DTI process which was based on C4.5. 10 fold cross validation was also used in the model building process. The average accuracy of the 10 folds was used during the model building process to determine which model to use in subsequent steps.

##### **Step 1. Concept Extraction**

As stated previously, the data set for this research was created and first used in a prior research project (Jarman and Berndt, 2010). Using a grammatical Natural Language Processing (NLP) pipeline, the concepts were extracted from the progress notes along with each concepts assertion status, either negated or not negated. The result is a set of CUIs, or medically relevant terms, that



**Figure 14.:** Flow of Decision Tree/Classification Process

**Table 29:** Term-by-Document Matrix

	Doc 1	Doc 2	Doc 3
ankle	False	False	True
care	False	True	False
fell	True	False	False
health	False	True	False
home	True	True	False
patient	True	False	True
twisted	False	False	True

represents each progress note.

## **Step 2. Statistical Text Mining (STM)**

Most of the time, one thinks of decision trees being built from structured data or in other words column row values from a database. In this case, the data is unstructured. Each note contains a set of CUIs and the number of CUIs representing each note varies from note to note. In order to apply structure, the data set was processed using STM. The result was a term-by-document matrix. Binary term occurrence was used to populate the vectors within the matrix. In other words, True represented the presence of a term in a specific document and False represented the lack of that term in a specific document. Table 29 shows an example of a term-by-document matrix. This matrix provides structure for three documents. Using binary term occurrence, regardless as to whether a term exists once or many times in a document, True is the value for the term for a specific document.

The handling of missing data is a challenge in data mining. There are many ways to deal with this challenge, however, the structure supplied by the term-by-document matrix also provides a solution for how to handle missing data. Refer back to the example matrix in table 12. More than half of the matrix contains “False” as a value. In essence, these are missing values. The term “ankle” becomes a variable to be used in the decision tree induction. It only exists in Document 3, therefore, there would be missing data for this variable for Documents 2 and 3. By using STM along with binary term occurrence, the value “False” for the variable “ankle” for Documents 2 and 3 provides an accurate value which represents a lack of the term’s existence.

The resulting term-by-document matrix contained 6,250 terms which became the attributes to be used in the tree induction process. Terms occurring in the documents once were removed

from the matrix and are not included in the 6,250. This process was used in our previous studies as well.

### **Step 3. DTI**

Since the primary goal of this research was to create an accurate classifier, the trees created by the induction process were evaluated on accuracy. The models were created using the TRAIN portion of the data set. After creating several models, each was evaluated and judged on its accuracy. The model with the best accuracy was selected and then pruned to make it more understandable by a person. Multiple steps were used to create this model. A description of this process and the steps used follows.

#### **Step 3a. DTI - Criterion**

The first step was to decide what criterion was to be used for the selection of the attributes and splitting decisions. Even though C4.5 traditionally uses Gain Ratio, RapidMiner's implementation allows other choices. Several models were built and the model with the best accuracy used Gain Ratio. Table 30 shows the evaluation metrics for the TRAIN partition of the data set.

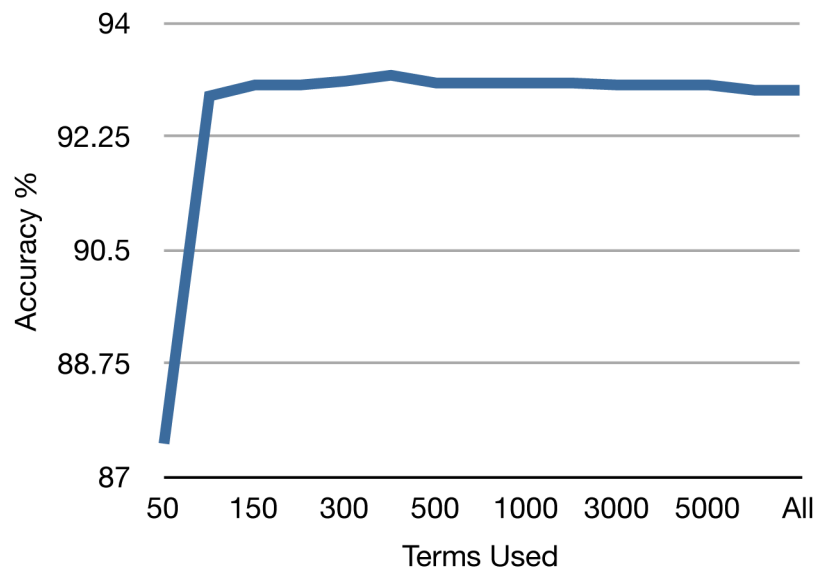
**Table 30:** Evaluation Metrics for Best Model All Terms (%)

<b>Metric</b>	<b>TRAIN</b>
Accuracy	93.0
Precision	81.7
Recall	89.5
Specificity	94.1
F-Measure	85.4
NPV	96.8

#### **Step 3b. DTI - Reduce Terms**

The next step in the model building process was to scale back the number of terms/attributes used in the model. With 6,250 unique concepts/terms extracted from the 5,009 documents, the term-by-document matrix would be quite extensive. The model building process would have to decide which of those 6,250 variables/attributes would be used in the decision nodes. By reducing the

dimensions of the matrix, the model building process is also reduced. Several models were constructed adjusting the number of attributes used {50, 100, 150, 200, 300, 400, 500, 1,000, 2,000, 3,000, 4,000, 5,000, 6,000, 6250}. To determine which terms to use, the STM process was adjusted. Before populating the term-by-document matrix vectors with binary occurrence values, gain ratio was used to determine the top terms from the documents. The matrix was then constructed using these top terms and then converted to binary occurrences, and the model with the best accuracy used 400 terms. Figure 15 shows the accuracy of the models starting with 50 terms and ending with using all of the terms. There is a relatively large jump in accuracy from 50 terms to 100 terms but after that, the change is minimal.

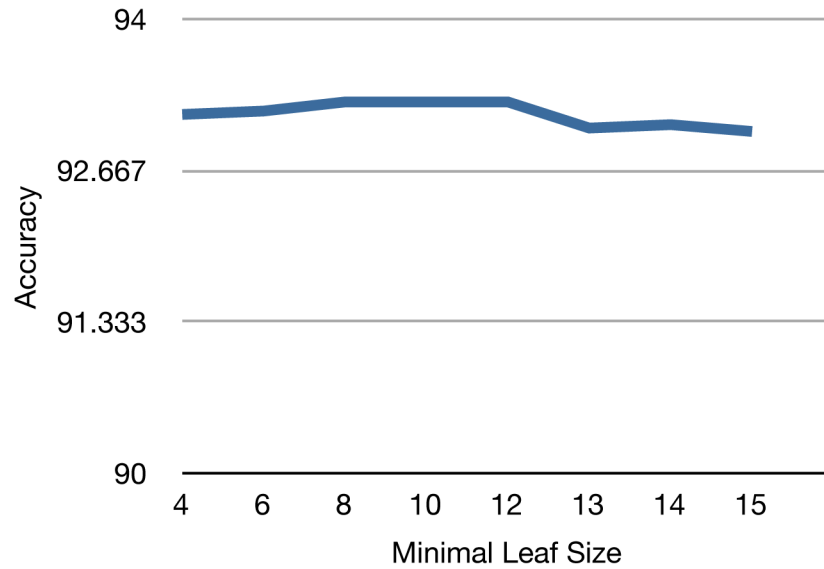


**Figure 15.:** Model Accuracy Based on Terms Used

### Step 3c. DTI - Adjust Minimum Leaf Size

To avoid over fitting the data, the minimal size of a leaf was then adjusted. The lowest leaf size tested was 4 and the largest 15. The results can be seen in figure 16. As can be seen from the figure, the difference in accuracy between the models is minimal. The best accuracy was achieved with a minimal leaf size of 8, 10, and 12. When the trees are pruned, the larger the leaf size the more aggressive pruning will be. The potential exists to remove more detail and more cases from

the tree by pruning leaves with 12 cases rather than 8 cases. It was for this reason that the model selected to use in the next step had a minimal leaf size of 8. Table 31 shows the evaluation metrics from this resultant model.



**Figure 16.:** Model Accuracy Based on Minimal Leaf Size

**Table 31:** Evaluation Metrics for Best Model Minimal Leaf (%)

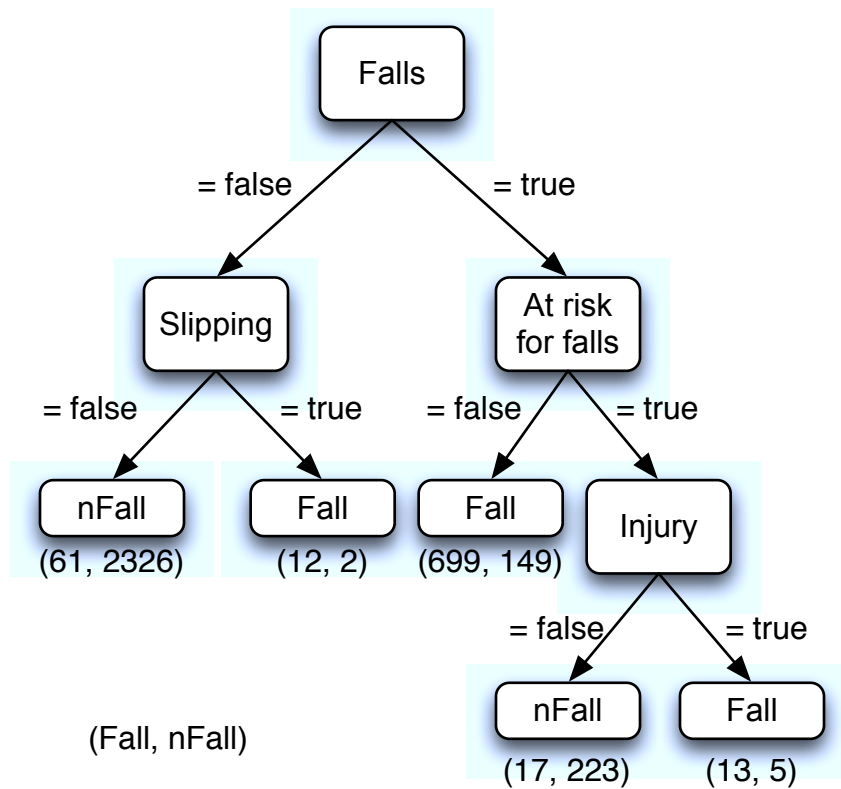
<b>Metric</b>	<b>TRAIN</b>
Accuracy	93.3
Precision	82.2
Recall	90.0
Specificity	94.2
F-Measure	86.0
NPV	97.0

### Step 3d. DTI - Prune

The model selected so far had the best accuracy of all of the models tested, however, the second goal of the research had not yet been addressed. Not only was the model supposed to be the most accurate but also be understandable or interpretable by a human. The resultant model had four decision nodes and five resulting production rules. In general, trees are made more understandable by pruning. The tree generated at this point was already understandable. Figure 17 shows the re-



sulting tree. The modeling process used the actual CUIs but in the figure, the concept terms are used to make it more easily read.



**Figure 17.:** 4 Level Decision Tree

The reason the tree was relatively small was based on the minimal leaf size decision made in the construction process. Even though pruning was not required on the tree that resulted from the classifier, RapidMiner allows for both pruning methods discussed earlier. The first is to start at the root and prune everything down from a specified level. The second is to prune from the bottom of the tree up.

The first method, pruning everything from a specified node down, was used to create a tree that was limited to a specified depth. Specifically, a one-level or “stump” tree was created using this method (Iba and Langley, 1992). This one-level tree split on the CUI representing “fall” and achieved 87.2% accuracy on TRAIN data. From there, levels were added to the tree and the evaluation metrics were collected. Table 32 shows some of these metrics for the models. The tree being evaluated only had four levels so no further levels could be added.

## DTI - Production Rules

As mentioned previously, the number of production rules corresponds to the number of leaf nodes. In the final pruned model there are five leaf nodes and thus five production rules. The rules follow:

- *If Falls = True and At risk for falls = True and Injury = True then class is Fall*
- *If Falls = True and At risk for falls = False then class is Fall*
- *If Falls = False and Slipping = True then class is Fall*
- *If Falls = False and Slipping = False then class is nFall*
- *If Falls = True and At risk for falls = True and Injury = False then class is nFall*

The first three rules could be provided to a clinician to enable them to gain some insight into the classifier.

**Table 32:** Pruned Trees Eval Metrics (%)

Levels	Accuracy	Prec	Recall	Specif
2	87.2	65.9	90.9	86.1
3	93.0	82.4	88.4	94.4
4	93.3	82.2	90.0	94.2

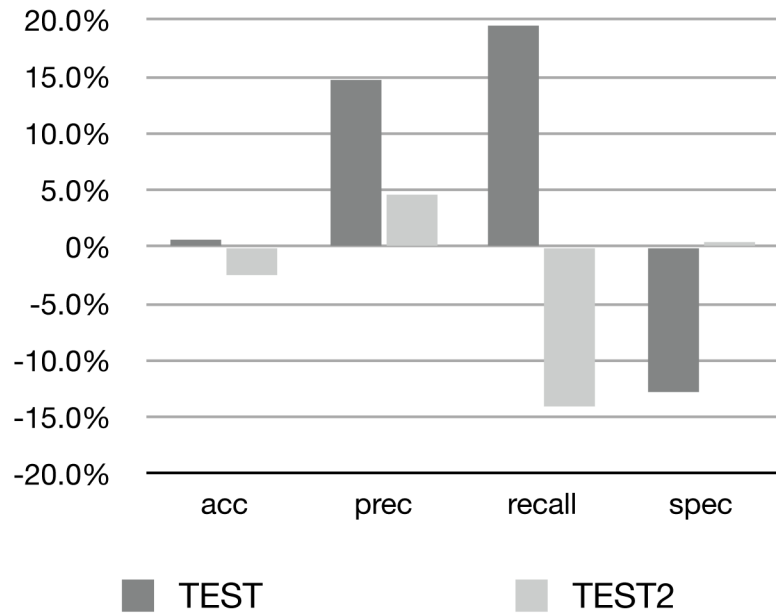
The second method, prune from the bottom up, is used to reduce the levels of a complex tree. Since the tree pruned with the first method is already relatively shallow, this method was not needed.

### 6.4.3 Results

After selecting the model that provided the best accuracy and pruning the resultant tree, the TEST data partition was then run against the model and the evaluation metrics were collected. In addition, another data set, created the same way, from three other medical centers was available from the prior research. The model created on TRAIN and evaluated against TEST was also evaluated against this other data set (TEST2). In the previous research that constructed the data sets, the final STM model trained on TRAIN and evaluated on TEST was also evaluated on TEST2. Table 33

**Table 33:** Evaluation Comparison of Different Data Sets (%)

Stat	TEST		TEST2	
	DTI	STM	DTI	STM
Accuracy	93.7	93.1	90.8	90.1
Precision	94.3	86.6	77.2	75.8
Recall	97.4	83.7	71.0	66.9
Specificity	82.6	96.0	95.3	95.2
NPV	91.6	95.0	93.6	92.8
F-measure	95.8	85.1	74.0	71.1



**Figure 18.:** % Difference Between Decision Tree and STM Classifiers

shows some of the evaluation metrics for the two classifiers on the two different data sets. Figure 18 graphically shows the percent difference between the two classifiers on the two data sets.

The baseline accuracy for this data (TEST and TRAIN) is 77.2%. In other words, if every document was classified as a nFALL document, 77.2% accuracy would be achieved. The accuracy of the final model was 93.7% as can be seen in table 25. A  $\chi^2$  test was performed to test the significance of the difference between the two results ( $\chi^2 = 164.4, p < 0.001$ ). Accuracy was the metric used to select the model along each step of the process. Other metrics could have been used but their values were such that the same model building decisions would have been made, resulting in the same final model.

There was no significance in the difference in the accuracy of the STM versus the DTI model for either TEST ( $\chi^2 = 0.4, p > 0.1$ ) or TEST2 ( $\chi^2 = 0.03, p > 0.1$ ). However, the resultant model from the DTI process was much more interpretable, where as the model from the STM process would be nearly impossible for a human to interpret.

## **6.5 Conclusions**

### **6.5.1 Error Analysis**

One of the main sources of error is the misspelling of “fall” and “fell.” “Fall” was misspelled as “feel”, “fal”, and “feal.” A human can read the note and tell by the context that “fall” was the correct term even though the clinician mistyped the term. In many cases a spell checker would not help because even though they misspelled “fall”, the misspelling was a correct word, such as “feel.” This accounted for a large portion of the percent difference between the STM and the DTI classifiers. The STM model can identify other patterns in addition to “fall.” That way, when “fall” is misspelled, those other patterns enable the STM model to still correctly classify a case.

The primary goal of this study was to identify documentation of the treatment for a fall-related injury during that episode of care. Documents were classified by clinicians as FALL if a fall was the mechanism of the patient’s injury. In other words, if the document said, “patient fell and injured his ankle yesterday,” that document was classified as a FALL document. Just the mere mention of the word fall did not guarantee a note would be classified as FALL. For example, if a document contained, “patient has fallen several times in the past year and is here today because she bumped her head getting out of the car,” that document was classified an nFALL because the

mechanism of her injury was not a fall even though it clearly states she has a history of falling. This category of documents, nFALL using fall terms, was the source of many Type I errors. In the TEST partition, of the 66 false positives, 30 of the cases fell into this category of annotations.

A few of the false positives were due to motor vehicle accidents being incorrectly classified by the model as a fall. In one note, the patient was "...thrown from a golf cart and fell on the ground." Another note stated, "... the patient fell off his motorcycle." When the annotation schema was created, it was agreed upon not to classify motor vehicle accidents as falls.

Some of the human classifications were based on judgment calls. A portion of the cases considered judgment call classifications contain a form of the term "syncope", the medical term for fainting. The annotators made the judgment that if a person suffered a syncope spell and was found on the floor, they fell when they lost consciousness. As can be seen from the decision tree, the term syncope does not occur frequently enough to be included in the tree. Most of these cases fell into the path down the left side of the tree in figure 17. The rule for this path was *If Falls = False and Slipping = False then class is nFall* and this rule resulted in the largest number of false negatives.

Some of the false positives were due to the term "fall" existing in the note but the term had a different meaning. For example, some notes contained, "...patient fell asleep..." or "... patient was seen by Dr XX last fall..." Improving the NLP pipeline, specifically by adding a more effective word sense disambiguation component would help in classifying these cases.

### **6.5.2 Future Work**

This is but one method of creating a rule-based classifier. Other methods include handcrafting rules as well as creating automated classification rules. Future work will examine these other methods and compare the overall accuracy as well as an economic analysis examining costs for these different methods.

For this initial research, classifying fall-related notes was a fairly straightforward target. As mentioned previously, just splitting on the concept "Falls" accurately classifies 87% of the cases. Now that we have a methodology for creating decision trees, it needs to be further tested on more complex medical targets.

The accuracy of the model is strongly dependent on the accuracy of the training data. In this

case the classification of the training data is adequate, however, the accuracy of the NLP process used to extract the CUIs can be improved.

Binary term occurrence was used to populate the term-by-document matrix vectors for this research. Other weighting schemes exist and need to be explored to determine if they help in creating a better classifier.

### **6.5.3 Concluding Remarks**

Currently, there is an initiative to implement EHR systems across the US. The VA Veterans Health Administration is the largest health care system in the US and exclusively uses an EHR and has for several years. Being able to parse through these large amounts of data and being able to provide clinicians information from the EHR is the next step in this initiative. CDSSs not only require accuracy but also a human interpretable classifier for clinicians to take full advantage of the system. Other classifying methods such as Support Vector Machines (SVMs) can be used and are especially useful where large variable sets exist, however, the resultant predictive model can seldom be interpreted by a human (Burges, 1998). One of the benefits of creating a rule-based classifier is that the rules themselves are interpretable. The method evaluated in this research is an approach that used NLP, STM, and DTI technologies to create an accurate, rule-based classifier.

## Chapter 7

### Conclusion

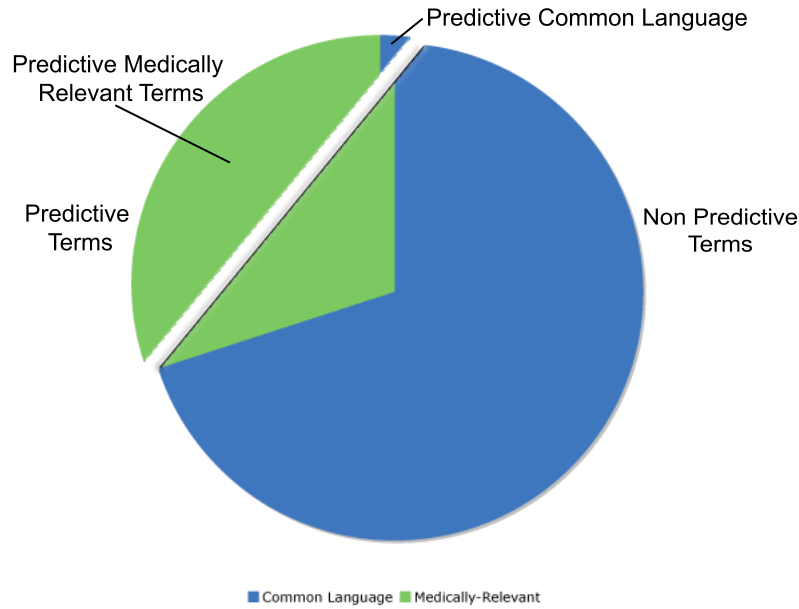
This dissertation has two main research questions.

- Can information (specialized language) be extracted from clinical progress notes that will represent the notes without loss of predictive information?
- Can classifiers be built for clinical progress notes that are represented by specialized language?

These questions were addressed in three related experiments. The first experiment addresses the first research question by separating the specialized language in a progress note from the common language and then building a classifier to answer the question. The second research question is addressed by two approaches, each of which use the resulting data set of specialized language from the first experiment. These two approaches also address the second question by building rule-based classifiers.

#### 7.1 Research Question 1

The first experiment developed a grammatical NLP process to extract the medically relevant terms from medical progress notes and then used STM to build a classifier to show that there was no loss of predictive information in separating the medically relevant terms from the common language. This was the main contribution of this experiment and it informs the two approaches that address research question two. Because of the importance of this finding, two figures from that experiment have been included again in this chapter. Figure 19 shows the breakdown of medically relevant terms compared to common language. The important idea this figure presents is that progress notes are comprised of medically relevant terms and common language and the ratio for

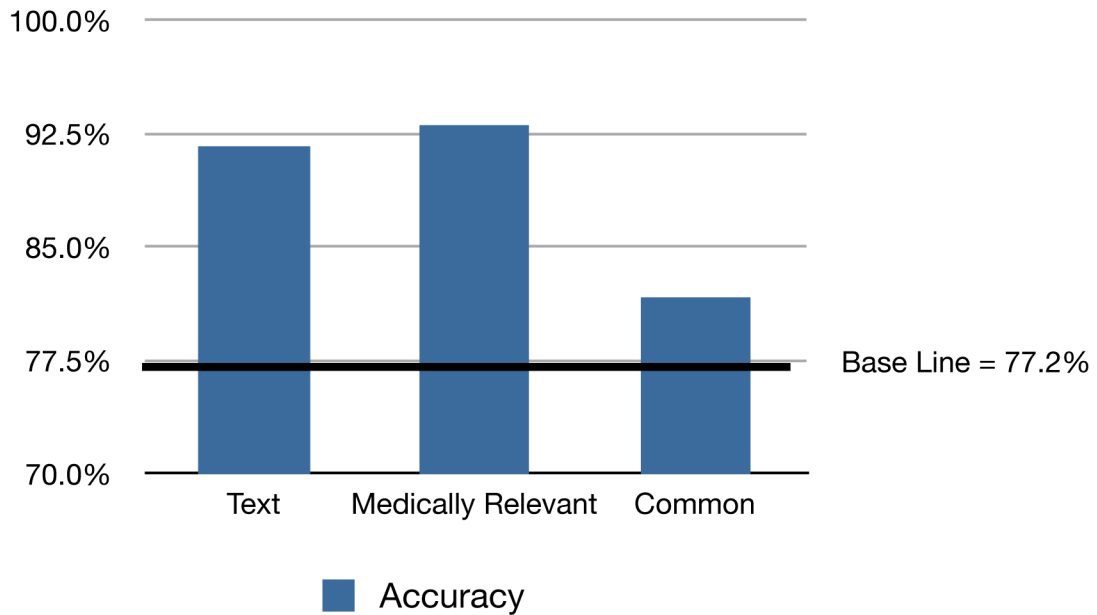


**Figure 19.:** Breakdown of Terms

this particular data set is approximately 1:3. Even more importantly, the majority of predictive information is contained within the medically relevant terms, with a very small amount of common language that also contains some predictive information, labeled as Predictive Common Language in the figure. The medically relevant terms in the pie chart (figure 19) reflect the extent of coverage in the controlled vocabulary. This is certainly domain specific and in some ways is a measure of vocabulary quality.

The second figure, figure 20, shows the accuracy of the STM classifiers. Three important points are shown with this figure. First, the classifier using just the medically relevant terms was more accurate on this data than either the entire text from the progress notes or the common language from those same notes. This suggests that accurate classifiers can be built using just the medically relevant terms extracted from medical progress notes. This also suggests that by extracting just the medically relevant terms that perhaps some noise may also be removed from the data. Secondly, is that a very small amount of common language from the previous figure, labeled as Predictive Common Language, suggests that there is some predictive information in the common language. The results in figure 20 support this assertion. The accuracy of the classifiers using any of the three data partitions is better than the baseline of 77.2%. The small wedge in figure 19 matches to the part of the common language bar between 77.2% and 81.7% in figure 20. Thirdly,





**Figure 20.:** Accuracy of Text, Medically Relevant Terms, and Common Language

even though the common language contains some predictive information, no loss in classifier accuracy occurred by removing this information. Excluding this predictive information contained in the common language does not reduce the performance of the model but instead actually increases the performance of the model. This is supported in that the accuracy of classifier using medically relevant terms is slightly higher than the accuracy of the complete text from the notes. This also supports the idea that removing the common language also removes some noise. The results of this experiment were key to informing the remaining two. The next two experiments address the second research question concerning rule-based classifiers.

## 7.2 Research Question 2

Research question 2 was addressed by two separate approaches. The first approach used a modified association rule mining process to create a rule-based classifier. The second approach used a decision tree induction process to create a rule-based classifier. The first experiment used logistic regression to build the classifier and within this process, LSA was used as a data reduction technique. Logistic regression models themselves may be difficult to interpret based on the number of variables in the model but with the addition of LSA vectors used as variables, which them-

selves are not intended to be interpreted, the resultant model was accurate but was not easily interpretable.

### **7.2.1 Approach 1: Classification Rule Mining**

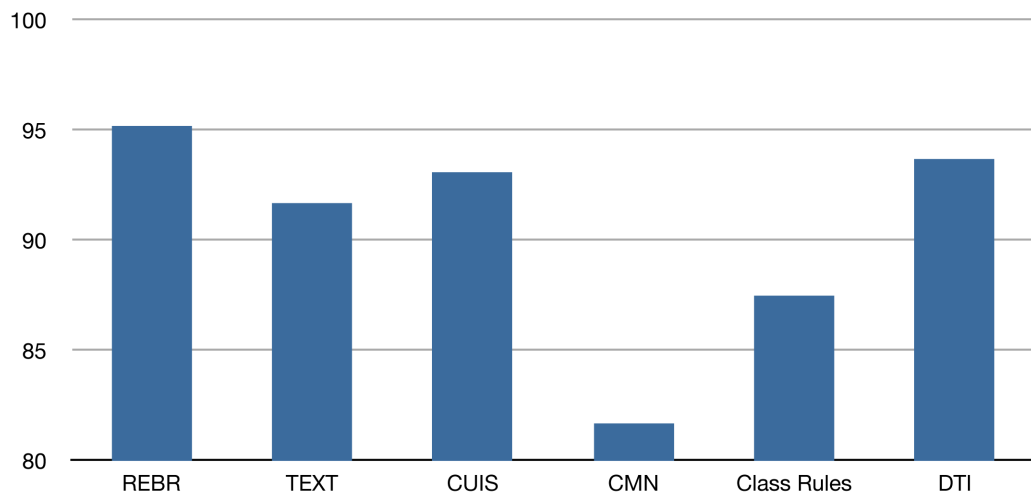
In the first approach, a hybrid approach including NLP, STM, association rule mining, and contrast sets was created in order to build a rule-based classifier. The goal of this experiment was to not only create an accurate classifier but to also create a classifier that was interpretable. The methodology used to create the rule-based classifier was the main contribution of this approach. This included using the NLP pipeline from the first experiment to extract medically relevant terms and using STM to create a term-by-document matrix to apply structure to the textual data, as well as creating frequent item sets, creating classification rules, and examining contrast sets. Unique in this approach was the way the classification rules were created. Typically, rules are built from a complete data set, ignoring any classification, meeting a specified confidence from frequent item sets that meet a specified support. What made this approach unique was that the data set was split into two partitions based on the classification. The rules were then generated for each classification partition separately and the consequent used was the actual class to be predicted. As a result, two sets of rules were generated: one for FRIs and the other for not FRIs. The hybrid approach used in this experiment was our first attempt at creating a classifier that was interpretable and even though the approach succeeded at that, the overall classifier accuracy was not as high as the logistic regression classifier from the first experiment. We had hoped to come closer to that accuracy. The second approach was an attempt at just that.

### **7.2.2 Approach 2: Decision Tree Induction**

The second approach is informed by both previous studies and builds a rule-based classifier. For this experiment, DTI was used and the resultant decision tree provided interpretable rules and better predictive accuracy than both the classification rule model from approach one and the logistic regression model from the first experiment. The main contribution of this experiment was again the methodology used to create the DTI classifier. This included using from the first experiment the NLP pipeline to extract medically relevant terms and using STM to create a term-by-document matrix to apply structure to the textual data, then a process for creating an accurate, pruned deci-

sion tree. A compact, accurate decision tree was created that provided five production rules that were interpretable.

Figure 21 shows the accuracy of all of the models tested in this dissertation. To reiterate the three experiments, experiment one created a logistic regression model using the complete notes (TEXT), the extracted medically relevant terms or specialized language (CUIS), and the common language (CMN). Experiment two created classification rules based classifier (Class Rules), and experiment three used decision tree induction to create a classifier (DTI). Both the figure and the table also include a model from an unpublished study in which regular expression based rules (REBR) were created by a human and tested against the same progress notes. Table 34 shows the results from the models from the three experiments comprising this dissertation as well as from the unpublished regular expression based rules study.  $\chi^2$  was used to test significance between the accuracy of each individual model built from an NLP data set compared to the accuracy of the complete textual notes model. The only model that was not significantly different was the model created against the specialized language (CUIS). The p-value for this test was 0.1489.



**Figure 21.:** Graph of accuracy for all models

### 7.3 Other Contributions

The studies in this dissertation also had some unplanned, secondary contributions. First is that by extracting the medically relevant terms, the PHI is also removed from the notes. Since there was

**Table 34:** Results from all studies ( $*p < 0.001$ )( $**p < 0.05$ )

Experiment		Accur	F-Meas	Precis (PPV)	Specif	Recall (Sens)	NPV	$\chi^2$
	REBR	95.2	90.2	84.8	94.9	96.2	98.8	14.1*
Exp 1	TEXT	91.7	81.9	82.2	94.7	81.5	94.5	
	CUIS	93.1	86.6	86.6	96.0	83.7	95.0	2.1
	CMN	81.7*	56.7	52.5	86.5	61.6	90.3	64.9*
Exp 2	Class Rules	87.5	74.0	70.4	90.3	77.8	93.2	15.3*
Exp 3	DTI	93.7	95.8	94.3	82.6	97.4	91.6	4.4**

no loss in predictive information, this turns out to be an excellent method to de-identify medical progress notes to be used in machine learning tasks. In fact, the notes can be turned into nothing but a sequence of concept identifiers, so that even the wording can be masked.

Secondly, the approach developed here includes a natural data reduction technique. In the data set used in this dissertation, the notes were made up of approximately 26% medically relevant terms and 74% common language. Extracting the medically relevant terms turns out to be an excellent data reduction technique. The amount of space needed to store just the terms is nearly 75% less than what is needed to store the entire set of progress notes. Not only is storage reduced but also processing time. Since the terms are a quarter of the original notes, there is much less to parse and execute the model against. In future work, a study needs to be conducted to examine the economic ramification of this data set reduction technique.

#### 7.4 Future Work

While each experiment addresses future work specific to that experiment, there are also some areas of future work that apply in general to this stream of research. Of primary concern is the selection of a medical topic and targets for any machine learning efforts. We assume the accuracy rates of the models in this dissertation are in part due to the low complexity of the task. First of all, it is a binary classification and secondly, as can be witnessed by the decision tree, just the concept “fall” accurately predicts approximately 87% accurately. This idea of prediction based on medically relevant terms needs to be tested on other more complex medical topics.

All three experiments constructed models that classified a single progress note as either FRI

or NOT FRI. The next step is to show that this can be introduced into either a surveillance or a clinical decision support system. In these cases it will be important to show that classifications can be rolled up from the note level to either the episode of care or to the patient level.

#### **7.4.1 Ontology Development**

In the research community, there is a lack of agreement as to what an ontology is. Regardless, most will agree that in its basic definition, an ontology represents knowledge. Because medical informatics is currently a hot topic, there is a drive for creating ontologies. The knowledge gleaned from the combination of these experiments can be used to inform the ontology building process. The first experiment results in an extraction of medically relevant terms, pertinent to the medical topic of interest. This resulting medical “vocabulary” can be used to aid clinicians in researching terms pertinent to the medical topic of interest. The second and third experiments resulted in interpretable rules. These rules can also be used to aid clinicians in discovering the relationships that exist between the terms in the vocabulary used. We are not proposing using these processes as an automated way to build an ontology but rather as an automated way to provide information to clinicians interested in building ontologies. The idea of using the results from this dissertation as an aid to ontology development is both future work and a contribution.

#### **7.4.2 NLP**

The medically relevant terms extracted are only as good as the NLP pipeline. Medical progress notes provide issues that other sources of text do not have. First, is sentence detection. Clinicians’ time is valuable and after spending time with patients, they have a small amount of time to document visits or treatments. Because of this, clinicians type more fragments than grammatically correct sentences. An entire stream of research exists to improve sentence detection within medical progress notes. As these sentence detectors become available, they need to be tested in our pipeline.

#### **WSD**

As mentioned previously, there is an important issue related to word sense disambiguation (WSD). The error analysis in the third experiment brought to the surface the many uses of the word *fall*.

One can fall asleep or be treated in the fall of the year, neither of which have anything to do with a FRI. Just as with sentence detection, there is a community of researchers addressing the WSD issue within health care.

### **Acronym Detection**

Similar to WSD is acronym detection. Acronyms are plentifully used in business and technology but even more so in health care related to the military. Not only are there many acronyms but many of them have multiple meanings depending on the medical topic. Most lay people understand *BP* to mean *Blood Pressure* but it also stands for *Binding Peptide*, *Bronchial Provocation*, *Bleomysin/cisplatin Protocol* and many other meanings. Improved WSD would also improve the accuracy of the NLP pipeline.

### **Templates**

An important issue within VistA, the VA open source EHR, is the use of templates. Clinicians enter textual notes or unstructured data into the EHR. Templates help apply structure to unstructured notes but they cause problems for a computer processing them. A template is text stored in the system that is pasted into a text field and allows the clinician to enter information without having to type all of the information. For example, a template might contain yes/no checklists pertinent to a clinical area.

*Heart Attack? Yes*

*Stroke? No*

*High Blood Pressure? Yes*

The template with the questions is pasted into the text area and clinicians enter *yes* or *no* as appropriate. The problem goes back to the sentence detection. If it uses the “?” as a sentence boundary, the negation module will apply “No” to *High Blood Pressure* rather than *Stroke* as was intended. If it uses end of line markers, then the negation in this example will be correctly applied, however, the processing of free text areas will be incorrect. Researchers are also currently working to improve template detection.

## **Cascading Errors**

Another area that needs to be addressed for this research stream is how do errors that occur in the NLP pipeline effect steps further in the process? The results of the models suggest that in spite of the cascading effect of error, or the introduction of noise, the processes still build accurate models. This is evidenced by the results in experiments one and three since the accuracy as well as other evaluation metrics was higher for these models than was the classification of the unprocessed textual notes.

## **7.5 Final Discussion**

The goal of this dissertation was to use AI technologies such as grammatical NLP, STM, and data mining to address two research questions. First, can information be extracted from clinical progress notes that will represent the notes without loss of predictive information? And secondly, can rule-based approaches be used to build classifiers that will accurately classify progress notes based on the extracted medically relevant terms while at the same time creating rules that can be understood and interpreted by a person? Three separate experiments helped address these questions, with the last two building off of the first experiment. These three experiments showed that accurate, rule-based classifiers can be created for data sets consisting of progress notes reduced to only medically relevant terms. There are still questions to be answered in this area, but these experiments lay the ground work for a future research stream combining NLP, STM, and data mining as tools for better understanding the electronic health record.

## References

2005. ANSI/NISO Z39.19-2005 Guidelines for the Construction, Format, and Management of Monolingual Controlled Vocabularies 1–184.
2011. HHS website on Health Information Privacy. website. URL <http://www.hhs.gov/ocr/privacy/>. Retrieved 04/04/2011.
2011. Taxonomy Warehouse Website. website. URL <http://taxonomywarehouse.com/index.asp>. Retrieved 04/04/2011.
- Agrawal, R., Imielinski, T., and Swami, A. 1993. Mining Association Rules Between Sets of Items in Large Databases. *ACM SIGMOD Record*. **22**(2) 207–216.
- Agrawal, R. and Srikant, R. 1994. Fast algorithms for mining association rules. *Proc. 20th Int. Conf. Very Large Data Bases* 1–32.
- Alberts, L.K. 1993. Ymir: An ontology for engineering design. Ph.D. thesis, University of Twente, 7500 AE Enschede, The Netherlands.
- Albright, R. 2004. Taming Text with the SVD. *SAS White paper* 1–17.
- Apache Software Foundation. 2009. website. URL <http://uima.apache.org>. Retrieve 8/23/2009.
- Applegate, L.M. 2002. Rigor and Relevance in MIS Research Introduction. *MISQ*. **23**(1) 2.
- Apte, C. and Damerau, F. 1997. Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information and Systems*. **12**(3) 233–251.
- Armstrong, D., Kline-Rogers, E., Jani, S., Goldman, E., Fang, J., Mukherjee, D., Nallamothe, B., and Eagle, K. 2005. Potential Impact of the HIPAA Privacy Rules on Data Collection in a Reg-



- istry of Patients with Acute Coronary Syndrome. *Archive of Internal Medicine*. **165**(10) 1125–1129.
- Aronson, A.R. 2001. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: the MetaMap Program. *Proceedings of the AMIA Symposium* 17–21.
- Austin, J.L. 1962. *How to do Things with Words*. Harvard University Press, Cambridge, MA.
- Baldini, N., Neri, F., and Pettoni, M. 2007. A Multilanguage Platform for Open Source Intelligence. *Eighth International Conference on Data, Text and Web Mining and their Business Applications* 325–334.
- Bateman, J.A. 1990. Upper Modeling: A General Organization of Knowledge for Natural Language Processing. *Proceedings of the International Language Generation Workshop* 1–15.
- Bay, S.D. and Pazzani, M.J. 2001. Detecting Group Differences: Mining Contrast Sets. *Data Mining and Knowledge Discovery*. **5** 213–246.
- Beckwith, B.A., Mahaadevan, R., Balis, U.J., and Kuo, F. 2006. Development and Evaluation of an Open Source Software Tool for Deidentification of Pathology Reports. *BMC Medical Informatics and Decision Making*. **6**(12) 1–10.
- Bird, S. 2003. Phonology. R. Mitkov, ed., *The Oxford Handbook of Computational Linguistics*. Oxford University Press, New York, NY, 3–24.
- Bollen, J., Mao, H., and Zeng, X. 2011. Twitter Mood Predicts the Stock Market. *Journal of Computational Science*. **2** 1–8.
- Bradford, J.P., Kunz, C., Kohavi, R., Brunk, C., and Brodley, C.E. 1998. Pruning Decision Trees with Misclassification Costs. *Lecture Notes in Computer Science*. **1398** 131–136.
- Breiman, L., H., F.J., Olshen, R.A., and Stone, C.J. 1984. *Classification and Regression Trees*. Wadsworth, Belmont, CA.
- Bresnan, J. 1976. Evidence for a Theory of Unbounded Transformations. *Linguistic Analysis*. **2** 353–393.
- Bruner, J.S., Goodnow, J.J., and Austin, G.A. 1956. *A Study of Thinking*. Wiley, New York, NY.

- Burges, C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*. **2** 121–167.
- Chaplan, M.A. 1995. Mapping” Laborline Thesaurus” Terms to Library of Congress Subject Headings: Implications for Vocabulary Switching. *The Library Quarterly*. **65**(1) 39–61.
- Chapman, W.W., Bridewell, W., Hanbury, P., Cooper, G.F., and Buchanan, B.G. 2001. A Simple Algorithm for Identifying Negated Findings and Diseases in Discharge Summaries. *Journal of Biomedical Informatics*. **34** 301–310.
- Chen, Y., Perl, Y., Geller, J., and Cimino, J. 2007. Analysis of a Study of the Users, and Future Agenda of the UMLS. *Journal of the American Medical Informatics Association*. **14**(2) 221–231.
- Chomsky, N. 1957. *Syntactic Structures*. Mouton, Paris, France.
- Cimino, J., Hripcsak, G., Johnson, S., and Clayton, P.D. 2008. Designing an Introspective, Multi-purpose, Controlled Medical Vocabulary. *Proc Annu Symp Comput Appl Med Care* 513–518.
- Cocke, J. 1969. *Programming Languages and Their Compilers: Preliminary Notes*. Courant Institute of Mathematical Sciences, New York, NY.
- Cohen, A.M. and Hersh, W.R. 2005. A Survey of Current Work in Biomedical Text Mining. *Briefings in Bioinformatics*. **6**(1) 57–71.
- Cohen, J. 2005. Weighted Kappa: Nominal Scale Agreement with Provision for Scaled Disagreement or Partial Credit. *Psychological Bulletin*. **70**(4) 213–220.
- Cohen, J. 2006. A Coefficient of Agreement for Nominal Scales. *Educational and psychological measurement*. **20**(1) 37–46.
- Cortes, C. and Vapnik, V. 2011. Support-Vector Networks. *Machine Learning*. **20** 273–297.
- Cunningham, H. 2002. GATE, a General Architecture for Text Engineering. *Computers and the Humanities*. **36** 223–254.

- Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. 2002. GATE: an Architecture for Development of Robust HLT Applications. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* 168–175.
- Darwin, C. 1859. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. John Murray, London.
- Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., and Harshman, R. 2010. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*. **41**(6) 391–407.
- Dumais, S. 1998. Using SVMs for Text Categorization. *IEEE Intelligent Systems*.
- Elkin, P., Brown, S., Husser, C., Bauer, B., Wahner-Roedler, D., Rosenbloom, T., and Speroff, T. 2006. Evaluation of the Content Coverage of SNOMED CT: Ability of SNOMED Clinical Terms to Represent Clinical Problem Lists. *Mayo Clinic Proceedings*. **81**(6) 741–748.
- Eom, J.H., Kim, S.C., and Zhang, B.T. 2008. AptaCDSS-E: A classifier ensemble-based clinical decision support system for cardiovascular disease level prediction. *Expert Systems with Applications*. **34** 2465–2479.
- Fernandez, R., Ginzburg, J., Gregory, H., and Lappin, S. 2011. SHARDS: Fragment Resolution in Dialogue. *Proceedings of the 4th International Workshop on Computational Semantics* 156–172.
- Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A.A., Lally, A., McDermott, J.W., Nyberg, E., Prager, J., Schlaefer, N., and Welty, C. 2010. Building Watson: An overview of the DeepQA project. *AI Magazine* 59–79.
- Ferrucci, D. and Lally, A. 2004. UIMA: an Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering*. **10**(3/4) 327–348.
- Foley, R. 2011. Text Analytics: How to Capture the Online Bully. website. URL <http://www.b-eye-network.com/view/15001>.

- Freitas, A.A. 2000. Understanding the Crucial Differences Between Classification and Discovery of Association Rules: a Position Paper. *ACM SIGKDD Explorations Newsletter*. **2**(1) 65–69.
- Friedman, C., Shagina, L., Lussier, Y., and Hripesak, G. 2004. Automated Encoding of Clinical Documents Based on Natural Language Processing. *Journal of the American Medical Informatics Association*. **11**(5) 392–402.
- Friedman, J.H. 1977. A Recursive Partitioning Decision Rule for Nonparametric Classification. *IEEE Transactions on Computers*. **C-26**(4) 404–408.
- Ganz, D.A., Bao, Y., Shekelle, P.G., and Rubenstein, L.Z. 2007. Will My Patient Fall? *JAMA*. **297**(1) 77–86.
- Gazdar, G., Klein, E.H., Pullum, G.K., and Sag, I.A. 1985. *Generalized Phrase Structure Grammar*. Harvard University Press, Cambridge, MA.
- Gennari, J.H., Musen, M.A., Fergerson, R.W., Grosso, W.E., Crubezy, M., Eriksson, H., Noy, N.F., and Tu, S.W. 2003. The Evolution of Protege: An Environment for Knowledge-Based Systems Development. *International Journal of Human Computer Studies*. **58** 89–123.
- Geurts, P., Fillet, M., De Seny, D., Meuwis, M.A., Malaise, M.P., and Wehenkel, L. 2005. Proteomic Mass Spectra Classification Using Decision Tree Based Ensemble Methods. *Bioinformatics*. **21**(15) 3138–3145.
- Goclenius, R. 1613. *Lexicon Philosophicum*. Frankfurt.
- Gold, E.M. 2011. Language Identification in the Limit. *Information and Control*. **10** 447–474.
- Haussler, D. 2011. Overview of the Probably Approximately Correct (PAC) Learning Framework 1–78.
- Hayes, P.J., Andersen, P.M., Nirenburg, I.B., and Schmandt, L.M. 1990. TCS: A Shell for Content-Based Text Categorization. *Sixth Conference on Artificial Intelligence for Applications*. IEEE Comput. Soc. Press, 320–326.
- Hayes, P.J. and Weinstein, S.P. 1990. CONSTRUE/TIS: A System for Content-Based Indexing of a Database of News Stories. *Proceedings of the Second Annual Conference on Innovative Applications of Artificial Intelligence*.

- Hearst, M.A. 1999. Untangling Text Data Mining. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics* 3–10.
- Hevner, A.R. and March, S.T. 2003. The Information Systems Research Cycle. *IT Systems Perspectives* 3.
- Hirschman, L., Morgan, A.A., and Yeh, A.S. 2002. Rutabaga by Any Other Name: Extracting Biological Names. *Journal of Biomedical Informatics*. **35** 247–259.
- Holton, C. 2009. Identifying Disgruntled Employee Systems Fraud Risk Through Text Mining: a Simple Solution for a Multi-Billion Dollar Problem. *Decision Support Systems*. **46** 853–864.
- Humphreys, B.L., Lindberg, D.A., Schoolman, H.M., and Barnett, G.O. 1998. The Unified Medical Language System An Informatics Research Collaboration. *Journal of the American Medical Informatics Association*. **5**(1) 1–11.
- Hunt, E.B., Marin, J., and J, S.P. 1966. *Experiments in Induction*. Academic Press, New York, NY.
- Iba, W. and Langley, P. 1992. Induction of One-Level Decision Trees. *Proceedings of the Ninth International Workshop on Machine Learning* 1–8.
- Jarman, J. and Berndt, D.J. 2010. Throw the Bath Water Out, Keep the Baby: Keeping Medically-Relevant Terms for Text Mining. *AMIA 2010 Annual Symposium Proceedings* 1–5.
- Jarman, J. and Berndt, D.J. 2011. Combining Natural Language Processing and Statistical Text Mining: Classifying Fall-Related Progress Notes . *Proceedings of the 2011 VA HSR&D Conference*.
- Kalafatis, T. 2010. Social media insights from predictive analytics.
- Kaplan, R.M. 2003. Syntax. R. Mitkov, ed., *The Oxford Handbook of Computational Linguistics*. Oxford University Press, New York, NY, 70–90.
- Kasami, T. 1965. An Efficient Recognition and Syntax Analysis Algorithm for Context-free Languages. Tech. rep., Bedford, MA.
- Kearns, M.J. and Vazirani, U.V. 1994. *An Introduction to Computation Learning Theory* . MIT Press, Cambridge, MA.

- Kipper-Schuler, K., Kaggal, V., Ogren, P., and Savova, G.K. 2008. System Evaluation on a Named Entity Corpus from Clinical Notes. *Proceedings of 2008 LREC* 3007–3011.
- Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I. 1994. Finding Interesting Rules From Large Sets of Discovered Association Rules. *Proceedings of the Third International Conference on Information and Knowledge Management*.
- Kucera, H. and Francis, W.N. 1967. *Computational Analysis of Present Day American English*. Brown University Press, Providence, RI.
- Kullback, S. and Leibler, R.A. 1951. On information and sufficiency. **22**(1) 79–86.
- Kurt, I., Ture, M., and Kurum, A.T. 2008. Comparing Performances of Logistic Regression, Classification and Regression Tree, and Neural Networks for Predicting Coronary Artery Disease. *Expert Systems with Applications*. **34** 366–374.
- Lack, D. 1940. Evolution of the Galapagos Finches. *Nature*. **146** 324–327.
- Ladefoged, P. 2001. *A Course in Phonetics*. Hartcourt College Publishers, Fort Worth.
- Lappin, S. 2003. Semantics. R. Mitkov, ed., *The Oxford Handbook of Computational Linguistics*. Oxford University Press, New York, NY, 91–111.
- Leech, G. and Weisser, M. 2003. Pragmatics and Dialogue. R. Mitkov, ed., *The Oxford Handbook of Computational Linguistics*. Oxford University Press, New York, NY, 136–156.
- Li, W., Han, J., and Pei, J. 2001. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. *IEEE International Conference on Data Mining 2001 Proceedings* 369–376.
- Library of Congress. 2011. website. URL <http://id.loc.gov/authorities/>. Retrieved 04/04/2011.
- Liu, B., Hsu, W., and Ma, Y. 1998. Integrating Classification and Association Rule Mining. *KDD-98 Proceedings* 1–7.
- Lorhard, J. 1613. *Theatrum Philosophicum*. Basilea.

- Lovins, J. 1968. Development of a Stemming Algorithm . Tech. rep.
- Luther, S.L., French, D., Powell-Cope, G., Rubenstein, L., and Campbell, R. 2005. Using Administrative Data to Track Fall-Related Ambulatory Care Services in the Veterans Administration Healthcare System . *Aging Clin and Exp Res.* **17**(5) 412–418.
- Mansour, Y. 1997. Pessimistic Decision Tree Pruning Based on Tree Size. *Proceedings of the ICML* 1–7.
- Mehta, M., Riassanen, J., and Agrawal, R. 1995. MDL-Based Decision Tree Pruning. *Knowledge Discovery and Data Mining (KDD'95)* 216–221.
- Michalski, R. 1990. Concept Learning. *Encyclopedia of Artificial Intelligence*. Wiley-Interscience Publication, New York, NY, 185–194.
- Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., and Euler, T. 2006. YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* 935–940.
- Mitchell, K.J., Becich, M.J., Berman, J.J., Chapman, W.W., Gilbertson, J., Gupta, D., Harrison, J., Legowski, E., and Crowley, R.S. 2004. Implementation and Evaluation of a Negation Tagger in a Pipeline-Based System for Information Extraction from Pathology Reports. *Medinfo* 663–667.
- Munn, K. and Smith, B. 2008. *Applied Ontology*. An Introduction, Transaction Books, Piscataway, NJ.
- Murphy, C.K. 2001. Identifying Diagnostic Errors with Induced Decision Trees. *Medical Decision Making.* **21** 368.
- Musen, M.A., Fagan, L., Combs, D.M., and Shortliffe, E.H. 1987. Use of a domain model to drive an interactive knowledge-editing tool. *International Journal of Man-Machine Studies.* **26** 105–121.
- National Library of Medicine. 2010.
- Neamatullah, I., Douglass, M., Lehman, L., Reisner, A., Villarroel, M., Long, W.J., Szolovits, P., Moody, G., Mark, R.G., and Clifford, G.D. 2008. Automated De-identification of Free-Text Medical Records. *BMC Medical Informatics and Decision Making.* **8**(32) 1–17.

- Ness, R.B. 2007. Influence of the HIPAA Privacy Rule on Health Research. *Journal of the American Medical Association*. **298**(18) 2164–2170.
- Norton, J.D. 2010. A Survey of Inductive Generalization 1–144.
- Ogren, P.V. 2006. Knowtator: A Plug-in for Creating Training and Evaluation Data Sets for Biomedical Natural Language Systems. *9th International Protege Conference* 1–4.
- Ogren, P.V., Savova, G.K., and Chute, C.G. 2008. Constructing Evaluation Corpora for Automated Clinical Named Entity Recognition. *Proceedings of the Language Resources and Evaluation Conference* 3143–3150.
- Olanow, C.W. and Koller, W.C. 2007. An Algorithm (Decision Tree) for the Management of Parkinson's Disease. *Neurology*. **50**(Suppl 3:S1) 81.
- Pakhomov, S., Pedersen, T., and Chute, C.G. 2005. Abbreviation and Acronym Disambiguation in Clinical Discourse. *AMIA 2005 Symposium Proceedings*.
- Pollard, C. and Sag, I.A. 1988. *Information-based Syntax and Semantics: Vol 1: Fundamentals*. Center for the Study of Language and Information, Stanford, CA.
- Qu, Y., Adam, B.L., Yasui, Y., Ward, M.D., Cazares, L.H., Schellhammer, P.F., Feng, Z., Semmes, O.J., and Wright, G.L.J. 2002. Boosted Decision Tree Analysis of Surface-Enhanced Laser Desorption/Ionization Mass Spectral Serum Profiles Discriminates Prostate Cancer From Non-cancer Patients. *Clinical Chemistry*. **48**(10) 1835–1843.
- Quinlan, J. 1986. Induction of Decision Trees. *Machine Learning*. **1** 81–106.
- Quinlan, J. 1987. Generating Production Rules From Decision Trees. *Proceedings of the 10th International Joint Conference on Artificial Intelligence*. **1** 304–307.
- Quinlan, J. 1988. *Decision Trees and Multi-valued Attributes published in Machine Learning*. Oxford University Press, Oxford, UK.
- Quinlan, J. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.



- Rastogi, R. and Shim, K. 2000. Public: a Decision Tree Classifier That Integrates Building and Pruning. *Data Mining and Knowledge Discovery*. **4** 315–344.
- Savova, G., Kipper-Schuler, K., Buntrock, J.D., and Chute, C. 2008. UIMA-based Clinical Information Extraction System. *LREC Conf Proceedings 2008* 39–42.
- Searle, J. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, MA.
- Shannon, C.E. 1948. A Mathematical Theory of Communication. *The Bell System Technical Journal*. **27** 379–423, 623–656.
- Shannon, C.E. 1951. Prediction and Entropy of Printed English. *The Bell System Technical Journal* 50–65.
- Shannon, C.E. and Weaver, W. 1949. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, IL.
- Shell, A. 2011. Wall Street Traders Mine Tweets to Gain a Trading Edge . *USA Today*.
- Smallheiser, N.R. 2001. Predicting Emerging Technologies with the Aid of Text-Based Data Mining: The Micro Approach. *Technovation*. **21** 689–693.
- Smallheiser, N.R. and Swanson, D. 1998. Calcium-independent phospholipase A2 and schizophrenia. *Archives of General Psychiatry*. **55**(8) 752–753.
- Smallheiser, N.R. and Swanson, D. 2007a. Indomethacin and Alzheimer’s Disease. *Neurology*. **46** 583.
- Smallheiser, N.R. and Swanson, D. 2007b. Linking Estrogen to Alzheimer’s Disease: An Informatics Approach. *Neurology*. **47** 809–810.
- Sokolova, M. and Lapalme, G. 2009. A Systematic Analysis of Performance Measures for Classification Tasks. *Information processing & management*. **45**(4) 427–437.
- Swanson, D. 1986a. Fish Oil, Raynaud’s Syndrome, and Undiscovered Public Knowledge. *Perspective in Biology and Medicine*. **30**(1) 7–18.

- Swanson, D. 1986b. Undiscovered Public Knowledge. *The Library Quarterly*. **56**(2) 103–118.
- Swanson, D. 1987. Two Medical Literatures That Are Logically but not Bibliographically Connected. *Journal of the American Society for Information Science*. **38**(4) 228–233.
- Swanson, D. 1988. Migraine and Magnesium: Eleven Neglected Connections. *Perspectives in Biology and Medicine*. **31** 526–557.
- Swanson, D. 1989. Online Search for Logically-Related Noninteractive Medical Literatures: A Systematic Trial-and-Error Strategy. *Journal of the American Society for Information Science*. **40**(5) 356–358.
- Swanson, D. 1990. Somatomedin C and arginine: implicit connections between mutually isolated literatures. *Perspectives in Biology and Medicine*. **33**(2) 157–186.
- The Getty Research Institute. 2011. website. URL <http://www.getty.edu/research/tools/vocabularies/>. Retrieved 04/04/2011.
- Tremblay, M.C., Berndt, D.J., Luther, S.L., Foulis, P.R., and French, D.D. 2009. Identifying Fall-Related Injuries: Text Mining the Electronic Medical Record. *Information Technology and Management*. **10**(4) 253–265.
- Trochim, W. 2000. *The Research Methods Knowledge Base, 2nd Ed*. 2nd ed. Atomic Dog Publishing, Cincinnati, OH.
- Trost, H. 2003. Morphology. R. Mitkov, ed., *The Oxford Handbook of Computational Linguistics*. Oxford University Press, New York, NY, 25–47.
- Tuttle, M.S., Blois, M.S., Erlbaum, M.S., Nelson, S.J., and Sheretz, D.D. 2008. Toward a Bio-Medical Thesaurus: Building the Foundation of the UMLS. *Proc Annu Symp Comput Appl Med Care* 191–195.
- Valient, L. 1983. A Theory of the Learnable. *Communications of the ACM*. **27**(11) 1134–1142.
- van Heijst, G., Schreiber, A.T., and Wielinga, B.J. 1997. Using Explicit Ontologies in KBS Development. *International Journal of Human Computer Studies*. **45** 183–292.

- Vapnik, V.N. 1998. *Statistical Learning Theory* . Wiley-Interscience, New York, NY.
- Vapnik, V.N. 2000. *The Nature Statistical Learning Theory* . Springer-Verlag, New York, NY.
- Wielinga, B.J. and Schreiber, A.T. 2010. Reusable and Shareable Knowledge Bases: A European Perspective. *Knowledge Building and Knowledge Sharing* 110–120.
- Wilcox, A. and Hripcsak, G. 1999. Classification Algorithms Applied to Narrative Reports. *Proceedings of the AMIA Symposium*.
- Wolf, M.S. and Bennett, C.L. 2006. Local Perspective of the Impact of the HIPAA Privacy Rule on Research. *Cancer*. **106**(2) 474–479.
- Wolff, C. 1736. *Philosophia Prima sive Ontologia* . Leipzig.
- Won, Y., Song, H.J., Kang, T.W., Kim, J.J., Han, B.D., and Lee, S.w. 2003. Pattern analysis of serum proteome distinguishes renal cell carcinoma from other urologic diseases and healthy persons. *Proteomics*. **3**.
- Younger, D.H. 1967. Recognition and parsing of context-free languages in time  $n^3$ . *Information and Control*. **10**(2) 189–208.
- Zeng, Q., Goryachev, S., Weiss, S., Sordo, M., Murphy, S., and Lazarus, R. 2006. Extracting Principal Diagnosis, co-morbidity and smoking status for asthma research: evaluation of a natural language processing system. *BMC Medical Informatics and Decision Making*. **6**(30) 1–9.
- Zimmermann, A. and Raedt, L. 2004. Lecture Notes in Computer Science. *Discovery Science: Lecture Notes in Computer Science*. **3245** 60–72.

## **Appendix A**

### **Terms and Definitions**

Corpus - A large set of texts or documents

cTAKES - Clinical Text Analysis and Knowledge Extraction System

DTI - Decision Tree Induction

EMR - Electronic Medical Record

EHR - Electronic Health Record

FRAE - Fall Related Ambulatory Event

FRI - Fall Related Injury

GATE - General Architecture for Text Engineering

ICD-9 - International Statistical Classification of Diseases and Related Health Problems

Lexicon - The vocabulary including its words and expression for a language

LSA - Latent Semantic Analysis

NIH - National Institutes of Health

NLM - National Library of Medicine

NLP - Natural Language Processing

PHR - Personal Health Record

POS - Part of Speech

SOAP - Subjective Objective Assessment Plan SOAPE - SOAP + Education

STM - Statistical Text Mining

SVD - Singular Value Decomposition

SVM - Support Vector Machine

UIMA - Unstructured Information Management Architecture

UMLS - Unified Medical Language System

## **Appendix B**

### **Tools**

Most of the NLP tools are based on a pipeline architecture, which means that the output of one component is the input to the next component in the pipeline much like water running through pieces of pipe. Basic to most NLP tools are components that parse a document in segments, sentences, tokens, parts of speech, and chunks. From there, customized modules are implemented to process the language in the desired way. For example, a named entity component would look up terms in some dictionary to determine if they are named entities or not. Temporal relationships is another use for an NLP component. Yet another is negation. Once named entities are determined, it is often necessary to determine their negation status.

#### **B.1 Negation**

The problem of negation assertion is a phenomenon that continues to challenge linguistic researchers. One might think it is as easy as associating a negation term with a term in the sentence. Syntax can cause many issues with this problem. The English language does not contain double negatives as do some European languages but negating a negative term occurs. For example, “The patient does not seem unhappy.” Also is the concept “rule out” in the sentence, “we have not ruled out diabetes.” Negation terms can also apply to more than one term in a sentence. In the sentence, “Patient denies dizziness and falling” the patient is denying both falling and dizziness. Negation terms can also appear in a sentence either before or after the term to be negated. One of the more complex situation is where the negation appears in either the sentence before or after the sentence containing the term to be negated. Some approaches such as regular expressions and heuristics have been used to approach this problem but currently the most widely used method within the NLP community is NegEx (Chapman et al., 2001). After a sentence has been parsed into tokens, NegEx uses a list of terms, both pre and post negation terms, to determine if a negation term ex-

ists. If a negation term exists it then negates the terms either preceding or following it within the sentence. Terms within a sentence are marked as “not negated”, “negated”, “possibly negated”, or “not found” if the entity is not found within the sentence. NegEx does not allow for negation in another sentence.

## **B.2 MetaMap/MMtx**

MetaMap is a program that was developed by the National Library of Medicine (NLM) and is public domain (Aronson, 2001). It was designed to map Metathesaurus concepts to biomedical text. This software was evaluated as a potential source for this dissertation. It was rejected for two reasons; one it is written in prolog and a prolog developer was not available and two, it was not available for the Microsoft Windows platform. At the time, it was only available for different versions of the UNIX/Linux OS.

## **B.3 UIMA/cTAKES**

Unstructured Information Management Architecture (UIMA) was originally created by IBM and is now an Apache open source project (Apache Software Foundation, 2009). It is an architecture that allows developers to create systems that process large volumes of unstructured information to discover information relevant to end users. UIMA allows components, analysis engines (AEs) to be combined to create a processing pipeline. It is more than just an application framework. It is an enterprise level architecture. Much research has been done either building NLP applications using UIMA or evaluating components built based on UIMA (Ogren et al., 2008; Kipper-Schuler et al., 2008).

Probably the most popular medical informatics pipeline based on the UIMA is the Mayo clinic’s Clinical Text Analysis and Knowledge Extraction System (cTAKES) (Savova et al., 2008). It was designed specifically to find cancer specific concepts in health documents but has been released under an open source license and can be configured to locate any medically relevant concepts within documents. We initially investigated using this pipeline for this dissertation, however, because of the complex framework of both UIMA and cTAKES, modifying the pipeline became too complicated and time consuming.

#### **B.4 GATE**

Another pipeline architecture is the General Architecture for Text Engineering (GATE) (Cunningham et al., 2002; Cunningham, 2002). This tool was developed by a group at the University of Sheffield and is also available as open source software. Many other text processing pipelines, both medical and non-medical, have been created using this tool (Pakhomov et al., 2005; Cunningham et al., 2002; Mitchell et al., 2004; Holton, 2009). The Health Information Text Extraction (HITEx) tool was created originally to extract smoking status for asthma research (Zeng et al., 2006). In the same way cTAKES is a medically related pipeline built on the UIMA architecture, HITEx is a medically related pipeline built on the GATE architecture. Based on an evaluation of these systems, GATE was selected as the preferred tool for this research.

## **Appendix C**

### **SAS Enterprise Miner**

The focus of this appendix is to show the model building process used by SAS Enterprise Miner. SAS has several statistical applications such as basic SAS and Enterprise Guide. Enterprise Miner is a separate application that allows users to create pipelines that perform data mining tasks. There is an additional module that can be added to Enterprise Miner called Text Miner that allows for text mining processes to be created. For the first experiment in this dissertation, SAS Enterprise Miner 6.2 with Text Miner 4.2 was used to create pipelines that analyzed the three data sets: clinical progress notes (TEXT), CUIs (CUIs), and common language (CMN).

Each component in SAS Enterprise Miner has both input and output. The modules are connected to each other in a pipeline fashion to create a process. Each module or component has properties that can be manipulated to make the process behave in the desired manner. SAS is a proprietary software application and as such, does not allow components to be created by users as open source solutions do. What SAS does have to address this is a code component that allows a SAS developer to create SAS code components that deliver behavior as coded. What follows is a description of the pipeline process used to mine the CUIs. However, all three pipeline processes were created similarly.

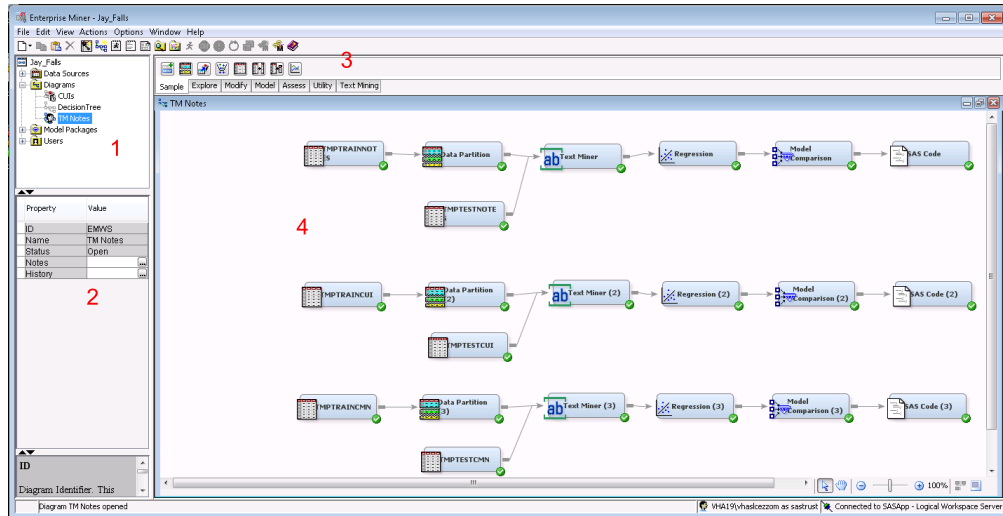
#### **C.1 Graphical User Interface**

Figure C.22 shows a screen shot of the entire SAS Enterprise Miner graphical user interface (GUI). The numbers identify the main portions of the GUI are explained here in further detail.

##### **C.1.1 1 - Project**

This first pane shows the Enterprise Miner project. The main components from this pane are the Data Sources and the Diagrams. Multiple SAS data sources can be created in used in the different





**Figure C.22.:** SAS Enterprise Miner Graphical User Interface

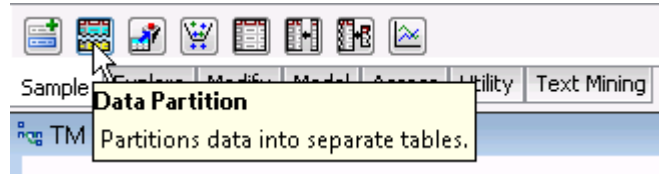
SAS applications. There are six SAS data sets used in this project. A user can also create multiple diagrams in a single project. There are three list here but the TM Notes diagram is the only one that is discussed here.

### C.1.2 2 - Properties

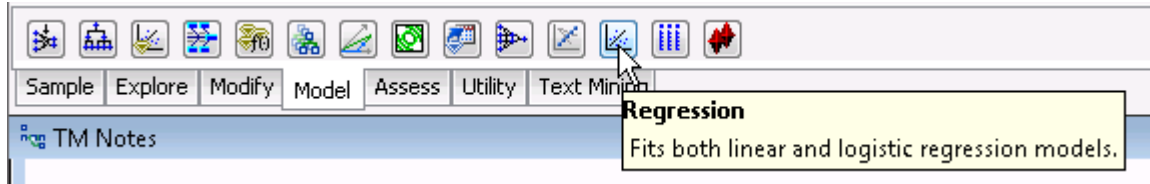
This pane shows the properties for the selected component. Displayed here are the properties for the diagram but as each component in the pipeline is selected, the appropriate properties will be displayed in this pane.

### C.1.3 3 - Tab Bar

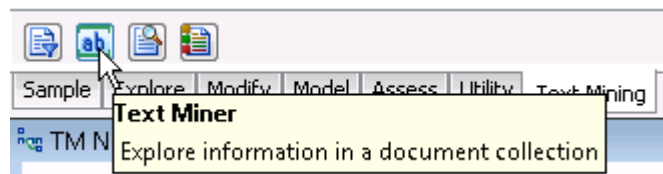
The tab bar shows the different components that are installed in Enterprise Miner. The Sample Tab, figure C.23 is used to create SAS data sets and data partition components. The Model Tab, figure C.24, has the available model building algorithms. Because this installation also has SAS Text Miner installed, there is a Text Mining tab, figure C.25, that has the components available for text mining. The components are dragged from the tab bar into the diagram workspace in order to build the pipeline.



**Figure C.23.:** Sample Tab



**Figure C.24.:** Model Tab



**Figure C.25.:** Text Miner Tab

#### C.1.4 4 - Diagram Workspace

This is the main pane where the process is created. Shown in figure C.22 are three separate pipelines created in a single diagram workspace. Each one is independent of the others. The icon on the bottom right corner of each component symbolizes its completion status. The green checks in this figure let the user know that each component has executed through completion successfully. If a component had failed it would have a red X and the pipeline would halt. One of the benefits of SAS Enterprise Miner is that only the components that have been changed or have not run through completion will execute. If a pipeline has a component that processes the data set and takes a significant amount of time but really only needs to run once will not slow down the total time of the pipeline process.

#### C.2 Pipeline

The following sections explain the pipeline process for the CUIs. The purpose of each component is discussed along with its corresponding properties. The TEXT and CMN pipelines use the same

components with only slight changes to the properties. These changes are discussed in Chapter 4.

### C.2.1 TMPTRAINCUI

This is the training set containing the CUIs for each progress note. It also contains the ID for each note and the classification, FALL nFALL, decided upon by the clinicians. Figure C.26 show the properties for this component. The Output Type is changed from the default View to Data. This is done to make the SAS code module located at the end of the pipeline work correctly. SAS Views are difficult to access.

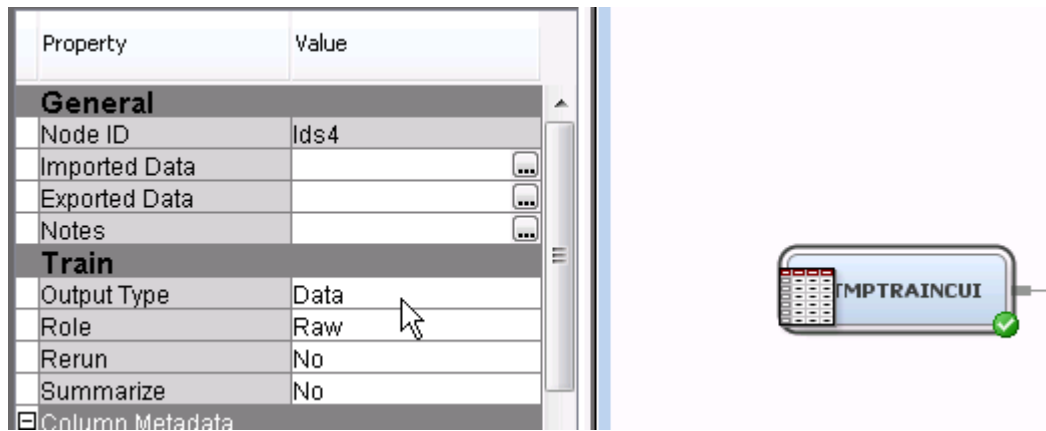


Figure C.26.: Training Data Set Properties

### C.3 Data Partition

The data partition component in SAS Enterprise Miner allows for a data set to be divided into up to three partitions: Training, Validation, and Test. The model is built using the Training partition. The model can be tuned using the Validation partition. The Test partition is unseen data and the model is tested on this data set. For this dissertation the data was separated into two partitions: Training and Test. Using SAS, the Training partition was then separated into two further partitions: Training and Validation, using a 70% 30% split. It was also changed from View to Data for the same reasons as explained with the SAS data set above. Figure C.27 show the properties that were changed for this component.

Property	Value
<b>General</b>	
Node ID	Part2
Imported Data	...
Exported Data	...
Notes	...
<b>Train</b>	
Variables	...
Output Type	Data
Partitioning Method	Default
Random Seed	12345
<b>Data Set Allocations</b>	
Training	70.0
Validation	30.0
Test	0.0

**Figure C.27.:** TRAINING Data Partition Properties

### TMPTESTCUI

This is another SAS data set and contains the Test partition of the CUIs data set. Figure C.28 show the properties for this component. The Output Type was changed from View to Data for this data set for the same reasons as the Training data set.

<b>General</b>	
Node ID	Ids3
Imported Data	...
Exported Data	...
Notes	...
<b>Train</b>	
Output Type	Data
Role	Test
Rerun	No
Summarize	No

**Figure C.28.:** TEST Data Partition Properties

### C.3.1 Text Miner

This component is available to Enterprise Miner because the Text Miner package was installed. Figure C.29 shows the properties for this module.

## **Parse**

The standard stoplist that comes with Enterprise Miner was used. This removed common words that have no predictive power such as articles. Terms or CUIs that occurred in single documents were also removed. In the case of the CUIs, since CUIs are just alpha numeric representations of concepts, there was no need to treat punctuation, numbers, parts of speech, or noun groups differently. However, these were all used on the TEXT and CMN data sets.

## **Transform**

Since LSA was used, SVD dimensions were computed. Figure C.29 shows 200 dimensions created using Binary Frequency Weightings. 200 roll-up terms were also used for this specific component. In SAS speak, a roll-up term is simply a term kept for analysis rather than being excluded.

## **Cluster**

In this example, no clusters were used as can be seen by the Automatic Cluster property being set to NO. Once this property is set to no, all of the remaining properties in this section are ignored.

### **C.3.2 Regression**

Logistic regression was the model building technique used in the first experiment. Figure C.30 show the properties for this component. Many different logistic regression models were tested by changing the properties of the component with the final model being explained in Chapter 4.

### **C.3.3 Model Comparison**

The model comparison component was used to take the model built from the TRAINING data and execute it on the unseen TEST data. The default properties were used for this component.

### **C.3.4 SAS Code**

A SAS code component was used to extract the confusion matrix from embedded SAS data sets for both TRAINING and TEST. Figure C.31 shows the code executed.

variables	
Interactive	
Force Run	No
<input type="checkbox"/> Parse	
Parse Variable	concepts
Language	ENGLISH
Stop List	Sashelp.stoplst
Start List	
Stem Terms	No
Terms in Single Document	No
Punctuation	No
Numbers	No
Different Parts of Speech	No
Ignore Parts of Speech	
Noun Groups	No
Synonyms	Sashelp.engsynms
Find Entities	No
Types of Entities	
<input type="checkbox"/> Transform	
Compute SVD	Yes
SVD Resolution	Low
Max SVD Dimensions	200
Scale SVD Dimensions	No
Frequency weighting	Binary
Term Weight	None
Roll up Terms	Yes
No. of Rolled-up Terms	200
Drop Other Terms	No
<input type="checkbox"/> Cluster	
Automatically Cluster	No
Exact or Maximum Number of Clusters	Maximum
Number of Clusters	40
Cluster Algorithm	EXPECTATION-MAXIMIZATION
Ignore Outliers	No
Hierarchy Levels	.
Descriptive Terms	5
What to Cluster	SVD Dimensions

**Figure C.29.:** Text Miner Properties

Equation	
Main Effects	Yes
Two-Factor Interactions	No
Polynomial Terms	No
Polynomial Degree	2
User Terms	No
Term Editor	...
Class Targets	
Regression Type	Logistic Regression
Link Function	Probit
Model Options	
Suppress Intercept	No
Input Coding	Deviation
Model Selection	
Selection Model	Forward
Selection Criterion	Schwarz Bayesian Crit
Use Selection Defaults	Yes
Selection Options	...
Optimization Options	
Technique	Congra
Default Optimization	Yes
Max Iterations	0
Max Function Calls	0
Maximum Time	1 Hour
Convergence Criteria	
Uses Defaults	Yes
Options	...
Print Output Options	
Confidence Limits	No
Covariance	No
Correlation	No
Statistics	No
Suppress Output	No
Details	No
Design Matrix	No

**Figure C.30.:** Regression Properties

Training Code

```
libname falls "D:\EM Projects\vhslcezzom\Jay_Falls\Workspaces\EMWS";  
PROC SQL;  
  CREATE TABLE jay.results_tampa_cui_test_reg AS  
  SELECT t1._class,  
         t1.I__Class,  
         (COUNT(t1.filename)) AS COUNT_of_valid_filename  
  FROM falls.MdlComp2_test AS t1  
  GROUP BY t1._class, t1.I__Class;  
  CREATE TABLE jay.results_tampa_cui_valid_reg AS  
  SELECT t2._class,  
         t2.I__Class,  
         (COUNT(t2.filename)) AS COUNT_of_filename  
  FROM falls.MdlComp2_validate AS t2  
  GROUP BY t2._class, t2.I__Class;  
QUIT;
```

**Figure C.31.:** SAS Code Component

For each data partition this code creates a SAS data set containing the counts of the true positives, true negatives, false positives, and false negatives. In this code , \_class (figure C.32) is the actual classification and I\_class (figure C.33) is the classification determined by the model. These variables are accessed by clicking on the ellipses for the Exported Variables From the dialog window that opens, figure C.34, select the data set, TEST in this case, and click on the Properties button. From there select the the Variables tab to see the variables for that data set.



Properties - EMWS.MdlComp2\_TEST

Table Variables

Columns:  Label  Mining  Basic  Statistics

Name	Role	Level
SVD_9	Input	Interval
SVD_90	Rejected	Interval
SVD_91	Rejected	Interval
SVD_92	Rejected	Interval
SVD_93	Rejected	Interval
SVD_94	Rejected	Interval
SVD_95	Rejected	Interval
SVD_96	Rejected	Interval
SVD_97	Rejected	Interval
SVD_98	Rejected	Interval
SVD_99	Rejected	Interval
WARN_	Assessment	Nominal
class	Target	Binary
concept	Text	Nominal
document_	Rejected	Interval
b_class	Segment	Interval
filename	ID	Nominal
rc	Rejected	Interval

**Figure C.32.:** Actual Classification Variable

The screenshot shows a dialog box titled "Properties - EMWS.MdlComp2\_TEST". It has two tabs: "Table" and "Variables". The "Variables" tab is active. Below the tabs, there are four checkboxes: "Label", "Mining", "Basic", and "Statistics", all of which are unchecked. The main area contains a table with three columns: "Name", "Role", and "Level".

Name	Role	Level
F_class	Classification	Nominal
I_class	Classification	Nominal
P_classfa	Prediction	Interval
P_classnotfa	Prediction	Interval
R_classfall	Residual	Interval
R_classnotfa	Residual	Interval
U_class	Classification	Nominal
ROLL_1	Rejected	Interval
ROLL_10	Rejected	Interval
ROLL_100	Rejected	Interval
ROLL_101	Input	Interval
ROLL_102	Rejected	Interval
ROLL_103	Rejected	Interval
ROLL_104	Rejected	Interval
ROLL_105	Rejected	Interval
ROLL_106	Rejected	Interval
ROLL_107	Rejected	Interval
ROLL_108	Rejected	Interval

Figure C.33.: Predicted Classification Variable

The screenshot shows a dialog box titled "Exported Data - Model Comparison (2)". It contains a table with four columns: "Port", "Table", "Role", and "Data Exists". The "TEST" row is highlighted in blue. At the bottom of the dialog, there are four buttons: "Browse...", "Explore...", "Properties..." (with a mouse cursor over it), and "OK".

Port	Table	Role	Data Exists
TRAIN	EMWS.MdlComp2_TRAIN	Train	Yes
VALIDATE	EMWS.MdlComp2_VALIDATE	Validate	Yes
TEST	EMWS.MdlComp2_TEST	Test	Yes
RANK	EMWS.MdlComp2_EMRANK	Score Rankings	Yes
SCOREDIST	EMWS.MdlComp2_EMSCOREDIST	Score Distribution	Yes
REPORTFIT	EMWS.MdlComp2_EMREPORTFIT	Fit Statistics	Yes

Figure C.34.: Data Set Variables Properties

## **Appendix D**

### **General Architecture for Text Engineering (GATE)**

The focus of this appendix is to show the model building process used by GATE, an open source NLP program. This example follows the pipeline process used to create the data sets used in this dissertation. The first experiment uses three different data sets extracted from a single set of clinical progress notes: the original notes (TEXT), extracted CUIs (CUIs), and common language (CMN). This pipeline process extracts both the CUIs and the common language.

#### **D.1 Graphical User Interface**

Figure D.35 shows a screen shot of the entire GATE graphical user interface (GUI). The numbers identify the main portions of the GUI are explained here in further detail.

##### **D.1.1 1 - Navigation**

This part of the GUI shows the all of the GATE components that are available to the user.

- Applications - Each pipeline process is stored in an application
- Language Resources - Corpora can be loaded in interface and used execute a pipeline against
- Processing Resources - The components used to create the pipeline in GATE are referred to as Processing Resources. There are several supplied with the application plus users can create their own by extending classes supplied with GATE.

##### **D.1.2 2 - Loaded Processing Resources**

This part of the application show the Processing Resources available to be used in the pipeline. They must be loaded here before they can be used in the pipeline.

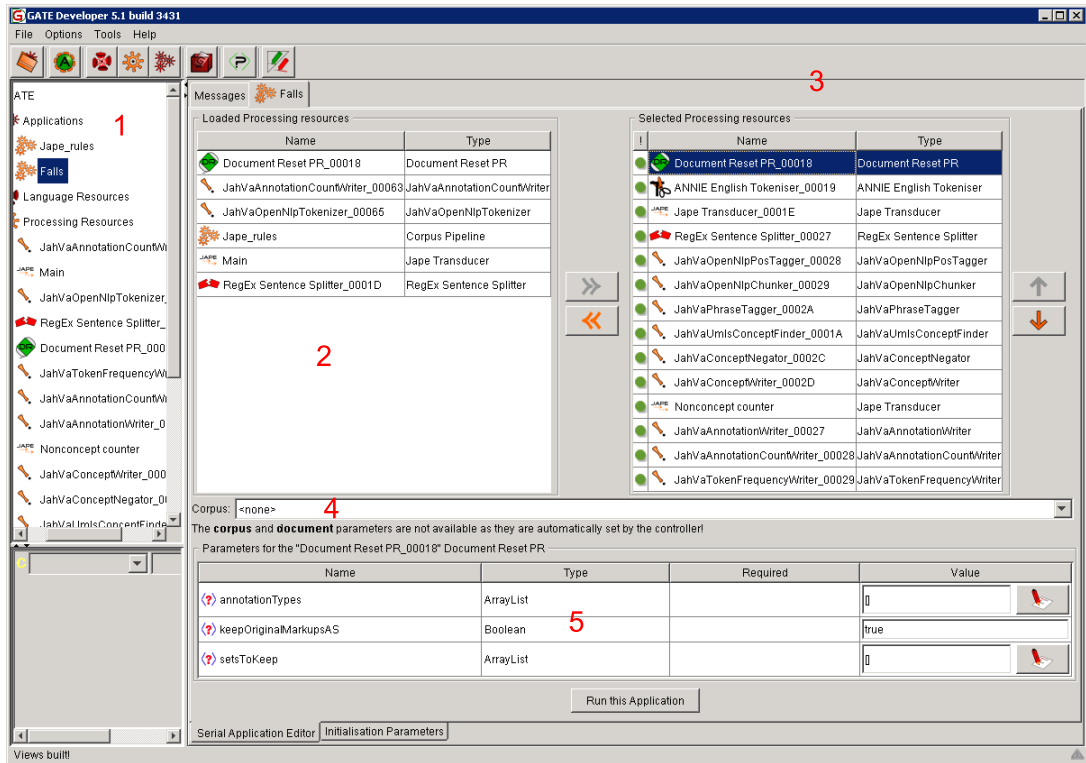


Figure D.35.: GATE Graphical User Interface

### **D.1.3 3 - Selected Processing Resources**

This part of the application shows the actual pipeline. Not only are the Processing Resources listed here but the order of the resources to be used in the pipeline is also defined here.

### **D.1.4 4 - Corpus**

A corpus can be listed here. If so, the pipeline is executed against it when the application is used. The corpus used in this dissertation was relatively large and the application executed faster from a command line supplied by GATE.

### **D.1.5 5 - Parameters**

Each of the processing resources has a list of corresponding parameters that can be set by the user.

## **D.2 Pipeline Processing Resources**

What follows is a list of the processing resource that comprise the NLP pipeline used to create the CUIS and the CMN data sets along with their descriptions. Screen shots of their parameters are also included if they were changed from the default settings.

Several of the processing resources start with JAHVA, James A. Haley VA, these are the in house developed processing resources. Some are wrappers around predeveloped components such as OpenNLP components. Other were develop purely in house in order to meet a specific need of the pipeline.

### **D.2.1 Document Reset PR**

This is the first processing resource and simple resets all of the annotations for note within the corpus.

### **D.2.2 ANNIE English Tokeniser**

ANNIE, A Nearly New Information Extraction System, is an IE system that is distributed with GATE. Some of these prebuilt components were used in this pipeline. The tokeniser separates the document into tokens. Tokens are not words but contain text. There are different types of tokens.

- Word - contiguous upper or lower characters including a hyphen and no other punctuation.
- Number - Any combination of consecutive digits.
- Symbol - Examples are currency symbols but not punctuation.
- Punctuation - There are three types of punctuation: start e.g. (, end e.g. ), and other e.g. .  
Each symbol is a separate token.
- Space - White space(s) are defined as a single token.

### **D.2.3 JAPE Transducer**

JAPE, Java Annotation Patterns Engine, and allows you to recognize regular expressions in annotations on documents. The JAPE processing resources were used in this pipeline to extract the common language and get counts of the different data set sizes.

### **D.2.4 RegEx Sentence Splitter**

There are different methods for sentence splitting such as a rule based approach or a machine learning approach but based on the format of the clinical progress notes in the data set for this dissertation, the regular expression based sentence splitter gave the best performance.

### **D.2.5 JAHVAOpenNLPPOSTagger**

This is a wrapper around the OpenNLP Part of Speech Tagger. The OpenNLP POS Tagger uses a probability model to predict the correct POS tag out of the tag set.

### **D.2.6 JAHVAOpenNLPCChunker and JAHVAPhraseTagger**

The chunker in combination with the phrase tagger takes the POS annotations from the POS Tagger and combines them into phrases such as noun and verb phrases.

### **D.2.7 JAHVAUMLSConceptFinder**

There are several UMLS concept finders in existence but none seemed to have the functionality desired for these studies. They all have the basic functionality of looking up a given phrase in the

UMLS database. Where they failed is that they returned too much information. For example, the phrase, “Regional Medical Center” would return seven CUIs: one each for “Regional”, “Medical”, “Center”, “Regional Medical”, “Medical Center”, and “Regional Medical Center”. This arbitrarily inflated the counts of the CUIs representing the progress notes. It was desired for these projects to have only the CUI that described the largest phrase or in this case “Regional Medical Center.” Therefore a component was created that did just that. Figure D.36 shows the parameter settings for this processing resource. In most cases, UMLS concepts are only looked up in noun phrases, however, since most “fall” related words are verbs, verb phrases were also added to the lookup.

Parameters for the "JahVaUmlsConceptFinder_0001A" JahVaUmlsConceptFinder			
Name	Type	Required	
annotationSetName	String		
phraseTypes	HashSet	✓	[VP, NP]

**Figure D.36.:** JAHVA UMLS Concept Finder Parameters

### D.2.8 JAHVAConceptNegator

This is an implementation of the NegEx algorithm. It takes a given phrase and looks for predefined negation terms within a phrase. Negation terms are either pre-negation such as “Patient denies falling” or post-negation such “Broken leg is ruled out.” This component returns a negation status of either negated, not negated, or possibly negated. Figure D.37 shows the parameter settings for this resource.

Parameters for the "JahVaConceptNegator_0002C" JahVaConceptNegator			
Name	Type	Required	Value
annotationSetName	String		
includePossibleNegation	Boolean	✓	true

**Figure D.37.:** JAHVA Concept Negation Parameters

### D.2.9 JAHVAConceptWriter

This processing resource creates two different unique tokens to be used in the STM process. First, it combines the CUI found in the ConceptFinder and the negation status found in the Concept-

Negator by concatenating them with a predefined character. For example, C0013478\_nn is the CUI representing the concept “WHATEVER” and nn is the not negated status of the CUI. Secondly, it handles the WSD issue by combining all of the CUIs returned for a single term to make a single CUI. For example, there are seven CUIs that represent the different meanings of the term “cold.” This dissertation was not concerned with which was the correct CUI but only that the term “cold” was found in the text. Returning seven different CUIs arbitrarily inflated the word count representing the clinical notes so the handles this, all of the CUIs returned representing a single term were concatenated to form a single CUI. Figure D.38 shows the parameter values for the resource.

Parameters for the "JahVaConceptWriter_0002D" JahVaConceptWriter			
Name	Type	Required	Value
annotationSetName	String		
groupBeginCharacter	String		[
groupEndCharacter	String		]
groupSeparatorCharacter	String		
includeNegation	Boolean		true

**Figure D.38.:** JAHVA Concept Writer Parameters

### D.2.10 Nonconcept Counter (JAPE Tranducer)

This JAPE transducer was used to count all of the terms used to create the Common Language data set.

### D.2.11 JAHVAAnnotationWriter

This processing resource writes annotations. The type of annotation is defined in the parameters and in this case nonconcept annotations were used. This is how the Common Language data set was created. Figure D.39 shows the parameter settings.

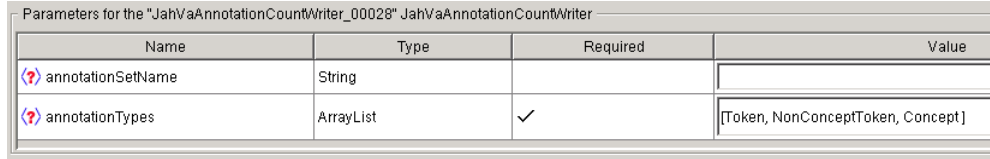
Parameters for the "JahVaAnnotationWriter_00027" JahVaAnnotationWriter			
Name	Type	Required	Value
annotationSetName	String		
annotationType	String	✓	NonConceptToken
featureSeparator	String		
featureTypes	HashSet	✓	{string }

**Figure D.39.:** JAHVA Annotation Writer Parameters



## D.2.12 JAHVAAnnotationCountWriter

This component provides counts of the different data sets used. Figure D.40 shows the parameter settings for this resource.



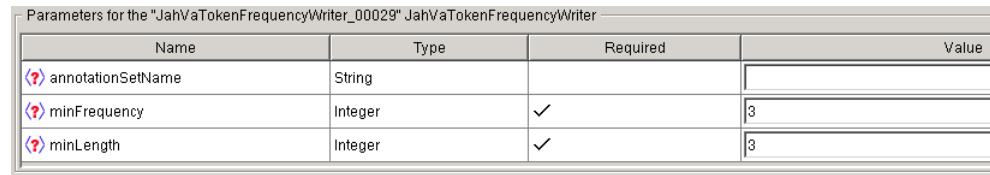
Parameters for the "JahVaAnnotationCountWriter\_00028" JahVaAnnotationCountWriter

Name	Type	Required	Value
annotationSetName	String		
annotationTypes	ArrayList	✓	[Token, NonConceptToken, Concept]

**Figure D.40.:** JAHVA Annotation Count Writer Parameters

## D.2.13 JAHVATokenFrequencyWriter

This component provides the frequency tokens occur in the corpora. For example, the CUI representing fall may appear in a corpus 704 times. Figure D.41 shows the parameter settings for this resource.



Parameters for the "JahVaTokenFrequencyWriter\_00029" JahVaTokenFrequencyWriter

Name	Type	Required	Value
annotationSetName	String		
minFrequency	Integer	✓	3
minLength	Integer	✓	3

**Figure D.41.:** JAHVA Token Frequency Writer Parameters

## Appendix E

### RapidMiner

RapidMiner started at the Artificial Intelligence Unit of the University Dortmund. In 2006, two of the original team members formed a company, Rapid-I, that is now the main contributor of the RapidMiner product. The community edition of RapidMiner was used for this dissertation. The focus of this appendix is to show the model building process used by RapidMiner, an open source statistical program. This example follows the building of the decision tree induction process used in the third experiment in this dissertation.

#### E.1 Graphical User Interface

Figure E.42 shows a screen shot of the entire RapidMiner graphical user interface (GUI). The numbers identify the main portions of the GUI are explained here in further detail.

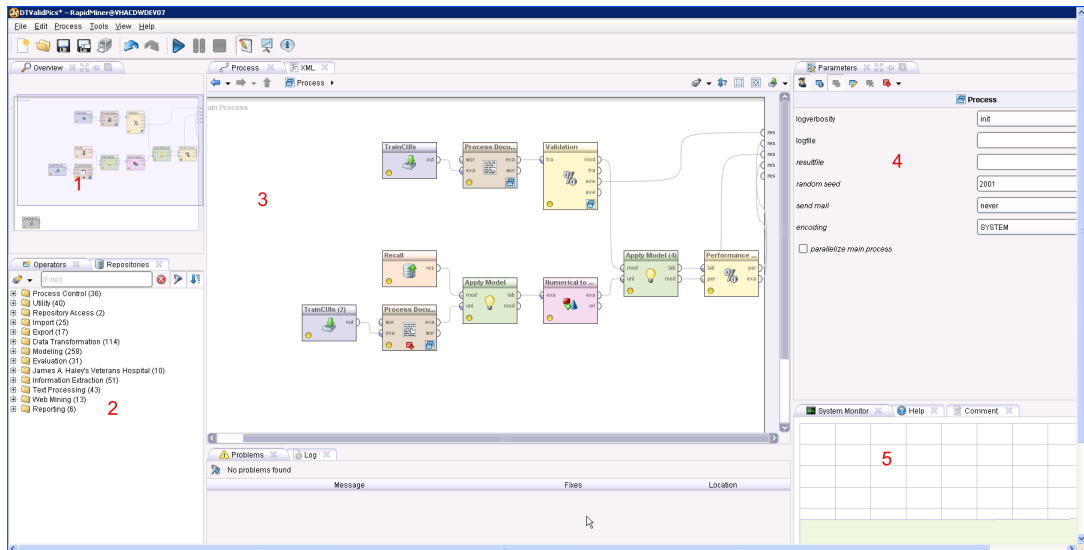


Figure E.42.: RapidMiner Graphical User Interface

### **E.1.1 1 - Overview**

The part of the GUI shows the overview of the entire process to be executed. It is not meant as an editable window but merely to show the overall process.

### **E.1.2 2 - Operators and Repository**

The modules that make up the process in RapidMiner are called operators. One of the tabs in this section shows all of the available operators to this installation of RapidMiner. Since it is an extensible environment, RapidMiner also allows for the creation of operators and these will be listed here as well. This installation has 10 custom operators installed in the James A. Haley folder.

RapidMiner also has a concept of a repository. Processes and datasets can be stored in the repository to be used again at a later time or in another process. The repository tab lists all of these. It is not shown here because it was not used for this single process.

### **E.1.3 3 - Process**

This is the main pane where the work is created. Each of the icons in this pane is an operator and when they are connected together, they make up the process. Each operator has parameters that are set by the user to determine how the operator will behave. Some of the operators are containers for other operators. In this example, the Validation operator is a container operator. It will be explained further in a subsequent section. Each operator icon has a colored dot in the bottom left corner to show its status: red signifies an error, yellow signifies the operator has not executed yet, and green signifies that the operator has executed successfully. Most operators have inputs, with the exception of data, and all have outputs. What those are, are determined by the individual operator. For example, `exa` is the example data set and `mod` is the model. These inputs and outputs are how the operators are connected together to create a process.

Breakpoints can also be set in the process pane. This will allow the user to see the progress of the operator and any outputs when it is finished or during its execution.

Below the process pane are status tabs that display errors that may occur and any logs that may have been recorded.

RapidMiner stores the actual process in an XML file. There is an XML tab in this pane that allows the user to edit the actual XML if desired instead of using the GUI.

### E.1.4 4 - Parameters

Each operator has parameters that can be set by the user to determine how the operator behaves. These are set on the parameters tab. The available parameters are determined by the operator developer. Each operators parameters will be discussed in detail in a subsequent section.

### E.1.5 5 - Monitoring and Help

Each operator will have help displayed on the help tab that gives a brief description and a list of its inputs and outputs. The monitoring tab shows the memory usage of the process throughout its execution.

## E.2 DTI Process

This will expand upon the DTI process used to create the decision tree in the third experiment. Only the operators and their parameters are discussed here. The data set and methods for creating the DTI process are discussed in Chapter 6. Each subsection is title with the title of the operator discussed. Figure E.43 shows the main process for the DTI. Each operator and its corresponding parameters, inputs, and outputs are discussed in further detail.

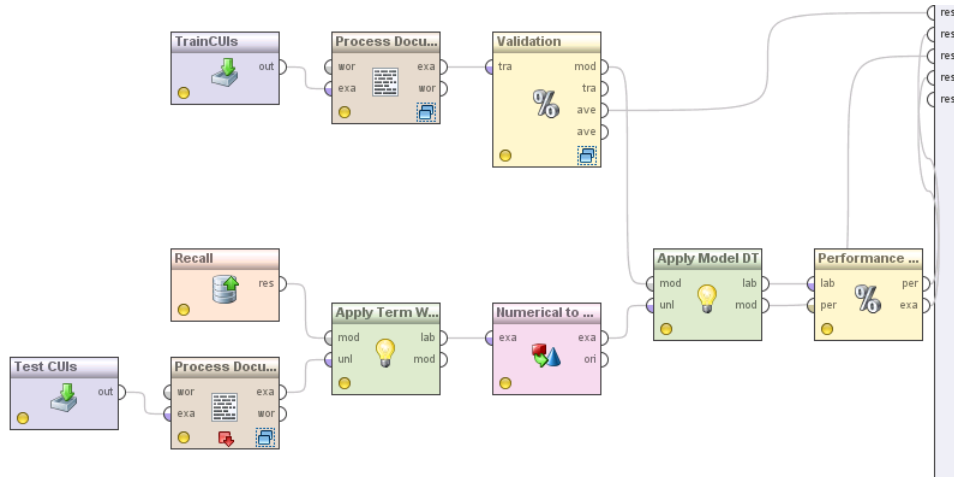


Figure E.43.: Main DTI Process

### E.2.1 TrainCUIs

This is the training data set. The parameters set here is the file with the actual data and is not shown. In this case, the CUIs are in a single, tab delimited file with format of: Note ID (ID), CUIs (Nominal), classification (label). A wizard is also included with this operator that allows the data parameters to be set correctly. Each note is listed on a separate line in this file.

### E.2.2 Process Document

Figure E.44 shows the properties for the Process Document operator. This operator takes the raw data and converts it into a term by document matrix. There are two Process Document operators used in this process but only one is discussed. They have the same purpose. One is used on training data and the other on test data.

The screenshot shows the configuration interface for the 'Process Documents from Data' operator. The settings are as follows:

- create word vector
- vector creation: Binary Term Occurrences
- add meta information
- keep text
- prune method: absolute
- prune below absolute: 2
- prune above absolute: 6000
- datamanagement: double\_sparse\_array
- select attributes and weights
- parallelize vector creation

**Figure E.44.:** Process Document Parameters

- Vector Creation - Binary term occurrences was used. This is a simple, does the term occur at least once or not.
- Prune Method -

- Prune Below Absolute - It was decided that if the term occurred less than twice in the entire corpus it was not to be included in the matrix.
- Prune Above Absolute - This was a mandatory parameter and this was chosen as an arbitrarily large number larger than the total number of notes. This would never be reached to prune terms because the data set contained only 5,009 notes.

### Inputs

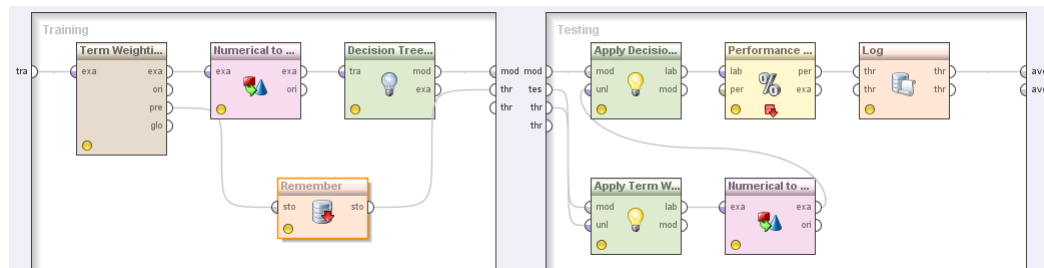
- Example set - from the original data set

### Outputs

- Example set - Term by document matrix

## E.2.3 Validation

This is a container operator and was used to create the cross validation process of of the DTI. Figure E.45 shows the operators that are contained in the validation operator and are explain in this section. The Validation operator is split into two containers: training and testing. Term Weightings, Numerical to Binomial, Remember, and Decision Tree are operators placed in the training container. Apply Model 2 and 3, Numerical to Binomial, Performance, and Log are operators placed in the testing container.



**Figure E.45:** Validation Operator

### Term Weightings

Figure E.46 shows the parameters for the Term Weights operators. Term weightings are calculated based on training data and the weightings model is remembered. All of the terms in the term by

document matrix are assigned weightings based on Gain Ratio. This operator has an option to either keep all of the terms or just the N terms. This screen shot shows keeping the top 400 terms. This is the setting that was used in the experiment to adjust how many terms were kept for each iteration of the process.

Term Weighting (2) (Term Weighting)	
local weighting scheme	None
global weighting scheme	Gain Ratio (class)
retain terms	Top n
<b>parameter top n</b>	400
normalization scheme	None

**Figure E.46.:** Term Weights Parameters

### Inputs

- Example data set

### Outputs

- Example data set
- Preprocessing Model

### Numerical to Binomial

At this point the data is a term by document matrix containing weights for term values. This is then converted to binomial. The term by document matrix is passed through this operator and converted from weights to simply the presence or absence of a term in the output.

### Remember

The term weightings are created on training data and those weightings need to be used on the test data as well. A remember operator allows this to happen. It remembers the weightings model built on the training data and can be used in subsequent steps. The weightings cannot be also built on test data because the test data needs to be unseen by all steps of the process therefore, the weights

are remembered from the training data and applied to the test sets. If a “term” exists in test data and not in the training data, therefore, not having a weight, this term is excluded from the set and not used in the model.

## Decision Tree

Figure E.47 shows the parameters for the Decision Tree operator. The reason for these parameters is explained further in Chapter 6.

Decision Tree (2) (Decision Tree)	
criterion	gini_index
minimal size for split	4
minimal leaf size	8
minimal gain	0.1
maximal depth	100
confidence	0.25
number of prepruning alternatives	3
<input type="checkbox"/> no pre pruning	
<input type="checkbox"/> no pruning	

**Figure E.47.:** Decision Tree Parameters

### Inputs

- Training - Example Set

### Outputs

- Model - The actual decision tree
- Averagable - The statistics of the averages of the folds

### Apply Model (Term Weights)

Apply Model operators are used every time some sort of created model needs to be applied to the example set in the process. It is not just used because a final model was built and needs to be applied to data. In other words, for every created model or preprocessing model on the training side



of the Validation operator, there needs to be a corresponding Apply Model operator on the test side of the Validation operator that applies the model, whatever it is, to the test data set. This particular Apply Model operator applies the term weights that were calculated on the training data and applies them to the test data set.

#### **Inputs**

- Model - The model being applied Unlabeled Data - Data before the model is applied

#### **Outputs**

- Labeled Data - the model applied data

### **Numerical to Binomial**

This is the same as the operator discussed above.

### **Apply Model (Decision Tree)**

This operator applies the decision tree model created on the training data and applies it to the test data that is in binomial form after the training term weightings have been applied.

#### **Inputs**

- Model - The decision tree model created on the training data Unlabeled Data - Data before the model is applied

#### **Outputs**

- Labeled Data - the model applied data

### **Performance**

This operator calculates the evaluation metrics on the data input into it. The following were the metrics selected

- Accuracy
- Classification Error

- Precision
- Recall
- F Measure
- False Positive
- False Negative
- True Positive
- True Negative
- Sensitivity
- Specificity
- Positive Predictive Value
- Negative Predictive Value

#### **Inputs**

- Labeled Data - The example set

#### **Outputs**

- Performance - The evaluation metrics selected

#### **Log**

This operator writes whatever is input into it to a log file specified in the parameters. All of the evaluation metrics calculated above are logged by this operator.

#### **E.2.4 Test CUIs**

This is another data operator and it contains a link to the file containing the test data set. The format for this data is identical to the training data set.

#### **Outputs**

- Out - The test data set

### **E.2.5 Preprocess Document**

This is the same as the previous Preprocess Document operator and creates the term by document matrix from the test data.

#### **Inputs**

- Example - The test data set

#### **Outputs**

- Example - The test data with applied term by document matrix

### **E.2.6 Recall**

The Recall operator recalls whatever it is told to. In this case, it remembers the term weightings from the Remember operator. In other words, the term weightings were calculated on the terms in the training data set and applied weightings. Those weightings are now applied to the test data set.

#### **Outputs**

- Result - Term Weightings

### **E.2.7 Apply Model (Term Weightings)**

This operator performs the same functionality as the Apply Model for term weightings already discussed in the Validation operator.

#### **Inputs**

- Model - Term Weightings
- Unlabeled Data - Term by document matrix on test data

#### **Outputs**

- Labeled Data - The new term by document matrix with applied term weightings

### **E.2.8 Numerical to Binomial**

This operator performs the same functionality as the Numerical to Binomial operator already discussed in the Validation operators.

#### **Inputs**

- Example - The example set

#### **Outputs**

- Example - The term by document matrix with values converted to binomial

### **E.2.9 Apply Model (DT)**

This Apply Model operator takes the decision tree model output from the Validation operator and applies it to the test data set.

#### **Inputs**

- Model - The decision tree
- Unlabeled data - The binomial term by document matrix

#### **Outputs**

- Labeled Data
- Model

### **E.2.10 Performance**

This Performance operator executes identically as the previously discussed Performance operator with the exception that it reports the performance of the decision tree on the test data rather than the training data.

#### **Inputs**

- Labeled Data - Example set
- Performance

## **Outputs**

- Performance - The evaluation metrics selected
- Example - Example set

## **Appendix F**

### **Regulation**

A discussion of a health care related topic would not be complete without a discussion of health care related regulations. These regulations are designed to protect both clinicians and patients but indirectly effect health care related research.

#### **F.1 HIPAA**

The Health Insurance Portability and Accountability Act of 1996 (HIPAA) was first introduced by senators Kennedy and Kassebaum and has two titles; Title I protects health insurance coverage for workers and their families, and Title II requires the establishment of standards for EHR transactions (HHS, 2011) . Privacy regulations that are dictated by Title II are what affect research and apply to covered entities. A covered entity is an entity that has access to Protected Health Information (PHI). Some examples of covered entities are billing services, health insurance companies, and health care providers. The VA health care system is different from typical U.S. health systems in that it is a single entity that bills, provides health care, and provides “insurance” but because it is a covered entity, is regulated by the act. HIPAA defines what entails PHI. In general terms, PHI is anything that can identify an individual and as researchers, it is our responsibility to secure all PHI from these progress notes. The act defines 18 PHI items and they are listed below.

1. Names
2. Geographical locations containing fewer than 20,000 people
3. Dates other than year
4. Phone numbers
5. Fax numbers

6. Email addresses
7. Social Security numbers
8. Medical record numbers
9. Health plan numbers
10. Account numbers
11. License and certification numbers
12. Vehicle Identification Numbers (VIN) and license numbers
13. Device identifiers
14. Uniform Resource Locators (URLs)
15. Internet Protocol (IP) addresses
16. Biometric identifiers
17. Full face images
18. Any other unique identifier

## **F.2 HITECH**

The Health Information Technology for Economic and Clinical Health Act was part of the American Recovery and Reinvestment Act of 2009 (HHS, 2011). It addresses the issues of electronically transmitting health information, including PHI. The part of the act that is of concern for researcher is the part that enacts procedures for accounting for PHI. When, HIPAA was enacted, the EHR was not as popular as it is now. HITECH extends the accounting of PHI to include EHRs.

## **F.3 Effect on Research**

The restrictions HIPAA has had on health care related research has significantly effected researchers' abilities to perform their jobs which includes chart review. Several studies have been conducted that show this to be true. One such study showed that the percent of follow-up surveys completed

by patients of a study dropped from 96% to 34% due to HIPAA regulations (Armstrong et al., 2005). Another study based on surveys of clinical researchers shows that 67.8% of the respondents agreed the HIPAA has made health care related research more difficult while at the same time only a quarter of the respondents said that HIPAA has enhanced study participant's confidentiality and privacy (Ness, 2007). Another study showed that due to HIPAA, they have had a 73% drop in subject accrual which tripled recruitment costs (Wolf and Bennett, 2006). With the increasing amounts of identity theft, one can assume that privacy is going to become more of an issue rather than less. Because of this, there exists a need to help research adhere to HIPAA and other health care, privacy related regulations while at the same time making it easier for them to navigate these regulations without increasing costs and decreasing participation in health care related studies.

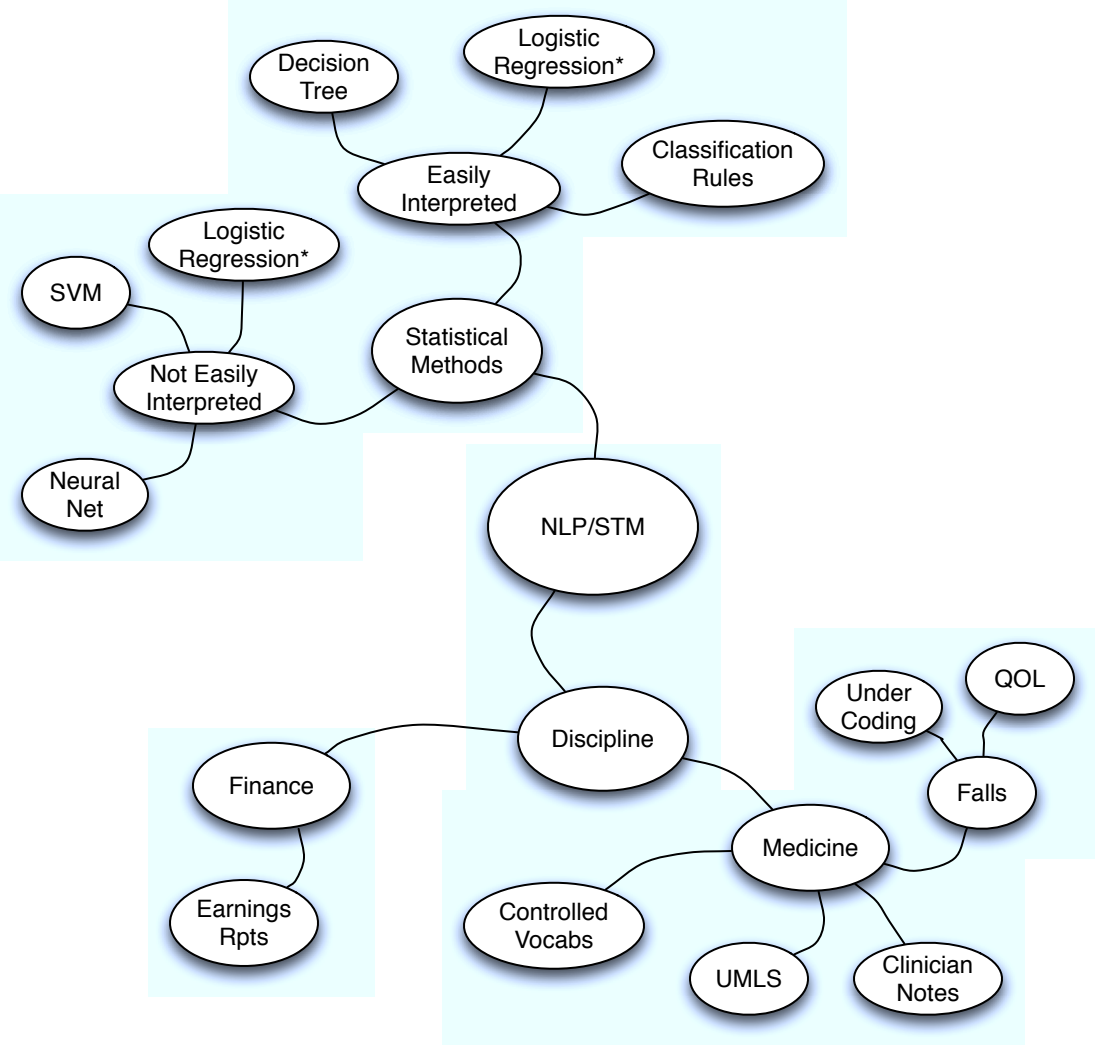


## **Appendix G**

### **Mind Map**

Based on the research methodology and background information, figure G.48 is a mind map of the research topic and how it motivates this dissertation as well as how the topic will be approached. The health care discipline was selected because of the availability of controlled vocabularies to allow for the extraction of the specialized language. Even though clinical progress notes are unstructured they are documented by clinicians in a very structured manner. Clinicians are taught to follow the SOAP (Subjective, Objective, Assessment, and Plan) method of documenting clinical encounters. A low complexity yet still relevant target was needed and fall-related injuries (FRIs) met both of these requirements. This allowed the processes to be developed without being hindered by the complexity of the target. A history of a previous fall is one of the most important clinical indicators that identifies an elderly patient as high risk for additional falls and targets them for fall prevention programs (Ganz et al., 2007). However, information about FRIs in administrative databases has been found to be significantly under-coded, thereby limiting a clinicians access to information about a history of falls (Luther et al., 2005). Therefore, the topic FRIs is very relevant.

As far as choosing a statistical method, the choices were either methods that were or were not easily interpreted by a person. In health care there are two types of systems that take advantage of predictive models; surveillance and Clinical Decision Support Systems (CDSS). Typically, in a surveillance system, the accuracy of the prediction model is the most important feature, however, in a CDSS, not only is the accuracy important but the interpretability of the resulting model is also important. In this dissertation we chose to implement rule-based models for their ease of interpretation. In the second experiment, rules will be created and from those rules, a classifier will be built. In the third experiment, this will be done in the opposite order. DTI will be used to create a classifier and from that classifier, rules will be created. Both methods are addressing the second goal of this dissertation, creating a rule-based classifier.



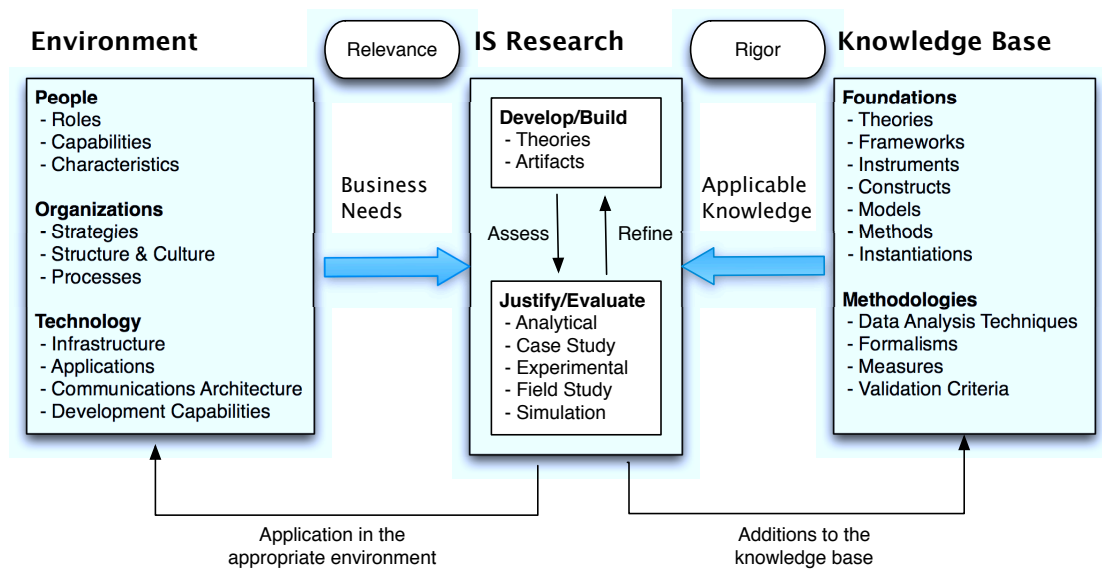
**Figure G.48.:** Dissertation Mind Map

## Appendix H

### Information Systems Research Framework

The research performed for this dissertation was conducted using the information systems research framework, specifically, the design science paradigm. The technologies and methods used to perform this research have their roots in Computational Linguistic, Computational Learning, and Information theories. These technologies and methods will be used to create the artifacts, which are classifiers. Classifiers were built using various machine learning (ML) techniques. These classifiers were also evaluated in how well they classified clinical progress notes.

“The goal of information systems research is to produce knowledge that enables the application of information technology for managerial and organizational purposes (Hevner and March, 2003).” Hevner et al. propose a conceptual framework for performing information systems (IS) research. A diagram mapping out this framework can be seen in figure H.49.



**Figure H.49.:** Information Systems Research Framework

This framework contains two paradigms that researchers use to perform IS research, behav-

ioral science and design science. These two research paradigms are circular in nature, and together these paradigms allow theories to be built that address phenomena related to business needs and artifacts to be built and evaluated that address those business needs. In the rigor versus relevance argument (Applegate, 2002), behavioral science addresses rigor with the development of theories and design science addresses relevance with the development of artifacts that address business needs. Within this framework, prior IS research and reference disciplines provide the foundations for future research. This dissertation utilizes the design science paradigm. The technologies and methods used to perform this research have their roots in information and computational learning theories. These methods and technologies will be used to create the artifacts, which are classifiers.

## H.1 Design Science

The design science paradigm defines IT artifacts as constructs, models, methods, and instantiations. Table H.35 shows the different IT artifacts used in this research and the artifacts marked by \*\* are the artifacts created by this research.

**Table H.35:** Dissertation IT Artifacts

<b>Artifact</b>	<b>Explanation</b>
Construct	Computational linguistic vocabulary
Model	Machine Learning Model
Methods **	Methodologies used to create classifiers
Instantiation **	Implementation of the classifier algorithms