Summer 2015

The role of confidence and diversity in dynamic ensemble class prediction systems

Şenay Yaşar Sağlam
University of Iowa

THE ROLE OF CONFIDENCE AND DIVERSITY IN DYNAMIC ENSEMBLE CLASS PREDICTION SYSTEMS

by

Şenay Yaşar Sağlam

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Business Administration
in the Graduate College of
The University of Iowa

August 2015

Thesis Supervisor: Professor W. Nick Street

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Şenay Yaşar Sağlam

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Business Administration at the August 2015 graduation.

Thesis Committee:

W. Nick Street, Thesis Supervisor

Qihang Lin

Jeffrey W. Ohlmann

Gautam Pant

Kang Zhao

ACKNOWLEDGEMENTS

I would like to sincerely thank my advisor Prof. Nick Street for his tremendous support and guidance for refining my Ph.D research during all stages of this study. Even after I moved to New Zealand, he believed that I could achieve this. Besides my advisor, I greatly appreciate the rest of my thesis committee: Prof. Gautam Pant, Prof. Jeffrey W Ohlmann, Prof. Quihang Lin, and Prof. Kang Zhao, for their time, help, and comments.

To all of my fellow graduate students, Charisse Madlock-Brown, Si-Chi Chin, Michael Rechenthin, Stacy Vocia, Fahrettin Cakir, Wenjun Wang, Mohammed Khoshneshin, Lian Duan, Ray Hylock, Jieqiu Chen, Justin Goodson, Kamal Lamsal, Amit Kumar Verma, Akpene Gbegnon, and Brian Almquist, I am grateful that I had the chance to get to know you, and to work alongside with you. I would like to thank for the support of my former Rockwell Collins coworkers: David Dietzel, Stephen Wilkinson-Gruber, Courtney Gilroy, Taras Dunduchenko, Donna Canavan, David Thatcher, and Earl Edwards. In particular, I am truly grateful to to my colleague and my best friend Scott R. Elliott for being there whenever I needed, day and night.

I would like to thank Barbara Carr and Renea Jay for all their assistance during my time at the University of Iowa.

This dissertation would not have been possible without the love and support of my parents, Hasibe and M. Sait Yaşar, who taught me to dream big and to never give up. I would like to thank my siblings, Sevgi Yaşar Gedik, Servet Yaşar, and Volkan Yaşar, my brother in-law Buğra Gedik, and my father in-law,Ömer Sağlam for their emotional support.

Finally, I wish to thank my beloved husband, Yiğit for his love, support, and patience. Without him, I would not be able to get through all these years and have this dissertation. When I doubted myself and felt down, he was always there to pick me up and to show me that I can finish this. He helped me focus on things. He always believed in me, supported me, and showed me the light at the end of the tunnel. I also would like to thank my awesome daughter, Azra. Her smile always lights up my day and gives me the strength I need. She has always been my angel and inspiration.

Thank you all for assisting me to turn this dream into reality.

ABSTRACT

Classification is a data mining problem that arises in many real-world applications. A popular approach to tackle these classification problems is using an ensemble of classifiers that combines the collective knowledge of several classifiers. Most popular methods create a static ensemble, in which a single ensemble is constructed or chosen from a pool of classifiers and used for all new data instances. Two factors that have been frequently used to construct a static ensemble are the accuracy of and diversity among the individual classifiers. There have been many studies investigating how these factors should be combined and how much diversity is required to increase the ensemble's performance. These results have concluded that it is not trivial to build a static ensemble that generalizes well. Recently, a different approach has been undertaken: dynamic ensemble construction. Using a different set of classifiers for each new data instance rather than a single static ensemble of classifiers may increase performance since the dynamic ensemble is not required to generalize across the feature space. Most studies on dynamic ensembles focus on classifiers' competency in the local region in which a new data instance resides or agreement among the classifiers. In this thesis, we propose several other approaches for dynamic class prediction.

Existing methods focus on assigned labels or their correctness. We hypothesize that using the class probability estimates returned by the classifiers can enhance our estimate of the competency of classifiers on the prediction. We focus on how to use class prediction probabilities (confidence) along with accuracy and diversity to create dynamic ensembles and analyze the contribution of confidence to the system. Our results show that confidence is a significant factor in the dynamic setting. However, it is still unclear how *accurate*, *diverse*, and *confident* ensemble can best be formed to increase the prediction capability of the system.

Second, we propose a system for dynamic ensemble classification based on a new distance measure to evaluate the distance between data instances. We first map data instances into a space defined by the class probability estimates from a pool of two-class classifiers. We dynamically select classifiers (features) and the k-nearest neighbors of a new instance by minimizing the distance between the neighbors and the new instance in a two-step framework. Results of our experiments show that our measure is effective for finding similar instances and our framework helps making more accurate predictions.

Classifiers' agreement in the region where a new data instance resides has been considered a major factor in dynamic ensembles. We postulate that the classifiers chosen for a dynamic ensemble should behave similarly in the region in which the new instance resides, but differently outside of this area. In other words, we hypothesize that high local accuracy, combined with high diversity in other regions, is desirable. To verify the validity of this hypothesis we propose two approaches. The first approach focuses on finding the k-nearest data instances to the new instance, which then defines a neighborhood, and maximizes simultaneously local accuracy and distant diversity, based on data instances outside of the neighborhood. The second method considers all data instances to be in the neighborhood, and assigns them weights depending on the distance to the new instance. We demonstrate through several experiments that weighted distant diversity and weighted local accuracy outperform all benchmark methods.

PUBLIC ABSTRACT

In classification problems, observations fall into preassigned groups. Examples include identifying customers who would buy a product, and detecting whether a credit card expense is made by a customer. A popular approach to tackle these problems is using a collection of models that combines the collective knowledge of them. It has been shown that employing multiple models outperforms a single model. A common approach has been to use the same collection for all observations, which is also known as the static approach. Recently, there have been more attempts in using a different collection that is more specialized for each observation, depending on the features of observations. This is referred to as the dynamic approach.

In this thesis, we adopt the dynamic approach and explore what sort of characteristics we would like our models or the collections to exhibit. Two factors have been used frequently in the literature: accuracy and diversity of models. The second factor is about how different models are in a collection. In addition to these two factors, we consider a third one: confidence of models. First, we investigate to what extent confidence can enhance the competency of models on the prediction. Second, we propose a new measure in the dynamic approach to evaluate the similarity between observations. We show that our measure is effective for finding similar observations and our framework helps making more accurate predictions. Finally, we return our attention to the diversity factor and analyze how diversity should be assessed in a dynamic setting.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1
INTRODUCTION

Data mining is the automatic or semi-automatic process of extracting previously unknown interesting patterns (i.e. cluster analysis, anomaly detection), or predicting trends and behaviors (i.e. classification, association rule mining) using large quantities of data. Data may originate from a variety of sources, so numerous applications of data mining tasks may be employed depending on the object of interest. Anomaly detection, which identifies candidates for unusual data, is often used for credit card fraud detection, network intrusion, and severe weather prediction. To find which products are frequently bought or bundled together, association learning method such as market basket analysis are suitable to create an advertisement plan. Another example is cluster analysis, which could be used to recommend new movies based on consumer preferences or to group genes with related expression patterns.

Classification is a common data mining task arising in many real-world applications. In a classification problem, each observation is assumed to be a member in exactly one of k classes. A classification method then predicts the membership of the observations. Examples include identifying customers who are likely to buy a particular product in a supermarket or to select a particular movie, and determining which expenses in an account are fraudulent. Due to its wide range of applications, there is an extensive literature studying classification problems [2, 11, 30, 28, 43, 59, 67, 71].

Classifier performance depends heavily on the characteristics of the data. There is no single classifier that works best on all problems. Therefore, various classification algorithms have been proposed. Examples of these algorithms include neural networks [2, 30], support vector machines [8, 71], k-nearest neighbors [11, 28], naive Bayes [31, 43], and decision tree [59, 61].

There have been numerous empirical studies that compare classifiers' performances [21, 36] and discuss how this evaluation should be carried out [14, 63]. Studies have shown that it is impossible to model a classifier that perfectly generalizes the data and always predicts correctly. The no-free lunch theorem also states that "there is no strategy of any kind that outperforms all others on all problems." [33]. To solve this issue, the focus has shifted from building an expert classifier to creating multiple classifier systems, which are also known as ensemble systems. The goal of the ensemble systems is to improve the performance of the system in making the right decision, by combining the decisions of multiple classifiers to reach a final decision. The assumption here is that classifiers in an ensemble compensate each other's incorrect predictions, so they decrease the error rate of the whole system. Therefore, when building an ensemble, each classifier should be different from the others as there are no gains from having the same classifier multiple times in the ensemble. The accuracy of each classifier's decision is expected to show some degree of variability. This issue raises several questions regarding the characteristics of an ensemble system:

1. How do we generate different classifiers?

2. How can we measure the diversity among classifiers?

3. What is the desired level of diversity for the classifiers in an ensemble system?

4. How do we combine the predictions of the classifiers in an effective ensemble system?

5. How many classifiers are needed to generate an ensemble system?

6. Should we have a different ensemble for each test instance?

Over the years, several algorithms and methods have been proposed to address each one of these questions. Breiman's bagging [3], Freund and Schapires' boosting [24], and Ho's random subspace methods [32] are most common algorithms used for ensemble generation. Bagging and boosting methods manipulate the training data in order to generate a diverse ensemble. Bagging generates several training sets by sampling with replacement from the original training set. Unlike bagging, boosting may use all instances. At each iteration, it assigns a weight to each instance in the training set that reflects its importance. These weights are then adjusted per the performance of the classifiers from previous iterations. It favors instances that are previously misclassified. Rather than choosing or weighting training samples, the random subspace method [32] modifies the feature space. It randomly selects features without replacement from original feature set. In all of the cases, the predictions from all of the members are combined based on some voting scheme.

There have been many empirical studies comparing ensemble generation methods [1, 17, 66, 60]. These studies compared these three methods and analyzed the conditions under which each method is more effective. However, there are several issues that remain to be solved, including how to decide the size of the ensemble and how to control the diversity among the classifiers. Breiman provided some insights on the size of the ensemble, and some pruning algorithms have been proposed for Boosting. Hansen and Salamon [29] argue that if the members of an ensemble perform better than random guessing and they are diverse enough, then an ensemble performs better than its members. In the last couple of decades, several studies have proposed diversity measures [7, 12, 29, 32, 62, 78] and have used diversity explicitly to build an ensemble in an attempt to obtain better performance and to control the size of the ensemble. In bagging and boosting, the growth of an ensemble could be stopped when diversity and accuracy satisfy a certain condition. Another way of using diversity explicitly is by adapting the *overproduce and choose paradigm* [18, 26, 49, 50]. It involves generating a pool of classifiers and selecting classifiers that are most diverse and accurate rather than generating a certain number of classifiers to form an ensemble system as in boosting. Several studies have been conducted to explain the relationship between diversity and the ensemble accuracy and there have been conflicting results regarding usefulness of the diversity [6, 13, 35, 39, 42, 68].

After selecting a set of diverse classifiers, the question of how to combine the classifiers has drawn attention. There are three main strategies: fusion, selection, and stacking. In the fusion approach, average and majority voting schemes are popular. The selection approach is based on classifiers' local competence and it chooses a classifier from the set to label the input. Unlike these first two approaches, the stacking approach trains another classifier by using the output of the members of an ensemble and uses its outcome to label the data [74].

Most studies have focused on finding a static set of classifiers that generalizes well to make a prediction. Recently, there have been studies regarding how to utilize the pool of classifiers more effectively in the overproduce-and-choose paradigm. Since the classifiers in the pool are different from each other, it is reasonable to deduce that they are experts in some region of data. This refers to the last question above regarding a different ensemble for each test instance, so dynamic classifier and ensemble selection methods have been proposed. Some of these methods treat the classifiers' local competence in the region as an indicator for classifier confidence to correctly predict the data point [16, 27, 80]. Other methods use predictions or class prediction probabilities returned by the classifiers to measure the confidence of an ensemble and to identify which ensemble works best for a given instance [18, 20, 37, 44, 45]. In this paper, we propose methods to utilize the initial pool of classifiers to increase the prediction capability of the system. More specifically, we focus on how to use class prediction probabilities (class supports) returned by the classifiers in the pool to create dynamic ensembles, to measure similarity between data points, and to measure diversity for building an ensemble.

The remainder of this paper is organized as follows. We discuss the literature which explores the improvements in the field of ensemble systems in Chapter 2. Chapter 3 introduces the first method for creating dynamic ensembles. In doing so, we investigate the role of confidence in dynamic ensemble creation. In Chapter 4, we propose a method to map data instances to classifier-based space by using the pool of classifiers. We define two distance measures to evaluate similarity among data instances. Then, we propose a framework for dynamic class prediction using these measures. In Chapter 5, we return our attention to diversity among classifiers. Since diversity affects generalization capability of a static ensemble, we explore to what extent diversity contributes to prediction capability of dynamic ensembles. Finally, we conclude our results and propose possible future directions in Chapter 6.

CHAPTER 2
LITERATURE REVIEW

This chapter explains the improvements in the field of ensemble systems and their advantages and shortcomings. First, in Section 2.1 we discuss the three ensemble generation methods used in this dissertation: Bagging, Random Subspace Method (RSM), and Stacking. In Section 2.2, we describe two base learning algorithms: Support Vector Machine (SVM) and k-Nearest Neighbor (k-NN). In particular, we choose the SVM method to train the base classifiers in our experiments, even though Bagging and RSM are designed for and usually applied to decision trees. We will explain our rationale for the use of SVM as well as the procedure in Section 2.2.1. Later in Section 2.2.2, the k-NN algorithm is discussed as a new similarity measure based on class probabilities for data points is proposed in Chapter 4 and k-NN is used to demonstrate the effectiveness of the measures.

There have been several studies in the literature regarding the factors that contribute to the generalization ability of an ensemble. The first is the accuracy of the base classifiers. It is clear that constructing an ensemble with the most accurate classifiers help lower the generalization error. Furthermore, since classifiers must make different errors in order for the ensemble to be effective, diversity among the members of an ensemble is also important. In Section 2.3, we survey the literature for diversity measures and their effectiveness.

As many recent studies have shown, we believe that each test instance should be treated differently, so we suggest using different classifiers/ensembles to predict the label of an unknown instance. In Section 2.4, we review some of the important studies on *dynamic* classifier and ensemble selection methods.

2.1 Ensemble Creation Methods

Ensemble creation methods can be divided into three categories based on how base classifiers are constructed. One way to force a learning algorithm to construct multiple hypotheses is to run the algorithm several times and provide it with somewhat different data in each run with a single learning method (e.g., Bagging). Another way is to provide different training parameters for the learning algorithm (e.g., using different initial weights for each neural network in an ensemble). A third way is to use a different learning algorithm to generate diverse set of classifiers using the same data to build an ensemble (e.g., Stacking). Bagging, Boosting, and Random Subspace methods have been the most commonly used methods. Several studies compare and contrast their performances and show that they are most beneficial for regression and decision trees [3, 24, 32].

It was demonstrated that boosting often gives a better performance than bagging, and the RSM may outperform them both. In Breiman [4], it was stated that bagging and boosting are useful for unstable classifiers as they reduce the variance of the classifiers. However, it was shown that a large variance is not a requirement for boosting to be effective. Here we will only discuss Bagging, Random Subspace Method, and Stacking.

2.1.1 Bagging

The Bagging (bootstrap aggregating) method was proposed by Breiman [3] to generate diverse classifiers by manipulating the training data. The idea behind bagging is to sample data with replacement from the original data set and to train a base classifier on the sample data set. Let $\mathbf{X}$ denote the instance space and $\mathbf{Y}$ the set of class labels. Assume $\mathbf{Y} = \{-1, +1\}$. A training set $\mathbf{D} = (x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$ where $x_i \in \mathbf{X}$ and $y_i \in \mathbf{Y}(i = 1, ..., N)$. Each training sample x_i is a p-dimensional vector $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$. The algorithm specified in [3] can be summarized as follows:

Algorithm 2.1 Bagging Algorithm

```
1. procedure BAGGING PROCEDURE
2.     Start with a training set $\mathcal{D}$ of $N$ instances.
3.     for $j = 1 : 1 : M$ do
4.         Uniformly draw $N$ observations with replacement from $\mathcal{D}$.
5.         Define the new set of observations as $\mathcal{D}_j$.
6.         Construct classifier $\mathcal{C}_j$ using $\mathcal{D}_j$.
```

To classify an observation, use the majority class prediction among the M base classifiers. It has been shown that Bagging is most useful when the learning algorithm is unstable. In other words, Bagging works primarily with the learning algorithms whose performance changes with a small change in the training data. Bagging reduces/eliminates the instability in learning algorithms (variance) and decreases the classification error by averaging base classifiers' errors [3].

2.1.2 Random Subspace Sampling

The Random Subspace Method [32] randomly selects subsets of features to be used in constructing classifiers inside this reduced subspace. More specifically, in the original study [32], 50% of the available features were selected for each decision tree. 100 decision trees were created using that method. Given the data description in Section 2.1.1, the algorithm specified in [32] can be summarized as follows:

Algorithm 2.2 Random Subspace Sampling Algorithm

```
1. procedure RANDOM SUBSPACE ALGORITHM
2.   Start with a training set $\mathcal{D}$ of $N$ instances with $p$ features.
3.   for $j = 1 : 1 : M$ do
4.     Uniformly draw $p^* < p$ features without replacement from $\mathcal{D}$.
5.     Define the new set of $N$ observations with $p^*$ features as $\mathcal{D}_j$.
6.     Construct classifier $\mathcal{C}_j$ using $\mathcal{D}_j$.
```

The aggregation is performed using weighted voting on the basis of the base classifiers' accuracy. Ho [32] showed that RSM is most effective when the data set has a large number of features and sample sizes compared to Boosting and Bagging. The RSM is not good when the data set has a small number of samples with very few features. Later, Kuncheva and Whitaker [41] performed an experiment with a systematic partition of a set of 10 features into three subsets containing (4,4,2) or (4,3,3) features to generate an ensemble with three base classifiers to investigate what kind of improvement should be expected from an ensemble based on the best or worst classifier accuracy. Also, Kuncheva and Whitaker [41] used nine different combination schemes including majority vote and behavior-knowledge space, to analyze whether the choice of the combination method would be a factor. The experiments showed that there are no best combinations for all situations and that there is no assurance that in all cases an ensemble will outperform the single best individual.

2.1.3 Stacking

Another approach to generate classifiers is by applying different learning algorithms (with heterogenous model representations) to a single data set. More complicated methods for combining classifiers are typically used in these settings. Stacking [74] is often used to learn a combining method in addition to the ensemble of classifiers. In step 1 of Stacking, classifiers are generated by using different learning algorithms on a single data set. In step 2, a meta-level classifier is learned by using the outputs of the base-level classifiers as the input attributes. The target attribute remains as in the original data set. Stacking [74] and SCANN [53] used classifiers' prediction as the meta-level data. Ting and Witten [69] extended Stacking by using the probabilities predicted for each possible class by the base classifiers to form the meta-level classifier. As a meta classifier, a variant of least squares linear regression adapted for classification tasks was used. It performs worse on multi-class than on two-class data sets.

Stacking performance can be improved by using output probabilities for every class label from the base-level classifiers. It has been shown that with stacking, the ensemble performs (at best) comparably to selecting the best classifier from the ensemble by cross validation [22].

StackingC [65] is a Stacking variation. StackingC builds a meta model for each class label unlike Stacking. In StackingC, each base classifier is trained and tested on the output probabilities returned for that particular class while stacking output probabilities for all classes and from all base classifiers. The target attribute is modified to represent whether the data instance belongs to that class label. StackingC is shown to outperform Stacking especially for multi-class data sets. The reason is that the dimensionality of the meta data is reduced by a factor equal to the number of classes. Considering fewer attributes in the meta data also makes the meta classifier learning phase faster. StackingC also resolves the weakness of Stacking in the extension proposed by Ting and Witten [69] and offers a balanced performance on two-class and multi-class data sets.

As mentioned, during the training phase of the base classifier, StackingC uses *one-against-all class binarization*. This class binarization is believed to be a problematic method especially when class distribution is non-symmetric. An alternative to one-against-all class binarization has been proposed by Lu and Yao [48]. It is called *the one-against-one binarization*. In this method, a multiple class problem is converted into a series of two-class problems by training one classifier for each pair of classes, using only training instances of these two classes and ignoring all others. A new instance is classified by submitting it to each of the binary classifiers, and combining their predictions. By using this binarization method, Menahem et al. [52] develop three-stage stacking algorithm named Troika.

2.2 Base Learning Algorithms

As mentioned, the choice of base classifier affects the performance of the ensemble. Also, it is important to decide which ensemble creation method should be used. Although most of the early studies focused on decision trees or regression models as the base learning algorithm, in this section, we will discuss only two learning algorithms: k-Nearest Neighbors (k-NN) and Support Vector Machine (SVM).

2.2.1 Support Vector Machine

Support vector machines (SVMs) [71] are considered among the best classification algorithms; robust and accurate. Its success relies on margin maximization. In a two-class learning task, the aim of SVM is to find the best classification function to distinguish between members of the two classes in the training data. For a linearly separable data set, a linear classification function corresponds to a separating hyperplane $f(x)$ that passes between the two classes, separating them. Once this function is determined, new data instance x_n can be classified by simply testing the sign of the function $f(x_n)$; x_n belongs to the positive class if $f(x_n) > 0$. Because there are many such hyperplanes, SVM chooses the hyperplane that maximizes the margin between the two classes. This problem can be formulated as:

$$\min_{w} \left\{ \frac{\|\mathbf{w}\|^2}{2} \right\} \tag{2.1}$$

$$\text{s.t. } y_i \left(\mathbf{w} \cdot \mathbf{x_i} + b\right) \geq 1, \forall i = 1, \ldots, l$$

where $\mathbf{w}$ is the vector of attribute weights and y_i is the class label of the data point x_i.

This formulation can be modified to handle cases where data is not completely linearly separable. In other words, no matter which hyperplane is drawn to be the decision boundary, there will always be some data points which will be on the opposite side of the decision boundary. The "soft margin" approach was introduced by Vapnik [71]. With this approach, some misclassifications are allowed and SVM tries to minimize the cost associated with them. Therefore, the modified formulation includes a slack variable ξ, which provides an estimate of the error of the decision boundary on the training instance. Then, the objective function is changed to penalize the misclassification to avoid having a wide margin with too many misclassifications. The parameter C is user-defined to represent the penalty of misclassifying the training instances. It is a regularization term, which provides a way to control overfitting: as C becomes large, the less the training error will be. However, if the value of C is increased too much, the generalization capability of the classifier will be decreased.

$$\min_{w} \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^{N} \xi_i \tag{2.2}$$

$$\text{s.t. } y_i \left(\mathbf{w} \cdot \mathbf{x_i} + b\right) + \xi_i \geq 1, \forall i = 1, \ldots, l$$

$$\xi_i \geq 0, \forall i = 1, \ldots, l.$$

The SVM formulation can also be extended to handle cases where the training instances cannot be separated by a linear decision boundary. This can be done by transforming data into a new space which will separate the training instances (points) more so that a linear decision boundary can be drawn in this new space. Performing all the computations in the transformed space especially finding the similarity (dot products) of two instances could be problematic. However, the "Kernel trick" is proposed to compute the dot product of training instances, $\Phi(X_i) \cdot \Phi(X_j)$, in the transformed space using the original attribute set, where $\Phi(X)$ is a nonlinear mapping function applied to transform the training instances. Simply, computing the dot product on the transformed data instances is equivalent to applying a kernel function to the original data. Many approaches have been proposed on how to define the kernel functions [64].

$$\min_{w} \frac{\|\mathbf{w}\|^2}{2} \tag{2.3}$$

$$\text{s.t. } y_i \left(\mathbf{w} \cdot \Phi(\mathbf{x_i}) + b\right) \geq 1, \forall i = 1, ..., l.$$

Even though SVM is computationally inefficient, it is stable, performs well, and can be used for ranking elements [76].

2.2.2 kNN: k-Nearest Neighbor Classification

Given a training set D and a new instance (t), k-nearest neighbor classification [11, 28] calculates the distance of the test instance to the training instances ($x_i \in D$), finds a group of k data points in the training set that are "closest" or "most similar" to the test instance, and the class labels of these nearest neighbors are then used to determine the predicted class label of the test instance. Given the data description in Section 2.1.1, the algorithm can be summarized as follows:

Algorithm 2.3 kNN Algorithm

```
1. procedure kNN PROCEDURE
2.   Given a data instance,t, a training set $\mathcal{D}$ of $N$ instances and a distance function, f.
3.     for $i = 1 : N$ do
4.       Calculate the distance between $(x_t, y_t)$ and $(x_i, y_i)$.
5.     endfor
6.       Find the $k$ closest instances to the new instance $t$.
7.       Define the set of $k$ closest observations as $\mathcal{D}_t$.
8.       Calculate the predicted label $\hat{y}_t = argmax_z \sum_{(x_i,y_i) \in D_t} I(z = y_i)$
```

There are several key issues that affect the performance of k-NN: the value of k, the distance measure, and the fusion method. The choice of k is important because if k is a very small number, then the result can be sensitive to noise. On the other hand, if k is a very large number, then the neighborhood defined for the test instance may include many instances from other class labels. Another issue is the choice of distance measure (e.g. Euclidean distance, cosine similarity, Pearson's correlation). The distance measure should be chosen carefully as it affects the neighborhood of the test instance. This choice should be done based on the attribute data types, correlation between attributes, and homogeneity of the attributes [67]. For instance, if the data has all binary attributes then using Jaccard similarity may be more appropriate. However, if the data is sparse, cosine similarity may provide more information regarding similarity between two instances.

Some distance measures can also be affected by the high dimensionality of the data. In particular, it is well known that the Euclidean distance measure become less discriminating as the number of attributes increases. In case of the attributes having different scales, a standardization method should be used to prevent distance measures from being dominated by one of the attributes. Sometimes, the attributes in the data have different data types. A general approach to solve that issue is computing similarity between data points for each attribute separately and then using the average of all of similarities to find the overall similarity. In short, the type of distance measure should fit the type of data.

Another issue that affects the performance of k-NN is the approach to combining the class labels. The simplest method is to take a majority vote, but this can be a problem if the nearest neighbors vary widely in their distance as closer neighbors may be more reliable in terms of indicating the class label of the test instance. Weighing each neighbor's vote by the reciprocal of its distance to the test instance is more desirable and less sensitive to the choice of k. Also, attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes.

k-NN classifiers are lazy learners, that is, models are not built explicitly unlike decision trees, SVMs, etc. Thus, classifying a test instance is costly since it requires the computation of the distance of the test instance to all the instances in the training set to find the k nearest neighbors of the test instance. A number of techniques have been developed for finding the k nearest neighbors efficiently. Although there are some challenges associated with the k-NN algorithm, it is still desirable since it is easy to understand and to implement. In addition, there are no underlying assumptions regarding the distribution of the data. Despite its simplicity, it can perform well in many situations. The results of the study by [11] shows that the error of k-NN method asymptotically approaches that of the Bayes error as N goes to infinity.

2.3 Diversity

Ensembles of classifiers have garnered great interest in recent years as it has been shown by several studies that, both theoretically and empirically, they can outperform single classifiers when the members of the ensemble are as accurate as possible and make few coincident errors. Since it is highly unlikely to train the perfect classifier that makes no errors, we need an ensemble of classifiers in which members make different errors to complement each other. For instance, Zhang et al. [78] demonstrated that an ensemble of three identical classifiers with 95% accuracy is worse than an ensemble of three classifiers with 67% accuracy and least pairwise correlated error. Due to this reason, diversity among the ensemble members is crucial for having fewer prediction errors.

There are several methods proposed in the literature to generate a diverse set of classifiers. These studies can be divided into two categories based on how they create diversity in an ensemble: implicitly and explicitly. Methods such as Bagging [3], Boosting [24], and Random Forest [4] introduce diversity by sub-sampling or re-weighting the training instances for the base classifiers. Two problems with these methods are that it is hard to decide how diverse the classifiers are and what the ensemble size should be. Due to this reason, measuring diversity and controlling it to increase the performance of an ensemble have become important. To answer the first question, several diversity measures have been proposed, some of which are borrowed from statistics, including *disagreement measure*, *Q-statistics*, and *double fault*. Some of the diversity measures proposed in the literature are *Kohavi-Wolpert* [38], *generalized diversity* [57], *interrater agreement* [23], *measure of difficulty* [29], *conditional double fault* [78], etc.

To answer the second question, the following studies focused on using diversity in the process of pruning an ensemble or building an ensemble. Ensemble pruning methods build a pool of classifiers and chooses a subset of individual classifiers from this pool to form a smaller ensemble for prediction. The classifiers in this new ensemble need to be carefully chosen so that it will perform similar to or better than the original ensemble. This problem is NP-hard as there are $2^M - 1$ possible subsets of an ensemble of size M. Several studies related to ensemble pruning formulated it as mathematical or optimization problem [56, 55, 78, 79]. Studies in this field mainly differ from each other based on the choice of the diversity measure, the search algorithm, and the fitness function for leading the search. For example, [50] considered Kappa statistics. However, Giacinto and Roli [26] used the *Q-statistic* as a diversity measure. Zhou et al. [79] employed genetic algorithms and Fu et al. [25] used particle swarm optimization to select an optimal set of individual neural networks.

Margineantu and Dietterich [50] suggested a Kappa-error plot for which diversity of every pair of classifiers in an ensemble is plotted against average of the individual errors of the two classifiers. This plot showed that best pairs are the ones with low errors and high diversity. However, the shape of the curve also revealed that there is a trade-off between the accuracy of the pair and its diversity. So, the success of ensemble pruning methods lies in balanced accuracy/diversity trade-off, where choosing only the most diverse classifiers or the most accurate individual classifiers to form the sub-ensemble decreases the generalization capability.

Despite the fact that ensemble pruning methods discussed here increased the performance of the system, a pool of classifiers and an effective search strategy are needed for these methods, which are time consuming. Several studies have been proposed to explicitly integrate diversity into the ensemble building. Liu et al. [47] simultaneously train neural networks in an ensemble using negative correlation learning to make individual networks negatively correlated, hence diverse. Opitz et al. [56, 55] first create an initial set of networks, then produce new networks by crossing over the nodes of two networks and adding new nodes to the child networks as a mutation step to increase the diversity of the population. To guide the search they use an objective function that incorporates both accuracy and diversity terms. Rather than focusing on the model parameters, Tumer et al. [70] reduce the correlation between classifiers in an ensemble by exposing them to different feature subsets. M classifiers are trained, one corresponding to each class in an M-class problem. For each class, a subset of features that have a low correlation to that class is eliminated. The degree of correlation between classifiers is controlled by the number of features that is eliminated. Zenobi and Cunningham [77] also build ensembles based on different feature subsets. In their approach, feature selection is done using a hill-climbing strategy based on classifier error and diversity. A classifier is rejected if the improvement of one of the metrics leads to a considerable decrease of the other per a pre-set threshold.

Melville and Mooney [51] proposed a method called DECORATE which increases diversity of an ensemble by making use of artificial data to train the base classifiers. At each iteration, DECORATE generates artificial data based on the performance of the current ensemble's response. More specifically, the labels for these artificially generated training instances are chosen so as to differ maximally from the current ensemble's predictions. Then artificial data is added to the original training data to train a new classifier. They compared the diversity with the ensemble error reduction, i.e., the difference between the average error of the ensemble members and the error of the entire ensemble by computing Spearman's rank correlation between the two. It was claimed that the fairly strong correlation between the two is another indication for increasing ensemble diversity to reduce generalization error.

Even though there have been several studies related to diversity, there is no agreed upon diversity definition in the literature. This shifted the focus from how to generate diverse ensembles to how these measures relate to each other and how they affect accuracy. To answer that question, the relationship between ensemble performance and diversity has been put under a microscope. Kuncheva et al. [42] compared ten diversity measures. All of the measures are based on oracle outputs and the majority vote rule. The Oracle output only concerns whether a sample is classified correctly or not. Hence, the Oracle output provides a general model for analyzing a classifier ensemble, and conclusions drawn on this model can be easily generalized to various ensemble learning methods. Kuncheva et al. found that these measures are highly correlated. However, they could not reach a conclusion on how to utilize diversity for the production of effective classifier ensembles.

Later, Tang et al. [68] presented an in-depth analysis of six of the diversity measures discussed in [42] and showed their relationship with the minimum margin of an ensemble. They analyzed how generalization of an ensemble changed with respect to diversity when the average accuracy of the base classifiers is fixed, and how diversity and the average base classifier accuracy interact with each other. Based on the experiments, the claim was that exploiting diversity measures to seek diversity explicitly is ineffective. First, the change of measured diversity cannot provide consistent guidance on whether a set of base classifiers has good generalization performance. Second, the diversity measures are negatively correlated to the average accuracy of the base classifiers. Finally, it was shown that if the average accuracy of the base classifiers is regarded as a constant and the maximum diversity is achievable, maximizing the diversity among the base classifiers is equivalent to maximizing the minimum margin of the ensemble on the training samples. Since the true distribution of the data is unknown, it was suggested that like SVM a diversity measure should contain a regularization term and all of the discussed diversity measures contain no regularization term. Therefore, even if these existing diversity measures can be maximized, the achieved ensemble may overfit.

Kapp et al. [35] also investigated the relationship between diversity and margin theory from the perspective of which margin definition is used. There are three main measures related to margin theory: minimum margin, cumulative margin distributions, and average margin. They show that individual performances of the base classifiers is one factor that contributes to the overall ensemble performance, but it is not sufficient. Thus, some diversity is needed to get the highest majority vote performance. Experimental results showed that the average margin is stable and minimum margin is unstable. But, maximizing average margin chooses the ensembles with the strongest individual members in a given pool. This leads to low diversity as the base classifiers chosen in this manner will be very similar. Rather than using average margin, it is recommended using Chebyshev's inequality (CI) which states that probability of error should be less than the variance of margins divided by average margin squared. This implies that the ensembles must be sufficiently confident on their decisions with a certain majority vote, at least 50% in average.

De Olivieria et al. [13] addressed the accuracy/diversity dilemma for heterogenous ensembles using single and multi-objective GA approaches for analyzing accuracy and diversity separately, and jointly. The Q-statistic is used to measure diversity of the ensemble. It was concluded that similar to homogeneous systems the combination of diversity and accuracy can lead to more accurate ensemble systems, when compared with these two parameters individually. Even though these studies attempted to show the relationship between accuracy and diversity and to answer how much diversity is enough, these questions are not completely answered.

Hsu and Srivasta [34] changed the direction of the focus and analyzed the relationship between diversity and correlation of the classifiers instead of ensemble accuracy. The relationship between diversity and correlation of the classifiers in ensembles was analyzed theoretically and formulated in a nonlinear function. They were able to derive a critical value for disagreement measure. It was shown that before the critical value, higher diversity reduces correlation which is usually associated with a better ensemble. When diversity crosses the critical point, increasing diversity increases the correlation while highly correlated classifiers usually correspond to an inferior ensemble. Brown [5] investigated the relationship between accuracy and diversity by decomposing the ensemble's mutual information into accuracy and diversity terms and showed that the diversity of an ensemble exists at multiple orders of correlation. However, estimating the interaction of these multiple correlations is complicated and there is no proposal for applying this in practice. Later good and bad diversity concepts were introduced, and their relationship with the upper/lower limits were defined on majority voting error by Brown and Kuncheva [6]. Here, the majority vote error is divided into three components: average individual accuracy, “good” diversity and “bad” diversity. The two diversity terms are related to the majority vote limits. Good diversity is derived to be the number of incorrect votes when the ensemble is correct. On the other hand, bad diversity is the number of correct votes when the ensemble is incorrect. They argued that a diversity measure should be naturally derived as a direct consequence of two factors: the loss function and the combiner function. It is recommended to construct algorithms like DECORATE as using artificially constructed data examples produces diversity in majority voting by making the individuals disagree wherever possible with the ensemble.

2.4 Dynamic Ensemble Selection

Most existing methods construct static ensembles, in which only one ensemble is chosen from a pool of classifiers and is used for all new data instances. Recently, there have been studies in which each new data instance is treated individually. Since different instances are often associated with different classification difficulties, it is hypothesized that using different classifiers for the classification task rather than a single static ensemble of classifiers can increase performance. Earlier studies regarding this dynamic scheme focus on selecting a classifier based on different features or different regions of the instances defined based on similarities among them [16, 15, 27, 42, 75].

Woods et al. [75] proposed a method called *Dynamic Classifier Selection by Local Accuracy (DCS-LA)* which estimates each classifier's accuracy in a local region of the original feature space surrounding the new instance, and then use the decision of the most locally accurate classifier. The *DCS* method proposed in [27] differs from the DCS-LA with respect to how the local region of a new instance is defined. Giacinto and Roli [27] defined the similarity between two data instances in the classifier-based space, also known as multiple classifier behavior (MCB). The defined similarity measure considers the number of classifiers which assign the same label to the instances. This measure is later referred as Template Matching (TM) in [9]. The training instances that are more similar to the test instance than some threshold value are selected to form the local region of the new instance. As the performance of these two studies were affected by the choice of k, Didaci and Giacinto [15] investigated the benefits of using an adaptive metric, $DANN$, for finding neighbors of a new instance, and a dynamic choice of the value of k. Based on their experimental results, it was shown that the performance of *DCS-LA* based on local accuracy estimates can be improved by using an adaptive metric instead of Euclidean or city block distances.

These studies [75, 27, 15] showed the effectiveness of the *DCS-LA* approach. However, it was outperformed by the oracle. Didaci et al. [16] claimed that the performance of an oracle was not a realistic upper bound to compare the *DCS-LA*'s performance against. Thus, they proposed three upper bounds for *DCS-LA* to provide a more realistic limit. Oracle k-best upper bound exhibits the maximum accuracy attainable using the considered selection mechanism, under the assumption that, for each new instance, the optimal size of the local region can be estimated correctly. On the other hand, oracle $k\text{-}best_{AVE}$ makes the assumption that the optimal k parameter for the *LA* could be estimated. Oracle $\theta\text{-}best_{AVE}$ is similar to oracle $k\text{-}best_{AVE}$ but it is for the adaptive distance metric. It was shown that the considered *DCS-LA* methods performed close to the proposed upper bounds. Also, results demonstrated that a size-adaptive local region can boost the performances of *DCS-LA* mechanism.

To speed up the process of defining the neighborhood of a given test instance, Zhu et al. [80] proposed a method called *Attribute-Oriented Dynamic Classifier Selection (AO-DCS)*. This method first statistically partitioned the evaluation set into disjoint subsets by using the attribute values of the instances. In other words, for a given attribute, the evaluation set is divided into disjoint partitions and this step was repeated for all attributes. Then, the performance of the trained classifiers on these disjoint subsets was calculated. Finally, based on new instance's attribute values, the corresponding disjoint subsets were decided and the "best" performing classifier on these subsets was selected to classify the test instance. Kuncheva [39] took a similar approach by combining *Cluster and Select*, and

Decision Templates. In [39], data is first divided using K-means clustering and trained base classifiers are evaluated on these clusters. Unlike [80], it was argued that the classifier which performs statistically significantly better than all the other classifiers should be considered as a local expert. For each cluster, it was determined whether the best classifier is dominant in the region by looking at the confidence interval overlap between the best classifier and the other classifiers. If there is no overlap, the best classifier is in charge of making predictions for that cluster. For those clusters which do not have dominant classifiers, the *Decision Templates* model is presented in [40]. However, the results show that the proposed scheme is not better than the performance of the nearest neighbor classifier.

Similar to the static classifier selection methods, the drawback of these methods is that the choice of a single individual classifier over the rest depends on how much we trust in that classifier's performance. If that classifier makes an incorrect decision, we will not be able to correct that decision. In addition, the choice of k and the distance measure affect the performance of the *DCS*- based methods. Therefore, a dynamic ensemble selection (DES) approach has been proposed by later studies.

In [19], it was assumed that the degree of agreement among the members of an ensemble represents a high confidence level of classification. Instead of searching for an ensemble for each instance point from a pool of classifiers, Dos Santos et al. [19] integrated an optimization step into their experiments. First, a set of candidate ensembles is populated from a large initial pool of candidate classifiers using single and multi-objective genetic algorithms. To guide the search, the error rate and four diversity measures were applied to find the best performing N ensembles. Then, for each new example the ensemble with highest degree of agreement among its members (lowest ambiguity) was chosen from the candidate ensembles to make the prediction. It is shown that this method outperformed the *DCS* methods but not the best candidate ensemble (static selection). Dos Santos et al. [20] extended this study by using two more measures as a second-level objective after optimizing on accuracy and diversity: “Margin” and “Strength relative to the closest class”. *Margin* is defined as the difference between the number of votes assigned to the two classes with the highest number of votes. However, *Strength relative to the closest class* normalizes the *Margin* by dividing it by the performance of the candidate ensemble on the validation points with the assigned class label. The proposed methods outperformed the static selection and the fusion of the initial pool of classifiers. The performance of both [19] and [20] depend on the quality of the ensembles generated in the optimization step. In addition, the data set needs to be divided into four parts to apply these methods which is not suitable for small data sets. Furthermore, using only the candidate ensembles generated in the optimization step could be limiting as it was shown that there was a decrease in the oracle power of the system.

Ko et al. [37] proposed *KNORA* which, for any new instance, finds its nearest k neighbors in the validation set using the original feature set, and dynamically chooses an ensemble for each new instance based on the estimated accuracy of the classifiers in this local region of a new instance. More specifically, this method chooses all of the classifiers that correctly classified at least one of the neighbors (or all of the neighbors) for creating the ensemble. This method was compared against *DCS* methods proposed in [16, 75] and a single k-NN classifier. The variant of *KNORA* which chooses all of the classifiers that correctly classified all of the neighbors performed better than the compared methods.

Following this approach, Cavalin et al. [9] adopted a hybrid framework. They used the confidence measure defined in [20] to build the ensembles. However, when the confidence value of any of the candidate ensembles was not enough (above a pre-set threshold), it searched for the "closest" or "most similar" instance to the new instance in the validation set and assigned its label to the new instance. To measure closeness (similarity), two methods were proposed based on classifiers' predictions for the instances in the validation set and for the new instance. The results showed integrating contextual information helped improving the performance of the system.

Recently, Li et al. [46] argued that the most confident M classifiers should be used to create the ensemble for classifying the new instance. The decision of classifiers are weighted by their class prediction probability (confidence) values for the new instance. Unlike our methods discussed in Chapter 3, M is estimated by using margin distribution defined by classification confidence.

Rather than defining confidence of an ensemble, Woloszynski and Kuryzynski [73] proposed a method for calculating the classifier competence using a probabilistic model. New instances are classified using each base classifier, and then a randomized reference classifier (RRC) is created that produces, on average, the same vector of class supports for that instance as the base classifier. Using the probability distribution information encapsulated in the RRC model, a "competence" value is then calculated for each classifier on each new instance. This competence measure represents the quality of the classifier's decision based on the relative values returned in the class support vector. DCS — selecting the single most-competence classifier — and *DES* — building an ensemble of all classifiers more competent than a random classifier — schemes were proposed. Further, two variants of *DES* were implemented: *DES-CV*, in which each classifier's vote is weighed by its competence, and *DES-CS*, in which each classifier's class support vector is weighed by its competence. The proposed methods were compared against other dynamic schemes, including [16, 20, 75]. It was shown that the *DES-CS* had higher average rank than the others.

Vriesmann et al. [72] focused on improving the *KNORA* method proposed in [37]. First, the effect of using different distance measures for defining the local region of a test instance on accuracy was investigated and [72] concluded that the choice of distance measure had no effect on the performance of *KNORA*. Then, different strategies were proposed for combining the information obtained from k neighbors of the new instance and the output of *KNORA*. These strategies differed in which method should be used conditional on one of the methods meeting a certain criteria. Based on the experimental results, [72] suggested that additional information provided by the k-NN improved the performance of *KNORA*.

In Chapter 4, we hypothesize that using class supports returned by classifiers and the information retrieved from the neighborhood of a new instance together will improve the performance of the system. This was supported by the results presented in [9, 72].

CHAPTER 3
THE ROLE OF CONFIDENCE IN DYNAMIC ENSEMBLES

Classifier ensembles, in which multiple predictive models are combined to produce predictions for new data instances, generally outperform a single classifier. Most existing methods construct a static ensemble, which features a single collection of classifiers for all new instances. Recently, dynamic ensemble construction algorithms have been proposed; we discuss the relevant literature in Chapter 2. These algorithms choose an ensemble specifically for each data instance from a large pool of classifiers. To compare and contrast the benefit of dynamic ensemble selection, two factors have been commonly used: the accuracy of the individual classifiers and the diversity of the ensemble.

Performance of the ensembles is evaluated based on their accuracy, on the unknown data instances (i.e. test instances), which is referred to as *generalization*. "Acc", which measures the average accuracy of individual classifiers on the validation dataset, is not necessarily the best indicator of generalization. We hypothesize that using the class probability estimates returned by the classifiers can enhance our estimate of the competency of classifiers on the prediction. Therefore, ensembles composed of *accurate*, *diverse*, and *confident* classifiers should improve generalization since having classifiers that make accurate prediction with high probability increases reliability of an ensemble, and diverse classifiers can compensate each other's incorrect predictions. We employ heuristic optimization, prediction ranking, and logistic regression methods to examine the relevance of this third factor: the confidence of each classifier's prediction on a specific data instance. Since the confidence of a classifier depends on the data instance, incorporating it into our ensemble selection process requires dynamic ensemble selection.

3.1 Dynamic Ensemble Construction using Heuristic Search Methods

Given a set of classifiers, $\mathcal{C}$, which have already been trained in the overproduction phase, heuristic search algorithms are employed to find the best subset of classifiers in the selection phase. In this optimization problem, two important aspects are to be analyzed: (1) the search objective and (2) the search algorithm.

For the search algorithms, we model classifier selection as a *multi-objective optimization problem* (MOOP) or a Pareto optimization problem, rather than combining objectives. A MOOP is an optimization problem with multiple objective functions and in general a single solution will not be better than all other solutions with respect to all objectives. Therefore, our aim is to find the tradeoffs among multiple, usually conflicting objective functions. In a MOOP, a solution s_1 is said to dominate another solution s_2 if s_1 is no worse than s_2 on all the objective functions and if s_1 is better than s_2 in at least one objective function. The set of non-dominated solutions represent different tradeoffs between the multiple objective functions and are referred to as the *Pareto front*.

Our ensemble construction problem is described as follows: For a given data instance t and an existing pool of component classifiers $\mathcal{C} = \{C_a \mid a = 1, \ldots, M\}$, find the best subset of classifiers to create an ensemble of classifiers $E \subseteq \mathcal{C}$ with size N:

$$\max_{E} \{Acc(E), Div(E), Conf(E,t)\} \tag{3.1}$$

$$\text{s.t. } |E| = N \tag{3.2}$$

where the term $|E|$ refers to the ensemble size. In this optimization problem, we attempt to

Table 3.1: An example of the role of diversity in ensemble prediction (majority voting)

	Classifiers with 70% Accuracy				Classifiers with 80% Accuracy			
Data Instance	C_1	C_2	C_3	Prediction	C_1	C_2	C_3	Prediction
v_1	1	1	1	1	1	1	1	1
v_2	1	1	0	1	1	1	1	1
v_3	1	1	0	1	1	1	1	1
v_4	1	1	0	1	1	1	1	1
v_5	1	0	1	1	1	1	1	1
v_6	1	0	1	1	1	1	1	1
v_7	1	0	1	1	1	1	1	1
v_8	0	1	1	1	1	0	0	0
v_9	0	1	1	1	0	1	0	0
v_{10}	0	1	1	1	0	0	1	0
Accuracy	0.7	0.7	0.7	1.0	0.8	0.8	0.8	0.7

simultaneously optimize three separate objectives, which we explain in more detail below.

The first objective is the *ensemble accuracy*, denoted by $Acc(E)$. To calculate this measure, we first evaluate each classifier's performance, denoted by C_a on a validation set: $Acc(C_a)$ represents the percentage of times classifier C_a correctly predicts data instances in the validation set. The ensemble accuracy, $Acc(E)$, is then the average of these individual accuracies for the classifiers in the ensemble. This simple objective may be sufficient by itself at times, but it is theoretically weak. For instance, an ensemble of three similar classifiers with 80% accuracy may perform worse than an ensemble of three diverse classifiers with an average accuracy of 70%. Table 3.1 illustrates such a case. Even though the ensemble with similar classifiers has $Acc(E) = 80\%$, generalization of the ensemble could be lower than the individual classifiers'. On the other hand, the ensemble with more diverse classifiers has $Acc(E) = 70\%$ accuracy but perfect generalization because they can compensate for each other's misclassifications and achieve higher generalization.

The second objective is the *ensemble diversity*, denoted by $Div(E)$. We calculate the ensemble diversity on the validation dataset using the conditional double fault method proposed in [78]. The pairwise similarity between classifiers C_a and C_b is defined to be the average probabilities of classifier C_a misclassifying a data instance given that classifier C_b misclassifies it, and vice versa. Pairwise diversity between C_a and C_b is calculated as follows:

$$Div_{ab} = 1 - \frac{\Pr\left(O_{i,a} = 0|O_{i,b} = 0\right) + \Pr\left(O_{i,b} = 0|O_{i,a} = 0\right)}{2} \tag{3.3}$$

$$\text{where for } m = a, b;\ O_{i,m} = \begin{cases} 1, & \text{if classifier } C_m \text{ is correct on data instance } i, \\ \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

The third objective is the *ensemble confidence*. The ensemble confidence is a function of the individual confidence (class prediction probability) values of the classifiers, also referred to as *classifier confidence*. It is noteworthy that without the ensemble confidence, ensemble E would be the same for all data instances, since both accuracy and diversity are defined for the validation set and do not depend on instance t. However, the ensemble confidence depends on each data instance, so the solution to the optimization problem in (3.1) for different data instances may potentially vary.

The confidence of a classifier on a data instance is defined as the probability of the most likely class. We consider two different measures for the ensemble confidence:

1. $Conf_1(E,t)$: The ensemble confidence is the average classifier confidence in the ensemble for instance t:

$$Conf_1(E,t) = \frac{\sum\limits_{C_a \in E} Conf(C_a, t)}{|E|} \tag{3.5}$$

   where $Conf_1(E,t)$ represents the ensemble confidence and $Conf(C_a,t)$ denotes classifier's confidence for instance t.

2. $Conf_2(E,t)$: The margin confidence of an ensemble is the difference between the sum of confidence values for each class label. In other words, we first split the classifiers into two groups, depending on their classification, then calculate the difference between the sum of confidence values for each label. Let $\delta^{\ell^1}_{C_a(t)}$ denote the indicator function, which equals 1 if a classifier's prediction for instance t, denoted by $\ell_{a,t}$, is ℓ^1, and 0 otherwise. The confidence of an ensemble for instance t is then defined as follows:

$$Conf_2(E,t) = \frac{\left| \sum\limits_{C_a \in E} \left[\delta^{\ell_1}_{C_a,t} - \delta^{\ell_2}_{C_a,t}\right] \; Conf(C_a,t) \right|}{|E|} \tag{3.6}$$

where $\delta^{\ell_2}_{C_a,t}$ is defined similarly to $\delta^{\ell_1}_{C_a,t}$.

3.1.1 Search Methods

It is important to have a search method that provides accurate and reliable solutions in a reasonable time as we aim to select a new ensemble for each test point, which is computationally intensive. The algorithm running time depends on the number of data instances in our test set. Therefore, the choice of the search method is an important decision. There are many heuristic search methods with a scalar objective function and we consider three of them: (1) first improvement local search, (2) best improvement local search, and (3) multi-objective genetic algorithm (MOGA).

For local search algorithms, the neighborhood definition is critical as this determines the quality of the local optima and the computational effort required to identify them. Let $\mathcal{S}$ be the solution space, where in our case a solution s is an ensemble of $|E|$ classifiers. For each solution $s \in \mathcal{S}$, we define the neighborhood as the set $\mathcal{N}(s) = \{s' : s \triangle s' = 1\}$ where $\triangle$ is the symmetric difference operator. In words, s and s' are two feasible solutions that have all but one common classifiers. The size of $\mathcal{N}(s)$ is $|E| \times (M - |E|)$, where M is the total number of classifiers in the pool. In addition, each classifier is given a unique identifier between $[0, M - 1]$.

3.1.1.1 First Improvement

Our implementation of first-improvement local search begins by randomly generating a feasible solution s and evaluating the objective function for this solution $f(s)$; see Algorithm 3.1. In each iteration, the algorithm randomly chooses both a classifier $C_a \in s$ to remove and a classifier $C_b \notin s$ from the pool to replace it, creating a new ensemble, s_{new}. Then, it evaluates the objective function for this new solution, $f(s_{new})$. If $f(s_{new}) > f(s)$, then the algorithm continues the search from s_{new}. It terminates when an iteration count, denoted by N_{stop}, has been met.

3.1.1.2 Best Improvement

Our implementation of best-improvement local search also begins with a random ensemble s; see Algorithm 3.2. At each step, it iterates over all solutions in the neighborhood of s, picking the one that yields the best improvement in the objective function, if any. Then, it moves on to the next neighborhood. It terminates when there is no neighbor better than solution s.

3.1.1.3 Multi-Objective Genetic Algorithm (MOGA)

For this study, we also implement a modified version of the MOGA proposed in [54]. This method is similar to a single-objective genetic algorithm. However, the selection pro-

Algorithm 3.1 First Improvement Algorithm

```
1.  procedure FIRST IMPROVEMENT PROCEDURE
2.    input: pool of classifier $\mathcal{C}$
3.    $f$ := 0// objective function value
4.    $s$ :=<> // solution = ensemble
5.    $s$ := randomly generate $|E|$ different numbers in range [0,M-1]
6.    $s_{best} := s$
7.    $f$ := evaluate objective at $s$
8.    $f_{best} := f$
9.    for $l = 1 : N_{stop}$ do
10.     $C_b$ := randomly pick a classifier $\notin s$
11.     $C_a$ := randomly pick a classifier $\in s$
12.     $s_{new} := s \setminus C_a \cup C_b$
13.     $f_{new}$ := evaluate objective at $s_{new}$
14.     if $f_{new} > f_{best}$ then
15.       $f_{best} := f_{new}$
16.       $s_{best} := s_{new}$
17.     end if
18.   end for
19.   output: $f_{best}, s_{best}$
```

Algorithm 3.2 Best Improvement Algorithm

```
1.  procedure BEST IMPROVEMENT PROCEDURE
2.    input: pool of classifier $\mathcal{C}$
3.    $f := 0$// objective function value
4.    $s :=<>$ // solution = ensemble
5.    $s :=$ randomly generate $|E|$ different numbers in range [0,M-1]
6.    $s_{best} := s$
7.    $f :=$ evaluate objective at $s$
8.    $f_{best} := f$
9.    improved := true
10.   while improved AND $l < N_{stop}$ do
11.     $C_a :=$ randomly pick a classifier $\in s$ that has not been evaluated
12.     $improved := false$
13.     $l := l + 1$
14.     foreach $C_b \notin s$ do
15.       $l := l + 1$
16.       $s_{new} := s \setminus C_a \cup C_b$
17.       $f_{new} :=$ evaluate objective at $s_{new}$
18.       if $f_{new} > f_{best}$ then
19.         $f_{best} := f_{new}$
20.         $s_{best} := s_{new}$
21.         $improved := true$
22.       end if
23.     end foreach
24.   end while
25.   output: $f_{best}, s_{best}$
```

cedure we adopt in this model is different than [54]. In their selection procedure, Murata and Ishibuchi [54] take a weighted sum of the individuals' objectives and choose individuals for crossover accordingly. The weights of the various objectives are assigned randomly at each generation, changing the direction of the search as it proceeds. However, in our implementation of MOGA, we keep the weights constant during all generations of a *single* run to increase coverage of the objective space and make the search method comparable to our other methods.

There are two phases of MOGA: (1) selection and (2) recombination. In the selection phase, during each successive generation, a proportion of the existing population is selected to breed a new generation. In this study, individual solutions are selected through a fitness-based process called roulette-wheel selection or fitness proportionate selection. In this procedure, the evaluation function assigns fitness to possible solutions. This fitness level, f_k, is used to associate a probability of selection with each individual, $p_k = \frac{f_k}{\sum_{k=1}^{L} f_k}$ where L is the number of individuals in the population.

In the reproduction phase, offspring are generated by combining two parents (two individuals in the population). It is worth noting that each individual in this population represents an ensemble and is characterized by a decimal vector of unique identifiers of the classifiers in this ensemble (chromosome). To create children, a one-point crossover technique is implemented. All data beyond the crossover point, cp, is swapped between the two parents. In other words, the classifier identifiers (decimal numbers) from the beginning of a vector to the crossover point are copied from one parent, the rest of the numbers are copied from the second parent. The two resulting individuals are the children or offspring. Mutation

Algorithm 3.3 Multi Objective Genetic Algorithm

```
1.  procedure MOGA
2.    input: Initial population population, Fitness vector for population F,
3.            Number of generations numGen, Mutation rate mr,
4.            crossover point cp, Selection rate sr
5.    $f_{best} := Max(F)$ and $s_{best}$ is the associated individual
6.    for $g = 1 : numGen$ do
7.      parents := Selection(population,F,sr)
8.      children := Crossover(parents, cp)
9.      children := Mutation(children, mr)
10.     $F_{children}$ := evaluate(children)
11.     [population, F] := NextGeneration(population, children, F, $F_{children}$)
12.
    picks best $L$ individuals from current population including children to create the new generation
13.     $f_{GA} := Max(F)$
14.     if $f_{GA} > f_{best}$ then
15.       $f_{best} := f_{GA}$
16.       $s_{best} := s_{GA}$ is the individual with $f_best$
17.     end if
18.   end for
19.   output: $f_{best}, s_{best}$
```

Algorithm 3.4 Selection Algorithm

```
 1. procedure SELECTION PROCEDURE
 2.   input: population, fitness vector for population $F$, selection rate $sr$
 3.   $L := \|F\|$
 4.   Calculate $p_i$ for each individual in the population
 5.   $c :=$ cumulative distribution of $p_i$
 6.   $r := rand(sr * L)//$ randomly generate sr*L numbers between $[0, 1)$
 7.   parents := $\varnothing$
 8.   for $t = 1 : sr * L$ do
    sr*L individuals will be chosen as parents
 9.     for $j = 1 : L$ do
10.       if $c(j) > r(t)$ then
11.         parents(t) := $population(j - 1)$
12.         $j := L$
    To break the for loop
13.       end if
14.     end for
15.   parents := unique(parents)
16.   output: parents
```

is applied to children that are bred after the crossover method is applied to prevent the population from becoming too similar. Assuming that M is equal to 2^k, for the mutation implementation each classifier is represented by the k-bit binary representation of its unique identifier. The vector that represents an individual in the population is converted into a binary vector of size $|E| * k$. Therefore, each chromosome or individual is represented by $|E| * k$ bits. After the modification on the representation of the chromosomes, mutation is done probabilistically by flipping a bit in this binary vector. At the end of the mutation, we convert the binary vectors into decimal ones. For this study, the resulting generation of offspring is merged with the current population and the best L individuals are kept in the population.

Algorithm 3.5 Crossover Algorithm

```
1.  procedure CROSSOVER PROCEDURE
2.    input: initial population $population$, crossover point $cp$,
3.           size of the ensemble $|E|$, parents from Selection Algorithm $parents$
4.    Randomly order parents
5.    for $i = 1 : \|\text{parents}\|/2$ do
6.      $p_1$ := parents(2 i - 1) // parent 1
7.      $p_2$ := parents(2 i) // parent 2
8.      $o_1 := \{p_1^1, \ldots, p_1^{cp}\} \cup \left\{p_2^{cp+1}, \ldots, p_2^{|E|}\right\}$ // offspring 1
9.      $o_2 := \{p_2^1, \ldots, p_2^{cp}\} \cup \left\{p_1^{cp+1}, \ldots, p_1^{|E|}\right\}$ // offspring 2
10.     children(2 (i - 1)) := $o_1$
11.     children(2 i) := $o_2$
12.   output: children
```

Algorithm 3.6 Mutation Algorithm

```
1.  procedure MUTATION PROCEDURE
2.    input: $children$, mutation rate $mr$,
3.    B := number of bits for each individual
4.    for $i = 1 : \|\text{children}\|$ do
    children $i$ is chosen
5.      for $j = 1 : B$ do
6.      $r := rand(0, 1)$
7.        if $r < mr$ then
8.        flip bit children$_i^j$
9.      end for
10.   end for
11.   output: children
```

3.1.2 Experimental Results

Our hypothesis is that ensembles based on accuracy, diversity, and confidence should make more accurate predictions on the new data instances (i.e. improve generalization). However, since we do not know how confidence interacts with accuracy and diversity, we experiment with all combinations of the three objectives described in Section 3.1. The values for the objective functions are linearly scaled to the interval $[0, 1]$, by converting the minimum and maximum values to zero and one, respectively. To find the minimum and maximum values of $Div(E)$, we use the results of the single-objective heuristic search.

We consider linear combinations of the three objectives described above to apply single-objective search algorithms to our MOOP. Since the optimal trade-off among the various objectives is unknown, randomly-selected weights are set on the three objectives. The search ends when N_{stop} is 1000 for the first improvement and is 10000 for the best improvement. In addition, to be able to cover as much of the objective space as possible for each new data instance, we start the search from 100 random initial solutions in the First and Best Improvement algorithms. To compare the results from the MOGA algorithm with the First and Best Improvement algorithms, we set the population size to 100.

There are several methods that can be used to select a single solution (ensemble of size 25) to classify each new data instance among generated solutions. First, we examine only those solutions on the Pareto front (i.e. non-dominated solutions), and choose one in two different ways: (1) Pareto-ranking, which indicates the number of other local solutions that each one dominates, and (2) validation accuracy. Alternatively, we also experiment with choosing the final solution based on its validation accuracy of the solution among all generated solutions.

In our experiments, the base classifiers are RBF kernel support vector machines (SVMs) [71], which are generated using LIBSVM. LIBSVM uses Platt's formula [58] to calculate the class conditional probabilities (class supports).

We employ LIBSVM in MATLAB to construct $M = 1024$ RBF kernel SVM classifiers, and choose the parameters such that classifiers overfit the data. We perform the experiments on four data sets retrieved from the LIBSVM site: *Australian*, *Diabetes*, *Heart Disease*, and *Liver Disorder*. We construct a highly diverse initial pool of classifiers using a combination of bootstrap instance sampling (as in bagging) and random subspace selection. We perform 5-fold cross-validation and repeat this experiment 5 times.

3.1.2.1 Comparison of Search Methods

We first carry out an experiment on the *Diabetes* dataset to decide on the search algorithm. Unlike other experiments specified in this section, we only perform 5-fold cross validation once. We consider the first definition of the confidence, $Conf_1$. For this experiment, the final ensemble is chosen from all of the generated solutions based on its validation accuracy. Table 3.2 illustrates the results. The First Improvement algorithm is the fastest in terms of computational time, but the tradeoff is lower objective value. MOGA improves the objective value (about 20–30%) in exchange for a 4-fold increase in computational time. Best Improvement is the slowest. Compared to MOGA, the objective value in Best Improvement is lower (due to the stopping rule) and its test accuracy is higher. Compared to First Improvement, the objective value is higher, but at the expense of (generally) lower test

Table 3.2: Comparison of search methods on the *Diabetes* dataset

	First Improvement		
Objectives	Speed[a]	Obj Value[c]	Test Acc
Div	13 sec	0.546	0.730
Acc + Conf[b]	15sec/pt	0.584	0.733
Acc + Div	32sec	0.754	0.746
Conf[b]+ Div	27sec/pt	0.573	0.752
Acc + Conf[b]+ Div	43 sec/pt	0.621	0.745
	MOGA		
Objectives	Speed[a]	Obj Value	Test Acc
Div	60sec	0.625	0.739
Acc + Conf[b]	91sec/pt	0.866	0.739
Acc + Div	102sec	0.923	0.730
Conf[b]+ Div	121sec/pt	0.898	0.730
Acc + Conf[b]+ Div	137sec/pt	0.635	0.752
	Best Improvement		
Objectives	Speed[a]	Obj Value	Test Acc
Div	300 sec	0.612	0.7411
Acc + Conf[b]	400 sec/pt	0.824	0.746
Acc + Div	480sec	0.845	0.756
Conf[b]+ Div	513sec/pt	0.643	0.735
Acc + Conf[b]+ Div	600sec/pt	0.604	0.757

[a] Speed is measured in seconds per point when confidence is included in the objective.
[b] $Conf_1$ is used as the measure of confidence.
[c] Obj Value represents the linear combination of objectives.

accuracy.

While MOGA and Best Improvement result in relatively more promising solutions, they are dramatically slower than First Improvement. In addition, to use MOGA effectively we need to run a preprocessing step to set the parameters (e.g. crossover rate, mutation rate), which further slows down the process for each dataset. The First Improvement algorithm does not compromise on the test accuracy, while being much faster in terms of computational time. Due to the computation time required for these experiments, we adopt the First

Improvement algorithm as the search algorithm for the rest of our experiments.

3.1.2.2 Comparison of Search Objectives

In this section, we perform two sets of experiments, one for each of the two confidence definitions introduced in Section 3.1. We present the results for the two selection schemes (i.e. Pareto optimal solutions vs. all solutions). Performance of the ensembles is evaluated based on their accuracy in the test dataset.

Results shown in Table 3.3 are obtained by choosing the final solution among the solutions on the Pareto front based on validation error. Comparing the two confidence definitions, diversity (Div) is lower for $Conf_1$ when confidence is the only criterion in the objective, but is higher when confidence is employed alongside other factors. This implies that ensemble which contains confident classifiers with certain level of agreement on the label of a new data instance are not necessarily similar. Meanwhile, accuracy (Acc) and generalization (Gen) do not change much. When choosing classifiers based solely on their confidence, it appears that $Conf_2$ is slightly better than $Conf_1$ in three of the four datasets. The biggest difference is in the Liver Disorder dataset where $Conf_2$ yields 0.650 relative to 0.641 for $Conf_1$. However, when it is complemented with the accuracy or diversity objectives, $Conf_1$ is better than $Conf_2$. For instance, when Div and Conf are incorporated, generalization for $Conf_1$ equals 0.801 in the Australian dataset, compared to 0.605 for $Conf_2$. When all three objectives are incorporated together, the results are inconclusive.

Contrary to our expectations, Table 3.3 shows that ensemble confidence (Conf) appears to be a poor complement to ensemble accuracy: choosing classifiers based solely on

Table 3.3: Summary results of the final solution based on Pareto ranking

	Australian							
	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.842	0.308	0.824	0.833	0.737	0.308	0.824
Conf	0.553	0.951	0.388	0.792	0.553	0.944	0.447	0.791
Div	0.593	0.643	0.576	0.826	0.592	0.185	0.576	0.823
Acc+Conf	0.629	0.715	0.462	0.663	0.608	0.589	0.303	0.628
Acc+Div	0.606	0.676	0.567	0.818	0.609	0.236	0.569	0.824
Div+Conf	0.598	0.695	0.558	0.801	0.596	0.595	0.264	0.605
Acc+Conf+Div	0.612	0.699	0.548	0.794	0.615	0.430	0.486	0.794

	Diabetes							
	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.775	0.352	0.745	0.762	0.615	0.352	0.745
Conf	0.583	0.968	0.442	0.741	0.582	0.960	0.500	0.741
Div	0.575	0.681	0.559	0.736	0.577	0.171	0.559	0.734
Acc+Conf	0.662	0.768	0.401	0.704	0.668	0.651	0.349	0.691
Acc+Div	0.595	0.711	0.552	0.741	0.599	0.234	0.553	0.743
Div+Conf	0.586	0.753	0.546	0.747	0.657	0.645	0.362	0.697
Acc+Conf+Div	0.626	0.757	0.494	0.727	0.650	0.589	0.413	0.728

	Heart							
	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.725	0.411	0.773	0.781	0.556	0.411	0.773
Conf	0.540	0.889	0.408	0.729	0.539	0.879	0.507	0.730
Div	0.582	0.612	0.575	0.763	0.583	0.154	0.576	0.765
Acc+Conf	0.600	0.672	0.473	0.682	0.599	0.519	0.340	0.653
Acc+Div	0.586	0.639	0.569	0.753	0.591	0.194	0.570	0.758
Div+Conf	0.576	0.659	0.562	0.758	0.586	0.516	0.323	0.646
Acc+Conf+Div	0.585	0.658	0.560	0.767	0.594	0.376	0.499	0.756

	Liver Disorder							
	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.727	0.339	0.707	0.746	0.576	0.339	0.707
Conf	0.554	0.961	0.441	0.641	0.554	0.946	0.481	0.650
Div	0.568	0.657	0.556	0.683	0.567	0.154	0.557	0.684
Acc+Conf	0.607	0.737	0.436	0.669	0.608	0.599	0.370	0.651
Acc+Div	0.582	0.685	0.549	0.699	0.582	0.215	0.550	0.691
Div+Conf	0.571	0.718	0.541	0.701	0.597	0.600	0.366	0.642
Acc+Conf+Div	0.584	0.718	0.531	0.700	0.599	0.521	0.449	0.706

their confidence, instead of either accuracy (Acc) or diversity (Div) alone, reduces performance (Gen) in three out of four datasets (i.e. 0.792 compared to 0.826 or 0.824 in the Australian dataset). Confidence has a slightly positive effect for the $Conf_1$ definition when it is combined with diversity (i.e. Gen equals 0.747 in the Diabetes and 0.701 in the Liver Disorder datasets), and a detrimental effect for the $Conf_2$ definition (i.e. Gen drops to 0.697 and 0.6642 for these two datasets).

Columns 2–5 and 6–9 of Tables 3.4 and 3.5 analyze the effect of Pareto-optimal solutions on the results (Acc, Conf,Div, Gen) across different objectives. In some cases, the size of the Pareto front is quite small (as low as three ensembles). Therefore, we choose the final ensemble among *all generated solutions* based on validation performance to check if the results differ. We find that choosing ensembles among generated solutions based on validation accuracy (Acc) increases the performance (Gen) in general. For instance, in Table 3.4, which employs the $Conf_1$ definition, maximum generalization in the Liver Disorder dataset is attained at 0.707 when only Acc is considered in the objective. However, another effective objective, which maximizes generalization in some datasets, is when all three criteria (Acc, Conf, Div) are taken into account (i.e. in the Australian, Diabetes, and Heart datasets for $Conf_1$, and Australian and Diabetes datasets for $Conf_2$).

Furthermore, when we consider accuracy and confidence as our objectives, Pareto-optimal solutions perform better than *all solutions* in accuracy and confidence, but worse in diversity and generalization which could imply that diversity should be a factor. For the other objectives, the results are quite similar. These findings are also robust across the two confidence definitions.

Table 3.4: Summary results of the final solution based on its validation data performance for the Conf_1 definition.

Australian								
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.842	0.308	0.824	0.833	0.842	0.308	0.824
Conf	0.553	0.951	0.388	0.792	0.553	0.951	0.388	0.792
Div	0.577	0.639	0.569	0.823	0.577	0.639	0.569	0.823
Acc+Conf	0.626	0.708	0.498	0.741	0.603	0.683	0.550	0.812
Acc+Div	0.601	0.660	0.569	0.822	0.576	0.640	0.567	0.820
Div+Conf	0.589	0.669	0.571	0.823	0.574	0.652	0.566	0.825
Acc+Conf+Div	0.594	0.667	0.569	0.828	0.580	0.654	0.567	0.823

Diabetes								
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.775	0.352	0.745	0.762	0.775	0.352	0.745
Conf	0.583	0.968	0.442	0.741	0.583	0.968	0.442	0.741
Div	0.568	0.688	0.553	0.730	0.568	0.688	0.553	0.730
Acc+Conf	0.646	0.766	0.457	0.732	0.620	0.749	0.508	0.740
Acc+Div	0.592	0.692	0.553	0.741	0.584	0.687	0.550	0.736
Div+Conf	0.576	0.722	0.554	0.742	0.573	0.708	0.551	0.743
Acc+Conf+Div	0.590	0.715	0.549	0.745	0.583	0.708	0.549	0.745

Heart								
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.725	0.411	0.773	0.781	0.725	0.411	0.773
Conf	0.540	0.889	0.408	0.729	0.540	0.889	0.408	0.729
Div	0.560	0.611	0.565	0.769	0.560	0.611	0.565	0.769
Acc+Conf	0.597	0.665	0.522	0.726	0.578	0.645	0.554	0.755
Acc+Div	0.586	0.618	0.572	0.769	0.571	0.609	0.567	0.768
Div+Conf	0.575	0.636	0.570	0.752	0.561	0.620	0.564	0.764
Acc+Conf+Div	0.579	0.632	0.569	0.781	0.566	0.619	0.565	0.778

Liver Disorder								
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.727	0.339	0.707	0.746	0.727	0.339	0.707
Conf	0.554	0.961	0.441	0.641	0.554	0.961	0.441	0.641
Div	0.562	0.655	0.547	0.681	0.562	0.655	0.547	0.681
Acc+Conf	0.607	0.720	0.470	0.690	0.594	0.703	0.504	0.690
Acc+Div	0.584	0.667	0.548	0.688	0.570	0.665	0.547	0.688
Div+Conf	0.565	0.691	0.550	0.699	0.562	0.675	0.547	0.701
Acc+Conf+Div	0.576	0.684	0.545	0.688	0.570	0.676	0.544	0.699

Table 3.5: Summary results of the final solution based on its validation data performance for the Conf_2 definition.

	Australian							
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.737	0.308	0.824	0.833	0.737	0.308	0.824
Conf	0.553	0.944	0.447	0.791	0.553	0.944	0.447	0.791
Div	0.571	0.141	0.569	0.824	0.571	0.141	0.569	0.824
Acc+Conf	0.622	0.476	0.406	0.723	0.611	0.356	0.487	0.757
Acc+Div	0.602	0.215	0.567	0.822	0.580	0.168	0.566	0.813
Div+Conf	0.588	0.530	0.331	0.646	0.583	0.511	0.323	0.653
Acc+Conf+Div	0.596	0.244	0.568	0.823	0.582	0.203	0.566	0.825
	Diabetes							
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.615	0.352	0.745	0.762	0.615	0.352	0.745
Conf	0.582	0.960	0.500	0.741	0.582	0.960	0.500	0.741
Div	0.574	0.173	0.552	0.743	0.574	0.173	0.552	0.743
Acc+Conf	0.669	0.595	0.391	0.709	0.642	0.494	0.475	0.734
Acc+Div	0.603	0.234	0.549	0.746	0.585	0.198	0.550	0.736
Div+Conf	0.623	0.550	0.482	0.740	0.628	0.526	0.482	0.741
Acc+Conf+Div	0.595	0.291	0.547	0.750	0.588	0.263	0.548	0.749
	Heart							
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.556	0.411	0.773	0.781	0.556	0.411	0.773
Conf	0.539	0.879	0.507	0.730	0.539	0.879	0.507	0.730
Div	0.562	0.112	0.567	0.769	0.562	0.112	0.567	0.769
Acc+Conf	0.607	0.418	0.429	0.707	0.592	0.285	0.527	0.737
Acc+Div	0.585	0.163	0.572	0.757	0.565	0.120	0.566	0.770
Div+Conf	0.579	0.429	0.442	0.717	0.579	0.400	0.462	0.718
Acc+Conf+Div	0.582	0.208	0.568	0.763	0.568	0.160	0.565	0.760
	Liver Disorder							
	Pareto-Optimal Solutions				*All Solutions*			
Objectives	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.576	0.339	0.707	0.746	0.576	0.339	0.707
Conf	0.554	0.946	0.481	0.650	0.554	0.946	0.481	0.650
Div	0.560	0.147	0.549	0.682	0.560	0.147	0.549	0.682
Acc+Conf	0.616	0.490	0.429	0.693	0.602	0.400	0.484	0.703
Acc+Div	0.585	0.211	0.541	0.693	0.568	0.179	0.542	0.691
Div+Conf	0.584	0.513	0.469	0.690	0.588	0.491	0.473	0.695
Acc+Conf+Div	0.580	0.272	0.542	0.694	0.572	0.239	0.542	0.692

3.1.2.3 Effect of Pool Diversity

In addition to the experiments mentioned in Section 3.1.2.2, we further explore whether classifiers with less than 50% accuracy (i.e. *bad classifiers*) should be discarded from the pool. These classifiers may make incorrect predictions with high confidence value, and hence, they can decrease the performance of the system. We perform the same experiments specified in Section 3.1.2.2 with bad classifiers omitted from the initial classifier pool. The results are presented in Tables 3.6–3.8.

When compared with the results presented in Section 3.1.2.2, it becomes clear that diversity of an ensemble is really important for two reasons. The first reason is that generalization has not increased due to the smaller classifier pool. In fact, Acc is now the only objective (by itself) which attains the maximum generalization. The second reason is that Div is no longer a crucial component of the objective: any objective that includes diversity results in a substantially lower generalization value. For instance, as depicted in Tables 3.4 and 3.7, when all three (Acc, Conf, Div) are considered in the objective, it could achieve the maximum generalization (0.745 and 0.750 for the Diabetes dataset). However, once the bad classifiers are excluded from the pool, the same objective yields 0.730 and 0.696, respectively, for the same dataset. Therefore, bad classifiers add information to the system as they increase the diversity of the pool.

When we look at the results for $Conf_2$, we conclude that having bad classifiers in the initial pool is essential as they increase the diversity of the pool and the ensemble. The objective (Div,$Conf_2$) for the Australian dataset is also interesting as after certain diversity is achieved, some level of confidence is required to make correct predictions.

Table 3.6: Summary results of the final solution based on Pareto ranking. Classifiers in the initial pool have at least 50% validation accuracy.

Australian

	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.842	0.308	0.824	0.833	0.737	0.308	0.824
Conf	0.588	0.945	0.325	0.797	0.594	0.940	0.226	0.794
Div	0.630	0.648	0.493	0.632	0.630	0.312	0.493	0.632
Acc+Conf	0.660	0.726	0.382	0.590	0.621	0.673	0.252	0.570
Acc+Div	0.656	0.697	0.453	0.646	0.604	0.531	0.234	0.555
Div+Conf	0.654	0.709	0.426	0.617	0.638	0.580	0.361	0.618
Acc+Conf+Div	0.659	0.714	0.415	0.622	0.564	0.644	0.044	0.555

Diabetes

	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.775	0.352	0.745	0.762	0.615	0.352	0.745
Conf	0.669	0.961	0.309	0.734	0.669	0.957	0.342	0.731
Div	0.658	0.682	0.494	0.723	0.656	0.368	0.494	0.715
Acc+Conf	0.696	0.786	0.360	0.715	0.692	0.727	0.332	0.687
Acc+Div	0.682	0.715	0.464	0.729	0.686	0.513	0.371	0.697
Div+Conf	0.682	0.761	0.435	0.726	0.684	0.708	0.357	0.704
Acc+Conf+Div	0.694	0.769	0.397	0.727	0.675	0.687	0.213	0.656

Heart

	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.725	0.411	0.773	0.781	0.556	0.411	0.773
Conf	0.593	0.879	0.287	0.715	0.596	0.875	0.228	0.715
Div	0.623	0.625	0.483	0.637	0.623	0.308	0.483	0.637
Acc+Conf	0.643	0.684	0.372	0.639	0.622	0.632	0.266	0.583
Acc+Div	0.643	0.649	0.446	0.649	0.604	0.506	0.232	0.558
Div+Conf	0.638	0.671	0.422	0.640	0.631	0.559	0.363	0.641
Acc+Conf+Div	0.643	0.672	0.413	0.645	0.576	0.614	0.068	0.558

Liver Disorder

	Conf$_1$ Definition				Conf$_2$ Definition			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.727	0.339	0.707	0.746	0.576	0.339	0.707
Conf	0.602	0.947	0.266	0.639	0.602	0.942	0.340	0.637
Div	0.603	0.658	0.510	0.677	0.604	0.285	0.512	0.673
Acc+Conf	0.640	0.742	0.382	0.687	0.630	0.668	0.330	0.618
Acc+Div	0.631	0.689	0.480	0.696	0.618	0.451	0.367	0.608
Div+Conf	0.624	0.718	0.463	0.679	0.626	0.618	0.392	0.677
Acc+Conf+Div	0.635	0.722	0.444	0.699	0.600	0.641	0.192	0.582

Table 3.7: Summary results of the final solution based on its validation data performance for the Conf_1 definition. Classifiers in the initial pool have at least 50% validation accuracy.

	Australian							
	Pareto-Optimal Solutions				*All Solutions*			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.842	0.308	0.824	0.833	0.842	0.308	0.824
Conf	0.588	0.945	0.325	0.797	0.588	0.945	0.325	0.797
Div	0.647	0.671	0.446	0.696	0.647	0.671	0.446	0.697
Acc+Conf	0.661	0.715	0.394	0.636	0.652	0.693	0.408	0.665
Acc+Div	0.657	0.681	0.464	0.674	0.647	0.675	0.445	0.678
Div+Conf	0.648	0.688	0.455	0.675	0.647	0.680	0.443	0.679
Acc+Conf+Div	0.654	0.686	0.453	0.683	0.650	0.680	0.444	0.684

	Diabetes							
	Pareto-Optimal Solutions				*All Solutions*			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.775	0.352	0.745	0.762	0.775	0.352	0.745
Conf	0.669	0.961	0.309	0.734	0.669	0.961	0.309	0.734
Div	0.674	0.701	0.466	0.726	0.674	0.701	0.466	0.726
Acc+Conf	0.700	0.759	0.383	0.733	0.696	0.745	0.402	0.732
Acc+Div	0.686	0.704	0.450	0.734	0.680	0.702	0.458	0.737
Div+Conf	0.672	0.731	0.466	0.726	0.677	0.717	0.458	0.726
Acc+Conf+Div	0.684	0.723	0.450	0.730	0.683	0.720	0.449	0.727

	Heart							
	Pareto-Optimal Solutions				*All Solutions*			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.725	0.411	0.773	0.781	0.725	0.411	0.773
Conf	0.593	0.879	0.287	0.715	0.593	0.879	0.287	0.715
Div	0.639	0.642	0.443	0.677	0.639	0.642	0.443	0.677
Acc+Conf	0.650	0.663	0.402	0.666	0.646	0.654	0.420	0.680
Acc+Div	0.645	0.641	0.448	0.660	0.643	0.642	0.442	0.663
Div+Conf	0.636	0.656	0.451	0.679	0.641	0.648	0.442	0.686
Acc+Conf+Div	0.644	0.649	0.444	0.688	0.643	0.647	0.440	0.689

	Liver Disorder							
	Pareto-Optimal Solutions				*All Solutions*			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.727	0.339	0.707	0.746	0.727	0.339	0.707
Conf	0.602	0.947	0.266	0.639	0.602	0.947	0.266	0.639
Div	0.617	0.663	0.487	0.697	0.617	0.663	0.487	0.697
Acc+Conf	0.646	0.716	0.406	0.692	0.639	0.699	0.428	0.702
Acc+Div	0.630	0.671	0.477	0.703	0.623	0.668	0.471	0.691
Div+Conf	0.616	0.693	0.490	0.695	0.621	0.680	0.479	0.704
Acc+Conf+Div	0.629	0.687	0.471	0.699	0.627	0.681	0.469	0.695

Table 3.8: Summary results of the final solution based on its validation data performance for the Conf_2 definition. Classifiers in the initial pool have at least 50% validation accuracy.

	Pareto-Optimal Solutions				*All Solutions*			
Australian								
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.833	0.737	0.308	0.824	0.833	0.737	0.308	0.824
Conf	0.594	0.940	0.226	0.794	0.594	0.940	0.226	0.794
Div	0.630	0.312	0.493	0.632	0.647	0.379	0.446	0.696
Acc+Conf	0.647	0.531	0.351	0.637	0.638	0.515	0.346	0.639
Acc+Div	0.630	0.412	0.374	0.575	0.594	0.505	0.210	0.574
Div+Conf	0.636	0.464	0.424	0.661	0.639	0.429	0.424	0.673
Acc+Conf+Div	0.595	0.599	0.160	0.572	0.585	0.600	0.125	0.572
Diabetes								
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.762	0.615	0.352	0.745	0.762	0.615	0.352	0.745
Conf	0.669	0.957	0.342	0.731	0.669	0.957	0.342	0.731
Div	0.656	0.368	0.494	0.715	0.671	0.411	0.469	0.728
Acc+Conf	0.700	0.599	0.379	0.723	0.696	0.563	0.396	0.729
Acc+Div	0.680	0.457	0.414	0.709	0.679	0.455	0.409	0.709
Div+Conf	0.672	0.524	0.465	0.729	0.677	0.481	0.458	0.731
Acc+Conf+Div	0.693	0.634	0.340	0.696	0.692	0.625	0.351	0.697
Heart								
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.781	0.556	0.411	0.773	0.781	0.556	0.411	0.773
Conf	0.596	0.875	0.228	0.715	0.596	0.875	0.228	0.715
Div	0.623	0.308	0.483	0.637	0.639	0.352	0.443	0.677
Acc+Conf	0.646	0.462	0.377	0.653	0.644	0.427	0.397	0.657
Acc+Div	0.621	0.394	0.361	0.573	0.604	0.451	0.265	0.573
Div+Conf	0.634	0.418	0.442	0.672	0.639	0.393	0.437	0.684
Acc+Conf+Div	0.620	0.535	0.260	0.598	0.619	0.533	0.261	0.598
Liver Disorder								
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Acc	0.746	0.576	0.339	0.707	0.746	0.576	0.339	0.707
Conf	0.602	0.942	0.340	0.637	0.602	0.942	0.340	0.637
Div	0.604	0.285	0.512	0.673	0.620	0.333	0.484	0.697
Acc+Conf	0.646	0.514	0.399	0.697	0.640	0.465	0.418	0.700
Acc+Div	0.620	0.369	0.433	0.655	0.618	0.371	0.430	0.646
Div+Conf	0.618	0.427	0.485	0.693	0.622	0.390	0.476	0.704
Acc+Conf+Div	0.635	0.544	0.349	0.652	0.633	0.529	0.362	0.652

3.2 Prediction Ranking (PRank)

The results of several experiments performed using dynamic ensemble selection with the First Improvement algorithm are inconclusive, especially in regards to showing that ensembles with accurate and confident classifiers make more accurate predictions. We hypothesize that when the predictions of classifiers are ranked based on their validation accuracy and/or confidence on their predictions, there should be more correct predictions ranked above the incorrect predictions. Therefore, ensembles formed with those classifiers should also achieve higher accuracy. For this reason, we propose a measure called the *prediction ranking*, denoted by $PRank$. Since the relationship between validation accuracy and confidence values for the predictions is not known, we consider three variations of $PRank$: (1) confidence of the classifier on the prediction (confidence), (2) validation accuracy of the classifier which makes the prediction (accuracy), and (3) the average of accuracy and confidence. Once all predictions from all classifiers are combined and sorted based on one of the three criteria, for any given correct prediction, $PRank$ calculates the proportion of incorrect predictions ranked above and below. The following procedure describes how to calculate $PRank_{Conf}$:

1. Combine the predictions for the test dataset from all of the classifiers.
2. Sort them according to their confidence values.
3. For each correct prediction, find the incorrect predictions above (IC_a) and below (IC_b) it.

Table 3.9: Example: Calculating the Prediction Ranking for the Confidence

Correctness	1	0	1	0	0	0	1	0	1	1
Confidence	0.55	0.63	0.74	0.76	0.85	0.85	0.85	0.87	0.93	0.95
$InCor_{Below}$	0	–	1	–	–	–	2	–	5	5
$InCor_{Above}$	5	–	4	–	–	–	1	–	0	0
Subtotal	5	–	5	–	–	–	3	–	5	5

4. Calculate the prediction ranking for confidence:

$$PRank_{Conf} = \frac{\sum IC_b}{\sum IC_b + \sum IC_a} \tag{3.7}$$

$PRank_{Acc}$ and $PRank_{AC}$ are calculated in the same way with the exception that in Step 2 above, the relevant criterion is used for sorting.

Table 3.9 shows an example of how to compute the PRank per Confidence. There are five correct and five incorrect predictions. The first row of the table indicates the correctness of the predictions. Correct and incorrect predictions are shown 1 and 0, respectively. The second row corresponds to the sorted confidence values for each prediction. The next three rows contain the counts IC_b, IC_a, and Subtotal. The table is then filled from left to right. Initially, the first correct prediction is ranked the lowest. Thus, all of the incorrect predictions are ranked higher. Thus, the $IC_b = 0$ and $IC_a = 5$. This process is repeated until we reach the end of the list. The prediction ranking per Confidence is calculates as:

$$PRank_{Conf} = \frac{1+2+5+5}{5+4+3+5+5} = \frac{13}{22}$$

3.2.1 Experimental Results

For this experiment, we use the initial pool of classifiers built for the experiment described in Section 3.1.2. The predictions from all of the classifiers from all of the folds and runs are pooled together to form the data for this analysis. The purpose of this experiment is to analyze which properties (validation accuracy, confidence, or both) of a classifier should be considered when evaluating the correctness of a prediction. However, the relationship between validation accuracy and confidence values for the predictions is not known. Therefore, we combine these two classifier properties in two different ways. First, we take a naive approach and calculate their simple average for each prediction. Then, we use the computed values for sorting the predictions and calculating the prediction ranking, $PRank_{AC}$. In the second approach, we gradually filter out classifiers that do not satisfy a certain cut-off value for validation accuracy, and consider only the predictions from the remaining classifiers. As the cut-off value for the validation accuracy increases, the prediction ranking for confidence, $PRank_{Conf}$, should also increase as the predictions are made by more accurate classifiers.

The results shown in Figure 3.1 and Table 3.10 clearly indicate that accurate and confident classifiers make better predictions. When we start the experiment with all of the classifiers and gradually filtered them out based on the associated validation accuracies, the $PRank_{Acc}$ decreases and $PRank_{Conf}$ increases. At the beginning, the pool includes the classifiers with low validation accuracy which may make incorrect predictions with high confidence values. In this case, the validation accuracy of a classifier can be used as an indicator for correct predictions. Therefore, the $PRank_{Acc}$ is higher than $PRank_{Conf}$. However, as we filter out the bad classifiers, the remaining ones are more accurate and have similar accuracy. Predictions from the classifiers with the same accuracy cannot provide ranking. Thus, instead of accuracy, the confidence of a classifier becomes a strong signal for correct predictions. In other words, if we have two classifiers with the same accuracy, the predictions

Table 3.10: Summary results of prediction ranking for "confidence only", "accuracy only", and "accuracy with confidence".

		Australian		
Min. Val. Acc.[a]	$PRank_{Conf}$	$PRank_{Acc}$	$PRank_{AC}$	Classifiers Left
0.00	0.5558	0.6074	0.5753	25000
0.30	0.5558	0.6073	0.5799	24996
0.40	0.5557	0.6073	0.5819	24985
0.45	0.5646	0.5883	0.5743	19292
0.50	0.5632	0.5869	0.5728	18861
0.55	0.5617	0.5940	0.5725	17696
0.60	0.6668	0.5792	0.6579	2888
0.65	0.6564	0.5455	0.6453	2608
0.70	0.6526	0.5242	0.6408	2422
0.75	0.6520	0.5065	0.6404	2188
0.80	0.6542	0.5017	0.6459	1175
		Diabetes		
Min. Val. Acc.[a]	$PRank_{Conf}$	$PRank_{Acc}$	$PRank_{AC}$	Classifiers Left
0.00	0.5636	0.6540	0.6305	25000
0.30	0.5629	0.6529	0.6425	24853
0.40	0.5702	0.5687	0.5811	19102
0.45	0.5643	0.5561	0.5704	18512
0.50	0.5597	0.5469	0.5628	18008
0.55	0.5563	0.5394	0.5572	17465
0.60	0.5549	0.5346	0.5544	16818
0.65	0.5726	0.5335	0.5702	11969
0.70	0.6261	0.5036	0.6181	4181
0.75	0.6459	0.4971	0.6414	920
0.80	0.6562	0.50445	0.6545	29
		Heart		
Min. Val. Acc.[a]	$PRank_{Conf}$	$PRank_{Acc}$	$PRank_{AC}$	Classifiers Left
0.00	0.5448	0.6118	0.5755	25000
0.30	0.5447	0.6117	0.5819	24988
0.40	0.5439	0.6115	0.5840	24841
0.45	0.5557	0.5821	0.5654	16853
0.50	0.5531	0.5781	0.5617	16356
0.55	0.5524	0.5807	0.5618	15875
0.60	0.6538	0.5558	0.6452	3799
0.65	0.6683	0.5257	0.6572	2751
0.70	0.6743	0.5113	0.6618	1935
0.75	0.6709	0.5011	0.6580	854
0.80	0.6390	0.4923	0.6253	198
		Liver Disorder		
Min. Val. Acc.[a]	$PRank_{Conf}$	$PRank_{Acc}$	$PRank_{AC}$	Classifiers Left
0.00	0.5381	0.5955	0.5614	25000
0.30	0.5381	0.5955	0.5683	24998
0.40	0.5392	0.5951	0.5709	24330
0.45	0.5542	0.5707	0.5659	19617
0.50	0.5550	0.5596	0.5616	17821
0.55	0.5556	0.5543	0.5602	16040
0.60	0.6029	0.5308	0.6002	5493
0.65	0.6046	0.5092	0.5979	3658
0.70	0.6002	0.4964	0.5927	1587
0.75	0.5977	0.4943	0.5919	551
0.80	0.5936	0.4670	0.5877	53

[a] "Min. Val. Acc." represents the cut-off value for the validation accuracy.

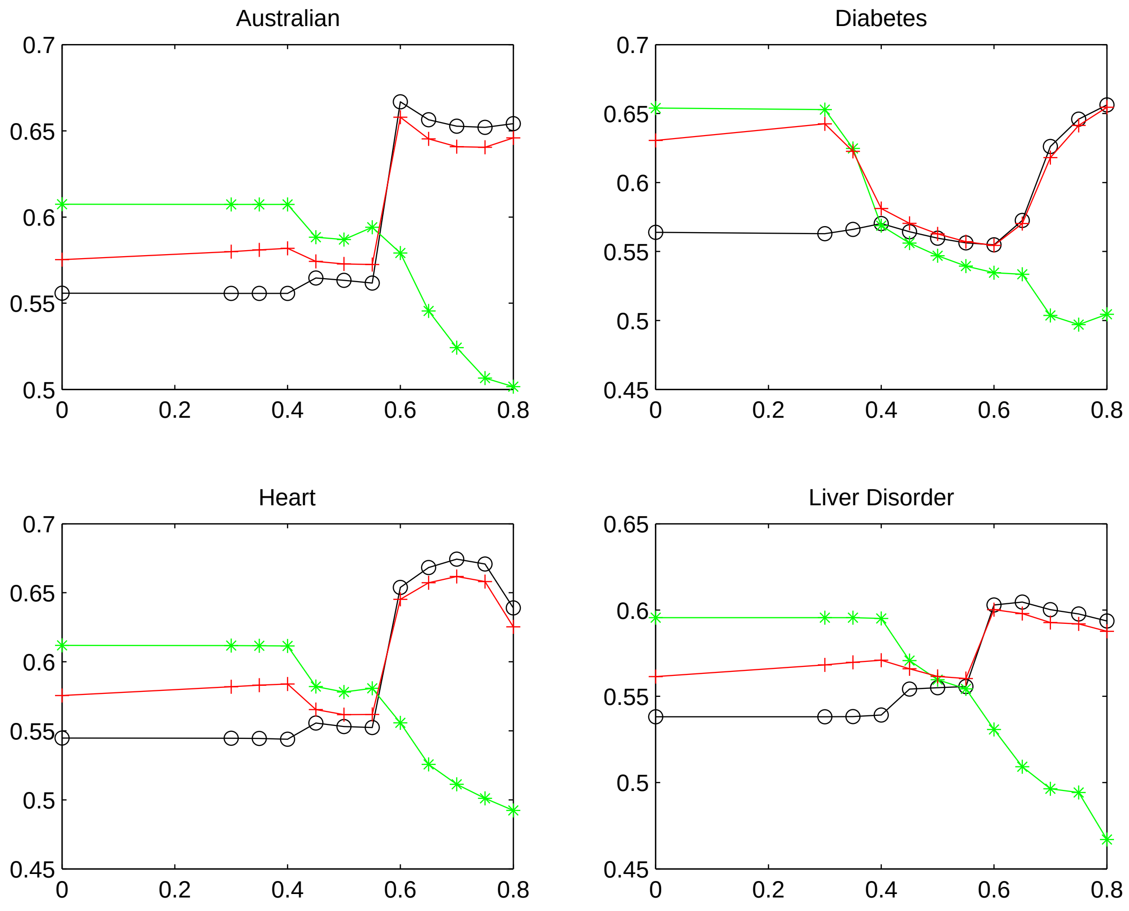


Figure 3.1: Prediction ranking for different validation accuracy cut-off values.

Note: The horizontal axis displays cut-off value for validation accuracy. The vertical axis represents prediction ranking $PRank$. Predictions from classifiers with validation accuracy less than the cut-off value are omitted when prediction ranking is calculated. Black, green, and red lines represents $PRank_{Conf}$, $PRank_{Acc}$, and $PRank_{AC}$, respectively.

made by these classifiers are not comparable by just considering the classifiers' accuracy. In fact, the classifier with the higher confidence value should be chosen to make the prediction. The $PRank_{AC}$ exhibits a similar behavior as $PRank_{Conf}$. However, it never is better than the $PRank_{Conf}$: the validation accuracy of a classifier becomes irrelevant when classifiers reach a certain classification accuracy.

In addition, $PRank_{Conf}$ and $PRank_{AC}$ decrease once the cut-off value exceeds a threshold for the validation accuracy. This is because the number of remaining classifiers with really high validation accuracy decreases drastically as shown in Table 3.10. Consequently, the more accurate they get (in the validation dataset), the more similar they become, so diversity is still a factor affecting the performance.

This experiment attempts to show that accurate, confident and diverse classifiers should make better predictions. Even though results support this hypothesis, we still need to explore the underlying relationship among these factors.

3.3 Logistic Regression

So far, we have established conflicting results in Sections 3.1.2 and 3.2 in investigating if ensembles with accurate, confident, and diverse classifiers make more accurate predictions. The objective function used in Section 3.1.2 is a linear combination of these three factors, and the weights used for these factors were randomly chosen for the generated solutions. However, it is still unclear which set of weights produced the best solutions, or what the underlying relationship among these factors is. In addition, this linear formulation of the objective assumes that there is no interaction between these factors.

Table 3.11: Description of the Logistic Regression Models

Model Name	Specification[a]
Model 1	Y $\sim$ Acc
Model 2	Y $\sim$ Conf
Model 3	Y $\sim$ Div
Model 4	Y $\sim$ Acc + Conf
Model 5	Y $\sim$ Acc + Conf + Acc $\times$ Conf
Model 6	Y $\sim$ Acc + Div
Model 7	Y $\sim$ Acc + Div + Acc $\times$ Div
Model 8	Y $\sim$ Conf + Div
Model 9	Y $\sim$ Conf + Div + Conf $\times$ Div
Model 10	Y $\sim$ Acc + Conf + Div
Model 11	Y $\sim$ Acc + Conf + Div + Acc $\times$ Conf + Acc $\times$ Div + Conf $\times$ Div + Acc $\times$ Conf $\times$ Div

[a] The response Y represents the binary variable for correct prediction.

To address these questions, two experiments are performed. First, several logistic regression models are built to assess the overall significance of the factors affecting ensembles' performance using the whole data. In the second approach, we explore over which accuracy and diversity ranges the confidence of an ensemble is a major predictor for making correct predictions.

3.3.1 Experimental Results

To generate the data for the logistic regression experiments, 100 random ensembles of size 25 are created for each data instance from the pool of trained classifiers discussed in Section 3.1.2. The diversity (Div(E)), the average validation accuracy (Acc(E)), and the confidence (Conf(E,t)) of the ensembles are then calculated. Both confidence definitions described in Section 3.1 are considered. Finally, each ensemble is evaluated for each data instance to determine whether its prediction is correct.

3.3.1.1 Logistic Regression Experiment 1

For each dataset, the models shown in Table 3.11 are built. This process is repeated five times. The results of the experiment for $Conf_1$ and $Conf_2$ are reported in Tables 3.12 and 3.13, respectively. The tables show the *deviance* of the model and the significance of the regression coefficients at the 5% significance level. Deviance is similar to the idea of applying the sum of squares of residuals in the Least Squares approach, and is useful to report for cases where a model estimated using the method of Maximum Likelihood. The goodness of a model is assessed based on its deviance value; a model with lower deviance is better. The first column of Tables 3.12 and 3.13 shows the (mean, standard deviation) of the deviance measure from all five runs for each models. The second column of these tables shows the (mean,standard deviation) of the model accuracy from all five runs. The remainder of the columns indicate how many times a given predictor for the model is individually significant (out of the five runs).

According to Table 3.12, accuracy is not a significant predictor by itself except in the Heart dataset (see model 1), while confidence and diversity each seem to perform better (in models 2 and 3, respectively). When two of these three measures are included, the results improve (models 4, 6, and 8), but the interaction terms do not contribute much (in models 5, 7, and 9). In fact, the deviance does not decrease significantly and the coefficients are more likely to become insignificant. According to our results, having all three measures without interaction terms (model 10) is the best model, as the deviance goes down considerably and the coefficients are mostly individually significant. The results are very similar in Table 3.13, except that the deviance is significantly lower in all models compared to those in Table 3.12.

Table 3.12: Logistic regression results for the $Conf_1$ definition ($\alpha = 0.05$).

Australian

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(93665,239)	(0.584,0.01)	1	–	–	–	–	–	–
Model2	(92779,227)	(0.585,0.01)	–	5	–	–	–	–	–
Model3	(93335,300)	(0.582,0.01)	–	–	5	–	–	–	–
Model4	(92468,259)	(0.591,0.01)	5	5	–	–	–	–	–
Model5	(92454,265)	(0.591,0.01)	4	4	–	4	–	–	–
Model6	(93311,308)	(0.582,0.01)	4	–	5	–	–	–	–
Model7	(93309,308)	(0.582,0.01)	2	–	2	–	1	–	–
Model8	(92157,343)	(0.590,0.01)	–	5	5	–	–	–	–
Model9	(92129,343)	(0.590,0.01)	–	5	5	–	–	4	–
Model10	(92008,344)	(0.594,0.01)	5	5	5	–	–	–	–
Model11	(91958,353)	(0.594,0.01)	2	2	2	2	2	2	2

Diabetes

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(94826,228)	(0.692,0.00)	3	–	–	–	–	–	–
Model2	(93892,349)	(0.692,0.00)	–	5	–	–	–	–	–
Model3	(94814,257)	(0.692,0.00)	–	–	5	–	–	–	–
Model4	(93866,337)	(0.692,0.00)	4	5	–	–	–	–	–
Model5	(93859,341)	(0.692,0.00)	3	2	–	3	–	–	–
Model6	(94585,169)	(0.692,0.00)	5	–	5	–	–	–	–
Model7	(94565,181)	(0.692,0.00)	4	–	4	–	4	–	–
Model8	(93794,348)	(0.692,0.00)	–	5	5	–	–	–	–
Model9	(93785,345)	(0.692,0.00)	–	3	3	–	–	3	–
Model10	(93658,241)	(0.692,0.00)	5	5	5	–	–	–	–
Model11	(93595,244)	(0.692,0.00)	5	5	5	5	5	5	5

Heart

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(35885,135)	(0.612,0.01)	5	–	–	–	–	–	–
Model2	(35118,141)	(0.610,0.01)	–	5	–	–	–	–	–
Model3	(35694,161)	(0.609,0.01)	–	–	5	–	–	–	–
Model4	(35111,145)	(0.611,0.01)	3	5	–	–	–	–	–
Model5	(35102,144)	(0.611,0.01)	3	3	–	3	–	–	–
Model6	(35592,138)	(0.610,0.01)	5	–	5	–	–	–	–
Model7	(35580,134)	(0.610,0.01)	5	–	5	–	5	–	–
Model8	(34877,113)	(0.615,0.01)	–	5	5	–	–	–	–
Model9	(34869,116)	(0.615,0.01)	–	1	3	–	–	3	–
Model10	(34864,97)	(0.615,0.01)	2	5	5	–	–	–	–
Model11	(34824,106)	(0.614,0.01)	1	1	1	1	1	1	1

Liver Disorder

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(45625,211)	(0.625,0.01)	2	–	–	–	–	–	–
Model2	(45399,312)	(0.625,0.01)	–	5	–	–	–	–	–
Model3	(45532,256)	(0.625,0.01)	–	–	5	–	–	–	–
Model4	(45391,311)	(0.625,0.01)	2	5	–	–	–	–	–
Model5	(45380,308)	(0.625,0.01)	3	3	–	2	–	–	–
Model6	(45512,273)	(0.625,0.01)	3	–	5	–	–	–	–
Model7	(45507,272)	(0.625,0.01)	3	–	1	–	1	–	–
Model8	(45262,338)	(0.624,0.01)	–	5	5	–	–	–	–
Model9	(45248,342)	(0.625,0.01)	–	2	3	–	–	4	–
Model10	(45246,340)	(0.624,0.01)	4	5	5	–	–	–	–
Model11	(45208,337)	(0.625,0.01)	2	2	2	2	2	1	1

Table 3.13: Logistic regression results for the $Conf_2$ definition ($\alpha = 0.05$).

Australian

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(93665,239)	(0.584,0.01)	1	–	–	–	–	–	–
Model2	(89569,407)	(0.612,0.01)	–	5	–	–	–	–	–
Model3	(93335,300)	(0.582,0.01)	–	–	5	–	–	–	–
Model4	(88540,506)	(0.636,0.00)	5	5	–	–	–	–	–
Model5	(88127,699)	(0.638,0.00)	5	5	–	5	–	–	–
Model6	(93311,308)	(0.582,0.01)	4	–	5	–	–	–	–
Model7	(93309,308)	(0.582,0.01)	2	–	2	–	1	–	–
Model8	(75327,946)	(0.732,0.01)	–	5	5	–	–	–	–
Model9	(71912,1142)	(0.763,0.00)	–	5	5	–	–	5	–
Model10	(74003,878)	(0.733,0.01)	5	5	5	–	–	–	–
Model11	(69470,1102)	(0.769,0.00)	5	5	5	5	5	5	5

Diabetes

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(94826,228)	(0.692,0.00)	3	–	–	–	–	–	–
Model2	(89724,769)	(0.699,0.01)	–	5	–	–	–	–	–
Model3	(94814,257)	(0.692,0.00)	–	–	5	–	–	–	–
Model4	(88556,1039)	(0.700,0.00)	5	5	–	–	–	–	–
Model5	(88402,1005)	(0.701,0.00)	5	5	–	5	–	–	–
Model6	(94585,169)	(0.692,0.00)	5	–	5	–	–	–	–
Model7	(94565,181)	(0.692,0.00)	4	–	4	–	4	–	–
Model8	(87437,642)	(0.709,0.00)	–	5	5	–	–	–	–
Model9	(87363,647)	(0.709,0.00)	–	5	5	–	–	5	–
Model10	(87270,710)	(0.708,0.00)	4	5	5	–	–	–	–
Model11	(87044,672)	(0.710,0.00)	5	5	5	5	5	4	4

Heart

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(35885,135)	(0.612,0.01)	5	–	–	–	–	–	–
Model2	(34636,299)	(0.633,0.01)	–	5	–	–	–	–	–
Model3	(35694,161)	(0.609,0.01)	–	–	5	–	–	–	–
Model4	(34498,347)	(0.635,0.01)	5	5	–	–	–	–	–
Model5	(34492,341)	(0.636,0.01)	4	5	–	3	–	–	–
Model6	(35592,138)	(0.610,0.01)	5	–	5	–	–	–	–
Model7	(35580,134)	(0.610,0.01)	5	–	5	–	5	–	–
Model8	(31591,356)	(0.685,0.01)	–	5	5	–	–	–	–
Model9	(31040,370)	(0.700,0.01)	–	5	5	–	–	5	–
Model10	(31407,407)	(0.686,0.01)	5	5	5	–	–	–	–
Model11	(30529,338)	(0.704,0.01)	5	5	5	5	5	5	5

Liver Disorder

	Dev	Model Acc	Acc	Conf	Div	Acc Conf	Acc Div	Conf Div	Acc Div Conf
Model1	(45625,211)	(0.625,0.01)	2	–	–	–	–	–	–
Model2	(44681,409)	(0.629,0.01)	–	5	–	–	–	–	–
Model3	(45532,256)	(0.625,0.01)	–	–	5	–	–	–	–
Model4	(44571,428)	(0.633,0.01)	5	5	–	–	–	–	–
Model5	(44555,422)	(0.633,0.01)	4	4	–	4	–	–	–
Model6	(45512,273)	(0.625,0.01)	3	–	5	–	–	–	–
Model7	(45507,272)	(0.625,0.01)	1	–	1	–	1	–	–
Model8	(43832,567)	(0.651,0.01)	–	5	5	–	–	–	–
Model9	(43777,590)	(0.651,0.01)	–	5	5	–	–	5	–
Model10	(43787,541)	(0.652,0.01)	5	5	5	–	–	–	–
Model11	(43682,559)	(0.653,0.01)	1	1	3	1	1	1	1

Table 3.14: Beta values in model 10.

	Conf$_1$ Definition				Conf$_2$ Definition			
Australian								
	Intercept	Acc	Conf	Div	Intercept	Acc	Conf	Div
Run1	-5.6573	-5.5017	12.6596	2.5063	-2.5639	-17.5092	15.0365	19.9921
Run2	-5.0361	-5.6619	12.4422	1.5691	-2.3783	-16.7989	14.3218	18.8959
Run3	-5.2644	-6.0351	13.2773	1.4791	-0.15526	-20.4622	14.4459	18.4641
Run4	-5.1766	-4.5167	11.6312	1.716	-1.5382	-18.2346	14.248	18.8874
Run5	-5.0956	-5.6275	12.3901	1.7929	-3.0063	-18.3737	16.3648	20.9814
Diabetes								
	Intercept	Acc	Conf	Div	Intercept	Acc	Conf	Div
Run1	-9.8375	6.1132	6.9379	4.7102	-5.6519	0.41152	5.0942	10.184
Run2	-7.8664	2.2049	8.781	2.8035	-2.1086	-5.4031	6.0623	9.3774
Run3	-7.7236	3.8702	6.5391	3.6121	-4.1372	-2.6267	5.5559	10.3991
Run4	-7.7347	1.5168	9.1457	2.745	-1.9717	-5.004	5.7728	8.7564
Run5	-6.4255	1.1986	7.7113	2.5673	-1.1788	-6.5845	5.9873	8.9376
Heart								
	Intercept	Acc	Conf	Div	Intercept	Acc	Conf	Div
Run1	-13.0559	3.1337	15.699	4.7857	-7.6478	-11.4295	12.9056	24.4371
Run2	-11.5497	1.3854	15.4982	3.8867	-6.0142	-13.0473	13.0636	22.9733
Run3	-13.7502	5.0976	15.8396	3.8558	-7.6361	-5.9812	12.0192	18.9395
Run4	-10.9849	0.2232	15.5666	4.0115	-4.7328	-13.7103	12.6643	21.3005
Run5	-12.8734	-0.0011	18.8719	3.9258	-6.4838	-9.9538	11.6143	20.9693
Liver Disorder								
	Intercept	Acc	Conf	Div	Intercept	Acc	Conf	Div
Run1	-3.9126	-1.6688	6.8863	1.6776	0.32608	-6.1744	3.6762	5.7779
Run2	-4.6417	2.4498	3.6864	2.8412	-1.8838	-2.7563	3.512	6.5025
Run3	-2.8214	0.53075	3.4023	1.6625	-1.1331	-4.0267	3.5169	6.44
Run4	-5.7942	3.2448	4.6101	3.1859	-3.4248	-1.1974	3.8997	7.9174
Run5	-7.3983	1.802	7.7816	3.7693	-3.7197	-2.5959	4.5793	9.6789

Given that we select model 10 as the best model, we provide some insights to the magnitude and sign of the coefficients in Table 3.14 for the two confidence definitions. According to Table 3.14, all three measures have a positive impact (except in the Australian Dataset). Higher classifier accuracy, ensemble confidence, and diversity each increase the probability of correct classification, after accounting for the other two factors. The sign of the coefficient on accuracy switches with the second confidence definition. This implies that holding the ensemble confidence and diversity constant, classifiers with higher accuracy are less likely to classify correctly. Considering the deviance and beta values obtained for Model 10 with the $Conf_2$ definition, we conclude that using an ensemble with high diversity and high confidence margin (margin implies level of agreement) on the prediction increases the correctness of predictions.

3.3.1.2 Logistic Regression Experiment 2

In the second experiment, we focus on Model 2 (see the description in Table 3.11) and explore in detail whether confidence is a good predictor for certain ranges of accuracy and diversity values. To do this, we divide the data into 16 partitions based on ensemble accuracy and diversity. We first sort classifiers according to their accuracy and diversity values, and then categorize them into four groups, with respect to each variable, as shown in Tables 3.15 and 3.16. For instance, the $75Perc$ category for accuracy includes the classifiers whose accuracy is between 50% and 75%, while the $25Perc$ category for diversity includes classifiers whose diversity is between 0% and 25%.

For each partition, we build a logistic regression model with just confidence as a predictor. For each of these combinations, we report, in the first row, the mean and standard deviation of the deviance as well as the number of runs (out of 5 runs) for which the coefficient on confidence is significant at the 5% significance level. In the second row, we show the mean and standard error of the coefficient and the number of runs (out of 5 runs) for which the coefficient is positive.

To illustrate, consider the combination $(100Perc, 100Perc)$ (the top left cell) in Table 3.15. The deviance of this model for these classifiers is 3232 with a standard deviation of 687. The coefficient is individually significant in all 5 runs, and its average value in these runs equals 16 with a standard error of 2. Finally, the coefficient is positive in all five runs.

As can be seen from Table 3.15, the coefficient on confidence is positive and statistically significant in all five runs (except for the Liver Disorder sataset where it is significant in three runs). The average value of the coefficient is different across the four datasets: the coefficient is highest in the Heart Dataset, followed by Australian, Diabetes, and Liver Disorder. Additionally, the coefficient on the confidence increases with the percentile range for accuracy (e.g 10 for $25Perc$ accuracy and 16 for $100Perc$ accuracy given $100Perc$ for diversity in the Australian dataset).

While the results on individual significance and the coefficient sign stay similar with the second confidence definition, as shown in Table 3.16, the coefficient seems to *decrease* with higher accuracy percentile ranges. It is also worth noting that while the deviance is not comparable across different percentile ranges, it is comparable across different confidence definitions for any given percentile range. According to Tables 3.15 and 3.16, the deviance is lower with the second confidence definition across the board for any dataset and any percentile range. Therefore, margin confidence as a predictor better fits the logistic model.

The first confidence definition ($Conf_1$) does not provide any indication of whether the classifiers agree on the prediction; it only implies how confident the classifiers in the ensemble are. Table 3.15 shows that to make better predictions, we need high accuracy and diversity as well. This is because highly accurate and diverse classifiers will make similar (accurate) predictions with high confidence. In contrast, the bottom right corner of the table clearly shows that confidence does not have high impact (high beta value) for ensembles that are formed by similar classifiers that are not accurate. On the other hand, the second confidence definition is based on the margin definition, which is also an indicator of the level fo agreement between the classifiers on the prediction. The results in Table 3.16 show that if the classifiers have a high level of agreement on the decision and if they are diverse, the overall performance of the classifiers becomes irrelevant.

3.4 Conclusion

We introduce two ensemble confidence definitions, $Conf_1$ and $Conf_2$, and examine the role of classifier confidence in several experimental settings. In a multi-objective optimization setting, the primary problem to overcome is performance: dynamic optimization for each data instance requires significant time and reduces the size and complexity of the datasets we can reasonably experiment on. Our results, based on a thorough examination of four small-scale datasets, indicate that confidence (and therefore, dynamic construction) leads to little performance improvement for some datasets and none for others. One challenge with these small-scale datasets is to maintain reasonable accuracy and diversity in the initial

Table 3.15: Summary results of model 2 for the $Conf_1$ definition

Australian[a]

Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(3232,687)	5	(4216,474)	5	(6345,900)	5	(8682,797)	5
	(16,2)	5	(11,2)	5	(11,1)	5	(16,2)	5
75Perc	(4086,407)	5	(5000,340)	5	(6305,355)	5	(7386,134)	5
	(15,2)	5	(11,1)	5	(14,1)	5	(15,3)	5
50Perc	(5363,312)	5	(6009,171)	5	(6071,365)	5	(5509,553)	5
	(14,2)	5	(11,3)	5	(12,2)	5	(12,4)	5
25Perc	(8579,1252)	5	(8260,661)	5	(4665,869)	5	(1664,106)	5
	(10,2)	5	(8,3)	5	(10,1)	5	(12,2)	5

Diabetes[a]

Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(732,358)	4	(1741,590)	5	(5379,308)	5	(15559,1263)	5
	(12,6)	5	(12,2)	5	(11,1)	5	(9,2)	5
75Perc	(3267,727)	5	(5042,372)	5	(8567,1167)	5	(6328,518)	5
	(10,2)	5	(9,1)	5	(7,1)	5	(5,2)	5
50Perc	(7891,1093)	5	(7285,789)	5	(6181,577)	5	(1688,676)	2
	(9,1)	5	(7,1)	5	(6,1)	5	(1,3)	4
25Perc	(10848,1445)	5	(9220,874)	5	(3470,993)	3	(350,152)	0
	(9,1)	5	(7,1)	5	(4,1)	5	(2,3)	5

Heart[a]

Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(1747,135)	5	(940,57)	5	(1814,107)	5	(3872,167)	5
	(20,1)	5	(22,2)	5	(16,2)	5	(17,1)	5
75Perc	(2146,88)	5	(1337,63)	5	(2180,133)	5	(2794,95)	5
	(24,3)	5	(23,5)	5	(15,5)	5	(17,2)	5
50Perc	(2541,148)	5	(2017,75)	5	(2282,144)	5	(1633,147)	3
	(22,3)	5	(19,5)	5	(15,2)	5	(11,5)	5
25Perc	(1217,215)	5	(4448,183)	5	(2793,181)	1	(757,176)	1
	(20,6)	5	(9,3)	5	(4,2)	5	(2,7)	3

Liver Disorder[a]

Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(1935,552)	4	(1756,439)	5	(2835,571)	5	(4602,810)	5
	(7,4)	5	(6,2)	5	(6,1)	5	(5,2)	5
75Perc	(2547,319)	5	(2241,432)	5	(2828,233)	5	(3611,573)	3
	(8,3)	5	(8,1)	5	(4,1)	5	(4,3)	5
50Perc	(3277,309)	4	(2806,327)	5	(2754,467)	5	(2328,415)	3
	(7,4)	5	(7,3)	5	(7,2)	5	(3,4)	4
25Perc	(3183,691)	3	(4365,713)	3	(2988,402)	3	(1072,826)	1
	(5,3)	5	(4,2)	5	(3,4)	4	(0,5)	3

[a] Each cell contains: mean and standard deviation of the deviance (**top left**), the coefficient with its standard error (**bottom left**), the number of runs for which the coefficient is significant (**top right**) and positive (**bottom right**).

Table 3.16: Summary results of model 2 for the $Conf_2$ definition

Australian[a]								
Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(2727,535)	5	(2776,330)	5	(4293,602)	5	(6584,898)	5
	(10,1)	5	(12,1)	5	(13,1)	5	(12,1)	5
75Perc	(3418,361)	5	(3314,262)	5	(4398,339)	5	(6295,114)	5
	(13,1)	5	(17,1)	5	(18,2)	5	(13,2)	5
50Perc	(4573,308)	5	(4305,134)	5	(4694,309)	5	(5179,465)	5
	(15,1)	5	(20,2)	5	(21,1)	5	(10,2)	5
25Perc	(7712,1194)	5	(7364,592)	5	(4356,805)	5	(1667,111)	5
	(17,2)	5	(14,1)	5	(14,1)	5	(4,1)	5
Diabetes[a]								
Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(696,347)	5	(1634,560)	5	(4978,249)	5	(14093,1186)	5
	(5,1)	5	(5,0)	5	(5,0)	5	(5,0)	5
75Perc	(3085,708)	5	(4728,332)	5	(7870,1116)	5	(5625,374)	5
	(6,0)	5	(5,0)	5	(6,0)	5	(7,1)	5
50Perc	(7505,1078)	5	(6823,759)	5	(5685,437)	5	(1545,599)	5
	(6,0)	5	(6,1)	5	(6,1)	5	(7,1)	5
25Perc	(10537,1383)	5	(8865,894)	5	(3257,917)	5	(320,132)	5
	(6,0)	5	(6,1)	5	(6,1)	5	(6,1)	5
Heart[a]								
Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(1684,122)	5	(862,42)	5	(1532,137)	5	(3291,250)	5
	(9,1)	5	(9,1)	5	(10,1)	5	(10,1)	5
75Perc	(2060,124)	5	(1216,83)	5	(1824,169)	5	(2483,123)	5
	(12,2)	5	(12,1)	5	(14,1)	5	(12,1)	5
50Perc	(2401,166)	5	(1828,114)	5	(1958,106)	5	(1518,131)	5
	(15,1)	5	(14,1)	5	(16,1)	5	(11,0)	5
25Perc	(1155,202)	5	(4183,174)	5	(2573,195)	5	(703,161)	5
	(17,1)	5	(13,1)	5	(12,1)	5	(11,3)	5
Liver Disorder[a]								
Acc\Div	100Perc		75Perc		50Perc		25Perc	
100Perc	(1920,554)	5	(1721,404)	5	(2723,573)	5	(4280,765)	5
	(3,1)	5	(3,1)	5	(3,0)	5	(4,1)	5
75Perc	(2519,320)	5	(2181,414)	5	(2710,213)	5	(3410,540)	5
	(4,1)	5	(4,1)	5	(4,1)	5	(4,1)	5
50Perc	(3245,314)	5	(2747,329)	5	(2659,451)	5	(2225,405)	5
	(4,1)	5	(4,0)	5	(4,1)	5	(4,1)	5
25Perc	(3142,674)	5	(4266,686)	5	(2920,430)	5	(1051,819)	5
	(4,1)	5	(4,1)	5	(3,1)	5	(3,1)	5

[a] Each cell contains: mean and standard deviation of the deviance (**top left**), the coefficient with its standard error (**bottom left**), the number of runs for which the coefficient is significant (**top right**) and positive (**bottom right**).

classifier pool. A potential solution can be to experiment on larger-scale datasets.

We propose the *PRank* method to show when confidence of a classifier could be considered as a criteria for making correct predictions. Our analysis indicate that confidence becomes a major indicator after the classifiers in the pool reach a certain level of accuracy. Unfortunately, this level is different for each dataset.

The logistic regression results for the margin confidence definition are encouraging and highlight the benefits of having confident and diverse classifiers as having accuracy in addition to confidence and diversity does not improve the performance based on the results presented in Table 3.13. Margin confidence considers the agreement level among the classifiers in the ensemble on the prediction. Thus, instead of taking into account the overall accuracy of a classifier, we may focus on the local accuracy of a classifier, which can more information regarding how much classifiers truly agree on data instances similar to the new instance.

CHAPTER 4
DYNAMIC CLASS PREDICTION WITH CLASSIFIER BASED DISTANCE MEASURE

In this chapter, we continue our analysis of the dynamic class prediction models and set up a system based on a new distance measure to evaluate the similarity between data points and to make predictions. We propose a method to make dynamic predictions to address the following questions:

- To find the k-nearest neighbors of a new instance, should the original feature space or a classifier-based space be used?
- Is using class probability estimates better than just classifiers' predictions to find similar instances?
- Is classifiers' performance on the validation instances helpful when finding similar instances?
- Should all of the classifiers in the pool be used for computing similarity between instances? If not, what criteria should be considered to eliminate certain classifiers?
- After neighbors are found, should we use them to form an ensemble or to make a prediction directly?

In Sections 4.1 and 4.2, the general structure of the system and its running time are explained. In Sections 4.3 and 4.4, we explain proposed and baseline distance measures. We carry out several experiments to evaluate performance of our framework. The results are presented in Section 4.5. Finally, Section 4.8 summarizes this chapter.

4.1 Proposed Framework

The prediction probability returned by a classifier can be considered as a measure of proximity of a data instance to the decision boundary. This measure can be used to compare multiple instances and to decide whether that particular classifier considers those data instances similar. Our proposed framework consists of two main steps: a static step and a dynamic step. We illustrate both steps as a flow chart in Figure 4.1.

In the static step, we have a pool of classifiers, $\mathcal{C}$, of size M, and we map data instances into a new space defined by the class probability estimates from each classifier for a given class label. Since we consider two-class problems, the choice of class label does not change our results. Next, we find each new instance's k-nearest neighbors in the validation set, of size N, using this space. These k neighbors can be seen as the data instances that, on average, the classifiers agree are similar to the new instance. This step is termed "static" as, for all new instances, we consider the output of all M classifiers in the pool.

In the dynamic step of our framework (on the right in Figure 4.1), we use the results from the static step to select a subset of classifiers $\mathcal{C}'$, of size $E < M$, from the original pool, $\mathcal{C}$, that are suitable for classifying a given new instance. Reducing the size of the space is important because the number of classifiers in our framework is large and (dis)similarity becomes less meaningful as the feature space dimensionality increases. Our selection method favors classifiers that have high confidence in their predictions for the neighbors identified in the static step (i.e. classifiers for which the predictions are far away from the 50% decision boundary).[1] Confidence of a classifier C_m on k neighbors is calculated as follows:

$$Conf_m = \frac{\sum_{i=1}^{k} \Pr(\ell_{i,m} \mid i)}{k} \tag{4.1}$$

where $\ell_{i,m}$ represents the label assigned to instance i by classifier C_m and $\Pr(\ell_{i,m}|i)$ represents the probability estimate returned by the classifier for the assigned class label. In other words, the confidence of a classifier on k neighbors is the average of probability estimates returned for the decisions. The most confident E classifiers are chosen to form the reduced space, $\mathcal{C}'$. Reducing the size of the classifier set in this manner avoids the unstable behavior that may occur near the decision boundary of some classifiers.

Once the classifier subset, $\mathcal{C}'$, is generated, the original distance measure is reapplied to find a new set of neighbors $\mathcal{V}^*$ for the new instance. This neighborhood is then used for the final classification of the instance; the new instance is assigned to the class label most common among its k neighbors.

4.2 Running Time Analysis

To analyze the running time of the system, we first evaluate the static and dynamic steps separately. The running time of the static step can be analyzed in two parts: (1) forming the probability space, and (2) finding k neighbors of a new instance. Forming the probability space consists of getting predictions for N validation instances from M classifiers, so the computational time is $O(M \times N)$. This is a preprocessing step as once this space is formed, it will remain the same for all new data instances. Finding k neighbors requires getting prediction for each new instance from M classifiers ($O(M)$), calculating the distance

[1]For completeness, other selection methods are considered in Section 4.7.4.

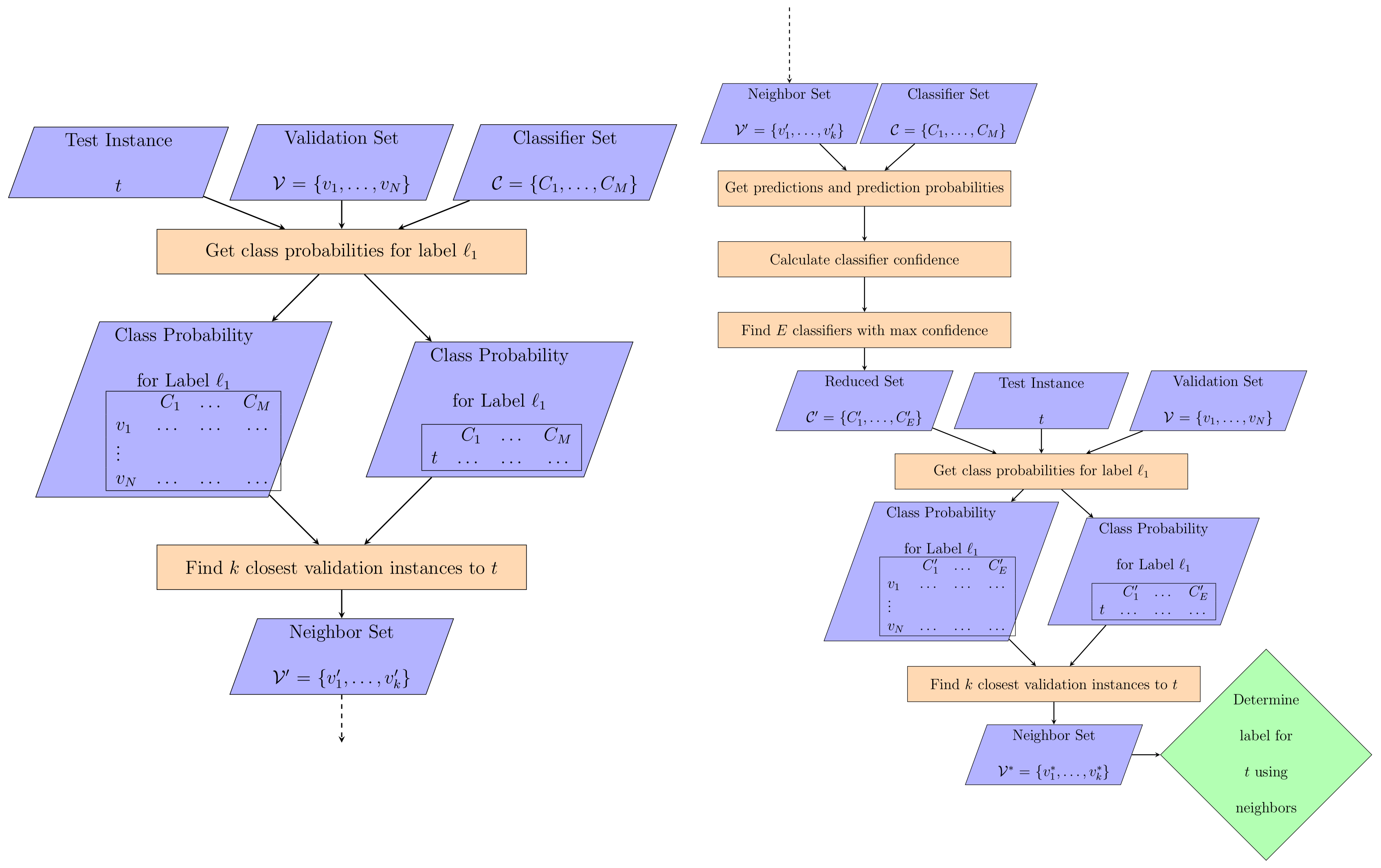

Test Instance
t
Validation Set
$\mathcal{V} = \{v_1, \ldots, v_N\}$
Classifier Set
$\mathcal{C} = \{C_1, \ldots, C_M\}$
Get class probabilities for label ℓ_1
Class Probability
for Label ℓ_1
C_1 … C_M
v_1
v_N
Class Probability
for Label ℓ_1
C_1 … C_M
t
Find k closest validation instances to t
Neighbor Set
$\mathcal{V}' = \{v'_1, \ldots, v'_k\}$
Neighbor Set
$\mathcal{V}' = \{v'_1, \ldots, v'_k\}$
Classifier Set
$\mathcal{C} = \{C_1, \ldots, C_M\}$
Get predictions and prediction probabilities
Calculate classifier confidence
Find E classifiers with max confidence
Reduced Set
$\mathcal{C}' = \{C'_1, \ldots, C'_E\}$
Test Instance
t
Validation Set
$\mathcal{V} = \{v_1, \ldots, v_N\}$
Get class probabilities for label ℓ_1
Class Probability
for Label ℓ_1
C'_1 … C'_E
v_1
v_N
Class Probability
for Label ℓ_1
C'_1 … C'_E
t
Find k closest validation instances to t
Neighbor Set
$\mathcal{V}^* = \{v^*_1, \ldots, v^*_k\}$
Determine
label for
t using
neighbors

Figure 4.1: An overview of the proposed framework: Static Step (on the left) and Dynamic Step (on the right)

between the new instance and validation instances ($O(M \times N)$), and finally finding k instances that are closest to the new instance ($O(k \times N)$). Overall, the running time for this static step is $O(M \times N)$, which is for calculating the distance between a new instance and the validation instance.

The dynamic step is composed of two main parts: (1) reducing the classifier space, and (2) finding new k neighbors of a new instance. Since predictions for validation instances and k neighbors of a new instance are already found in the static step, the running time of reducing the space consists of calculating the confidence of classifiers on the neighbors ($O(k \times M)$) and finding the most confident E classifiers ($O(E \times M)$). The running time of the rest of the dynamic step can be analyzed in a similar manner to the static step ($O(k \times E)$). The overall running time of the whole system is dominated by the running time of the static step as $k < N$ and $E < M$.

4.3 Baseline Distance Measures

In this section, we consider three different distance measures for our baseline: euclidean distance, template matching, and oracle-based template matching. Euclidean distance in the original feature space is used as a benchmark since we hypothesize that the space in which similarity among data instances assessed should be different than the ones used for classifier training. Even though template matching and oracle-based template matching measures [9] use classifier output space instead of the original feature space, they only consider assigned labels. We argue that including probability estimates returned by the classifiers can prove useful for assessing the similarity among data instances. These measures are explained in further detail below.

4.3.1 Euclidean Distance

We first consider the Euclidean Distance (ED) between two data instances i and j:

$$ED_{i,j} = \sqrt{\frac{\sum_{d=1}^{D} (\phi_{i,j,d})^2}{D}} \tag{4.2}$$

$$\phi_{i,j,d} = F_d(i) - F_d(j) \tag{4.3}$$

where a dataset consists of the feature set $\mathcal{F} = \{F_d \mid d = 1, \ldots, D\}$ and $F_d(i)$ represents the value of feature F_d for instance i.

4.3.2 Template Matching

Given a set of classifiers $\mathcal{C} = \{C_m \mid n = 1, \ldots, M\}$, Template Matching (TM) considers the percentage of classifiers, denoted by $TM_{i,j}$, that agree on the label of test instance i and validation instance j:

$$TM_{i,j} = \frac{1}{M} \sum_{m=1}^{M} \alpha_{i,j,m} \tag{4.4}$$

$$\alpha_{i,j,m} = \begin{cases} 1, & \text{if } \ell_{i,m} = \ell_{j,m} \\ 0, & \text{otherwise.} \end{cases} \tag{4.5}$$

where $\ell_{j,m}$ represents the label assigned to instance j by classifier C_m and $\alpha_{i,j,m}$ represents whether classifier C_m assigns the same label to instances i and j. It follows that the higher $TM_{i,j}$, the more similar the pair of instances (i, j).

4.3.3 Oracle-Based Template Matching

Oracle-based Template Matching (OTM) extends TM by incorporating the *correctness* of classifier predictions on data instances. Specifically, when evaluating the similarity of test instance i to validation instance j using OTM, only those classifiers that correctly classify j are considered.

Let $\ell_{j,m}$ indicate the label assigned to instance j by classifier C_m, while $\ell_j^\star$ is the correct label for instance j. This measure, denoted by $OTM_{i,j}$, can be formulated as follows:

$$OTM_{i,j} = \frac{\sum\limits_{m=1}^{M} \beta_{i,j,m}}{\sum\limits_{m=1}^{M} \gamma_{j,m}} \tag{4.6}$$

$$\beta_{i,j,m} = \begin{cases} 1, & \text{if } \ell_{i,m} = \ell_{j,m} = \ell_j^\star \\ 0, & \text{otherwise.} \end{cases} \tag{4.7}$$

$$\gamma_{j,m} = \begin{cases} 1, & \text{if } \ell_{j,m} = \ell_j^\star \\ 0, & \text{otherwise.} \end{cases} \tag{4.8}$$

In this formulation, $\beta_{i,j,m}$ indicates if classifier C_m correctly predicts both instances i and j, while $\gamma_{j,m}$ equals 1 if classifier C_m correctly predicts instance j. Similar to TM, a higher value of $OTM_{i,j}$ implies greater similarity between instances i and j.

4.4 Proposed Measures

For our dynamic class prediction framework, we propose two new distance measures: Probability-Based Template Matching and Probability-Based Template Matching with Accuracy. These measures are described in detail below.

4.4.1 Probability-Based Template Matching

Probability-Based Template Matching (PTM) maps each data instance into an alternate feature space constructed by using the probability estimates of each classifier in the pool as the values of the features. As mentioned before, the probability estimates for the two-class problems are taken with respect to a particular class label, and the choice of label is arbitrary. The similarity between instances i and j, denoted by $PTM_{i,j}$, is calculated as the Euclidean distance between them in this alternate feature space:

$$PTM_{i,j} = \sqrt{\frac{\sum_{m=1}^{M} (\phi_{i,j,m})^2}{M}} \tag{4.9}$$

$$\phi_{i,j,m} = \widetilde{\Pr}_{i,m} - \widetilde{\Pr}_{j,m} \tag{4.10}$$

where $\widetilde{\Pr}_{i,n}$ represents the probability estimate in the alternative feature space for instance i returned by classifier C_m. Similar to the ED, the pair of instances (i, j) with the smallest $PTM_{i,j}$ is considered to be the most similar.

4.4.2 Probability-Based Template Matching with Accuracy

Probability-Based Template Matching with Accuracy (PTMA) integrates the correctness of classifiers on validation instance j. We focus on the probability estimates returned by the classifier for the *correct* class label of the new and validation instances. Therefore, we make adjustments on the probability estimates returned for the validation instance. For example, if instance j is misclassified by classifier C_m, we swap the probability estimates returned for both class labels for j by C_m because the assigned label for j is incorrect and it should have the lower probability estimate. Since we only consider two-class problems, swapping is straightforward: $1 - \Pr_{C_m,j}$. There are four different cases associated with the adjustment of probability estimates when we consider the assigned labels of instances i and j, and the correctness of the prediction for instance j. We do not know the true label of the new instance i: we cannot really decide whether the probability estimates returned for instance i need adjustments. Therefore, each case specified in equation (4.11) below should have equal weight. To avoid one case eliminating the effect of the other cases, we re-scale all values to be between $[0, 1]$.

$$PTMA_{i,j} = \sqrt{\frac{\sum_{m=1}^{M} (\delta_{i,j,m})^2}{M}} \tag{4.11}$$

$$\delta_{i,j,m} = \begin{cases} 2 \mid \mathrm{Pr}_{i,m} - \mathrm{Pr}_{j,m} \mid, & \text{if } \ell_{i,m} = \ell_{j,m} \& \ell_{j,m} = \ell_j^\star \\ \mid 1 - \mathrm{Pr}_{i,m} - \mathrm{Pr}_{j,m} \mid, & \text{if } \ell_{i,m} = \ell_{j,m} \& \ell_{j,m} \neq \ell_j^\star \\ \mid \mathrm{Pr}_{i,m} - \mathrm{Pr}_{j,m} \mid, & \text{if } \ell_{i,m} \neq \ell_{j,m} \& \ell_{j,m} = \ell_j^\star \\ 2 \mid 1 - \mathrm{Pr}_{i,m} - \mathrm{Pr}_{j,m} \mid, & \text{if } \ell_{i,m} \neq \ell_{j,m} \& \ell_{j,m} \neq \ell_j^\star \end{cases} \tag{4.12}$$

We illustrate how these measures differ with a simple example. Table 4.1 shows the assigned labels and the corresponding probability estimates for class label l_1 returned by classifiers $C_1, ..., C_5$. Suppose that v_1 is the instance we would like to predict.

Four out of five classifiers in our example assigned the same label to instances v_1 and v_2, while only two classifiers assigned the same label to v_1 and v_3. Thus, $TM_{1,2} = 4/5 = 0.8$ and $TM_{1,3} = 2/5 = 0.4$. Meanwhile, as explained above, OTM accounts for correctness of assigned labels. Only two classifiers (out of five given) correctly predict v_2, whose actual label is l_2. Out of these two classifiers, only one assigned the same label to v_1. Thus, $OTM_{1,2} = 0.5$. Similar algebra yields $OTM_{1,3} = 0.66$. Consequently, we may conclude that v_1 and v_2 are more similar than v_1 and v_3, according to TM, and the similarity is about the

Table 4.1: Assigned class labels and class probabilities by classifiers

Assigned Class Labels						
	C_1	C_2	C_3	C_4	C_5	Correct Label
v_1	1	1	2	1	1	–
v_2	1	1	2	1	2	2
v_3	2	2	1	1	1	1
Class Probabilities for $\ell = 1$						
	C_1	C_2	C_3	C_4	C_5	
v_1	0.53	0.57	0.44	0.90	0.85	
v_2	0.92	0.89	0.24	0.52	0.35	
v_3	0.46	0.43	0.53	0.88	0.90	

same according the OTM measure.

Let's turn our attention to the PTM and PTMA measures. Given the class probabilities for $\ell = 1$ in Table 4.1, we can directly calculate the two PTM measures:

$$PTM_{1,2} = \sqrt{\frac{0.39^2 + 0.32^2 + 0.20^2 + 0.38^2 + 0.50^2}{5}} \cong 0.3712 \tag{4.13}$$

$$PTM_{1,3} = \sqrt{\frac{0.07^2 + 0.14^2 + 0.29^2 + 0.02^2 + 0.05^2}{5}} \cong 0.0843 \tag{4.14}$$

which implies that v_1 and v_3 are considered more similar instances according to the PTM measure. For the PTMA measure, this result is more pronounced. Using the formula in equation (4.11), we calculate $PTMA_{1,2} = 0.9973$ and $PTMA_{1,3} = 0.1118$. As a result, considering only the assigned labels to determine similarity may lead to incorrect decisions. This is especially true for instances that are closer to the decision boundary of a classifier, as the classifier provides less confidence in its prediction for those instances.

4.5 Experimental Setup

In our experiments, we use 13 data sets with varying numbers of features and data instances retrieved from the LIBSVM website ([10]). A summary of the data sets and the

Table 4.2: Summary of the data sets and classifiers

Data Set	#Data Points	#Features	Class Ratio[a]	%Good Classifiers[b]	Diversity[c]
a1a	1605	119	0.33	100	0.126
australian	690	14	0.80	64.17	0.229
breast cancer	683	10	0.54	97.14	0.050
diabetes	768	8	0.54	99.74	0.222
german	1000	24	0.43	100	0.208
heart	269	13	0.79	59.44	0.286
ionosphere	351	34	0.56	88.75	0.071
liver disorder	345	6	0.73	79.46	0.310
mushrooms	8124	112	0.93	54.05	0.006
rcv[d]	20242	47236	0.93	56.65	0.050
sonar	208	60	0.87	54.04	0.300
splice	1000	60	0.93	52.77	0.277
w1a	2477	300	0.03	100	0.004

[a] The variable *Class Ratio* represents the ratio of minority class to the majority class.
[b] The variable *%Good Classifiers* represents the ratio of classifiers with at least 50% accuracy to the total number of classifiers over 100 runs.
[c] The variable *Diversity* represents the diversity of the classifiers in the pool over 100 runs based disagreement measure.
[d] 1000 classifiers are generated for all datasets except the *rcv* dataset, for which 100 classifiers are generated due to its size.

classifiers generated for each is presented in Table 4.2. The last column of this table illustrates the percent of classifiers with at least 50% accuracy, referred to as *%GoodClassifiers*.

The programming code was written in MATLAB and LIBSVM ([10]) was used to construct RBF kernel SVM classifiers. The training parameters were chosen such that classifiers overfit the data. Each experiment was repeated 100 times, with each run registering a unique seed value for the random number generator.

For each run, the data sets are randomly divided into three subsets such that 60% of the instances is used to train classifiers, 20% is used for validation, and the remaining 20% is used for testing. A pool of 1000 classifiers is then constructed for each data set using a combination of bootstrap instance sampling (as in bagging) and random subspace selection on the training set.[2] In doing so, we ensure that the initial classifier pool is highly diverse. Finally, classifiers with an error rate above 50% are removed from the pool. The last column of Table 4.2 represents the percentage of the classifiers with less than 50% error rate in the generated pool of classifiers.

In the following sections, we first analyze the static step. In Section 4.6.1, we perform an experiment to set the value of k for baseline and proposed (dis)similarity measures. Then, we compare the effectiveness of these measures to find the closest k neighbors to make the predictions in Section 4.6.2. In Section 4.6.3, we investigate whether the classifier-based feature space can improve the *KNORA* method results. In addition, we choose the appropriate distance measure for *KNORA* before comparing it with the static step. We conclude our comparison for the static step by assessing its performance against common benchmarks in Section 4.6.4.

Once our comparison for the static step is done, we turn our attention to the dynamic step. We first determine the optimal reduced space size in Section 4.7.1. In Section 4.7.2, we examine the contribution of the dynamic step in our framework in addition to the static step, while we compare the dynamic step against common benchmarks in Section 4.7.3. We then explore various strategies in evaluating the dynamic step in Section 4.7.4. Finally, in Section 4.7.5, we evaluate the performance of the dynamic step against that of the ensembles formed by the classifiers for the reduced space.

[2]Due to its size, only 100 classifiers are generated for the *rcv* data set.

4.6 Results: Static Step

4.6.1 Evaluation of Neighborhood Size

The number of neighbors considered for each similarity measure is crucial. As a preprocessing step, we perform an experiment in which k is varied over the odd integers from 1 to 25 to find the optimal value. In this step, for each run of a data set, we predict the validation instances using training instances and decide for which k value maximum accuracy is obtained. We repeated this 100 times, and considered two versions of the experiment: unweighted and weighted averages. In the weighted version, the vote of each neighbor is multiplied by the inverse of its distance to the new instance.

Table 4.3 shows the unweighted and weighted cases for the best k value for each similarity measure, averaged over all of the runs. We find that the average value of k is smaller for the OTM (ranging from 1 to 4) and PTMA (ranging from 1 to 3) measures for both weighted and unweighted versions. A possible explanation is that these two measures take into account the correctness of the decision associated with the validation instances. Consequently, a small neighborhood of validation instances is sufficient to correctly classify the new instance.

Table 4.4 shows the average accuracy of each similarity measure for the validation data instances. The results indicate that there is very little difference between the weighted and unweighted versions. Consequently, for the rest of the experiments carried out in this chapter, we consider the unweighted case in which each neighbor has equal weight.

We also analyze the percentage of common neighbors between proposed measures and baseline measures for the unweighted case. For a given data set and a k value, we look at the

Table 4.3: Average best k value for similarity measures

	Unweighted Case				
Dataset	TM	OTM	ED	PTM	PTMA
a1a	9.60	3.30	14.24	9.80	1.42
australian	9.34	1.50	12.14	11.24	1.46
breast cancer	9.06	1.66	5.14	7.74	1.08
diabetes	11.14	3.22	13.34	12.30	2.66
german	14.52	3.52	14.08	13.68	1.28
heart	7.48	1.56	13.32	7.06	1.86
ionosphere	9.64	2.16	2.84	9.34	1.12
liver disorder	10.02	2.20	9.02	12.12	3.82
mushrooms	9.56	1.00	1.00	10.48	1.00
rcv	16.50	4.90	1.94	16.00	3.62
sonar	5.94	1.12	1.86	8.26	1.22
splice	3.46	1.16	10.02	5.48	1.00
w1a	1.34	1.00	2.20	1.34	1.00
	Weighted Case				
Dataset	TM	OTM	ED	PTM	PTMA
a1a	9.60	3.30	14.42	9.8	1.42
australian	9.34	1.50	12.94	11.24	1.46
breast cancer	9.06	1.66	5.40	7.74	1.08
diabetes	11.14	3.22	13.62	12.3	2.66
german	14.52	3.52	15.08	13.68	1.28
heart	7.48	1.56	12.92	7.06	1.86
ionosphere	9.64	2.16	2.86	9.34	1.12
liver disorder	10.02	2.20	10.82	12.12	3.82
mushrooms	9.56	1.00	1.00	10.48	1.00
rcv	16.50	4.90	2.90	16.00	3.62
sonar	5.94	1.12	2.44	8.26	1.22
splice	3.46	1.16	10.00	5.48	1.00
w1a	1.34	1.00	2.08	1.34	1.00

Table 4.4: Summary results for validation instances in the preprocessing step

			Unweighted Case[a]		
Dataset	TM	OTM	ED	PTM	PTMA
a1a	(71.87,4.04)	(76.83,0.02)	(80.61,0.93)	(71.57,3.90)	(77.74,0.01)
australian	(84.85,0.82)	(85.50,0.01)	(85.01,1.37)	(84.44,1.10)	(85.47,0.02)
breast cancer	(73.31,3.39)	(96.61,0.01)	(96.52,0.31)	(73.44,2.99)	(96.64,0.01)
diabetes	(59.91,2.60)	(75.28,0.04)	(73.19,1.25)	(60.93,2.01)	(75.17,0.02)
german	(66.96,3.42)	(71.17,0.02)	(71.14,1.46)	(67.39,3.26)	(71.34,0.01)
heart	(80.74,0.42)	(81.55,0.06)	(81.34,2.21)	(80.46,0.40)	(79.80,0.03)
ionosphere	(73.49,4.03)	(93.25,0.03)	(80.96,3.14)	(72.74,3.33)	(93.96,0.02)
liver disorder	(69.65,0.90)	(70.18,0.07)	(62.11,0.73)	(67.85,1.61)	(68.46,0.10)
mushrooms	(95.96,4.13)	(100.0,0.00)	(99.98,0.02)	(96.06,4.10)	(100.0,0.00)
rcv	(63.56,5.29)	(97.05,0.00)	(92.56,0.38)	(65.09,5.05)	(96.91,0.00)
sonar	(80.24,0.35)	(81.28,0.08)	(71.66,4.37)	(79.73,0.44)	(61.10,0.02)
splice	(56.37,0.16)	(56.37,0.01)	(68.14,0.85)	(56.45,0.31)	(47.48,0.01)
w1a	(96.80,0.86)	(97.16,0.01)	(97.19,0.12)	(96.89,0.63)	(97.14,0.00)

			Weighted Case[a]		
Dataset	TM	OTM	ED	PTM	PTMA
a1a	(71.88,3.99)	(76.83,0.02)	(80.33,0.82)	(71.64,3.70)	(77.75,0.01)
australian	(84.87,0.83)	(85.50,0.01)	(84.76,1.38)	(84.51,1.07)	(85.48,0.02)
breast cancer	(73.48,3.59)	(96.61,0.01)	(96.58,0.33)	(73.62,3.20)	(96.64,0.01)
diabetes	(59.89,2.51)	(75.29,0.05)	(73.44,1.27)	(61.12,2.03)	(75.17,0.01)
german	(66.95,3.49)	(71.17,0.02)	(71.76,1.51)	(67.39,3.35)	(71.34,0.01)
heart	(80.75,0.47)	(81.56,0.07)	(81.22,2.04)	(80.51,0.34)	(79.80,0.03)
ionosphere	(73.45,3.95)	(93.25,0.04)	(81.97,2.55)	(73.05,3.23)	(93.96,0.01)
liver disorder	(69.80,0.90)	(70.20,0.06)	(63.91,0.79)	(67.97,1.53)	(68.47,0.09)
mushrooms	(96.16,4.14)	(100.0,0.00)	(100.0,0.00)	(96.27,4.07)	(100.0,0.00)
rcv	(63.74,5.19)	(97.05,0.01)	(93.23,0.33)	(65.12,5.01)	(96.91,0.01)
sonar	(80.33,0.26)	(81.30,0.09)	(74.71,3.77)	(79.76,0.47)	(61.09,0.02)
splice	(56.38,0.12)	(56.37,0.01)	(68.48,0.60)	(56.45,0.25)	(47.48,0.00)
w1a	(96.80,0.81)	(97.16,0.01)	(97.19,0.14)	(96.95,0.37)	(97.14,0.00)

[a] The values in parentheses are average accuracy and its standard deviation, respectively.

Table 4.5: Common neighbors between PTM(A) and baseline measures

	PTM vs.[a]		
Dataset	TM	OTM	ED
a1a	(0, 20.58, 52.83)	(0, 2.22, 4.89)	(0, 0.21, 0.70)
australian	(0, 16.99, 39.57)	(0, 5.94, 13.32)	(0, 0.77, 2.53)
breast cancer	(0, 26.13, 72.87)	(0, 1.75, 4.30)	(0, 0.59, 1.92)
diabetes	(0, 11.96, 24.99)	(0, 2.96, 6.88)	(0, 0.44, 1.44)
german	(0, 16.31, 38.67)	(0, 3.68, 8.31)	(0, 0.34, 1.14)
heart	(0, 18.17, 42.50)	(0, 9.67, 21.33)	(0, 1.92, 6.20)
ionosphere	(0, 25.72, 73.11)	(0, 2.50, 6.71)	(0, 1.04, 3.42)
liver disorder	(0, 8.61, 18.73)	(0, 5.11, 12.42)	(0, 1.11, 3.55)
mushrooms	(0, 30.36, 91.73)	(0, 0.12, 0.40)	(0, 0.067, 0.22)
rcv	(0, 14.69, 47.40)	(0, 14.77, 47.62)	(0, 14.74, 47.54)
sonar	(0, 17.49, 39.86)	(0, 9.91, 22.15)	(0, 2.14, 6.57)
splice	(0, 10.41, 21.18)	(0, 4.52, 10.61)	(0, 0.28, 0.95)
w1a	(0, 26.98, 75.33)	(0, 0.22, 0.64)	(0, 0.14, 0.47)

	PTMA vs.[a]			
Dataset	TM	OTM	ED	PTM
a1a	(0, 1.22, 3.05)	(0, 1.01, 2.83)	(0, 0.32, 1.02)	(0, 1.19, 3.03)
australian	(0, 1.03, 3.25)	(0, 1.09, 3.50)	(0, 0.84, 2.74)	(0, 1.12, 3.43)
breast cancer	(0, 0.70, 2.23)	(0, 1.12, 3.33)	(0, 0.97, 3.12)	(0, 0.64, 2.08)
diabetes	(0, 0.65, 2.08)	(0, 0.91, 2.93)	(0, 0.58, 1.94)	(0, 1.11, 3.07)
german	(0, 0.87, 2.53)	(0, 1.31, 3.72)	(0, 0.37, 1.20)	(0, 1.07, 2.99)
heart	(0, 2.32, 7.56)	(0, 2.44, 8.06)	(0, 1.93, 6.32)	(0, 2.62, 8.17)
ionosphere	(0, 1.5, 4.51)	(0, 3.06, 8.88)	(0, 1.42, 4.38)	(0, 1.49, 4.51)
liver disorder	(0, 1.63, 5.10)	(0, 1.86, 6.08)	(0, 1.15, 3.67)	(0, 3.31, 8.71)
mushrooms	(0, 0.15, 0.53)	(0, 2.45, 5.72)	(0, 0.17, 0.55)	(0, 0.15, 0.52)
rcv	(0, 14.71, 47.47)	(0, 14.79, 47.72)	(0, 14.75, 47.59)	(0, 14.77, 47.67)
sonar	(0, 1.98, 6.23)	(0, 2.09, 5.68)	(0, 1.67, 5.51)	(0, 2.18, 6.83)
splice	(0, 0.55, 1.39)	(0, 0.04, 0.12)	(0, 0.45, 1.54)	(0, 1.45, 6.81)
w1a	(0, 2.41, 5.76)	(0, 0.84, 2.03)	(0, 0.21, 0.66)	(0, 1.64, 4.00)

[a] The values in parentheses are (min, mean, max) percentage of common neighbors.

percentage of common neighbors (training instances) found to predict the tuning instance between proposed measures and baseline measures. The results are averaged over 100 runs. Then, we repeat this experiment for odd numbered k values between 1 and 25.

The entries in Table 4.5, specified as (x_1, x_2, x_3), represent (minimum, mean, maximum) of the percentage of overlaps in the neighborhood. PTM is most common with TM, with the percent value having a mean around 15–20% and being as much as 92%. PTM does not have many common neighbors with OTM or ED, for which the percent values are mostly less than 10%. Meanwhile, PTMA does not have many common neighbors with any of the other measures, including PTM, and the average percent value is around 1–2% (with the exception of the *rcv* dataset). In short, PTM and TM have more common neighbors than others. Nonetheless, PTM and PTMA find different neighbors than other baseline measures to make the predictions.

4.6.2 Evaluation of Distance Measures

After the best k value is determined for each of the similarity measures presented in Section 4.3, we perform experiments to assess the accuracy of these measures in classifying new data instances. These experiments use the k-NN algorithm to define the neighborhood for and classify each new instance based on the validation instances deemed to be similar. Table 4.6 summarizes the results of these experiments: we report mean and standard deviation of accuracy from 100 runs in parentheses, respectively. Entries highlighted in bold in Table 4.6 indicate the method that achieved the best accuracy for the corresponding data set. While the mean accuracy varies considerably across datasets, its variation across the (dis)similarity measures is not statistically significant in most cases. However, it is important to note that PTMA achieves the highest mean accuracy in 7 out of 13 datasets. In doing so, PTMA also managed to reduce the standard deviation, which can be quite high (around 15–20% in some cases), especially for OTM and TM. Consequently, PTMA achieves better performance, on average, than the other measures.

We also perform pairwise t-tests at the 5% significance level to compare the proposed distance measures with the baseline measures. Table 4.7 illustrates the relevant results.

Table 4.6: Summary results of (dis)similarity measures for the unweighted case

Dataset[a,b]	TM	OTM	ED	PTM	PTMA
a1a	(79.65, 1.34)	(76.18, 10.63)	(78.92, 1.28)	**(81.11, 2.01)**	(77.54, 1.04)
australian	(85.21, 2.84)	(84.2, 5.22)	(85.76, 3.03)	(85.54, 2.93)	**(86.06, 2.70)**
breast cancer	(96.26, 2.43)	(96.03, 4.64)	(96.13, 1.67)	(96.54, 1.49)	**(96.82, 1.38)**
diabetes	(71.03, 11.74)	(52.8, 20.31)	(72.04, 3.26)	(74.15, 3.87)	**(75.36, 2.50)**
german	(71.35, 1.20)	(71.9, 1.12)	(71.16, 2.07)	**(72.72, 1.79)**	(71.46, 0.92)
heart	(81.96, 5.16)	(82.97, 4.76)	(80.68, 5.36)	(80.83, 5.56)	**(83.08, 5.12)**
ionosphere	(90.93, 12.01)	(82.48, 22.58)	(80.71, 5.93)	(93.74, 2.79)	**(94.04, 2.49)**
liver disorder	(69.14, 6.15)	**(71.06, 5.71)**	(58.09, 5.71)	(69.14, 5.11)	(70.45, 4.09)
mushrooms	(99.75, 0.09)	(98.67, 0.08)	(98.56, 0.05)	(99.84, 0.04)	**(99.87, 0.07)**
rcv	(86.77, 16.49)	(55.51, 17.08)	(90.54, 0.51)	(96.93, 0.38)	**(97.04, 0.26)**
sonar	(83.16, 5.84)	**(85.83, 5.63)**	(72.15, 7.73)	(82.87, 6.65)	(77.42, 6.54)
splice	(69.05, 7.92)	(66.25, 9.03)	(65.76, 3.82)	**(77.55, 4.84)**	(53.5, 0.96)
w1a	(97.18, 0.27)	(92.91, 0.10)	**(97.19, 0.43)**	(97.17, 0.41)	(97.13, 0.15)

[a] The values in parentheses are the mean and standard deviation of accuracy.
[b] For each dataset, the entries in bold represents the method that achieved maximum mean accuracy.

Table 4.7: Pairwise t-test results for dis/similarity measures

	TM	OTM	ED	PTM
PTM	(6,6,1)	(8,2,3)	(10,3,0)	-
PTMA	(7,3,3)	(5,5,3)	(8,3,2)	(6,3,4)

Entries are specified as (x_1, x_2, x_3) and represent (wins,ties,losses) of the proposed methods against the other measures in all 13 datasets. For example, when compared to the TM measure, the PTM measure performs statistically significantly better for 6 datasets and worse for only 1 dataset. For the remaining 6 datasets, the differences between two measures is not statistically significant. PTM, TM and ED use the same distance measure but in different spaces. As demonstrated by the pairwise t-test results for the PTM measure against the TM and ED measures, the use of the class probability space improves accuracy over either the class prediction space or the original feature space, even without consideration of the classifiers' accuracies for the validation instances. Based on the results for the PTMA measure against the PTM measure, we can conclude that integration of the classifier accuracy into the distance measure further improves accuracy.

4.6.3 Static Step vs. KNORA Method

Ko et al. [37] propose a method which, for each new data instance, finds its nearest k neighbors in the validation set using the original feature set, and dynamically chooses an ensemble based on the estimated accuracy of the classifiers in this local region of the new data instance. Two different versions of this method are proposed by [37] in terms of how classifiers are chosen to form an ensemble for the new instance: KNORA-Eliminate and KNORA-Union. KNORA-Eliminate uses the classifiers that correctly classify every instance in the local region of the new instance, whereas the KNORA-Union method utilizes classifiers that correctly predict at least one of the neighbors.

Our proposed framework in Section 4.1 in both static and dynamic steps finds k nearest neighbors of a new instance. Unlike *KNORA*, which uses the neighbors to form an ensemble of classifiers, our proposed framework uses their labels to make predictions. In this section, we perform experiments to compare our static step against *KNORA* methods. In addition, *KNORA* uses the Euclidean distance measure to find the neighbors of a new instance and the choice of distance measure had no effect on the performance of *KNORA* ([72]). However, the measures considered by [72] are based on finding the distance in the original feature space. We extend [72] to analyze the effectiveness of *KNORA* in the classifier-based space and find the best distance measure for the *KNORA* methods before comparing with the static step.

4.6.3.1 Evaluating Similarity Measures for KNORA Method

KNORA-Eliminate and *KNORA-Union* are implemented using the measures discussed in Sections 4.3 and 4.4: TM, OTM, ED, PTM, and PTMA. A pairwise t-test at the 5% significance level is performed to compare PTM and PTMA against other measures. Entries in Table 4.8 represent the number of (wins, ties, and losses), respectively, of the proposed measures against the baseline measures (including PTMA vs. PTM) for the *KNORA* methods in all 13 datasets.

For the *KNORA-Eliminate* method, PTM does not outperform TM or OTM, especially when the k value is bigger. For low k values, PTM, on average, wins 5–6 datasets while losing around 3–4. However, as k increases, OTM and TM outperform PTM. Therefore, PTM seems to be more suitable for smaller k values. Meanwhile, PTMA wins convincingly against all four measures (including PTM): for any k value, PTMA does not lose more than 5 datasets, whereas as k increases, PTMA is clearly dominant, winning up to 10–11 datasets.

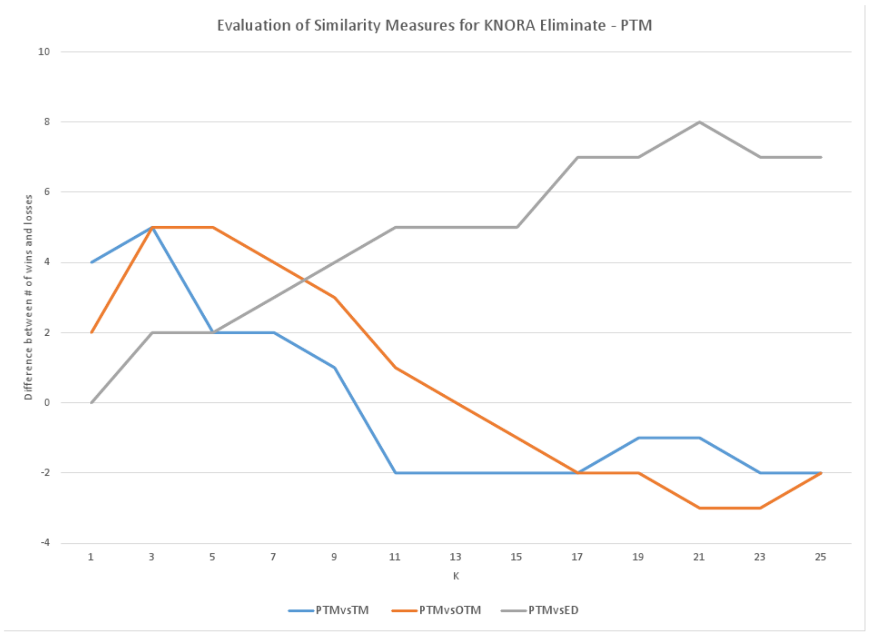
For the *KNORA-Union* method, while the proposed measures prove to be more useful, PTM, in particular, performs better than the baseline measures and PTMA. While PTM does not lose more than 1 dataset, we see a lot of ties in this method. Meanwhile, PTMA performs considerably worse compared to others, winning around 2 datasets.

Figures 4.2 and 4.3 visualize these results. The vertical axis indicates the difference between wins and losses across all 13 datasets for a given k value, whereas the horizontal axis indicates the k values. The top and bottom plots are for the *KNORA-Eliminate* and *KNORA-Union* methods, respectively. As a result of this experiment, we can conclude that *KNORA* performs well with our proposed measures in a classifier based space.

Table 4.8: Pairwise t-test: comparison of (dis)similarity measures

KNORA-Eliminate Method							
	PTM vs.			*PTMA vs.*			
k	TM	OTM	ED	TM	OTM	ED	PTM
1	(5,7,1)	(6,3,4)	(3,7,3)	(7,1,5)	(5,5,3)	(6,2,5)	(6,3,4)
3	(6,6,1)	(7,4,2)	(5,5,3)	(7,2,4)	(6,4,3)	(8,1,4)	(8,1,4)
5	(5,5,3)	(8,2,3)	(3,9,1)	(7,2,4)	(8,2,3)	(8,2,3)	(7,2,4)
7	(5,5,3)	(7,3,3)	(5,6,2)	(7,3,3)	(7,3,3)	(8,3,2)	(7,3,3)
9	(4,6,3)	(6,4,3)	(6,5,2)	(5,5,3)	(8,2,3)	(9,1,3)	(7,5,1)
11	(4,3,6)	(6,2,5)	(7,4,2)	(5,6,2)	(7,3,3)	(10,1,2)	(9,3,1)
13	(4,3,6)	(5,3,5)	(7,4,2)	(5,5,3)	(6,4,3)	(11,1,1)	(10,2,1)
15	(4,3,6)	(5,2,6)	(7,4,2)	(5,6,2)	(6,3,4)	(11,1,1)	(11,1,1)
17	(4,3,6)	(5,1,7)	(8,4,1)	(6,5,2)	(5,3,5)	(11,1,1)	(11,1,1)
19	(4,4,5)	(5,1,7)	(8,4,1)	(7,5,1)	(5,3,5)	(11,1,1)	(10,2,1)
21	(4,4,5)	(4,2,7)	(9,3,1)	(10,3,0)	(5,3,5)	(11,1,1)	(11,1,1)
23	(2,7,4)	(4,2,7)	(8,4,1)	(10,3,0)	(6,2,5)	(11,1,1)	(11,1,1)
25	(2,7,4)	(4,3,6)	(8,4,1)	(10,3,0)	(7,2,4)	(11,1,1)	(11,2,0)

KNORA-Union Method							
	PTM vs.			*PTMA vs.*			
k	TM	OTM	ED	TM	OTM	ED	PTM
1	(5,7,1)	(6,3,4)	(3,6,4)	(7,1,5)	(5,5,3)	(7,1,5)	(6,3,4)
3	(7,6,0)	(6,5,2)	(5,7,1)	(2,6,5)	(4,5,4)	(2,5,6)	(1,6,6)
5	(5,8,0)	(6,6,1)	(5,7,1)	(2,4,7)	(3,5,5)	(2,6,5)	(2,3,8)
7	(5,8,0)	(4,8,1)	(3,9,1)	(2,4,7)	(2,7,4)	(2,6,5)	(1,5,7)
9	(5,8,0)	(4,8,1)	(6,6,1)	(1,5,7)	(1,8,4)	(2,7,4)	(1,5,7)
11	(3,10,0)	(4,6,3)	(4,9,0)	(2,4,7)	(2,6,5)	(2,5,6)	(1,5,7)
13	(3,9,1)	(5,6,2)	(5,8,0)	(1,5,7)	(1,7,5)	(2,5,6)	(1,5,7)
15	(5,7,1)	(4,8,1)	(4,8,1)	(1,5,7)	(1,7,5)	(2,7,4)	(1,5,7)
17	(3,9,1)	(3,9,1)	(5,7,1)	(1,6,6)	(1,7,5)	(2,8,3)	(1,5,7)
19	(3,10,0)	(4,8,1)	(5,7,1)	(1,7,5)	(1,8,4)	(2,8,3)	(1,6,6)
21	(4,8,1)	(4,8,1)	(5,7,1)	(1,8,4)	(1,8,4)	(2,5,6)	(1,6,6)
23	(3,9,1)	(2,9,2)	(5,6,2)	(1,7,5)	(1,7,5)	(2,5,6)	(1,7,5)
25	(3,10,0)	(1,11,1)	(5,7,1)	(1,9,3)	(1,9,3)	(2,7,4)	(1,7,5)



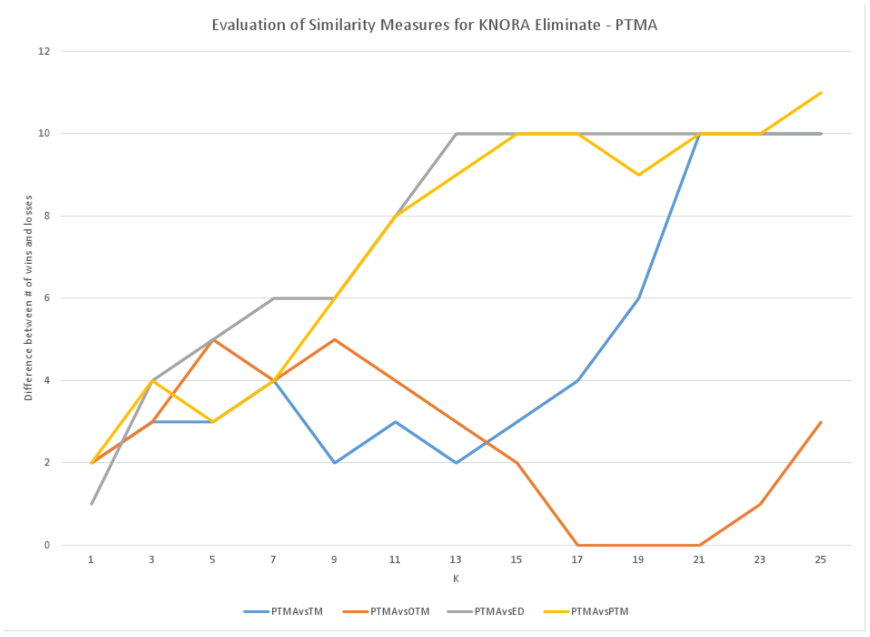
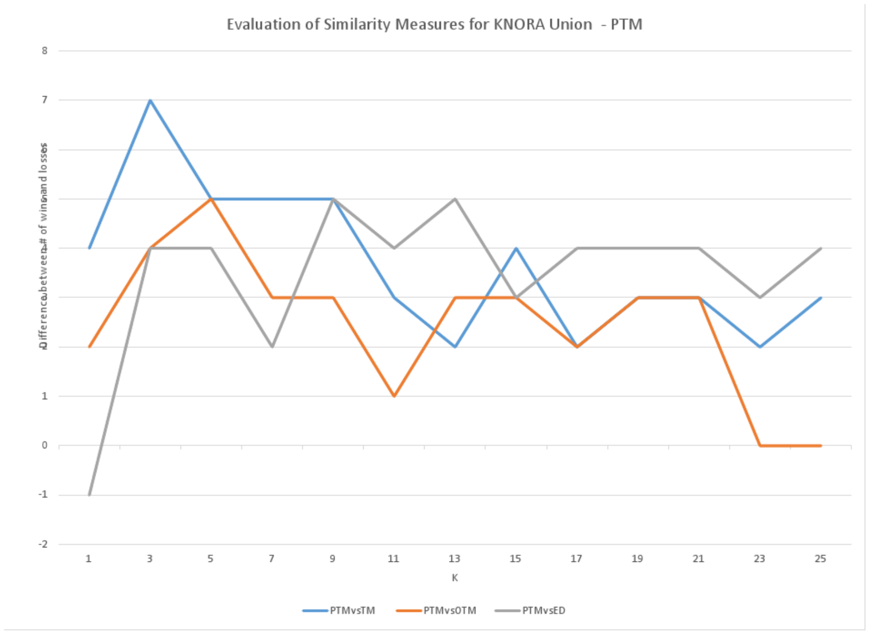


Figure 4.2: Comparisons of PTM(A) against baseline measures for KNORA-Eliminate



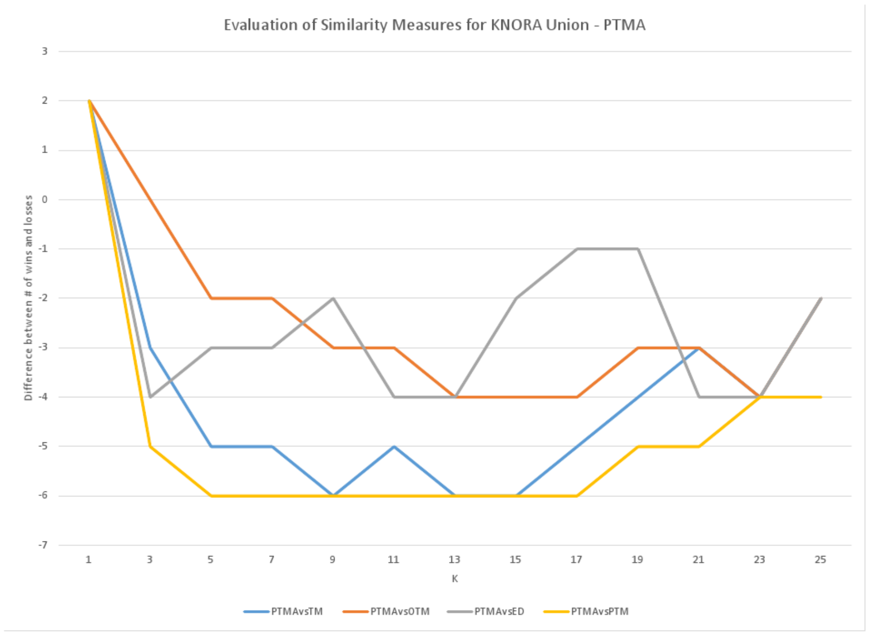


Figure 4.3: Comparisons of PTM(A) against baseline measures for KNORA-Union

4.6.3.2 Comparison of Static Step with KNORA Method

Since the size of an ensemble used to make predictions for each new instance changes for the *KNORA*-Eliminate and -Union methods, we analyze the performance of *KNORA*-Eliminate and -Union against the *static* step of our framework. We explore whether k-NN (Static Step) is better than *KNORA*, after neighbors are found by using the proposed measures, since the experiments in Section 4.6.3.1 indicate that *KNORA* performs better with PTM and PTMA measures compared to ED, TM, and OTM. Table 4.9 shows the t-test performed to compare *KNORA* and k-NN in the classifier-based probability space.

Figure 4.4 illustrates the difference between the number of wins and losses of the *KNORA* methods against the static step of our framework. For instance, for $k = 1$ and distance measure PTM, static step of our framework outperforms *KNORA*-Eliminate for 10 datasets and under-performs for only 1 dataset. For the remaining 2 datasets, there is a tie. Therefore, for $k = 1$, Figure 4.4 indicates $1 - 10 = -9$. The results in Table 4.9 and in Figure 4.4 show that the static step (k-NN with PTM or PTMA) outperforms the *KNORA*-Eliminate and -Union methods with PTM or PTMA. More specifically, the static step with either proposed measure wins at least half of the 13 datasets, with the static step using the PTM measure winning up to 12 datasets.

4.6.4 Static Step vs. Common Benchmarks

Since the set of classifiers used in the static step is constant for all test instances, we run experiments to compare the static step against some of the common static classifier/ensemble selection methods to assess the merit of our framework. The baseline bench-

Table 4.9: Pairwise t-test: KNORA methods vs. static step

	KNORA-Eliminate vs.[a]		*KNORA-Union vs.*[a]	
k	PTM-S	PTMA-S	PTM-S	PTMA-S
1	(1,2,10)	(3,3,7)	(1,2,10)	(3,3,7)
3	(1,2,10)	(3,3,7)	(5,4,4)	(3,4,6)
5	(1,1,11)	(3,1,9)	(5,4,4)	(3,3,7)
7	(0,2,11)	(3,2,8)	(5,4,4)	(3,4,6)
9	(0,2,11)	(3,1,9)	(5,3,5)	(3,4,6)
11	(0,1,12)	(3,1,9)	(4,4,5)	(3,5,5)
13	(0,1,12)	(3,1,9)	(5,3,5)	(3,3,7)
15	(0,1,12)	(2,2,9)	(4,4,5)	(3,4,6)
17	(0,1,12)	(2,1,10)	(4,2,7)	(3,4,6)
19	(0,1,12)	(2,1,10)	(4,2,7)	(3,5,5)
21	(0,1,12)	(2,1,10)	(4,2,7)	(3,3,7)
23	(0,1,12)	(2,1,10)	(4,2,7)	(3,2,8)
25	(0,1,12)	(2,1,10)	(4,3,6)	(3,3,7)

[a] The values in parentheses are (wins, ties, losses), respectively.

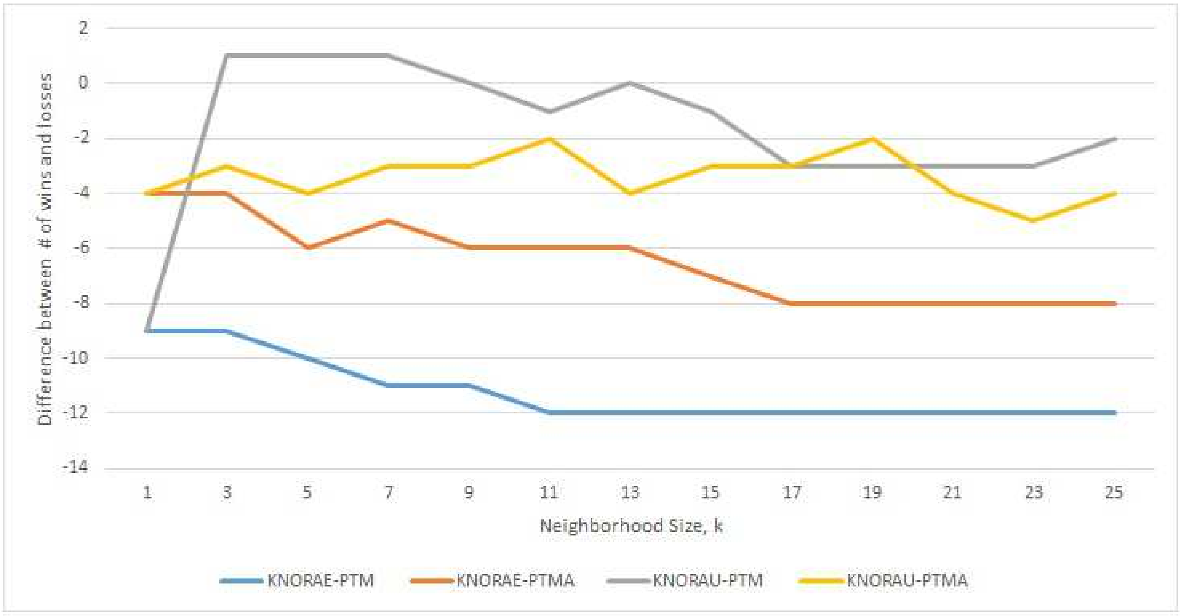


Figure 4.4: Comparison of the static step with KNORA methods

Table 4.10: Summary results of the benchmark methods and static step

Dataset[a,b]	Best Classifier	Best 25	Single SVM	PTM	PTMA
a1a	(75.50, 2.33)	(78.06, 1.05)	(75.40, 0.78)	**(81.11, 2.01)**	(77.54, 1.04)
australian	(75.93, 6.64)	(84.95, 3.03)	(82.07, 2.94)	(85.54, 2.93)	**(86.06, 2.70)**
breast cancer	(94.80, 2.19)	(96.45, 1.40)	(95.80, 1.40)	(96.54, 1.49)	**(96.82, 1.38)**
diabetes	(65.94, 2.05)	(65.12, 0.38)	(73.95, 2.66)	(74.15, 3.87)	**(75.36, 2.52)**
german	(69.89, 0.60)	(70.01, 0.01)	(70.37, 1.22)	**(72.72, 1.79)**	(71.46, 0.92)
heart	(72.96, 6.76)	(80.85, 5.30)	(73.28, 5.71)	(80.83, 5.56)	**(83.08, 5.12)**
ionosphere	(91.80, 3.63)	(93.27, 3.03)	(92.19, 3.09)	(93.74, 2.79)	**(94.04, 2.49)**
liver disorder	(63.03, 6.25)	(64.22, 3.86)	**(71.13, 4.71)**	(69.14, 5.11)	(70.45, 4.09)
mushrooms	(99.76, 0.62)	99.97, 0.51)	(99.99, 0.01)	**(100, 0.0)**	**(100, 0.0)**
rcv	(95.17, 1.61)	(97.03, 0.26)	**(97.17, 0.25)**	(96.93, 0.38)	(97.04, 0.26)
sonar	(72.28, 7.73)	(79.54, 6.81)	(73.85, 6.26)	**(82.87, 6.65)**	(77.42, 6.50)
splice	(55.37, 3.61)	(56.68, 1.44)	(56.19, 1.22)	**(77.55, 4.84)**	(53.50, 0.96)
w1a	(97.10, 0.18)	(97.11, 0.12)	(97.16, 0.25)	**(97.17, 0.41)**	(97.13, 0.15)

[a] The values in parentheses are the mean and standard deviation of accuracy.
[b] For each dataset, the entries in bold represents the method that achieved maximum mean accuracy.

marks considered here are:

- Use of the best classifier from set $\mathcal{C}$,
- Use of the 25 best-performing classifiers from set $\mathcal{C}$ with majority voting, and
- Use of a single SVM classifier trained on all of the training data.

Table 4.10 compares the performance of our proposed measures against these benchmarks in terms of average accuracy. The static step with PTM or PTMA achieves the maximum mean accuracy in all but 2 datasets, while lowering the sandard deviation as well.

Furthermore, Table 4.11 carries out the pairwise t-tests to evaluate the performance of the static step and common benchmarks. Both PTM and PTMA win at least 11 datasets against *Best Classifier*, and 7 datasets against *Best 25* and *Single SVM*. Therefore, we conclude that even PTM, a basic dissimilarity measure in the probability space, performs better than any of these benchmarks.

Table 4.11: Pairwise t-test: static step vs. benchmark methods

	Best Classifier	Best 25	Single SVM
PTM	(12,1,0)	(7,5,1)	(8,3,2)
PTMA	(11,1,1)	(7,3,3)	(8,3,2)

Table 4.12: Pairwise t-test: dynamic vs. static steps across reduced space size (E)

	PTM-D vs.		*PTMA-D vs.*	
E	PTM-S	PTMA-S	PTM-S	PTMA-S
10	(0,7,6)	(3,3,7)	(5,4,4)	(4,6,3)
15	(0,8,5)	(3,3,7)	(5,4,4)	(5,7,1)
20	(0,9,4)	(3,4,6)	(6,3,4)	(5,8,0)
25	(1,8,4)	(3,4,6)	(6,3,4)	(5,8,0)
30	(1,8,4)	(3,3,7)	(7,2,4)	(5,8,0)
35	(2,7,4)	(4,3,6)	(6,3,4)	(5,8,0)
40	(0,9,4)	(4,4,5)	(7,2,4)	(5,8,0)
45	(0,10,3)	(4,3,6)	(6,3,4)	(5,8,0)
50	(1,9,3)	(4,5,4)	(5,4,4)	(4,9,0)

4.7 Results: Dynamic Step

4.7.1 Evaluating Reduced Space Size

It has been shown that the marginal improvement in ensemble performance diminishes for sizes beyond 25 [3]. In this section, we test this assumption by investigating the effects of the reduced space size on the performance of our framework. We carry out experiments for various reduced space sizes E, in increments of 5 over the range of 10 to 50. The results are tabulated in Table 4.12. The choice of the reduced space size overall has minimal effect on the performance of the framework. Therefore, for the experiments in Sections 4.7.2–4.7.4, the size of the reduced space is fixed at 25.

4.7.2 Dynamic Step vs. Static Step

Even though our analysis in Section 4.2 indicates that the running time of our system is dominated by the static step, performing the tasks in the dynamic step has additional time cost. This experiment is performed to evaluate the gain achieved by having the dynamic step as part of our framework.

Tables 4.13 compares the results from the static and dynamic steps for both the PTM and PTMA distance measures. We report, in parentheses, mean and standard deviation of accuracy from 100 runs, respectively. The static step with the ED measure fails to achieve the maximum mean accuracy in all 13 datasets. Meanwhile, the static model with the PTM and PTMA measures (PTM-S and PTMA-S) performs no worse than the dynamic model (PTM-D and PTMA-D): PTM-S and PTMA-S wins exclusively 7 datasets and ties for the maximum mean accuracy for another 2.

Table 4.14 provides information on the pairwise comparisons. PTM-D and PTMA-D consistently outperform the static step for the ED measure. When we check the comparison between PTM-S and PTM-D, we find that the PTM measure works better with the static step: PTM-S wins 4 datasets and ties for another 8. Overall, PTM-D performs significantly worse than PTM-S and PTMA-S.

In contrast, PTMA-D dominates PTMA-S, winning 5 datasets and losing none. PTMA-D also wins more against PTM-S (6 wins to 4 losses). Therefore, PTMA-D performs at least as well as PTM-S and PTMA-S.

When the classifiers for the reduced space are chosen, the aim is to find expert classifiers on the neighbor instances. The results from this experiment imply that expert classifiers

Table 4.13: Summary results of the static and dynamic steps

Dataset[a,b]	ED	PTM-S	PTMA-S	PTM-D	PTMA-D
a1a	(78.92, 1.28)	**(81.11, 2.01)**	(77.54, 1.04)	(79.37, 2.08)	(79.69, 1.63)
australian	(85.76, 3.03)	(85.54, 2.93)	**(86.06, 2.7)**	(85.43, 2.79)	(85.79, 2.81)
breast cancer	(96.13, 1.67)	(96.54, 1.49)	**(96.82, 1.38)**	(96.63, 1.49)	**(96.82, 1.37)**
diabetes	(72.04, 3.26)	(74.15, 3.87)	(75.36, 2.5)	(74.39, 3.57)	**(75.51, 2.58)**
german	(71.16, 2.07)	**(72.72, 1.79)**	(71.46, 0.92)	(71.65, 1.87)	(71.89, 1.26)
heart	(80.68, 5.36)	(80.83, 5.56)	**(83.08, 5.12)**	(80.33, 5.45)	(82.95, 4.92)
ionosphere	(80.71, 5.93)	(93.74, 2.79)	(94.04, 2.49)	**(94.13, 2.64)**	**(94.13, 2.47)**
liver disorder	(58.09, 5.71)	(69.14, 5.11)	(70.45, 4.09)	(68.62, 5.71)	**(70.49, 4.52)**
mushrooms	(98.56, 0.05)	**(100, 0)**	**(100,0)**	**(100, 0)**	**(100, 0)**
rcv	(90.54, 0.51)	(96.93, 0.38)	**(97.04, 0.26)**	(96.93, 0.37)	(97.03, 0.26)
sonar	(72.15, 7.7)	**(82.87, 6.65)**	(77.42, 6.5)	(82.68, 5.82)	(80.77, 6.15)
splice	(65.76, 3.82)	**(77.55, 4.84)**	(53.5, 0.96)	(75.85, 4.06)	(55.22, 1.2)
w1a	(97.19, 0.43)	(97.17, 0.41)	(97.13, 0.15)	(96.79, 0.58)	**(97.19, 0.24)**

[a] The values in parentheses are the mean and standard deviation of accuracy.
[b] For each dataset, the entries in bold represents the method that achieved maximum mean accuracy.

Table 4.14: Pairwise t-test: dynamic vs. static steps

	ED	PTM-S	PTMA-S
PTM-D	(8,4,1)	(1,8,4)	(3,4,6)
PTMA-D	(10,2,1)	(6,3,4)	(5,8,0)

can be identified by considering both the confidence and accuracy of the classifiers on the neighbors, which explains the performance difference between PTMA-D and PTM-D.

4.7.3 Dynamic Step vs. Common Benchmarks

We also compared the performance obtained with our framework against the common benchmarks presented in Section 4.6.4. In addition, we used “Modified Best Improvement” algorithm to find the best ensemble of size E over the validation set as an ensemble's performance on the validation set has been seen as an indicator of generalization. Since it may be impractical to evaluate all $\binom{N}{E}$ possible ensembles, as required for the traditional

Table 4.15: Summary results of benchmark methods and dynamic step

Dataset[a,b]	Best Classifier	Best 25	Single SVM	Best Ens	PTM-D	PTMA-D
a1a	(75.50, 2.33)	(78.06, 1.05)	(75.40, 0.78)	(77.11, 1.10)	(79.37, 2.08)	**(79.69, 1.63)**
australian	(75.93, 6.64)	(84.95, 3.03)	(82.07, 2.94)	(85.33, 2.87)	(85.43, 2.79)	**(85.79, 2.81)**
breast cancer	(94.80, 2.19)	(96.45, 1.40)	(95.80, 1.40)	(96.50, 1.47)	(96.63, 1.49)	**(96.82, 1.37)**
diabetes	(65.94, 2.05)	(65.12, 0.38)	(73.95, 2.66)	(75.01, 2.90)	(74.39, 3.57)	**(75.51, 2.58)**
german	(69.89, 0.60)	(70.01, 0.01)	(70.37, 1.22)	(71.31, 1.15)	(71.65, 1.87)	**(71.89, 1.26)**
heart	(72.96, 6.76)	(80.85, 5.30)	(73.28, 5.71)	(80.89, 5.21)	(80.33, 5.45)	**(82.95, 4.92)**
ionosphere	(91.80, 3.63)	(93.27, 3.03)	(92.19, 3.09)	(93.21, 2.84)	**(94.13, 2.64)**	**(94.13, 2.47)**
liver disorder	(63.03, 6.25)	(64.22, 3.86)	**(71.13, 4.71)**	(68.57, 4.28)	(68.62, 5.71)	(70.49, 4.52)
mushrooms	(99.76, 0.62)	(99.97, 0.51)	(99.99, 0.01)	(99.99,0.01)	**(100, 0.0)**	**(100, 0.0)**
rcv	(95.17, 1.61)	(97.03, 0.26)	**(97.17, 0.25)**	(97.05, 0.24)	(96.93, 0.37)	(97.03, 0.26)
sonar	(72.28, 7.73)	(79.54, 6.81)	(73.85, 6.26)	(80.71, 6.50)	**(82.68, 5.82)**	(80.77, 6.15)
splice	(55.37, 3.61)	(56.68, 1.44)	(56.185, 1.22)	(56.83, 1.98)	**(75.85, 4.06)**	(55.22, 1.22)
w1a	(97.10, 0.18)	(97.11, 0.12)	(97.16, 0.25)	(97.13, 0.16)	(96.79, 0.58)	**(97.19, 0.24)**

[a] The values in parentheses are the mean and standard deviation of accuracy.

[b] For each dataset, the entries in bold represents the method that achieved maximum mean accuracy.

"Best Improvement" algorithm, a modified form of the algorithm is employed: given one of the E classifier slots in the ensemble, each of the $N - E$ unused classifiers is, in turn, substituted into that slot, and the accuracy of the ensemble is re-evaluated. The classifier that provides the best ensemble performance is permanently assigned to the slot, and the displaced classifier is returned to the pool of unused classifiers. This process is then repeated for the next classifier slot in the ensemble until all slots have been processed. We perform an experiment in which E is set to 25 to find the best ensemble on the validation set for a given data set. Tables 4.15 and 4.16 compare the performance of our proposed measures against these benchmarks in terms of average accuracy and pairwise t-test results, respectively. From these tables, we conclude that PTMA outperforms all of the benchmarks. Even PTM, a basic distance measure in the probability space, performs better than any of these benchmarks.

Table 4.16: Pairwise t-test: dynamic step vs. benchmark methods

	Best Classifier	Best 25	Single SVM	Best Ens
PTM-D	(12,0,1)	(7,4,2)	(8,2,3)	(4,7,2)
PTMA-D	(12,1,0)	(9,3,1)	(8,3,2)	(9,3,1)

4.7.4 Evaluating Different Strategies for Dynamic Step

As described in Section 4.1, our proposed framework takes into account the class prediction probabilities assigned to the k-nearest neighbors found in the static step to choose the classifiers whose outputs will be used for generating the reduced probability space for the dynamic step. In this section, we investigate different strategies for choosing a subset of classifiers from the original pool to be used in the dynamic step. In addition to the selection strategy employed earlier, five alternate strategies are implemented:

- **Local Accuracy (LA):** Classifiers which correctly predict at least one of the k neighbors are added to $\mathcal{C}'$. If $|\mathcal{C}'| > 25$, then only 25 top-performing classifiers are considered.
- **Local Accuracy with Accuracy in Validation Set (LAA):** This strategy is similar to LA; however, if $|\mathcal{C}'| < 25$, additional classifiers are selected to reach a size of 25. These classifiers are chosen from the best-performing classifiers on the validation instances that are not already in $\mathcal{C}'$.
- **Conditional Local Accuracy (CLA):** According to this strategy, only the classifiers with at least 50% accuracy in the neighborhood are considered. If there is no such classifier found, then the label for a new instance is assigned randomly. If there are more than 25 classifiers, then 25 most accurate classifiers in the region are considered.

Table 4.17: Pairwise t-test: dynamic vs. static steps under different strategies

	PTM-D vs.		*PTMA-D vs.*	
Strategy	PTM-S	PTMA-S	PTM-S	PTMA-S
LA	(1,7,5)	(4,4,5)	(4,5,4)	(2,6,5)
LAA	(0,6,7)	(4,2,7)	(2,6,5)	(1,7,5)
CLA	(0,6,7)	(4,2,7)	(3,6,4)	(1,7,5)
LACl	(0,6,7)	(4,2,7)	(2,6,5)	(1,7,5)
MinDist	(0,6,7)	(4,2,7)	(3,5,5)	(1,7,5)

- **Local Accuracy with Closeness (LACl):** If the total number of classifiers that correctly predict at least one of the k neighbors is less than 25, the classifiers that are closest to making the correct decision on the neighbors are also added. Closeness is defined as the absolute difference between the probability estimate returned by the classifier for the correct class label and the 50% decision boundary.
- **Minimum Distance (MinDist):** This strategy chooses classifiers that minimize the average distance between a test instance and its neighbors.

Table 4.17 lists the pairwise t-test results for each of these classifier selection strategies against the static step of our framework. None of these strategies yield improved accuracy over the static step. In contrast, the selection strategy discussed in Section 4.7.2 improves performance over the static step when the PTMA measure is used. Intuitively, this can be due to the complementary natures of the classifier selection method, which favors *high* levels of confidence, and the similarity measure, which favors *similar* levels of confidence. Furthermore, while the PTM and PTMA measures achieve improved performance by considering instance classification as a continuous range of probabilities rather than as discrete labels–thereby being able to recognize the similarity of two instances close to but on opposite sides of the decision boundary–the area near the decision boundary still reflects uncertainty regarding the nature of the new instance. Therefore, especially refining the static step with the PTMA measure to favor high classification confidence on the neighbors while still considering the full range of class label probabilities, as is done in the dynamic step, further improves the accuracy of our framework. However, none of the strategies proposed in this section removes the uncertainty associated with the decision boundary, and so they do not result in improved performance.

4.7.5 Dynamic Step vs. Classifiers in the Reduced Space

The reduced space technique of our framework can be considered a dynamic ensemble selection method. Instead of performing k-NN in this reduced space, those classifiers can be used to form an ensemble to make predictions on the new instance. We performed this experiment with several ensemble and neighborhood size values, (E and k) respectively. As mentioned in Section 4.7.4, we employed different strategies to choose the classifiers to form the reduced space, $\mathcal{C}'$. For the experiments in this section, we consider three strategies: (1) Maximum Confidence, (2) Local Accuracy, and (3) Minimum Distance. Figures 4.5–4.7 are plotted based on the difference between the number of wins and losses of dynamic step of our framework against dynamic ensemble selection strategy for a given (k, E) pair.

Figure 4.5 summarizes Table 4.18, which illustrates the comparison results between dynamic framework (with PTM/PTMA) against classifiers chosen to form the reduced space per their confidence on the k neighbors chosen in the static step. The top plot in Figure 4.5 shows that for the PTM measure, if $k > 7$ and $E > 11$ our dynamic framework performs at least as well as using the E classifiers that form the reduced space. According to the bottom plot in Figure 4.5, the dynamic framework is still better than the ensemble built by using the classifiers that are chosen to form the reduced space based on their high confidence on the neighbors of a test instance. Dynamic framework with PTMA does not dominate the ensemble as strongly as PTM does, but it almost always performs at least as well as the ensemble.

The local accuracy based classifier selection methods that are used for reducing the space in the dynamic step of our framework performed similarly. We present the results for the LA method in Figure 4.6 and Table 4.19. According to the top plot, PTM-D performs worse or similar to dynamic ensemble generated based on local accuracy. However, PTMA-D performs especially better for smaller k values; see the bottom plot in Figure 4.6. Changing the ensemble size does not have any effect in the PTMA-D performance.

According to Figure 4.7 and Table 4.20, PTM-D and PTMA-D clearly dominate the dynamic ensembles formed by the classifiers that minimize the average distance between a test instance and its neighbors in the static step.

4.8 Conclusion

This work proposes a framework for dynamic class prediction using two dissimilarity measures, PTM and PTMA, defined based on the probability estimates returned for a particular class by classifiers for data instances. These measures are compared to three baseline dis/similarity measures. We conclude that the two proposed measures outperform the baseline measures. Our experiments also reveal that using classifiers' outputs is better than using the original feature space to find similar instances.

In addition, we compare the results from the static and dynamic steps of our framework with some of the standard classifier and ensemble selection methods. We conclude that performing k-NN in the probability-based classifier space is better than using the best classifier, 25 top performing classifiers, or an SVM classifier trained using the entire training dataset. The performance of the dynamic framework against KNORA and against dynamic ensemble selection methods show that the dynamic step of our framework is important to improve performance.

Table 4.18: Comparison of the dynamic step and *maximum confidence* ensemble across (k, E) pairs

PTM-D[a,b]

	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(1,1,11)	(2,0,11)	(1,1,11)	(1,1,11)	(1,1,11)	(1,2,10)	(1,2,10)	(1,2,10)	(1,1,11)	(1,2,10)	(1,2,10)	(1,2,10)	(1,2,10)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)	(1,3,9)
3	(0,2,11)	(1,2,10)	(1,5,7)	(1,4,8)	(1,4,8)	(1,5,7)	(1,6,6)	(2,5,6)	(1,6,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,6,5)	(3,4,6)	(3,5,5)	(3,5,5)	(3,5,5)	(2,6,5)	(2,6,5)	(2,6,5)	(3,5,5)	(4,4,5)
5	(0,1,12)	(1,4,8)	(1,5,7)	(1,6,6)	(2,5,6)	(3,5,5)	(1,7,5)	(3,5,5)	(2,6,5)	(3,5,5)	(3,6,4)	(3,6,4)	(3,6,4)	(4,6,3)	(3,6,4)	(4,5,4)	(5,4,4)	(5,4,4)	(7,2,4)	(6,3,4)	(6,3,4)	(5,4,4)	(4,5,4)	(5,4,4)	(4,5,4)
7	(0,1,12)	(1,4,8)	(1,8,4)	(1,8,4)	(2,7,4)	(3,6,4)	(3,8,2)	(2,10,1)	(3,8,2)	(3,8,2)	(3,8,2)	(3,7,3)	(4,8,1)	(5,5,3)	(5,6,2)	(5,6,2)	(5,6,2)	(6,5,2)	(5,5,3)	(6,4,3)	(6,5,2)	(6,5,2)	(7,3,3)	(6,3,4)	(6,4,3)
9	(0,2,11)	(1,4,8)	(1,6,6)	(2,6,5)	(2,7,4)	(2,9,2)	(2,9,2)	(2,10,1)	(3,7,3)	(4,7,2)	(3,8,2)	(5,6,2)	(5,6,2)	(5,7,1)	(6,6,1)	(6,6,1)	(5,6,2)	(6,5,2)	(5,7,1)	(5,6,2)	(4,8,1)	(4,8,1)	(5,7,1)	(6,6,1)	(6,6,1)
11	(0,4,9)	(2,5,6)	(1,7,5)	(2,8,3)	(2,8,3)	(2,9,2)	(4,6,3)	(5,6,2)	(4,5,4)	(3,9,1)	(3,9,1)	(4,8,1)	(5,7,1)	(5,7,1)	(5,8,0)	(5,7,1)	(5,8,0)	(4,8,1)	(6,7,0)	(7,6,0)	(7,4,2)	(7,6,0)	(6,7,0)	(6,6,1)	(5,7,1)
13	(0,5,8)	(1,7,5)	(3,6,4)	(2,7,4)	(2,10,1)	(3,9,1)	(2,10,1)	(4,8,1)	(4,7,2)	(5,7,1)	(4,7,2)	(5,7,1)	(5,7,1)	(5,7,1)	(4,8,1)	(4,9,0)	(4,8,1)	(5,7,1)	(5,8,0)	(5,8,0)	(5,8,0)	(6,7,0)	(5,8,0)	(4,9,0)	(4,8,1)
15	(0,4,9)	(2,6,5)	(3,7,3)	(2,8,3)	(2,10,1)	(2,10,1)	(3,9,1)	(4,8,1)	(3,9,1)	(4,7,2)	(6,6,1)	(6,6,1)	(7,5,1)	(5,6,2)	(3,9,1)	(5,7,1)	(5,8,0)	(5,8,0)	(4,9,0)	(5,8,0)	(6,7,0)	(7,6,0)	(6,7,0)	(6,7,0)	(5,8,0)
17	(0,5,8)	(2,7,4)	(3,6,4)	(2,10,1)	(2,10,1)	(3,9,1)	(3,9,1)	(3,9,1)	(4,8,1)	(4,8,1)	(4,8,1)	(6,5,2)	(5,6,2)	(4,7,2)	(4,8,1)	(6,6,1)	(6,7,0)	(5,8,0)	(5,8,0)	(5,7,1)	(5,8,0)	(5,8,0)	(5,8,0)	(6,7,0)	(6,7,0)
19	(0,5,8)	(1,9,3)	(2,8,3)	(2,9,2)	(2,10,1)	(3,9,1)	(2,10,1)	(3,9,1)	(5,7,1)	(5,7,1)	(4,8,1)	(6,6,1)	(6,5,2)	(5,7,1)	(4,8,1)	(5,7,1)	(4,9,0)	(5,8,0)	(5,8,0)	(4,9,0)	(6,6,1)	(5,6,2)	(4,9,0)	(5,7,1)	(4,9,0)
21	(0,4,9)	(2,7,4)	(2,8,3)	(3,8,2)	(2,9,2)	(2,10,1)	(4,8,1)	(5,7,1)	(4,8,1)	(6,6,1)	(5,6,2)	(6,6,1)	(3,8,2)	(3,9,1)	(4,8,1)	(4,9,0)	(5,8,0)	(5,7,1)	(3,10,0)	(4,9,0)	(4,9,0)	(4,9,0)	(3,10,0)	(5,7,1)	(6,6,1)
23	(0,6,7)	(2,8,3)	(2,9,2)	(2,10,1)	(3,9,1)	(4,8,1)	(3,9,1)	(3,8,2)	(4,7,2)	(6,6,1)	(4,8,1)	(4,7,2)	(4,7,2)	(5,7,1)	(4,9,0)	(4,7,2)	(4,9,0)	(3,10,0)	(6,7,0)	(5,8,0)	(4,9,0)	(4,9,0)	(3,10,0)	(4,9,0)	(3,10,0)
25	(0,6,7)	(2,8,3)	(2,7,4)	(2,9,2)	(3,8,2)	(3,8,2)	(2,10,1)	(3,9,1)	(3,10,0)	(3,9,1)	(4,9,0)	(4,7,2)	(6,6,1)	(5,7,1)	(5,7,1)	(4,8,1)	(4,9,0)	(4,9,0)	(5,8,0)	(4,9,0)	(5,7,1)	(4,9,0)	(4,9,0)	(4,9,0)	(4,9,0)

PTMA-D[a,b]

	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(4,8,1)	(3,9,1)	(5,8,0)	(4,9,0)	(5,8,0)	(2,11,0)	(3,9,1)	(4,8,1)	(4,8,1)	(3,9,1)	(4,8,1)	(6,5,2)	(5,6,2)	(4,7,2)	(5,7,1)	(4,7,2)	(6,5,2)	(4,7,2)	(6,5,2)	(4,6,3)	(4,7,2)	(4,6,3)	(4,7,2)	(5,6,2)	(5,6,2)
3	(2,10,1)	(4,9,0)	(6,6,1)	(2,11,0)	(3,9,1)	(2,10,1)	(4,8,1)	(4,8,1)	(4,7,2)	(2,9,2)	(2,9,2)	(2,10,1)	(2,9,2)	(2,10,1)	(3,9,1)	(3,9,1)	(4,7,2)	(3,8,2)	(3,8,2)	(4,7,2)	(5,6,2)	(4,7,2)	(4,7,2)	(5,6,2)	(5,6,2)
5	(1,12,0)	(3,10,0)	(1,11,1)	(2,9,2)	(3,8,2)	(3,9,1)	(4,7,2)	(3,7,3)	(2,8,3)	(1,10,2)	(1,10,2)	(2,10,1)	(2,9,2)	(3,8,2)	(4,7,2)	(2,8,3)	(4,5,4)	(4,7,2)	(2,9,2)	(6,4,3)	(5,5,3)	(5,6,2)	(5,6,2)	(6,5,2)	(5,6,2)
7	(2,9,2)	(3,10,0)	(1,11,1)	(2,10,1)	(2,10,1)	(2,9,2)	(4,8,1)	(2,10,1)	(3,9,1)	(4,7,2)	(5,6,2)	(1,10,2)	(4,7,2)	(5,7,1)	(5,7,1)	(4,8,1)	(2,10,1)	(3,8,2)	(3,9,1)	(5,6,2)	(5,7,1)	(4,7,2)	(4,7,2)	(5,6,2)	(4,6,3)
9	(0,11,2)	(3,10,0)	(2,10,1)	(3,8,2)	(4,8,1)	(4,8,1)	(2,10,1)	(4,8,1)	(2,10,1)	(3,9,1)	(4,8,1)	(5,6,2)	(4,8,1)	(7,4,2)	(5,5,3)	(2,10,1)	(4,7,2)	(5,7,1)	(5,7,1)	(4,7,2)	(5,6,2)	(5,7,1)	(4,8,1)	(5,5,3)	(5,6,2)
11	(0,11,2)	(3,9,1)	(4,8,1)	(3,9,1)	(3,9,1)	(2,10,1)	(3,9,1)	(3,9,1)	(3,9,1)	(3,9,1)	(4,8,1)	(3,9,1)	(3,9,1)	(5,7,1)	(4,8,1)	(3,9,1)	(5,7,1)	(6,5,2)	(4,8,1)	(5,7,1)	(5,7,1)	(4,7,2)	(5,6,2)	(3,8,2)	(5,6,2)
13	(1,9,3)	(2,10,1)	(1,10,2)	(3,9,1)	(3,9,1)	(2,10,1)	(4,8,1)	(2,10,1)	(2,10,1)	(3,9,1)	(2,9,2)	(4,8,1)	(4,8,1)	(3,9,1)	(3,9,1)	(3,9,1)	(4,8,1)	(5,7,1)	(4,8,1)	(4,8,1)	(4,8,1)	(4,7,2)	(5,7,1)	(4,7,2)	(4,7,2)
15	(1,8,4)	(1,11,1)	(3,8,2)	(3,8,2)	(2,10,1)	(2,10,1)	(3,9,1)	(3,9,1)	(3,9,1)	(4,8,1)	(4,8,1)	(3,9,1)	(5,7,1)	(3,9,1)	(2,10,1)	(3,9,1)	(4,8,1)	(4,8,1)	(4,7,2)	(4,7,2)	(3,9,1)	(3,9,1)	(5,7,1)	(5,6,2)	(4,7,2)
17	(1,8,4)	(1,11,1)	(3,9,1)	(3,9,1)	(3,9,1)	(2,10,1)	(3,9,1)	(4,8,1)	(4,8,1)	(4,8,1)	(4,8,1)	(3,9,1)	(4,8,1)	(3,9,1)	(3,9,1)	(3,9,1)	(3,9,1)	(4,8,1)	(4,8,1)	(4,7,2)	(4,8,1)	(4,7,2)	(4,7,2)	(3,9,1)	(4,7,2)
19	(1,9,3)	(1,11,1)	(1,11,1)	(1,11,1)	(1,11,1)	(1,11,1)	(1,11,1)	(3,9,1)	(5,7,1)	(3,8,2)	(3,8,2)	(4,8,1)	(3,9,1)	(5,7,1)	(4,8,1)	(4,8,1)	(2,10,1)	(2,10,1)	(2,10,1)	(3,9,1)	(2,10,1)	(1,10,2)	(2,9,2)	(2,9,2)	(2,9,2)
21	(0,8,5)	(1,11,1)	(3,9,1)	(4,8,1)	(2,10,1)	(3,9,1)	(3,8,2)	(4,8,1)	(4,8,1)	(4,8,1)	(3,9,1)	(3,9,1)	(2,10,1)	(2,10,1)	(3,9,1)	(3,9,1)	(2,10,1)	(4,8,1)	(2,10,1)	(2,10,1)	(2,9,2)	(3,9,1)	(1,11,1)	(2,8,3)	(3,8,2)
23	(1,8,4)	(1,11,1)	(1,11,1)	(2,9,2)	(2,10,1)	(3,9,1)	(2,10,1)	(2,10,1)	(3,9,1)	(2,9,2)	(2,10,1)	(3,9,1)	(4,8,1)	(3,9,1)	(3,9,1)	(3,9,1)	(3,8,2)	(3,9,1)	(3,9,1)	(2,10,1)	(3,9,1)	(2,9,2)	(2,8,3)	(2,8,3)	(1,10,2)
25	(1,9,3)	(1,11,1)	(2,10,1)	(2,10,1)	(2,10,1)	(2,10,1)	(3,9,1)	(2,10,1)	(4,8,1)	(4,8,1)	(3,9,1)	(3,8,2)	(2,10,1)	(3,9,1)	(4,7,2)	(4,8,1)	(4,8,1)	(3,9,1)	(4,8,1)	(4,8,1)	(5,6,2)	(2,9,2)	(2,8,3)	(3,8,2)	(3,8,2)

[a] The rows (i.e. $1, 3, \ldots, 25$) indicate the k value.

[b] The columns (i.e. $1, 3, \ldots, 49$) indicate the E value.

Table 4.19: Comparison of the dynamic step with *local accuracy* ensemble across (k, E) pairs

PTM-D[a,b]

	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(0,2,11)	(0,1,12)	(0,1,12)	(0,1,12)	(0,1,12)	(0,1,12)	(0,1,12)	(0,3,10)	(0,3,10)	(0,2,11)	(0,2,11)	(0,2,11)	(0,3,10)	(0,2,11)	(0,2,11)	(0,2,11)	(0,2,11)	(0,2,11)	(0,2,11)	(0,3,10)	(0,2,11)	(0,3,10)	(0,4,9)	(0,3,10)	(0,3,10)
3	(2,4,7)	(3,4,6)	(4,3,6)	(3,3,7)	(3,4,6)	(2,4,7)	(2,4,7)	(2,4,7)	(2,4,7)	(2,4,7)	(2,4,7)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,4,7)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,4,7)	(2,4,7)	(2,4,7)	(2,4,7)	(2,5,6)
5	(3,5,5)	(4,3,6)	(4,3,6)	(3,4,6)	(3,4,6)	(3,4,6)	(3,4,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,4,5)	(4,3,6)	(4,3,6)	(4,3,6)	(3,3,7)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,4,5)	(3,5,5)	(3,4,6)	(2,6,5)	(3,4,6)	(2,6,5)
7	(4,4,5)	(5,2,6)	(4,4,5)	(4,3,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(3,4,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,4,5)	(5,3,5)	(5,5,3)	(4,4,5)	(3,6,4)	(3,4,6)	(5,3,5)	(3,5,5)	(4,4,5)
9	(4,4,5)	(5,3,5)	(4,4,5)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(5,3,5)	(5,3,5)	(5,3,5)	(4,4,5)	(4,4,5)	(4,4,5)	(3,4,6)	(4,3,6)	(3,5,5)	(5,3,5)	(5,3,5)	(4,4,5)	(4,4,5)
11	(5,3,5)	(6,2,5)	(4,4,5)	(4,3,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(3,4,6)	(4,3,6)	(4,4,5)	(4,4,5)	(4,4,5)	(5,3,5)	(4,4,5)	(4,3,6)	(5,3,5)	(4,4,5)	(4,4,5)	(4,4,5)	(4,3,6)	(4,4,5)	(4,3,6)	(4,3,6)	(4,3,6)
13	(7,1,5)	(5,3,5)	(5,3,5)	(5,2,6)	(5,2,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(4,3,6)	(5,2,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,4,5)	(4,3,6)	(4,3,6)	(3,5,5)	(4,4,5)	(3,5,5)	(3,5,5)	(3,5,5)	(2,6,5)	(3,5,5)
15	(6,2,5)	(5,3,5)	(5,3,5)	(4,3,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(5,2,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(4,4,5)	(4,4,5)	(4,4,5)	(4,5,4)	(4,6,3)	(3,7,3)	(2,8,3)	(3,7,3)	(3,7,3)
17	(7,2,4)	(4,4,5)	(5,2,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(4,3,6)	(4,3,6)	(5,2,6)	(3,4,6)	(4,3,6)	(4,3,6)	(4,4,5)	(3,5,5)	(3,5,5)	(3,5,5)	(3,5,5)	(2,7,4)	(2,6,5)	(2,7,4)	(3,7,3)	(3,7,3)	(2,9,2)	(2,8,3)
19	(7,2,4)	(3,5,5)	(4,3,6)	(5,2,6)	(5,2,6)	(5,2,6)	(4,3,6)	(3,4,6)	(4,3,6)	(3,4,6)	(2,5,6)	(1,6,6)	(2,5,6)	(2,5,6)	(2,6,5)	(2,6,5)	(2,6,5)	(3,6,4)	(2,6,5)	(2,6,5)	(2,6,5)	(2,7,4)	(2,6,5)	(2,6,5)	(2,7,4)
21	(6,3,4)	(4,4,5)	(4,3,6)	(4,3,6)	(4,3,6)	(4,3,6)	(3,4,6)	(4,3,6)	(3,4,6)	(3,4,6)	(2,4,7)	(2,6,5)	(2,5,6)	(2,6,5)	(2,6,5)	(2,7,4)	(2,7,4)	(2,7,4)	(2,6,5)	(2,6,5)	(2,5,6)	(2,6,5)	(2,7,4)	(2,8,3)	(2,6,5)
23	(5,4,4)	(4,4,5)	(4,4,5)	(4,3,6)	(4,3,6)	(4,3,6)	(3,4,6)	(2,5,6)	(2,4,7)	(3,4,6)	(2,6,5)	(2,5,6)	(2,5,6)	(2,5,6)	(2,5,6)	(2,6,5)	(2,6,5)	(2,6,5)	(2,6,5)	(2,6,5)	(2,6,5)	(2,5,6)	(2,6,5)	(2,6,5)	(1,7,5)
25	(5,4,4)	(4,4,5)	(4,3,6)	(3,4,6)	(3,4,6)	(4,3,6)	(2,5,6)	(3,3,7)	(3,5,5)	(2,6,5)	(2,6,5)	(2,5,6)	(2,6,5)	(2,5,6)	(2,6,5)	(2,6,5)	(2,5,6)	(2,6,5)	(2,6,5)	(2,7,4)	(3,6,4)	(3,6,4)	(3,6,4)	(3,5,5)	(3,6,4)

PTMA-D[a,b]

	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(7,5,1)	(9,3,1)	(9,4,0)	(8,5,0)	(9,4,0)	(8,4,1)	(9,3,1)	(10,2,1)	(10,2,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(11,1,1)	(11,1,1)	(10,2,1)	(11,1,1)	(10,2,1)	(11,1,1)	(11,1,1)	(11,1,1)	(11,1,1)	(11,1,1)	(11,1,1)
3	(6,7,0)	(9,4,0)	(10,3,0)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)	(6,6,1)	(7,5,1)	(6,6,1)	(6,5,2)	(6,5,2)	(6,5,2)	(8,3,2)	(7,4,2)	(7,4,2)	(8,3,2)	(7,4,2)	(7,4,2)	(7,4,2)	(8,3,2)	(8,3,2)	(7,4,2)	(8,3,2)	(7,4,2)
5	(9,4,0)	(6,6,1)	(6,6,1)	(8,3,2)	(7,4,2)	(6,6,1)	(6,6,1)	(6,6,1)	(7,4,2)	(7,4,2)	(7,5,1)	(7,4,2)	(7,4,2)	(7,4,2)	(7,4,2)	(7,4,2)	(7,4,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(7,4,2)	(8,3,2)	(8,3,2)
7	(8,3,2)	(7,5,1)	(6,5,2)	(6,5,2)	(7,4,2)	(6,6,1)	(5,7,1)	(5,7,1)	(6,6,1)	(6,6,1)	(7,4,2)	(6,5,2)	(6,5,2)	(6,5,2)	(6,5,2)	(6,5,2)	(7,4,2)	(6,5,2)	(6,5,2)	(7,4,2)	(6,5,2)	(6,5,2)	(7,4,2)	(7,4,2)	(7,4,2)
9	(7,4,2)	(6,4,3)	(5,5,3)	(6,4,3)	(5,6,2)	(4,7,2)	(4,7,2)	(4,7,2)	(5,6,2)	(6,5,2)	(6,4,3)	(8,3,2)	(6,5,2)	(7,4,2)	(7,4,2)	(5,5,3)	(7,3,3)	(6,5,2)	(6,5,2)	(6,4,3)	(6,5,2)	(7,4,2)	(6,4,3)	(7,3,3)	(7,3,3)
11	(6,5,2)	(5,5,3)	(5,6,2)	(5,6,2)	(5,6,2)	(4,7,2)	(3,8,2)	(3,8,2)	(4,7,2)	(5,6,2)	(4,7,2)	(6,4,3)	(6,4,3)	(5,5,3)	(6,4,3)	(5,5,3)	(6,4,3)	(7,3,3)	(7,3,3)	(7,3,3)	(7,3,3)	(7,3,3)	(7,3,3)	(6,4,3)	(7,4,2)
13	(6,5,2)	(6,5,2)	(5,6,2)	(4,7,2)	(5,6,2)	(5,6,2)	(5,6,2)	(5,6,2)	(5,6,2)	(5,6,2)	(5,5,3)	(5,5,3)	(5,5,3)	(4,6,3)	(4,6,3)	(5,5,3)	(5,5,3)	(6,4,3)	(6,4,3)	(6,4,3)	(6,4,3)	(5,5,3)	(5,5,3)	(5,5,3)	(5,5,3)
15	(6,5,2)	(5,6,2)	(5,6,2)	(6,5,2)	(4,7,2)	(4,7,2)	(3,8,2)	(5,6,2)	(4,6,3)	(5,5,3)	(5,5,3)	(4,6,3)	(5,5,3)	(5,5,3)	(5,5,3)	(5,5,3)	(5,5,3)	(6,4,3)	(4,6,3)	(6,4,3)	(6,4,3)	(6,4,3)	(5,5,3)	(5,5,3)	(5,5,3)
17	(7,4,2)	(6,4,3)	(6,5,2)	(5,6,2)	(4,7,2)	(4,7,2)	(4,7,2)	(5,5,3)	(5,6,2)	(5,5,3)	(4,6,3)	(5,5,3)	(5,5,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(5,5,3)
19	(7,4,2)	(5,5,3)	(5,6,2)	(5,6,2)	(4,7,2)	(4,7,2)	(4,7,2)	(5,6,2)	(6,5,2)	(5,5,3)	(6,4,3)	(5,5,3)	(5,5,3)	(5,5,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)
21	(5,6,2)	(4,6,3)	(5,6,2)	(4,7,2)	(4,7,2)	(4,7,2)	(4,7,2)	(4,7,2)	(4,7,2)	(6,5,2)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(5,5,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)	(4,6,3)
23	(5,7,1)	(4,7,2)	(5,6,2)	(3,8,2)	(4,7,2)	(4,7,2)	(4,6,3)	(4,6,3)	(4,7,2)	(3,7,3)	(4,6,3)	(3,7,3)	(3,7,3)	(3,7,3)	(2,8,3)	(4,6,3)	(4,6,3)	(4,6,3)	(5,5,3)	(4,6,3)	(3,7,3)	(4,6,3)	(4,6,3)	(4,6,3)	(3,8,2)
25	(5,7,1)	(5,6,2)	(5,6,2)	(2,9,2)	(3,8,2)	(3,7,3)	(3,7,3)	(4,7,2)	(4,7,2)	(3,7,3)	(3,7,3)	(3,7,3)	(3,7,3)	(3,7,3)	(3,7,3)	(3,7,3)	(4,6,3)	(3,7,3)	(3,7,3)	(3,7,3)	(4,6,3)	(3,8,2)	(3,8,2)	(3,8,2)	(3,7,3)

[a] The rows (i.e. $1, 3, \ldots, 25$) indicate the k value.

[b] The columns (i.e. $1, 3, \ldots, 49$) indicate the E value.

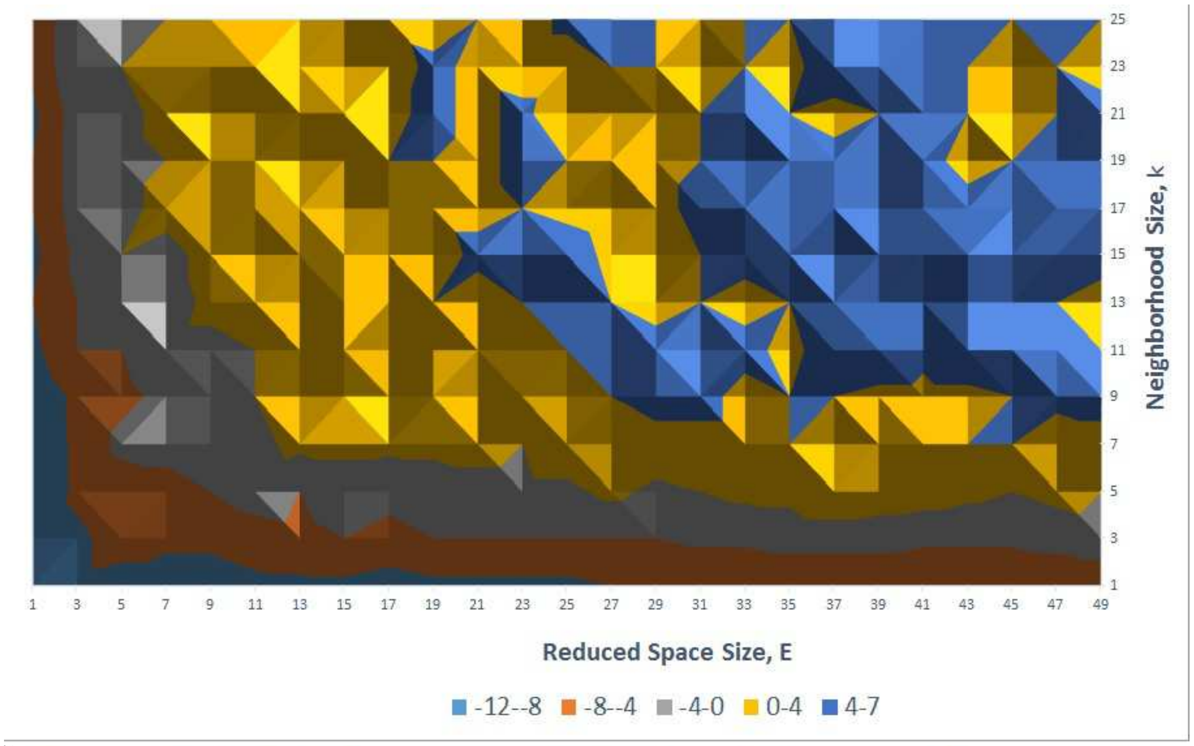
Table 4.20: Comparison of the dynamic step with *minimum distance* ensemble across (k, E) pairs

	PTM-D[a,b]																								
	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(2,3,8)	(5,0,8)	(5,0,8)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(4,2,7)	(3,3,7)	(4,2,7)	(4,2,7)	(3,3,7)
3	(4,2,7)	(6,1,6)	(7,1,5)	(6,5,2)	(6,4,3)	(6,3,4)	(6,6,1)	(6,6,1)	(7,4,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(9,2,2)	(9,2,2)	(9,2,2)	(9,2,2)	(9,2,2)	(9,2,2)	(9,2,2)
5	(5,2,6)	(6,3,4)	(6,6,1)	(9,3,1)	(9,3,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)
7	(5,2,6)	(6,6,1)	(8,4,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,3,0)	(10,3,0)	(10,3,0)	(10,2,1)	(10,2,1)	(10,3,0)	(10,2,1)	(10,2,1)	(11,1,1)
9	(5,2,6)	(7,4,2)	(8,4,1)	(9,3,1)	(10,2,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,3,0)	(10,3,0)	(10,2,1)	(10,3,0)	(10,3,0)	(10,3,0)	(10,3,0)	(10,3,0)	(10,2,1)	(10,2,1)	(10,2,1)
11	(4,4,5)	(7,5,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,3,0)	(10,3,0)	(10,3,0)	(10,3,0)	(10,3,0)	(11,2,0)	(11,2,0)	(10,3,0)	(10,3,0)	(10,3,0)	(9,4,0)	(9,4,0)
13	(5,3,5)	(8,4,1)	(8,4,1)	(10,2,1)	(10,2,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
15	(5,3,5)	(7,5,1)	(8,4,1)	(8,4,1)	(9,3,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
17	(5,3,5)	(7,4,2)	(8,4,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
19	(5,4,4)	(7,5,1)	(7,5,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
21	(6,3,4)	(6,6,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
23	(6,4,3)	(7,5,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,4,0)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)
25	(6,3,4)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(9,3,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,4,0)	(9,3,1)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)	(9,4,0)

	PTMA-D[a,b]																								
	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49
1	(3,1,9)	(3,3,7)	(4,5,4)	(4,5,4)	(4,6,3)	(4,6,3)	(4,6,3)	(5,6,2)	(6,6,1)	(6,7,0)	(5,8,0)	(5,6,2)	(5,6,2)	(6,7,0)	(5,8,0)	(5,7,1)	(5,7,1)	(5,7,1)	(6,6,1)	(7,5,1)	(7,5,1)	(6,6,1)	(6,6,1)	(6,6,1)	(6,6,1)
3	(4,1,8)	(4,6,3)	(5,6,2)	(5,6,2)	(7,4,2)	(8,3,2)	(8,4,1)	(7,5,1)	(7,5,1)	(7,5,1)	(8,4,1)	(7,5,1)	(7,5,1)	(8,4,1)	(8,4,1)	(9,3,1)	(10,2,1)	(9,3,1)	(9,3,1)	(10,2,1)	(10,2,1)	(10,2,1)	(10,2,1)	(9,3,1)	(9,3,1)
5	(3,2,8)	(5,6,2)	(6,5,2)	(8,3,2)	(7,4,2)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)
7	(3,2,8)	(6,5,2)	(6,6,1)	(6,6,1)	(6,6,1)	(7,5,1)	(6,6,1)	(7,5,1)	(8,4,1)	(8,4,1)	(7,5,1)	(9,3,1)	(8,4,1)	(9,3,1)	(8,4,1)	(8,4,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,5,1)	(8,4,1)	(7,5,1)	(7,5,1)	(7,5,1)
9	(3,3,7)	(6,6,1)	(6,6,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)
11	(3,4,6)	(6,7,0)	(8,5,0)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)
13	(3,4,6)	(6,6,1)	(7,6,0)	(8,4,1)	(8,4,1)	(9,3,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,2,2)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)
15	(3,4,6)	(7,4,2)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)
17	(3,5,5)	(6,6,1)	(8,5,0)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,3,2)	(8,3,2)	(8,4,1)	(8,3,2)
19	(3,6,4)	(7,4,2)	(8,4,1)	(8,4,1)	(9,3,1)	(9,3,1)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)
21	(3,6,4)	(7,4,2)	(8,4,1)	(8,4,1)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,4,2)	(7,4,2)	(7,4,2)
23	(3,6,4)	(7,6,0)	(8,5,0)	(8,5,0)	(7,5,1)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(8,4,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,5,1)	(7,5,1)
25	(3,6,4)	(8,3,2)	(8,5,0)	(8,4,1)	(7,5,1)	(8,4,1)	(7,4,2)	(8,3,2)	(7,4,2)	(7,5,1)	(7,5,1)	(7,4,2)	(8,4,1)	(8,4,1)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(8,3,2)	(7,4,2)	(8,3,2)	(8,3,2)	(8,3,2)

[a] The rows (i.e. $1, 3, \ldots, 25$) indicate the k value.

[b] The columns (i.e. $1, 3, \ldots, 49$) indicate the E value.



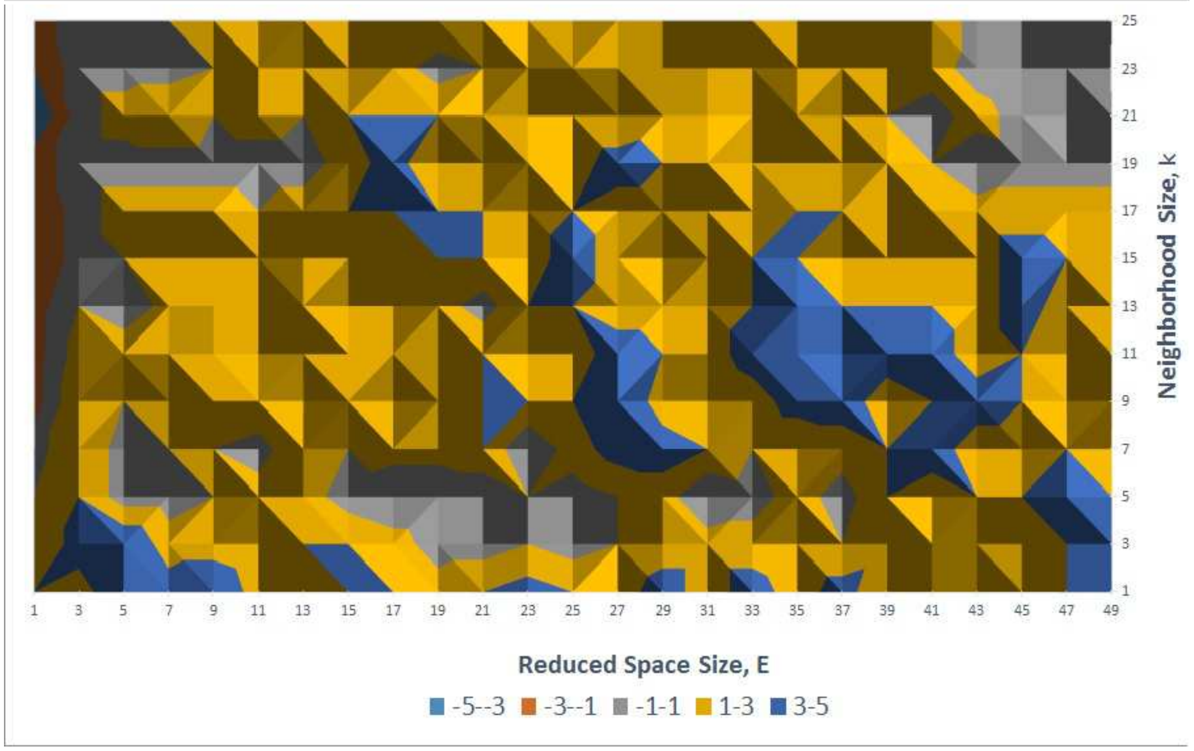


Figure 4.5: Comparison of the dynamic step (PTM-D in the top plot, PTMA-D in the bottom plot) and the *maximum confidence* ensemble

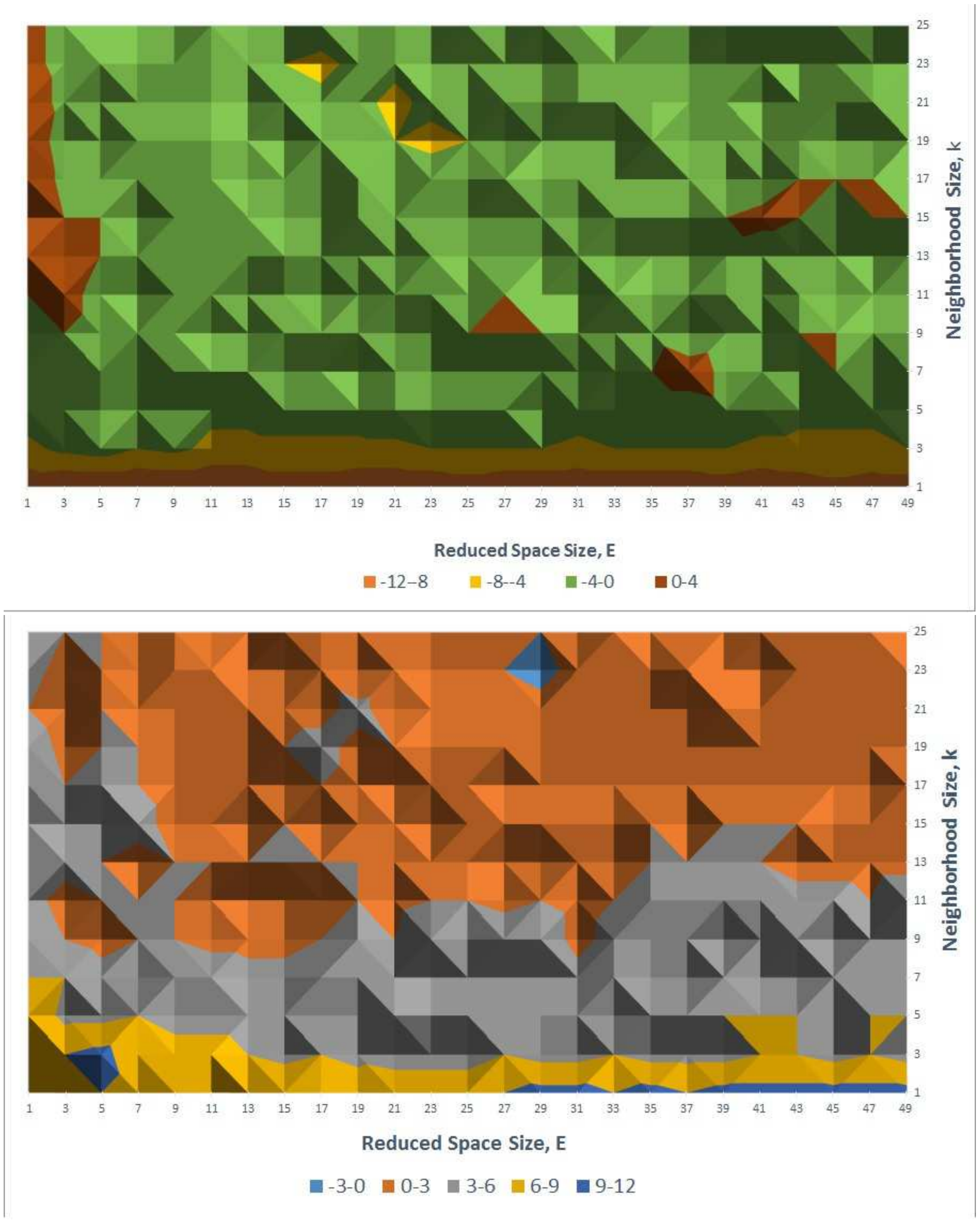
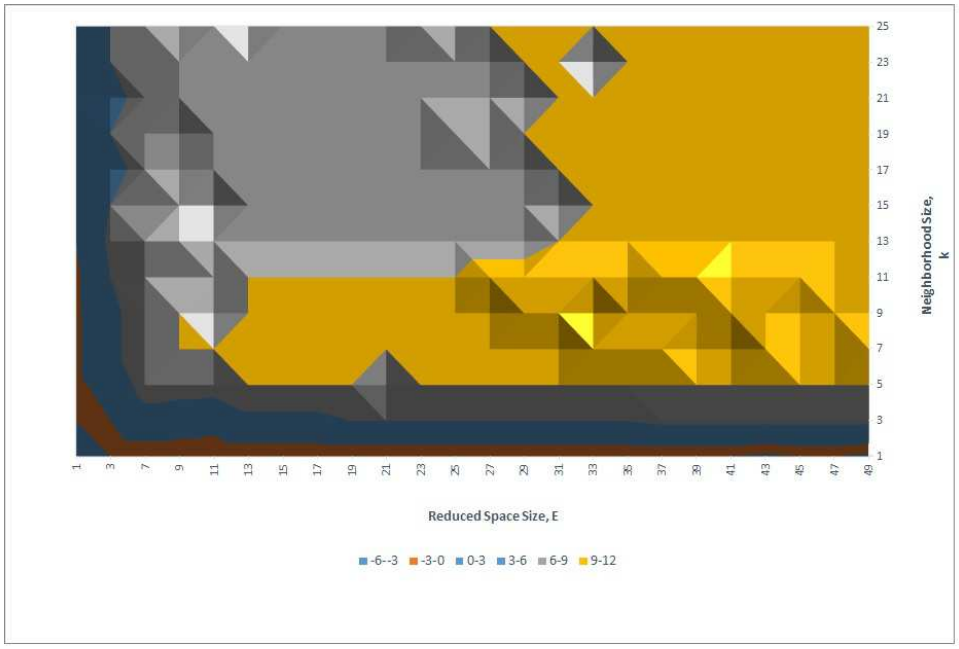


Figure 4.6: Comparison of the dynamic step (PTM-D in the top plot, PTMA-D in the bottom plot) and the *local accuracy* ensemble



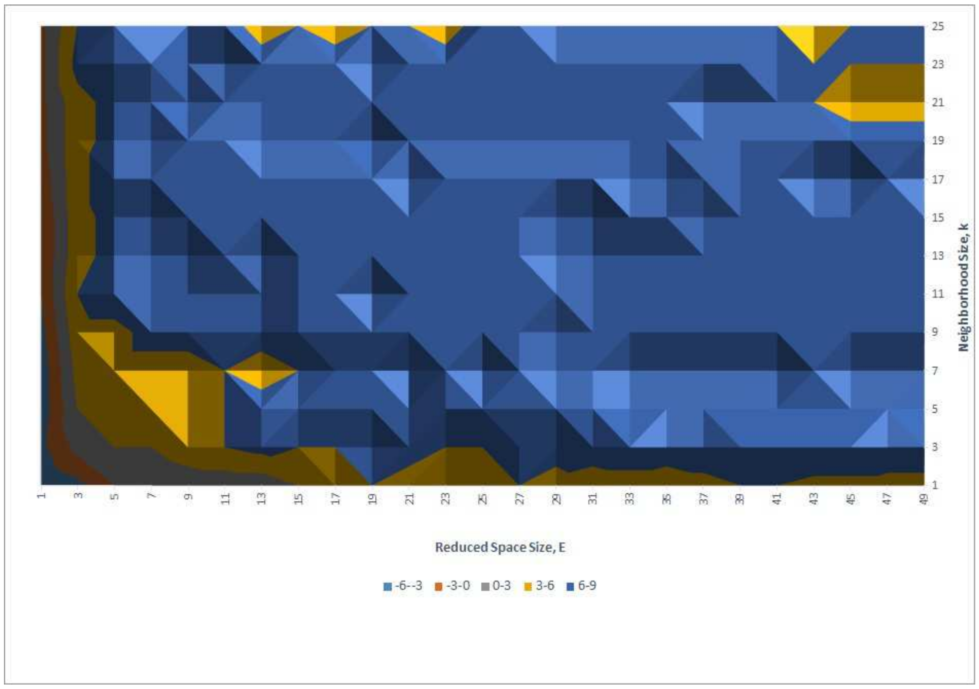


Figure 4.7: Comparison of the dynamic step (PTM-D in the top plot, PTMA-D in the bottom plot) and the *minimum distance* ensemble

CHAPTER 5
EFFECT OF DIVERSITY IN DYNAMIC CLASS PREDICTION

Ensembles of classifiers have been shown to improve the performance of a prediction system compared to a single classifier. The two key factors affecting an ensemble's performance are the accuracy of classifiers and the diversity among them, as no gain can be achieved by having the same classifier in the ensemble multiple times. Earlier studies regarding static ensembles have focused on generating diverse classifiers by changing the training set [3, 24, 4]. These methods do not explicitly maximize the diversity; on the contrary, it is achieved implicitly. Due to the success of these methods, many studies have proposed new diversity measures and have aimed to choose the classifiers with high accuracy and diversity to form the ensemble. These studies have conflicting results, which put the benefits of diversity into question [38, 23].

Margineantu and Dietterich [50] show that there is a trade-off between accuracy and diversity using a Kappa-error plot. Later, Kuncheva et al. [42] and Tang et al. [68] provide an in-depth analysis of some of the diversity measures and show their relationships and shortcomings. These studies conclude that diversity is beneficial only to a certain degree and that maximizing it decreases performance. Hsu and Srivasta [34] analyze the relationship between diversity and correlation among classifiers to provide a proxy for the relationship between diversity and accuracy. The motivation for their approach is that the connection between correlation and accuracy has been well established [70, 4]. A critical value for the disagreement measure (which is a measure of diversity) is formulated in terms of the correlation among classifiers in the ensemble, and it is shown that beyond this critical value, the system performance decreases. In [6], they argue that methods implicitly integrating diversity should be used to form the ensembles.

The perspective on diversity has changed with the introduction of dynamic class prediction systems. These systems treat each new data instance individually while making a prediction. They dynamically adjust for each new data instance by choosing a classifier(s) from the pool of already-trained classifiers. Most of the studies in the field of dynamic class prediction have focused on finding the neighborhood of the new data instance and choosing the most competent classifier(s) to make predictions. Competency is generally defined in terms of the local accuracy of the classifiers in the region. When classifiers are chosen to form an ensemble, these studies do not take into account the diversity factor as the system is not required to generalize. In other words, when focused on a single instance, the general approach is that classifiers' performance on the different regions of the instance space is irrelevant. However, if we ignore the diversity of the classifiers in other regions, then we may form an ensemble of similar classifiers. These *similar* classifiers will be unable to compensate for each other's incorrect predictions.

Throughout this thesis, we have also proposed several methods for making predictions on new instances based on classifiers' responses. In these methods, we rely on the effectiveness of ensembles as they use the collective knowledge of several classifiers. However, unlike static ensembles, our focus has been on exploring the benefits of confidence in making dynamic predictions. Consequently, we either add confidence as one of the objectives in selecting a dynamic ensemble to make a prediction (as in Chapter 3), or as a criterion for choosing classifiers to form the reduced probability space (as in Chapter 4). Similar to the other dynamic class prediction methods, we do not focus on diversity among the classifiers. However, the results we obtain in those chapters lead us to investigate the effect of diversity.

We believe that the agreement among different classifiers is more important than that among similar ones in the local region in which a new data instance resides. In other words, we hypothesize that high *local* accuracy and high diversity outside of the local region may be desirable. Consequently, just like in static ensembles, diversity still plays a role in the dynamic class prediction.

It is worth noting that even though in Chapter 3 we use the conditional double fault measure, in this chapter we decide to use the disagreement measure for the diversity calculation as the conditional double fault cannot assess the pairwise diversity when one of the classifiers makes no error for the data instances considered.

5.1 Analysis of Diversity in Dynamic Ensemble Selection

The results of the experiments performed in Section 3.1.2 of Chapter 3 show that diversity is important for performance. In particular, when we analyze the effect of the diversity of the pool, we observe that diversity values decrease for all objectives, which in turn decreases the performance. In addition, to assess the importance of diversity in the dynamic setting, we compare the performance achieved with the $Conf$ objective against $Conf + Div$, as well as $Acc + Conf$ against $Acc + Conf + Div$ objectives (see Table 5.1). During the comparison, we observe that the objective function with high average diversity performs well compared to the other one for all datasets except “Liver disorder” for the $Conf_2$ definition. This motivates us to investigate the effect of diversity further even though

Table 5.1: Summary results of the final solution based on its validation data performance for the Conf$_1$ and Conf$_2$ definitions

	Australian							
	$Conf_1$				$Conf_2$			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Conf	0.553	0.951	0.388	0.792	0.553	0.944	**0.447**	**0.791**
Div+Conf	0.574	0.652	**0.566**	**0.825**	0.583	0.511	0.323	0.653
Acc+Conf	0.603	0.683	0.550	0.812	0.611	0.356	0.487	0.757
Acc+Conf+Div	0.580	0.654	**0.567**	**0.823**	0.582	0.203	**0.566**	**0.825**
	Diabetes							
	$Conf_1$				$Conf_2$			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Conf	0.583	0.968	0.442	0.741	0.582	0.960	**0.500**	**0.741**
Div+Conf	0.573	0.708	**0.551**	**0.743**	0.628	0.526	0.482	0.741
Acc+Conf	0.620	0.749	0.508	0.740	0.642	0.494	0.475	0.734
Acc+Conf+Div	0.583	0.708	**0.549**	**0.745**	0.588	0.263	**0.548**	**0.749**
	Heart							
	$Conf_1$				$Conf_2$			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Conf	0.540	0.889	0.408	0.729	0.539	0.879	**0.507**	**0.730**
Div+Conf	0.561	0.620	**0.564**	**0.764**	0.579	0.400	0.462	0.718
Acc+Conf	0.578	0.645	0.554	0.755	0.592	0.285	0.527	0.737
Acc+Conf+Div	0.566	0.619	**0.565**	**0.778**	0.568	0.160	**0.565**	**0.760**
	Liver Disorder							
	$Conf_1$				$Conf_2$			
	Acc	Conf	Div	Gen	Acc	Conf	Div	Gen
Conf	0.554	0.961	0.441	0.641	0.554	0.946	**0.481**	0.650
Div+Conf	0.562	0.675	**0.547**	**0.701**	0.588	0.491	0.473	**0.695**
Acc+Conf	0.594	0.703	0.504	0.690	0.602	0.400	0.484	**0.703**
Acc+Conf+Div	0.570	0.676	**0.544**	**0.699**	0.572	0.239	**0.542**	0.692

we do not need to generalize in the dynamic setting.

In this section, we aim to perform an in-depth analysis of diversity in the experimental setting discussed in Chapter 3 using more datasets. We can involve diversity in two ways to the system. First, we can add it as a search criterion to the objective function. Second, it can be used as a final ensemble selection criterion for the candidate solutions generated

Table 5.2: Pairwise t-test: Comparing average diversity and performance of search objectives that include diversity against the corresponding alternative

	Average Diversity[a]		Average Performance[a]	
	Conf	Acc+Conf	Conf	Acc+Conf
Conf+Div	(11,0,0)	-	(3,3,5)	-
Acc+Conf+Div	-	(11,0,0)	-	(2,6,3)

[a] Final ensembles are chosen based on validation accuracy.

using different weight combinations for a given new data instance. We explore these two ideas in this section.

5.1.1 Dynamic Ensemble Construction with Heuristic Search

To assess whether the contribution of the diversity is significant we follow the same approach undertaken in Section 3.1 of Chapter 1. However, we employ the classifiers constructed for Chapter 4. We run the experiment for all 11 datasets discussed in Chapter 4 except for "Mushrooms" and "Rcv". In Chapter 3, we find that choosing the final ensemble among the candidates based on ensemble validation accuracy outperforms other selection methods.

Table 5.2 shows the t-test results at the 5% significance level when performance and diversity values of ensembles are compared, respectively. Entries are specified as (x_1, x_2, x_3) and represent (wins, ties, losses) of the search objectives that include $Div(E)$ against the corresponding alternatives in all 11 datasets. Table 5.2 illustrates that when $Div(E)$ is included in the search objective, it generates more diverse classifiers. However, including diversity in the objective function does not necessarily improve the performance.

In addition, we analyze whether choosing the final ensemble among the generated

Table 5.3: Pairwise t-test: Diversity vs Ensemble validation accuracy

	Acc+Conf	Conf+Div	Acc+Conf+Div
Performance	(2,6,3)	(0,4,7)	(0,6,5)
Diversity	(11,0,0)	(11,0,0)	(11,0,0)

Table 5.4: Performance and diversity values for different ensemble selection criteria

	Ensemble Validation Accuracy					
	Acc+Conf		*Conf+Div*		*Acc+Conf+Div*	
Dataset	Div	Perf	Div	Perf	Div	Perf
a1a	0.140	0.782	0.162	0.783	0.155	0.785
australian	0.165	0.857	0.255	0.853	0.238	0.854
breast cancer	0.038	0.968	0.067	0.970	0.059	0.967
diabetes	0.195	0.764	0.237	0.752	0.224	0.757
german	0.200	0.714	0.256	0.716	0.240	0.716
heart	0.261	0.822	0.323	0.820	0.304	0.823
ionosphere	0.065	0.938	0.085	0.937	0.078	0.935
liver disorder	0.283	0.709	0.344	0.702	0.328	0.701
sonar	0.276	0.835	0.349	0.833	0.328	0.839
splice	0.191	0.573	0.497	0.635	0.488	0.628
w1a	0.004	0.972	0.007	0.971	0.006	0.972
	Ensemble Diversity					
a1a	0.157	0.782	0.170	0.782	0.169	0.784
australian	0.230	0.853	0.304	0.841	0.301	0.842
breast cancer	0.053	0.970	0.076	0.967	0.075	0.968
diabetes	0.232	0.753	0.258	0.741	0.258	0.742
german	0.242	0.716	0.271	0.715	0.270	0.715
heart	0.305	0.819	0.347	0.814	0.346	0.817
ionosphere	0.083	0.936	0.092	0.934	0.091	0.936
liver disorder	0.332	0.699	0.373	0.685	0.372	0.687
sonar	0.329	0.830	0.373	0.821	0.370	0.824
splice	0.368	0.578	0.504	0.612	0.500	0.604
w1a	0.006	0.971	0.008	0.972	0.008	0.971

solutions using diversity outperforms choosing based on ensemble validation accuracy. Table 5.3 illustrates the t-test result at 5% significance level, while Table 5.4 displays the diversity and performance values for these criteria. Entries (x_1, x_2, x_3) in Table 5.3 represent (wins, ties, losses) of the selection method based on diversity in all 11 datasets. While having diversity as the final ensemble selection criterion instead of ensemble validation accuracy increases diversity values for all datasets, it lowers the performance.

5.1.2 Regression Analysis for the Static Ensemble

Given a set of classifiers $\mathcal{C} = \{C_a | a = 1, .., M\}$, a validation dataset $\mathcal{D}$, and a test dataset $\mathcal{T}$, four regression models are built to assess the effect of diversity when an ensemble is chosen to be used for all new data instances (i.e. static ensemble):

- *Model 1*: $Y \sim Conf(E)$
- *Model 2*: $Y \sim Conf(E) + Div(E)$
- *Model 3*: $Y \sim Acc(E) + Conf(E)$
- *Model 4*: $Y \sim Acc(E) + Conf(E) + Div(E)$

We define the predictors, Acc(E), Conf(E), and Div(E) and the response as follows:

1. *Acc*(E): Average accuracy of the classifiers in the ensemble on the validation dataset

$$Acc(E) = \sum_{C_a \in E} \frac{Acc_{C_a}}{|E|} \tag{5.1}$$

   where Acc_{C_a} represents accuracy of classifier C_a on the validation set.

2. $Conf(E)$: Average confidence of an ensemble on all data instances in $\mathcal{T}$.

$$Conf(E,t) = \sum_{C_a \in E} \frac{Conf_{C_a}(t)}{|E|} \tag{5.2}$$

$$Conf(E) = \sum_{t \in \mathcal{T}} \frac{Conf(E,t)}{|\mathcal{T}|} \tag{5.3}$$

where $Conf_{C_a}(t)$ denotes the confidence of classifier C_a on instance t and $Conf(E,t)$ represents the average of classifier confidence on instance t in the ensemble.

3. $Div(E)$: Disagreement among the classifiers in the ensemble is measured as:

$$Div_{ab} = \sum_{v \in \mathcal{D}} \frac{\delta(\ell_{v,a} \neq \ell_{v,b})}{|\mathcal{D}|} \tag{5.4}$$

$$Div(E) = 2 \sum_{C_a \in E} \sum_{C_b \neq C_a \in E} \frac{Div_{ab}}{|E| \, (|E| \, - 1)} \tag{5.5}$$

where Div_{ab} represents the disagreements between C_a and C_b, and δx denotes the indicator function, which equals 1 if the x is true, and 0 otherwise.

4. Y: Average accuracy of the ensemble on the test dataset, $\mathcal{T}$.

For this experiment, 1000 ensembles are chosen randomly to make predictions on the test dataset $\mathcal{T}$. Table 5.5 displays the results. The entries are specified as (x_1, x_2) and represent the total number of runs for which confidence and diversity are significant predictors, respectively. The results indicate that diversity is a significant predictor after accounting for the other predictors, $Acc(E)$ and $Conf(E)$. Unlike diversity, confidence seems to be a weak predictor when we consider a static ensemble.

5.1.3 Logistic Regression Analysis for the Dynamic Ensemble

Several logistic regression models are built to assess the overall significance of the diversity in dynamic class prediction. In other words, we investigate whether having classi-

Table 5.5: Summary results for the regression models

Dataset	Model 1	Model 2	Model 3	Model 4
a1a	(100,-)	(86,100)	(97,-)	(76,100)
australian	(83,-)	(57,48)	(61,-)	(50,72)
breast cancer	(66,-)	(41,59)	(54,-)	(38,65)
diabetes	(97,-)	(88,67)	(80,-)	(80,56)
german	(62,-)	(58,64)	(60,-)	(53,65)
heart	(59,-)	(62,60)	(53,-)	(56,77)
ionosphere	(71,-)	(67,61)	(73,-)	(66,68)
liver disorder	(99,-)	(98,80)	(89,-)	(91,90)
sonar	(78,-)	(86,94)	(67,-)	(74,96)
splice	(98,-)	(32,100)	(85,-)	(57,100)
w1a	(46,-)	(47,32)	(46,-)	(45,36)

fiers that behave differently increases the probability of correct prediction for a given data instance. We build four models as follows:

- *Model 1*: $Y \sim Conf(E,t)$
- *Model 2*: $Y \sim Conf(E,t) + Div(E)$
- *Model 3*: $Y \sim Acc(E) + Conf(E,t)$
- *Model 4*: $Y \sim Acc(E) + Conf(E,t) + Div(E)$

In these models, the response Y denotes the correctness of a prediction. $Acc(E)$ and $Div(E)$ are defined in the same way as in Section 5.1.2. For the confidence measure, we use average classifier confidence $Conf(E,t)$ in the ensemble for instance t.

100 ensembles are generated randomly for *each* data instance in $\mathcal{T}$. For each generated ensemble, we calculate the values of the predictors and determine the correctness of the prediction. Table 5.6 shows the relevant results. The entries are specified as (x_1, x_2) and represent the total number of runs for which confidence and diversity are significant predictors, respectively. The results indicate that diversity is not a significant predictor.

Table 5.6: Summary results for the logistic regression models

Dataset	Model 1	Model 2	Model 3	Model 4
a1a	(100,-)	(100,55)	(100,-)	(100,47)
australian	(100,-)	(100,94)	(100,-)	(100,40)
breast cancer	(100,-)	(100,15)	(100,-)	(100,8)
diabetes	(100,-)	(100,29)	(100,-)	(100,18)
german	(100,-)	(100,29)	(100,-)	(100,31)
heart	(98,-)	(99,17)	(98,-)	(99,18)
ionosphere	(100,-)	(100,18)	(100,-)	(100,20)
liver disorder	(95,-)	(95,29)	(94,-)	(94,31)
sonar	(100,-)	(100,58)	(100,-)	(100,66)
splice	(100,-)	(100,54)	(100,-)	(100,32)
w1a	(100,-)	(100,12)	(100,-)	(100,9)

Unlike generalization capability of a *static* ensemble, the prediction capability of a *dynamic* ensemble is not affected by the diversity among the classifiers in the ensemble. Therefore, for individual instances, diversity loses its importance while confidence on the prediction becomes more crucial.

5.2 Analysis of Diversity in Dynamic Class Prediction Framework

In Chapter 4, our main focus is on the probability estimates returned by the classifiers when they make predictions. The key component of the framework introduced in that chapter is the reduction of the probability space. We introduce several methods for selecting classifiers to form the reduced space. One of the interesting results of that chapter is that the methods based on local accuracy do not perform well compared to the method based on confidence. Even though the results in Section 5.1 do not support the importance of diversity in dynamic class prediction, we suspect that choosing classifiers based on confidence provides diversity implicitly in the neighborhood. In this section, we carry out an analysis to compare the diversity among the classifiers in the reduced space to the diversity of:

Table 5.7: Pairwise t-test: Reduced space vs. benchmark diversity

	CombinedRS	Best Ens	Pool	LAcc
PTM	(10,0,1)	(2,0,9)	(2,0,9)	(3,3,5)
PTMA	(10,0,1)	(2,2,7)	(2,2,7)	(5,1,5)

1. the best static ensemble chosen by maximizing validation accuracy,
2. the whole classifier pool,
3. all chosen classifiers for the reduced space based on confidence combined,
4. chosen classifiers for the reduced space based on local accuracy.

Once the first set of k neighbors of a new data instance is determined, we find the confident or accurate classifiers on the neighbors and calculate the diversity among them. Overall running time of the system is quite fast even though we perform k-NN.

We perform pairwise t-test at the 5% significance level. Entries in Tables 5.7 and 5.8 are specified as (x_1, x_2, x_3) and represent (wins, ties, losses) for diversity. The best static ensemble and the classifier pool $\mathcal{C}$ are more diverse than the ensembles formed the reduced space. Even though classifiers with less than 50% accuracy are omitted from the pool, the pool's diversity is higher than the confident classifiers chosen to create the reduced space. When we use these ensembles to make predictions, the results indicate that using the ensemble formed by the confident classifiers chosen for the reduced space to make predictions is better than the other methods.

Our analysis here reveals conflicting results with those from Section 5.1 regarding the usefulness of diversity in the dynamic setting. However, we still believe that diversity has

Table 5.8: Pairwise t-test: Reduced space vs. benchmark performance

	CombinedRS	Best Ens	Pool	LAcc
PTM	(7,3,1)	(9,2,0)	(7,4,0)	(6,2,3)
PTMA	(4,5,2)	(8,2,1)	(5,5,2)	(8,3,0)

positive effect towards increasing the performance of the system. In Sections 5.3 and 5.4, we propose two methods to show the effect of diversity.

5.3 Distant Diversity

We believe that the classifiers chosen for a dynamic ensemble should behave similarly in the region in which the new instance resides, but differently outside of this area. In other words, we hypothesize that high *local* accuracy, combined with high diversity in other regions, may be desirable, and that just like in static ensembles, diversity still plays a role in dynamic class prediction. In this section, we propose the concept of "Distant Diversity" and a method for making dynamic predictions by increasing local accuracy and distant diversity.

5.3.1 Problem Formulation

Given a set of classifiers $\mathcal{C} = \{C_a | a = 1, .., M\}$, a validation dataset $\mathcal{D}$, and a new data instance, t, we first calculate the distance between t and v, $\forall v \in D$ and find the closest k instances. These instances form the neighborhood of instance t, which is defined as $\mathcal{D}_{in}(t)$. The data instances that are not in the neighborhood of t are defined as $\mathcal{D}_{out}(t) = \mathcal{D} \setminus \mathcal{D}_{in}(t)$.

$$\max_{E} \{LAcc(E,t), DDiv(E,t)\} \tag{5.6}$$

$$|E| = n \tag{5.7}$$

In this formulation, we attempt to simultaneously optimize two separate objectives. The first objective is the local ensemble accuracy, denoted by $LAcc(E,t)$. To calculate this measure, we first evaluate the performance of each classifier (denoted by C_a) on the neighborhood of the new data instance, $\mathcal{D}_{in}(t)$. $LAcc(C_a(\mathcal{D}_{in}(t)))$ represents the percentage of data instances in $\mathcal{D}_{in}(t)$ for which the classification of C_a is correct. The local ensemble accuracy is then the average of these individual classifier accuracies for each classifier in the ensemble.

$$LAcc_{C_a} = \sum_{v \in \mathcal{D}_{in}} \frac{\delta(\ell_{v,a} = \ell_v^\star)}{|\mathcal{D}_{in}|} \tag{5.8}$$

$$LAcc(E,t) = \sum_{C_a \in E} \frac{LAcc_{C_a}}{|E|} \tag{5.9}$$

where $\ell_{v,a}$ is the assigned label for instance v by classifier C_a and $\ell_v^\star$ is the actual label of v.

The second measure is the ensemble diversity outside of neighborhood of instance t, denoted by $DDiv(E,t)$. We calculate the ensemble diversity on the validation dataset using a *Disagreement Measure*. First, the pairwise diversity between two classifiers, C_a and C_b, is formulated as follows:

$$DDiv_{ab} = \sum_{v \in \mathcal{D}_{out}} \frac{\delta(\ell_{v,a} \neq \ell_{v,b})}{|\mathcal{D}_{out}|} \tag{5.10}$$

The diversity of the ensemble is then the average of the pairwise diversities for each possible combination of classifiers in the ensemble:

$$DDiv(E,t) = \sum_{C_a \in E} \sum_{\substack{C_b \in E \\ C_b \neq C_a}} \frac{DDiv_{ab}(t)}{|E|\ (|E|\ -1)} \tag{5.11}$$

Since our problem formulation involves two separate objectives, a method must be defined to combine them into a single objective function, f, for which a maximum can be sought. In this study, we consider linear combinations of the objectives. Since the relative scale and significance of the *local* accuracy and *distant* diversity is unknown in a dynamic setup, the weighting of each objective is dynamically varied to cover as much of the objective space as possible. Given the set of weights, $\mathcal{W}$:

$$f = w\ LAcc(E,t) + (1-w)\ DDiv(E,t),\ \text{where } w \in \mathcal{W}. \tag{5.12}$$

The First Improvement algorithm, discussed in Section 3.1.1 of Chapter 3, is slightly modified for these experiments. Since the purpose is to show the benefit of distant diversity alongside local accuracy, and to have comparable result, we start the search from the most locally accurate ensemble instead of a random one. Then, this algorithm is employed to search for the candidate classifier subsets (candidate ensembles) $\mathcal{CS}$. Even though this is a conservative approach and do not fully assess the benefit of diversity, we believe even in this situation we should benefit from diversity.

5.3.2 Effect of Weights

In this experiment, we use the PTM measure from Chapter 4. We decrease the weight on local accuracy, w, from 1 to 0 with 0.1 increments to analyze the effect of the chosen weights. Table 5.9 demonstrates the results of this experiment for neighborhood size equals to 13. The results indicate that the weight of the diversity $(1-w)$ does not affect the performance, especially for $w > 0.5$. Similar results are obtained for $k = 1, 7, 19, 25$; see Tables A.1 - A.4 in Appendix.

Since the results from previous experiment indicate that the weight value does not affect the performance once distant diversity is included in the objective function, we repeat

Table 5.9: Performance, local accuracy, and distant diversity values when $k = 13$

	Performance										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.765	0.805	0.805	0.804	0.805	0.805	0.805	0.805	0.805	0.805	0.802
australian	0.845	0.849	0.849	0.849	0.849	0.849	0.845	0.846	0.853	0.840	0.846
breast cancer	0.977	0.973	0.973	0.973	0.973	0.972	0.974	0.973	0.975	0.976	0.971
diabetes	0.765	0.757	0.757	0.757	0.757	0.758	0.758	0.759	0.760	0.757	0.743
german	0.714	0.729	0.729	0.729	0.729	0.729	0.729	0.730	0.731	0.732	0.727
heart	0.811	0.801	0.801	0.801	0.801	0.801	0.803	0.798	0.800	0.809	0.803
ionosphere	0.933	0.937	0.937	0.937	0.937	0.937	0.937	0.940	0.937	0.935	0.932
liver disorder	0.646	0.696	0.696	0.693	0.697	0.697	0.701	0.706	0.690	0.688	0.670
sonar	0.788	0.846	0.846	0.846	0.844	0.841	0.839	0.836	0.795	0.803	0.798
splice	0.566	0.777	0.777	0.775	0.776	0.774	0.771	0.764	0.762	0.746	0.689
w1a	0.971	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.972

	Local Accuracy										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.902	0.902	0.902	0.901	0.901	0.901	0.900	0.899	0.898	0.896	0.885
australian	0.924	0.924	0.923	0.921	0.916	0.908	0.892	0.862	0.833	0.814	0.801
breast cancer	0.998	0.998	0.998	0.998	0.998	0.997	0.997	0.996	0.994	0.991	0.987
diabetes	0.884	0.884	0.884	0.884	0.883	0.882	0.879	0.870	0.837	0.780	0.751
german	0.909	0.909	0.908	0.907	0.906	0.904	0.902	0.898	0.893	0.881	0.854
heart	0.933	0.933	0.933	0.932	0.930	0.927	0.919	0.894	0.833	0.784	0.764
ionosphere	0.982	0.982	0.982	0.982	0.982	0.981	0.981	0.980	0.979	0.974	0.970
liver disorder	0.834	0.834	0.834	0.833	0.831	0.825	0.808	0.769	0.700	0.652	0.627
sonar	0.862	0.862	0.862	0.859	0.856	0.848	0.830	0.772	0.707	0.679	0.665
splice	0.839	0.839	0.832	0.797	0.741	0.701	0.675	0.661	0.653	0.648	0.643
w1a	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996

	Distant Diversity										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.150	0.165	0.166	0.167	0.167	0.168	0.169	0.169	0.169	0.170	0.170
australian	0.208	0.218	0.222	0.229	0.237	0.247	0.260	0.276	0.285	0.290	0.290
breast cancer	0.054	0.071	0.072	0.072	0.072	0.073	0.074	0.074	0.074	0.075	0.075
diabetes	0.214	0.223	0.223	0.223	0.224	0.226	0.228	0.233	0.243	0.253	0.255
german	0.243	0.248	0.250	0.253	0.256	0.257	0.259	0.261	0.263	0.265	0.266
heart	0.281	0.282	0.283	0.286	0.288	0.292	0.299	0.312	0.333	0.342	0.342
ionosphere	0.077	0.099	0.099	0.100	0.100	0.101	0.101	0.101	0.102	0.103	0.103
liver disorder	0.272	0.273	0.273	0.275	0.279	0.287	0.300	0.320	0.344	0.353	0.354
sonar	0.227	0.229	0.231	0.239	0.245	0.254	0.269	0.299	0.322	0.326	0.327
splice	0.082	0.098	0.134	0.234	0.341	0.391	0.413	0.421	0.424	0.424	0.425
w1a	0.004	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007

[a] The column headers are for w, which denotes the weight on local accuracy.

the experiment by changing the weight of the local accuracy in the single objective from 1 to 0.9 for several neighborhood sizes. Table 5.10 demonstrates the improvement of including distant diversity in our optimization relative to having only local accuracy of the classifiers ($Perf(w = 0.9) - Perf(w = 1)$). For the *breast cancer*, *diabetes*, and *heart* datasets, we see small deterioration in the performance. For the remaining datasets, including distant diversity as a factor when choosing locally accurate classifiers has a positive effect. Therefore, as we increase the neighborhood size, diversity increases and in turn the overall performance of the system improves. However, for some datasets (e.g. splice) after a certain k, value we observe a decrease in the performance.

To sum up, the results support our hypothesis that diversity is useful in the dynamic setup when we consider it along with local accuracy. As we increase the weight of diversity in Equation 5.12, we do not see significant increase in the performance compared to $w = 0.9$ case. One reason is that for the *breast cancer* and *w1a* datasets, the classifiers are similar and highly accurate. Therefore, the diversity among the classifiers is considerably low and constructing ensembles in this manner becomes meaningless. Another reason is that our search algorithm starts from the best locally accurate classifier. Even though we change the weights, we find similar ensembles at the end of optimization. In other words, we may not be fully exploring the search space. In addition, for small neighborhood sizes distant diversity does not affect the performance. For small neighborhood sizes, there are several candidate classifiers with high local accuracy which imply that there are more possible ensembles compared to the case with large neighborhood. Therefore, our search algorithm may be terminating before it evaluates most of the candidate ensembles.

Table 5.10: Average performance difference between $w = 1$ and $w = 0.9$ cases

Dataset / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
a1a	0.027	0.033	0.030	0.035	0.037	0.035	0.040	0.040	0.036	0.037	0.040	0.037	0.036
australian	-0.006	-0.001	0.003	0.007	0.005	0.006	0.004	0.004	0.001	0.005	0.007	0.007	0.007
breast cancer	-0.004	-0.004	-0.004	-0.003	-0.003	-0.003	-0.004	-0.003	-0.003	-0.002	-0.003	-0.004	-0.004
diabetes	-0.042	-0.029	-0.029	-0.018	-0.014	-0.010	-0.008	-0.010	-0.007	0.000	0.001	0.002	0.003
german	0.012	0.005	0.007	0.013	0.008	0.013	0.015	0.009	0.013	0.010	0.012	0.012	0.009
heart	-0.017	-0.000	-0.015	0.006	-0.005	-0.002	-0.009	-0.007	-0.013	0.000	-0.005	-0.011	-0.004
ionosphere	0.004	-0.003	-0.003	0.003	0.001	0.004	0.004	0.003	0.003	0.003	0.003	0.003	0.003
liver disorder	0.020	0.023	0.029	0.039	0.041	0.051	0.049	0.046	0.048	0.059	0.048	0.045	0.055
sonar	0.034	0.058	0.058	0.048	0.053	0.053	0.058	0.048	0.048	0.056	0.051	0.058	0.065
splice	0.202	0.210	0.222	0.215	0.217	0.215	0.211	0.204	0.197	0.193	0.184	0.184	0.181
w1a	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001

[a] The column headers are for k, which denotes the neighborhood size.

5.3.3 Effect of Distance Measure

Since finding k nearest neighbors is a crucial step of the proposed method, we investigate whether the choice of distance measure has a significant effect on the results. In that regard, we consider the five distance measures discussed in Chapter 4.

We perform an experiment for $w = 0.9$. Table 5.11 shows the percent improvement of PTM over the compared distance measures. Positive (negative) entries indicate that PTM performs better (worse) compared to the other distance measure. The results indicate that there is not much difference in performance for different distance measures. Table A.5 in the Appendix shows results from this experiment for $w = \{0.8, 0.6, 0.4, 0.2\}$. We reach the same conclusion. Therefore, we decide to continue our analysis with PTM.

5.3.4 Comparison Against Baseline Methods

We perform two more experiments to evaluate the effect of distant diversity (DDiv) compared to global (Div) and local diversity (LDiv), while maximizing local accuracy (LAcc). Unlike distant diversity, global diversity considers disagreements between classifiers on all data instances (both inside and outside of the neighborhood). Local diversity only considers the disagreements on the data instances in the neighborhood. In addition, we compare dynamic optimization with distant diversity and local accuracy against the static model with global accuracy (Acc) and global diversity.

One may argue that diversity among the classifiers should not be restricted to the outside of the neighborhood: the size of the neighborhood affects this diversity measure and we will not be able to capture its full effect. Additionally, another argument can be made

Table 5.11: Comparison of distance measures when $w = 0.9$

Comparison / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
PTM vs. PTMA	2.682	3.944	3.853	4.248	4.406	4.319	4.262	4.109	3.846	4.123	3.735	3.868	3.695
PTM vs. TM	4.818	2.201	0.883	0.923	0.876	1.091	0.850	0.702	0.497	1.091	0.818	0.676	0.890
PTM vs. OTM	0.961	1.158	0.257	1.013	0.729	1.070	1.360	0.993	0.914	1.440	1.212	1.319	1.589
PTM vs. ED	1.618	2.0319	1.781	1.230	1.298	1.186	1.316	0.485	0.473	0.804	0.425	0.286	0.582
PTMA vs. TM	2.797	-0.932	-1.945	-2.408	-2.549	-2.320	-2.544	-2.559	-2.536	-2.259	-2.198	-2.447	-2.079
PTMA vs. OTM	-1.204	-2.097	-2.647	-2.372	-2.749	-2.376	-2.090	-2.343	-2.201	-1.963	-1.854	-1.881	-1.458
PTMA vs. ED	-0.318	-1.094	-1.207	-2.114	-2.166	-2.252	-2.121	-2.763	-2.528	-2.477	-2.516	-2.748	-2.302

[a] The column headers are for k, which denotes the neighborhood size.

Table 5.12: Performance of several diversity definitions for $k = 13$ and $w = 0.9$

Dataset	DDiv+LAcc	Div+LAcc	LDiv+LAcc	Div+Acc
a1a	0.805	0.805	0.805	0.772
australian	0.849	0.849	0.849	0.857
breast cancer	0.973	0.973	0.973	0.972
diabetes	0.757	0.757	0.757	0.765
german	0.729	0.729	0.729	0.717
heart	0.801	0.801	0.801	0.816
ionosphere	0.937	0.937	0.937	0.932
liver disorder	0.696	0.696	0.696	0.690
sonar	0.846	0.846	0.846	0.808
splice	0.777	0.777	0.777	0.573
w1a	0.973	0.973	0.973	0.972

in favor of local diversity. Specifically, once the neighborhood of a new instance is defined, we should follow a similar approach to finding static ensemble by maximizing accuracy and diversity in this local region, since this chosen ensemble could be seen as the most competent ensemble in this given neighborhood.

We focus on finding the closest k instances to the new data instance to form its neighborhood. Unlike the whole data space, in this neighborhood the data instances should have similar features including class labels. Therefore, an expert classifier for this region should be highly locally accurate. An ensemble composed of the expert classifiers for a given region should behave similarly in the associated region. However, maximizing global diversity or local diversity would lead the classifiers in an ensemble to behave differently in the local region, which is not what we need. However, diversity outside the local region is still important, since we want predictors that come to the same conclusion in different ways. Consequently, diversity outside of the local region should be considered instead of local diversity.

Table 5.12 displays the mean performance of each method over 10 runs for $k = 13$ and

$w = 0.9$. Based on the results provided in Table 5.12, dynamically maximizing local accuracy and distant diversity is better than maximizing global diversity and accuracy except for three datasets: *australian*, *diabetes*, and *heart*. However, we do not observe much difference among global, local, and distant diversity combined with local accuracy in terms of their contribution to the performance. For high k and small w values we observe performance differences between these diversity concepts and benefit of distant diversity. This could also be an artifact of our search algorithm as we start the search from most locally accurate ensemble.

Distant Diversity is also compared against following methods as well:

- Static step of the framework discussed in 4(Static Step)
- KNORA Eliminate (KNORA_E), and KNORA Union (KNORA_U)
- Best Classifier
- Ensemble composed of best 25 classifiers (Best 25)
- Single SVM
- Ensemble with maximum validation accuracy(Best Ens)

Similar to our method, the static step and KNORA methods are dynamic class prediction methods. Unlike our method, once the neighborhood is defined, the static step uses the labels of the neighbors to make the prediction. Hence, comparing our method against the static step could inform us whether we should rely on the label of the data instances in the neighborhood. The KNORA methods follow a similar approach to our method as they find the competent classifiers in the neighborhood. KNORA Eliminate focuses on finding the most accurate classifier(s) in the neighborhood and does not take diversity into account as

Table 5.13: Performance of benchmarks methods

Dataset	DDiv+LAcc	Static Step	KNORA_E	KNORA_U	Best Classifier	Best 25	Single SVM	Best Ens
a1a	**0.805**	0.798	0.732	0.732	0.756	0.778	0.751	0.764
australian	0.849	**0.853**	0.751	0.751	0.768	0.842	0.814	0.850
breast cancer	0.973	**0.976**	0.973	0.973	0.952	0.974	0.969	0.974
diabetes	0.757	0.750	0.632	0.632	0.654	0.651	0.738	**0.762**
german	**0.729**	**0.729**	0.668	0.668	0.697	0.700	0.696	0.712
heart	0.801	0.803	0.749	0.749	0.709	0.805	0.748	**0.807**
ionosphere	**0.937**	0.934	0.917	0.917	0.905	0.929	0.926	0.935
liver disorder	0.696	0.687	0.538	0.538	0.620	0.628	**0.719**	0.664
sonar	**0.846**	0.820	0.699	0.699	0.714	0.755	0.719	0.779
splice	**0.777**	0.771	0.624	0.624	0.570	0.566	0.562	0.566
w1a	0.973	**0.974**	0.965	0.965	0.971	0.971	0.970	0.971

a factor. Meanwhile, KNORA Union considers diversity by weighingthe predictions of classifiers on a new data instance based on their performance in the neighborhood. In addition to dynamic class prediction methods, having static methods as benchmarks could help us determine whether the trade-off between accuracy and computational cost is worth taking. Performance of these baselines is presented in Table 5.13. In this table, for static step and KNORA methods, we consider $k = 13$. Maximizing distant diversity and local accuracy outperforms all of the baselines except “static step of the framework” and “best ensemble” over all datasets. Distant diversity along with local accuracy performs better than the static step of the framework and the best ensemble on the validation set for 6 and 7 datasets, respectively.

To perform in detail analysis, we carry out an experiment to compare the performance of distant diversity against static step and KNORA methods for various weight and neighborhood size values. Pairwise t-test at 5% significance level is performed to compare the performance of these methods. The results are provided in Tables A.6, A.7, and A.8 in the Appendix. Entries in these tables represent (wins, ties, losses) for distant diversity. Table A.6 shows that static step of the framework performs better than constructing ensembles solely based on local accuracy ($w = 1$). However, when distant diversity along with local accuracy is considered ($w < 1$), system performance becomes similar to static step of the framework. Tables A.7, and A.8 indicate that the proposed method outperforms KNORA Eliminate and KNORA Union.

To sum up, the results presented in this section clearly show that when constructing a dynamic ensemble, diversity among the classifiers is a significant factor regardless of diversity concept (distant, global or local).

5.4 Weighted Distant Diversity and Weighted Local Accuracy

The results obtained from Section 5.3.4 indicate that there could be a scaling issue between local accuracy and distant diversity. In particular, when we look at the change in local accuracy and diversity for different weights, we observe the effect of change in diversity on the performance for high k values and high w values. For the rest, local accuracy is dominating the objective function.

Our hypothesis is that the ensemble composed of classifiers that have agreement on the data instances close to the new data and disagree on the data instances further away from the new instance should make more accurate prediction on this new instance. Therefore, when the local accuracy of a classifier is evaluated, correct predictions on the points that are close to the test instance should weigh more than the points further away. Similarly, when diversity between two classifiers is considered, the disagreement for the data instances further away from the new data instance should be more valuable. To avoid the effect of neighborhood size, we decide to weigh the accuracy of classifiers and diversity among them based on the distance between the new instance and all the other instances. The new classifier accuracy and disagreement measure definitions are as follows:

Table 5.14: Performance of weighted distant diversity and weighted local accuracy

Dataset / w[a]	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0
a1a	0.773	0.810	0.810	0.810	0.810	0.807	0.806	0.806	0.802	0.801	0.796
australian	0.849	0.860	0.860	0.861	0.858	0.853	0.853	0.849	0.843	0.841	0.834
breast cancer	0.977	0.975	0.975	0.975	0.975	0.972	0.974	0.976	0.976	0.974	0.974
diabetes	0.770	0.760	0.760	0.760	0.766	0.757	0.763	0.758	0.750	0.749	0.748
german	0.724	0.730	0.731	0.731	0.730	0.729	0.732	0.732	0.730	0.727	0.728
heart	0.811	0.809	0.803	0.801	0.796	0.803	0.805	0.803	0.816	0.822	0.820
ionosphere	0.935	0.938	0.938	0.938	0.938	0.939	0.942	0.937	0.935	0.934	0.935
liver disorder	0.646	0.700	0.700	0.700	0.703	0.696	0.683	0.690	0.683	0.678	0.681
sonar	0.788	0.858	0.858	0.858	0.858	0.856	0.829	0.820	0.813	0.803	0.798
splice	0.566	0.648	0.650	0.653	0.645	0.610	0.597	0.594	0.595	0.591	0.591
w1a	0.973	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.973

[a] The column headers are for w, which denotes the weight on local accuracy.

$$LwAcc_{C_a} = \frac{\sum\limits_{v \in \mathcal{D}} \delta(\ell_{v,a} = \ell^{\star}_{v}) \frac{1}{d(t,v)}}{\sum\limits_{v \in \mathcal{D}} \frac{1}{d(t,v)}} \tag{5.13}$$

$$DwDiv_{ab}(t) = \frac{\sum\limits_{v \in \mathcal{D}} \delta(C_a(v) \neq C_b(v)) d(t,v)}{\sum\limits_{v \in \mathcal{D}} d(t,v)} \tag{5.14}$$

$$DwDiv(E,t) = \frac{\sum\limits_{C_a \in E} \sum\limits_{C_b \neq C_a \in E} Div_{ab}(t)}{(|E|\ (|E|\ - 1))} \tag{5.15}$$

where $d(t,v)$ is the distance between data instances t and v.

When the new local accuracy and new diversity measures are applied, for weight values $1 > w > 0.6$ adding diversity in the objective function increases performance. Table 5.14 shows the performance values for weighted distant diversity and weighted local accuracy for different weights. Similar to our analysis in Section 5.3.2, incorporating diversity in addition to accuracy provides higher performance except for the *breast cancer, diabetes* and *heart* datasets.

Table 5.15: Performance of global diversity and global accuracy

Dataset / w[a]	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0
a1a	0.769	0.772	0.770	0.767	0.771	0.774	0.771	0.769	0.775	0.770	0.775
australian	0.849	0.857	0.851	0.862	0.853	0.839	0.848	0.841	0.838	0.842	0.841
breast cancer	0.973	0.972	0.974	0.976	0.977	0.974	0.976	0.974	0.975	0.972	0.974
diabetes	0.763	0.765	0.758	0.762	0.753	0.758	0.747	0.747	0.754	0.747	0.747
german	0.710	0.717	0.712	0.713	0.716	0.719	0.711	0.718	0.713	0.717	0.716
heart	0.807	0.816	0.820	0.803	0.809	0.798	0.814	0.816	0.811	0.796	0.794
ionosphere	0.933	0.932	0.936	0.930	0.932	0.939	0.936	0.935	0.936	0.937	0.930
liver disorder	0.701	0.690	0.699	0.696	0.681	0.688	0.681	0.678	0.658	0.697	0.677
sonar	0.786	0.808	0.793	0.822	0.822	0.805	0.776	0.796	0.820	0.800	0.794
splice	0.567	0.573	0.581	0.585	0.576	0.578	0.584	0.578	0.580	0.589	0.579
w1a	0.972	0.972	0.972	0.972	0.971	0.972	0.972	0.971	0.971	0.972	0.972

[a] The column headers are for w, which denotes the weight on local accuracy.

Table 5.16: Performance of weighted distant diversity and local accuracy against other diversity definitions for $k = 13$ and $w = 0.9$

Dataset	DDiv+LAcc[a]	Div+Acc[b]	DwDiv+LwAcc[c]
a1a	0.805	0.772	**0.810**
australian	0.849	0.857	**0.860**
breast cancer	0.973	0.972	**0.975**
diabetes	0.757	**0.765**	0.760
german	0.729	0.717	**0.730**
heart	0.801	**0.816**	0.809
ionosphere	0.937	0.932	**0.938**
liver disorder	0.696	0.690	**0.700**
sonar	0.846	0.808	**0.858**
splice	**0.777**	0.573	0.648
w1a	0.973	0.972	**0.974**

[a] DDiv and LAcc denote distant diversity and local accuracy.
[b] Div and Acc denote global diversity and global accuracy.
[c] DwDiv and LwAcc represent weighted distant diversity and weighted local accuracy.

Table 5.17: Performance of weighted distant diversity and local accuracy against other benchmarks methods

Dataset	Static Step	Best Classifier	Best 25	Single SVM	Best Ens	DwDiv+LwAcc
a1a	0.798	0.756	0.778	0.751	0.764	**0.810**
australian	0.853	0.768	0.842	0.814	0.850	**0.860**
breast cancer	**0.976**	0.952	0.974	0.969	0.974	0.975
diabetes	0.750	0.654	0.651	0.738	**0.762**	0.760
german	0.729	0.697	0.700	0.696	0.712	**0.730**
heart	0.803	0.709	0.805	0.748	0.807	**0.809**
ionosphere	0.934	0.905	0.929	0.926	0.935	**0.938**
liver disorder	0.687	0.620	0.628	**0.719**	0.664	0.700
sonar	0.820	0.714	0.755	0.719	0.779	**0.858**
splice	**0.771**	0.570	0.566	0.562	0.566	0.648
w1a	0.974	0.971	0.971	0.970	0.971	**0.974**

Based on Tables 5.14 and 5.15, except for the *heart* dataset, adjusting the weight of data instances while focusing on each new test case performs better than assuming every data instance has the same importance for the test case. Similar to our analysis in 5.3.4, we compare performance achieved by maximizing weighted distant diversity and weighted local accuracy against the baselines specified in that section. The results are presented in Tables 5.16 and 5.17 for $k = 13$ and $w = 0.9$.

Table 5.16 shows the comparison results against other diversity and accuracy concepts including distant diversity. Weighted distant diversity and weighted local accuracy together performs better than all the other diversity and accuracy concepts.

We also carry out an experiment to compare the proposed diversity and local accuracy measures against several other benchmarks. The results indicate that our method dominates all of the benchmarks specified in Table 5.17. For detail comparison, we carry out an experiment to compare the performance of distant diversity against static step. The results are provided in Table A.9 in the Appendix. Entries in this table represent (wins, ties, losses) for the case where weighted distant diversity and weighted local accuracy are maximized simultaneously. This table shows that the system with our proposed diversity and accuracy measures performs similar to the static step discussed in Chapter 4.

5.5 Conclusion

In this chapter we have investigated the role of diversity and proposed two diversity concepts for dynamic ensemble selection. First, we carried out regression analysis for the static ensemble selection and confirmed that diversity was necessary for generalization while confidence is not very relevant. Then, we ran logistic regression models for the dynamic steps and found that the results for the static case have been reversed: confidence was a significant predictor while diversity did not contribute much to the prediction capability, which was contrary to our expectation. Second, we anaylized whether diversity was part of the success of our framework in Chapter 4. According to our results, diversity of the reduced space was irrelevant.

The findings from the previous exercises did not support our hypothesis. Therefore, we proposed alternative ways to test our claim. The first alternative was distant diversity, which considered disagreements among the classifiers outside of the neighborhood. Based on our experimental results, we concluded that incorporating diversity with local accuracy improved the performance of dynamic ensembles regardless of the diversity concept (global, local, distant).

To avoid the issue of finding the optimal neighborhood size for each dataset, we proposed a second alternative that changes the definition of neighborhood. In this approach, all data instances were in the neighborhood. However, the importance of data instances for accuracy and diversity depended on the distance to the new instance. We demonstrated that weighted distant diversity and weighted local accuracy outperformed all benchmark methods.

Our proposed methods perform better for some datasets than others. This depends on the effect of the characteristics of those datasets on the generated pool of classifiers. For instance, the "diabetes" dataset has 8 features and 768 data instances. The ratio of minority to majority class label is 0.54. The best classifier in our pool has 65.4% accuracy. However, a single SVM generated using the whole dataset has 73.8% accuracy. This may indicate the need for the full feature set to generate expert classifiers. Additionally, the performance of the methods that take into account the whole dataset (i.e. "Best Ensemble", "DwDiv+LwAcc", and "Div+Acc") are better than the methods that focus on locality (i.e. "Static Step" and "DDiv+LAcc"). This could be due to the minority and majority class data instances being highly mixed in the neighborhood. Furthermore, even though the "breast cancer" dataset has similar characteristics to the "diabetes" dataset, the classifiers in the generated pool have high accuracy but are not diverse (see Table 4.2). This is because data instances belonging to the two classes are well separated. Let's also consider the "heart" and "liver disorder" datasets. The number of features and data instances in these datasets are low. Even though we are able to create diverse and accurate classifiers, the number of validation data instances are not enough to clearly define a local neighborhood of a new instance (e.g. for the "heart" dataset, 24 and 30 data instances from minority and majority class labels, respectively). Since the diversity of the ensemble is maximized in our proposed methods, the pool of base classifiers should be diverse. However, for the “ionosphere” and the “w1a” datasets, the pool is not diverse at all. Hence, our proposed methods perform similarly to the static ensemble generation methods. The generated pool of classifiers for the remaining datasets have diverse classifiers. In addition, the size of the validation data for these datasets is large enough to establish locality and the ratio between minority and majority classes is not too low. Consequently, our proposed methods outperform the benchmark methods for these datasets.

CHAPTER 6
CONCLUSIONS AND FUTURE DIRECTIONS

Ensembles of classifiers have been shown to improve prediction performance compared to a single classifier. Even though, each new data instance is different, existing static methods treat them similarly. Therefore, dynamic class prediction methods focus on individual data instances to increase the generalization. It has been shown that dynamic class prediction methods are better than static ones. However,while constructing an ensemble for each new data instance, these methods do not consider neither confidence nor diversity. In this thesis, we proposed several novel ways to incorporate the diversity and confidence of classifiers to approach dynamic class prediction.

In Chapter 3, we hypothesized that *confidence* of a classifier could be used as an indicator of its competency. We proposed two measures for evaluating the confidence of an ensemble. First, we carried out several experiments to assess whether confidence by itself would be enough or it should be combined with *accuracy* or *diversity* or both. The results of these experiments were inconclusive, especially in regards to showing that ensembles with accurate and confident classifiers make more accurate predictions. We therefore proposed "prediction ranking" and showed that accurate, confident and diverse classifiers should make better predictions. Even though the results using prediction ranking support our claim, the underlying relationship among these factors are yet to be defined. We further assessed the overall significance of the factors affecting ensembles' performance using the whole data. These results highlighted the benefits of having confident and diverse classifiers. Thus, instead of taking into account the overall accuracy of a classifier, we might focus on the local accuracy or local confidence of a classifier.

We used a prediction ranking to measure the effect of accuracy and confidence on generalization. One extension to our work could be using prediction ranking for choosing classifiers for dynamic ensemble construction. Each classifier's predictions could be ordered by the associated confidence value. After the classifier predicts a new data instance with some confidence, we could find the ranking of this new instance and choose classifiers based on these ranking values.

In Chapter 4, we changed our approach to dynamic class prediction. Instead of choosing classifiers to form an ensemble to make the prediction, we focused on finding similar instances to the test instance and assigned a label based on those of similar data instances. We proposed a framework using two dissimilarity measures, PTM and PTMA, based on the probability estimates returned for a particular class. We concluded that the two proposed measures outperformed the baseline measures. Our experiments also revealed that using classifiers' outputs was better than using the original feature space to find similar instances. Furthermore, we showed that performing k-NN in the probability-based classifier space was better than several benchmark static and dynamic methods.

Diversity has not been considered as a factor in dynamic class prediction since a dynamic ensemble is chosen for a particular data instance. However, not accounting for diversity may lead to having similar classifiers in the ensemble, which makes it harder to average out the prediction error (if similar classifiers are wrong in their prediction). We investigated the role of diversity and proposed two diversity concepts for dynamic ensemble selection in Chapter 5. The first method is "distant diversity", which considers disagreements among the classifiers outside of the new instance's neighborhood. Based on our experimental results, we concluded that incorporating diversity with local accuracy improves the performance of dynamic ensembles. To avoid the issue of finding the optimal neighborhood size for each dataset, we proposed a second method that considers all data instances to be in the neighborhood, and assigns them weights depending on the distance to the new instance. We demonstrated that weighted distant diversity and weighted local accuracy outperformed all benchmark methods.

It is worth nothing that the methods proposed in Chapters 4 and 5 make use of a pool of diverse classifiers to increase the prediction capability of the system. Particularly, these methods focus on defining a neighborhood of data instances based on the classifiers' responses, except for weighted distant diversity and weighted local accuracy. Hence, when generating the pool of classifiers, the characteristics of the datasets should be considered since these have considerable impact on the generated classifiers. The approach that we follow for generating training data is random sampling of features and data instances. If the number of data instances or features is low, or if the features are correlated, then the generated classifiers could have low accuracy. Two factors we should consider are: (1) the distribution of class labels and (2) the degree of separation of data instances from each class in the feature space. These two factors affect the diversity of the generated pool. Besides having enough features and data instances, our proposed methods work best for the datasets in which the minority and majority classes are not perfectly separated and there are enough observations for each class label. Furthermore, if data instances from both class labels appear in each other's neighborhood, then the weighted distant diversity and weighted local accuracy method may be more convenient.

If the class ratio is too low, our methods could be modified to focus on the minority class to avoid the bias towards the majority class. Once the validation data instances from minority and majority class labels are identified, the classifiers that incorrectly classify the minority instances can be removed from the pool. Then, the probability-based space can be formed using the remaining classifiers, and the proposed methods discussed in Chapters 4 or 5 can be followed accordingly. After generating the probability space, an alternative approach is to assess the similarity of new data instances to those from each class label separately. To do so, the similarity between the new data and validation data instances from the minority class label is first calculated. Then, the average of the calculated distance values is taken. The same procedure can be applied to the data instances from the majority class label. Consequently, we can decide whether a new data instance is more similar to those from minority or majority class labels and assign the label accordingly.

The downside of dynamic prediction systems is the running time since, for each new data instance, a search is performed either for finding the dynamic ensemble or similar data instances to assign a label to this new instance. One way to speed up the process could be clustering data instances using the PTM or PTMA measures proposed in Chapter 4, and generating several candidate ensembles for each cluster using the proposed diversity methods in Chapter 5 as a preprocessing step. When a new data instance is received, it will be assigned to a cluster. Either data instances or candidate ensembles for that cluster could then be used to make prediction on this new instance.

APPENDIX A
TABLES FOR CHAPTER 5

Table A.1: Performance, local accuracy, and distant diversity values when $k = 1$

	Performance										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.760	0.787	0.795	0.787	0.788	0.791	0.787	0.787	0.788	0.788	0.790
australian	0.828	0.822	0.818	0.814	0.817	0.814	0.818	0.816	0.816	0.817	0.817
breast cancer	0.977	0.972	0.972	0.974	0.972	0.972	0.973	0.974	0.974	0.973	0.971
diabetes	0.758	0.716	0.718	0.727	0.723	0.726	0.718	0.727	0.716	0.719	0.721
german	0.714	0.726	0.717	0.720	0.715	0.717	0.721	0.720	0.720	0.722	0.723
heart	0.809	0.792	0.787	0.794	0.781	0.798	0.790	0.788	0.796	0.790	0.788
ionosphere	0.929	0.933	0.933	0.927	0.930	0.932	0.927	0.932	0.929	0.929	0.933
liver disorder	0.626	0.646	0.641	0.643	0.641	0.649	0.654	0.658	0.638	0.651	0.638
sonar	0.788	0.822	0.825	0.829	0.822	0.829	0.827	0.827	0.829	0.825	0.827
splice	0.566	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.768	0.768
w1a	0.970	0.971	0.972	0.971	0.971	0.971	0.972	0.971	0.971	0.971	0.971

	Local Accuracy										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
australian	0.993	0.993	0.993	0.993	0.993	0.993	0.993	0.993	0.993	0.993	0.993
breast cancer	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
diabetes	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995
german	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
heart	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
ionosphere	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
liver disorder	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
sonar	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
splice	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
w1a	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997

	Distant Diversity										
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.138	0.165	0.165	0.165	0.165	0.165	0.165	0.165	0.165	0.165	0.165
australian	0.221	0.272	0.272	0.272	0.272	0.272	0.272	0.272	0.272	0.272	0.272
breast cancer	0.051	0.070	0.070	0.070	0.070	0.070	0.070	0.070	0.070	0.070	0.070
diabetes	0.216	0.241	0.241	0.241	0.241	0.241	0.241	0.241	0.241	0.241	0.241
german	0.206	0.254	0.254	0.255	0.254	0.254	0.254	0.254	0.254	0.254	0.254
heart	0.280	0.326	0.326	0.326	0.326	0.326	0.326	0.326	0.326	0.326	0.326
ionosphere	0.067	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089
liver disorder	0.298	0.336	0.336	0.336	0.336	0.336	0.336	0.335	0.335	0.336	0.336
sonar	0.254	0.301	0.301	0.301	0.300	0.301	0.301	0.302	0.301	0.301	0.301
splice	0.105	0.189	0.189	0.188	0.189	0.189	0.189	0.189	0.189	0.188	0.188
w1a	0.004	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007

[a] The column headers are for w, which denotes the weight on local accuracy.

Table A.2: Performance, local accuracy, and distant diversity values when $k = 7$

					Performance						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.765	0.800	0.800	0.800	0.800	0.800	0.800	0.800	0.799	0.800	0.797
australian	0.845	0.851	0.851	0.851	0.852	0.851	0.852	0.852	0.849	0.845	0.841
breast cancer	0.977	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.974	0.975
diabetes	0.765	0.747	0.747	0.747	0.747	0.747	0.745	0.746	0.755	0.747	0.748
german	0.714	0.727	0.727	0.727	0.727	0.727	0.727	0.729	0.731	0.730	0.730
heart	0.811	0.816	0.816	0.816	0.816	0.816	0.813	0.813	0.809	0.803	0.824
ionosphere	0.933	0.936	0.936	0.936	0.936	0.936	0.936	0.936	0.937	0.937	0.935
liver disorder	0.646	0.686	0.686	0.686	0.688	0.686	0.684	0.684	0.688	0.697	0.651
sonar	0.788	0.837	0.837	0.837	0.834	0.841	0.834	0.844	0.817	0.824	0.820
splice	0.566	0.781	0.781	0.781	0.780	0.778	0.775	0.771	0.771	0.763	0.710
w1a	0.971	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.972

					Local Accuracy						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.912	0.912	0.912	0.912	0.911	0.911	0.910	0.909	0.908	0.905	0.884
australian	0.937	0.937	0.936	0.935	0.932	0.927	0.917	0.894	0.853	0.820	0.799
breast cancer	0.998	0.998	0.998	0.998	0.998	0.998	0.997	0.997	0.996	0.992	0.984
diabetes	0.915	0.915	0.915	0.915	0.914	0.914	0.912	0.905	0.882	0.809	0.747
german	0.949	0.949	0.949	0.948	0.946	0.945	0.942	0.938	0.934	0.922	0.879
heart	0.970	0.970	0.969	0.969	0.967	0.964	0.959	0.946	0.897	0.819	0.777
ionosphere	0.981	0.981	0.981	0.981	0.981	0.980	0.980	0.979	0.978	0.972	0.963
liver disorder	0.876	0.876	0.876	0.876	0.875	0.872	0.863	0.831	0.750	0.674	0.633
sonar	0.916	0.916	0.916	0.915	0.913	0.907	0.894	0.862	0.768	0.724	0.702
splice	0.852	0.851	0.844	0.814	0.771	0.741	0.722	0.711	0.704	0.699	0.691
w1a	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.994

					Distant Diversity						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.147	0.163	0.163	0.164	0.164	0.165	0.165	0.166	0.166	0.167	0.168
australian	0.213	0.227	0.229	0.234	0.239	0.246	0.253	0.265	0.279	0.285	0.286
breast cancer	0.052	0.070	0.070	0.070	0.070	0.070	0.071	0.071	0.071	0.072	0.072
diabetes	0.213	0.222	0.222	0.222	0.223	0.224	0.225	0.229	0.236	0.248	0.252
german	0.232	0.243	0.245	0.247	0.250	0.252	0.254	0.256	0.258	0.260	0.261
heart	0.279	0.284	0.286	0.288	0.292	0.295	0.298	0.306	0.321	0.335	0.337
ionosphere	0.071	0.092	0.092	0.092	0.092	0.092	0.093	0.093	0.093	0.094	0.095
liver disorder	0.278	0.279	0.280	0.281	0.283	0.286	0.293	0.309	0.335	0.349	0.352
sonar	0.254	0.259	0.260	0.263	0.267	0.273	0.285	0.302	0.333	0.341	0.342
splice	0.098	0.122	0.156	0.243	0.325	0.363	0.379	0.385	0.387	0.388	0.389
w1a	0.004	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007

[a] The column headers are for w, which denotes the weight on local accuracy.

Table A.3: Performance, local accuracy, and distant diversity values when $k =$ 19

					Performance						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.765	0.802	0.802	0.802	0.802	0.802	0.802	0.802	0.802	0.804	0.803
australian	0.845	0.850	0.850	0.850	0.852	0.850	0.850	0.859	0.857	0.838	0.849
breast cancer	0.977	0.974	0.974	0.974	0.974	0.974	0.975	0.975	0.977	0.977	0.975
diabetes	0.765	0.765	0.765	0.764	0.765	0.766	0.765	0.766	0.763	0.757	0.749
german	0.714	0.724	0.724	0.724	0.724	0.726	0.726	0.726	0.726	0.727	0.726
heart	0.811	0.811	0.811	0.811	0.811	0.813	0.811	0.805	0.790	0.801	0.805
ionosphere	0.933	0.936	0.936	0.936	0.936	0.936	0.936	0.939	0.939	0.932	0.930
liver disorder	0.646	0.706	0.706	0.706	0.707	0.700	0.712	0.713	0.694	0.688	0.665
sonar	0.788	0.844	0.844	0.844	0.844	0.832	0.839	0.817	0.812	0.800	0.786
splice	0.566	0.759	0.759	0.759	0.755	0.755	0.756	0.747	0.744	0.731	0.676
w1a	0.971	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.971

					Local Accuracy						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.898	0.898	0.897	0.897	0.896	0.896	0.895	0.895	0.894	0.893	0.885
australian	0.919	0.918	0.917	0.914	0.909	0.899	0.877	0.846	0.827	0.814	0.806
breast cancer	0.998	0.998	0.997	0.997	0.997	0.996	0.995	0.994	0.993	0.990	0.988
diabetes	0.866	0.866	0.866	0.865	0.865	0.863	0.859	0.846	0.813	0.771	0.751
german	0.883	0.883	0.882	0.881	0.880	0.878	0.876	0.871	0.866	0.855	0.832
heart	0.904	0.904	0.904	0.904	0.901	0.897	0.887	0.844	0.784	0.756	0.741
ionosphere	0.981	0.981	0.980	0.980	0.979	0.978	0.977	0.976	0.974	0.970	0.968
liver disorder	0.807	0.807	0.807	0.806	0.803	0.794	0.773	0.721	0.669	0.635	0.622
sonar	0.821	0.821	0.821	0.820	0.817	0.808	0.779	0.715	0.669	0.655	0.645
splice	0.833	0.833	0.827	0.791	0.729	0.683	0.654	0.639	0.631	0.627	0.622
w1a	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.996

					Distant Diversity						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.152	0.166	0.167	0.169	0.170	0.170	0.171	0.171	0.171	0.171	0.172
australian	0.207	0.214	0.220	0.228	0.238	0.251	0.268	0.285	0.292	0.294	0.295
breast cancer	0.055	0.073	0.073	0.074	0.075	0.076	0.077	0.077	0.077	0.078	0.078
diabetes	0.216	0.225	0.225	0.226	0.227	0.229	0.232	0.239	0.249	0.257	0.258
german	0.249	0.251	0.254	0.256	0.258	0.260	0.262	0.265	0.266	0.268	0.269
heart	0.274	0.274	0.274	0.275	0.280	0.285	0.293	0.315	0.334	0.339	0.341
ionosphere	0.083	0.103	0.104	0.106	0.107	0.109	0.109	0.110	0.110	0.111	0.111
liver disorder	0.269	0.269	0.270	0.272	0.277	0.287	0.305	0.332	0.351	0.357	0.358
sonar	0.204	0.205	0.205	0.207	0.213	0.224	0.248	0.282	0.297	0.300	0.300
splice	0.084	0.098	0.129	0.230	0.348	0.405	0.429	0.438	0.441	0.441	0.441
w1a	0.004	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007

[a] The column headers are for w, which denotes the weight on local accuracy.

Table A.4: Performance, local accuracy, and distant diversity values when $k = 25$

					Performance						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.765	0.800	0.800	0.800	0.800	0.801	0.801	0.801	0.801	0.803	0.801
australian	0.845	0.852	0.852	0.853	0.849	0.850	0.850	0.844	0.844	0.843	0.853
breast cancer	0.977	0.973	0.973	0.973	0.972	0.973	0.973	0.973	0.974	0.977	0.974
diabetes	0.765	0.768	0.768	0.768	0.767	0.770	0.767	0.760	0.759	0.757	0.747
german	0.714	0.723	0.723	0.723	0.725	0.725	0.726	0.726	0.725	0.728	0.725
heart	0.811	0.807	0.807	0.805	0.807	0.807	0.809	0.803	0.796	0.822	0.805
ionosphere	0.933	0.936	0.936	0.936	0.936	0.936	0.937	0.936	0.936	0.933	0.932
liver disorder	0.646	0.701	0.700	0.700	0.704	0.703	0.714	0.696	0.675	0.680	0.675
sonar	0.788	0.854	0.854	0.854	0.858	0.856	0.851	0.817	0.796	0.789	0.786
splice	0.566	0.746	0.746	0.746	0.746	0.745	0.743	0.741	0.737	0.714	0.668
w1a	0.971	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.973	0.971
					Local Accuracy						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.895	0.895	0.895	0.894	0.894	0.893	0.893	0.892	0.892	0.890	0.885
australian	0.916	0.916	0.915	0.911	0.904	0.890	0.864	0.838	0.823	0.814	0.807
breast cancer	0.997	0.997	0.996	0.996	0.995	0.994	0.993	0.992	0.991	0.988	0.986
diabetes	0.856	0.856	0.855	0.855	0.854	0.852	0.847	0.834	0.801	0.767	0.753
german	0.865	0.865	0.865	0.864	0.862	0.860	0.858	0.854	0.847	0.835	0.814
heart	0.876	0.876	0.876	0.876	0.875	0.871	0.854	0.796	0.750	0.732	0.724
ionosphere	0.965	0.965	0.965	0.965	0.964	0.963	0.961	0.958	0.954	0.950	0.947
liver disorder	0.788	0.788	0.787	0.787	0.782	0.770	0.737	0.684	0.648	0.629	0.618
sonar	0.807	0.807	0.806	0.804	0.800	0.784	0.739	0.686	0.656	0.645	0.640
splice	0.829	0.829	0.823	0.787	0.721	0.671	0.639	0.622	0.613	0.610	0.605
w1a	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997	0.997
					Distant Diversity						
Dataset / w[a]	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0
a1a	0.154	0.167	0.169	0.170	0.171	0.172	0.172	0.173	0.173	0.173	0.173
australian	0.204	0.209	0.215	0.225	0.240	0.257	0.277	0.291	0.297	0.298	0.299
breast cancer	0.058	0.074	0.074	0.076	0.077	0.078	0.080	0.080	0.080	0.081	0.081
diabetes	0.220	0.228	0.228	0.229	0.231	0.233	0.237	0.244	0.255	0.261	0.262
german	0.253	0.253	0.255	0.258	0.261	0.263	0.265	0.267	0.270	0.271	0.273
heart	0.259	0.259	0.260	0.260	0.262	0.267	0.280	0.311	0.326	0.330	0.330
ionosphere	0.085	0.103	0.104	0.105	0.106	0.107	0.109	0.111	0.112	0.113	0.113
liver disorder	0.264	0.264	0.265	0.267	0.275	0.289	0.316	0.343	0.357	0.360	0.360
sonar	0.153	0.154	0.155	0.161	0.169	0.187	0.224	0.251	0.262	0.265	0.265
splice	0.087	0.098	0.127	0.229	0.353	0.416	0.443	0.452	0.455	0.455	0.456
w1a	0.005	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007

[a] The column headers are for w, which denotes the weight on local accuracy.

Table A.5: Comparison of distance measures for $w = 0.2, 0.4, 0.6, 0.8$

$w = 0.2$

Comparison / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
PTM vs. PTMA	2.3673	3.8931	3.8733	3.9809	4.1607	4.0419	4.1189	3.8901	3.9838	3.4197	3.194	3.5856	3.3759
PTM vs. TM	4.7835	1.8536	0.774	0.59681	1.2145	0.44808	0.28167	0.4828	0.80592	0.67113	0.02129	0.76282	0.36433
PTM vs. OTM	0.86257	1.0892	0.36871	0.86618	0.7488	0.60768	0.98834	1.3401	1.4562	0.88326	0.5587	1.0768	0.86631
PTM vs. ED	1.5776	1.2495	0.89308	0.24009	0.93385	0.68366	0.13014	0.35592	0.17661	0.003805	-0.38692	0.29034	-0.40215
PTMA vs. TM	3.0957	-1.2051	-2.1796	-2.5325	-2.1304	-2.741	-3.0096	-2.6616	-2.4705	-2.1046	-2.5227	-2.1698	-2.3594
PTMA vs. OTM	-0.97053	-2.0897	-2.6107	-2.2433	-2.6326	-2.6008	-2.3376	-1.8525	-1.9042	-1.9034	-2.0343	-1.9256	-1.9161
PTMA vs. ED	-0.099236	-1.6487	-1.9912	-2.7456	-2.3264	-2.4206	-3.058	-2.6276	-2.9137	-2.5478	-2.7131	-2.4232	-2.9371

$w = 0.4$

Comparison / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
PTM vs. PTMA	2.7818	3.849	3.9106	3.9829	4.1062	4.1409	4.0587	3.9424	3.7684	4.1506	3.8958	3.5349	3.5205
PTM vs. TM	4.8187	2.1359	1.0336	0.61969	0.5617	0.91107	0.82839	0.89788	0.74174	1.0213	1.1503	0.70567	1.2096
PTM vs. OTM	1.0894	1.1589	0.31434	0.80448	0.62402	1.1169	1.3514	1.3334	1.1362	1.5963	1.7631	1.4847	1.8821
PTM vs. ED	1.6984	1.6683	1.5858	0.96201	1.0834	0.92544	0.6827	0.22604	0.14505	0.8305	0.52138	0.2134	0.64611
PTMA vs. TM	2.6189	-0.94086	-1.823	-2.4826	-2.6535	-2.3684	-2.3941	-2.2719	-2.2592	-2.396	-2.0654	-2.1763	-1.6512
PTMA vs. OTM	-1.2051	-2.0242	-2.6439	-2.3724	-2.6443	-2.189	-1.9391	-1.897	-1.928	-1.8657	-1.5071	-1.4633	-1.0424
PTMA vs. ED	-0.47034	-1.3211	-1.4016	-2.1427	-2.1491	-2.3117	-2.4875	-2.8413	-2.7455	-2.4518	-2.5582	-2.4934	-2.078

$w = 0.6$

Comparison / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
PTM vs. PTMA	1.9852	3.9203	3.8098	4.2092	4.2829	4.3357	4.2267	3.9949	3.6655	3.9989	3.7837	3.7846	3.5735
PTM vs. TM	4.3856	2.2308	0.91664	0.92898	0.82666	1.0533	0.75789	0.73863	0.48836	0.98941	0.79507	0.6214	0.94473
PTM vs. OTM	0.70175	1.1455	0.27267	1.0386	0.7136	1.1222	1.3342	0.99489	0.92501	1.3644	1.3149	1.3994	1.7114
PTM vs. ED	1.5985	1.8935	1.5387	1.2975	1.2761	1.2255	1.1454	0.36548	0.32943	0.5555	0.43039	0.12854	0.46398
PTMA vs. TM	3.1207	-0.88639	-1.8775	-2.3926	-2.5538	-2.3819	-2.6257	-2.4235	-2.3972	-2.2885	-2.2778	-2.4509	-1.9487
PTMA vs. OTM	-0.70263	-2.1057	-2.6046	-2.3456	-2.7088	-2.351	-2.1048	-2.2419	-2.0493	-1.9629	-1.807	-1.7468	-1.2727
PTMA vs. ED	0.37648	-1.214	-1.3978	-2.0284	-2.1276	-2.2279	-2.2638	-2.7487	-2.5096	-2.6397	-2.5552	-2.833	-2.3274

$w = 0.8$

Comparison / k[a]	1	3	5	7	9	11	13	15	17	19	21	23	25
PTM vs. PTMA	2.8646	3.9767	3.8856	4.2861	4.3614	4.4151	4.2898	4.1143	3.8685	4.1629	3.7975	3.8512	3.6528
PTM vs. TM	4.6781	2.1926	0.87855	0.93435	0.9303	1.0789	0.85039	0.76448	0.51924	1.0651	0.76585	0.73394	0.85172
PTM vs. OTM	0.86485	1.1736	0.2589	1.0464	0.75047	1.1241	1.359	0.97147	0.91621	1.4378	1.1993	1.3416	1.5452
PTM vs. ED	1.51	1.995	1.7813	1.2928	1.4016	1.2483	1.3869	0.47854	0.50075	0.7741	0.40964	0.27124	0.55767
PTMA vs. TM	2.5029	-0.95135	-1.9666	-2.4523	-2.4884	-2.4076	-2.5495	-2.4962	-2.5283	-2.3144	-2.2743	-2.3778	-2.0836
PTMA vs. OTM	-1.4536	-2.0942	-2.6602	-2.3899	-2.7146	-2.3887	-2.102	-2.3646	-2.2154	-2.0048	-1.8993	-1.8476	-1.4686
PTMA vs. ED	-0.67873	-1.1314	-1.226	-2.1026	-2.05	-2.2599	-2.0553	-2.7671	-2.515	-2.5393	-2.5591	-2.7461	-2.2927

[a] The term k denotes the neighborhood size.

Table A.6: Pairwise t-test: Performance comparison of distant diversity vs static step

w[a]/ k[b]	1	3	5	7	9	11	13	15	17	19	21	23	25
1	(0,5,6)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)	(0,7,4)
0.9	(0,8,3)	(0,9,2)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,11,0)	(1,9,1)	(2,8,1)	(1,9,1)	(1,9,1)	(1,9,1)
0.8	(0,7,4)	(0,9,2)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,11,0)	(1,9,1)	(2,8,1)	(1,9,1)	(1,9,1)	(1,9,1)
0.7	(0,7,4)	(0,8,3)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,11,0)	(1,9,1)	(2,8,1)	(1,9,1)	(1,9,1)	(1,9,1)
0.6	(0,6,5)	(0,8,3)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,11,0)	(1,9,1)	(2,8,1)	(1,9,1)	(1,9,1)	(2,8,1)
0.5	(0,8,3)	(0,8,3)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,11,0)	(1,9,1)	(1,9,1)	(2,8,1)	(1,9,1)	(1,9,1)
0.4	(0,7,4)	(0,8,3)	(1,10,0)	(0,11,0)	(1,10,0)	(1,9,1)	(1,10,0)	(1,10,0)	(1,9,1)	(2,8,1)	(0,10,1)	(1,9,1)	(1,9,1)
0.3	(0,8,3)	(1,7,3)	(1,10,0)	(0,11,0)	(0,11,0)	(1,10,0)	(1,10,0)	(1,10,0)	(0,10,1)	(1,9,1)	(0,10,1)	(1,9,1)	(1,9,1)
0.2	(0,8,3)	(0,9,2)	(0,10,1)	(0,11,0)	(0,11,0)	(0,11,0)	(0,11,0)	(0,11,0)	(0,10,1)	(1,9,1)	(1,9,1)	(0,10,1)	(0,10,1)
0.1	(0,8,3)	(0,9,2)	(0,10,1)	(0,10,1)	(0,11,0)	(0,11,0)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)
0	(0,8,3)	(0,10,1)	(0,9,2)	(1,8,2)	(1,8,2)	(0,9,2)	(0,9,2)	(0,9,2)	(1,8,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)

[a] The term w denotes the weight on local accuracy.

[b] The term k denotes the neighborhood size.

Table A.7: Distant diversity and local accuracy vs KNORA_Eliminate

w^a/ k^b	1	3	5	7	9	11	13	15	17	19	21	23	25
1	(1,7,3)	(2,7,2)	(2,6,3)	(2,8,1)	(3,7,1)	(4,6,1)	(6,4,1)	(8,3,0)	(8,3,0)	(8,3,0)	(8,3,0)	(8,3,0)	(10,1,0)
0.9	(0,10,1)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.8	(0,10,1)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.7	(0,11,0)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.6	(0,8,3)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.5	(0,8,3)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.4	(0,8,3)	(5,6,0)	(5,5,1)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.3	(0,9,2)	(5,6,0)	(5,6,0)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.2	(0,9,2)	(5,6,0)	(5,6,0)	(8,3,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0.1	(0,9,2)	(5,6,0)	(5,6,0)	(7,4,0)	(8,3,0)	(10,1,0)	(9,2,0)	(9,2,0)	(10,1,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)
0	(0,10,1)	(4,7,0)	(5,6,0)	(6,5,0)	(6,5,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(9,2,0)	(10,1,0)

[a] The term w denotes the weight on local accuracy.

[b] The term k denotes the neighborhood size.

Table A.8: Distant diversity vs KNORA_UNION

w[a]/ k[b]	1	3	5	7	9	11	13	15	17	19	21	23	25
1	(1,8,2)	(0,8,3)	(0,8,3)	(0,8,3)	(0,8,3)	(0,8,3)	(0,8,3)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)
0.9	(0,11,0)	(1,8,2)	(3,7,1)	(2,8,1)	(2,9,0)	(3,8,0)	(3,8,0)	(2,9,0)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)
0.8	(0,11,0)	(1,8,2)	(3,7,1)	(2,8,1)	(2,9,0)	(3,8,0)	(3,8,0)	(2,9,0)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)
0.7	(0,11,0)	(1,8,2)	(3,7,1)	(2,8,1)	(2,9,0)	(3,8,0)	(3,8,0)	(2,9,0)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)
0.6	(0,9,2)	(1,8,2)	(3,7,1)	(2,8,1)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)
0.5	(0,9,2)	(1,8,2)	(3,7,1)	(3,7,1)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)
0.4	(0,9,2)	(2,7,2)	(3,7,1)	(3,7,1)	(2,9,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)
0.3	(0,10,1)	(2,8,1)	(2,8,1)	(3,7,1)	(2,8,1)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)
0.2	(0,10,1)	(1,9,1)	(2,7,2)	(3,8,0)	(3,7,1)	(3,7,1)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(3,8,0)	(2,9,0)
0.1	(0,10,1)	(1,9,1)	(2,7,2)	(3,7,1)	(3,8,0)	(3,7,1)	(3,6,2)	(3,8,0)	(3,8,0)	(3,8,0)	(3,7,1)	(3,7,1)	(3,8,0)
0	(0,11,0)	(1,9,1)	(2,7,2)	(3,6,2)	(3,7,1)	(3,6,2)	(3,6,2)	(3,8,0)	(4,6,1)	(3,7,1)	(3,6,2)	(3,7,1)	(3,7,1)

[a] The term w denotes the weight on local accuracy.

[b] The term k denotes the neighborhood size.

Table A.9: Weighted Distant Diversity and Weighted Local Accuracy vs Static Step

w[a]/ k[b]	1	3	5	7	9	11	13	15	17	19	21	23	25
1	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)	(1,8,2)
0.9	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)
0.8	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)
0.7	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)
0.6	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)	(2,8,1)
0.5	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)
0.4	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)	(1,9,1)
0.3	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)
0.2	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)
0.1	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)	(0,10,1)
0	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)	(0,9,2)

[a] The term w denotes the weight on local accuracy.

[b] The term k denotes the neighborhood size.

Table A.10: Pairwise t-test: Weighted distant diversity and weighted local accuracy vs. static benchmark methods

w[a]	Best Classifier	Best 25	Single SVM	Best Ens
1.0	(8,3,0)	(4,7,0)	(10,0,1)	(2,9,0
0.9	(11,0,0)	(7,4,0)	(8,3,0)	(6,5,0))
0.8	(11,0,0)	(8,3,0)	(8,3,0)	(6,5,0)
0.7	(11,0,0)	(7,4,0)	(8,3,0)	(6,5,0)
0.6	(11,0,0)	(7,4,0)	(9,2,0)	(6,5,0)
0.5	(11,0,0)	(7,4,0)	(7,4,0)	(5,6,0)
0.4	(10,1,0)	(8,3,0)	(9,1,1)	(4,7,0)
0.3	(11,0,0)	(8,3,0)	(9,2,0)	(5,6,0)
0.2	(11,0,0)	(8,3,0)	(8,3,0)	(4,7,0)
0.1	(11,0,0)	(7,4,0)	(7,3,1)	(4,7,0)
0.0	(11,0,0)	(8,3,0)	(7,4,0)	(4,7,0)

[a] The term w denotes the weight on weighted local accuracy.

REFERENCES

[1] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

[2] Christopher M Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[3] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[4] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] Gavin Brown. An information theoretic perspective on multiple classifier systems. In *Multiple Classifier Systems*, pages 344–353. Springer, 2009.

[6] Gavin Brown and Ludmila I. Kuncheva. “good” and “bad” diversity in majority vote ensembles. In Neamat Gayar, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems*, volume 5997 of *Lecture Notes in Computer Science*, pages 124–133. Springer Berlin Heidelberg, 2010.

[7] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: A survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[8] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[9] Paulo R. Cavalin, Robert Sabourin, and Ching Y. Suen. Dynamic selection of ensembles of classifiers using contextual information. In Neamat Gayar, Josef Kittler, and Fabio Roli, editors, *Multiple Classifier Systems*, volume 5997 of *Lecture Notes in Computer Science*, pages 145–154. Springer Berlin Heidelberg, 2010.

[10] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[11] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[12] Pádraig Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. In Ramon López de Mántaras and Enric Plaza, editors, *Machine Learning: ECML 2000*, volume 1810 of *Lecture Notes in Computer Science*, pages 109–116. Springer Berlin Heidelberg, 2000.

[13] Diogo F de Oliveira, Anne MP Canuto, and Marcilio CP de Souto. Use of multi-objective genetic algorithms to investigate the diversity/accuracy dilemma in heterogeneous ensembles. In *International Joint Conference on Neural Networks*, IJCNN'09, pages 2339–2346. IEEE, 2009.

[14] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[15] Luca Didaci and Giorgio Giacinto. Dynamic classifier selection by adaptive K-nearest-neighbourhood rule. In *Multiple Classifier Systems*, pages 174–183. Springer, 2004.

[16] Luca Didaci, Giorgio Giacinto, Fabio Roli, and Gian Luca Marcialis. A study on the performances of dynamic classifier selection based on local accuracy estimation. *Pattern Recognition*, 38(11):2188–2191, 2005.

[17] Thomas G Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.

[18] Eulanda M Dos Santos, Robert Sabourin, and Patrick Maupin. Single and multi-objective genetic algorithms for the selection of ensemble of classifiers. In *International Joint Conference on Neural Networks*, IJCNN'06, pages 3070–3077. IEEE, 2006.

[19] Eulanda M Dos Santos, Robert Sabourin, and Patrick Maupin. Ambiguity-guided dynamic selection of ensemble of classifiers. In *Proceedings of International Conference on Information Fusion*, 2007.

[20] Eulanda M Dos Santos, Robert Sabourin, and Patrick Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern Recognition*, 41(10):2993–3009, 2008.

[21] Robert PW Duin. A note on comparing classifiers. *Pattern Recognition Letters*, 17(5):529–536, 1996.

[22] Saso Džeroski and Bernard Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54(3):255–273, 2004.

[23] Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of inter-rater agreement. *Statistical Methods for Rates and Proportions*, 2:212–236, 1981.

[24] Yoav Freund and Robert E Schapire. Experiments with a new boosting algorithm. In *International Workshop on Machine Learning*, volume 96, pages 148–156. Morgan Kaufmann, 1996.

[25] Qiang Fu, Shang-Xu Hu, and Sheng-Ying Zhao. A pso-based approach for neural network ensemble. *Journal of Zhejiang University (Engineering Science)*, 38(12):1596–1600, 2004.

[26] Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9):699–707, 2001.

[27] Giorgio Giacinto and Fabio Roli. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, 34(9):1879–1882, 2001.

[28] Eui-Hong Sam Han, George Karypis, and Vipin Kumar. *Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification*. Springer, 2001.

[29] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(10):993–1001, 1990.

[30] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *International Joint Conference on Neural Networks*, IJCNN'89, pages 593–605. IEEE, 1989.

[31] David Heckerman. Bayesian networks for data mining. *Data Mining and Knowledge Discovery*, 1(1):79–119, 1997.

[32] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[33] Yu-Chi Ho and David L Pepyne. Simple explanation of the no-free-lunch theorem and its implications. *Journal of Optimization Theory and Applications*, 115(3):549–570, 2002.

[34] Kuo-Wei Hsu and Jaideep Srivastava. Relationship between diversity and correlation in multi-classifier systems. In Mohammed J. Zaki, Jeffrey Xu Yu, B. Ravindran, and Vikram Pudi, editors, *Advances in Knowledge Discovery and Data Mining*, volume 6119 of *Lecture Notes in Computer Science*, pages 500–506. Springer Berlin Heidelberg, 2010.

[35] Marcelo N Kapp, Robert Sabourin, and Patrick Maupin. An empirical study on diversity measures and margin theory for ensembles of classifiers. In *Tenth International Conference on Information Fusion*, pages 1–8. IEEE, 2007.

[36] Ross D. King, Cao Feng, and Alistair Sutherland. Statlog: Comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence an International Journal*, 9(3):289–333, 1995.

[37] Albert HR Ko, Robert Sabourin, and Alceu Souza Britto Jr. From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, 41(5):1718–1731, 2008.

[38] Ron Kohavi and David H Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Machine Learning: Proceedings of the Thirteenth International*, pages 275–283, 1996.

[39] Ludmila I. Kuncheva. That elusive diversity in classifier ensembles. In Francisco Jos Perales, Aurlio J.C. Campilho, Nicols Prez Blanca, and Alberto Sanfeliu, editors, *Pattern Recognition and Image Analysis*, volume 2652 of *Lecture Notes in Computer Science*, pages 1126–1138. Springer Berlin Heidelberg, 2003.

[40] Ludmila I Kuncheva, James C Bezdek, and Robert PW Duin. Decision templates for multiple classifier fusion: An experimental comparison. *Pattern Recognition*, 34(2):299–314, 2001.

[41] Ludmila I Kuncheva and Christopher J Whitaker. Feature subsets for classifier combination: An enumerative experiment. In Josef Kittler and Fabio Roli, editors, *Multiple Classifier Systems*, volume 2096 of *Lecture Notes in Computer Science*, pages 228–237. Springer, 2001.

[42] Ludmila I. Kuncheva and Christopher J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.

[43] Pat Langley, Wayne Iba, and Kevin Thompson. An analysis of bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, volume 90, pages 223–228, 1992.

[44] Beitao Li and Kingshy Goh. Confidence-based dynamic ensemble for image annotation and semantics discovery. In *Proceedings of the Eleventh ACM International Conference on Multimedia*, pages 195–206. ACM, 2003.

[45] Leijun Li, Bo Zou, Qinghua Hu, Xiangqian Wu, and Daren Yu. Dynamic classifier ensemble using classification confidence. *Neurocomputing*, 2012.

[46] Leijun Li, Bo Zou, Qinghua Hu, Xiangqian Wu, and Daren Yu. Dynamic classifier ensemble using classification confidence. *Neurocomputing*, 99:581–591, 2013.

[47] Yong Liu and Xin Yao. Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399–1404, 1999.

[48] Bao-Liang Lu and Masami Ito. Task decomposition and module combination based on class relations: A modular neural network for pattern classification. *Neural Networks, IEEE Transactions on*, 10(5):1244–1256, 1999.

[49] Shasha Mao, LC Jiao, Lin Xiong, and Shuiping Gou. Greedy optimization classifiers ensemble based on diversity. *Pattern Recognition*, 44(6):1245–1261, 2011.

[50] Dragos D Margineantu and Thomas G Dietterich. Pruning adaptive boosting. In *International Workshop on Machine Learning*, volume 97, pages 211–218, 1997.

[51] Prem Melville and Raymond J Mooney. Creating diversity in ensembles using artificial data. *Information Fusion*, 6(1):99–111, 2005.

[52] Eitan Menahem, Lior Rokach, and Yuval Elovici. Troika–an improved stacking schema for classification tasks. *Information Sciences*, 179(24):4097–4122, 2009.

[53] Christopher J Merz. Using correspondence analysis to combine classifiers. *Machine Learning*, 36(1-2):33–58, 1999.

[54] Tadahiko Murata and Hisao Ishibuchi. MOGA: Multi-objective genetic algorithms. In *IEEE International Conference on Evolutionary Computation*, volume 1, pages 289–294. IEEE, 1995.

[55] David W Opitz. Feature selection for ensembles. In *AAAI/IAAI*, pages 379–384, 1999.

[56] David W Opitz and Jude W Shavlik. Actively searching for an effective neural network ensemble. *Connection Science*, 8(3-4):337–354, 1996.

[57] Derek Partridge and W Krzanowski. Software diversity: Practical statistics for its measurement and exploitation. *Information and Software Technology*, 39(10):707–717, 1997.

[58] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. volume 10, pages 61–74. Cambridge, MA, 1999.

[59] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[60] J Ross Quinlan. Bagging, boosting, and c4. 5. In *AAAI/IAAI, Vol. 1*, pages 725–730, 1996.

[61] John Ross Quinlan. *C4. 5: Programs for Machine Learning*, volume 1. Morgan Kaufmann, 1993.

[62] Dymitr Ruta and Bogdan Gabrys. Analysis of the correlation between majority voting error and the diversity measures in multiple classifier systems. In *Proceedings of the Fourth Internation Symposium on Soft Computing*, number 1824-025 in SOCO/ISFI 2001. ICSC-NAISO Academic Press, 2001.

[63] Steven L Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3):317–328, 1997.

[64] Bernhard Schölkopf and Alexander J Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT press, 2002.

[65] Alexander K Seewald. Towards a theoretical framework for ensemble classification. In *IJCAI*, pages 1443–1444. Citeseer, 2003.

[66] Marina Skurichina and Robert PW Duin. Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis & Applications*, 5(2):121–135, 2002.

[67] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining.* Pearson Education, Inc., 2006.

[68] E Ke Tang, Ponnuthurai N Suganthan, and Xin Yao. An analysis of diversity measures. *Machine Learning*, 65(1):247–271, 2006.

[69] Kai Ming Ting and Ian H Witten. Stacking bagged and dagged models. In *ICML*, pages 367–375, 1997.

[70] Kagan Tumer and Joydeep Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8:385–404, 1996.

[71] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* springer, 1995.

[72] Leila Maria Vriesmann, Alceu De Souza Britto Jr, Luiz Eduardo Soares De Oliveira, Robert Sabourin, and Albert Houng-Ren Ko. Improving a dynamic ensemble selection method based on oracle information. *International Journal of Innovative Computing and Applications*, 4(3):184–200, 2012.

[73] Tomasz Woloszynski and Marek Kurzynski. A probabilistic model of classifier competence for dynamic ensemble selection. *Pattern Recognition*, 44(10):2656–2668, 2011.

[74] David H Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

[75] Kevin Woods, W Philip Kegelmeyer Jr, and Kevin Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):405–410, 1997.

[76] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, and S Yu Philip. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.

[77] Gabriele Zenobi and Pdraig Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In Luc Raedt and Peter Flach, editors, *Machine Learning: ECML 2001*, volume 2167 of *Lecture Notes in Computer Science*, pages 576–587. Springer Berlin Heidelberg, 2001.

[78] Yi Zhang, Samuel Burer, and W Nick Street. Ensemble pruning via semi-definite programming. *The Journal of Machine Learning Research*, 7:1315–1338, 2006.

[79] Zhi-Hua Zhou, Jian-Xin Wu, Yuan Jiang, and Shi-Fu Chen. Genetic algorithm based selective neural network ensemble. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, volume 1 of *IJCAI-01*, page 797. Citeseer, 2001.

[80] Xingquan Zhu, Xindong Wu, and Ying Yang. Dynamic classifier selection for effective mining from noisy data streams. In *Fourth IEEE International Conference on Data Mining*, ICDM'04, pages 305–312. IEEE, 2004.