May 2020

A Statistical Model for the Prediction of the Taxi Trip Time in the City of Chicago

Frank Bitter
University of Wisconsin-Milwaukee

A STATISTICAL MODEL FOR THE PREDICTION OF THE TAXI TRIP TIME IN THE CITY OF CHICAGO

by

Frank Bitter

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Mathematics

at

The University of Wisconsin-Milwaukee

May 2020

ABSTRACT

A STATISTICAL MODEL FOR THE PREDICTION OF TAXI TRIP DURATION IN THE CITY OF CHICAGO

by

Frank Bitter

The University of Wisconsin-Milwaukee, 2020
Under the Supervision of Professor Daniel Gervini

This thesis addresses the problem of prediction of taxi trip duration for any given day, time, pickup point and dropoff point. Data on taxi trips from the Chicago Data Portal is used. The main idea of the model is to cluster similar trips together and use the mean duration of all those clustered taxi trips to predict the duration of a new taxi trip in that cluster. Furthermore, for a possible additional reduction of prediction error, estimators from different days which are not significantly different from each other are pooled together. It is shown that this procedure improves prediction error.

TABLE OF CONTENTS

1 Introduction **1**

2 Introduction to the data **3**

3 The model for trip duration prediction **4**

3.1 Clustering geodata 4

3.2 Estimation of trip duration 5

3.3 Prediction of the trip time and the prediction error 5

3.4 Pooling estimators together 6

4 Application of the model to the Chicago taxi trip data **8**

4.1 Data splitting and clustering 8

4.2 Computation of estimators and prediction error 13

4.3 Pooling estimators 13

5 Error analysis **15**

5.1 Analysis of prediction error 15

5.2 Analysis of prediction error after pooling 24

6 Conclusions and outlook **30**

7 References **32**

Appendix R-Code **33**

LIST OF FIGURES

1 Pickup points on Mondays 9

2 Dropoff points on Mondays 10

3 The error $E(t, d)$ over time t for Monday, Wednesday, Thursday and Saturday. 15

4 Error $E(t, d)$ over time t for Tuesday, Friday and Sunday 16

5 Error $E(t, d)$ over time t for Tuesday, Friday and Sunday, with values above 1500 eliminated. 17

6 Number of existing $E(i, t, d)$s. 19

7 Boxplot of errors $E(i, t, d)$ on Sunday at time interval [5 am, 5.30 am) . . 20

8 Boxplot of errors $E(i, t, d)$ with outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am) 21

9 Boxplot of errors $E(i, t, d)$ without outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am) . . . 22

10 Boxplots of $E(i, t, d)$s for Friday at time intervals [2 pm, 2.30 pm), [5.30 pm, 6 pm) and [10 pm, 10.30 pm), with outliers. 23

11 Boxplots of $E(i, t, d)$s for Friday at time intervals [2 pm, 2.30 pm), [5.30 pm, 6 pm) and [10 pm, 10.30 pm), without outliers. 23

12 Errors $E(d, k)$ for different days d and pooling thresholds k. 24

13 Ratios $E(d, k)/E(d)$ for several days d and pooling thresholds k. 25

14 Errors $E(t, d, k)$ for Wednesday for different values of k 27

15 Errors $E(t, d, k)$ for Monday for different pooling thresholds k 28

LIST OF TABLES

1 Selected quantiles, maximum and minimum of pickup and dropoff latitudes and longitudes 10

2 Maxima and minima of pickup and dropoff latitudes and longitudes for reduced domains 12

3 Average error $E(d)$ 16

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Prof. Daniel Gervini for giving me the opportunity to work on this interesting project and for and all the fruitful discussions we had together.

Secondly, I would like to thank Prof. Hinow and Prof. Spade for being part of my thesis committee.

Lastly, I would like to thank Prof. Boyd for the great support in these crazy and difficult times. In situations like this it is important to have a leader like her, who helps everybody to navigate through the difficulties everybody faces now.

1 Introduction

The question of how long it takes a taxi to go from a pickup point A to a dropoff location B is an important question for several reasons. Taxi drivers want to know if they can pickup a person at a specific time at a specific place. They also have to manage their trips to minimize the time where they are driving without carrying people. Potential passengers want to estimate how long a taxi trip will take when they schedule a taxi pick-up at a specific time and place. So the question is how to estimate a taxi trip duration from a given pickup place A to a dropoff place B, at a given time t and week day d.

This thesis proposes a model where taxi trip duration is estimated using historic data. To generate this estimator, taxi trips with pickup points and dropoff points near each other are clustered together. Then the average trip duration is computed for every cluster. The prediction error is calculated as the mean absolute deviation between the actual trip durations and the predicted ones. For a possible additional reduction of the error, the method pools together different days for which the averages are not significantly different, hence gaining accuracy by increasing the effective sample size for each cluster.

For the analysis of the model the taxi trip data set of the city of Chicago is used. For estimation the data from year 2018 is used. For the computation of the prediction error, the data from year 2019 is used. The focus of the thesis is the prediction of the trip duration. So the main goal is to minimize prediction error. A question of interest is if pooling data from different clusters together reduces prediction error, and if so, if there exists an optimal pooling that minimizes the error. This leads to several hypotheses:

- The arithmetic average is a good predictor of taxi trip duration
- Pooling data together reduces prediction error
- There exists a pooling that minimize the overall prediction error

The thesis will address these hypotheses in turn. In a first step the error is analyzed without pooling, to get a general sense of the method's quality. In a second step, the error

is analyzed when pooling is implemented. Finally, we explore the possibility of finding an optimal pooling threshold.

The thesis is structured as follows. Chapter 2 gives an introduction to the Chicago taxi trip data. Chapter 3 introduces the general model that will be used for prediction and estimation, as well as the data pooling scheme. This general model can, of course, be applied to other cities. Chapter 4 applies the model to the specific case of the Chicago taxi trip data set; here, general parameters are estimated and computational issues are explained in detail. Chapter 5 analyzes the prediction results, including those with data pooling. Chapter 6 states some conclusions and outlook.

2 Introduction to the data

As data the "taxi trips data set" of the city of Chicago will be considered. The data can be found at the Chicago Data Portal, https://data.cityofchicago.org/ Transportation/Taxi-Trips/wrvz-psew. For each taxi trip in the city, this data set includes the following 23 variables: Trip ID, Taxi ID, Trip Start Timestamp, Trip End Timestamp, Trip Seconds, Trip Miles, Pickup Census Tract, Dropoff Census Tract, Pickup Community Area, Dropoff Community Area, Fare, Tips, Tolls, Extras, Trip Total, Payment Type, Company, Pickup Centroid Latitude, Pickup Centroid Longitude, Pickup Centroid Location, Dropoff Centroid Latitude, Dropoff Centroid Longitude, Dropoff Centroid Location.

For our analysis of the data only the following variables are needed: Trip Start Timestamp, Trip Seconds, Pickup Centroid Latitude, Pickup Centroid Longitude, Dropoff Centroid Latitude, Dropoff Centroid Longitude. The Trip Start Timestamp is recorded in 15-minute intervals. This means that, for example, if a taxi trip starts at 11:10 a.m., the Trip Start Timestamp will be 11:00 a.m.. So the data is already grouped in 15-minute intervals. The variable Trip Seconds is the trip duration, given in seconds. Because of the given trip duration, the Time End Timestamp is not needed. The Pickup Centroid Latitude and the Pickup Centroid Longitude give the starting point of the taxi trip in terms of latitude and longitude. The Dropoff Centroid Latitude and the Dropoff Centroid Longitude give the ending point of the taxi trip in terms of latitude and longitude. Both latitude and longitude variables are given as numeric numbers, so they can be directly used for computation.

The database goes from the year 2013 until 2020. For computing time reasons, only the data for 2018 is used as training data for estimation of the predictors, and the data from 2019 is used as test set to estimate prediction error.

3 The model for trip duration prediction

In this chapter, the model for predicting trip duration from a pickup place A to a dropoff place B is introduced. The main idea is to cluster the data into small clusters and compute the average trip duration based on these clusters. The prediction error is also defined. Then a scheme to pool estimators together, with the goal of reducing prediction error, is introduced.

The model is related to one proposed by Krishnaswamy et al. (2001). However, the main difference between their model and the one proposed in this thesis is that trip distance and path are not considered here. This is an important simplification over the model of Krishnaswamy et al. (2001).

3.1 Clustering geodata

To estimate and predict the trip duration from a pickup place A to a dropoff place B, let $[a, b] \times [c, d]$ the space of possible pickup places and $[e, f] \times [g, h]$ the space of possible dropoff places. Therefore, $[a, b]$ is the range of the latitude and $[c, d]$ is the range of the longitude of the possible pickup places A, while $[e, f]$ is the range of the latitude and $[g, h]$ is the range of the longitude of the possible dropoff places B. Every pickup point $A = (x, y)$ will be considered as a pair $(x, y) \in [a, b] \times [c, d]$, where x represents latitude and y represents longitude. Similarly, every dropoff point is represented as $B = (v, w) \in [e, f] \times [g, h]$. To estimate the trip duration from A to B, similar trips will be clustered. Two trips are similar if their respective pickup places A and their respective dropoff places B are near each other. To find such similar trips, the large rectangles $[a, b] \times [c, d]$ and $[e, f] \times [g, h]$ are divided into small subrectangles $[a_i, a_{i+1}] \times [c_j, c_{j+1}]$ and $[e_k, e_{k+1}] \times [g_l, g_{l+1}]$, respectively, where $a = a_0 < \cdots < a_n = b$, $c = c_0 < \cdots < c_m = d$, $e = e_0 < \cdots < e_n = f$ and $g = g_0 < \cdots < g_m = h$. All trips with pickup places A and dropoff places B lying in the same subrectangle will be clustered together. The size of the subrectangles can be different. In this model it is assumed that the sizes of all subrectangles are equal. After this clustering, all the data is grouped up into data subsets

with specific pickup area i and dropoff area j. The respective subsets are now labeled R_{ij}.

After clustering the data by pickup and dropoff places as explained above, the data is also clustered by time of the day (in intervals of 30 minutes, for example) and day of the week. So these clusters now depend on the time t and the day d, and are denoted by $R_{ij}(t,d)$. It is possible to choose time intervals of different lengths for different days, but in our model all time intervals will have equal lengths. This simplifies the model and makes the comparison of estimators easier.

3.2 Estimation of trip duration

There are several ways to estimate trip duration from a pickup place A to a dropoff place B. In this model, the estimator is the arithmetic mean of the trip durations in each cluster. This estimator is defined as $\mu(i,j,t,d) = \frac{1}{p}\sum_{q=1}^{p} t_q(i,j,t,d)$, where p is the number of taxi trips in cluster $R_{ij}(t,d)$ defined above and $t_q(i,j,t,d)$ is the trip duration. The arithmetic mean is a simple estimator, easy and fast to compute, which is very important for massive data sets like the Chicago trip data.

3.3 Prediction of the trip time and the prediction error

After finding the estimator $\mu(i,j,t,d)$, the prediction error is computed as follows. A test data set of taxi trips is taken (for example, trips corresponding to a different year than the ones used for estimation) and clusters $R_{ij}(t,d)$ are defined analogously as in chapter 3.1. After this clustering, the absolute mean deviation is calculated in the following way: let n be the number of data points in the test cluster $R_{ij}(t,d)$ (which could be zero) and x_k the respective trip durations. Then the prediction error $e(i,j,t,d)$ for $\mu(i,j,t,d)$ is defined as

$$e(i,j,t,d) = \frac{\sum_{k=1}^{n} |x_k - \mu(i,j,t,d)|}{n}$$

and left undefined if $n = 0$. This is the error for a specific pickup area i and a specific pickup area j. The overall error for a specific pickup area i is defined as

$$E(i,t,d) = \frac{\sum_{i=1}^{n_1 \cdot m_1} e(i,j,t,d)}{\widetilde{n_1 \cdot m_1}},$$

where $\widetilde{n_1 * m_1}$ is the number of non-empty test clusters; it is left undefined if $\widetilde{n_1 \cdot m_1} = 0$. The overall error for a specific dropoff area j is defined in a similar way as

$$E(j,t,d) = \frac{\sum_{i=1}^{n_2 \cdot m_2} e(i,j,t,d)}{\widetilde{n_2 \cdot m_2}}.$$

The overall prediction error for time interval t and day d is then defined as

$$E(t,d) = \frac{\sum_{i=1}^{n_2 \cdot m_2} E(i,t,d)}{(n \cdot m)^*},$$

where $(n \cdot m)^*$ is the number of well-defined errors $E(i,t,d)$.

3.4 Pooling estimators together

To pool estimators together, a criterion is needed to decide if estimations for two days (at a given time t) are significantly different or not. The criterion for pooling the data is as follows: let $\mu(i,j,t,d)$ and $\mu(i,j,t,d')$ be two estimators for two different days. Let $\text{se} = \left(\frac{Var(R_{ij}(t,d))}{|R_{ij}(t,d)|} + \frac{Var(R_{ij}(t,d'))}{|R_{ij}(t,d')|}\right)^{1/2}$ be the standard error of the difference of means, where $|R_{ij}(t,d)|$ and $|R_{ij}(t,d')|$ are the number of points in $R_{ij}(t,d)$ and $R_{ij}(t,d')$, respectively. Then two estimators are not significantly different if the standardized difference

$$\text{sd} = \frac{|\mu(i,j,t,d) - \mu(i,j,t,d')|}{\text{se}} \leq k,$$

for k an arbitrarily chosen constant.

If an estimator $\mu(i,j,t,d_1)$ is not significantly different to another estimator $\mu(i,j,t,d_2)$,

then they are pooled together. This means that $\mu(i,j,t,d_1)$ is replaced by

$$\mu^*(i,j,t,d_1) = \frac{\mu(i,j,t,d_1) + \mu(i,j,t,d_2)}{2},$$

which is the average of both estimators. If an estimator $\mu(i,j,t,d_1)$ is not significantly different to several estimators $\mu(i,j,t,d_2)$, $\mu(i,j,t,d_3)$, ..., $\mu(i,j,t,d_n)$ with $n \leq 7$, then the new estimator $\mu^*(i,j,t,d_1)$ is defined as

$$\mu^*(i,j,t,d_1) = \frac{\sum_{k=1}^{n} \mu(i,j,t,d_k)}{n}.$$

This is done sequentially for every day of the week.

After this step, the errors $e^*(i,j,t,d)$, $E^*(i,t,d)$ and $E^*(t,d)$ are recomputed and compared with the old $e^*(i,j,t,d)$, $E^*(i,t,d)$ and $E^*(t,d)$. The question is if the errors are reduced or increased by pooling.

4 Application of the model to the Chicago taxi trip data

In this chapter, the model will be applied to the taxi trip data set of the city of Chicago. For estimation of taxi trip duration the data from year 2018 are used. For calculation of the prediction error the data from year 2019 are used.

4.1 Data splitting and clustering

The whole dataset is ordered by day and time. For estimation all Mondays, Tuesdays, Wednesdays, Thursdays, Fridays, Saturdays and Sundays of the year are pooled together. Then the data is split into time intervals. By the Trip Start Timestamp the data are clustered into 15-minute time intervals. So it is possible to choose as time interval either 15 minutes or 30 minutes or 45 minutes or 60 minutes and so on. A time interval of one hour might be too long, especially during rush hour when demand increases. A 15-minute time interval could lead to too many clusters with too few observations per cluster. So we choose 30-minute time intervals as a reasonable time span to cluster the data.

For the division of the city into subrectangles, a size of these subrectangles is needed. The smaller the subrectangles are, the better is the estimation for a specific pickup point A and a dropoff point B. Too small subrectangles lead to very long computing times and to very small sample sizes per cell. As a compromise we choose subrectangles of size 3×3 city blocks. This size is small enough to provide a reasonably good approximation to the actual pickup and dropoff points, but large enough that computations do not take too long and effective sample sizes are not too small. It is possible to choose different subrectangle sizes for pickup points and dropoff points, but here we choose subrectangles of the same size. The length of such a subrectangle is then 0.003703 and the width is 0.004206.

To find the intervals $[a, b] \times [c, d]$ and $[e, f] \times [g, h]$ of the large rectangles of the city of Chicago, the distribution of the data for the pickup points and the dropoff points are considered in a plot. Because of the large number of data, the plots (Figures 1 and 2)

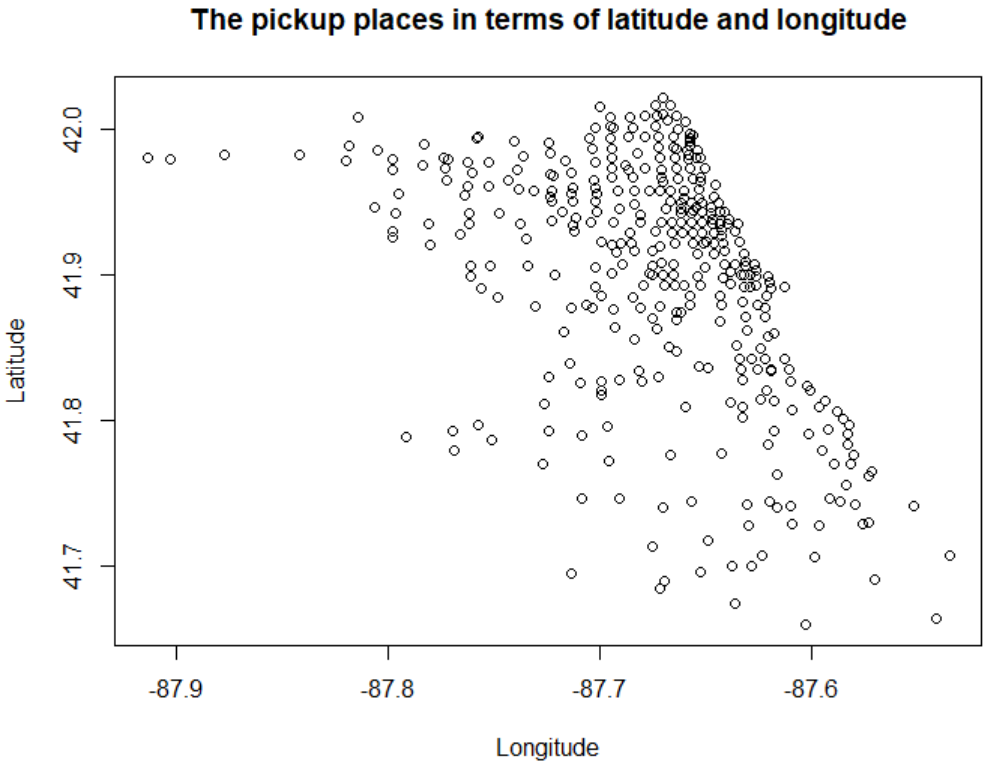


Figure 1: Pickup points on Mondays

only show the pickup points and the dropoff points for Mondays. Also, selected quantiles, maximum and minimum of the distribution of pickup and dropoff latitudes and longitudes are shown in Table 1.

The maxima and minima of the latitude and the longitude of pickup and dropoff points are very similar. The same holds for the quantiles, so it is possible to set $a = e$, $b = f$, $c = g$ and $d = h$. In the two plots it is observed that the data for the pickup locations and for the dropoff locations are concentrated in the upper right quarter of the plots. There are several outliers going down into the lower right quarter and into the upper left quarter. There are no data points in the lower left quarter. If every data point were considered, the interval borders would be the maximum and minimum of the respective latitude and longitude. If the size of the subrectangles is as indicated above, 3×3 city blocks, the number of rectangles for the pickup places would be $\lceil \frac{42.02122-41.65022}{0.004206} \rceil \times \lceil \frac{-87.53139-(-87.91362)}{0.003703} \rceil = 9091$. So the number of estimators for one time interval and one day would be $9091 \times 9091 \approx 85,000,000$. This is not easy to handle computationally. A

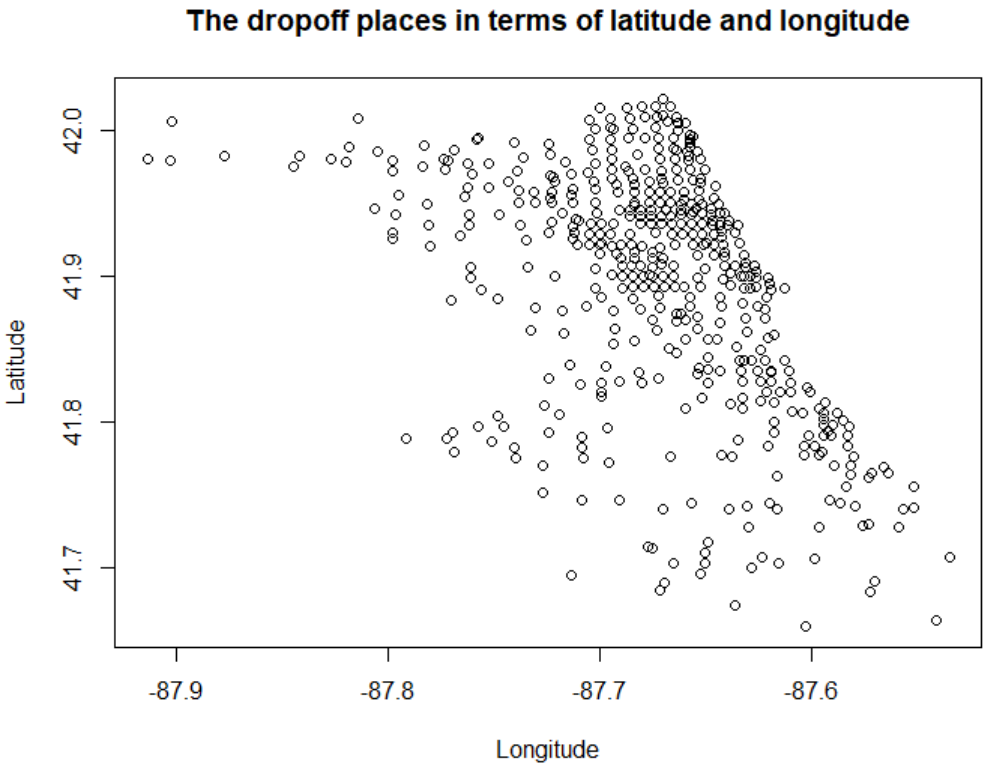
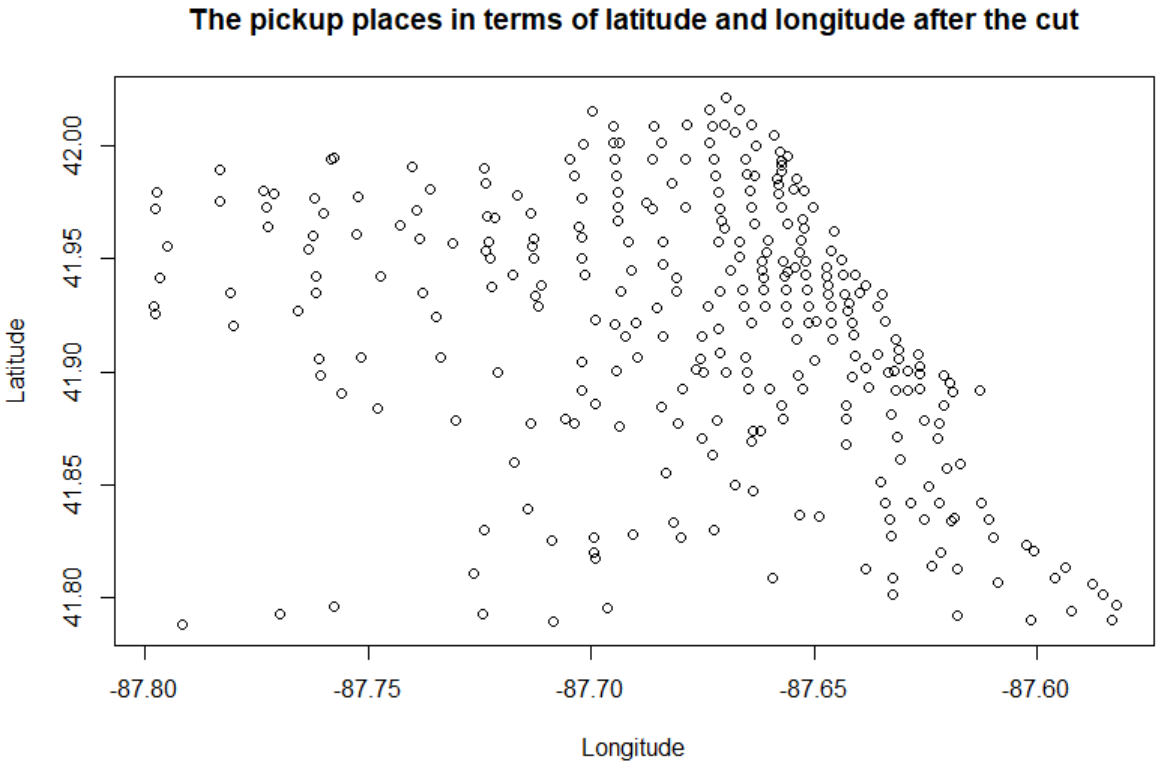


Figure 2: Dropoff points on Mondays

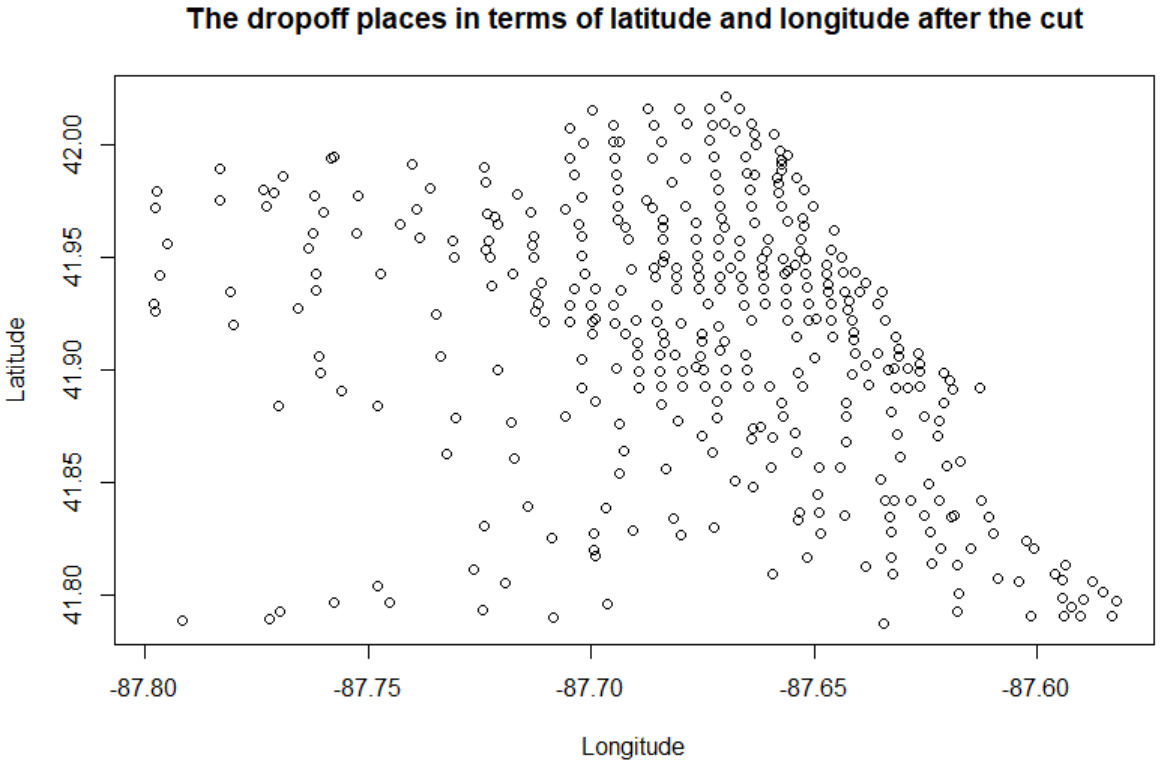
Latitude:	Minimum	1%-quantile	2%-quantile	5%-quantile	Maximum
Pickup places	41.66014	41.786	41.786	41.84925	42.02122
Dropoff places	41.66014	41.77888	41.7929	41.85027	42.02122
Longitude:	Minimum	1%-quantile	5%-quantile	10%-quantile	Maximum
Pickup places	-87.91362	-87.91362	-87.90304	-87.90304	-87.5349
Dropoff places	-87.91362	-87.91362	-87.80403	-87.69501	-87.5349

Table 1: Selected quantiles, maximum and minimum of pickup and dropoff latitudes and longitudes



solution for that problem would be to take shorter intervals $[a, b] \times [c, d]$ and $[e, f] \times [g, h]$. From the quantiles in Table 1 we observe that cutting the data at the 1% quantile of the latitude downsizes the interval of the latitude in both plots by about half. For longitude, a cut could be the 10% quantile for the pickup places and a 5% quantile for the dropoff places. With this cut, the length of the respective interval is reduced by about a third. About 17% of the data are discarded this way, but the computational speed increases considerably. To see what this reduction results in, the new plots of pickup locations and dropoff locations for Mondays are shown in Figures 4.1 and 4.1. In addition, the maxima and minima of the domains are shown in Table 2.

It can be observed, that the structure in the center of the plot before is not really changed. Also, for the new matrix the maxima and minima are very similar to the set borders. As a result of this analysis, we choose the following values as interval borders: $a = e = 41.786$, $b = f = 42.02122$, $c = g = -87.80453$, and $d = h = -87.53139$. The intervals $[a, b]$ and $[e, f]$ are then split into 64 subintervals. The intervals $[c, d]$ and $[g, h]$ are split into 65 subintervals.



Latitude:	Minimum	Maximum
Pickup places:	41.7885	42.02122
Dropoff places:	-87.79803	-87.58237
Longitude :	Minimum	Maximum
Pickup places:	41.78728	42.02122
Dropoff places:	-87.79803	-87.58237

Table 2: Maxima and minima of pickup and dropoff latitudes and longitudes for reduced domains

4.2 Computation of estimators and prediction error

After clustering and splitting the data, the estimator $\mu(i,j,t,d)$ is computed by the formula given in Chapter 3.2. To find out if two estimators are significantly different, the variance and the number of observations is also needed. The variance of the estimator is computed by the empirical variance of the dataset. Those three matrices can be calculated within one function. The values of the estimator $\mu(i,j,t,d)$, the variance and the number of obervations are collected as matrices depending on the number of pickup areas i and the number of dropoff areas j. These matrices are then saved as data-matrices. After finding the estimators $\mu(i,j,t,d)$, there is, for every day d and every time interval t, one matrix for the estimators $\mu(i,j,t,d)$, the variances and the numbers of observations. If for a specific combination (i,j,t,d) no data points exist, the respective estimator $\mu(i,j,t,d)$ is set to NA. If there is only one data point for a given (i,j,t,d), the estimator $\mu(i,j,t,d)$ will then be the trip duration of that datapoint and the variance of $\mu(i,j,t,d)$ will be 0.

After computing the estimators, the errors $e(i,j,t,d)$, $E(i,t,d)$ and $E(t,d)$ are computed using the formula given in Chapter 3.3. The errors $e(i,j,t,d)$ are collected and saved in the same way as the estimators $\mu(i,j,t,d)$. The average error $E(i,t,d)$ are vectors of dimension $64 \times 65 = 4160$. If for a specific (i,t,d) either no estimator $\mu(i,j,t,d)$ or no data points exist, then $e(i,j,t,d)$ is set to NA. If for an error $E(i,t,d)$ no errors $e(i,j,t,d)$ exist, then $E(i,t,d)$ is set to NA.

4.3 Pooling estimators

The pooling of the estimators is done separately for each k in a grid of possible ks. For each k, the pooling is done for each pickup-dropoff pair (i,j) at a time. Then the estimators $\mu(i,j,t,d)$, the variances and the number of observations for the fixed i and j for all time intervals t and days d are considered in three 48×7 matrices. The pooling is carried out as explained in Chapter 3.4, day by day. Note that statistical significance is not transitive: it is possible, for example, that for day $d = d_1$ the estimator $\mu(i,j,t,d_1)$ is pooled together with the estimators $\mu(i,j,t,d_2)$ and $\mu(i,j,t,d_3)$, but for day $d = d_2$ the estimator $\mu(i,j,t,d_2)$ is only pooled with estimator $\mu(i,j,t,d_1)$ and not with $\mu(i,j,t,d_3)$.

Therefore, for each day d the pooled estimator $\mu^*(i,j,t,d)$ is defined as

$$\mu^*(i,j,t,d) = \frac{\mu(i,j,t,d) + \sum_{d'} \mu(i,j,t,d') \cdot \mathbb{1}_{\{\text{sd} \leq k\}}}{n_r},$$

where n_r is the number of estimators that are pooled together. This computation is done for every row and so for every time interval t. The whole computation is then done for every i and j. After this step, the new errors $e^*(i,j,t,d)$, $E^*(i,t,d)$ and $E^*(t,d)$ are computed. The new errors $E^*(t,d)$ are then compared with the old errors $E(t,d)$.

The pooling is applied for several values of k. The values of k can be chosen in several ways. In general the values of k need not be too large, since the sd's are standardized quantities. Values of $k \in \{1,2,4,8,16\}$ seem reasonable.

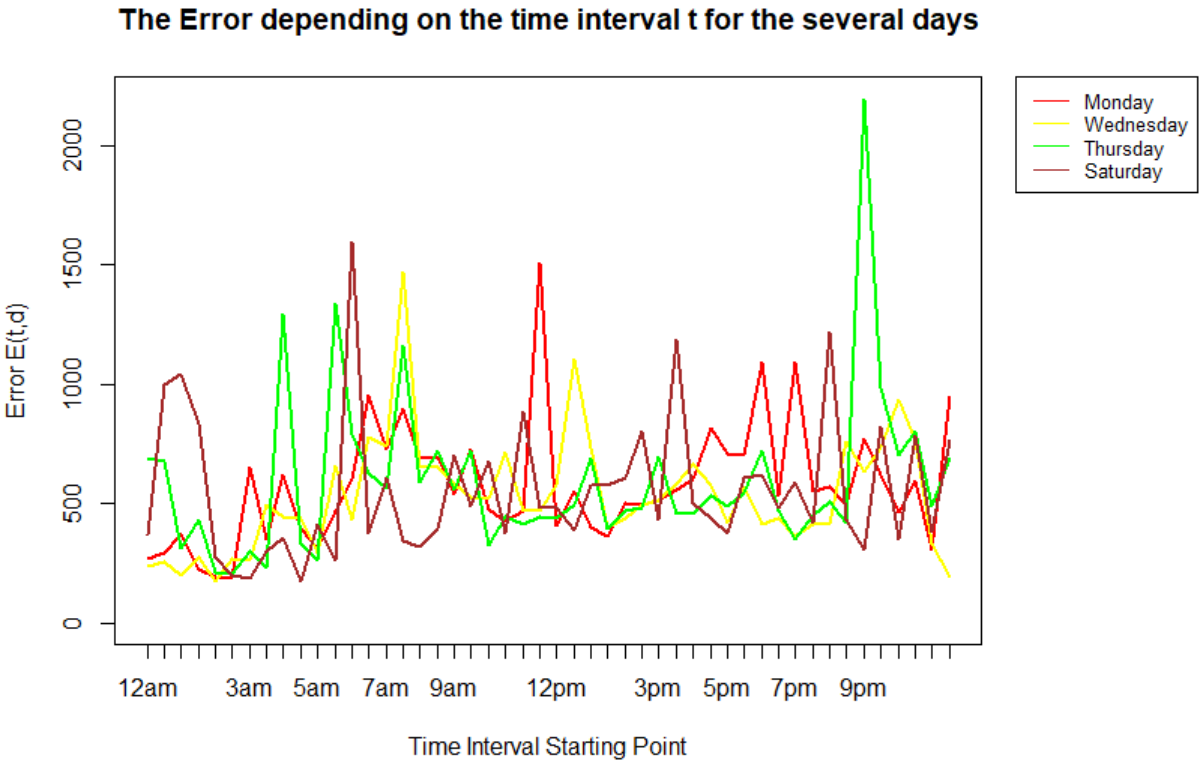


Figure 3: The error $E(t, d)$ over time t for Monday, Wednesday, Thursday and Saturday.

5 Error analysis

In this section the results of applying the model to the taxi trips of the city of Chicago are analyzed.

5.1 Analysis of prediction error

In this subsection the prediction error is analyzed. For this analysis, several graphs of the errors $E(t, d)$ and the errors $E(i, t, d)$ are considered. First, the errors $E(t, d)$ are shown as functions of t for every day d. Because of strong outliers on Tuesday, Friday and Sunday, we show three different plots (Figures 3–5). Also, the average error $E(d) = \frac{\sum_t E(t,d)}{48}$ is shown in Table 3.

There are several observations:

- All plots show some outliers in the error $E(t, d)$. Especially on Friday and Sunday there are errors $E(t, d)$ which are very high. Those errors $E(t, d)$ could be strongly

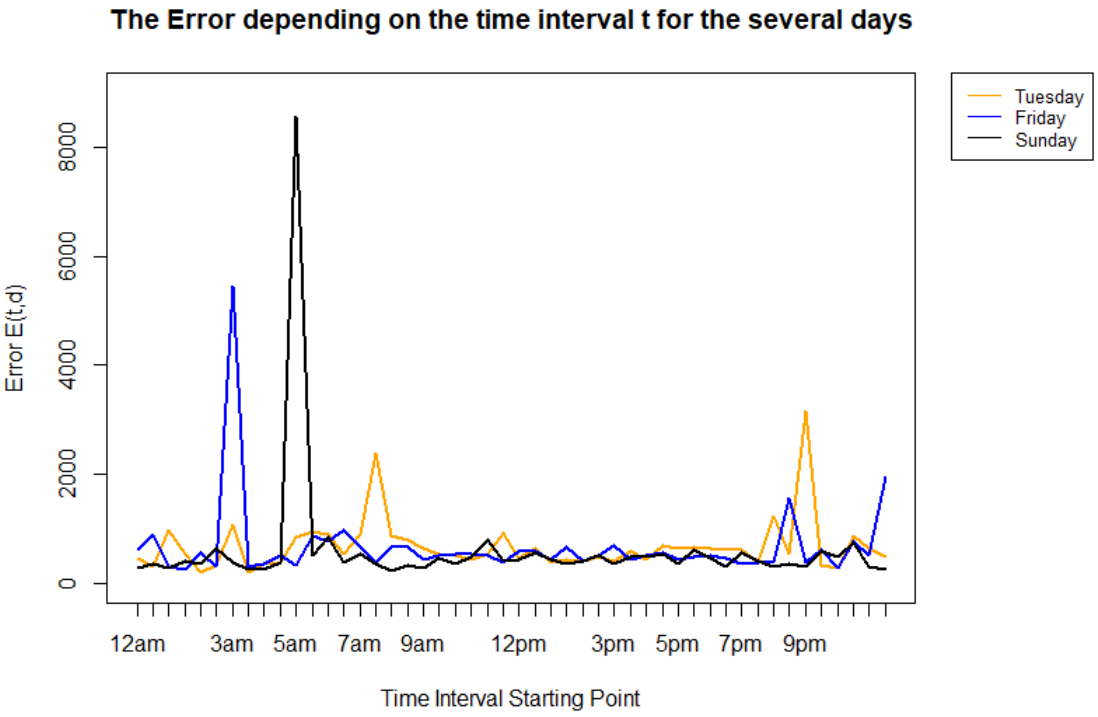


Figure 4: Error $E(t, d)$ over time t for Tuesday, Friday and Sunday

Monday:	577.5909
Tuesday:	677.5192
Wednesday:	529.6511
Thursday:	596.4945
Friday:	668.3813
Saturday:	559.1744
Sunday:	593.8636

Table 3: Average error $E(d)$

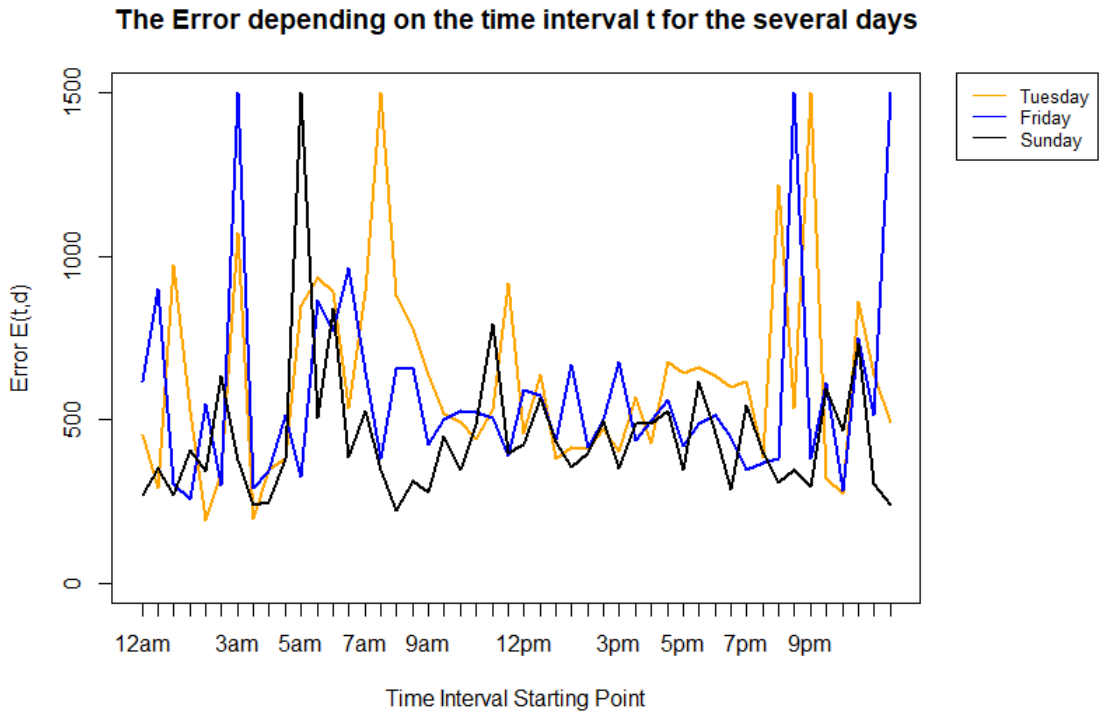


Figure 5: Error $E(t, d)$ over time t for Tuesday, Friday and Sunday, with values above 1500 eliminated.

influenced by outliers. Outliers can occure if either an estimator $\mu(i, j, t, d)$ is influenced by outliers or the respective data set to compute the error is influenced by outliers. In both cases the error $e(i, j, t, d)$ is high. If there are only a few errors $e(i, j, t, d)$ involved in the computation of $E(i, t, d)$ for a specific i, then $E(i, t, d)$ is strongly influenced by outliers. If, in turn, only a few $E(i, t, d)$s exist, the error $E(t, d)$ is also strongly influenced by outliers. This effect is analyzed in more detail later in the chapter.

- The error $E(t, d)$ differs strongly for different days. This could be due to outliers of specific days and time intervals. If they occur more often or are stronger within one time interval t and a day d, the error $E(t, d)$ is higher. This could be a sign of a weakness of the estimator $\mu(i, j, t, d)$: outliers have a strong influence on $\mu(i, j, t, d)$. This makes sense, since the sample mean can be strongly distorted by outliers.

- There is no time interval where the error for all days is low. In general, between 9am and 4pm most of the errors are below 1000, but the prediction of trip duration

is not very accurate for specific time intervals. A reason for this could be that for every time interval t and day d there are $4160 \times 4160 = 17,305,600$ cells (i, j, t, d) but only about $30,000$ data points. So, on the one hand, there are many places where no estimation exists. On the other hand, many estimators $\mu(i, j, t, d)$ are based on a single data point. This data point can be an outlier. So the chances that several $\mu(i, j, t, d)$s are outliers are high. Every outlying $\mu(i, j, t, d)$ can result in a large error $e(i, j, t, d)$. If there are only few errors $e(i, j, t, d)$ for a given (i, t, d), the error $E(i, t, d)$ is strongly influenced by such outliers. If, then, only a few errors $E(i, t, d)$ exist for each (t, d), the overall error $E(t, d)$ is also strongly influenced by outliers. This can lead to the observations made in the graphs.

- Between 7am and 9am on Monday, Tuesday, Wednesday and Thursday the error $E(t, d)$ is very high, on Saturday and Sunday the error $E(t, d)$ is relatively low. On Friday, the error lies between the error on Sunday and the error on Tuesday. The reason for this observation could be that from Monday to Thursday between 7am and 9am many people are driving to work. So there is much traffic on the streets. So trip duration can vary a lot. This results in a high volatility of the respective estimator and in a high error. On Saturday and Sunday between 7am and 9am, there is generally not much traffic, so trip durations are more stable. For the observation that on Friday the error lies between the error on Tuesday and the error on Sunday, there is no explanation.

- In contrast to the error $E(t, d)$ the overall error of the seven days does not vary too much. The range of the error $E(d)$ is about 140. The reason for this observation could be, that the data are not too extremly distributed over the time t. So outliers do not have such a strong influence on the overall error $E(d)$.

Based on these observations, the question arises as to how many errors $E(i, t, d)$ exist. The more $E(i, t, d)$s exist, the less $E(t, d)$ should be influenced by outliers. So the number of errors $E(i, t, d)$ that exist are shown in Figure 6. It can be observed in Figure 6, that for about all days and time intervals the number of existing errors $E(i, t, d)$ is below

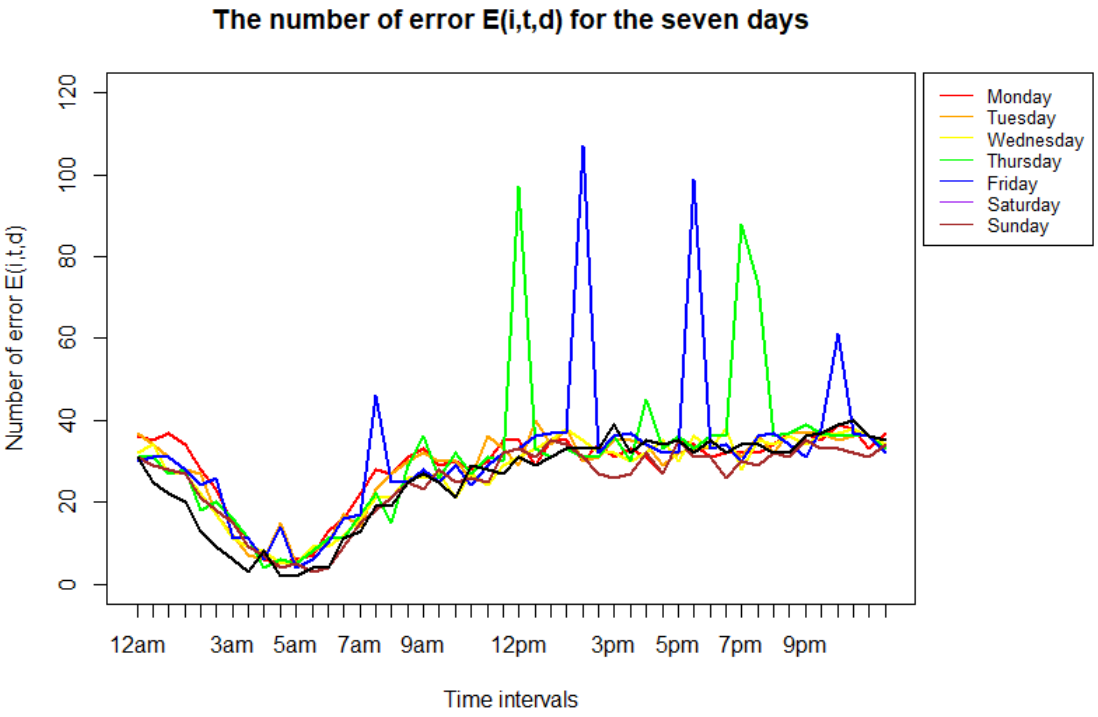


Figure 6: Number of existing $E(i,t,d)$s.

100. Especially on all days between 2 am and 4 am, there are only a few existing errors $E(i,t,d)$. Based on this observation, if one error $E(i,t,d)$ is strongly influenced by outliers, the respective error $E(t,d)$ becomes very large. The question is now if the value of some errors $E(t,d)$ are high because of single outliers, because of few errors $E(i,t,d)$ or because of large errors in this time interval on this day in general.

To answer this, boxplots of selected errors $E(i,t,d)$ for fixed t and d are analyzed. First, the boxplot of errors $E(i,t,d)$s with very high values of $E(t,d)$ are considered. The first boxplot is for $E(i,t,d)$s on Sunday at 5 am (Figure 7). It can be observed in Figure 7 that the error $E(t,d)$ is based on only a few values of $E(i,t,d)$. In this case only two errors $E(i,t,d)$ exist. One of those $E(i,t,d)$s is a strong outlier, the other one is also quite high. Thus $E(t,d)$ is strongly influenced by outliers. A reason for this is that very few data points are in the respective time interval on the specific day. So it can be possible that single estimators $\mu(i,j,t,d)$ are based on a few outliers in the datapoints. If estimators $\mu(i,j,t,d)$ are influenced by outliers in the dataset, the respective error $e(i,j,t,d)$ is very high. Because there exist only a few estimators, there are also only a

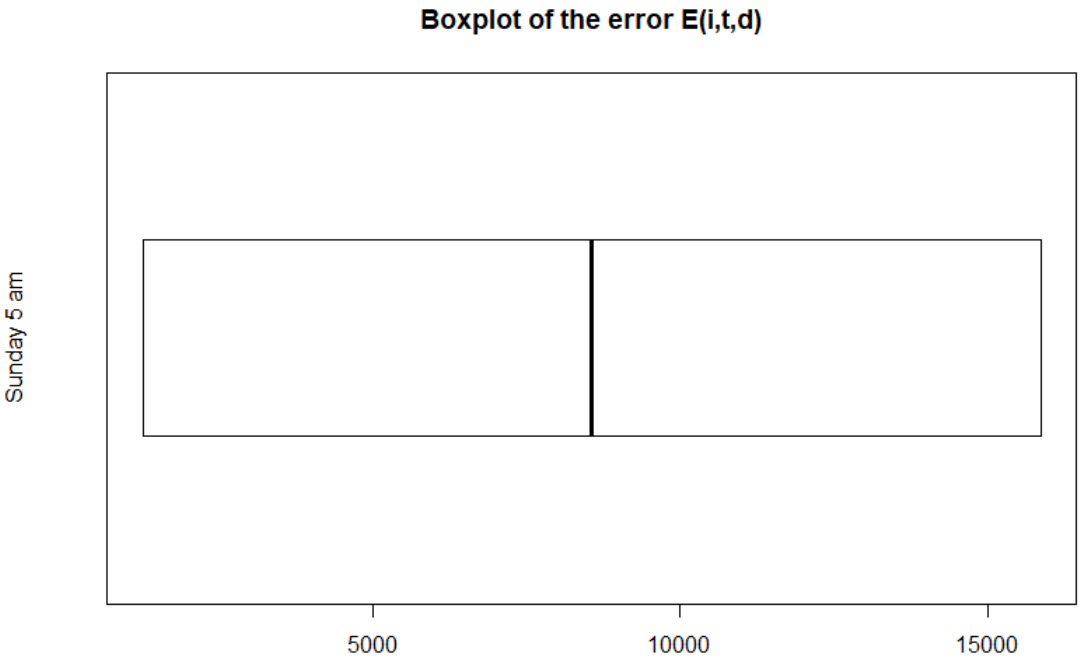


Figure 7: Boxplot of errors $E(i,t,d)$ on Sunday at time interval [5 am, 5.30 am)

few errors $e(i,j,t,d)$. Those errors $e(i,j,t,d)$ then have a strong influence on $E(i,t,d)$. This can result in the above observation. Another reason for that observation could be that there are only few data in the test dataset used for prediction. Then there can be some outliers in this data set causing a high error $e(i,j,t,d)$. Again, because there are only a few errors $e(i,j,t,d)$, they have a strong influence to the error $E(i,t,d)$.

Now the outliers on Tuesday at 9 pm and on Friday at 3 am are analysed. Because of the outliers, here also the boxplot without outliers is considered. The outliers of a boxplot are data points which are ouside of the interval $[Q_1 - 1.5 \times \text{IQR}, Q_3 + 1.5 \times \text{IQR}]$, where Q_1 and Q_3 are the first and third quartiles, respectively, and IQR is the interquartile range $Q_3 - Q_1$. In this case, we see in Figures 8 and 9 that the error $E(t,d)$ is influenced by single outliers. The error of these outliers is high enough that they have a strong influence on $E(t,d)$. In contrast to the boxplot for Sunday at 5 am, the number of error $E(i,t,d)$ is much higher. As a result, although the outlier on Tuesday at 9 pm is eight times higher than the outlier on Sunday at 5 am, the error $E(t,d)$ on Tuesday at 9 pm is much smaller. The second graph also shows that most errors $E(i,t,d)$ are quite low.

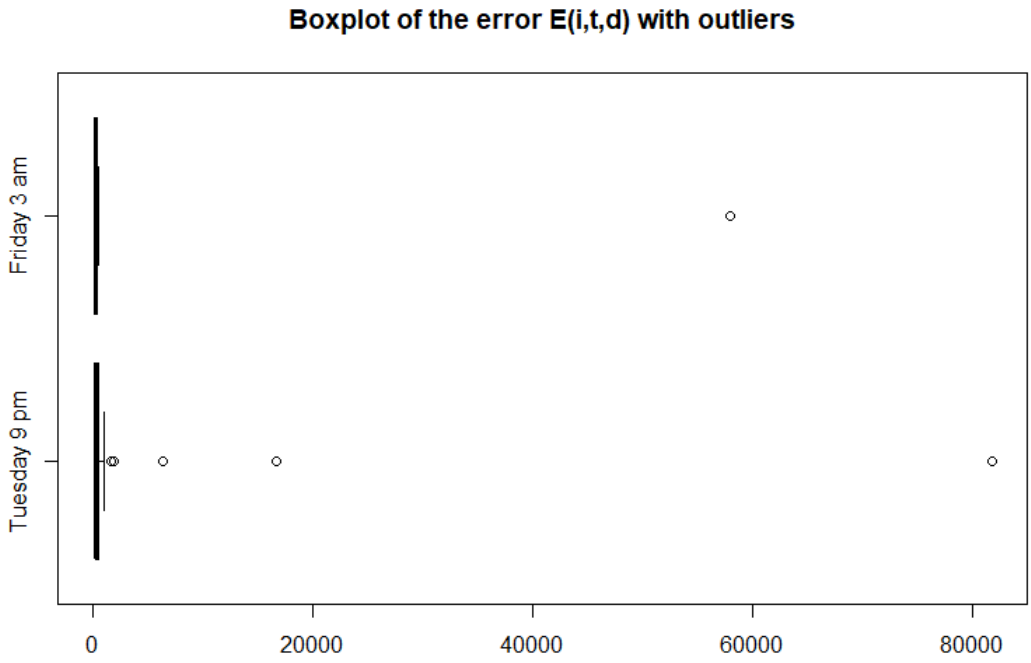


Figure 8: Boxplot of errors $E(i,t,d)$ with outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am)

The median of both errors $E(i,t,d)$ is below 300. So the error is not high in general, but influenced by outliers.

All in all, it is sometimes the case, that the outliers of $E(t,d)$ are caused by too few existing $E(i,t,d)$s. But it is also possible that the outliers are so extreme that the error $E(t,d)$ is high even though there are many $E(i,t,d)$s. The third possibility, a high error in general, does not occur in the three observed cases.

Let us consider now the $E(t,d)$ with the largest number of existing $E(i,t,d)$s. For better visualisation, the first plot (Figure 10) includes outliers and the second one (Figure 11) does not. It can be observed that there are some outliers on Friday at 2 pm and at 5.30 pm. On Friday at 10 pm there are only a few outliers and not as large as those at 2 pm or at 5.30 pm. In contrast to the previous boxplots, they do not have such a strong influence on the error $E(t,d)$. The reason is that the outliers are not as large as in the previous boxplots. Another reason is that there are more existing errors $E(i,t,d)$s, leading to a situation where single outliers do not have such a strong influence on $E(t,d)$.

In Figures 11 and 9, the plots without outliers, it can be observed that most errors

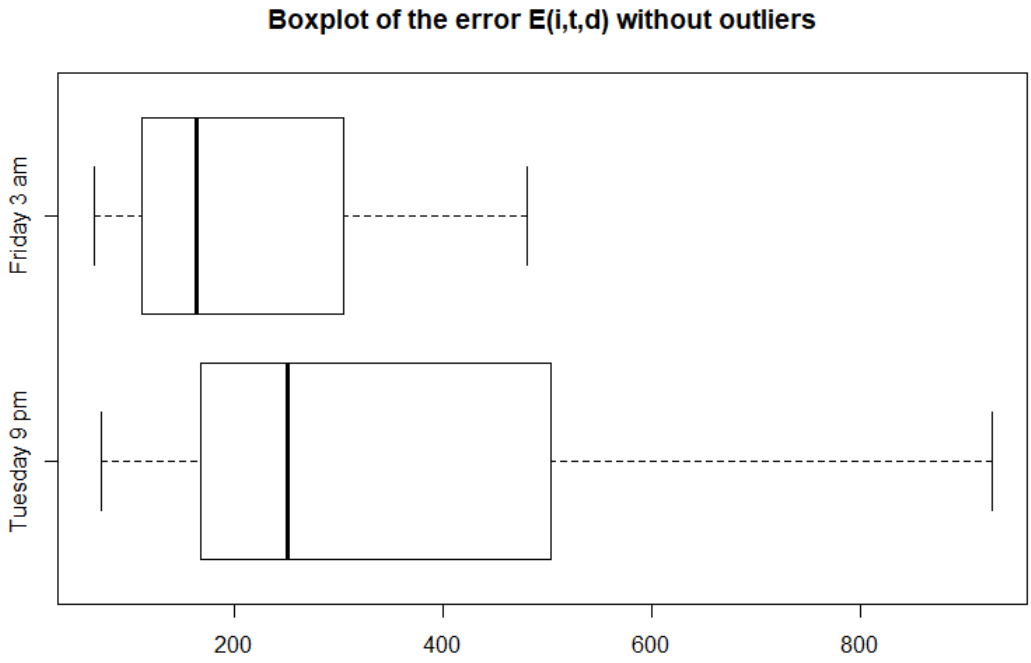


Figure 9: Boxplot of errors $E(i,t,d)$ without outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am)

$E(i,t,d)$ are below 1000. The median of the error in both graphs is below 300. This shows that the errors $E(i,t,d)$ lie mostly between 0 and 600, which is a fine result. It can be expected that this also holds for the other days and time intervals, which are not considered here.

All in all, the errors $E(t,d)$ and $E(d)$ are not so high. The question now is if pooling the data reduces the error or not. This is analyzed in the next chapter.

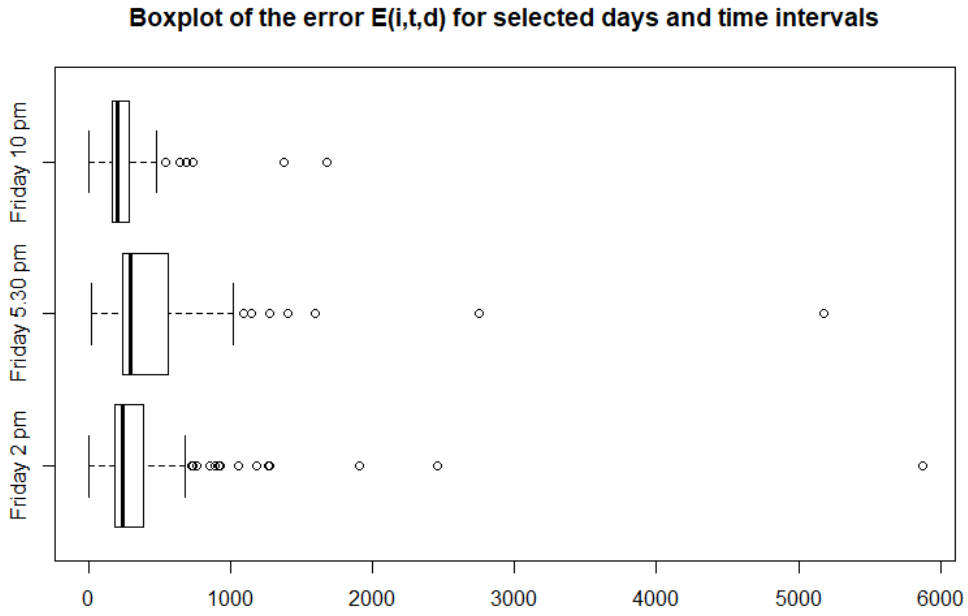
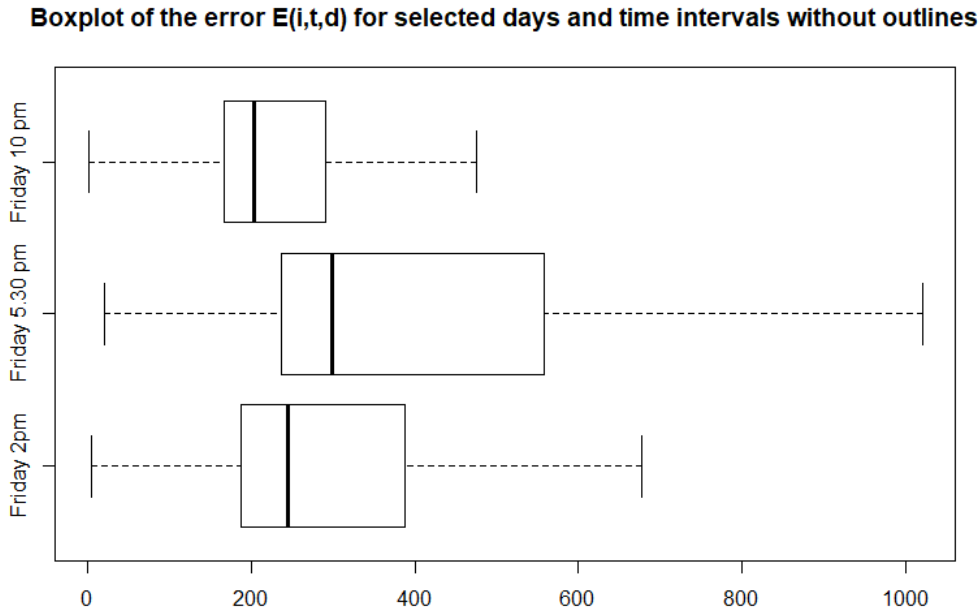


Figure 10: Boxplots of $E(i,t,d)$s for Friday at time intervals [2 pm, 2.30 pm), [5.30 pm, 6 pm) and [10 pm, 10.30 pm), with outliers.

Figure 11: Boxplots of $E(i,t,d)$s for Friday at time intervals [2 pm, 2.30 pm), [5.30 pm, 6 pm) and [10 pm, 10.30 pm), without outliers.



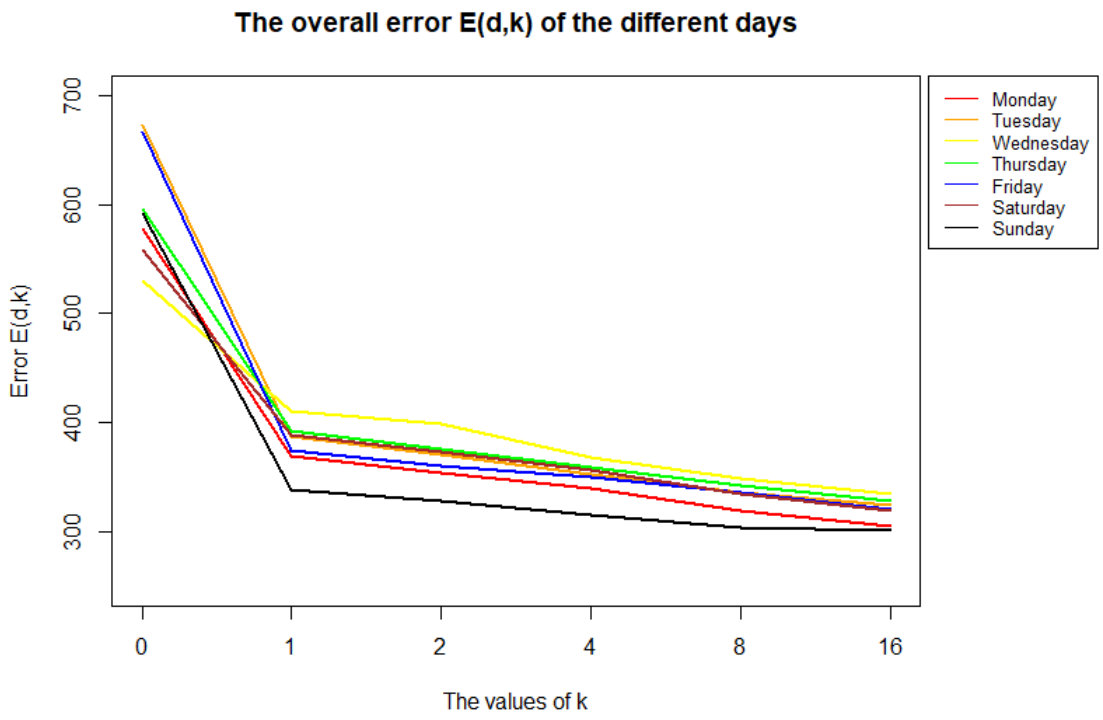


Figure 12: Errors $E(d, k)$ for different days d and pooling thresholds k.

5.2 Analysis of prediction error after pooling

In this chapter, the effects on the errors of pooling estimators are analyzed. The errors $E(d, k)$ for the respective days are shown in Figure 12. The x-axis in this graph gives the ks. The computation is only done for specific numbers of k, so the function is not continuous. Also note that the scale of the values k is not linear. The ratio $E(d, k)/E(d, 0)$, where $E(d, 0)$ is the original error with no pooling, is also shown in Figure 13.

Several observations can be made:

- Overall, the error decreases as k increases. The minimum $E(d, k)$ for all days d occurs at $k = 16$. So the question arises if there is a finite k such that the error $E(d, k)$ is minimized or if it is best to pool all seven days together to attain a minimum. In general, there should be a finite value of k such that the error is minimized. Pooling all days together would intuitively not make sense, since, for example, it should take more time to go from A to B on Monday morning than on Sunday morning. A reason for this observation could be the existence of outliers in

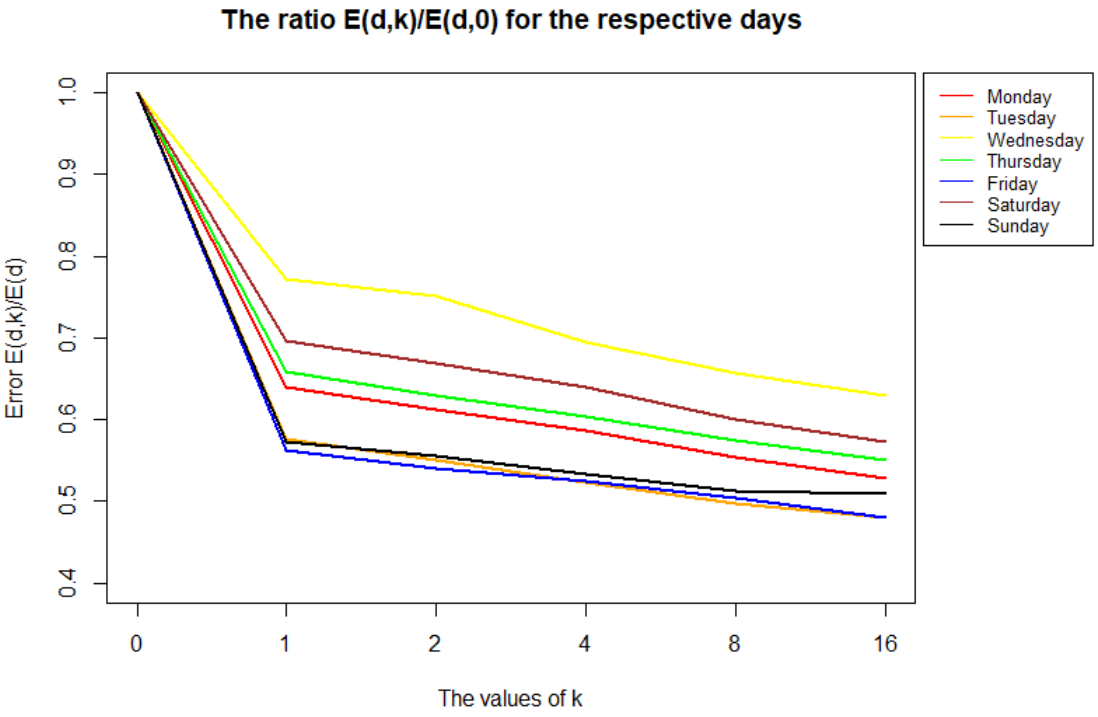


Figure 13: Ratios $E(d,k)/E(d)$ for several days d and pooling thresholds k.

the estimators $\mu(i,j,t,d)$. For some of those outliers a high k might be needed to pool them with other estimators, and this pooling reduces the error even for high values of k.

- For all seven days there is a large difference between the error $E(d)$ without pooling and the error $E(d,1)$. At $k=0$ all errors $E(d)$ are above 500, at $k=1$ all errors $E(d,1)$ (except for Wednesday) are below 400. The error $E(d,1)$ for Wednesday is slightly above 400. So even for a small value of k, there is a large reduction in prediction error. A reason for that could be that several outlying estimators are getting pooled with non-outlying estimators. This can strongly reduce the error. Later in this chapter, this is analyzed in more depth.

- The error does not change too much between $k=1$ and $k=16$. This can be an indicator that there exists a minimum or that the error converges to a minimum, when all days are pooled together. A reason for that could be that most outliers are already pooled at $k=1$. After that, there might be only a few estimators that

are pooled and the error reduction is then not so remarkable. This is analyzed in more detail later in this chapter.

- For $k = 16$, the error of the estimator for the respective days does not vary as strongly as for $k = 0$. A reason for this could be that $E(d, 16)$ is much less influenced by outliers than $E(d, 0)$. There exists a different number of outliers for each day, as seen in Chapter 5.1. So, if at $k = 16$ the error $E(d, 16)$ is less influenced by outliers, the error would not vary as much as before. Another reason could be that the error $E(d, 16)$ is at a lower level than the error $E(d, 0)$. So the ratio $\frac{"Range\quad of\quad the\quad error"}{"mean\quad of\quad the\quad error"}$ might be about equal, but because the error is smaller, the range of the errors reduces.

- The errors $E(d, k)$ for Wednesday show a different curve shape than the others. At $k = 0$ the error is the lowest of the seven days, but from $k = 1$ to $k = 16$ the error is higher than the others. In Figure 13, where the ratio $\frac{E(d,k)}{E(d,0)}$ is observed, this effect can be observed more clearly. A reason for this observation could be that on Wednesday the error $E(d, t)$ for the specific time intervals were much less influenced by outliers than the other errors. So the pooling reduces the error, but not as strongly as for the other days. Also, it might be possible that the outliers are more evenly distributed over several estimators $\mu(i, j, t, d)$. So it takes a higher k to pool those estimators and reduce the error. This is analyzed more deeply later in this chapter.

- The ratios $\frac{E(d,k)}{E(d,0)}$ for Tuesday, Friday and Sunday are smaller than for the other days. The reason is that the outliers on Tuesday, Friday and Sunday were much higher than for the other days. So pooling those outliers with other estimators reduces the error more substantially for those days than for the others.

The question now is why the curve on Wednesday differs from the other curves. To answer that, the error $E(t, d, k)$ for different values of k is analyzed (Figure 14).It can be observed that from 6 am to 11 pm the curves for $k = 1, 2, 4, 8, 16$ lie mostly below the curve of the error without pooling. Also, between 6 am and 11 pm there are no outliers.

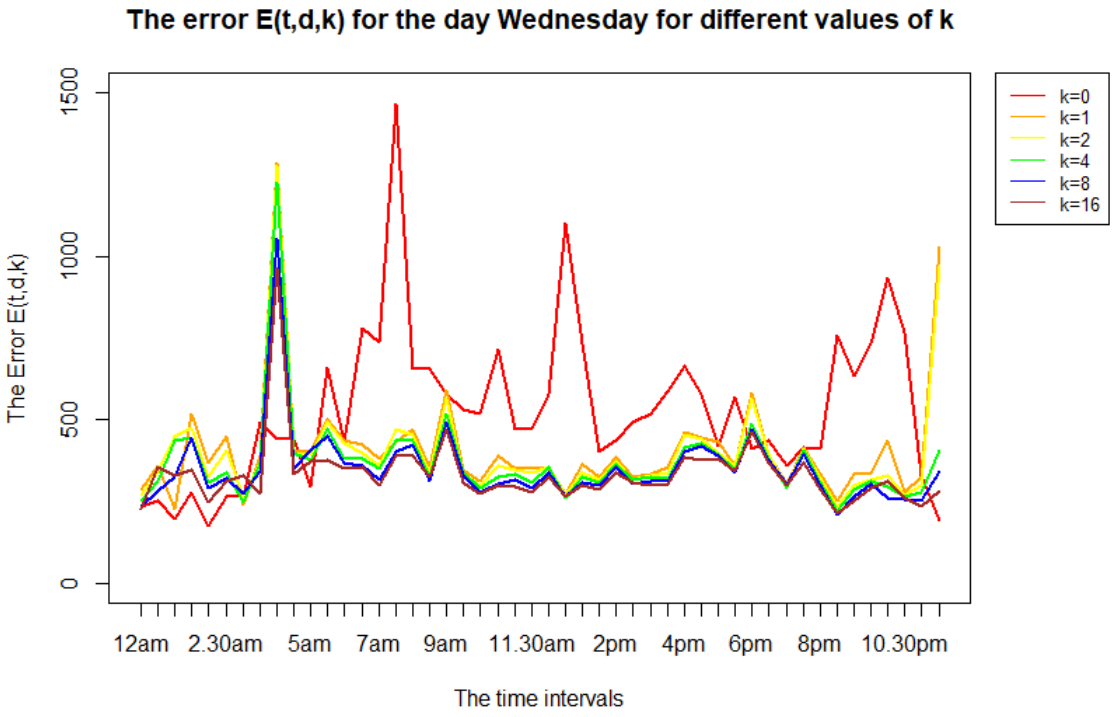


Figure 14: Errors $E(t, d, k)$ for Wednesday for different values of k

At 4 am the error $E(t, d, k)$ is higher than $E(t, d, 0)$ for all k. This explains why the ratio $\frac{E(d,k)}{E(d,0)}$ remains very high and why the error $E(d, k)$ for $k = 1, 2, 4, 8, 16$ is highest on Wednesday. A reason for this could be that the estimator at 4 am gets pooled with an estimator which is strongly influenced by outliers. So the error of this estimator increases. Also, at 4 am the error is highest for $k = 1$ and it decreases as k increases. A reason for this could be that, for higher values of k, more estimators are pooled together, so the error in general decreases. Another observation is that at 11.30 pm the error $E(d, t, k)$ for all k is higher than the error $E(d, t, 0)$. The difference between this time interval and the time interval [4 am, 4.30 am) is that for $k = 1$ and $k = 2$, the error is about 1000, very high. For $k = 4$, $k = 8$ and $k = 16$, the error is about 400. So for $k = 4$, $k = 8$ and $k = 16$ the error is much smaller than for $k = 1$ and $k = 2$, and it is closer to the original error $E(t, d, 0)$. This explains why the overall error $E(d, k)$ decreases more between $k = 2$ and $k = 4$. It also explains the reduction of the ratio $\frac{E(d,k)}{E(d,0)}$ between $k = 2$ and $k = 4$. A reason for this could be that, for a small k, some estimators are pooled together with outliers. For higher values of k, the error is pooled with additional estimators, so the

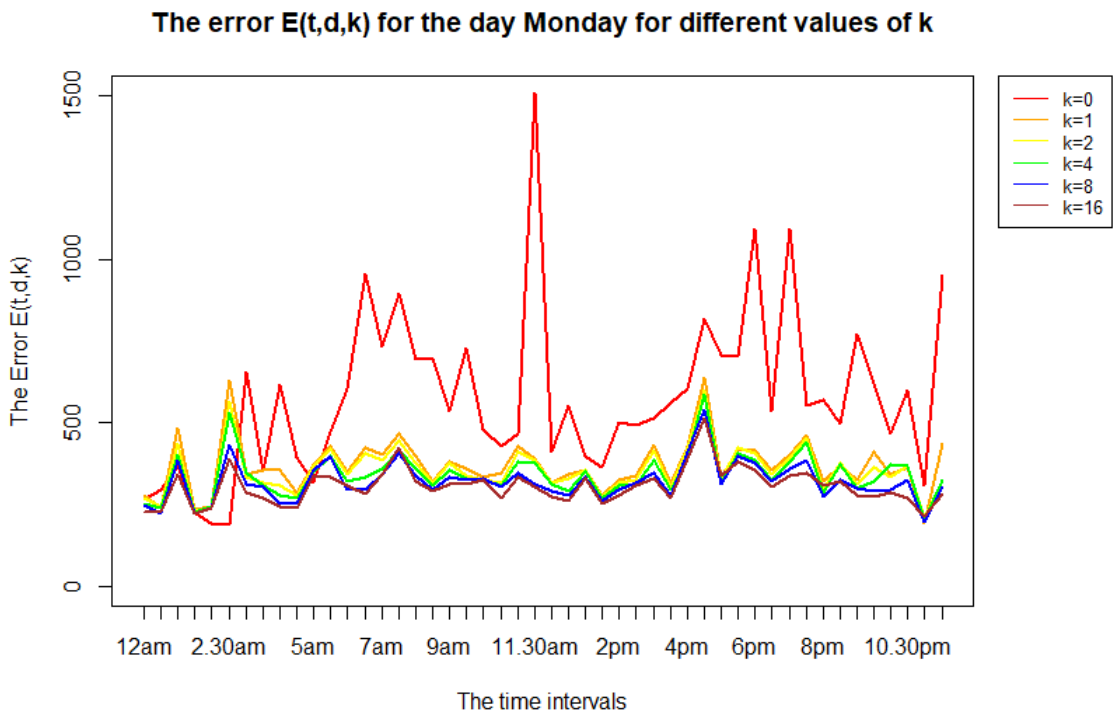


Figure 15: Errors $E(t, d, k)$ for Monday for different pooling thresholds k

effect of the outlier is reduced. Between 12 am and 3 am, the error $E(t, d, k)$ for all k is higher than the error $E(t, d, 0)$. Since the error does not decrease as k increases, there is no explanation for this behavior.

Now consider the errors $E(t, d, k)$ for Monday for different values of k (Figure 15). It can be observed that from 3 am to 11.30 pm the error $E(t, d, k)$ for all k is below the error $E(t, d, 0)$. Especially the outlier at 12 am does not occur. This explains the course of the curve in Figures 12 and 13. It seems that pooling reduces the influence of outliers and reduces the overall error. Only from 12 am to 2.30 am and especially at 2.30 am is the error $E(d, t, k)$ partially higher than the original error $E(t, d, 0)$. But at many other time intervals the error is much lower than 1000 and not as high as the original error $E(t, d, 0)$. Here it could be possible that the original estimators $\mu(i, j, t, d)$ for the time interval [2.30 am, 3 am) for all i and j are already very good estimates. After pooling, the quality of the new estimators may not be as good as before, which results in an increased value of $E(t, d, k)$. Here, for all values of $k > 0$, the error mostly decreases as k increases. There are a few exceptions, but there are no outliers as on Wednesday. Also, for a fixed

time interval t, the errors $E(t, d, k)$ are very similar for different values of k.

6 Conclusions and outlook

A main goal of this thesis was to analyze if pooling reduces the prediction error and if there exists a k such that the error is minimized. All in all, pooling reduces the error $E(d,k)$ and the error $E(t,d,k)$ as seen in Chapter 5.2. Also, the influence of outliers on estimators can be reduced by pooling, which reduces the error $E(t,d,k)$. In addition, the error $E(t,d,k)$ has a smaller range as k increases. This could indicate that pooling estimators improves the quality of the estimators. The question of whether there exists a finite k that minimizes $E(d,k)$ cannot be answered. The reason is that the minimum occurs at the highest value of the ks we tried, $k = 16$. So we do not know if the error keeps decreasing as k goes to infinity or if there exists a k such that the error is minimized. However, the change in the error $E(d,k)$ and $E(t,d,k)$ for $k = 8$ and $k = 16$ is very small. So $k = 16$ could be a good choice of k to minimize the error.

Another goal of this thesis was to determine if the cluster mean was good for predicting trip duration. On the one hand the mean is very sensitive to outliers and it can be observed that some estimators $\mu(i,j,t,d)$ are strongly influenced by outliers. Pooling estimators is needed to reduce the influence of outliers and it is also needed to reduce the range of the errors $E(t,d,k)$. On the other hand, the mean is an estimator that can be computed, recomputed and pooled with minimal computational effort, and it works well when there are no outliers. So overall the mean is a reasonably good estimator.

However, this thesis rises some questions: for about every day and every time interval the number of existing errors $E(i,t,d)$ is small, for some days and time intervals there exist less than ten errors $E(i,t,d)$ out of 4160 possible (i,t,d)s. This leads to the question if there are enough data for estimation. For computational reasons, the data of only one year was used for estimation. With more computational power, it is possible to use more years. If more data points are used, more $E(i,t,d)$s might exist. Also, single estimatros could be less influenced by outliers, which might reduce the errors $e(i,j,t,d)$ and also the errors $E(i,t,d)$ and $E(t,d)$. Also, it might be useful to use a second year for the test data set for the computation of the errors. In the context of many errors $e(i,j,t,d)$ which do not exist, the question is if clustering the possible pickup and dropoff points by rectangles

is the best approach. In Figure 1 it can be observed, that the pickup places are more trapezoidally distributed. So the question is, if a different form of the possible pickup places and dropoff places can be used. Then it is also possible to take all points into account without increasing the computational time to much. These are some questions that could be analyzed in the future.

7 References

[1] City of Chicago (2020). *Taxi Trips, Data Portal.* https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew

[2] Krishnaswamy, S. Singh, A., Wu, W, and Xiang,S. (2001). *Taxi Trip Time Prediction Using Similar Trips and Road Network Data.* 2015 IEEE International Conference on Big Data (Big Data), pp.2892-2894.

Appendix R-Code

For the statistical computation the software R is used:

Programm 1: R-Code for sorting the data. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
#Reading the data for 2019

Daten1=read.csv("Taxi_Trips-2018-Jan-1.csv")
Daten1=Daten1[,c(1,3,5,18,19,21,22)]
Daten2=read.csv("Taxi_Trips-2018-Jan-2.csv")
Daten2=Daten2[,c(1,3,5,18,19,21,22)]
Jan=rbind(Daten1, Daten2)
Daten1=NULL #Those Data will not be used any more, this is
        #for improving the speed and stability of the program
Daten2=NULL

#Do the same process for the other 11 months

#Find the maxima and minima of the Latitude and Longitude
        #of the pickup and dropoff points
Lat_max=max(max(Jan$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Jan$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Feb$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Feb$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Mar$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Mar$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Apr$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Apr$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(May$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(May$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Jun$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Jun$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Jul$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Jul$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Aug$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Aug$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Sep$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Sep$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Oct$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Oct$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Nov$Pickup.Centroid.Latitude, na.rm = TRUE),
            max(Nov$Dropoff.Centroid.Latitude, na.rm = TRUE),
            max(Dec$Pickup.Centroid.Latitude, na.rm = TRUE),
```

```
            max(Dec$Dropoff.Centroid.Latitude, na.rm = TRUE))

#Do the same process with Lat_min, Long_max and Long_min

Information=c(Lat_min, Lat_max, Long_min, Long_max)
write.csv(Information,"Values-for-estimation.csv")

#Split the data into the days via functions (Monday, Tuesday,
        # Wednesday, Thursday, Friday, Saturday, Sunday)
Day_Split_Mo=function(Daten) {
  Test=filter(Daten, Daten$Pickup.Centroid.Latitude != ""
        & Daten$Dropoff.Centroid.Latitude != "")
  Monday=filter(Test,
        str_detect(Trip.Start.Timestamp, '01/01/2018') |
        str_detect(Trip.Start.Timestamp, '01/08/2018') |
        str_detect(Trip.Start.Timestamp, '01/15/2018') |
        str_detect(Trip.Start.Timestamp, '01/22/2018') |
        str_detect(Trip.Start.Timestamp, '01/29/2018') |
        str_detect(Trip.Start.Timestamp, '02/05/2018') |
        str_detect(Trip.Start.Timestamp, '02/12/2018') |
        str_detect(Trip.Start.Timestamp, '02/19/2018') |
        str_detect(Trip.Start.Timestamp, '02/26/2018') |
        str_detect(Trip.Start.Timestamp, '03/05/2018') |
        str_detect(Trip.Start.Timestamp, '03/12/2018') |
        str_detect(Trip.Start.Timestamp, '03/19/2018') |
        str_detect(Trip.Start.Timestamp, '03/26/2018') |
        str_detect(Trip.Start.Timestamp, '04/02/2018') |
        str_detect(Trip.Start.Timestamp, '04/09/2018') |
        str_detect(Trip.Start.Timestamp, '04/16/2018') |
        str_detect(Trip.Start.Timestamp, '04/23/2018') |
        str_detect(Trip.Start.Timestamp, '04/30/2018') |
        str_detect(Trip.Start.Timestamp, '05/07/2018') |
        str_detect(Trip.Start.Timestamp, '05/14/2018') |
        str_detect(Trip.Start.Timestamp, '05/21/2018') |
        str_detect(Trip.Start.Timestamp, '05/28/2018') |
        str_detect(Trip.Start.Timestamp, '06/04/2018') |
        str_detect(Trip.Start.Timestamp, '06/11/2018') |
        str_detect(Trip.Start.Timestamp, '06/18/2018') |
        str_detect(Trip.Start.Timestamp, '06/25/2018') |
        str_detect(Trip.Start.Timestamp, '07/02/2018') |
        str_detect(Trip.Start.Timestamp, '07/09/2018') |
        str_detect(Trip.Start.Timestamp, '07/16/2018') |
        str_detect(Trip.Start.Timestamp, '07/23/2018') |
        str_detect(Trip.Start.Timestamp, '07/30/2018') |
        str_detect(Trip.Start.Timestamp, '08/06/2018') |
        str_detect(Trip.Start.Timestamp, '08/13/2018') |
```

```
          str_detect(Trip.Start.Timestamp, '08/20/2018') |
          str_detect(Trip.Start.Timestamp, '08/27/2018') |
          str_detect(Trip.Start.Timestamp, '09/03/2018') |
          str_detect(Trip.Start.Timestamp, '09/10/2018') |
          str_detect(Trip.Start.Timestamp, '09/17/2018') |
          str_detect(Trip.Start.Timestamp, '09/24/2018') |
          str_detect(Trip.Start.Timestamp, '10/01/2018') |
          str_detect(Trip.Start.Timestamp, '10/08/2018') |
          str_detect(Trip.Start.Timestamp, '10/15/2018') |
          str_detect(Trip.Start.Timestamp, '10/22/2018') |
          str_detect(Trip.Start.Timestamp, '10/29/2018') |
          str_detect(Trip.Start.Timestamp, '11/05/2018') |
          str_detect(Trip.Start.Timestamp, '11/12/2018') |
          str_detect(Trip.Start.Timestamp, '11/19/2018') |
          str_detect(Trip.Start.Timestamp, '11/26/2018') |
          str_detect(Trip.Start.Timestamp, '12/03/2018') |
          str_detect(Trip.Start.Timestamp, '12/10/2018') |
          str_detect(Trip.Start.Timestamp, '12/17/2018') |
          str_detect(Trip.Start.Timestamp, '12/24/2018') |
          str_detect(Trip.Start.Timestamp, '12/31/2018'))
  return(Monday)
}

#Do the same process for the other six days.
          #The dates change.

#Generate datasets for the seven days depending on
          #the respective month
Jan_Mo=Day_Split_Mo(Jan)
Jan_Tu=Day_Split_Tu(Jan)
Jan_We=Day_Split_We(Jan)
Jan_Th=Day_Split_Th(Jan)
Jan_Fr=Day_Split_Fr(Jan)
Jan_Sa=Day_Split_Sa(Jan)
Jan_Su=Day_Split_Su(Jan)
Jan=NULL

#Do the same process for the other months.

#Generate the datasets for one specific day
          #Monday, Tuesday, etc. independent of the
          #month.
Monday=rbind(Jan_Mo, Feb_Mo, Mar_Mo, Apr_Mo, May_Mo, Jun_Mo,
          Jul_Mo, Aug_Mo, Sep_Mo, Oct_Mo, Nov_Mo, Dec_Mo)
Jan_Mo=NULL
Feb_Mo=NULL
```

```
Mar_Mo=NULL
Apr_Mo=NULL
May_Mo=NULL
Jun_Mo=NULL
Jul_Mo=NULL
Aug_Mo=NULL
Sep_Mo=NULL
Oct_Mo=NULL
Nov_Mo=NULL
Dec_Mo=NULL

#Do the same process for the other six days.

# Split the matrix into time intervals
#Monday
Monday=read.csv("Monday-2018.csv")
Monday_0000=filter(Monday,
        str_detect(Trip.Start.Timestamp, '12:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '12:15:00 AM'))
Monday_0030=filter(Monday,
        str_detect(Trip.Start.Timestamp, '12:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '12:45:00 AM'))
Monday_0100=filter(Monday,
        str_detect(Trip.Start.Timestamp, '01:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '01:15:00 AM'))
Monday_0130=filter(Monday,
        str_detect(Trip.Start.Timestamp, '01:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '01:45:00 AM'))
Monday_0200=filter(Monday,
        str_detect(Trip.Start.Timestamp, '02:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '02:15:00 AM'))
Monday_0230=filter(Monday,
        str_detect(Trip.Start.Timestamp, '02:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '02:45:00 AM'))
Monday_0300=filter(Monday,
        str_detect(Trip.Start.Timestamp, '03:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '03:15:00 AM'))
Monday_0330=filter(Monday,
        str_detect(Trip.Start.Timestamp, '03:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '03:45:00 AM'))
Monday_0400=filter(Monday,
        str_detect(Trip.Start.Timestamp, '04:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '04:15:00 AM'))
Monday_0430=filter(Monday,
        str_detect(Trip.Start.Timestamp, '04:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '04:45:00 AM'))
```

```
Monday_0500=filter(Monday,
        str_detect(Trip.Start.Timestamp, '05:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '05:15:00 AM'))
Monday_0530=filter(Monday,
        str_detect(Trip.Start.Timestamp, '05:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '05:45:00 AM'))
Monday_0600=filter(Monday,
        str_detect(Trip.Start.Timestamp, '06:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '06:15:00 AM'))
Monday_0630=filter(Monday,
        str_detect(Trip.Start.Timestamp, '06:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '06:45:00 AM'))
Monday_0700=filter(Monday,
        str_detect(Trip.Start.Timestamp, '07:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '07:15:00 AM'))
Monday_0730=filter(Monday,
        str_detect(Trip.Start.Timestamp, '07:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '07:45:00 AM'))
Monday_0800=filter(Monday,
        str_detect(Trip.Start.Timestamp, '08:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '08:15:00 AM'))
Monday_0830=filter(Monday,
        str_detect(Trip.Start.Timestamp, '08:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '08:45:00 AM'))
Monday_0900=filter(Monday,
        str_detect(Trip.Start.Timestamp, '09:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '09:15:00 AM'))
Monday_0930=filter(Monday,
        str_detect(Trip.Start.Timestamp, '09:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '09:45:00 AM'))
Monday_1000=filter(Monday,
        str_detect(Trip.Start.Timestamp, '10:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '10:15:00 AM'))
Monday_1030=filter(Monday,
        str_detect(Trip.Start.Timestamp, '10:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '10:45:00 AM'))
Monday_1100=filter(Monday,
        str_detect(Trip.Start.Timestamp, '11:00:00 AM') |
        str_detect(Trip.Start.Timestamp, '11:15:00 AM'))
Monday_1130=filter(Monday,
        str_detect(Trip.Start.Timestamp, '11:30:00 AM') |
        str_detect(Trip.Start.Timestamp, '11:45:00 AM'))
Monday_1200=filter(Monday,
        str_detect(Trip.Start.Timestamp, '12:00:00 PM') |
        str_detect(Trip.Start.Timestamp, '12:15:00 PM'))
Monday_1230=filter(Monday,
```

```
223         str_detect(Trip.Start.Timestamp, '12:30:00 PM') |
224         str_detect(Trip.Start.Timestamp, '12:45:00 PM'))
225 Monday_1300=filter(Monday,
226         str_detect(Trip.Start.Timestamp, '01:00:00 PM') |
227         str_detect(Trip.Start.Timestamp, '01:15:00 PM'))
228 Monday_1330=filter(Monday,
229         str_detect(Trip.Start.Timestamp, '01:30:00 PM') |
230         str_detect(Trip.Start.Timestamp, '01:45:00 PM'))
231 Monday_1400=filter(Monday,
232         str_detect(Trip.Start.Timestamp, '02:00:00 PM') |
233         str_detect(Trip.Start.Timestamp, '02:15:00 PM'))
234 Monday_1430=filter(Monday,
235         str_detect(Trip.Start.Timestamp, '02:30:00 PM') |
236         str_detect(Trip.Start.Timestamp, '02:45:00 PM'))
237 Monday_1500=filter(Monday,
238         str_detect(Trip.Start.Timestamp, '03:00:00 PM') |
239         str_detect(Trip.Start.Timestamp, '03:15:00 PM'))
240 Monday_1530=filter(Monday,
241         str_detect(Trip.Start.Timestamp, '03:30:00 PM') |
242         str_detect(Trip.Start.Timestamp, '03:45:00 PM'))
243 Monday_1600=filter(Monday,
244         str_detect(Trip.Start.Timestamp, '04:00:00 PM') |
245         str_detect(Trip.Start.Timestamp, '04:15:00 PM'))
246 Monday_1630=filter(Monday,
247         str_detect(Trip.Start.Timestamp, '04:30:00 PM') |
248         str_detect(Trip.Start.Timestamp, '04:45:00 PM'))
249 Monday_1700=filter(Monday,
250         str_detect(Trip.Start.Timestamp, '05:00:00 PM') |
251         str_detect(Trip.Start.Timestamp, '05:15:00 PM'))
252 Monday_1730=filter(Monday,
253         str_detect(Trip.Start.Timestamp, '05:30:00 PM') |
254         str_detect(Trip.Start.Timestamp, '05:45:00 PM'))
255 Monday_1800=filter(Monday,
256         str_detect(Trip.Start.Timestamp, '06:00:00 PM') |
257         str_detect(Trip.Start.Timestamp, '06:15:00 PM'))
258 Monday_1830=filter(Monday,
259         str_detect(Trip.Start.Timestamp, '06:30:00 PM') |
260         str_detect(Trip.Start.Timestamp, '06:45:00 PM'))
261 Monday_1900=filter(Monday,
262         str_detect(Trip.Start.Timestamp, '07:00:00 PM') |
263         str_detect(Trip.Start.Timestamp, '07:15:00 PM'))
264 Monday_1930=filter(Monday,
265         str_detect(Trip.Start.Timestamp, '07:30:00 PM') |
266         str_detect(Trip.Start.Timestamp, '07:45:00 PM'))
267 Monday_2000=filter(Monday,
268         str_detect(Trip.Start.Timestamp, '08:00:00 PM') |
```

```
        str_detect(Trip.Start.Timestamp, '08:15:00 PM'))
Monday_2030=filter(Monday,
        str_detect(Trip.Start.Timestamp, '08:30:00 PM') |
        str_detect(Trip.Start.Timestamp, '08:45:00 PM'))
Monday_2100=filter(Monday,
        str_detect(Trip.Start.Timestamp, '09:00:00 PM') |
        str_detect(Trip.Start.Timestamp, '09:15:00 PM'))
Monday_2130=filter(Monday,
        str_detect(Trip.Start.Timestamp, '09:30:00 PM') |
        str_detect(Trip.Start.Timestamp, '09:45:00 PM'))
Monday_2200=filter(Monday,
        str_detect(Trip.Start.Timestamp, '10:00:00 PM') |
        str_detect(Trip.Start.Timestamp, '10:15:00 PM'))
Monday_2230=filter(Monday,
        str_detect(Trip.Start.Timestamp, '10:30:00 PM') |
        str_detect(Trip.Start.Timestamp, '10:45:00 PM'))
Monday_2300=filter(Monday,
        str_detect(Trip.Start.Timestamp, '11:00:00 PM') |
        str_detect(Trip.Start.Timestamp, '11:15:00 PM'))
Monday_2330=filter(Monday,
        str_detect(Trip.Start.Timestamp, '11:30:00 PM') |
        str_detect(Trip.Start.Timestamp, '11:45:00 PM'))
Monday=NULL

#Do the same process with the other six days.

#Write the data into csv data
#Monday
write.csv(Monday_0000, "Monday-0000-2018.csv")
write.csv(Monday_0030, "Monday-0030-2018.csv")
write.csv(Monday_0100, "Monday-0100-2018.csv")
write.csv(Monday_0130, "Monday-0130-2018.csv")
write.csv(Monday_0200, "Monday-0200-2018.csv")
write.csv(Monday_0230, "Monday-0230-2018.csv")
write.csv(Monday_0300, "Monday-0300-2018.csv")
write.csv(Monday_0330, "Monday-0330-2018.csv")
write.csv(Monday_0400, "Monday-0400-2018.csv")
write.csv(Monday_0430, "Monday-0430-2018.csv")
write.csv(Monday_0500, "Monday-0500-2018.csv")
write.csv(Monday_0530, "Monday-0530-2018.csv")
write.csv(Monday_0600, "Monday-0600-2018.csv")
write.csv(Monday_0630, "Monday-0630-2018.csv")
write.csv(Monday_0700, "Monday-0700-2018.csv")
write.csv(Monday_0730, "Monday-0730-2018.csv")
write.csv(Monday_0800, "Monday-0800-2018.csv")
write.csv(Monday_0830, "Monday-0830-2018.csv")
```

```
write.csv(Monday_0900, "Monday-0900-2018.csv")
write.csv(Monday_0930, "Monday-0930-2018.csv")
write.csv(Monday_1000, "Monday-1000-2018.csv")
write.csv(Monday_1030, "Monday-1030-2018.csv")
write.csv(Monday_1100, "Monday-1100-2018.csv")
write.csv(Monday_1130, "Monday-1130-2018.csv")
write.csv(Monday_1200, "Monday-1200-2018.csv")
write.csv(Monday_1230, "Monday-1230-2018.csv")
write.csv(Monday_1300, "Monday-1300-2018.csv")
write.csv(Monday_1330, "Monday-1330-2018.csv")
write.csv(Monday_1400, "Monday-1400-2018.csv")
write.csv(Monday_1430, "Monday-1430-2018.csv")
write.csv(Monday_1500, "Monday-1500-2018.csv")
write.csv(Monday_1530, "Monday-1530-2018.csv")
write.csv(Monday_1600, "Monday-1600-2018.csv")
write.csv(Monday_1630, "Monday-1630-2018.csv")
write.csv(Monday_1700, "Monday-1700-2018.csv")
write.csv(Monday_1730, "Monday-1730-2018.csv")
write.csv(Monday_1800, "Monday-1800-2018.csv")
write.csv(Monday_1830, "Monday-1830-2018.csv")
write.csv(Monday_1900, "Monday-1900-2018.csv")
write.csv(Monday_1930, "Monday-1930-2018.csv")
write.csv(Monday_2000, "Monday-2000-2018.csv")
write.csv(Monday_2030, "Monday-2030-2018.csv")
write.csv(Monday_2100, "Monday-2100-2018.csv")
write.csv(Monday_2130, "Monday-2130-2018.csv")
write.csv(Monday_2200, "Monday-2200-2018.csv")
write.csv(Monday_2230, "Monday-2230-2018.csv")
write.csv(Monday_2300, "Monday-2300-2018.csv")
write.csv(Monday_2330, "Monday-2330-2018.csv")

#Do the same with the other six days.
#This process is done for the year 2018.
        #For the year 2019, the same process is done.
```

Program 2:Computation of the estimator $\mu(i, j, t, d)$. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
Find_Estimator_Var=function(file_matrix, name_mean, name_var, name_coun

library(dplyr)
```

```
library(expss)
Dataset=file_matrix
Dataset=Dataset[,5:9]
Dataset=filter(Dataset,Dataset$Trip.Seconds!="")

Step_Lat=0.003703
Step_Long=0.004206
Lat_min=41.786 #changed after the analysis in Chapter 3.1
Lat_max=Max_Min_Values[2]
Long_min=-87.80453 #changed after the analysis in Chapter 3.1
Long_max=Max_Min_Values[4]

n=ceiling((Lat_max-Lat_min)/Step_Lat)
m=ceiling((Long_max-Long_min)/Step_Long)

Estimator_Dataset=matrix(0,  nrow = n*m, ncol = m*n)
Variance_Dataset=matrix(0,  nrow = n*m, ncol = m*n)
count_Dataset=matrix(0,  nrow = n*m, ncol = m*n)

#Computation of the estimator
for (i in 1:n) {
  Matrix_Help = filter(
    Dataset, Dataset$Pickup.Centroid.Latitude >=
        (Lat_min + (i - 1) * Step_Lat)
    & Dataset$Pickup.Centroid.Latitude <
    (Lat_min + i * Step_Lat))
  if (nrow(as.matrix(Matrix_Help)) == 0) {
    next
  }
  for (j in 1:m) {
    Matrix_Help2 = filter(Matrix_Help,
        Matrix_Help$Pickup.Centroid.Longitude >=
                (Long_min + (j - 1) * Step_Long)
        & Matrix_Help$Pickup.Centroid.Longitude <
                (Long_min + j * Step_Long)
    )
    if (nrow(as.matrix(Matrix_Help2)) == 0) {
      next
    }
    for (k in 1:n) {
      Matrix_Help3 = filter(Matrix_Help2,
        Matrix_Help2$Dropoff.Centroid.Latitude >=
                (Lat_min + (k - 1) * Step_Lat)
        &  Matrix_Help2$Dropoff.Centroid.Latitude <
        (Lat_min + k * Step_Lat)
      )
```

```
      if (nrow(as.matrix(Matrix_Help3)) == 0) {
        next
      }
      for (l in 1:m) {
        Matrix_Help4 = filter(Matrix_Help3,
        Matrix_Help3$Dropoff.Centroid.Longitude >=
                (Long_min + (l - 1) * Step_Long)
        & Matrix_Help3$Dropoff.Centroid.Longitude <
                (Long_min + l * Step_Long)
        )
        if (nrow(as.matrix(Matrix_Help4)) == 0) {
          next
        }
        Daten = as.vector(Matrix_Help4[, 1])
        Daten2 = as.numeric(sub(",", "", Daten,
                fixed = TRUE))
        Estimator_Dataset[(i - 1) * m + j, (k - 1) * m + l]
                = mean(Daten2)
        Variance_Dataset[(i - 1) * m + j, (k - 1) * m + l]
                = var(Daten2)
        if(is.na(Variance_Dataset
                [(i - 1) * m + j, (k - 1) * m + l])==TRUE)
          {Variance_Dataset
                [(i - 1) * m + j, (k - 1) * m + l]=0}
        count_Dataset[(i - 1) * m + j, (k - 1) * m + l]
                = length(Daten2)
      }
    }
  }
}
write.csv(Estimator_Dataset, name_mean)
write.csv(Variance_Dataset, name_var)
write.csv(count_Dataset, name_count)
return(Variance_Dataset)
}

#read Data
Max_Min=read.csv("Values-for-estimation.csv")
Max_Min=Max_Min[,2]
Monday_0000=read.csv("Monday-0000-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0000,
        "Estimator-Monday-0000.csv",
        "Variance-Monday-0000.csv",
        "Number-Monday-0000.csv", Max_Min)
Monday_0000=NULL
Monday_0030=read.csv("Monday-0030-2018.csv")
```

```
Est_Mo_0000=Find_Estimator_Var(Monday_0030,
         "Estimator-Monday-0030.csv",
         "Variance-Monday-0030.csv",
         "Number-Monday-0030.csv",Max_Min)
Monday_0030=NULL
Monday_0100=read.csv("Monday-0100-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0100,
         "Estimator-Monday-0100.csv",
         "Variance-Monday-0100.csv",
         "Number-Monday-0100.csv",Max_Min)
Monday_0100=NULL
Monday_0130=read.csv("Monday-0130-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0130,
         "Estimator-Monday-0130.csv",
         "Variance-Monday-0130.csv",
         "Number-Monday-0130.csv",Max_Min)
Monday_0130=NULL
Monday_0200=read.csv("Monday-0200-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0200,
         "Estimator-Monday-0200.csv",
         "Variance-Monday-0200.csv",
         "Number-Monday-0200.csv",Max_Min)
Monday_0200=NULL
Monday_0230=read.csv("Monday-0230-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0230,
         "Estimator-Monday-0230.csv",
         "Variance-Monday-0230.csv",
         "Number-Monday-0230.csv",Max_Min)
Monday_0230=NULL
Monday_0300=read.csv("Monday-0300-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0300,
         "Estimator-Monday-0300.csv",
         "Variance-Monday-0300.csv",
         "Number-Monday-0300.csv",Max_Min)
Monday_0300=NULL
Monday_0330=read.csv("Monday-0330-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0330,
         "Estimator-Monday-0330.csv",
         "Variance-Monday-0330.csv",
         "Number-Monday-0330.csv",Max_Min)
Monday_0330=NULL
Monday_0400=read.csv("Monday-0400-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0400,
         "Estimator-Monday-0400.csv",
         "Variance-Monday-0400.csv",
         "Number-Monday-0400.csv",Max_Min)
```

```
Monday_0400=NULL
Monday_0430=read.csv("Monday-0430-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0430,
         "Estimator-Monday-0430.csv",
         "Variance-Monday-0430.csv",
         "Number-Monday-0430.csv",Max_Min)
Monday_0430=NULL
Monday_0500=read.csv("Monday-0500-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0500,
         "Estimator-Monday-0500.csv",
         "Variance-Monday-0500.csv",
         "Number-Monday-0500.csv",Max_Min)
Monday_0500=NULL
Monday_0530=read.csv("Monday-0530-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0530,
         "Estimator-Monday-0530.csv",
         "Variance-Monday-0530.csv",
         "Number-Monday-0530.csv",Max_Min)
Monday_0530=NULL
Monday_0600=read.csv("Monday-0600-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0600,
         "Estimator-Monday-0600.csv",
         "Variance-Monday-0600.csv",
         "Number-Monday-0600.csv",Max_Min)
Monday_0600=NULL
Monday_0630=read.csv("Monday-0630-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0630,
         "Estimator-Monday-0630.csv",
         "Variance-Monday-0630.csv",
         "Number-Monday-0630.csv",Max_Min)
Monday_0630=NULL
Monday_0700=read.csv("Monday-0700-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0700,
         "Estimator-Monday-0700.csv",
         "Variance-Monday-0700.csv",
         "Number-Monday-0700.csv",Max_Min)
Monday_0700=NULL
Monday_0730=read.csv("Monday-0730-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0730,
         "Estimator-Monday-0730.csv",
         "Variance-Monday-0730.csv",
         "Number-Monday-0730.csv",Max_Min)
Monday_0730=NULL
Monday_0800=read.csv("Monday-0800-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0800,
         "Estimator-Monday-0800.csv",
```

```
         "Variance-Monday-0800.csv",
         "Number-Monday-0800.csv",Max_Min)
Monday_0800=NULL
Monday_0830=read.csv("Monday-0830-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0830,
         "Estimator-Monday-0830.csv",
         "Variance-Monday-0830.csv",
         "Number-Monday-0830.csv",Max_Min)
Monday_0830=NULL
Monday_0900=read.csv("Monday-0900-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0900,
         "Estimator-Monday-0900.csv",
         "Variance-Monday-0900.csv",
         "Number-Monday-0900.csv",Max_Min)
Monday_0900=NULL
Monday_0930=read.csv("Monday-0930-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0930,
         "Estimator-Monday-0930.csv",
         "Variance-Monday-0930.csv",
         "Number-Monday-0930.csv",Max_Min)
Monday_0930=NULL
Monday_1000=read.csv("Monday-1000-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1000,
         "Estimator-Monday-1000.csv",
         "Variance-Monday-1000.csv",
         "Number-Monday-1000.csv",Max_Min)
Monday_1000=NULL
Monday_1030=read.csv("Monday-1030-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1030,
         "Estimator-Monday-1030.csv",
         "Variance-Monday-1030.csv",
         "Number-Monday-1030.csv", Max_Min)
Monday_1030=NULL
Monday_1100=read.csv("Monday-1100-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1100,
         "Estimator-Monday-1100.csv",
         "Variance-Monday-1100.csv",
         "Number-Monday-1100.csv",Max_Min)
Monday_1100=NULL
Monday_1130=read.csv("Monday-1130-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1130,
         "Estimator-Monday-1130.csv",
         "Variance-Monday-1130.csv",
         "Number-Monday-1130.csv",Max_Min)
Monday_1130=NULL
Monday_1200=read.csv("Monday-1200-2018.csv")
```

```
Est_Mo_0000=Find_Estimator_Var(Monday_1200,
         "Estimator-Monday-1200.csv",
         "Variance-Monday-1200.csv",
         "Number-Monday-1200.csv", Max_Min)
Monday_1200=NULL
Monday_1230=read.csv("Monday-1230-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1230,
         "Estimator-Monday-1230.csv",
         "Variance-Monday-1230.csv",
         "Number-Monday-1230.csv", Max_Min)
Monday_1230=NULL
Monday_1300=read.csv("Monday-1300-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1300,
         "Estimator-Monday-1300.csv",
         "Variance-Monday-1300.csv",
         "Number-Monday-1300.csv", Max_Min)
Monday_1300=NULL
Monday_1330=read.csv("Monday-1330-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1330,
         "Estimator-Monday-1330.csv",
         "Variance-Monday-1330.csv",
         "Number-Monday-1330.csv",Max_Min)
Monday_1330=NULL
Monday_1400=read.csv("Monday-1400-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1400,
         "Estimator-Monday-1400.csv",
         "Variance-Monday-1400.csv",
         "Number-Monday-1400.csv",Max_Min)
Monday_1400=NULL
Monday_1430=read.csv("Monday-1430-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1430,
         "Estimator-Monday-1430.csv",
         "Variance-Monday-1430.csv",
         "Number-Monday-1430.csv",Max_Min)
Monday_1430=NULL
Monday_1500=read.csv("Monday-1500-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1500,
         "Estimator-Monday-1500.csv",
         "Variance-Monday-1500.csv",
         "Number-Monday-1500.csv",Max_Min)
Monday_1500=NULL
Monday_1530=read.csv("Monday-1530-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1530,
         "Estimator-Monday-1530.csv",
         "Variance-Monday-1530.csv",
         "Number-Monday-1530.csv",Max_Min)
```

```
Monday_1530=NULL
Monday_1600=read.csv("Monday-1600-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1600,
        "Estimator-Monday-1600.csv", "Variance-Monday-1600.csv", "Numbe
Monday_1600=NULL
Monday_1630=read.csv("Monday-1630-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1630,
        "Estimator-Monday-1630.csv",
        "Variance-Monday-1630.csv",
        "Number-Monday-1630.csv",Max_Min)
Monday_1630=NULL
Monday_1700=read.csv("Monday-1700-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1700,
        "Estimator-Monday-1700.csv",
        "Variance-Monday-1700.csv",
        "Number-Monday-1700.csv",Max_Min)
Monday_1700=NULL
Monday_1730=read.csv("Monday-1730-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1730,
        "Estimator-Monday-1730.csv",
        "Variance-Monday-1730.csv",
        "Number-Monday-1730.csv",Max_Min)
Monday_1730=NULL
Monday_1800=read.csv("Monday-1800-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1800,
        "Estimator-Monday-1800.csv",
        "Variance-Monday-1800.csv",
        "Number-Monday-1800.csv",Max_Min)
Monday_1800=NULL
Monday_1830=read.csv("Monday-1830-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1830,
        "Estimator-Monday-1830.csv",
        "Variance-Monday-1830.csv",
        "Number-Monday-1830.csv",Max_Min)
Monday_1830=NULL
Monday_1900=read.csv("Monday-1900-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1900,
        "Estimator-Monday-1900.csv",
        "Variance-Monday-1900.csv",
        "Number-Monday-1900.csv",Max_Min)
Monday_1900=NULL
Monday_1930=read.csv("Monday-1930-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_1930,
        "Estimator-Monday-1930.csv",
        "Variance-Monday-1930.csv",
        "Number-Monday-1930.csv",Max_Min)
```

```
Monday_1930=NULL
Monday_2000=read.csv("Monday-2000-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2000,
         "Estimator-Monday-2000.csv",
         "Variance-Monday-2000.csv",
         "Number-Monday-2000.csv",Max_Min)
Monday_2000=NULL
Monday_2030=read.csv("Monday-2030-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2030,
         "Estimator-Monday-2030.csv",
         "Variance-Monday-2030.csv",
         "Number-Monday-2030.csv",Max_Min)
Monday_2030=NULL
Monday_2100=read.csv("Monday-2100-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2100,
         "Estimator-Monday-2100.csv",
         "Variance-Monday-2100.csv",
         "Number-Monday-2100.csv",Max_Min)
Monday_2100=NULL
Monday_2130=read.csv("Monday-2130-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2130,
         "Estimator-Monday-2130.csv",
         "Variance-Monday-2130.csv",
         "Number-Monday-2130.csv",Max_Min)
Monday_2130=NULL
Monday_2200=read.csv("Monday-2200-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2200,
         "Estimator-Monday-2200.csv",
         "Variance-Monday-2200.csv",
         "Number-Monday-2200.csv",Max_Min)
Monday_2200=NULL
Monday_2230=read.csv("Monday-2230-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2230,
         "Estimator-Monday-2230.csv",
         "Variance-Monday-2230.csv",
         "Number-Monday-2230.csv",Max_Min)
Monday_2230=NULL
Monday_2300=read.csv("Monday-2300-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2300,
         "Estimator-Monday-2300.csv",
         "Variance-Monday-2300.csv",
         "Number-Monday-2300.csv",Max_Min)
Monday_2300=NULL
Monday_2330=read.csv("Monday-2330-2018.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_2330,
         "Estimator-Monday-2330.csv",
```

```
372           "Variance-Monday-2330.csv",
373           "Number-Monday-2330.csv",Max_Min)
374 Monday_2330=NULL
375
376 #Do the same process for the other six days.
```

Program 3: The Computation of the error. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
1  library(dplyr)
2  library(expss)
3
4  Find_Estimator_Var=function(file_matrix, file_mean, name_error1, name_e
5
6  Dataset=file_matrix
7  Dataset=Dataset[,5:9]
8  Dataset=filter(Dataset,Dataset$Trip.Seconds!="")
9  file_mean=file_mean[,-1]
10 Step_Lat=0.003703
11 Step_Long=0.004206
12 Lat_min=41.786
13 Lat_max=Max_Min_Values[2]
14 Long_min=-87.80453
15 Long_max=Max_Min_Values[4]
16
17 n=ceiling((Lat_max-Lat_min)/Step_Lat)
18 m=ceiling((Long_max-Long_min)/Step_Long)
19
20 Error_ij=matrix(0,  nrow = n*m, ncol = m*n)
21 Error_i=rep(0,m*n)
22 b=m*n
23
24 #Computation of the error
25 for (i in 1:n) {
26   Matrix_Help = filter(
27     Dataset, Dataset$Pickup.Centroid.Latitude >=
28         (Lat_min + (i - 1) * Step_Lat)
29     & Dataset$Pickup.Centroid.Latitude <
30         (Lat_min + i * Step_Lat))
31   if (nrow(as.matrix(Matrix_Help)) == 0) {
32     next
```

```
  }
  for (j in 1:m) {
    Matrix_Help2 = filter(Matrix_Help,
        Matrix_Help$Pickup.Centroid.Longitude >=
                (Long_min + (j - 1) * Step_Long)
        & Matrix_Help$Pickup.Centroid.Longitude <
        (Long_min + j * Step_Long)
    )
    if (nrow(as.matrix(Matrix_Help2)) == 0) {
      next
    }
    for (k in 1:n) {
      Matrix_Help3 = filter(Matrix_Help2,
        Matrix_Help2$Dropoff.Centroid.Latitude >=
                (Lat_min + (k - 1) * Step_Lat)
        &  Matrix_Help2$Dropoff.Centroid.Latitude <
                (Lat_min + k * Step_Lat)
      )
      if (nrow(as.matrix(Matrix_Help3)) == 0) {
        next
      }
      for (l in 1:m) {
        Matrix_Help4 = filter(Matrix_Help3,
                Matrix_Help3$Dropoff.Centroid.Longitude >=
                        (Long_min + (l - 1) * Step_Long)
                & Matrix_Help3$Dropoff.Centroid.Longitude <
                        (Long_min + l * Step_Long)
        )
        if (nrow(as.matrix(Matrix_Help4)) == 0) {
          next
        }
        if(file_mean
                [(i - 1) * m + j, (k - 1) * m + l]==0){next}
        Daten = as.vector(Matrix_Help4[, 1])
        Daten2 = as.numeric(
                sub(",", "", Daten, fixed = TRUE))
        n=length(Daten2)
        Daten2=abs(Daten2-rep(file_mean
                [(i - 1) * m + j, (k - 1) * m + l],n))
        Error_ij[(i - 1) * m + j, (k - 1) * m + l]
                =sum(Daten2)/n
      }
    }
  }
}
for (a in 1:b) {
```

```
  Help=Error_ij[,a]
  if(sum(Help)==0){next}
  Error_i[a]=mean(Help[Help!=0])
}

write.csv(Error_ij, name_error1)
write.csv(Error_i, name_error2)
return(Error_i)
}

#read Data
Max_Min=read.csv("Values-for-estimation.csv")
Max_Min=Max_Min[,2]
Monday_0000=read.csv("Monday-0000-2019.csv")
Estimator_Monday_0000=read.csv("Estimator-Monday-0000.csv")
Est_Mo_0000=Find_Estimator_Var(Monday_0000,
        Estimator_Monday_0000, "Error-ij-Monday-0000.csv",
        "Error-i-Monday-0000.csv", Max_Min)
Monday_0000=NULL
Estimator_Monday_0000=NULL
Monday_0030=read.csv("Monday-0030-2019.csv")
Estimator_Monday_0030=read.csv("Estimator-Monday-0030.csv")
Est_Mo_0030=Find_Estimator_Var(Monday_0030,
        Estimator_Monday_0030, "Error-ij-Monday-0030.csv",
        "Error-i-Monday-0030.csv", Max_Min)
Monday_0030=NULL
Estimator_Monday_0030=NULL
Monday_0100=read.csv("Monday-0100-2019.csv")
Estimator_Monday_0100=read.csv("Estimator-Monday-0100.csv")
Est_Mo_0100=Find_Estimator_Var(Monday_0100,
        Estimator_Monday_0100, "Error-ij-Monday-0100.csv",
        "Error-i-Monday-0100.csv", Max_Min)
Monday_0100=NULL
Estimator_Monday_0100=NULL
Monday_0130=read.csv("Monday-0130-2019.csv")
Estimator_Monday_0130=read.csv("Estimator-Monday-0130.csv")
Est_Mo_0130=Find_Estimator_Var(Monday_0130,
        Estimator_Monday_0130, "Error-ij-Monday-0130.csv",
        "Error-i-Monday-0130.csv", Max_Min)
Monday_0130=NULL
Estimator_Monday_0130=NULL
Monday_0200=read.csv("Monday-0200-2019.csv")
Estimator_Monday_0200=read.csv("Estimator-Monday-0200.csv")
Est_Mo_0200=Find_Estimator_Var(Monday_0200,
        Estimator_Monday_0200, "Error-ij-Monday-0200.csv",
        "Error-i-Monday-0200.csv", Max_Min)
```

```
Monday_0200=NULL
Estimator_Monday_0200=NULL
Monday_0230=read.csv("Monday-0230-2019.csv")
Estimator_Monday_0230=read.csv("Estimator-Monday-0230.csv")
Est_Mo_0230=Find_Estimator_Var(Monday_0230,
        Estimator_Monday_0230, "Error-ij-Monday-0230.csv",
        "Error-i-Monday-0230.csv", Max_Min)
Monday_0230=NULL
Estimator_Monday_0230=NULL
Monday_0300=read.csv("Monday-0300-2019.csv")
Estimator_Monday_0300=read.csv("Estimator-Monday-0300.csv")
Est_Mo_0300=Find_Estimator_Var(Monday_0300,
        Estimator_Monday_0300, "Error-ij-Monday-0300.csv",
        "Error-i-Monday-0300.csv", Max_Min)
Monday_0300=NULL
Estimator_Monday_0300=NULL
Monday_0330=read.csv("Monday-0330-2019.csv")
Estimator_Monday_0330=read.csv("Estimator-Monday-0330.csv")
Est_Mo_0330=Find_Estimator_Var(Monday_0330,
        Estimator_Monday_0330, "Error-ij-Monday-0330.csv",
        "Error-i-Monday-0330.csv", Max_Min)
Monday_0330=NULL
Estimator_Monday_0330=NULL
Monday_0400=read.csv("Monday-0400-2019.csv")
Estimator_Monday_0400=read.csv("Estimator-Monday-0400.csv")
Est_Mo_0400=Find_Estimator_Var(Monday_0400,
        Estimator_Monday_0400, "Error-ij-Monday-0400.csv",
        "Error-i-Monday-0400.csv", Max_Min)
Monday_0400=NULL
Estimator_Monday_0400=NULL
Monday_0430=read.csv("Monday-0430-2019.csv")
Estimator_Monday_0430=read.csv("Estimator-Monday-0430.csv")
Est_Mo_0430=Find_Estimator_Var(Monday_0430,
        Estimator_Monday_0430, "Error-ij-Monday-0430.csv",
        "Error-i-Monday-0430.csv", Max_Min)
Monday_0430=NULL
Estimator_Monday_0430=NULL
Monday_0500=read.csv("Monday-0500-2019.csv")
Estimator_Monday_0500=read.csv("Estimator-Monday-0500.csv")
Est_Mo_0500=Find_Estimator_Var(Monday_0500,
        Estimator_Monday_0500, "Error-ij-Monday-0500.csv",
        "Error-i-Monday-0500.csv", Max_Min)
Monday_0500=NULL
Estimator_Monday_0500=NULL
Monday_0530=read.csv("Monday-0530-2019.csv")
Estimator_Monday_0530=read.csv("Estimator-Monday-0530.csv")
```

```
Est_Mo_0530=Find_Estimator_Var(Monday_0530,
        Estimator_Monday_0530, "Error-ij-Monday-0530.csv",
        "Error-i-Monday-0530.csv", Max_Min)
Monday_0530=NULL
Estimator_Monday_0530=NULL
Monday_0600=read.csv("Monday-0600-2019.csv")
Estimator_Monday_0600=read.csv("Estimator-Monday-0600.csv")
Est_Mo_0600=Find_Estimator_Var(Monday_0600,
        Estimator_Monday_0600, "Error-ij-Monday-0600.csv",
        "Error-i-Monday-0600.csv", Max_Min)
Monday_0600=NULL
Estimator_Monday_0600=NULL
Monday_0630=read.csv("Monday-0630-2019.csv")
Estimator_Monday_0630=read.csv("Estimator-Monday-0630.csv")
Est_Mo_0630=Find_Estimator_Var(Monday_0630,
        Estimator_Monday_0630, "Error-ij-Monday-0630.csv",
        "Error-i-Monday-0630.csv", Max_Min)
Monday_0630=NULL
Estimator_Monday_0630=NULL
Monday_0700=read.csv("Monday-0700-2019.csv")
Estimator_Monday_0700=read.csv("Estimator-Monday-0700.csv")
Est_Mo_0700=Find_Estimator_Var(Monday_0700,
        Estimator_Monday_0700, "Error-ij-Monday-0700.csv",
        "Error-i-Monday-0700.csv", Max_Min)
Monday_0700=NULL
Estimator_Monday_0700=NULL
Monday_0730=read.csv("Monday-0730-2019.csv")
Estimator_Monday_0730=read.csv("Estimator-Monday-0730.csv")
Est_Mo_0730=Find_Estimator_Var(Monday_0730,
        Estimator_Monday_0730, "Error-ij-Monday-0730.csv",
        "Error-i-Monday-0730.csv", Max_Min)
Monday_0730=NULL
Estimator_Monday_0730=NULL
Monday_0800=read.csv("Monday-0800-2019.csv")
Estimator_Monday_0800=read.csv("Estimator-Monday-0800.csv")
Est_Mo_0800=Find_Estimator_Var(Monday_0800,
        Estimator_Monday_0800, "Error-ij-Monday-0800.csv",
        "Error-i-Monday-0800.csv", Max_Min)
Monday_0800=NULL
Estimator_Monday_0800=NULL
Monday_0830=read.csv("Monday-0830-2019.csv")
Estimator_Monday_0830=read.csv("Estimator-Monday-0830.csv")
Est_Mo_0830=Find_Estimator_Var(Monday_0830,
        Estimator_Monday_0830, "Error-ij-Monday-0830.csv",
        "Error-i-Monday-0830.csv", Max_Min)
Monday_0830=NULL
```

```
Estimator_Monday_0830=NULL
Monday_0900=read.csv("Monday-0900-2019.csv")
Estimator_Monday_0900=read.csv("Estimator-Monday-0900.csv")
Est_Mo_0900=Find_Estimator_Var(Monday_0900,
        Estimator_Monday_0900, "Error-ij-Monday-0900.csv",
        "Error-i-Monday-0900.csv", Max_Min)
Monday_0900=NULL
Estimator_Monday_0900=NULL
Monday_0930=read.csv("Monday-0930-2019.csv")
Estimator_Monday_0930=read.csv("Estimator-Monday-0930.csv")
Est_Mo_0930=Find_Estimator_Var(Monday_0930,
        Estimator_Monday_0930, "Error-ij-Monday-0930.csv",
        "Error-i-Monday-0930.csv", Max_Min)
Monday_0930=NULL
Estimator_Monday_0930=NULL
Monday_1000=read.csv("Monday-1000-2019.csv")
Estimator_Monday_1000=read.csv("Estimator-Monday-1000.csv")
Est_Mo_1000=Find_Estimator_Var(Monday_1000,
        Estimator_Monday_1000, "Error-ij-Monday-1000.csv",
        "Error-i-Monday-1000.csv", Max_Min)
Monday_1000=NULL
Estimator_Monday_1000=NULL
Monday_1030=read.csv("Monday-1030-2019.csv")
Estimator_Monday_1030=read.csv("Estimator-Monday-1030.csv")
Est_Mo_1030=Find_Estimator_Var(Monday_1030,
        Estimator_Monday_1030, "Error-ij-Monday-1030.csv",
        "Error-i-Monday-1030.csv", Max_Min)
Monday_1030=NULL
Estimator_Monday_1030=NULL
Monday_1100=read.csv("Monday-1100-2019.csv")
Estimator_Monday_1100=read.csv("Estimator-Monday-1100.csv")
Est_Mo_1100=Find_Estimator_Var(Monday_1100,
        Estimator_Monday_1100, "Error-ij-Monday-1100.csv",
        "Error-i-Monday-1100.csv", Max_Min)
Monday_1100=NULL
Estimator_Monday_1100=NULL
Monday_1130=read.csv("Monday-1130-2019.csv")
Estimator_Monday_1130=read.csv("Estimator-Monday-1130.csv")
Est_Mo_1130=Find_Estimator_Var(Monday_1130,
        Estimator_Monday_1130, "Error-ij-Monday-1130.csv",
        "Error-i-Monday-1130.csv", Max_Min)
Monday_1130=NULL
Estimator_Monday_1130=NULL
Monday_1200=read.csv("Monday-1200-2019.csv")
Estimator_Monday_1200=read.csv("Estimator-Monday-1200.csv")
Est_Mo_1200=Find_Estimator_Var(Monday_1200,
```

```
        Estimator_Monday_1200, "Error-ij-Monday-1200.csv",
        "Error-i-Monday-1200.csv", Max_Min)
Monday_1200=NULL
Estimator_Monday_1200=NULL
Monday_1230=read.csv("Monday-1230-2019.csv")
Estimator_Monday_1230=read.csv("Estimator-Monday-1230.csv")
Est_Mo_1230=Find_Estimator_Var(Monday_1230,
        Estimator_Monday_1230, "Error-ij-Monday-1230.csv",
        "Error-i-Monday-1230.csv", Max_Min)
Monday_1230=NULL
Estimator_Monday_1230=NULL
Monday_1300=read.csv("Monday-1300-2019.csv")
Estimator_Monday_1300=read.csv("Estimator-Monday-1300.csv")
Est_Mo_1300=Find_Estimator_Var(Monday_1300,
        Estimator_Monday_1300, "Error-ij-Monday-1300.csv",
        "Error-i-Monday-1300.csv", Max_Min)
Monday_1300=NULL
Estimator_Monday_1300=NULL
Monday_1330=read.csv("Monday-1330-2019.csv")
Estimator_Monday_1330=read.csv("Estimator-Monday-1330.csv")
Est_Mo_1330=Find_Estimator_Var(Monday_1330,
        Estimator_Monday_1330, "Error-ij-Monday-1330.csv",
        "Error-i-Monday-1330.csv", Max_Min)
Monday_1330=NULL
Estimator_Monday_1330=NULL
Monday_1400=read.csv("Monday-1400-2019.csv")
Estimator_Monday_1400=read.csv("Estimator-Monday-1400.csv")
Est_Mo_1400=Find_Estimator_Var(Monday_1400,
        Estimator_Monday_1400, "Error-ij-Monday-1400.csv",
        "Error-i-Monday-1400.csv", Max_Min)
Monday_1400=NULL
Estimator_Monday_1400=NULL
Monday_1430=read.csv("Monday-1430-2019.csv")
Estimator_Monday_1430=read.csv("Estimator-Monday-1430.csv")
Est_Mo_1430=Find_Estimator_Var(Monday_1430,
        Estimator_Monday_1430, "Error-ij-Monday-1430.csv",
        "Error-i-Monday-1430.csv", Max_Min)
Monday_1430=NULL
Estimator_Monday_1430=NULL
Monday_1500=read.csv("Monday-1500-2019.csv")
Estimator_Monday_1500=read.csv("Estimator-Monday-1500.csv")
Est_Mo_1500=Find_Estimator_Var(Monday_1500,
        Estimator_Monday_1500, "Error-ij-Monday-1500.csv",
        "Error-i-Monday-1500.csv", Max_Min)
Monday_1500=NULL
Estimator_Monday_1500=NULL
```

```
Monday_1530=read.csv("Monday-1530-2019.csv")
Estimator_Monday_1530=read.csv("Estimator-Monday-1530.csv")
Est_Mo_1530=Find_Estimator_Var(Monday_1530,
        Estimator_Monday_1530, "Error-ij-Monday-1530.csv",
        "Error-i-Monday-1530.csv", Max_Min)
Monday_1530=NULL
Estimator_Monday_1530=NULL
Monday_1600=read.csv("Monday-1600-2019.csv")
Estimator_Monday_1600=read.csv("Estimator-Monday-1600.csv")
Est_Mo_1600=Find_Estimator_Var(Monday_1600,
        Estimator_Monday_1600, "Error-ij-Monday-1600.csv",
        "Error-i-Monday-1600.csv", Max_Min)
Monday_1600=NULL
Estimator_Monday_1600=NULL
Monday_1630=read.csv("Monday-1630-2019.csv")
Estimator_Monday_1630=read.csv("Estimator-Monday-1630.csv")
Est_Mo_1630=Find_Estimator_Var(Monday_1630,
        Estimator_Monday_1630, "Error-ij-Monday-1630.csv",
        "Error-i-Monday-1630.csv", Max_Min)
Monday_1630=NULL
Estimator_Monday_1630=NULL
Monday_1700=read.csv("Monday-1700-2019.csv")
Estimator_Monday_1700=read.csv("Estimator-Monday-1700.csv")
Est_Mo_1700=Find_Estimator_Var(Monday_1700,
        Estimator_Monday_1700, "Error-ij-Monday-1700.csv",
        "Error-i-Monday-1700.csv", Max_Min)
Monday_1700=NULL
Estimator_Monday_1700=NULL
Monday_1730=read.csv("Monday-1730-2019.csv")
Estimator_Monday_1730=read.csv("Estimator-Monday-1730.csv")
Est_Mo_1730=Find_Estimator_Var(Monday_1730,
        Estimator_Monday_1730, "Error-ij-Monday-1730.csv",
        "Error-i-Monday-1730.csv", Max_Min)
Monday_1730=NULL
Estimator_Monday_1730=NULL
Monday_1800=read.csv("Monday-1800-2019.csv")
Estimator_Monday_1800=read.csv("Estimator-Monday-1800.csv")
Est_Mo_1800=Find_Estimator_Var(Monday_1800,
        Estimator_Monday_1800, "Error-ij-Monday-1800.csv",
        "Error-i-Monday-1800.csv", Max_Min)
Monday_1800=NULL
Estimator_Monday_1800=NULL
Monday_1830=read.csv("Monday-1830-2019.csv")
Estimator_Monday_1830=read.csv("Estimator-Monday-1830.csv")
Est_Mo_1830=Find_Estimator_Var(Monday_1830,
        Estimator_Monday_1830, "Error-ij-Monday-1830.csv",
```

```
        "Error-i-Monday-1830.csv", Max_Min)
Monday_1830=NULL
Estimator_Monday_1830=NULL
Monday_1900=read.csv("Monday-1900-2019.csv")
Estimator_Monday_1900=read.csv("Estimator-Monday-1900.csv")
Est_Mo_1900=Find_Estimator_Var(Monday_1900,
        Estimator_Monday_1900, "Error-ij-Monday-1900.csv",
        "Error-i-Monday-1900.csv", Max_Min)
Monday_1900=NULL
Estimator_Monday_1900=NULL
Monday_1930=read.csv("Monday-1930-2019.csv")
Estimator_Monday_1930=read.csv("Estimator-Monday-1930.csv")
Est_Mo_1930=Find_Estimator_Var(Monday_1930,
        Estimator_Monday_1930, "Error-ij-Monday-1930.csv",
        "Error-i-Monday-1930.csv", Max_Min)
Monday_1930=NULL
Estimator_Monday_1930=NULL
Monday_2000=read.csv("Monday-2000-2019.csv")
Estimator_Monday_2000=read.csv("Estimator-Monday-2000.csv")
Est_Mo_2000=Find_Estimator_Var(Monday_2000,
        Estimator_Monday_2000, "Error-ij-Monday-2000.csv",
        "Error-i-Monday-2000.csv", Max_Min)
Monday_2000=NULL
Estimator_Monday_2000=NULL
Monday_2030=read.csv("Monday-2030-2019.csv")
Estimator_Monday_2030=read.csv("Estimator-Monday-2030.csv")
Est_Mo_2030=Find_Estimator_Var(Monday_2030,
        Estimator_Monday_2030, "Error-ij-Monday-2030.csv",
        "Error-i-Monday-2030.csv", Max_Min)
Monday_2030=NULL
Estimator_Monday_2030=NULL
Monday_2100=read.csv("Monday-2100-2019.csv")
Estimator_Monday_2100=read.csv("Estimator-Monday-2100.csv")
Est_Mo_2100=Find_Estimator_Var(Monday_2100,
        Estimator_Monday_2100, "Error-ij-Monday-2100.csv",
        "Error-i-Monday-2100.csv", Max_Min)
Monday_2100=NULL
Estimator_Monday_2100=NULL
Monday_2130=read.csv("Monday-2130-2019.csv")
Estimator_Monday_2130=read.csv("Estimator-Monday-2130.csv")
Est_Mo_2130=Find_Estimator_Var(Monday_2130,
        Estimator_Monday_2130, "Error-ij-Monday-2130.csv",
        "Error-i-Monday-2130.csv", Max_Min)
Monday_2130=NULL
Estimator_Monday_2130=NULL
Monday_2200=read.csv("Monday-2200-2019.csv")
```

```
Estimator_Monday_2200=read.csv("Estimator-Monday-2200.csv")
Est_Mo_2200=Find_Estimator_Var(Monday_2200,
        Estimator_Monday_2200, "Error-ij-Monday-2200.csv",
        "Error-i-Monday-2200.csv", Max_Min)
Monday_2200=NULL
Estimator_Monday_2200=NULL
Monday_2230=read.csv("Monday-2230-2019.csv")
Estimator_Monday_2230=read.csv("Estimator-Monday-2230.csv")
Est_Mo_2230=Find_Estimator_Var(Monday_2230,
        Estimator_Monday_2230, "Error-ij-Monday-2230.csv",
        "Error-i-Monday-2230.csv", Max_Min)
Monday_2230=NULL
Estimator_Monday_2230=NULL
Monday_2300=read.csv("Monday-2300-2019.csv")
Estimator_Monday_2300=read.csv("Estimator-Monday-2300.csv")
Est_Mo_2300=Find_Estimator_Var(Monday_2300,
        Estimator_Monday_2300, "Error-ij-Monday-2300.csv",
        "Error-i-Monday-2300.csv", Max_Min)
Monday_2300=NULL
Estimator_Monday_2300=NULL
Monday_2330=read.csv("Monday-2330-2019.csv")
Estimator_Monday_2330=read.csv("Estimator-Monday-2330.csv")
Est_Mo_2330=Find_Estimator_Var(Monday_2330,
        Estimator_Monday_2330, "Error-ij-Monday-2330.csv",
        "Error-i-Monday-2330.csv", Max_Min)
Monday_2330=NULL
Estimator_Monday_2330=NULL

#Do the same process for the other six days.
```

Program 4: Computation of the standardized difference. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
library(dplyr)
library(expss)

Find_k=function(Day1_mean, Day2_mean, Day1_var, Day1_n,
        Day2_var, Day2_n, matrix_name){
  Mean_1=read.csv(Day1_mean)
  Mean_1=Mean_1[,-1]
  Mean_2=read.csv(Day2_mean)
  Mean_2=Mean_2[,-1]
  Var_1=read.csv(Day1_var)
```

```
  Var_1=Var_1[,-1]
  Var_2=read.csv(Day2_var)
  Var_2=Var_2[,-1]
  Num_1=read.csv(Day1_n)
  Num_1=Num_1[,-1]
  Num_2=read.csv(Day2_n)
  Num_2=Num_2[,-1]
  Numerator=abs(Mean_1-Mean_2)
  Denominator_1=sqrt(Var_1/Num_1)
  Denominator_1=as.matrix(Denominator_1)
  Denominator_2=sqrt(Var_2/Num_2)
  Denominator_2=as.matrix(Denominator_2)
  Matrix_Help=Numerator/(Denominator_1+Denominator_2)
  write.csv(Matrix_Help,file =  matrix_name)
  k=max(Matrix_Help, na.rm = TRUE)
  return(k)
}

k=rep(0,48)
k[1]=Find_k("Estimator-Monday-0000.csv",
              "Estimator-Tuesday-0000.csv",
              "Variance-Monday-0000.csv",
              "Variance-Tuesday-0000.csv",
              "Number-Monday-0000.csv",
              "Number-Tuesday-0000.csv",
              "k-Matrix-Mo-Tu-0000.csv")
k[2]=Find_k("Estimator-Monday-0030.csv",
              "Estimator-Tuesday-0030.csv",
              "Variance-Monday-0030.csv",
              "Variance-Tuesday-0030.csv",
              "Number-Monday-0030.csv",
              "Number-Tuesday-0030.csv",
              "k-Matrix-Mo-Tu-0030.csv")
k[3]=Find_k("Estimator-Monday-0100.csv",
             "Estimator-Tuesday-0100.csv",
              "Variance-Monday-0100.csv",
              "Variance-Tuesday-0100.csv",
              "Number-Monday-0100.csv",
              "Number-Tuesday-0100.csv",
              "k-Matrix-Mo-Tu-0100.csv")
k[4]=Find_k("Estimator-Monday-0130.csv",
             "Estimator-Tuesday-0130.csv",
              "Variance-Monday-0130.csv",
              "Variance-Tuesday-0130.csv",
              "Number-Monday-0130.csv",
              "Number-Tuesday-0130.csv",
```

```
                "k-Matrix-Mo-Tu-0130.csv")
k[5]=Find_k("Estimator-Monday-0200.csv",
              "Estimator-Tuesday-0200.csv",
               "Variance-Monday-0200.csv",
               "Variance-Tuesday-0200.csv",
               "Number-Monday-0200.csv",
               "Number-Tuesday-0200.csv",
               "k-Matrix-Mo-Tu-0200.csv")
k[6]=Find_k("Estimator-Monday-0230.csv",
              "Estimator-Tuesday-0230.csv",
               "Variance-Monday-0230.csv",
               "Variance-Tuesday-0230.csv",
               "Number-Monday-0230.csv",
               "Number-Tuesday-0230.csv",
               "k-Matrix-Mo-Tu-0230.csv")
k[7]=Find_k("Estimator-Monday-0300.csv",
              "Estimator-Tuesday-0300.csv",
               "Variance-Monday-0300.csv",
               "Variance-Tuesday-0300.csv",
               "Number-Monday-0300.csv",
               "Number-Tuesday-0300.csv",
               "k-Matrix-Mo-Tu-0300.csv")
k[8]=Find_k("Estimator-Monday-0330.csv",
              "Estimator-Tuesday-0330.csv",
               "Variance-Monday-0330.csv",
               "Variance-Tuesday-0330.csv",
               "Number-Monday-0330.csv",
               "Number-Tuesday-0330.csv",
               "k-Matrix-Mo-Tu-0330.csv")
k[9]=Find_k("Estimator-Monday-0400.csv",
              "Estimator-Tuesday-0400.csv",
               "Variance-Monday-0400.csv",
               "Variance-Tuesday-0400.csv",
               "Number-Monday-0400.csv",
               "Number-Tuesday-0400.csv",
               "k-Matrix-Mo-Tu-0400.csv")
k[10]=Find_k("Estimator-Monday-0430.csv",
               "Estimator-Tuesday-0430.csv",
               "Variance-Monday-0430.csv",
               "Variance-Tuesday-0430.csv",
               "Number-Monday-0430.csv",
               "Number-Tuesday-0430.csv",
               "k-Matrix-Mo-Tu-0430.csv")
k[11]=Find_k("Estimator-Monday-0500.csv",
               "Estimator-Tuesday-0500.csv",
               "Variance-Monday-0500.csv",
```

```
                "Variance-Tuesday-0500.csv",
                "Number-Monday-0500.csv",
                "Number-Tuesday-0500.csv",
                "k-Matrix-Mo-Tu-0500.csv")
k[12]=Find_k("Estimator-Monday-0530.csv",
                "Estimator-Tuesday-0530.csv",
                "Variance-Monday-0530.csv",
                "Variance-Tuesday-0530.csv",
                "Number-Monday-0530.csv",
                "Number-Tuesday-0530.csv",
                "k-Matrix-Mo-Tu-0530.csv")
k[13]=Find_k("Estimator-Monday-0600.csv",
                "Estimator-Tuesday-0600.csv",
                "Variance-Monday-0600.csv",
                "Variance-Tuesday-0600.csv",
                "Number-Monday-0600.csv",
                "Number-Tuesday-0600.csv",
                "k-Matrix-Mo-Tu-0600.csv")
k[14]=Find_k("Estimator-Monday-0630.csv",
                "Estimator-Tuesday-0630.csv",
                "Variance-Monday-0630.csv",
                "Variance-Tuesday-0630.csv",
                "Number-Monday-0630.csv",
                "Number-Tuesday-0630.csv",
                "k-Matrix-Mo-Tu-0630.csv")
k[15]=Find_k("Estimator-Monday-0700.csv",
                "Estimator-Tuesday-0700.csv",
                "Variance-Monday-0700.csv",
                "Variance-Tuesday-0700.csv",
                "Number-Monday-0700.csv",
                "Number-Tuesday-0700.csv",
                "k-Matrix-Mo-Tu-0700.csv")
k[16]=Find_k("Estimator-Monday-0730.csv",
                "Estimator-Tuesday-0730.csv",
                "Variance-Monday-0730.csv",
                "Variance-Tuesday-0730.csv",
                "Number-Monday-0730.csv",
                "Number-Tuesday-0730.csv",
                "k-Matrix-Mo-Tu-0730.csv")
k[17]=Find_k("Estimator-Monday-0800.csv",
                 "Estimator-Tuesday-0800.csv",
                 "Variance-Monday-0800.csv",
                 "Variance-Tuesday-0800.csv",
                 "Number-Monday-0800.csv",
                 "Number-Tuesday-0800.csv",
                 "k-Matrix-Mo-Tu-0800.csv")
```

```
k[18]=Find_k("Estimator-Monday-0830.csv",
             "Estimator-Tuesday-0830.csv",
             "Variance-Monday-0830.csv",
             "Variance-Tuesday-0830.csv",
             "Number-Monday-0830.csv",
             "Number-Tuesday-0830.csv",
             "k-Matrix-Mo-Tu-0830.csv")
k[19]=Find_k("Estimator-Monday-0900.csv",
             "Estimator-Tuesday-0900.csv",
             "Variance-Monday-0900.csv",
             "Variance-Tuesday-0900.csv",
             "Number-Monday-0900.csv",
             "Number-Tuesday-0900.csv",
             "k-Matrix-Mo-Tu-0900.csv")
k[20]=Find_k("Estimator-Monday-0930.csv",
             "Estimator-Tuesday-0930.csv",
             "Variance-Monday-0930.csv",
             "Variance-Tuesday-0930.csv",
             "Number-Monday-0930.csv",
             "Number-Tuesday-0930.csv",
             "k-Matrix-Mo-Tu-0930.csv")
k[21]=Find_k("Estimator-Monday-1000.csv",
            "Estimator-Tuesday-1000.csv",
            "Variance-Monday-1000.csv",
            "Variance-Tuesday-1000.csv",
            "Number-Monday-1000.csv",
            "Number-Tuesday-1000.csv",
            "k-Matrix-Mo-Tu-1000.csv")
k[22]=Find_k("Estimator-Monday-1030.csv",
            "Estimator-Tuesday-1030.csv",
            "Variance-Monday-1030.csv",
            "Variance-Tuesday-1030.csv",
            "Number-Monday-1030.csv",
            "Number-Tuesday-1030.csv",
            "k-Matrix-Mo-Tu-1030.csv")
k[23]=Find_k("Estimator-Monday-1100.csv",
            "Estimator-Tuesday-1100.csv",
            "Variance-Monday-1100.csv",
            "Variance-Tuesday-1100.csv",
            "Number-Monday-1100.csv",
            "Number-Tuesday-1100.csv",
            "k-Matrix-Mo-Tu-1100.csv")
k[24]=Find_k("Estimator-Monday-1130.csv",
            "Estimator-Tuesday-1130.csv",
            "Variance-Monday-1130.csv",
            "Variance-Tuesday-1130.csv",
```

```
                "Number-Monday-1130.csv",
                "Number-Tuesday-1130.csv",
                "k-Matrix-Mo-Tu-1130.csv")
k[25]=Find_k("Estimator-Monday-1200.csv",
                "Estimator-Tuesday-1200.csv",
                "Variance-Monday-1200.csv",
                "Variance-Tuesday-1200.csv",
                "Number-Monday-1200.csv",
                "Number-Tuesday-1200.csv",
                "k-Matrix-Mo-Tu-1200.csv")
k[26]=Find_k("Estimator-Monday-1230.csv",
                "Estimator-Tuesday-1230.csv",
                "Variance-Monday-1230.csv",
                "Variance-Tuesday-1230.csv",
                "Number-Monday-1230.csv",
                "Number-Tuesday-1230.csv",
                "k-Matrix-Mo-Tu-1230.csv")
k[27]=Find_k("Estimator-Monday-1300.csv",
                "Estimator-Tuesday-1300.csv",
                "Variance-Monday-1300.csv",
                "Variance-Tuesday-1300.csv",
                "Number-Monday-1300.csv",
                "Number-Tuesday-1300.csv",
                "k-Matrix-Mo-Tu-1300.csv")
k[28]=Find_k("Estimator-Monday-1330.csv",
                "Estimator-Tuesday-1330.csv",
                "Variance-Monday-1330.csv",
                "Variance-Tuesday-1330.csv",
                "Number-Monday-1330.csv",
                "Number-Tuesday-1330.csv",
                "k-Matrix-Mo-Tu-1330.csv")
k[29]=Find_k("Estimator-Monday-1400.csv",
                "Estimator-Tuesday-1400.csv",
                "Variance-Monday-1400.csv",
                "Variance-Tuesday-1400.csv",
                "Number-Monday-1400.csv",
                "Number-Tuesday-1400.csv",
                "k-Matrix-Mo-Tu-1400.csv")
k[30]=Find_k("Estimator-Monday-1430.csv",
                 "Estimator-Tuesday-1430.csv",
                 "Variance-Monday-1430.csv",
                 "Variance-Tuesday-1430.csv",
                 "Number-Monday-1430.csv",
                 "Number-Tuesday-1430.csv",
                 "k-Matrix-Mo-Tu-1430.csv")
k[31]=Find_k("Estimator-Monday-1500.csv",
```

```
                "Estimator-Tuesday-1500.csv",
                "Variance-Monday-1500.csv",
                "Variance-Tuesday-1500.csv",
                "Number-Monday-1500.csv",
                "Number-Tuesday-1500.csv",
                "k-Matrix-Mo-Tu-1500.csv")
k[32]=Find_k("Estimator-Monday-1530.csv",
                "Estimator-Tuesday-1530.csv",
                "Variance-Monday-1530.csv",
                "Variance-Tuesday-1530.csv",
                "Number-Monday-1530.csv",
                "Number-Tuesday-1530.csv",
                "k-Matrix-Mo-Tu-1530.csv")
k[33]=Find_k("Estimator-Monday-1600.csv",
                "Estimator-Tuesday-1600.csv",
                "Variance-Monday-1600.csv",
                "Variance-Tuesday-1600.csv",
                "Number-Monday-1600.csv",
                "Number-Tuesday-1600.csv",
                "k-Matrix-Mo-Tu-1600.csv")
k[34]=Find_k("Estimator-Monday-1630.csv",
                "Estimator-Tuesday-1630.csv",
                "Variance-Monday-1630.csv",
                "Variance-Tuesday-1630.csv",
                "Number-Monday-1630.csv",
                "Number-Tuesday-1630.csv",
                "k-Matrix-Mo-Tu-1630.csv")
k[35]=Find_k("Estimator-Monday-1700.csv",
                "Estimator-Tuesday-1700.csv",
                "Variance-Monday-1700.csv",
                "Variance-Tuesday-1700.csv",
                "Number-Monday-1700.csv",
                "Number-Tuesday-1700.csv",
                "k-Matrix-Mo-Tu-1700.csv")
k[36]=Find_k("Estimator-Monday-1730.csv",
                "Estimator-Tuesday-1730.csv",
                "Variance-Monday-1730.csv",
                "Variance-Tuesday-1730.csv",
                "Number-Monday-1730.csv",
                "Number-Tuesday-1730.csv",
                "k-Matrix-Mo-Tu-1730.csv")
k[37]=Find_k("Estimator-Monday-1800.csv",
                "Estimator-Tuesday-1800.csv",
                "Variance-Monday-1800.csv",
                "Variance-Tuesday-1800.csv",
                "Number-Monday-1800.csv",
```

```
                "Number-Tuesday-1800.csv",
                "k-Matrix-Mo-Tu-1800.csv")
k[38]=Find_k("Estimator-Monday-1830.csv",
                "Estimator-Tuesday-1830.csv",
                "Variance-Monday-1830.csv",
                "Variance-Tuesday-1830.csv",
                "Number-Monday-1830.csv",
                "Number-Tuesday-1830.csv",
                "k-Matrix-Mo-Tu-1830.csv")
k[39]=Find_k("Estimator-Monday-1900.csv",
                "Estimator-Tuesday-1900.csv",
                "Variance-Monday-1900.csv",
                "Variance-Tuesday-1900.csv",
                "Number-Monday-1900.csv",
                "Number-Tuesday-1900.csv",
                "k-Matrix-Mo-Tu-1900.csv")
k[40]=Find_k("Estimator-Monday-1930.csv",
                "Estimator-Tuesday-1930.csv",
                "Variance-Monday-1930.csv",
                "Variance-Tuesday-1930.csv",
                "Number-Monday-1930.csv",
                "Number-Tuesday-1930.csv",
                "k-Matrix-Mo-Tu-1930.csv")
k[41]=Find_k("Estimator-Monday-2000.csv",
                "Estimator-Tuesday-2000.csv",
                "Variance-Monday-2000.csv",
                "Variance-Tuesday-2000.csv",
                "Number-Monday-2000.csv",
                "Number-Tuesday-2000.csv",
                "k-Matrix-Mo-Tu-2000.csv")
k[42]=Find_k("Estimator-Monday-2030.csv",
                "Estimator-Tuesday-2030.csv",
                "Variance-Monday-2030.csv",
                "Variance-Tuesday-2030.csv",
                "Number-Monday-2030.csv",
                "Number-Tuesday-2030.csv",
                "k-Matrix-Mo-Tu-2030.csv")
k[43]=Find_k("Estimator-Monday-2100.csv",
                "Estimator-Tuesday-2100.csv",
                "Variance-Monday-2100.csv",
                "Variance-Tuesday-2100.csv",
                "Number-Monday-2100.csv",
                "Number-Tuesday-2100.csv",
                "k-Matrix-Mo-Tu-2100.csv")
k[44]=Find_k("Estimator-Monday-2130.csv",
                "Estimator-Tuesday-2130.csv",
```

```
                "Variance-Monday-2130.csv",
                "Variance-Tuesday-2130.csv",
                "Number-Monday-2130.csv",
                "Number-Tuesday-2130.csv",
                "k-Matrix-Mo-Tu-2130.csv")
k[45]=Find_k("Estimator-Monday-2200.csv",
                "Estimator-Tuesday-2200.csv",
                "Variance-Monday-2200.csv",
                "Variance-Tuesday-2200.csv",
                "Number-Monday-2200.csv",
                "Number-Tuesday-2200.csv",
                "k-Matrix-Mo-Tu-2200.csv")
k[46]=Find_k("Estimator-Monday-2230.csv",
                "Estimator-Tuesday-2230.csv",
                "Variance-Monday-2230.csv",
                "Variance-Tuesday-2230.csv",
                "Number-Monday-2230.csv",
                "Number-Tuesday-2230.csv",
                "k-Matrix-Mo-Tu-2230.csv")
k[47]=Find_k("Estimator-Monday-2300.csv",
                "Estimator-Tuesday-2300.csv",
                "Variance-Monday-2300.csv",
                "Variance-Tuesday-2300.csv",
                "Number-Monday-2300.csv",
                "Number-Tuesday-2300.csv",
                "k-Matrix-Mo-Tu-2300.csv")
k[48]=Find_k("Estimator-Monday-2330.csv",
                "Estimator-Tuesday-2330.csv",
                "Variance-Monday-2330.csv",
                "Variance-Tuesday-2330.csv",
                "Number-Monday-2330.csv",
                "Number-Tuesday-2330.csv",
                "k-Matrix-Mo-Tu-2330.csv")

# Do the whole process for every ensemble of
        #two days. This leads to 21 ensemble.
```

Program 5: Pooling estimators together.Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
library(dplyr)
```

```
library(expss)

Pooling=function(k_Matrix11, k_Matrix22, k_Matrix33,
        k_Matrix44, k_Matrix55, k_Matrix66,
    Day1_mean1, Day2_mean2, Day3_mean3,
    Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
    Day1_error1, Day1_Data1, k, Pool1,
    Pool2, Pool3, Pool4, Pool5, Pool6){

  m=65
  n=64
  Step_Lat=0.003703
  Step_Long=0.004206
  Lat_min=41.786
  Lat_max=42.02122
  Long_min=-87.80453
  Long_max=-87.53139
  mn=m*n
  Error_time_day1=0
  d=length(Pool1)
  if(d>1){
    for (c in 2:d) {
      i1=Pool1[c]%%mn
      j1=(Pool1[c]%/%mn)+1
      if(is.na(Day1_mean[i1,j1])==TRUE){next}
      if(is.na(Day2_mean[i1,j1])==TRUE){
        ticker =1
      }
      else{
        Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day2_mean[i1,j1]
        ticker=2
      }
      if(Pool1[c]%in%Pool2==TRUE){
        if(is.na(Day3_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day3_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool1[c],Pool2)
        Pool2=Pool2[-v]
      }
      if(Pool1[c]%in%Pool3==TRUE){
        if(is.na(Day4_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day4_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool1[c],Pool3)
```

```
48          Pool3=Pool3[-v]
49        }
50        if(Pool1[c]%in%Pool4==TRUE){
51          if(is.na(Day5_mean[i1,j1])==FALSE){
52            Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
53            ticker=ticker+1
54          }
55          v=match(Pool1[c],Pool4)
56          Pool4=Pool4[-v]
57        }
58        if(Pool1[c]%in%Pool5==TRUE){
59          if(is.na(Day6_mean[i1,j1])==FALSE){
60            Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
61            ticker=ticker+1
62          }
63          v=match(Pool1[c],Pool5)
64          Pool5=Pool5[-v]
65        }
66        if(Pool1[c]%in%Pool6==TRUE){
67          if(is.na(Day7_mean[i1,j1])==FALSE){
68            Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
69            ticker=ticker+1
70          }
71          v=match(Pool1[c],Pool6)
72          Pool6=Pool6[-v]
73        }
74        Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
75        i=(i1%/%m)+1
76        j=i1%%m
77        k=(j1%/%m)+1
78        l=j1%%m
79        Day1_help=filter(Day1_Data,
80                         Day1_Data$Pickup.Centroid.Latitude >=
81                                 (Lat_min + (i - 1) * Step_Lat)&
82                         Day1_Data$Pickup.Centroid.Latitude <
83                                 (Lat_min + i * Step_Lat) &
84                         Day1_Data$Pickup.Centroid.Longitude >=
85                                 (Long_min + (j - 1) * Step_Long) &
86                         Day1_Data$Pickup.Centroid.Longitude <
87                                 (Long_min + j * Step_Long) &
88                         Day1_Data$Dropoff.Centroid.Latitude >=
89                                 (Lat_min + (k - 1) * Step_Lat) &
90                         Day1_Data$Dropoff.Centroid.Latitude <
91                                 (Lat_min + k * Step_Lat) &
92                         Day1_Data$Dropoff.Centroid.Longitude >=
93                                 (Long_min + (l - 1) * Step_Long)&
```

```
                Day1_Data$Dropoff.Centroid.Longitude <
                        (Long_min + l * Step_Long))
      if (nrow(as.matrix(Day1_help)) != 0) {
        Daten = as.vector(Day1_help[, 1])
        Daten2 = as.numeric(sub
                (",", "", Daten, fixed = TRUE))
        n=length(Daten2)
        Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
        Day1_error[(i - 1) * m + j, (k - 1) * m + l]
        = sum(Daten2)/n
      }
    }
  }
  d=length(Pool2)
  if(d>1){
    for (c in 2:d) {
      i1=Pool2[c]%%mn
      j1=(Pool2[c]%/%mn)+1
      if(is.na(Day1_mean[i1,j1])==TRUE){next}
      if(is.na(Day3_mean[i1,j1])==TRUE){
        ticker =1
      }
      else{
        Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day3_mean[i1,j1]
        ticker=2
      }
      if(Pool2[c]%in%Pool3){
        if(is.na(Day4_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day4_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool2[c],Pool3)
        Pool3=Pool3[-v]
      }
      if(Pool2[c]%in%Pool4){
        if(is.na(Day5_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool2[c],Pool4)
        Pool4=Pool4[-v]
      }
      if(Pool2[c]%in%Pool5){
        if(is.na(Day6_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
          ticker=ticker+1
```

```
140           }
141           v=match(Pool2[c],Pool5)
142           Pool5=Pool5[-v]
143         }
144         if(Pool2[c]%in%Pool6){
145           if(is.na(Day7_mean[i1,j1])==FALSE){
146             Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
147             ticker=ticker+1
148           }
149           v=match(Pool2[c],Pool6)
150           Pool6=Pool6[-v]
151         }
152         Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
153         i=(i1%/%m)+1
154         j=i1%%m
155         k=(j1%/%m)+1
156         l=j1%%m
157                     Day1_Data$Pickup.Centroid.Latitude >=
158                             (Lat_min + (i - 1) * Step_Lat)&
159                     Day1_Data$Pickup.Centroid.Latitude <
160                             (Lat_min + i * Step_Lat) &
161                     Day1_Data$Pickup.Centroid.Longitude >=
162                             (Long_min + (j - 1) * Step_Long) &
163                     Day1_Data$Pickup.Centroid.Longitude <
164                             (Long_min + j * Step_Long) &
165                     Day1_Data$Dropoff.Centroid.Latitude >=
166                             (Lat_min + (k - 1) * Step_Lat) &
167                     Day1_Data$Dropoff.Centroid.Latitude <
168                             (Lat_min + k * Step_Lat) &
169                     Day1_Data$Dropoff.Centroid.Longitude >=
170                             (Long_min + (l - 1) * Step_Long)&
171                     Day1_Data$Dropoff.Centroid.Longitude <
172                             (Long_min + l * Step_Long))
173         if (nrow(as.matrix(Day1_help)) != 0) {
174           Daten = as.vector(Day1_help[, 1])
175           Daten2 = as.numeric(sub
176                   (",", "", Daten, fixed = TRUE))
177           n=length(Daten2)
178           Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
179           Day1_error[(i - 1) * m + j, (k - 1) * m + l]
180           = sum(Daten2)/n
181         }
182       }
183     }
184     d=length(Pool3)
185     if(d>1){
```

```
186     for (c in 2:d) {
187       i1=Pool3[c]%%mn
188       j1=(Pool3[c]%/%mn)+1
189       if(is.na(Day1_mean[i1,j1])==TRUE){next}
190       if(is.na(Day4_mean[i1,j1])==TRUE){
191         ticker =1
192       }
193       else{
194         Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day4_mean[i1,j1]
195         ticker=2
196       }
197       if(Pool3[c]%in%Pool4){
198         if(is.na(Day5_mean[i1,j1])==FALSE){
199           Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
200           ticker=ticker+1
201         }
202         v=match(Pool3[c],Pool4)
203         Pool4=Pool4[-v]
204       }
205       if(Pool3[c]%in%Pool5){
206         if(is.na(Day6_mean[i1,j1])==FALSE){
207           Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
208           ticker=ticker+1
209         }
210         v=match(Pool3[c],Pool5)
211         Pool5=Pool5[-v]
212       }
213       if(Pool3[c]%in%Pool6){
214         if(is.na(Day7_mean[i1,j1])==FALSE){
215           Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
216           ticker=ticker+1
217         }
218         v=match(Pool3[c],Pool6)
219         Pool6=Pool6[-v]
220       }
221       Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
222       i=(i1%/%m)+1
223       j=i1%%m
224       k=(j1%/%m)+1
225       l=j1%%m
226                 Day1_Data$Pickup.Centroid.Latitude >=
227                         (Lat_min + (i - 1) * Step_Lat)&
228                 Day1_Data$Pickup.Centroid.Latitude <
229                         (Lat_min + i * Step_Lat) &
230                 Day1_Data$Pickup.Centroid.Longitude >=
231                         (Long_min + (j - 1) * Step_Long) &
```

```
                Day1_Data$Pickup.Centroid.Longitude <
                        (Long_min + j * Step_Long) &
                Day1_Data$Dropoff.Centroid.Latitude >=
                        (Lat_min + (k - 1) * Step_Lat) &
                Day1_Data$Dropoff.Centroid.Latitude <
                        (Lat_min + k * Step_Lat) &
                Day1_Data$Dropoff.Centroid.Longitude >=
                        (Long_min + (l - 1) * Step_Long)&
                Day1_Data$Dropoff.Centroid.Longitude <
                        (Long_min + l * Step_Long))
      if (nrow(as.matrix(Day1_help)) != 0) {
        Daten = as.vector(Day1_help[, 1])
        Daten2 = as.numeric(sub
                (",", "", Daten, fixed = TRUE))
        n=length(Daten2)
        Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
        Day1_error[(i - 1) * m + j, (k - 1) * m + l]
        = sum(Daten2)/n
      }
    }
  }
  d=length(Pool4)
  if(d>1){
    for (c in 2:d) {
      i1=Pool4[c]%%mn
      j1=(Pool4[c]%/%mn)+1
      if(is.na(Day1_mean[i1,j1])==TRUE){next}
      if(is.na(Day5_mean[i1,j1])==TRUE){
        ticker =1
      }
      else{
        Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
        ticker=2
      }
      if(Pool4[c]%in%Pool5){
        if(is.na(Day6_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool4[c],Pool5)
        Pool5=Pool5[-v]
      }
      if(Pool4[c]%in%Pool6){
        if(is.na(Day7_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
          ticker=ticker+1
```

```
278           }
279           v=match(Pool4[c],Pool6)
280           Pool6=Pool6[-v]
281         }
282         Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
283         i=(i1%/%m)+1
284         j=i1%%m
285         k=(j1%/%m)+1
286         l=j1%%m
287           Day1_help=filter(Day1_Data,
288                    Day1_Data$Pickup.Centroid.Latitude >=
289                            (Lat_min + (i - 1) * Step_Lat)&
290                    Day1_Data$Pickup.Centroid.Latitude <
291                            (Lat_min + i * Step_Lat) &
292                    Day1_Data$Pickup.Centroid.Longitude >=
293                            (Long_min + (j - 1) * Step_Long) &
294                    Day1_Data$Pickup.Centroid.Longitude <
295                            (Long_min + j * Step_Long) &
296                    Day1_Data$Dropoff.Centroid.Latitude >=
297                            (Lat_min + (k - 1) * Step_Lat) &
298                    Day1_Data$Dropoff.Centroid.Latitude <
299                            (Lat_min + k * Step_Lat) &
300                    Day1_Data$Dropoff.Centroid.Longitude >=
301                            (Long_min + (l - 1) * Step_Long)&
302                    Day1_Data$Dropoff.Centroid.Longitude <
303                            (Long_min + l * Step_Long))
304         if (nrow(as.matrix(Day1_help)) != 0) {
305           Daten = as.vector(Day1_help[, 1])
306           Daten2 = as.numeric(sub
307                    (",", "", Daten, fixed = TRUE))
308           n=length(Daten2)
309           Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
310           Day1_error[(i - 1) * m + j, (k - 1) * m + l]
311           = sum(Daten2)/n
312         }
313       }
314     }
315     d=length(Pool5)
316     if(d>1){
317       for (c in 2:d) {
318         i1=Pool5[c]%%mn
319         j1=(Pool5[c]%/%mn)+1
320         if(is.na(Day1_mean[i1,j1])==TRUE){next}
321         if(is.na(Day6_mean[i1,j1])==TRUE){
322           ticker =1
323         }
```

```
      else{
        Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
        ticker=2
      }
      if(Pool5[c]%in%Pool6){
        if(is.na(Day7_mean[i1,j1])==FALSE){
          Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
          ticker=ticker+1
        }
        v=match(Pool5[c],Pool6)
        Pool6=Pool6[-v]
      }
      Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
      i=(i1%/%m)+1
      j=i1%%m
      k=(j1%/%m)+1
      l=j1%%m
        Day1_help=filter(Day1_Data,
                 Day1_Data$Pickup.Centroid.Latitude >=
                         (Lat_min + (i - 1) * Step_Lat)&
                 Day1_Data$Pickup.Centroid.Latitude <
                         (Lat_min + i * Step_Lat) &
                 Day1_Data$Pickup.Centroid.Longitude >=
                         (Long_min + (j - 1) * Step_Long) &
                 Day1_Data$Pickup.Centroid.Longitude <
                         (Long_min + j * Step_Long) &
                 Day1_Data$Dropoff.Centroid.Latitude >=
                         (Lat_min + (k - 1) * Step_Lat) &
                 Day1_Data$Dropoff.Centroid.Latitude <
                         (Lat_min + k * Step_Lat) &
                 Day1_Data$Dropoff.Centroid.Longitude >=
                         (Long_min + (l - 1) * Step_Long)&
                 Day1_Data$Dropoff.Centroid.Longitude <
                         (Long_min + l * Step_Long))
      if (nrow(as.matrix(Day1_help)) != 0) {
        Daten = as.vector(Day1_help[, 1])
        Daten2 = as.numeric(sub
                 (",", "", Daten, fixed = TRUE))
        n=length(Daten2)
        Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
        Day1_error[(i - 1) * m + j, (k - 1) * m + l]
        = sum(Daten2)/n
      }
    }
  }
  d=length(Pool6)
```

```
  if(d>1){
    for (c in 2:d) {
      i1=Pool6[c]%%mn
      j1=(Pool6[c]%/%mn)+1
      if(is.na(Day1_mean[i1,j1])==TRUE){next}
      if(is.na(Day7_mean[i1,j1])==TRUE){
        ticker =1
      }
      else{
        Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
        ticker=2
      }
      Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
      i=(i1%/%m)-1
      j=i1%%m+1
      k=(j1%/%m)-1
      l=j1%%m
        Day1_help=filter(Day1_Data,
                 Day1_Data$Pickup.Centroid.Latitude >=
                         (Lat_min + (i - 1) * Step_Lat)&
                 Day1_Data$Pickup.Centroid.Latitude <
                         (Lat_min + i * Step_Lat) &
                 Day1_Data$Pickup.Centroid.Longitude >=
                         (Long_min + (j - 1) * Step_Long) &
                 Day1_Data$Pickup.Centroid.Longitude <
                         (Long_min + j * Step_Long) &
                 Day1_Data$Dropoff.Centroid.Latitude >=
                         (Lat_min + (k - 1) * Step_Lat) &
                 Day1_Data$Dropoff.Centroid.Latitude <
                         (Lat_min + k * Step_Lat) &
                 Day1_Data$Dropoff.Centroid.Longitude >=
                         (Long_min + (l - 1) * Step_Long)&
                 Day1_Data$Dropoff.Centroid.Longitude <
                         (Long_min + l * Step_Long))
      if (nrow(as.matrix(Day1_help)) != 0) {
        Daten = as.vector(Day1_help[, 1])
        Daten2 = as.numeric(sub
                 (",", "", Daten, fixed = TRUE))
        n=length(Daten2)
        Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
        Day1_error[(i - 1) * m + j, (k - 1) * m + l]
        = sum(Daten2)/n
      }
    }
  }
  Error_i=rep(0,mn)
```

```
  for (e in 1:mn) {
    Help=Day1_error[,e]

    Error_i[e]=mean(Help, na.rm = TRUE)
    if(is.nan(Error_i[e])==TRUE){Error_i[e]=NA}
  }
  Error_time_day1=mean(Error_i, na.rm = TRUE)
  return(Error_time_day1)
}
Error_Mo_16=rep(0,48)
k=16

k_Matrix1=read.csv("k-Matrix-Mo-Tu-0000.csv")
k_Matrix1=k_Matrix1[,-1]
k_Matrix2=read.csv("k-Matrix-Mo-We-0000.csv")
k_Matrix2=k_Matrix2[,-1]
k_Matrix3=read.csv("k-Matrix-Mo-Th-0000.csv")
k_Matrix3=k_Matrix3[,-1]
k_Matrix4=read.csv("k-Matrix-Mo-Fr-0000.csv")
k_Matrix4=k_Matrix4[,-1]
k_Matrix5=read.csv("k-Matrix-Mo-Sa-0000.csv")
k_Matrix5=k_Matrix5[,-1]
k_Matrix6=read.csv("k-Matrix-Mo-Su-0000.csv")
k_Matrix6=k_Matrix6[,-1]
Day1_mean=read.csv("Estimator-Monday-0000.csv")
Day1_mean=Day1_mean[,-1]
Day2_mean=read.csv("Estimator-Tuesday-0000.csv")
Day2_mean=Day2_mean[,-1]
Day3_mean=read.csv("Estimator-Wednesday-0000.csv")
Day3_mean=Day3_mean[,-1]
Day4_mean=read.csv("Estimator-Thursday-0000.csv")
Day4_mean=Day4_mean[,-1]
Day5_mean=read.csv("Estimator-Friday-0000.csv")
Day5_mean=Day5_mean[,-1]
Day6_mean=read.csv("Estimator-Saturday-0000.csv")
Day6_mean=Day6_mean[,-1]
Day7_mean=read.csv("Estimator-Sunday-0000.csv")
Day7_mean=Day7_mean[,-1]
Day1_error=read.csv("Error-ij-Monday-0000.csv")
Day1_error=Day1_error[,-1]
Day1_Data=read.csv("Monday-0000-2019.csv")
Day1_Data=Day1_Data[,5:9]
Pool1=which(k_Matrix1>=0 &k_Matrix1<=k)
Pool1=c(0,Pool1)
Pool2=which(k_Matrix2>=0 &k_Matrix2<=k)
Pool2=c(0,Pool2)
```

```
Pool3=which(k_Matrix3>=0 &k_Matrix3<=k)
Pool3=c(0,Pool3)
Pool4=which(k_Matrix4>=0 &k_Matrix4<=k)
Pool4=c(0,Pool4)
Pool5=which(k_Matrix5>=0 &k_Matrix5<=k)
Pool5=c(0,Pool5)
Pool6=which(k_Matrix6>=0 &k_Matrix6<=k)
Pool6=c(0,Pool6)
Error_Mo_16[1]=Pooling(k_Matrix11, k_Matrix22, k_Matrix33,
    k_Matrix44, k_Matrix55, k_Matrix66,
    Day1_mean1, Day2_mean2, Day3_mean3,
    Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
    Day1_error1, Day1_Data1, k ,Pool1, Pool2,
    Pool3, Pool4, Pool5, Pool6)
k_Matrix1=NULL
k_Matrix2=NULL
k_Matrix3=NULL
k_Matrix4=NULL
k_Matrix5=NULL
k_Matrix6=NULL
Day1_mean=NULL
Day2_mean=NULL
Day3_mean=NULL
Day4_mean=NULL
Day5_mean=NULL
Day6_mean=NULL
Day7_mean=NULL
Day1_error=NULL
Day1_Data=NULL
Pool1=NULL
Pool2=NULL
Pool3=NULL
Pool4=NULL
Pool5=NULL
Pool6=NULL

k_Matrix1=read.csv("k-Matrix-Mo-Tu-0030.csv")
k_Matrix1=k_Matrix1[,-1]
k_Matrix2=read.csv("k-Matrix-Mo-We-0030.csv")
k_Matrix2=k_Matrix2[,-1]
k_Matrix3=read.csv("k-Matrix-Mo-Th-0030.csv")
k_Matrix3=k_Matrix3[,-1]
k_Matrix4=read.csv("k-Matrix-Mo-Fr-0030.csv")
k_Matrix4=k_Matrix4[,-1]
k_Matrix5=read.csv("k-Matrix-Mo-Sa-0030.csv")
k_Matrix5=k_Matrix5[,-1]
```

```
k_Matrix6=read.csv("k-Matrix-Mo-Su-0030.csv")
k_Matrix6=k_Matrix6[,-1]
Day1_mean=read.csv("Estimator-Monday-0030.csv")
Day1_mean=Day1_mean[,-1]
Day2_mean=read.csv("Estimator-Tuesday-0030.csv")
Day2_mean=Day2_mean[,-1]
Day3_mean=read.csv("Estimator-Wednesday-0030.csv")
Day3_mean=Day3_mean[,-1]
Day4_mean=read.csv("Estimator-Thursday-0030.csv")
Day4_mean=Day4_mean[,-1]
Day5_mean=read.csv("Estimator-Friday-0030.csv")
Day5_mean=Day5_mean[,-1]
Day6_mean=read.csv("Estimator-Saturday-0030.csv")
Day6_mean=Day6_mean[,-1]
Day7_mean=read.csv("Estimator-Sunday-0030.csv")
Day7_mean=Day7_mean[,-1]
Day1_error=read.csv("Error-ij-Monday-0030.csv")
Day1_error=Day1_error[,-1]
Day1_Data=read.csv("Monday-0030-2019.csv")
Day1_Data=Day1_Data[,5:9]
Pool1=which(k_Matrix1>=0 &k_Matrix1<=k)
Pool1=c(0,Pool1)
Pool2=which(k_Matrix2>=0 &k_Matrix2<=k)
Pool2=c(0,Pool2)
Pool3=which(k_Matrix3>=0 &k_Matrix3<=k)
Pool3=c(0,Pool3)
Pool4=which(k_Matrix4>=0 &k_Matrix4<=k)
Pool4=c(0,Pool4)
Pool5=which(k_Matrix5>=0 &k_Matrix5<=k)
Pool5=c(0,Pool5)
Pool6=which(k_Matrix6>=0 &k_Matrix6<=k)
Pool6=c(0,Pool6)
Error_Mo_16[1]=Pooling(k_Matrix11, k_Matrix22, k_Matrix33,
    k_Matrix44, k_Matrix55, k_Matrix66,
    Day1_mean1, Day2_mean2, Day3_mean3,
    Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
    Day1_error1, Day1_Data1, k ,Pool1, Pool2,
    Pool3, Pool4, Pool5, Pool6)
k_Matrix1=NULL
k_Matrix2=NULL
k_Matrix3=NULL
k_Matrix4=NULL
k_Matrix5=NULL
k_Matrix6=NULL
Day1_mean=NULL
Day2_mean=NULL
```

```
Day3_mean=NULL
Day4_mean=NULL
Day5_mean=NULL
Day6_mean=NULL
Day7_mean=NULL
Day1_error=NULL
Day1_Data=NULL
Pool1=NULL
Pool2=NULL
Pool3=NULL
Pool4=NULL
Pool5=NULL
Pool6=NULL

#Do the same process for every time interval t and
        #for every day d
```

Program 6: The analysis of the error. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```

#Find the error E(t,d) and E(d)

library(dplyr)
library(expss)
Times=c("12am", "12.30am", "1am","1.30am","2am","2.30am",
        "3am","3.30am", "4am","4.30am", "5am","5.30am",
        "6am","6.30am", "7am","7.30am", "8am","8.30am",
        "9am","9.30am", "10am","10.30am", "11am","11.30am",
        "12pm", "12.30pm", "1pm","1.30pm","2pm","2.30pm",
        "3pm","3.30pm", "4pm","4.30pm", "5pm","5.30pm",
        "6pm","6.30pm", "7pm","7.30pm", "8pm","8.30pm",
        "9pm","9.30pm", "10pm","10.30pm", "11pm","11.30pm")

Error_i_Monday_0000=read.csv("Error-i-Monday-0000.csv")
Error_i_Monday_0000=Error_i_Monday_0000[,-1]
Error_i_Monday_0030=read.csv("Error-i-Monday-0030.csv")
Error_i_Monday_0030=Error_i_Monday_0030[,-1]
Error_i_Monday_0100=read.csv("Error-i-Monday-0100.csv")
Error_i_Monday_0100=Error_i_Monday_0100[,-1]
Error_i_Monday_0130=read.csv("Error-i-Monday-0130.csv")
```

```
Error_i_Monday_0130=Error_i_Monday_0130[,-1]
Error_i_Monday_0200=read.csv("Error-i-Monday-0200.csv")
Error_i_Monday_0200=Error_i_Monday_0200[,-1]
Error_i_Monday_0230=read.csv("Error-i-Monday-0230.csv")
Error_i_Monday_0230=Error_i_Monday_0230[,-1]
Error_i_Monday_0300=read.csv("Error-i-Monday-0300.csv")
Error_i_Monday_0300=Error_i_Monday_0300[,-1]
Error_i_Monday_0330=read.csv("Error-i-Monday-0330.csv")
Error_i_Monday_0330=Error_i_Monday_0330[,-1]
Error_i_Monday_0400=read.csv("Error-i-Monday-0400.csv")
Error_i_Monday_0400=Error_i_Monday_0400[,-1]
Error_i_Monday_0430=read.csv("Error-i-Monday-0430.csv")
Error_i_Monday_0430=Error_i_Monday_0430[,-1]
Error_i_Monday_0500=read.csv("Error-i-Monday-0500.csv")
Error_i_Monday_0500=Error_i_Monday_0500[,-1]
Error_i_Monday_0530=read.csv("Error-i-Monday-0530.csv")
Error_i_Monday_0530=Error_i_Monday_0530[,-1]
Error_i_Monday_0600=read.csv("Error-i-Monday-0600.csv")
Error_i_Monday_0600=Error_i_Monday_0600[,-1]
Error_i_Monday_0630=read.csv("Error-i-Monday-0630.csv")
Error_i_Monday_0630=Error_i_Monday_0630[,-1]
Error_i_Monday_0700=read.csv("Error-i-Monday-0700.csv")
Error_i_Monday_0700=Error_i_Monday_0700[,-1]
Error_i_Monday_0730=read.csv("Error-i-Monday-0730.csv")
Error_i_Monday_0730=Error_i_Monday_0730[,-1]
Error_i_Monday_0800=read.csv("Error-i-Monday-0800.csv")
Error_i_Monday_0800=Error_i_Monday_0800[,-1]
Error_i_Monday_0830=read.csv("Error-i-Monday-0830.csv")
Error_i_Monday_0830=Error_i_Monday_0830[,-1]
Error_i_Monday_0900=read.csv("Error-i-Monday-0900.csv")
Error_i_Monday_0900=Error_i_Monday_0900[,-1]
Error_i_Monday_0930=read.csv("Error-i-Monday-0930.csv")
Error_i_Monday_0930=Error_i_Monday_0930[,-1]
Error_i_Monday_1000=read.csv("Error-i-Monday-1000.csv")
Error_i_Monday_1000=Error_i_Monday_1000[,-1]
Error_i_Monday_1030=read.csv("Error-i-Monday-1030.csv")
Error_i_Monday_1030=Error_i_Monday_1030[,-1]
Error_i_Monday_1100=read.csv("Error-i-Monday-1100.csv")
Error_i_Monday_1100=Error_i_Monday_1100[,-1]
Error_i_Monday_1130=read.csv("Error-i-Monday-1130.csv")
Error_i_Monday_1130=Error_i_Monday_1130[,-1]
Error_i_Monday_1200=read.csv("Error-i-Monday-1200.csv")
Error_i_Monday_1200=Error_i_Monday_1200[,-1]
Error_i_Monday_1230=read.csv("Error-i-Monday-1230.csv")
Error_i_Monday_1230=Error_i_Monday_1230[,-1]
Error_i_Monday_1300=read.csv("Error-i-Monday-1300.csv")
```

```
Error_i_Monday_1300=Error_i_Monday_1300[,-1]
Error_i_Monday_1330=read.csv("Error-i-Monday-1330.csv")
Error_i_Monday_1330=Error_i_Monday_1330[,-1]
Error_i_Monday_1400=read.csv("Error-i-Monday-1400.csv")
Error_i_Monday_1400=Error_i_Monday_1400[,-1]
Error_i_Monday_1430=read.csv("Error-i-Monday-1430.csv")
Error_i_Monday_1430=Error_i_Monday_1430[,-1]
Error_i_Monday_1500=read.csv("Error-i-Monday-1500.csv")
Error_i_Monday_1500=Error_i_Monday_1500[,-1]
Error_i_Monday_1530=read.csv("Error-i-Monday-1530.csv")
Error_i_Monday_1530=Error_i_Monday_1530[,-1]
Error_i_Monday_1600=read.csv("Error-i-Monday-1600.csv")
Error_i_Monday_1600=Error_i_Monday_1600[,-1]
Error_i_Monday_1630=read.csv("Error-i-Monday-1630.csv")
Error_i_Monday_1630=Error_i_Monday_1630[,-1]
Error_i_Monday_1700=read.csv("Error-i-Monday-1700.csv")
Error_i_Monday_1700=Error_i_Monday_1700[,-1]
Error_i_Monday_1730=read.csv("Error-i-Monday-1730.csv")
Error_i_Monday_1730=Error_i_Monday_1730[,-1]
Error_i_Monday_1800=read.csv("Error-i-Monday-1800.csv")
Error_i_Monday_1800=Error_i_Monday_1800[,-1]
Error_i_Monday_1830=read.csv("Error-i-Monday-1830.csv")
Error_i_Monday_1830=Error_i_Monday_1830[,-1]
Error_i_Monday_1900=read.csv("Error-i-Monday-1900.csv")
Error_i_Monday_1900=Error_i_Monday_1900[,-1]
Error_i_Monday_1930=read.csv("Error-i-Monday-1930.csv")
Error_i_Monday_1930=Error_i_Monday_1930[,-1]
Error_i_Monday_2000=read.csv("Error-i-Monday-2000.csv")
Error_i_Monday_2000=Error_i_Monday_2000[,-1]
Error_i_Monday_2030=read.csv("Error-i-Monday-2030.csv")
Error_i_Monday_2030=Error_i_Monday_2030[,-1]
Error_i_Monday_2100=read.csv("Error-i-Monday-2100.csv")
Error_i_Monday_2100=Error_i_Monday_2100[,-1]
Error_i_Monday_2130=read.csv("Error-i-Monday-2130.csv")
Error_i_Monday_2130=Error_i_Monday_2130[,-1]
Error_i_Monday_2200=read.csv("Error-i-Monday-2200.csv")
Error_i_Monday_2200=Error_i_Monday_2200[,-1]
Error_i_Monday_2230=read.csv("Error-i-Monday-2230.csv")
Error_i_Monday_2230=Error_i_Monday_2230[,-1]
Error_i_Monday_2300=read.csv("Error-i-Monday-2300.csv")
Error_i_Monday_2300=Error_i_Monday_2300[,-1]
Error_i_Monday_2330=read.csv("Error-i-Monday-2330.csv")
Error_i_Monday_2330=Error_i_Monday_2330[,-1]

Error_Monday=rep(0,48)
Error_Monday[1]=mean(Error_i_Monday_0000,  na.rm = TRUE)
```

```
Error_Monday[2]=mean(Error_i_Monday_0030,  na.rm = TRUE)
Error_Monday[3]=mean(Error_i_Monday_0100,  na.rm = TRUE)
Error_Monday[4]=mean(Error_i_Monday_0130,  na.rm = TRUE)
Error_Monday[5]=mean(Error_i_Monday_0200,  na.rm = TRUE)
Error_Monday[6]=mean(Error_i_Monday_0230,  na.rm = TRUE)
Error_Monday[7]=mean(Error_i_Monday_0300,  na.rm = TRUE)
Error_Monday[8]=mean(Error_i_Monday_0330,  na.rm = TRUE)
Error_Monday[9]=mean(Error_i_Monday_0400,  na.rm = TRUE)
Error_Monday[10]=mean(Error_i_Monday_0430,  na.rm = TRUE)
Error_Monday[11]=mean(Error_i_Monday_0500,  na.rm = TRUE)
Error_Monday[12]=mean(Error_i_Monday_0530,  na.rm = TRUE)
Error_Monday[13]=mean(Error_i_Monday_0600,  na.rm = TRUE)
Error_Monday[14]=mean(Error_i_Monday_0630,  na.rm = TRUE)
Error_Monday[15]=mean(Error_i_Monday_0700,  na.rm = TRUE)
Error_Monday[16]=mean(Error_i_Monday_0730,  na.rm = TRUE)
Error_Monday[17]=mean(Error_i_Monday_0800,  na.rm = TRUE)
Error_Monday[18]=mean(Error_i_Monday_0830,  na.rm = TRUE)
Error_Monday[19]=mean(Error_i_Monday_0900,  na.rm = TRUE)
Error_Monday[20]=mean(Error_i_Monday_0930,  na.rm = TRUE)
Error_Monday[21]=mean(Error_i_Monday_1000,  na.rm = TRUE)
Error_Monday[22]=mean(Error_i_Monday_1030,  na.rm = TRUE)
Error_Monday[23]=mean(Error_i_Monday_1100,  na.rm = TRUE)
Error_Monday[24]=mean(Error_i_Monday_1130,  na.rm = TRUE)
Error_Monday[25]=mean(Error_i_Monday_1200,  na.rm = TRUE)
Error_Monday[26]=mean(Error_i_Monday_1230,  na.rm = TRUE)
Error_Monday[27]=mean(Error_i_Monday_1300,  na.rm = TRUE)
Error_Monday[28]=mean(Error_i_Monday_1330,  na.rm = TRUE)
Error_Monday[29]=mean(Error_i_Monday_1400,  na.rm = TRUE)
Error_Monday[30]=mean(Error_i_Monday_1430,  na.rm = TRUE)
Error_Monday[31]=mean(Error_i_Monday_1500,  na.rm = TRUE)
Error_Monday[32]=mean(Error_i_Monday_1530,  na.rm = TRUE)
Error_Monday[33]=mean(Error_i_Monday_1600,  na.rm = TRUE)
Error_Monday[34]=mean(Error_i_Monday_1630,  na.rm = TRUE)
Error_Monday[35]=mean(Error_i_Monday_1700,  na.rm = TRUE)
Error_Monday[36]=mean(Error_i_Monday_1730,  na.rm = TRUE)
Error_Monday[37]=mean(Error_i_Monday_1800,  na.rm = TRUE)
Error_Monday[38]=mean(Error_i_Monday_1830,  na.rm = TRUE)
Error_Monday[39]=mean(Error_i_Monday_1900,  na.rm = TRUE)
Error_Monday[40]=mean(Error_i_Monday_1930,  na.rm = TRUE)
Error_Monday[41]=mean(Error_i_Monday_2000,  na.rm = TRUE)
Error_Monday[42]=mean(Error_i_Monday_2030,  na.rm = TRUE)
Error_Monday[43]=mean(Error_i_Monday_2100,  na.rm = TRUE)
Error_Monday[44]=mean(Error_i_Monday_2130,  na.rm = TRUE)
Error_Monday[45]=mean(Error_i_Monday_2200,  na.rm = TRUE)
Error_Monday[46]=mean(Error_i_Monday_2230,  na.rm = TRUE)
Error_Monday[47]=mean(Error_i_Monday_2300,  na.rm = TRUE)
```

```
Error_Monday[48]=mean(Error_i_Monday_2330,  na.rm = TRUE)
write.csv(Error_Monday,"Error-Monday.csv")

#Do the same process for every day.

#Plots
par(mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
plot(seq(1,48,1),Error_Monday,
     main = "The Error depending on the time interval t
                 for the several days", type = "l",
     xaxt="n", xlab = "Time Interval Starting Point",
     ylab = "Error E(t,d)", col="red", lwd=2,
     ylim = c(0,2200), pch=19)
axis(1,at = seq(1,48,1),label=Times)
lines(Error_Wednesday, type="l", col="yellow", lwd=2,pch=19)
lines(Error_Thursday, type="l", col="green", lwd=2,pch=19)
lines(Error_Saturday, type="l", col="brown", lwd=2,pch=19)
legend("topright",inset = c(-0.25,0),
    legend=c("Monday","Wednesday", "Thursday", "Saturday"),
    col=c("red", "yellow", "green", "brown"),cex=0.8,lty=1)

plot(seq(1,48,1),Error_Tuesday,
     main = "The Error depending on the time interval t
        for the several days", type = "l",
     xaxt="n", xlab = "Time Interval Starting Point",
     ylab = "Error E(t,d)", col="orange", lwd=2,
     ylim = c(0,9000))
axis(1,at = seq(1,48,1),label=Times)
lines(Error_Friday, type = "l", col="blue", lwd=2)
lines(Error_Sunday,type = "l", col="black", lwd=2)
legend("topright",inset = c(-0.22,0),
    legend=c("Tuesday","Friday",  "Sunday"),
    col=c("orange", "blue", "black"),cex=0.8,lty=1)

Error_Tuesday2=Error_Tuesday
Error_Tuesday2[Error_Tuesday2>1500]=1500
Error_Friday2=Error_Friday
Error_Friday2[Error_Friday2>1500]=1500
Error_Sunday2=Error_Sunday
Error_Sunday2[Error_Sunday2>1500]=1500

plot(seq(1,48,1),Error_Tuesday2,
     main = "The Error depending on the time interval t
        for the several days", type = "l",
     xaxt="n", xlab = "Time Interval Starting Point",
     ylab = "Error E(t,d)", col="orange", lwd=2,
```

```
     ylim = c(0,1500))
axis(1,at = seq(1,48,1),label=Times)
lines(Error_Friday2, type = "l", col="blue", lwd=2)
lines(Error_Sunday2,type = "l", col="black", lwd=2)
legend("topright",inset = c(-0.22,0),
     legend=c("Tuesday","Friday",  "Sunday"),
     col=c("orange", "blue", "black"),cex=0.8,lty=1)
mean(Error_Monday)

plot(seq(1,48,1),Error_Tuesday,
     main = "The Error depending on the time interval t
          for the day Tuesday", type = "l", xaxt="n",
          xlab = "Time Interval Start", ylab = "Error E(t,d)")
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Tuesday)

plot(seq(1,48,1),Error_Wednesday,
    main = "The Error depending on the time interval t
         for the day Wednesday",type = "l", xaxt="n",
         xlab = "Time Interval Start", ylab = "Error E(t,d)")
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Wednesday)

plot(seq(1,48,1),Error_Thursday,
    main = "The Error depending on the time interval t
         for the day Thursday",type = "l", xaxt="n",
         xlab = "Time Interval Start", ylab = "Error E(t,d)")
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Thursday)

plot(seq(1,48,1),Error_Friday,
    main = "The Error depending on the time interval t
        for the day Friday",type = "l", xaxt="n",
         xlab = "Time Interval Start", ylab = "Error E(t,d)")
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Friday)

plot(seq(1,48,1),Error_Saturday,
    type = "l", xaxt="n", xlab = "Time Interval Start",
    ylab = "Error E(t,d)")
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Saturday)

plot(seq(1,48,1),Error_Sunday,
    type = "l", xaxt="n", xlab = "Time Interval Start",
    ylab = "Error E(t,d)")
```

```
axis(1,at = seq(1,48,1),label=Times)
mean(Error_Sunday)

#Boxplots
par(mar=c(5.1, 4.1, 4.1, 2.1), mgp=c(3, 1, 0), las=0)
boxplot(Error_i_Friday_1400[Error_i_Friday_1400>0],
   Error_i_Friday_1730[Error_i_Friday_1730>0],
   Error_i_Friday_2200[Error_i_Friday_2200>0],
   horizontal = TRUE,
   main = "Boxplot of the error E(i,t,d) for
       selected days and time intervals",
       names = c("Friday 2 pm", "Friday 5.30 pm",
          "Friday 10 pm"))

boxplot(Error_i_Friday_1400[Error_i_Friday_1400>0],
   Error_i_Friday_1730[Error_i_Friday_1730>0],
   Error_i_Friday_2200[Error_i_Friday_2200>0],
   horizontal = TRUE, outline = FALSE,
   main = "Boxplot of the error E(i,t,d) for
     selected days and time intervals without outlines",
   names = c("Friday 2pm", "Friday 5.30 pm","Friday 10 pm"))
max(Error_i_Friday_1400[Error_i_Friday_1400>0])

boxplot(Error_i_Sunday_0500[Error_i_Sunday_0500>0],
   horizontal = TRUE,
   main = "Boxplot of the error E(i,t,d)",
   ylab ="Sunday 5 am")
boxplot(Error_i_Tuesday_2100[Error_i_Tuesday_2100>0],
   Error_i_Friday_0300[Error_i_Friday_0300>0],
   horizontal = TRUE,
   main= "Boxplot of the error E(i,t,d) with outliers",
   names=c("Tuesday 9 pm", "Friday 3 am"))

boxplot(Error_i_Tuesday_2100[Error_i_Tuesday_2100>0],
   Error_i_Friday_0300[Error_i_Friday_0300>0],
   horizontal = TRUE,
   main= "Boxplot of the error E(i,t,d) without outliers",
   names=c("Tuesday 9 pm", "Friday 3 am"), outline = FALSE)

Error_Tuesday

#Find number of error E(i,t,d)

number_Monday=rep(0,48)
number_Monday[1]= length(which(Error_i_Monday_0000>0))
number_Monday[2]= length(which(Error_i_Monday_0030>0))
```

```
number_Monday[3]= length(which(Error_i_Monday_0100>0))
number_Monday[4]= length(which(Error_i_Monday_0130>0))
number_Monday[5]= length(which(Error_i_Monday_0200>0))
number_Monday[6]= length(which(Error_i_Monday_0230>0))
number_Monday[7]= length(which(Error_i_Monday_0300>0))
number_Monday[8]= length(which(Error_i_Monday_0330>0))
number_Monday[9]= length(which(Error_i_Monday_0400>0))
number_Monday[10]= length(which(Error_i_Monday_0430>0))
number_Monday[11]= length(which(Error_i_Monday_0500>0))
number_Monday[12]= length(which(Error_i_Monday_0530>0))
number_Monday[13]= length(which(Error_i_Monday_0600>0))
number_Monday[14]= length(which(Error_i_Monday_0630>0))
number_Monday[15]= length(which(Error_i_Monday_0700>0))
number_Monday[16]= length(which(Error_i_Monday_0730>0))
number_Monday[17]= length(which(Error_i_Monday_0800>0))
number_Monday[18]= length(which(Error_i_Monday_0830>0))
number_Monday[19]= length(which(Error_i_Monday_0900>0))
number_Monday[20]= length(which(Error_i_Monday_0930>0))
number_Monday[21]= length(which(Error_i_Monday_1000>0))
number_Monday[22]= length(which(Error_i_Monday_1030>0))
number_Monday[23]= length(which(Error_i_Monday_1100>0))
number_Monday[24]= length(which(Error_i_Monday_1130>0))
number_Monday[25]= length(which(Error_i_Monday_1200>0))
number_Monday[26]= length(which(Error_i_Monday_1230>0))
number_Monday[27]= length(which(Error_i_Monday_1300>0))
number_Monday[28]= length(which(Error_i_Monday_1330>0))
number_Monday[29]= length(which(Error_i_Monday_1400>0))
number_Monday[30]= length(which(Error_i_Monday_1430>0))
number_Monday[31]= length(which(Error_i_Monday_1500>0))
number_Monday[32]= length(which(Error_i_Monday_1530>0))
number_Monday[33]= length(which(Error_i_Monday_1600>0))
number_Monday[34]= length(which(Error_i_Monday_1630>0))
number_Monday[35]= length(which(Error_i_Monday_1700>0))
number_Monday[36]= length(which(Error_i_Monday_1730>0))
number_Monday[37]= length(which(Error_i_Monday_1800>0))
number_Monday[38]= length(which(Error_i_Monday_1830>0))
number_Monday[39]= length(which(Error_i_Monday_1900>0))
number_Monday[40]= length(which(Error_i_Monday_1930>0))
number_Monday[41]= length(which(Error_i_Monday_2000>0))
number_Monday[42]= length(which(Error_i_Monday_2030>0))
number_Monday[43]= length(which(Error_i_Monday_2100>0))
number_Monday[44]= length(which(Error_i_Monday_2130>0))
number_Monday[45]= length(which(Error_i_Monday_2200>0))
number_Monday[46]= length(which(Error_i_Monday_2230>0))
number_Monday[47]= length(which(Error_i_Monday_2300>0))
number_Monday[48]= length(which(Error_i_Monday_2330>0))
```

```
344
345 #Do the same process for every day.
346
347 par(mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
348 plot(seq(1,48,1), number_Monday, type = "l", col = "red",
349    lwd=2, ylim = c(0,120), xaxt="n", xlab = "Time intervals",
350    ylab="Number of error E(i,t,d)",
351    main = "The number of error E(i,t,d) for the seven days")
352 lines( number_Tuesday, type = "l", lwd=2, col="orange")
353 lines( number_Wednesday, type = "l", lwd=2, col="yellow")
354 lines( number_Thursday, type = "l", lwd=2, col="green")
355 lines( number_Friday, type = "l", lwd=2, col="blue")
356 lines( number_Saturday, type = "l", lwd=2, col="brown")
357 lines( number_Sunday, type = "l", lwd=2, col="black")
358 axis(1,at = seq(1,48,1),label=Times)
359 legend("topright",inset = c(-0.22,0),
360    legend=c("Monday", "Tuesday", "Wednesday","Thursday",
361           "Friday", "Saturday", "Sunday"),
362    col=c("red","orange","yellow","green","blue", "purple",
363       "brown", "black"),cex=0.8,lty=1)
364
365 #Analysis of data points:
366 setwd("F:\\FB\\Uni\\10.Semester\\Master-Arbeit\\Roh-Daten")
367 Monday=read.csv("Monday.csv")
368 plot(Monday[,6], Monday[,5],
369     main="The pickup places in terms of
370         latitude and longitude",
371     xlab="Longitude", ylab="Latitude")
372 min(Monday[,5])
373 quantile(Monday[,5],0.01)
374 quantile(Monday[,5],0.02)
375 quantile(Monday[,5],0.05)
376 max(Monday[,5])
377 min(Monday[,6])
378 quantile(Monday[,6],0.01)
379 quantile(Monday[,6],0.05)
380 quantile(Monday[,6],0.1)
381 max(Monday[,6])
382
383 plot(Monday[,8], Monday[,7],
384     main="The dropoff places in terms of
385        latitude and longitude",
386     xlab="Longitude", ylab="Latitude")
387 min(Monday[,7])
388 quantile(Monday[,7],0.01)
389 quantile(Monday[,7],0.02)
```

```
quantile(Monday[,7],0.05)
max(Monday[,7])
min(Monday[,8])
quantile(Monday[,8],0.01)
quantile(Monday[,8],0.05)
quantile(Monday[,8],0.1)
max(Monday[,8])


Mo_1=filter(Monday,
    Monday$Pickup.Centroid.Latitude>41.786 &
    Monday$Dropoff.Centroid.Latitude>41.786 &
    Monday$Pickup.Centroid.Longitude>-87.80453 &
    Monday$Dropoff.Centroid.Longitude>-87.80453)
plot(Mo_1[,6], Mo_1[,5],main="The pickup places in terms
    of latitude and longitude after the cut",
    xlab="Longitude", ylab="Latitude")
min(Mo_1[,5])
max(Mo_1[,5])
min(Mo_1[,6])
max(Mo_1[,6])

plot(Mo_1[,8], Mo_1[,7],main="The dropoff places in terms
    of latitude and longitude after the cut",
    xlab="Longitude", ylab="Latitude")
min(Mo_1[,7])
max(Mo_1[,7])
min(Mo_1[,8])
max(Mo_1[,8])
```

Program 7: The analysis of pooling estimators together. Because of the length of the program only parts of the program are shown. Similar programcode is skipped.

```
library(dplyr)
library(expss)

Error_Monday=read.csv("Error-Monday.csv")
Error_Monday=Error_Monday[,-1]
Error_Tuesday=read.csv("Error-Tuesday.csv")
Error_Tuesday=Error_Tuesday[,-1]
Error_Wednesday=read.csv("Error-Wednesday.csv")
```

```
Error_Wednesday=Error_Wednesday[,-1]
Error_Thursday=read.csv("Error-Thursday.csv")
Error_Thursday=Error_Thursday[,-1]
Error_Friday=read.csv("Error-Friday.csv")
Error_Friday=Error_Friday[,-1]
Error_Saturday=read.csv("Error-Saturday.csv")
Error_Saturday=Error_Saturday[,-1]
Error_Sunday=read.csv("Error-Sunday.csv")
Error_Sunday=Error_Sunday[,-1]

Error_Mo_1=read.csv("Error-Mo-1.csv")
Error_Mo_1=Error_Mo_1[,-1]
Error_Tu_1=read.csv("Error-Tu-1.csv")
Error_Tu_1=Error_Tu_1[,-1]
Error_We_1=read.csv("Error-We-1.csv")
Error_We_1=Error_We_1[,-1]
Error_Th_1=read.csv("Error-Th-1.csv")
Error_Th_1=Error_Th_1[,-1]
Error_Fr_1=read.csv("Error-Fr-1.csv")
Error_Fr_1=Error_Fr_1[,-1]
Error_Sa_1=read.csv("Error-Sa-1.csv")
Error_Sa_1=Error_Sa_1[,-1]
Error_Su_1=read.csv("Error-Su-1.csv")
Error_Su_1=Error_Su_1[,-1]

Error_Mo_2=read.csv("Error-Mo-2.csv")
Error_Mo_2=Error_Mo_2[,-1]
Error_Tu_2=read.csv("Error-Tu-2.csv")
Error_Tu_2=Error_Tu_2[,-1]
Error_We_2=read.csv("Error-We-2.csv")
Error_We_2=Error_We_2[,-1]
Error_Th_2=read.csv("Error-Th-2.csv")
Error_Th_2=Error_Th_2[,-1]
Error_Fr_2=read.csv("Error-Fr-2.csv")
Error_Fr_2=Error_Fr_2[,-1]
Error_Sa_2=read.csv("Error-Sa-2.csv")
Error_Sa_2=Error_Sa_2[,-1]
Error_Su_2=read.csv("Error-Su-2.csv")
Error_Su_2=Error_Su_2[,-1]

Error_Mo_4=read.csv("Error-Mo-4.csv")
Error_Mo_4=Error_Mo_4[,-1]
Error_Tu_4=read.csv("Error-Tu-4.csv")
Error_Tu_4=Error_Tu_4[,-1]
Error_We_4=read.csv("Error-We-4.csv")
Error_We_4=Error_We_4[,-1]
```

```
Error_Th_4=read.csv("Error-Th-4.csv")
Error_Th_4=Error_Th_4[,-1]
Error_Fr_4=read.csv("Error-Fr-4.csv")
Error_Fr_4=Error_Fr_4[,-1]
Error_Sa_4=read.csv("Error-Sa-4.csv")
Error_Sa_4=Error_Sa_4[,-1]
Error_Su_4=read.csv("Error-Su-4.csv")
Error_Su_4=Error_Su_4[,-1]

Error_Mo_8=read.csv("Error-Mo-8.csv")
Error_Mo_8=Error_Mo_8[,-1]
Error_Tu_8=read.csv("Error-Tu-8.csv")
Error_Tu_8=Error_Tu_8[,-1]
Error_We_8=read.csv("Error-We-8.csv")
Error_We_8=Error_We_8[,-1]
Error_Th_8=read.csv("Error-Th-8.csv")
Error_Th_8=Error_Th_8[,-1]
Error_Fr_8=read.csv("Error-Fr-8.csv")
Error_Fr_8=Error_Fr_8[,-1]
Error_Sa_8=read.csv("Error-Sa-8.csv")
Error_Sa_8=Error_Sa_8[,-1]
Error_Su_8=read.csv("Error-Su-8.csv")
Error_Su_8=Error_Su_8[,-1]

Error_Mo_16=read.csv("Error-Mo-16.csv")
Error_Mo_16=Error_Mo_16[,-1]
Error_Tu_16=read.csv("Error-Tu-16.csv")
Error_Tu_16=Error_Tu_16[,-1]
Error_We_16=read.csv("Error-We-16.csv")
Error_We_16=Error_We_16[,-1]
Error_Th_16=read.csv("Error-Th-16.csv")
Error_Th_16=Error_Th_16[,-1]
Error_Fr_16=read.csv("Error-Fr-16.csv")
Error_Fr_16=Error_Fr_16[,-1]
Error_Sa_16=read.csv("Error-Sa-16.csv")
Error_Sa_16=Error_Sa_16[,-1]
Error_Su_16=read.csv("Error-Su-16.csv")
Error_Su_16=Error_Su_16[,-1]

Monday_k=rep(0,6)
Monday_k[1]=mean(Error_Monday)
Monday_k[2]=mean(Error_Mo_1)
Monday_k[3]=mean(Error_Mo_2)
Monday_k[4]=mean(Error_Mo_4)
Monday_k[5]=mean(Error_Mo_8)
Monday_k[6]=mean(Error_Mo_16)
```

```
101
102 Tuesday_k=rep(0,6)
103 Tuesday_k[1]=mean(Error_Tuesday)
104 Tuesday_k[2]=mean(Error_Tu_1)
105 Tuesday_k[3]=mean(Error_Tu_2)
106 Tuesday_k[4]=mean(Error_Tu_4)
107 Tuesday_k[5]=mean(Error_Tu_8)
108 Tuesday_k[6]=mean(Error_Tu_16)
109
110 Wednesday_k=rep(0,6)
111 Wednesday_k[1]=mean(Error_Wednesday)
112 Wednesday_k[2]=mean(Error_We_1)
113 Wednesday_k[3]=mean(Error_We_2)
114 Wednesday_k[4]=mean(Error_We_4)
115 Wednesday_k[5]=mean(Error_We_8)
116 Wednesday_k[6]=mean(Error_We_16)
117
118 Thursday_k=rep(0,6)
119 Thursday_k[1]=mean(Error_Thursday)
120 Thursday_k[2]=mean(Error_Th_1)
121 Thursday_k[3]=mean(Error_Th_2)
122 Thursday_k[4]=mean(Error_Th_4)
123 Thursday_k[5]=mean(Error_Th_8)
124 Thursday_k[6]=mean(Error_Th_16)
125
126 Friday_k=rep(0,6)
127 Friday_k[1]=mean(Error_Friday)
128 Friday_k[2]=mean(Error_Fr_1)
129 Friday_k[3]=mean(Error_Fr_2)
130 Friday_k[4]=mean(Error_Fr_4)
131 Friday_k[5]=mean(Error_Fr_8)
132 Friday_k[6]=mean(Error_Fr_16)
133
134 Saturday_k=rep(0,6)
135 Saturday_k[1]=mean(Error_Saturday)
136 Saturday_k[2]=mean(Error_Sa_1)
137 Saturday_k[3]=mean(Error_Sa_2)
138 Saturday_k[4]=mean(Error_Sa_4)
139 Saturday_k[5]=mean(Error_Sa_8)
140 Saturday_k[6]=mean(Error_Sa_16)
141
142 Sunday_k=rep(0,6)
143 Sunday_k[1]=mean(Error_Sunday)
144 Sunday_k[2]=mean(Error_Su_1)
145 Sunday_k[3]=mean(Error_Su_2)
146 Sunday_k[4]=mean(Error_Su_4)
```

```
Sunday_k[5]=mean(Error_Su_8)
Sunday_k[6]=mean(Error_Su_16)

#Plots of overall error depending on k
par(mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
k_s=c(0,1,2,4,8,16)
plot(seq(1,6,1),Monday_k, type = "l", ylim = c(250,700),
   main = "The overall error E(d,k) of the different days",
   col="red", lwd=2, xlab= "The values of k",
   ylab = "Error E(d,k)", xaxt="n")
lines(Tuesday_k, col="orange",lwd=2)
lines(Wednesday_k, col="yellow",lwd=2)
lines(Thursday_k, col="green",lwd=2)
lines(Friday_k, col="blue",lwd=2)
lines(Saturday_k, col="brown",lwd=2)
lines(Sunday_k,col="black",lwd=2)
axis(1, at=seq(1,6,1),labels = k_s)
legend("topright",inset = c(-0.22,0),
   legend=c("Monday", "Tuesday", "Wednesday",
       "Thursday","Friday", "Saturday", "Sunday"),
   col=c("red","orange","yellow","green","blue",
      "brown", "black"),cex=0.8,lty=1)

Monday_k_rel=Monday_k/Monday_k[1]
Tuesday_k_rel=Tuesday_k/Tuesday_k[1]
Wednesday_k_rel=Wednesday_k/Wednesday_k[1]
Thursday_k_rel=Thursday_k/Thursday_k[1]
Friday_k_rel=Friday_k/Friday_k[1]
Saturday_k_rel=Saturday_k/Saturday_k[1]
Sunday_k_rel=Sunday_k/Sunday_k[1]

k_s=c(0,1,2,4,8,16)
plot(seq(1,6,1),Monday_k_rel, type = "l", ylim = c(0.4,1),
   main = "The ratio E(d,k)/E(d,0) for the respective days",
   col="red", lwd=2, xlab= "The values of k",
   ylab = "Error E(d,k)/E(d)", xaxt="n")
lines(Tuesday_k_rel, col="orange",lwd=2)
lines(Wednesday_k_rel, col="yellow",lwd=2)
lines(Thursday_k_rel, col="green",lwd=2)
lines(Friday_k_rel, col="blue",lwd=2)
lines(Saturday_k_rel, col="brown",lwd=2)
lines(Sunday_k_rel,col="black",lwd=2)
axis(1, at=seq(1,6,1),labels = k_s)
legend("topright",inset = c(-0.22,0),
   legend=c("Monday", "Tuesday", "Wednesday",
      "Thursday","Friday", "Saturday", "Sunday"),
```

```
   col=c("red","orange","yellow","green",
      "blue", "brown","black"),cex=0.8,lty=1)

par(mar=c(5.1, 4.1, 4.1, 6.1), xpd=TRUE)
Times=c("12am", "12.30am", "1am","1.30am","2am","2.30am",
        "3am","3.30am", "4am","4.30am", "5am","5.30am",
        "6am","6.30am", "7am","7.30am", "8am","8.30am",
        "9am","9.30am", "10am","10.30am", "11am","11.30am",
        "12pm", "12.30pm", "1pm","1.30pm","2pm","2.30pm",
        "3pm","3.30pm", "4pm","4.30pm", "5pm","5.30pm",
        "6pm","6.30pm", "7pm","7.30pm", "8pm","8.30pm",
        "9pm","9.30pm", "10pm","10.30pm", "11pm","11.30pm")
plot(seq(1,48,1), Error_Monday, col="red",
   main = "The error E(t,d,k) for the day Monday
      for different values of k",
   xlab = "The time intervals",
   ylab = "The Error E(t,d,k)", xaxt="n", type = "l",
   lwd=2, ylim = c(0,1500))
lines(Error_Mo_1, col="orange", lwd=2)
lines(Error_Mo_2, col="yellow", lwd=2)
lines(Error_Mo_4, col="green", lwd=2)
lines(Error_Mo_8, col="blue", lwd=2)
lines(Error_Mo_16, col="brown", lwd=2)
axis(1,at = seq(1,48,1),label=Times)
legend("topright",inset = c(-0.16,0),
   legend=c("k=0", "k=1", "k=2","k=4","k=8", "k=16"),
   col=c("red","orange","yellow","green","blue",
      "brown", "black"),cex=0.8,lty=1)

plot(seq(1,48,1), Error_Wednesday, col="red",
   main = "The error E(t,d,k) for the day Wednesday
      for different values of k",
   xlab = "The time intervals",
   ylab = "The Error E(t,d,k)", xaxt="n",
   type = "l", lwd=2, ylim = c(0,1500))
lines(Error_We_1, col="orange", lwd=2)
lines(Error_We_2, col="yellow", lwd=2)
lines(Error_We_4, col="green", lwd=2)
lines(Error_We_8, col="blue", lwd=2)
lines(Error_We_16, col="brown", lwd=2)
axis(1,at = seq(1,48,1),label=Times)
legend("topright",inset = c(-0.16,0),
   legend=c("k=0", "k=1", "k=2","k=4","k=8", "k=16"),
   col=c("red","orange","yellow","green","blue","brown"),
   cex=0.8,lty=1)
```