

May 2020

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

Frank Bitter
University of Wisconsin-Milwaukee

Follow this and additional works at: <https://dc.uwm.edu/etd>



Part of the [Mathematics Commons](#)

Recommended Citation

Bitter, Frank, "A Statistical Model for the Prediction of the Taxi Trip Time in the City of Chicago" (2020).
Theses and Dissertations. 2350.
<https://dc.uwm.edu/etd/2350>

This Thesis is brought to you for free and open access by UWM Digital Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UWM Digital Commons. For more information, please contact open-access@uwm.edu.

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 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 $a = a_0 < \dots < a_n = b$, $c = c_0 < \dots < c_m = d$, $e = e_0 < \dots < e_n = f$ and $g = g_0 < \dots < g_m = h$. 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 $\widetilde{n_1 \cdot m_1}$ 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 $\text{se} = \left(\frac{\text{Var}(R_{ij}(t, d))}{|R_{ij}(t, d)|} + \frac{\text{Var}(R_{ij}(t, d'))}{|R_{ij}(t, d')|} \right)^{1/2}$ 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 $\mu(i, j, t, d_2), \mu(i, j, t, d_3), \dots, \mu(i, j, t, d_n)$ 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)

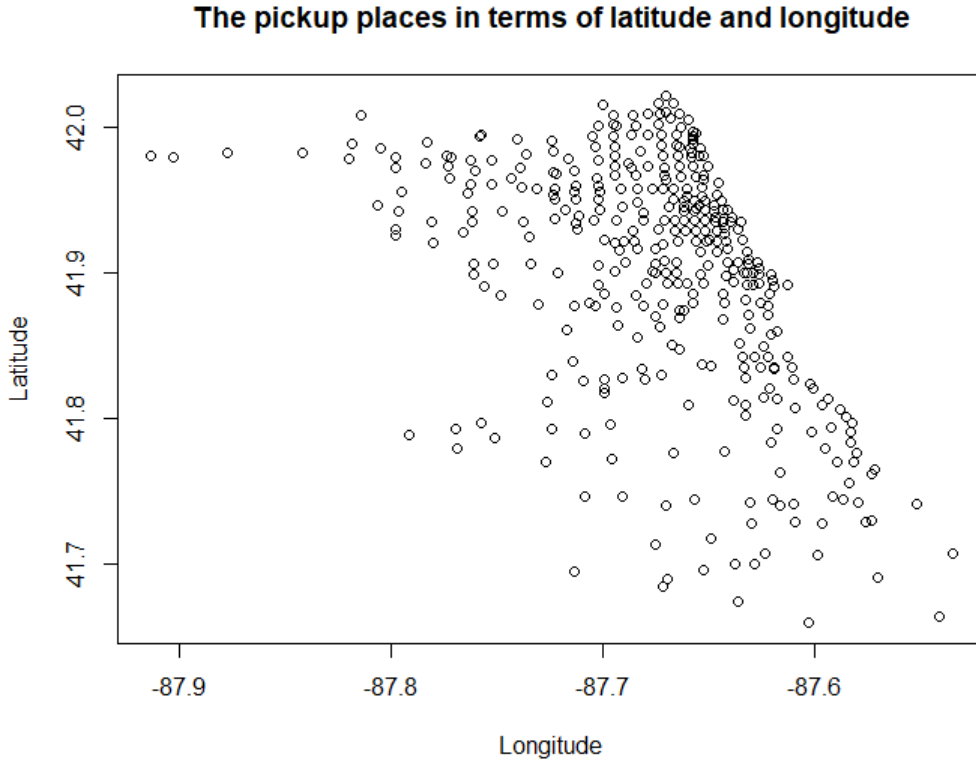


Figure 1: Pickup points on Mondays

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

The dropoff places in terms of latitude and longitude

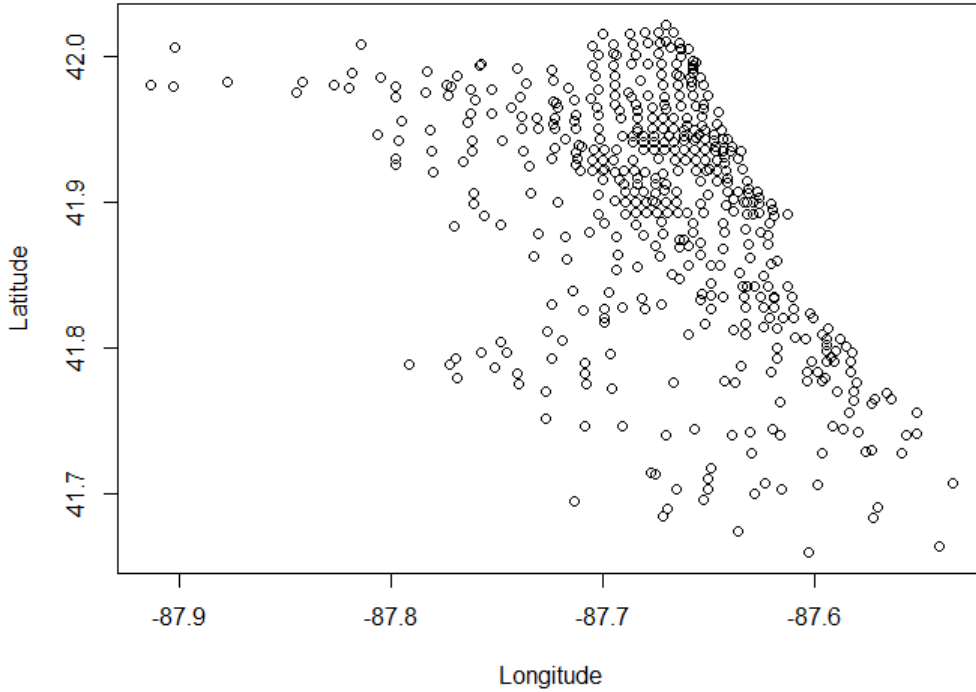
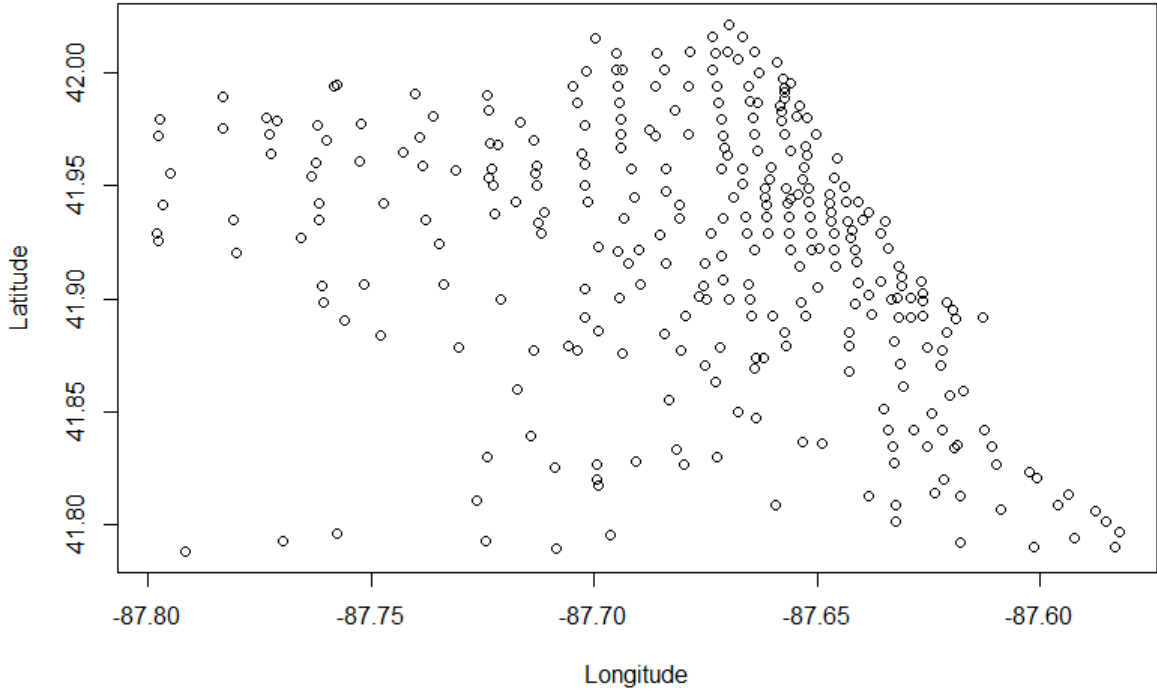


Figure 2: Dropoff points on Mondays

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

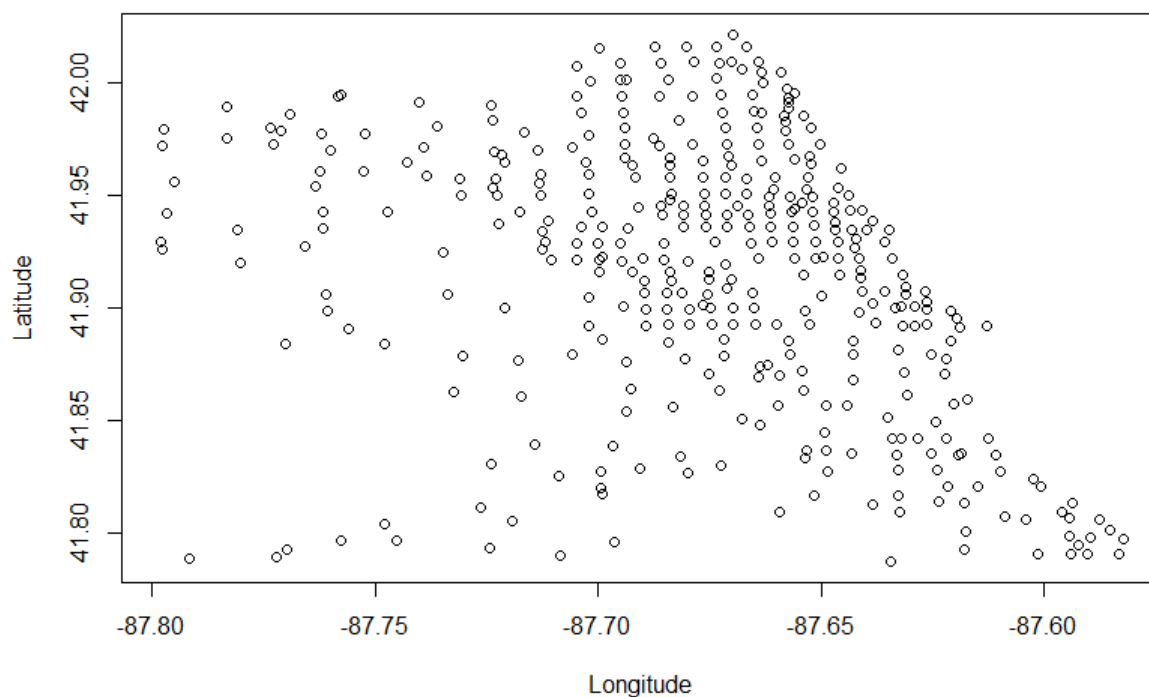
The pickup places in terms of latitude and longitude after the cut



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.

The dropoff places in terms of latitude and longitude after the cut



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 observations 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.

The Error depending on the time interval t for the several days

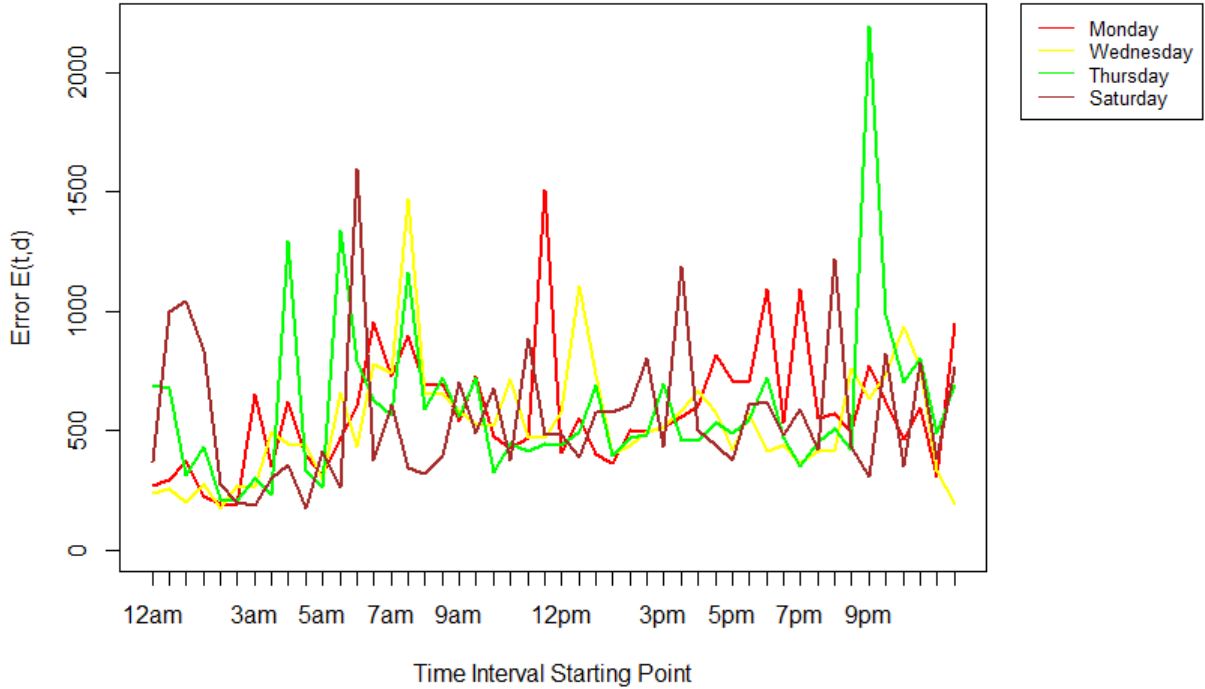


Figure 3: The error $E(t, d)$ over time t for Monday, Wednesday, Thursday and Saturday.

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

The Error depending on the time interval t for the several days

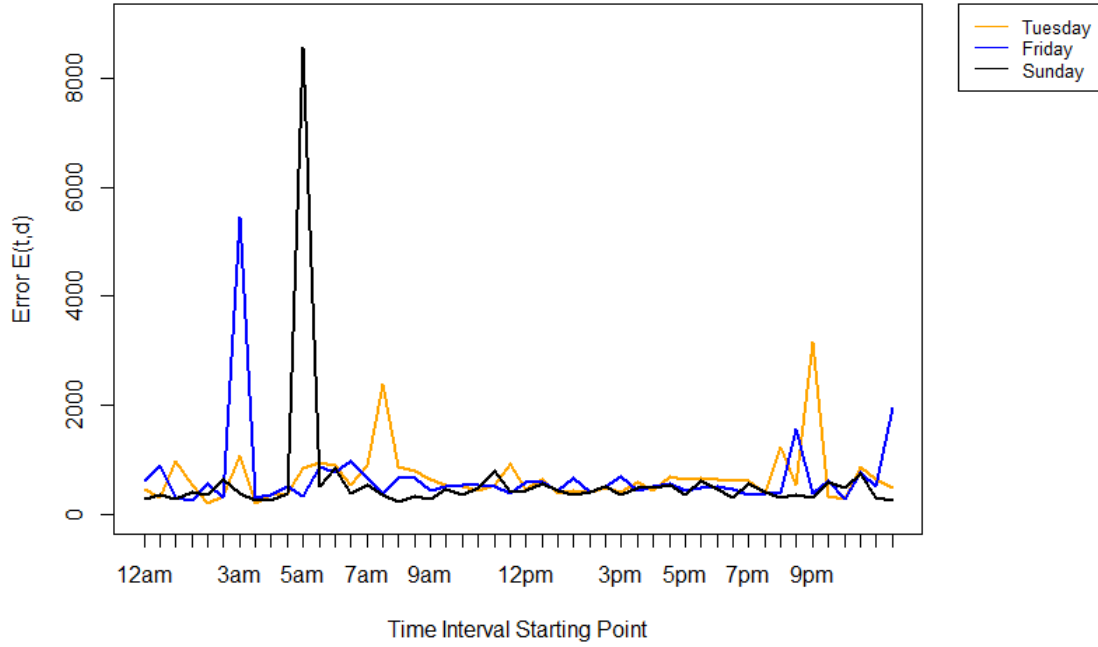


Figure 4: Error $E(t, d)$ over time t for Tuesday, Friday and Sunday

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 $E(d)$

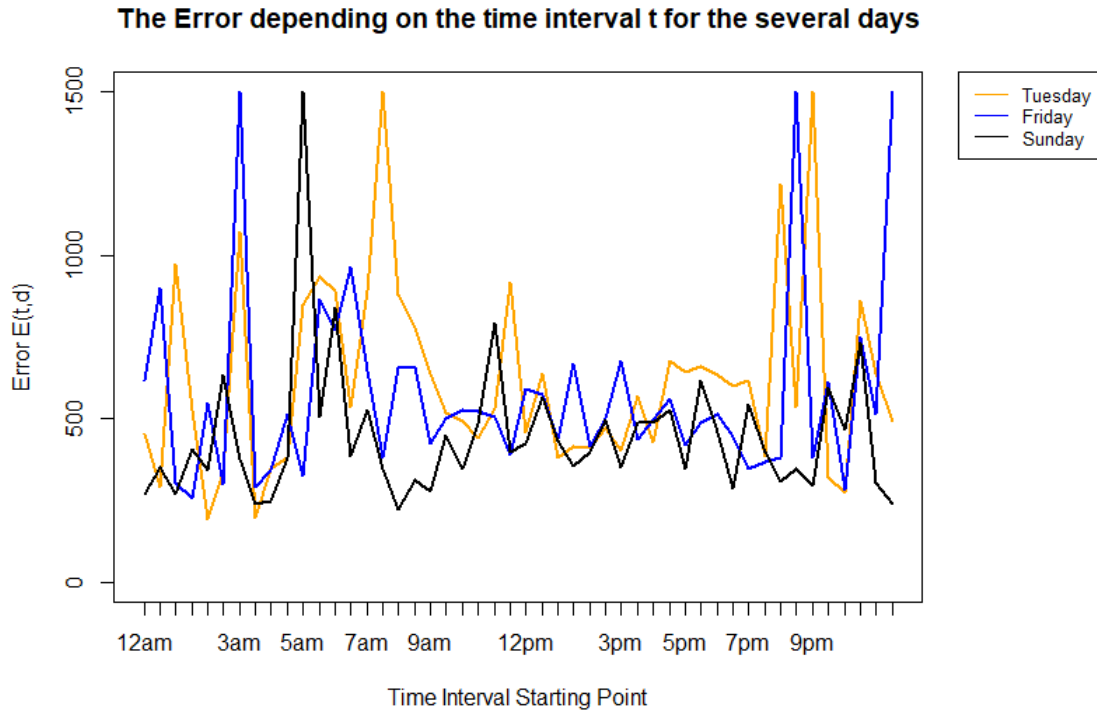


Figure 5: Error $E(t, d)$ over time t for Tuesday, Friday and Sunday, with values above 1500 eliminated.

influenced by outliers. Outliers can occur 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 extremely 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

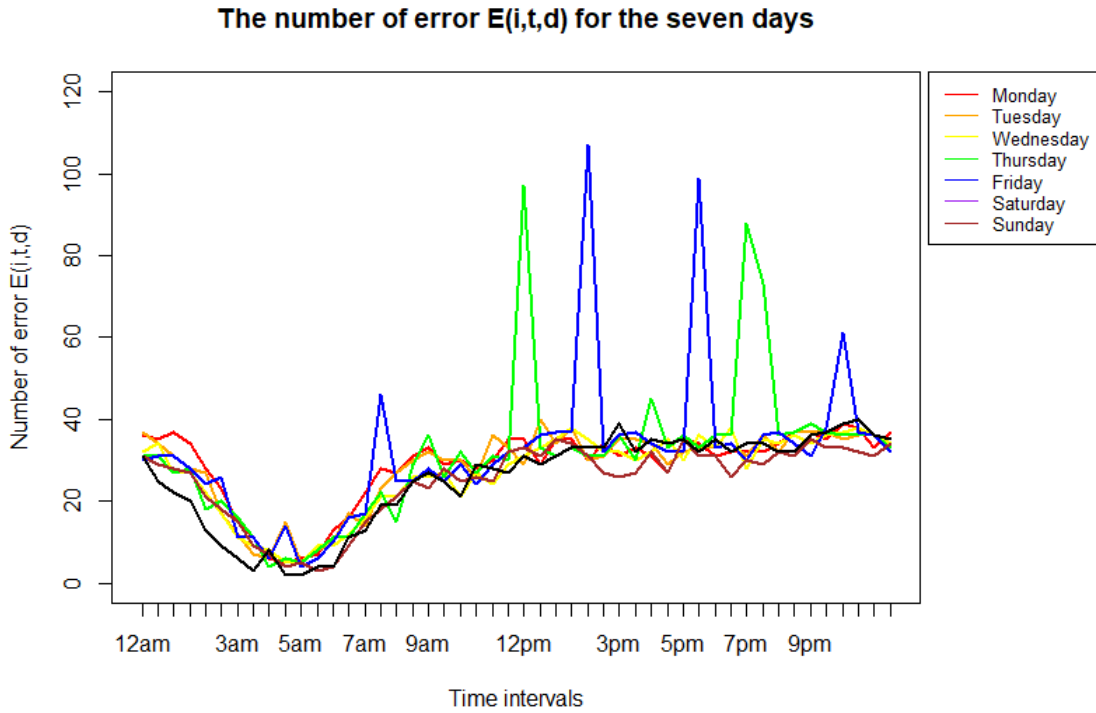


Figure 6: Number of existing $E(i, t, d)$ s.

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

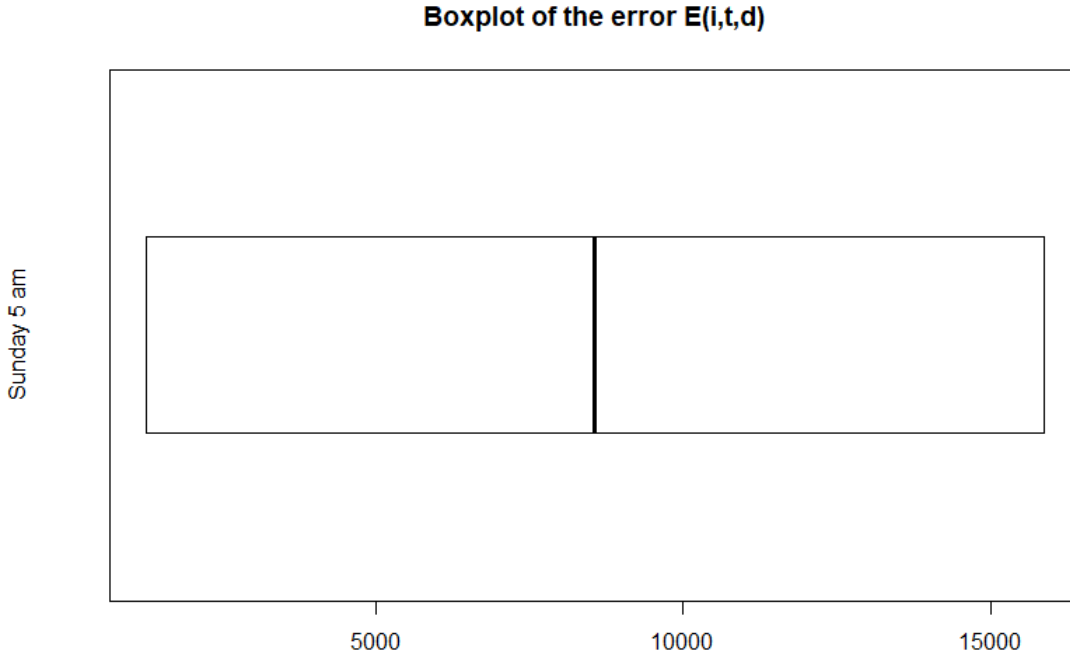


Figure 7: Boxplot of errors $E(i, t, d)$ on Sunday at time interval [5 am, 5.30 am)

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 outside 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.

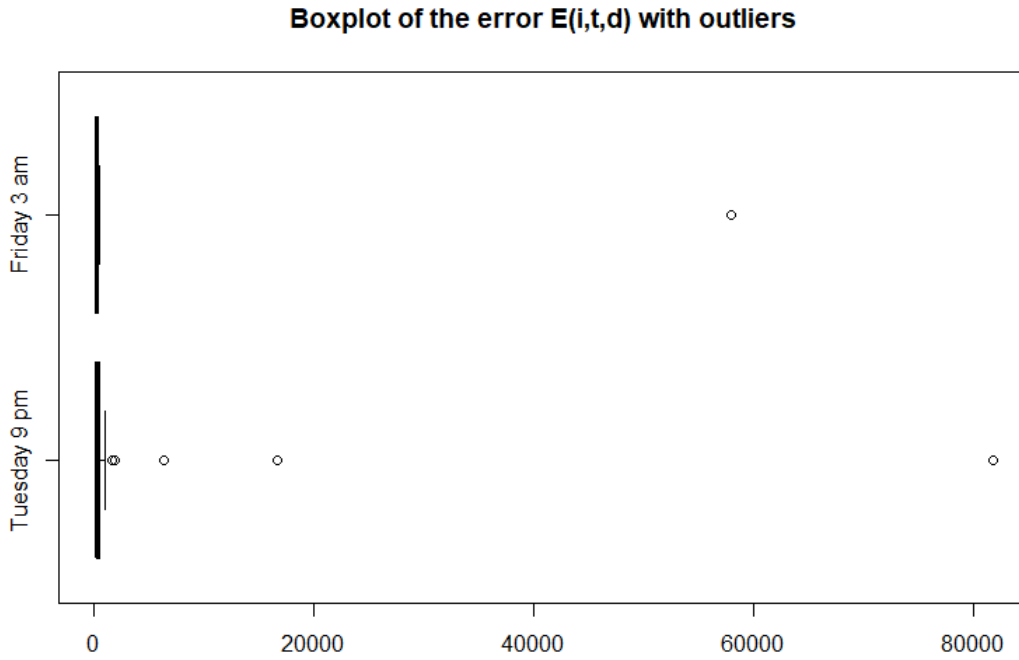


Figure 8: Boxplot of errors $E(i, t, d)$ with outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am)

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

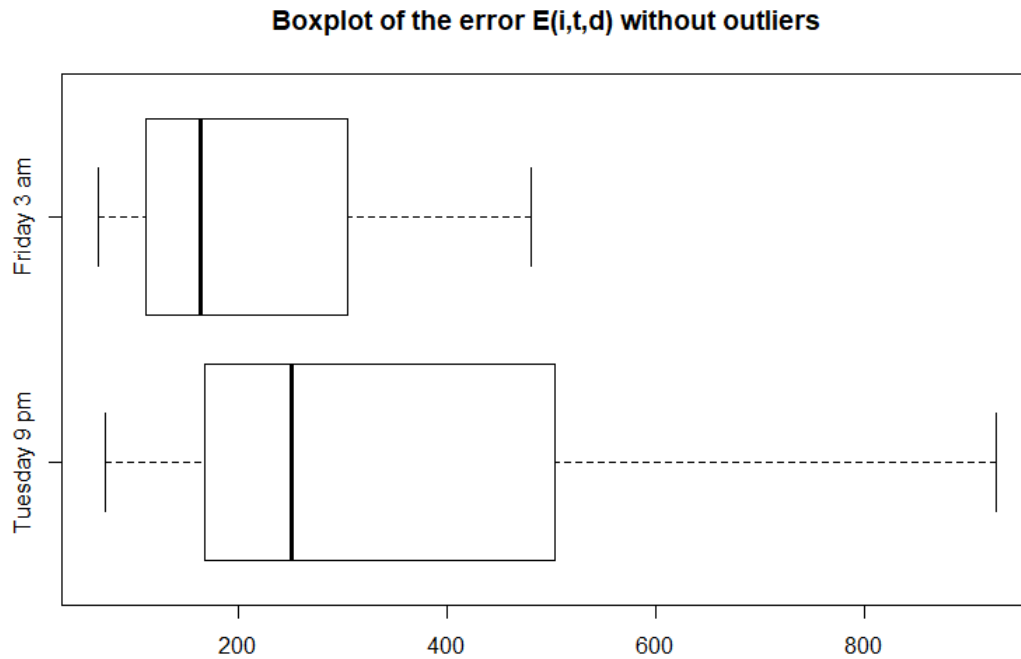


Figure 9: Boxplot of errors $E(i, t, d)$ without outliers for Tuesday at the time interval [9 pm, 9.30 pm) and for Friday at the time interval [3 am, 3.30 am)

$E(i, t, d)$ 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.

Boxplot of the error $E(i,t,d)$ for selected days and time intervals

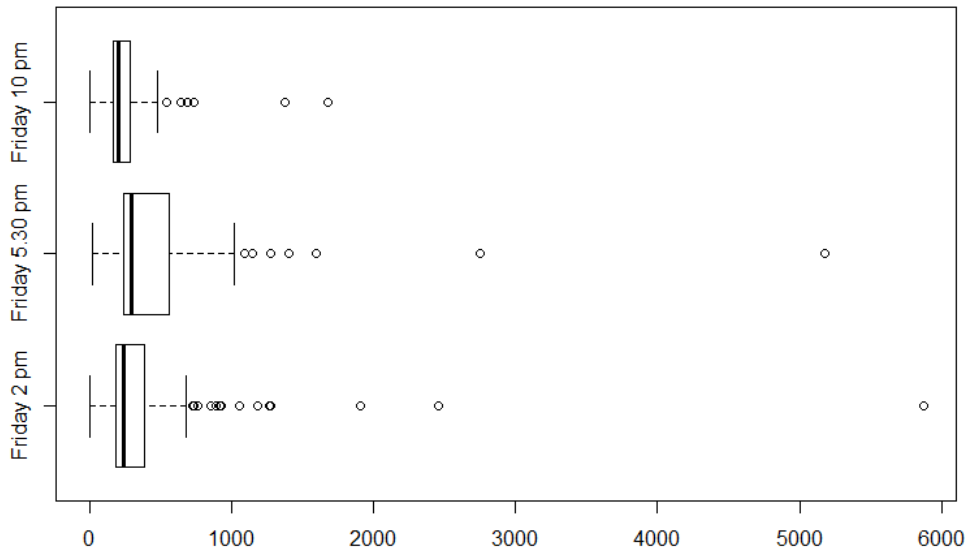
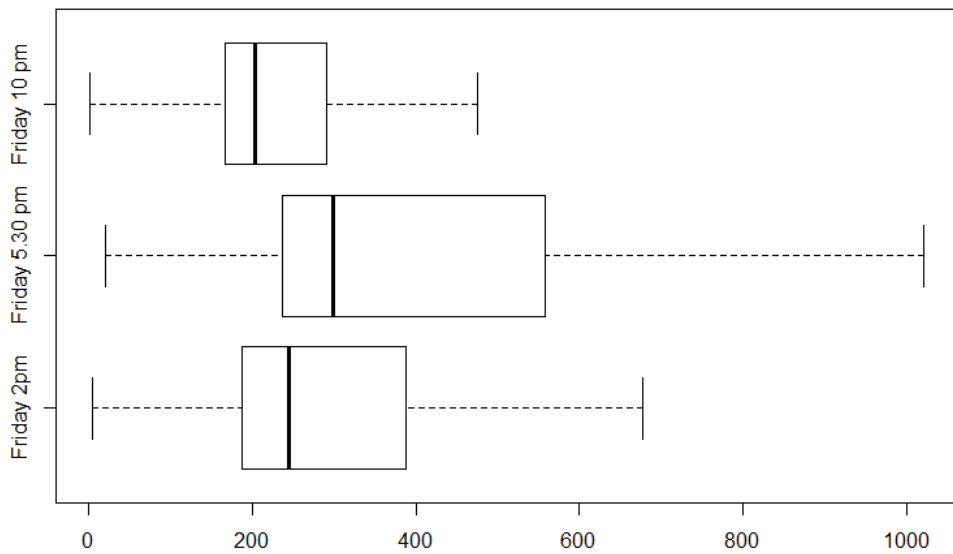


Figure 10: Boxplots of $E(i, t, d)$ s for Friday at time intervals [2 pm, 2.30 pm), [5.30 pm, 6 pm) and [10 pm, 10.30 pm), with outliers.

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.

Boxplot of the error $E(i,t,d)$ for selected days and time intervals without outliers



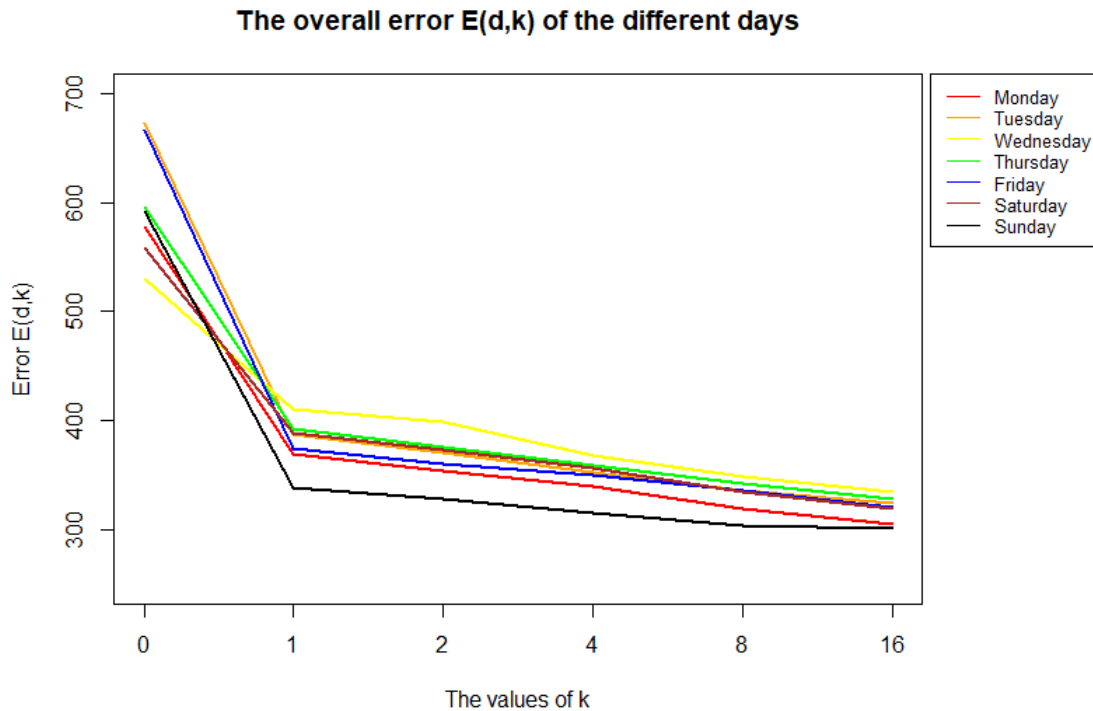


Figure 12: Errors $E(d, k)$ for different days d and pooling thresholds k .

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 k s. 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

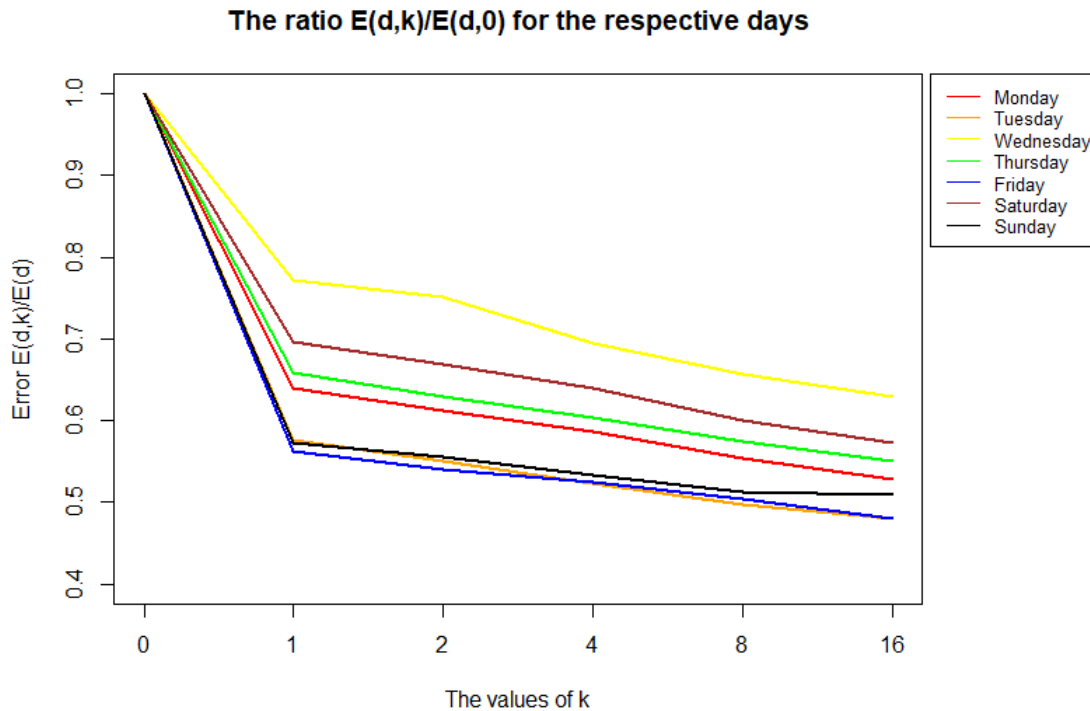


Figure 13: Ratios $E(d,k)/E(d)$ for several days d and pooling thresholds k .

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 $\frac{\text{"Range of the error"}}{\text{"mean of the error"}}$ 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.

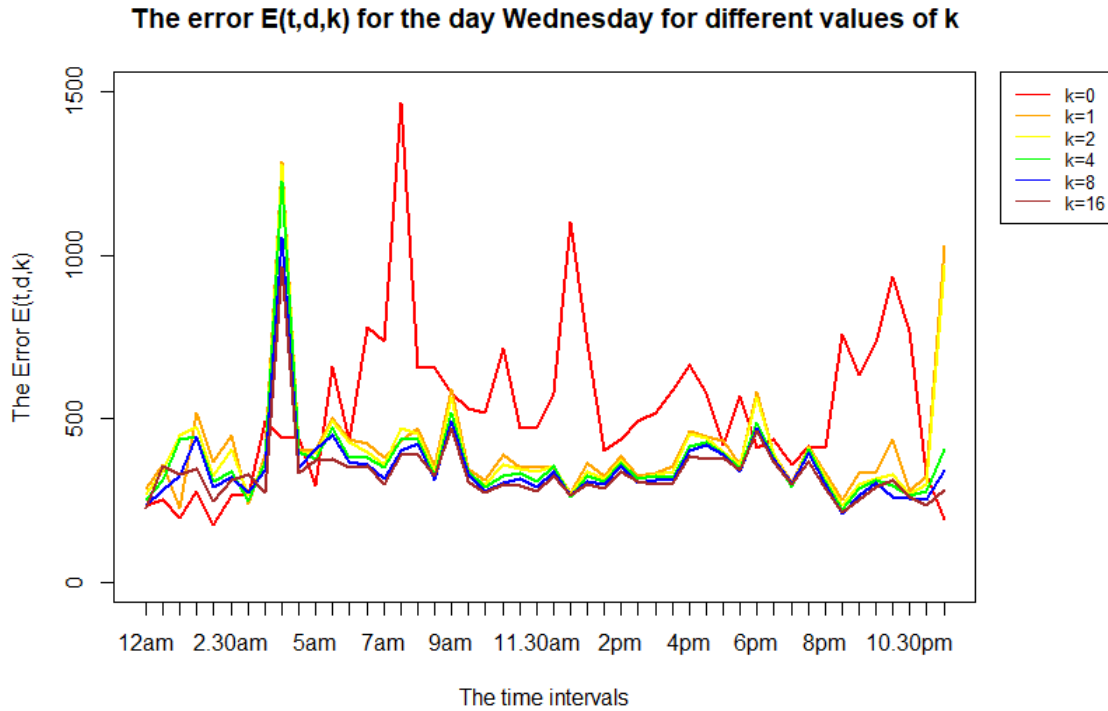


Figure 14: Errors $E(t, d, k)$ for Wednesday for different values of k

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

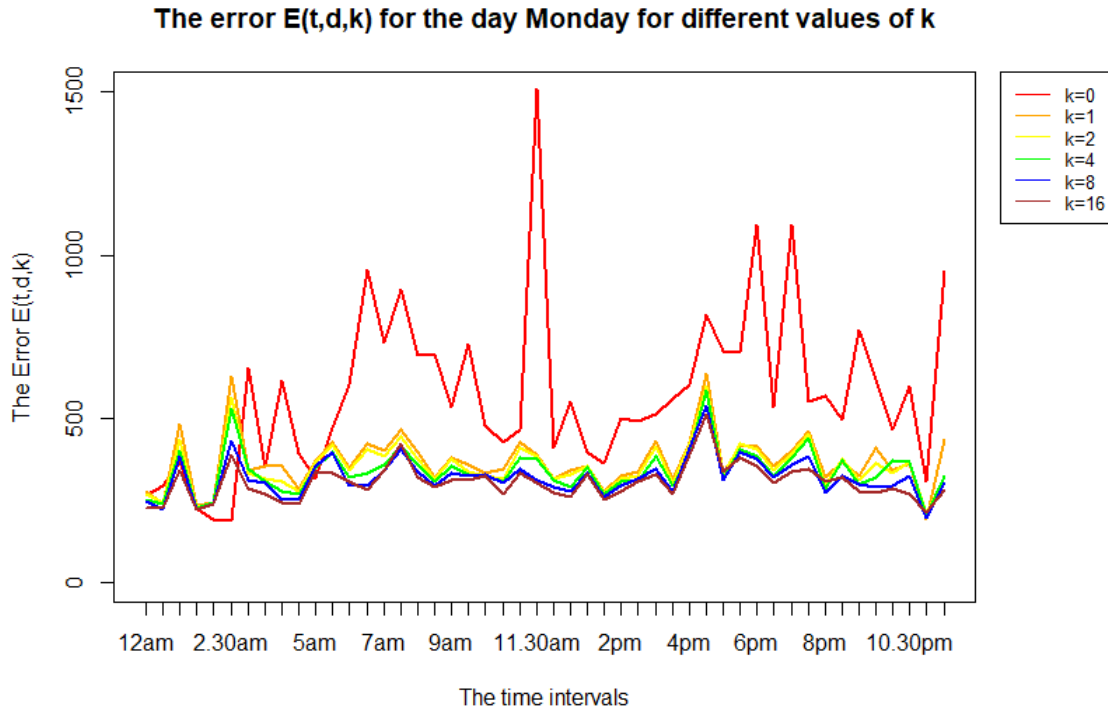


Figure 15: Errors $E(t, d, k)$ for Monday for different pooling thresholds k

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 $[2.30 \text{ am}, 3 \text{ am})$ 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 k s 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 estimators 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.

```
1 #Reading the data for 2019
2
3 Daten1=read.csv("Taxi_Trips-2018-Jan-1.csv")
4 Daten1=Daten1[,c(1,3,5,18,19,21,22)]
5 Daten2=read.csv("Taxi_Trips-2018-Jan-2.csv")
6 Daten2=Daten2[,c(1,3,5,18,19,21,22)]
7 Jan=rbind(Daten1, Daten2)
8 Daten1=NULL #Those Data will not be used any more, this is
9             #for improving the speed and stability of the program
10 Daten2=NULL
11
12 #Do the same process for the other 11 months
13
14 #Find the maxima and minima of the Latitude and Longitude
15     #of the pickup and dropoff points
16 Lat_max=max(max(Jan$Pickup.Centroid.Latitude, na.rm = TRUE),
17             max(Jan$Dropoff.Centroid.Latitude, na.rm = TRUE),
18             max(Feb$Pickup.Centroid.Latitude, na.rm = TRUE),
19             max(Feb$Dropoff.Centroid.Latitude, na.rm = TRUE),
20             max(Mar$Pickup.Centroid.Latitude, na.rm = TRUE),
21             max(Mar$Dropoff.Centroid.Latitude, na.rm = TRUE),
22             max(Apr$Pickup.Centroid.Latitude, na.rm = TRUE),
23             max(Apr$Dropoff.Centroid.Latitude, na.rm = TRUE),
24             max(May$Pickup.Centroid.Latitude, na.rm = TRUE),
25             max(May$Dropoff.Centroid.Latitude, na.rm = TRUE),
26             max(Jun$Pickup.Centroid.Latitude, na.rm = TRUE),
27             max(Jun$Dropoff.Centroid.Latitude, na.rm = TRUE),
28             max(Jul$Pickup.Centroid.Latitude, na.rm = TRUE),
29             max(Jul$Dropoff.Centroid.Latitude, na.rm = TRUE),
30             max(Aug$Pickup.Centroid.Latitude, na.rm = TRUE),
31             max(Aug$Dropoff.Centroid.Latitude, na.rm = TRUE),
32             max(Sep$Pickup.Centroid.Latitude, na.rm = TRUE),
33             max(Sep$Dropoff.Centroid.Latitude, na.rm = TRUE),
34             max(Oct$Pickup.Centroid.Latitude, na.rm = TRUE),
35             max(Oct$Dropoff.Centroid.Latitude, na.rm = TRUE),
36             max(Nov$Pickup.Centroid.Latitude, na.rm = TRUE),
37             max(Nov$Dropoff.Centroid.Latitude, na.rm = TRUE),
38             max(Dec$Pickup.Centroid.Latitude, na.rm = TRUE),
```



```

39         max(Dec$Dropoff.Centroid.Latitude, na.rm = TRUE))
40
41 #Do the same process with Lat_min, Long_max and Long_min
42
43 Information=c(Lat_min, Lat_max, Long_min, Long_max)
44 write.csv(Information, "Values-for-estimation.csv")
45
46 #Split the data into the days via functions (Monday, Tuesday,
47         # Wednesday, Thursday, Friday, Saturday, Sunday)
48 Day_Split_Mo=function(Daten) {
49     Test=filter(Daten, Daten$Pickup.Centroid.Latitude != ""
50                 & Daten$Dropoff.Centroid.Latitude != "")
51     Monday=filter(Test,
52                   str_detect(Trip.Start.Timestamp, '01/01/2018') |
53                   str_detect(Trip.Start.Timestamp, '01/08/2018') |
54                   str_detect(Trip.Start.Timestamp, '01/15/2018') |
55                   str_detect(Trip.Start.Timestamp, '01/22/2018') |
56                   str_detect(Trip.Start.Timestamp, '01/29/2018') |
57                   str_detect(Trip.Start.Timestamp, '02/05/2018') |
58                   str_detect(Trip.Start.Timestamp, '02/12/2018') |
59                   str_detect(Trip.Start.Timestamp, '02/19/2018') |
60                   str_detect(Trip.Start.Timestamp, '02/26/2018') |
61                   str_detect(Trip.Start.Timestamp, '03/05/2018') |
62                   str_detect(Trip.Start.Timestamp, '03/12/2018') |
63                   str_detect(Trip.Start.Timestamp, '03/19/2018') |
64                   str_detect(Trip.Start.Timestamp, '03/26/2018') |
65                   str_detect(Trip.Start.Timestamp, '04/02/2018') |
66                   str_detect(Trip.Start.Timestamp, '04/09/2018') |
67                   str_detect(Trip.Start.Timestamp, '04/16/2018') |
68                   str_detect(Trip.Start.Timestamp, '04/23/2018') |
69                   str_detect(Trip.Start.Timestamp, '04/30/2018') |
70                   str_detect(Trip.Start.Timestamp, '05/07/2018') |
71                   str_detect(Trip.Start.Timestamp, '05/14/2018') |
72                   str_detect(Trip.Start.Timestamp, '05/21/2018') |
73                   str_detect(Trip.Start.Timestamp, '05/28/2018') |
74                   str_detect(Trip.Start.Timestamp, '06/04/2018') |
75                   str_detect(Trip.Start.Timestamp, '06/11/2018') |
76                   str_detect(Trip.Start.Timestamp, '06/18/2018') |
77                   str_detect(Trip.Start.Timestamp, '06/25/2018') |
78                   str_detect(Trip.Start.Timestamp, '07/02/2018') |
79                   str_detect(Trip.Start.Timestamp, '07/09/2018') |
80                   str_detect(Trip.Start.Timestamp, '07/16/2018') |
81                   str_detect(Trip.Start.Timestamp, '07/23/2018') |
82                   str_detect(Trip.Start.Timestamp, '07/30/2018') |
83                   str_detect(Trip.Start.Timestamp, '08/06/2018') |
84                   str_detect(Trip.Start.Timestamp, '08/13/2018') |

```

```

85     str_detect(Trip.Start.Timestamp, '08/20/2018') |
86     str_detect(Trip.Start.Timestamp, '08/27/2018') |
87     str_detect(Trip.Start.Timestamp, '09/03/2018') |
88     str_detect(Trip.Start.Timestamp, '09/10/2018') |
89     str_detect(Trip.Start.Timestamp, '09/17/2018') |
90     str_detect(Trip.Start.Timestamp, '09/24/2018') |
91     str_detect(Trip.Start.Timestamp, '10/01/2018') |
92     str_detect(Trip.Start.Timestamp, '10/08/2018') |
93     str_detect(Trip.Start.Timestamp, '10/15/2018') |
94     str_detect(Trip.Start.Timestamp, '10/22/2018') |
95     str_detect(Trip.Start.Timestamp, '10/29/2018') |
96     str_detect(Trip.Start.Timestamp, '11/05/2018') |
97     str_detect(Trip.Start.Timestamp, '11/12/2018') |
98     str_detect(Trip.Start.Timestamp, '11/19/2018') |
99     str_detect(Trip.Start.Timestamp, '11/26/2018') |
100    str_detect(Trip.Start.Timestamp, '12/03/2018') |
101    str_detect(Trip.Start.Timestamp, '12/10/2018') |
102    str_detect(Trip.Start.Timestamp, '12/17/2018') |
103    str_detect(Trip.Start.Timestamp, '12/24/2018') |
104    str_detect(Trip.Start.Timestamp, '12/31/2018'))
105    return(Monday)
106  }
107
108  #Do the same process for the other six days.
109    #The dates change.
110
111  #Generate datasets for the seven days depending on
112    #the respective month
113  Jan_Mo=Day_Split_Mo(Jan)
114  Jan_Tu=Day_Split_Tu(Jan)
115  Jan_We=Day_Split_We(Jan)
116  Jan_Th=Day_Split_Th(Jan)
117  Jan_Fr=Day_Split_Fr(Jan)
118  Jan_Sa=Day_Split_Sa(Jan)
119  Jan_Su=Day_Split_Su(Jan)
120  Jan=NULL
121
122  #Do the same process for the other months.
123
124  #Generate the datasets for one specific day
125    #Monday, Tuesday, etc. independent of the
126    #month.
127  Monday=rbind(Jan_Mo, Feb_Mo, Mar_Mo, Apr_Mo, May_Mo, Jun_Mo,
128    Jul_Mo, Aug_Mo, Sep_Mo, Oct_Mo, Nov_Mo, Dec_Mo)
129  Jan_Mo=NULL
130  Feb_Mo=NULL

```

```

131 Mar_Mo=NULL
132 Apr_Mo=NULL
133 May_Mo=NULL
134 Jun_Mo=NULL
135 Jul_Mo=NULL
136 Aug_Mo=NULL
137 Sep_Mo=NULL
138 Oct_Mo=NULL
139 Nov_Mo=NULL
140 Dec_Mo=NULL
141
142 #Do the same process for the other six days.
143
144 # Split the matrix into time intervals
145 #Monday
146 Monday=read.csv("Monday-2018.csv")
147 Monday_0000=filter(Monday,
148     str_detect(Trip.Start.Timestamp, '12:00:00 AM') |
149     str_detect(Trip.Start.Timestamp, '12:15:00 AM'))
150 Monday_0030=filter(Monday,
151     str_detect(Trip.Start.Timestamp, '12:30:00 AM') |
152     str_detect(Trip.Start.Timestamp, '12:45:00 AM'))
153 Monday_0100=filter(Monday,
154     str_detect(Trip.Start.Timestamp, '01:00:00 AM') |
155     str_detect(Trip.Start.Timestamp, '01:15:00 AM'))
156 Monday_0130=filter(Monday,
157     str_detect(Trip.Start.Timestamp, '01:30:00 AM') |
158     str_detect(Trip.Start.Timestamp, '01:45:00 AM'))
159 Monday_0200=filter(Monday,
160     str_detect(Trip.Start.Timestamp, '02:00:00 AM') |
161     str_detect(Trip.Start.Timestamp, '02:15:00 AM'))
162 Monday_0230=filter(Monday,
163     str_detect(Trip.Start.Timestamp, '02:30:00 AM') |
164     str_detect(Trip.Start.Timestamp, '02:45:00 AM'))
165 Monday_0300=filter(Monday,
166     str_detect(Trip.Start.Timestamp, '03:00:00 AM') |
167     str_detect(Trip.Start.Timestamp, '03:15:00 AM'))
168 Monday_0330=filter(Monday,
169     str_detect(Trip.Start.Timestamp, '03:30:00 AM') |
170     str_detect(Trip.Start.Timestamp, '03:45:00 AM'))
171 Monday_0400=filter(Monday,
172     str_detect(Trip.Start.Timestamp, '04:00:00 AM') |
173     str_detect(Trip.Start.Timestamp, '04:15:00 AM'))
174 Monday_0430=filter(Monday,
175     str_detect(Trip.Start.Timestamp, '04:30:00 AM') |
176     str_detect(Trip.Start.Timestamp, '04:45:00 AM'))

```

```

177 Monday_0500=filter(Monday,
178     str_detect(Trip.Start.Timestamp, '05:00:00 AM') |
179     str_detect(Trip.Start.Timestamp, '05:15:00 AM'))
180 Monday_0530=filter(Monday,
181     str_detect(Trip.Start.Timestamp, '05:30:00 AM') |
182     str_detect(Trip.Start.Timestamp, '05:45:00 AM'))
183 Monday_0600=filter(Monday,
184     str_detect(Trip.Start.Timestamp, '06:00:00 AM') |
185     str_detect(Trip.Start.Timestamp, '06:15:00 AM'))
186 Monday_0630=filter(Monday,
187     str_detect(Trip.Start.Timestamp, '06:30:00 AM') |
188     str_detect(Trip.Start.Timestamp, '06:45:00 AM'))
189 Monday_0700=filter(Monday,
190     str_detect(Trip.Start.Timestamp, '07:00:00 AM') |
191     str_detect(Trip.Start.Timestamp, '07:15:00 AM'))
192 Monday_0730=filter(Monday,
193     str_detect(Trip.Start.Timestamp, '07:30:00 AM') |
194     str_detect(Trip.Start.Timestamp, '07:45:00 AM'))
195 Monday_0800=filter(Monday,
196     str_detect(Trip.Start.Timestamp, '08:00:00 AM') |
197     str_detect(Trip.Start.Timestamp, '08:15:00 AM'))
198 Monday_0830=filter(Monday,
199     str_detect(Trip.Start.Timestamp, '08:30:00 AM') |
200     str_detect(Trip.Start.Timestamp, '08:45:00 AM'))
201 Monday_0900=filter(Monday,
202     str_detect(Trip.Start.Timestamp, '09:00:00 AM') |
203     str_detect(Trip.Start.Timestamp, '09:15:00 AM'))
204 Monday_0930=filter(Monday,
205     str_detect(Trip.Start.Timestamp, '09:30:00 AM') |
206     str_detect(Trip.Start.Timestamp, '09:45:00 AM'))
207 Monday_1000=filter(Monday,
208     str_detect(Trip.Start.Timestamp, '10:00:00 AM') |
209     str_detect(Trip.Start.Timestamp, '10:15:00 AM'))
210 Monday_1030=filter(Monday,
211     str_detect(Trip.Start.Timestamp, '10:30:00 AM') |
212     str_detect(Trip.Start.Timestamp, '10:45:00 AM'))
213 Monday_1100=filter(Monday,
214     str_detect(Trip.Start.Timestamp, '11:00:00 AM') |
215     str_detect(Trip.Start.Timestamp, '11:15:00 AM'))
216 Monday_1130=filter(Monday,
217     str_detect(Trip.Start.Timestamp, '11:30:00 AM') |
218     str_detect(Trip.Start.Timestamp, '11:45:00 AM'))
219 Monday_1200=filter(Monday,
220     str_detect(Trip.Start.Timestamp, '12:00:00 PM') |
221     str_detect(Trip.Start.Timestamp, '12:15:00 PM'))
222 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') |

```

```

269         str_detect(Trip.Start.Timestamp, '08:15:00 PM'))
270 Monday_2030=filter(Monday,
271         str_detect(Trip.Start.Timestamp, '08:30:00 PM') |
272         str_detect(Trip.Start.Timestamp, '08:45:00 PM'))
273 Monday_2100=filter(Monday,
274         str_detect(Trip.Start.Timestamp, '09:00:00 PM') |
275         str_detect(Trip.Start.Timestamp, '09:15:00 PM'))
276 Monday_2130=filter(Monday,
277         str_detect(Trip.Start.Timestamp, '09:30:00 PM') |
278         str_detect(Trip.Start.Timestamp, '09:45:00 PM'))
279 Monday_2200=filter(Monday,
280         str_detect(Trip.Start.Timestamp, '10:00:00 PM') |
281         str_detect(Trip.Start.Timestamp, '10:15:00 PM'))
282 Monday_2230=filter(Monday,
283         str_detect(Trip.Start.Timestamp, '10:30:00 PM') |
284         str_detect(Trip.Start.Timestamp, '10:45:00 PM'))
285 Monday_2300=filter(Monday,
286         str_detect(Trip.Start.Timestamp, '11:00:00 PM') |
287         str_detect(Trip.Start.Timestamp, '11:15:00 PM'))
288 Monday_2330=filter(Monday,
289         str_detect(Trip.Start.Timestamp, '11:30:00 PM') |
290         str_detect(Trip.Start.Timestamp, '11:45:00 PM'))
291 Monday=NULL
292
293 #Do the same process with the other six days.
294
295 #Write the data into csv data
296 #Monday
297 write.csv(Monday_0000, "Monday-0000-2018.csv")
298 write.csv(Monday_0030, "Monday-0030-2018.csv")
299 write.csv(Monday_0100, "Monday-0100-2018.csv")
300 write.csv(Monday_0130, "Monday-0130-2018.csv")
301 write.csv(Monday_0200, "Monday-0200-2018.csv")
302 write.csv(Monday_0230, "Monday-0230-2018.csv")
303 write.csv(Monday_0300, "Monday-0300-2018.csv")
304 write.csv(Monday_0330, "Monday-0330-2018.csv")
305 write.csv(Monday_0400, "Monday-0400-2018.csv")
306 write.csv(Monday_0430, "Monday-0430-2018.csv")
307 write.csv(Monday_0500, "Monday-0500-2018.csv")
308 write.csv(Monday_0530, "Monday-0530-2018.csv")
309 write.csv(Monday_0600, "Monday-0600-2018.csv")
310 write.csv(Monday_0630, "Monday-0630-2018.csv")
311 write.csv(Monday_0700, "Monday-0700-2018.csv")
312 write.csv(Monday_0730, "Monday-0730-2018.csv")
313 write.csv(Monday_0800, "Monday-0800-2018.csv")
314 write.csv(Monday_0830, "Monday-0830-2018.csv")

```

```

315 write.csv(Monday_0900, "Monday-0900-2018.csv")
316 write.csv(Monday_0930, "Monday-0930-2018.csv")
317 write.csv(Monday_1000, "Monday-1000-2018.csv")
318 write.csv(Monday_1030, "Monday-1030-2018.csv")
319 write.csv(Monday_1100, "Monday-1100-2018.csv")
320 write.csv(Monday_1130, "Monday-1130-2018.csv")
321 write.csv(Monday_1200, "Monday-1200-2018.csv")
322 write.csv(Monday_1230, "Monday-1230-2018.csv")
323 write.csv(Monday_1300, "Monday-1300-2018.csv")
324 write.csv(Monday_1330, "Monday-1330-2018.csv")
325 write.csv(Monday_1400, "Monday-1400-2018.csv")
326 write.csv(Monday_1430, "Monday-1430-2018.csv")
327 write.csv(Monday_1500, "Monday-1500-2018.csv")
328 write.csv(Monday_1530, "Monday-1530-2018.csv")
329 write.csv(Monday_1600, "Monday-1600-2018.csv")
330 write.csv(Monday_1630, "Monday-1630-2018.csv")
331 write.csv(Monday_1700, "Monday-1700-2018.csv")
332 write.csv(Monday_1730, "Monday-1730-2018.csv")
333 write.csv(Monday_1800, "Monday-1800-2018.csv")
334 write.csv(Monday_1830, "Monday-1830-2018.csv")
335 write.csv(Monday_1900, "Monday-1900-2018.csv")
336 write.csv(Monday_1930, "Monday-1930-2018.csv")
337 write.csv(Monday_2000, "Monday-2000-2018.csv")
338 write.csv(Monday_2030, "Monday-2030-2018.csv")
339 write.csv(Monday_2100, "Monday-2100-2018.csv")
340 write.csv(Monday_2130, "Monday-2130-2018.csv")
341 write.csv(Monday_2200, "Monday-2200-2018.csv")
342 write.csv(Monday_2230, "Monday-2230-2018.csv")
343 write.csv(Monday_2300, "Monday-2300-2018.csv")
344 write.csv(Monday_2330, "Monday-2330-2018.csv")
345
346 #Do the same with the other six days.
347 #This process is done for the year 2018.
348     #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 program code is skipped.

```

1 Find_Estimator_Var=function(file_matrix, name_mean, name_var, name_coun
2
3 library(dplyr)

```

```

4 library(expss)
5 Dataset=file_matrix
6 Dataset=Dataset[,5:9]
7 Dataset=filter(Dataset, Dataset$Trip.Seconds!="")
8
9 Step_Lat=0.003703
10 Step_Long=0.004206
11 Lat_min=41.786 #changed after the analysis in Chapter 3.1
12 Lat_max=Max_Min_Values[2]
13 Long_min=-87.80453 #changed after the analysis in Chapter 3.1
14 Long_max=Max_Min_Values[4]
15
16 n=ceiling((Lat_max-Lat_min)/Step_Lat)
17 m=ceiling((Long_max-Long_min)/Step_Long)
18
19 Estimator_Dataset=matrix(0, nrow = n*m, ncol = m*n)
20 Variance_Dataset=matrix(0, nrow = n*m, ncol = m*n)
21 count_Dataset=matrix(0, nrow = n*m, ncol = m*n)
22
23 #Computation of the estimator
24 for (i in 1:n) {
25   Matrix_Help = filter(
26     Dataset, Dataset$Pickup.Centroid.Latitude >=
27       (Lat_min + (i - 1) * Step_Lat)
28     & Dataset$Pickup.Centroid.Latitude <
29       (Lat_min + i * Step_Lat))
30   if (nrow(as.matrix(Matrix_Help)) == 0) {
31     next
32   }
33   for (j in 1:m) {
34     Matrix_Help2 = filter(Matrix_Help,
35       Matrix_Help$Pickup.Centroid.Longitude >=
36       (Long_min + (j - 1) * Step_Long)
37     & Matrix_Help$Pickup.Centroid.Longitude <
38       (Long_min + j * Step_Long)
39     )
40     if (nrow(as.matrix(Matrix_Help2)) == 0) {
41       next
42     }
43     for (k in 1:n) {
44       Matrix_Help3 = filter(Matrix_Help2,
45         Matrix_Help2$Dropoff.Centroid.Latitude >=
46         (Lat_min + (k - 1) * Step_Lat)
47       & Matrix_Help2$Dropoff.Centroid.Latitude <
48         (Lat_min + k * Step_Lat)
49       )

```



```

50     if (nrow(as.matrix(Matrix_Help3)) == 0) {
51         next
52     }
53     for (l in 1:m) {
54         Matrix_Help4 = filter(Matrix_Help3,
55             Matrix_Help3$Dropoff.Centroid.Longitude >=
56                 (Long_min + (l - 1) * Step_Long)
57             & Matrix_Help3$Dropoff.Centroid.Longitude <
58                 (Long_min + l * Step_Long)
59         )
60         if (nrow(as.matrix(Matrix_Help4)) == 0) {
61             next
62         }
63         Daten = as.vector(Matrix_Help4[, 1])
64         Daten2 = as.numeric(sub(",", "", Daten,
65             fixed = TRUE))
66         Estimator_Dataset[(i - 1) * m + j, (k - 1) * m + 1]
67             = mean(Daten2)
68         Variance_Dataset[(i - 1) * m + j, (k - 1) * m + 1]
69             = var(Daten2)
70         if(is.na(Variance_Dataset
71             [(i - 1) * m + j, (k - 1) * m + 1])==TRUE)
72             {Variance_Dataset
73                 [(i - 1) * m + j, (k - 1) * m + 1]=0}
74         count_Dataset[(i - 1) * m + j, (k - 1) * m + 1]
75             = length(Daten2)
76     }
77 }
78 }
79 }
80 write.csv(Estimator_Dataset, name_mean)
81 write.csv(Variance_Dataset, name_var)
82 write.csv(count_Dataset, name_count)
83 return(Variance_Dataset)
84 }
85
86 #read Data
87 Max_Min=read.csv("Values-for-estimation.csv")
88 Max_Min=Max_Min[,2]
89 Monday_0000=read.csv("Monday-0000-2018.csv")
90 Est_Mo_0000=Find_Estimator_Var(Monday_0000,
91     "Estimator-Monday-0000.csv",
92     "Variance-Monday-0000.csv",
93     "Number-Monday-0000.csv", Max_Min)
94 Monday_0000=NULL
95 Monday_0030=read.csv("Monday-0030-2018.csv")

```

```

96 Est_Mo_0000=Find_Estimator_Var(Monday_0030,
97     "Estimator-Monday-0030.csv",
98     "Variance-Monday-0030.csv",
99     "Number-Monday-0030.csv",Max_Min)
100 Monday_0030=NULL
101 Monday_0100=read.csv("Monday-0100-2018.csv")
102 Est_Mo_0000=Find_Estimator_Var(Monday_0100,
103     "Estimator-Monday-0100.csv",
104     "Variance-Monday-0100.csv",
105     "Number-Monday-0100.csv",Max_Min)
106 Monday_0100=NULL
107 Monday_0130=read.csv("Monday-0130-2018.csv")
108 Est_Mo_0000=Find_Estimator_Var(Monday_0130,
109     "Estimator-Monday-0130.csv",
110     "Variance-Monday-0130.csv",
111     "Number-Monday-0130.csv",Max_Min)
112 Monday_0130=NULL
113 Monday_0200=read.csv("Monday-0200-2018.csv")
114 Est_Mo_0000=Find_Estimator_Var(Monday_0200,
115     "Estimator-Monday-0200.csv",
116     "Variance-Monday-0200.csv",
117     "Number-Monday-0200.csv",Max_Min)
118 Monday_0200=NULL
119 Monday_0230=read.csv("Monday-0230-2018.csv")
120 Est_Mo_0000=Find_Estimator_Var(Monday_0230,
121     "Estimator-Monday-0230.csv",
122     "Variance-Monday-0230.csv",
123     "Number-Monday-0230.csv",Max_Min)
124 Monday_0230=NULL
125 Monday_0300=read.csv("Monday-0300-2018.csv")
126 Est_Mo_0000=Find_Estimator_Var(Monday_0300,
127     "Estimator-Monday-0300.csv",
128     "Variance-Monday-0300.csv",
129     "Number-Monday-0300.csv",Max_Min)
130 Monday_0300=NULL
131 Monday_0330=read.csv("Monday-0330-2018.csv")
132 Est_Mo_0000=Find_Estimator_Var(Monday_0330,
133     "Estimator-Monday-0330.csv",
134     "Variance-Monday-0330.csv",
135     "Number-Monday-0330.csv",Max_Min)
136 Monday_0330=NULL
137 Monday_0400=read.csv("Monday-0400-2018.csv")
138 Est_Mo_0000=Find_Estimator_Var(Monday_0400,
139     "Estimator-Monday-0400.csv",
140     "Variance-Monday-0400.csv",
141     "Number-Monday-0400.csv",Max_Min)

```

```

142 Monday_0400=NULL
143 Monday_0430=read.csv("Monday-0430-2018.csv")
144 Est_Mo_0000=Find_Estimator_Var(Monday_0430,
145     "Estimator-Monday-0430.csv",
146     "Variance-Monday-0430.csv",
147     "Number-Monday-0430.csv",Max_Min)
148 Monday_0430=NULL
149 Monday_0500=read.csv("Monday-0500-2018.csv")
150 Est_Mo_0000=Find_Estimator_Var(Monday_0500,
151     "Estimator-Monday-0500.csv",
152     "Variance-Monday-0500.csv",
153     "Number-Monday-0500.csv",Max_Min)
154 Monday_0500=NULL
155 Monday_0530=read.csv("Monday-0530-2018.csv")
156 Est_Mo_0000=Find_Estimator_Var(Monday_0530,
157     "Estimator-Monday-0530.csv",
158     "Variance-Monday-0530.csv",
159     "Number-Monday-0530.csv",Max_Min)
160 Monday_0530=NULL
161 Monday_0600=read.csv("Monday-0600-2018.csv")
162 Est_Mo_0000=Find_Estimator_Var(Monday_0600,
163     "Estimator-Monday-0600.csv",
164     "Variance-Monday-0600.csv",
165     "Number-Monday-0600.csv",Max_Min)
166 Monday_0600=NULL
167 Monday_0630=read.csv("Monday-0630-2018.csv")
168 Est_Mo_0000=Find_Estimator_Var(Monday_0630,
169     "Estimator-Monday-0630.csv",
170     "Variance-Monday-0630.csv",
171     "Number-Monday-0630.csv",Max_Min)
172 Monday_0630=NULL
173 Monday_0700=read.csv("Monday-0700-2018.csv")
174 Est_Mo_0000=Find_Estimator_Var(Monday_0700,
175     "Estimator-Monday-0700.csv",
176     "Variance-Monday-0700.csv",
177     "Number-Monday-0700.csv",Max_Min)
178 Monday_0700=NULL
179 Monday_0730=read.csv("Monday-0730-2018.csv")
180 Est_Mo_0000=Find_Estimator_Var(Monday_0730,
181     "Estimator-Monday-0730.csv",
182     "Variance-Monday-0730.csv",
183     "Number-Monday-0730.csv",Max_Min)
184 Monday_0730=NULL
185 Monday_0800=read.csv("Monday-0800-2018.csv")
186 Est_Mo_0000=Find_Estimator_Var(Monday_0800,
187     "Estimator-Monday-0800.csv",

```

```

188         "Variance-Monday-0800.csv",
189         "Number-Monday-0800.csv",Max_Min)
190 Monday_0800=NULL
191 Monday_0830=read.csv("Monday-0830-2018.csv")
192 Est_Mo_0000=Find_Estimator_Var(Monday_0830,
193         "Estimator-Monday-0830.csv",
194         "Variance-Monday-0830.csv",
195         "Number-Monday-0830.csv",Max_Min)
196 Monday_0830=NULL
197 Monday_0900=read.csv("Monday-0900-2018.csv")
198 Est_Mo_0000=Find_Estimator_Var(Monday_0900,
199         "Estimator-Monday-0900.csv",
200         "Variance-Monday-0900.csv",
201         "Number-Monday-0900.csv",Max_Min)
202 Monday_0900=NULL
203 Monday_0930=read.csv("Monday-0930-2018.csv")
204 Est_Mo_0000=Find_Estimator_Var(Monday_0930,
205         "Estimator-Monday-0930.csv",
206         "Variance-Monday-0930.csv",
207         "Number-Monday-0930.csv",Max_Min)
208 Monday_0930=NULL
209 Monday_1000=read.csv("Monday-1000-2018.csv")
210 Est_Mo_0000=Find_Estimator_Var(Monday_1000,
211         "Estimator-Monday-1000.csv",
212         "Variance-Monday-1000.csv",
213         "Number-Monday-1000.csv",Max_Min)
214 Monday_1000=NULL
215 Monday_1030=read.csv("Monday-1030-2018.csv")
216 Est_Mo_0000=Find_Estimator_Var(Monday_1030,
217         "Estimator-Monday-1030.csv",
218         "Variance-Monday-1030.csv",
219         "Number-Monday-1030.csv",Max_Min)
220 Monday_1030=NULL
221 Monday_1100=read.csv("Monday-1100-2018.csv")
222 Est_Mo_0000=Find_Estimator_Var(Monday_1100,
223         "Estimator-Monday-1100.csv",
224         "Variance-Monday-1100.csv",
225         "Number-Monday-1100.csv",Max_Min)
226 Monday_1100=NULL
227 Monday_1130=read.csv("Monday-1130-2018.csv")
228 Est_Mo_0000=Find_Estimator_Var(Monday_1130,
229         "Estimator-Monday-1130.csv",
230         "Variance-Monday-1130.csv",
231         "Number-Monday-1130.csv",Max_Min)
232 Monday_1130=NULL
233 Monday_1200=read.csv("Monday-1200-2018.csv")

```

```

234 Est_Mo_0000=Find_Estimator_Var(Monday_1200,
235     "Estimator-Monday-1200.csv",
236     "Variance-Monday-1200.csv",
237     "Number-Monday-1200.csv", Max_Min)
238 Monday_1200=NULL
239 Monday_1230=read.csv("Monday-1230-2018.csv")
240 Est_Mo_0000=Find_Estimator_Var(Monday_1230,
241     "Estimator-Monday-1230.csv",
242     "Variance-Monday-1230.csv",
243     "Number-Monday-1230.csv", Max_Min)
244 Monday_1230=NULL
245 Monday_1300=read.csv("Monday-1300-2018.csv")
246 Est_Mo_0000=Find_Estimator_Var(Monday_1300,
247     "Estimator-Monday-1300.csv",
248     "Variance-Monday-1300.csv",
249     "Number-Monday-1300.csv", Max_Min)
250 Monday_1300=NULL
251 Monday_1330=read.csv("Monday-1330-2018.csv")
252 Est_Mo_0000=Find_Estimator_Var(Monday_1330,
253     "Estimator-Monday-1330.csv",
254     "Variance-Monday-1330.csv",
255     "Number-Monday-1330.csv",Max_Min)
256 Monday_1330=NULL
257 Monday_1400=read.csv("Monday-1400-2018.csv")
258 Est_Mo_0000=Find_Estimator_Var(Monday_1400,
259     "Estimator-Monday-1400.csv",
260     "Variance-Monday-1400.csv",
261     "Number-Monday-1400.csv",Max_Min)
262 Monday_1400=NULL
263 Monday_1430=read.csv("Monday-1430-2018.csv")
264 Est_Mo_0000=Find_Estimator_Var(Monday_1430,
265     "Estimator-Monday-1430.csv",
266     "Variance-Monday-1430.csv",
267     "Number-Monday-1430.csv",Max_Min)
268 Monday_1430=NULL
269 Monday_1500=read.csv("Monday-1500-2018.csv")
270 Est_Mo_0000=Find_Estimator_Var(Monday_1500,
271     "Estimator-Monday-1500.csv",
272     "Variance-Monday-1500.csv",
273     "Number-Monday-1500.csv",Max_Min)
274 Monday_1500=NULL
275 Monday_1530=read.csv("Monday-1530-2018.csv")
276 Est_Mo_0000=Find_Estimator_Var(Monday_1530,
277     "Estimator-Monday-1530.csv",
278     "Variance-Monday-1530.csv",
279     "Number-Monday-1530.csv",Max_Min)

```

```

280 Monday_1530=NULL
281 Monday_1600=read.csv("Monday-1600-2018.csv")
282 Est_Mo_0000=Find_Estimator_Var(Monday_1600,
283     "Estimator-Monday-1600.csv", "Variance-Monday-1600.csv", "Number-Monday-1600.csv")
284 Monday_1600=NULL
285 Monday_1630=read.csv("Monday-1630-2018.csv")
286 Est_Mo_0000=Find_Estimator_Var(Monday_1630,
287     "Estimator-Monday-1630.csv",
288     "Variance-Monday-1630.csv",
289     "Number-Monday-1630.csv",Max_Min)
290 Monday_1630=NULL
291 Monday_1700=read.csv("Monday-1700-2018.csv")
292 Est_Mo_0000=Find_Estimator_Var(Monday_1700,
293     "Estimator-Monday-1700.csv",
294     "Variance-Monday-1700.csv",
295     "Number-Monday-1700.csv",Max_Min)
296 Monday_1700=NULL
297 Monday_1730=read.csv("Monday-1730-2018.csv")
298 Est_Mo_0000=Find_Estimator_Var(Monday_1730,
299     "Estimator-Monday-1730.csv",
300     "Variance-Monday-1730.csv",
301     "Number-Monday-1730.csv",Max_Min)
302 Monday_1730=NULL
303 Monday_1800=read.csv("Monday-1800-2018.csv")
304 Est_Mo_0000=Find_Estimator_Var(Monday_1800,
305     "Estimator-Monday-1800.csv",
306     "Variance-Monday-1800.csv",
307     "Number-Monday-1800.csv",Max_Min)
308 Monday_1800=NULL
309 Monday_1830=read.csv("Monday-1830-2018.csv")
310 Est_Mo_0000=Find_Estimator_Var(Monday_1830,
311     "Estimator-Monday-1830.csv",
312     "Variance-Monday-1830.csv",
313     "Number-Monday-1830.csv",Max_Min)
314 Monday_1830=NULL
315 Monday_1900=read.csv("Monday-1900-2018.csv")
316 Est_Mo_0000=Find_Estimator_Var(Monday_1900,
317     "Estimator-Monday-1900.csv",
318     "Variance-Monday-1900.csv",
319     "Number-Monday-1900.csv",Max_Min)
320 Monday_1900=NULL
321 Monday_1930=read.csv("Monday-1930-2018.csv")
322 Est_Mo_0000=Find_Estimator_Var(Monday_1930,
323     "Estimator-Monday-1930.csv",
324     "Variance-Monday-1930.csv",
325     "Number-Monday-1930.csv",Max_Min)

```

```

326 Monday_1930=NULL
327 Monday_2000=read.csv("Monday-2000-2018.csv")
328 Est_Mo_0000=Find_Estimator_Var(Monday_2000,
329     "Estimator-Monday-2000.csv",
330     "Variance-Monday-2000.csv",
331     "Number-Monday-2000.csv",Max_Min)
332 Monday_2000=NULL
333 Monday_2030=read.csv("Monday-2030-2018.csv")
334 Est_Mo_0000=Find_Estimator_Var(Monday_2030,
335     "Estimator-Monday-2030.csv",
336     "Variance-Monday-2030.csv",
337     "Number-Monday-2030.csv",Max_Min)
338 Monday_2030=NULL
339 Monday_2100=read.csv("Monday-2100-2018.csv")
340 Est_Mo_0000=Find_Estimator_Var(Monday_2100,
341     "Estimator-Monday-2100.csv",
342     "Variance-Monday-2100.csv",
343     "Number-Monday-2100.csv",Max_Min)
344 Monday_2100=NULL
345 Monday_2130=read.csv("Monday-2130-2018.csv")
346 Est_Mo_0000=Find_Estimator_Var(Monday_2130,
347     "Estimator-Monday-2130.csv",
348     "Variance-Monday-2130.csv",
349     "Number-Monday-2130.csv",Max_Min)
350 Monday_2130=NULL
351 Monday_2200=read.csv("Monday-2200-2018.csv")
352 Est_Mo_0000=Find_Estimator_Var(Monday_2200,
353     "Estimator-Monday-2200.csv",
354     "Variance-Monday-2200.csv",
355     "Number-Monday-2200.csv",Max_Min)
356 Monday_2200=NULL
357 Monday_2230=read.csv("Monday-2230-2018.csv")
358 Est_Mo_0000=Find_Estimator_Var(Monday_2230,
359     "Estimator-Monday-2230.csv",
360     "Variance-Monday-2230.csv",
361     "Number-Monday-2230.csv",Max_Min)
362 Monday_2230=NULL
363 Monday_2300=read.csv("Monday-2300-2018.csv")
364 Est_Mo_0000=Find_Estimator_Var(Monday_2300,
365     "Estimator-Monday-2300.csv",
366     "Variance-Monday-2300.csv",
367     "Number-Monday-2300.csv",Max_Min)
368 Monday_2300=NULL
369 Monday_2330=read.csv("Monday-2330-2018.csv")
370 Est_Mo_0000=Find_Estimator_Var(Monday_2330,
371     "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(exps)
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

```



```

33 }
34 for (j in 1:m) {
35   Matrix_Help2 = filter(Matrix_Help,
36     Matrix_Help$Pickup.Centroid.Longitude >=
37     (Long_min + (j - 1) * Step_Long)
38     & Matrix_Help$Pickup.Centroid.Longitude <
39     (Long_min + j * Step_Long)
40   )
41   if (nrow(as.matrix(Matrix_Help2)) == 0) {
42     next
43   }
44   for (k in 1:n) {
45     Matrix_Help3 = filter(Matrix_Help2,
46       Matrix_Help2$Dropoff.Centroid.Latitude >=
47       (Lat_min + (k - 1) * Step_Lat)
48       & Matrix_Help2$Dropoff.Centroid.Latitude <
49       (Lat_min + k * Step_Lat)
50     )
51     if (nrow(as.matrix(Matrix_Help3)) == 0) {
52       next
53     }
54     for (l in 1:m) {
55       Matrix_Help4 = filter(Matrix_Help3,
56         Matrix_Help3$Dropoff.Centroid.Longitude >=
57         (Long_min + (l - 1) * Step_Long)
58         & Matrix_Help3$Dropoff.Centroid.Longitude <
59         (Long_min + l * Step_Long)
60       )
61       if (nrow(as.matrix(Matrix_Help4)) == 0) {
62         next
63       }
64       if(file_mean
65         [(i - 1) * m + j, (k - 1) * m + 1]==0){next}
66       Daten = as.vector(Matrix_Help4[, 1])
67       Daten2 = as.numeric(
68         sub(",", "", Daten, fixed = TRUE))
69       n=length(Daten2)
70       Daten2=abs(Daten2-rep(file_mean
71         [(i - 1) * m + j, (k - 1) * m + 1],n))
72       Error_ij[(i - 1) * m + j, (k - 1) * m + 1]
73         =sum(Daten2)/n
74     }
75   }
76 }
77 }
78 for (a in 1:b) {

```

```

79     Help=Error_ij[,a]
80     if(sum(Help)==0){next}
81     Error_i[a]=mean(Help[Help!=0])
82 }
83
84 write.csv(Error_ij, name_error1)
85 write.csv(Error_i, name_error2)
86 return(Error_i)
87 }
88
89 #read Data
90 Max_Min=read.csv("Values-for-estimation.csv")
91 Max_Min=Max_Min[,2]
92 Monday_0000=read.csv("Monday-0000-2019.csv")
93 Estimator_Monday_0000=read.csv("Estimator-Monday-0000.csv")
94 Est_Mo_0000=Find_Estimator_Var(Monday_0000,
95     Estimator_Monday_0000, "Error-ij-Monday-0000.csv",
96     "Error-i-Monday-0000.csv", Max_Min)
97 Monday_0000=NULL
98 Estimator_Monday_0000=NULL
99 Monday_0030=read.csv("Monday-0030-2019.csv")
100 Estimator_Monday_0030=read.csv("Estimator-Monday-0030.csv")
101 Est_Mo_0030=Find_Estimator_Var(Monday_0030,
102     Estimator_Monday_0030, "Error-ij-Monday-0030.csv",
103     "Error-i-Monday-0030.csv", Max_Min)
104 Monday_0030=NULL
105 Estimator_Monday_0030=NULL
106 Monday_0100=read.csv("Monday-0100-2019.csv")
107 Estimator_Monday_0100=read.csv("Estimator-Monday-0100.csv")
108 Est_Mo_0100=Find_Estimator_Var(Monday_0100,
109     Estimator_Monday_0100, "Error-ij-Monday-0100.csv",
110     "Error-i-Monday-0100.csv", Max_Min)
111 Monday_0100=NULL
112 Estimator_Monday_0100=NULL
113 Monday_0130=read.csv("Monday-0130-2019.csv")
114 Estimator_Monday_0130=read.csv("Estimator-Monday-0130.csv")
115 Est_Mo_0130=Find_Estimator_Var(Monday_0130,
116     Estimator_Monday_0130, "Error-ij-Monday-0130.csv",
117     "Error-i-Monday-0130.csv", Max_Min)
118 Monday_0130=NULL
119 Estimator_Monday_0130=NULL
120 Monday_0200=read.csv("Monday-0200-2019.csv")
121 Estimator_Monday_0200=read.csv("Estimator-Monday-0200.csv")
122 Est_Mo_0200=Find_Estimator_Var(Monday_0200,
123     Estimator_Monday_0200, "Error-ij-Monday-0200.csv",
124     "Error-i-Monday-0200.csv", Max_Min)

```

```

125 Monday_0200=NULL
126 Estimator_Monday_0200=NULL
127 Monday_0230=read.csv("Monday-0230-2019.csv")
128 Estimator_Monday_0230=read.csv("Estimator-Monday-0230.csv")
129 Est_Mo_0230=Find_Estimator_Var(Monday_0230,
130     Estimator_Monday_0230, "Error-ij-Monday-0230.csv",
131     "Error-i-Monday-0230.csv", Max_Min)
132 Monday_0230=NULL
133 Estimator_Monday_0230=NULL
134 Monday_0300=read.csv("Monday-0300-2019.csv")
135 Estimator_Monday_0300=read.csv("Estimator-Monday-0300.csv")
136 Est_Mo_0300=Find_Estimator_Var(Monday_0300,
137     Estimator_Monday_0300, "Error-ij-Monday-0300.csv",
138     "Error-i-Monday-0300.csv", Max_Min)
139 Monday_0300=NULL
140 Estimator_Monday_0300=NULL
141 Monday_0330=read.csv("Monday-0330-2019.csv")
142 Estimator_Monday_0330=read.csv("Estimator-Monday-0330.csv")
143 Est_Mo_0330=Find_Estimator_Var(Monday_0330,
144     Estimator_Monday_0330, "Error-ij-Monday-0330.csv",
145     "Error-i-Monday-0330.csv", Max_Min)
146 Monday_0330=NULL
147 Estimator_Monday_0330=NULL
148 Monday_0400=read.csv("Monday-0400-2019.csv")
149 Estimator_Monday_0400=read.csv("Estimator-Monday-0400.csv")
150 Est_Mo_0400=Find_Estimator_Var(Monday_0400,
151     Estimator_Monday_0400, "Error-ij-Monday-0400.csv",
152     "Error-i-Monday-0400.csv", Max_Min)
153 Monday_0400=NULL
154 Estimator_Monday_0400=NULL
155 Monday_0430=read.csv("Monday-0430-2019.csv")
156 Estimator_Monday_0430=read.csv("Estimator-Monday-0430.csv")
157 Est_Mo_0430=Find_Estimator_Var(Monday_0430,
158     Estimator_Monday_0430, "Error-ij-Monday-0430.csv",
159     "Error-i-Monday-0430.csv", Max_Min)
160 Monday_0430=NULL
161 Estimator_Monday_0430=NULL
162 Monday_0500=read.csv("Monday-0500-2019.csv")
163 Estimator_Monday_0500=read.csv("Estimator-Monday-0500.csv")
164 Est_Mo_0500=Find_Estimator_Var(Monday_0500,
165     Estimator_Monday_0500, "Error-ij-Monday-0500.csv",
166     "Error-i-Monday-0500.csv", Max_Min)
167 Monday_0500=NULL
168 Estimator_Monday_0500=NULL
169 Monday_0530=read.csv("Monday-0530-2019.csv")
170 Estimator_Monday_0530=read.csv("Estimator-Monday-0530.csv")

```

```

171 Est_Mo_0530=Find_Estimator_Var(Monday_0530,
172     Estimator_Monday_0530, "Error-ij-Monday-0530.csv",
173     "Error-i-Monday-0530.csv", Max_Min)
174 Monday_0530=NULL
175 Estimator_Monday_0530=NULL
176 Monday_0600=read.csv("Monday-0600-2019.csv")
177 Estimator_Monday_0600=read.csv("Estimator-Monday-0600.csv")
178 Est_Mo_0600=Find_Estimator_Var(Monday_0600,
179     Estimator_Monday_0600, "Error-ij-Monday-0600.csv",
180     "Error-i-Monday-0600.csv", Max_Min)
181 Monday_0600=NULL
182 Estimator_Monday_0600=NULL
183 Monday_0630=read.csv("Monday-0630-2019.csv")
184 Estimator_Monday_0630=read.csv("Estimator-Monday-0630.csv")
185 Est_Mo_0630=Find_Estimator_Var(Monday_0630,
186     Estimator_Monday_0630, "Error-ij-Monday-0630.csv",
187     "Error-i-Monday-0630.csv", Max_Min)
188 Monday_0630=NULL
189 Estimator_Monday_0630=NULL
190 Monday_0700=read.csv("Monday-0700-2019.csv")
191 Estimator_Monday_0700=read.csv("Estimator-Monday-0700.csv")
192 Est_Mo_0700=Find_Estimator_Var(Monday_0700,
193     Estimator_Monday_0700, "Error-ij-Monday-0700.csv",
194     "Error-i-Monday-0700.csv", Max_Min)
195 Monday_0700=NULL
196 Estimator_Monday_0700=NULL
197 Monday_0730=read.csv("Monday-0730-2019.csv")
198 Estimator_Monday_0730=read.csv("Estimator-Monday-0730.csv")
199 Est_Mo_0730=Find_Estimator_Var(Monday_0730,
200     Estimator_Monday_0730, "Error-ij-Monday-0730.csv",
201     "Error-i-Monday-0730.csv", Max_Min)
202 Monday_0730=NULL
203 Estimator_Monday_0730=NULL
204 Monday_0800=read.csv("Monday-0800-2019.csv")
205 Estimator_Monday_0800=read.csv("Estimator-Monday-0800.csv")
206 Est_Mo_0800=Find_Estimator_Var(Monday_0800,
207     Estimator_Monday_0800, "Error-ij-Monday-0800.csv",
208     "Error-i-Monday-0800.csv", Max_Min)
209 Monday_0800=NULL
210 Estimator_Monday_0800=NULL
211 Monday_0830=read.csv("Monday-0830-2019.csv")
212 Estimator_Monday_0830=read.csv("Estimator-Monday-0830.csv")
213 Est_Mo_0830=Find_Estimator_Var(Monday_0830,
214     Estimator_Monday_0830, "Error-ij-Monday-0830.csv",
215     "Error-i-Monday-0830.csv", Max_Min)
216 Monday_0830=NULL

```

```

217 Estimator_Monday_0830=NULL
218 Monday_0900=read.csv("Monday-0900-2019.csv")
219 Estimator_Monday_0900=read.csv("Estimator-Monday-0900.csv")
220 Est_Mo_0900=Find_Estimator_Var(Monday_0900,
221     Estimator_Monday_0900, "Error-ij-Monday-0900.csv",
222     "Error-i-Monday-0900.csv", Max_Min)
223 Monday_0900=NULL
224 Estimator_Monday_0900=NULL
225 Monday_0930=read.csv("Monday-0930-2019.csv")
226 Estimator_Monday_0930=read.csv("Estimator-Monday-0930.csv")
227 Est_Mo_0930=Find_Estimator_Var(Monday_0930,
228     Estimator_Monday_0930, "Error-ij-Monday-0930.csv",
229     "Error-i-Monday-0930.csv", Max_Min)
230 Monday_0930=NULL
231 Estimator_Monday_0930=NULL
232 Monday_1000=read.csv("Monday-1000-2019.csv")
233 Estimator_Monday_1000=read.csv("Estimator-Monday-1000.csv")
234 Est_Mo_1000=Find_Estimator_Var(Monday_1000,
235     Estimator_Monday_1000, "Error-ij-Monday-1000.csv",
236     "Error-i-Monday-1000.csv", Max_Min)
237 Monday_1000=NULL
238 Estimator_Monday_1000=NULL
239 Monday_1030=read.csv("Monday-1030-2019.csv")
240 Estimator_Monday_1030=read.csv("Estimator-Monday-1030.csv")
241 Est_Mo_1030=Find_Estimator_Var(Monday_1030,
242     Estimator_Monday_1030, "Error-ij-Monday-1030.csv",
243     "Error-i-Monday-1030.csv", Max_Min)
244 Monday_1030=NULL
245 Estimator_Monday_1030=NULL
246 Monday_1100=read.csv("Monday-1100-2019.csv")
247 Estimator_Monday_1100=read.csv("Estimator-Monday-1100.csv")
248 Est_Mo_1100=Find_Estimator_Var(Monday_1100,
249     Estimator_Monday_1100, "Error-ij-Monday-1100.csv",
250     "Error-i-Monday-1100.csv", Max_Min)
251 Monday_1100=NULL
252 Estimator_Monday_1100=NULL
253 Monday_1130=read.csv("Monday-1130-2019.csv")
254 Estimator_Monday_1130=read.csv("Estimator-Monday-1130.csv")
255 Est_Mo_1130=Find_Estimator_Var(Monday_1130,
256     Estimator_Monday_1130, "Error-ij-Monday-1130.csv",
257     "Error-i-Monday-1130.csv", Max_Min)
258 Monday_1130=NULL
259 Estimator_Monday_1130=NULL
260 Monday_1200=read.csv("Monday-1200-2019.csv")
261 Estimator_Monday_1200=read.csv("Estimator-Monday-1200.csv")
262 Est_Mo_1200=Find_Estimator_Var(Monday_1200,

```

```

263         Estimator_Monday_1200, "Error-ij-Monday-1200.csv",
264         "Error-i-Monday-1200.csv", Max_Min)
265 Monday_1200=NULL
266 Estimator_Monday_1200=NULL
267 Monday_1230=read.csv("Monday-1230-2019.csv")
268 Estimator_Monday_1230=read.csv("Estimator-Monday-1230.csv")
269 Est_Mo_1230=Find_Estimator_Var(Monday_1230,
270         Estimator_Monday_1230, "Error-ij-Monday-1230.csv",
271         "Error-i-Monday-1230.csv", Max_Min)
272 Monday_1230=NULL
273 Estimator_Monday_1230=NULL
274 Monday_1300=read.csv("Monday-1300-2019.csv")
275 Estimator_Monday_1300=read.csv("Estimator-Monday-1300.csv")
276 Est_Mo_1300=Find_Estimator_Var(Monday_1300,
277         Estimator_Monday_1300, "Error-ij-Monday-1300.csv",
278         "Error-i-Monday-1300.csv", Max_Min)
279 Monday_1300=NULL
280 Estimator_Monday_1300=NULL
281 Monday_1330=read.csv("Monday-1330-2019.csv")
282 Estimator_Monday_1330=read.csv("Estimator-Monday-1330.csv")
283 Est_Mo_1330=Find_Estimator_Var(Monday_1330,
284         Estimator_Monday_1330, "Error-ij-Monday-1330.csv",
285         "Error-i-Monday-1330.csv", Max_Min)
286 Monday_1330=NULL
287 Estimator_Monday_1330=NULL
288 Monday_1400=read.csv("Monday-1400-2019.csv")
289 Estimator_Monday_1400=read.csv("Estimator-Monday-1400.csv")
290 Est_Mo_1400=Find_Estimator_Var(Monday_1400,
291         Estimator_Monday_1400, "Error-ij-Monday-1400.csv",
292         "Error-i-Monday-1400.csv", Max_Min)
293 Monday_1400=NULL
294 Estimator_Monday_1400=NULL
295 Monday_1430=read.csv("Monday-1430-2019.csv")
296 Estimator_Monday_1430=read.csv("Estimator-Monday-1430.csv")
297 Est_Mo_1430=Find_Estimator_Var(Monday_1430,
298         Estimator_Monday_1430, "Error-ij-Monday-1430.csv",
299         "Error-i-Monday-1430.csv", Max_Min)
300 Monday_1430=NULL
301 Estimator_Monday_1430=NULL
302 Monday_1500=read.csv("Monday-1500-2019.csv")
303 Estimator_Monday_1500=read.csv("Estimator-Monday-1500.csv")
304 Est_Mo_1500=Find_Estimator_Var(Monday_1500,
305         Estimator_Monday_1500, "Error-ij-Monday-1500.csv",
306         "Error-i-Monday-1500.csv", Max_Min)
307 Monday_1500=NULL
308 Estimator_Monday_1500=NULL

```

```

309 Monday_1530=read.csv("Monday-1530-2019.csv")
310 Estimator_Monday_1530=read.csv("Estimator-Monday-1530.csv")
311 Est_Mo_1530=Find_Estimator_Var(Monday_1530,
312     Estimator_Monday_1530, "Error-ij-Monday-1530.csv",
313     "Error-i-Monday-1530.csv", Max_Min)
314 Monday_1530=NULL
315 Estimator_Monday_1530=NULL
316 Monday_1600=read.csv("Monday-1600-2019.csv")
317 Estimator_Monday_1600=read.csv("Estimator-Monday-1600.csv")
318 Est_Mo_1600=Find_Estimator_Var(Monday_1600,
319     Estimator_Monday_1600, "Error-ij-Monday-1600.csv",
320     "Error-i-Monday-1600.csv", Max_Min)
321 Monday_1600=NULL
322 Estimator_Monday_1600=NULL
323 Monday_1630=read.csv("Monday-1630-2019.csv")
324 Estimator_Monday_1630=read.csv("Estimator-Monday-1630.csv")
325 Est_Mo_1630=Find_Estimator_Var(Monday_1630,
326     Estimator_Monday_1630, "Error-ij-Monday-1630.csv",
327     "Error-i-Monday-1630.csv", Max_Min)
328 Monday_1630=NULL
329 Estimator_Monday_1630=NULL
330 Monday_1700=read.csv("Monday-1700-2019.csv")
331 Estimator_Monday_1700=read.csv("Estimator-Monday-1700.csv")
332 Est_Mo_1700=Find_Estimator_Var(Monday_1700,
333     Estimator_Monday_1700, "Error-ij-Monday-1700.csv",
334     "Error-i-Monday-1700.csv", Max_Min)
335 Monday_1700=NULL
336 Estimator_Monday_1700=NULL
337 Monday_1730=read.csv("Monday-1730-2019.csv")
338 Estimator_Monday_1730=read.csv("Estimator-Monday-1730.csv")
339 Est_Mo_1730=Find_Estimator_Var(Monday_1730,
340     Estimator_Monday_1730, "Error-ij-Monday-1730.csv",
341     "Error-i-Monday-1730.csv", Max_Min)
342 Monday_1730=NULL
343 Estimator_Monday_1730=NULL
344 Monday_1800=read.csv("Monday-1800-2019.csv")
345 Estimator_Monday_1800=read.csv("Estimator-Monday-1800.csv")
346 Est_Mo_1800=Find_Estimator_Var(Monday_1800,
347     Estimator_Monday_1800, "Error-ij-Monday-1800.csv",
348     "Error-i-Monday-1800.csv", Max_Min)
349 Monday_1800=NULL
350 Estimator_Monday_1800=NULL
351 Monday_1830=read.csv("Monday-1830-2019.csv")
352 Estimator_Monday_1830=read.csv("Estimator-Monday-1830.csv")
353 Est_Mo_1830=Find_Estimator_Var(Monday_1830,
354     Estimator_Monday_1830, "Error-ij-Monday-1830.csv",

```

```

355         "Error-i-Monday-1830.csv", Max_Min)
356 Monday_1830=NULL
357 Estimator_Monday_1830=NULL
358 Monday_1900=read.csv("Monday-1900-2019.csv")
359 Estimator_Monday_1900=read.csv("Estimator-Monday-1900.csv")
360 Est_Mo_1900=Find_Estimator_Var(Monday_1900,
361     Estimator_Monday_1900, "Error-ij-Monday-1900.csv",
362     "Error-i-Monday-1900.csv", Max_Min)
363 Monday_1900=NULL
364 Estimator_Monday_1900=NULL
365 Monday_1930=read.csv("Monday-1930-2019.csv")
366 Estimator_Monday_1930=read.csv("Estimator-Monday-1930.csv")
367 Est_Mo_1930=Find_Estimator_Var(Monday_1930,
368     Estimator_Monday_1930, "Error-ij-Monday-1930.csv",
369     "Error-i-Monday-1930.csv", Max_Min)
370 Monday_1930=NULL
371 Estimator_Monday_1930=NULL
372 Monday_2000=read.csv("Monday-2000-2019.csv")
373 Estimator_Monday_2000=read.csv("Estimator-Monday-2000.csv")
374 Est_Mo_2000=Find_Estimator_Var(Monday_2000,
375     Estimator_Monday_2000, "Error-ij-Monday-2000.csv",
376     "Error-i-Monday-2000.csv", Max_Min)
377 Monday_2000=NULL
378 Estimator_Monday_2000=NULL
379 Monday_2030=read.csv("Monday-2030-2019.csv")
380 Estimator_Monday_2030=read.csv("Estimator-Monday-2030.csv")
381 Est_Mo_2030=Find_Estimator_Var(Monday_2030,
382     Estimator_Monday_2030, "Error-ij-Monday-2030.csv",
383     "Error-i-Monday-2030.csv", Max_Min)
384 Monday_2030=NULL
385 Estimator_Monday_2030=NULL
386 Monday_2100=read.csv("Monday-2100-2019.csv")
387 Estimator_Monday_2100=read.csv("Estimator-Monday-2100.csv")
388 Est_Mo_2100=Find_Estimator_Var(Monday_2100,
389     Estimator_Monday_2100, "Error-ij-Monday-2100.csv",
390     "Error-i-Monday-2100.csv", Max_Min)
391 Monday_2100=NULL
392 Estimator_Monday_2100=NULL
393 Monday_2130=read.csv("Monday-2130-2019.csv")
394 Estimator_Monday_2130=read.csv("Estimator-Monday-2130.csv")
395 Est_Mo_2130=Find_Estimator_Var(Monday_2130,
396     Estimator_Monday_2130, "Error-ij-Monday-2130.csv",
397     "Error-i-Monday-2130.csv", Max_Min)
398 Monday_2130=NULL
399 Estimator_Monday_2130=NULL
400 Monday_2200=read.csv("Monday-2200-2019.csv")

```



```

401 Estimator_Monday_2200=read.csv("Estimator-Monday-2200.csv")
402 Est_Mo_2200=Find_Estimator_Var(Monday_2200,
403     Estimator_Monday_2200, "Error-ij-Monday-2200.csv",
404     "Error-i-Monday-2200.csv", Max_Min)
405 Monday_2200=NULL
406 Estimator_Monday_2200=NULL
407 Monday_2230=read.csv("Monday-2230-2019.csv")
408 Estimator_Monday_2230=read.csv("Estimator-Monday-2230.csv")
409 Est_Mo_2230=Find_Estimator_Var(Monday_2230,
410     Estimator_Monday_2230, "Error-ij-Monday-2230.csv",
411     "Error-i-Monday-2230.csv", Max_Min)
412 Monday_2230=NULL
413 Estimator_Monday_2230=NULL
414 Monday_2300=read.csv("Monday-2300-2019.csv")
415 Estimator_Monday_2300=read.csv("Estimator-Monday-2300.csv")
416 Est_Mo_2300=Find_Estimator_Var(Monday_2300,
417     Estimator_Monday_2300, "Error-ij-Monday-2300.csv",
418     "Error-i-Monday-2300.csv", Max_Min)
419 Monday_2300=NULL
420 Estimator_Monday_2300=NULL
421 Monday_2330=read.csv("Monday-2330-2019.csv")
422 Estimator_Monday_2330=read.csv("Estimator-Monday-2330.csv")
423 Est_Mo_2330=Find_Estimator_Var(Monday_2330,
424     Estimator_Monday_2330, "Error-ij-Monday-2330.csv",
425     "Error-i-Monday-2330.csv", Max_Min)
426 Monday_2330=NULL
427 Estimator_Monday_2330=NULL
428
429 #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.

```

1 library(dplyr)
2 library(exps)
3
4 Find_k=function(Day1_mean, Day2_mean, Day1_var, Day1_n,
5     Day2_var, Day2_n, matrix_name){
6     Mean_1=read.csv(Day1_mean)
7     Mean_1=Mean_1[, -1]
8     Mean_2=read.csv(Day2_mean)
9     Mean_2=Mean_2[, -1]
10    Var_1=read.csv(Day1_var)

```

```

11 Var_1=Var_1[, -1]
12 Var_2=read.csv(Day2_var)
13 Var_2=Var_2[, -1]
14 Num_1=read.csv(Day1_n)
15 Num_1=Num_1[, -1]
16 Num_2=read.csv(Day2_n)
17 Num_2=Num_2[, -1]
18 Numerator=abs(Mean_1-Mean_2)
19 Denominator_1=sqrt(Var_1/Num_1)
20 Denominator_1=as.matrix(Denominator_1)
21 Denominator_2=sqrt(Var_2/Num_2)
22 Denominator_2=as.matrix(Denominator_2)
23 Matrix_Help=Numerator/(Denominator_1+Denominator_2)
24 write.csv(Matrix_Help, file = matrix_name)
25 k=max(Matrix_Help, na.rm = TRUE)
26 return(k)
27 }
28
29 k=rep(0, 48)
30 k[1]=Find_k("Estimator-Monday-0000.csv",
31            "Estimator-Tuesday-0000.csv",
32            "Variance-Monday-0000.csv",
33            "Variance-Tuesday-0000.csv",
34            "Number-Monday-0000.csv",
35            "Number-Tuesday-0000.csv",
36            "k-Matrix-Mo-Tu-0000.csv")
37 k[2]=Find_k("Estimator-Monday-0030.csv",
38            "Estimator-Tuesday-0030.csv",
39            "Variance-Monday-0030.csv",
40            "Variance-Tuesday-0030.csv",
41            "Number-Monday-0030.csv",
42            "Number-Tuesday-0030.csv",
43            "k-Matrix-Mo-Tu-0030.csv")
44 k[3]=Find_k("Estimator-Monday-0100.csv",
45            "Estimator-Tuesday-0100.csv",
46            "Variance-Monday-0100.csv",
47            "Variance-Tuesday-0100.csv",
48            "Number-Monday-0100.csv",
49            "Number-Tuesday-0100.csv",
50            "k-Matrix-Mo-Tu-0100.csv")
51 k[4]=Find_k("Estimator-Monday-0130.csv",
52            "Estimator-Tuesday-0130.csv",
53            "Variance-Monday-0130.csv",
54            "Variance-Tuesday-0130.csv",
55            "Number-Monday-0130.csv",
56            "Number-Tuesday-0130.csv",

```

```

57         "k-Matrix-Mo-Tu-0130.csv")
58 k[5]=Find_k("Estimator-Monday-0200.csv",
59             "Estimator-Tuesday-0200.csv",
60             "Variance-Monday-0200.csv",
61             "Variance-Tuesday-0200.csv",
62             "Number-Monday-0200.csv",
63             "Number-Tuesday-0200.csv",
64             "k-Matrix-Mo-Tu-0200.csv")
65 k[6]=Find_k("Estimator-Monday-0230.csv",
66             "Estimator-Tuesday-0230.csv",
67             "Variance-Monday-0230.csv",
68             "Variance-Tuesday-0230.csv",
69             "Number-Monday-0230.csv",
70             "Number-Tuesday-0230.csv",
71             "k-Matrix-Mo-Tu-0230.csv")
72 k[7]=Find_k("Estimator-Monday-0300.csv",
73             "Estimator-Tuesday-0300.csv",
74             "Variance-Monday-0300.csv",
75             "Variance-Tuesday-0300.csv",
76             "Number-Monday-0300.csv",
77             "Number-Tuesday-0300.csv",
78             "k-Matrix-Mo-Tu-0300.csv")
79 k[8]=Find_k("Estimator-Monday-0330.csv",
80             "Estimator-Tuesday-0330.csv",
81             "Variance-Monday-0330.csv",
82             "Variance-Tuesday-0330.csv",
83             "Number-Monday-0330.csv",
84             "Number-Tuesday-0330.csv",
85             "k-Matrix-Mo-Tu-0330.csv")
86 k[9]=Find_k("Estimator-Monday-0400.csv",
87             "Estimator-Tuesday-0400.csv",
88             "Variance-Monday-0400.csv",
89             "Variance-Tuesday-0400.csv",
90             "Number-Monday-0400.csv",
91             "Number-Tuesday-0400.csv",
92             "k-Matrix-Mo-Tu-0400.csv")
93 k[10]=Find_k("Estimator-Monday-0430.csv",
94             "Estimator-Tuesday-0430.csv",
95             "Variance-Monday-0430.csv",
96             "Variance-Tuesday-0430.csv",
97             "Number-Monday-0430.csv",
98             "Number-Tuesday-0430.csv",
99             "k-Matrix-Mo-Tu-0430.csv")
100 k[11]=Find_k("Estimator-Monday-0500.csv",
101             "Estimator-Tuesday-0500.csv",
102             "Variance-Monday-0500.csv",

```

```

103         "Variance-Tuesday-0500.csv",
104         "Number-Monday-0500.csv",
105         "Number-Tuesday-0500.csv",
106         "k-Matrix-Mo-Tu-0500.csv")
107 k[12]=Find_k("Estimator-Monday-0530.csv",
108             "Estimator-Tuesday-0530.csv",
109             "Variance-Monday-0530.csv",
110             "Variance-Tuesday-0530.csv",
111             "Number-Monday-0530.csv",
112             "Number-Tuesday-0530.csv",
113             "k-Matrix-Mo-Tu-0530.csv")
114 k[13]=Find_k("Estimator-Monday-0600.csv",
115             "Estimator-Tuesday-0600.csv",
116             "Variance-Monday-0600.csv",
117             "Variance-Tuesday-0600.csv",
118             "Number-Monday-0600.csv",
119             "Number-Tuesday-0600.csv",
120             "k-Matrix-Mo-Tu-0600.csv")
121 k[14]=Find_k("Estimator-Monday-0630.csv",
122             "Estimator-Tuesday-0630.csv",
123             "Variance-Monday-0630.csv",
124             "Variance-Tuesday-0630.csv",
125             "Number-Monday-0630.csv",
126             "Number-Tuesday-0630.csv",
127             "k-Matrix-Mo-Tu-0630.csv")
128 k[15]=Find_k("Estimator-Monday-0700.csv",
129             "Estimator-Tuesday-0700.csv",
130             "Variance-Monday-0700.csv",
131             "Variance-Tuesday-0700.csv",
132             "Number-Monday-0700.csv",
133             "Number-Tuesday-0700.csv",
134             "k-Matrix-Mo-Tu-0700.csv")
135 k[16]=Find_k("Estimator-Monday-0730.csv",
136             "Estimator-Tuesday-0730.csv",
137             "Variance-Monday-0730.csv",
138             "Variance-Tuesday-0730.csv",
139             "Number-Monday-0730.csv",
140             "Number-Tuesday-0730.csv",
141             "k-Matrix-Mo-Tu-0730.csv")
142 k[17]=Find_k("Estimator-Monday-0800.csv",
143             "Estimator-Tuesday-0800.csv",
144             "Variance-Monday-0800.csv",
145             "Variance-Tuesday-0800.csv",
146             "Number-Monday-0800.csv",
147             "Number-Tuesday-0800.csv",
148             "k-Matrix-Mo-Tu-0800.csv")

```

```

149 k[18]=Find_k("Estimator-Monday-0830.csv",
150             "Estimator-Tuesday-0830.csv",
151             "Variance-Monday-0830.csv",
152             "Variance-Tuesday-0830.csv",
153             "Number-Monday-0830.csv",
154             "Number-Tuesday-0830.csv",
155             "k-Matrix-Mo-Tu-0830.csv")
156 k[19]=Find_k("Estimator-Monday-0900.csv",
157             "Estimator-Tuesday-0900.csv",
158             "Variance-Monday-0900.csv",
159             "Variance-Tuesday-0900.csv",
160             "Number-Monday-0900.csv",
161             "Number-Tuesday-0900.csv",
162             "k-Matrix-Mo-Tu-0900.csv")
163 k[20]=Find_k("Estimator-Monday-0930.csv",
164             "Estimator-Tuesday-0930.csv",
165             "Variance-Monday-0930.csv",
166             "Variance-Tuesday-0930.csv",
167             "Number-Monday-0930.csv",
168             "Number-Tuesday-0930.csv",
169             "k-Matrix-Mo-Tu-0930.csv")
170 k[21]=Find_k("Estimator-Monday-1000.csv",
171             "Estimator-Tuesday-1000.csv",
172             "Variance-Monday-1000.csv",
173             "Variance-Tuesday-1000.csv",
174             "Number-Monday-1000.csv",
175             "Number-Tuesday-1000.csv",
176             "k-Matrix-Mo-Tu-1000.csv")
177 k[22]=Find_k("Estimator-Monday-1030.csv",
178             "Estimator-Tuesday-1030.csv",
179             "Variance-Monday-1030.csv",
180             "Variance-Tuesday-1030.csv",
181             "Number-Monday-1030.csv",
182             "Number-Tuesday-1030.csv",
183             "k-Matrix-Mo-Tu-1030.csv")
184 k[23]=Find_k("Estimator-Monday-1100.csv",
185             "Estimator-Tuesday-1100.csv",
186             "Variance-Monday-1100.csv",
187             "Variance-Tuesday-1100.csv",
188             "Number-Monday-1100.csv",
189             "Number-Tuesday-1100.csv",
190             "k-Matrix-Mo-Tu-1100.csv")
191 k[24]=Find_k("Estimator-Monday-1130.csv",
192             "Estimator-Tuesday-1130.csv",
193             "Variance-Monday-1130.csv",
194             "Variance-Tuesday-1130.csv",

```

```

195         "Number -Monday -1130 . csv " ,
196         "Number -Tuesday -1130 . csv " ,
197         "k -Matrix -Mo -Tu -1130 . csv " )
198 k [25] = Find _k ( "Estimator -Monday -1200 . csv " ,
199         "Estimator -Tuesday -1200 . csv " ,
200         "Variance -Monday -1200 . csv " ,
201         "Variance -Tuesday -1200 . csv " ,
202         "Number -Monday -1200 . csv " ,
203         "Number -Tuesday -1200 . csv " ,
204         "k -Matrix -Mo -Tu -1200 . csv " )
205 k [26] = Find _k ( "Estimator -Monday -1230 . csv " ,
206         "Estimator -Tuesday -1230 . csv " ,
207         "Variance -Monday -1230 . csv " ,
208         "Variance -Tuesday -1230 . csv " ,
209         "Number -Monday -1230 . csv " ,
210         "Number -Tuesday -1230 . csv " ,
211         "k -Matrix -Mo -Tu -1230 . csv " )
212 k [27] = Find _k ( "Estimator -Monday -1300 . csv " ,
213         "Estimator -Tuesday -1300 . csv " ,
214         "Variance -Monday -1300 . csv " ,
215         "Variance -Tuesday -1300 . csv " ,
216         "Number -Monday -1300 . csv " ,
217         "Number -Tuesday -1300 . csv " ,
218         "k -Matrix -Mo -Tu -1300 . csv " )
219 k [28] = Find _k ( "Estimator -Monday -1330 . csv " ,
220         "Estimator -Tuesday -1330 . csv " ,
221         "Variance -Monday -1330 . csv " ,
222         "Variance -Tuesday -1330 . csv " ,
223         "Number -Monday -1330 . csv " ,
224         "Number -Tuesday -1330 . csv " ,
225         "k -Matrix -Mo -Tu -1330 . csv " )
226 k [29] = Find _k ( "Estimator -Monday -1400 . csv " ,
227         "Estimator -Tuesday -1400 . csv " ,
228         "Variance -Monday -1400 . csv " ,
229         "Variance -Tuesday -1400 . csv " ,
230         "Number -Monday -1400 . csv " ,
231         "Number -Tuesday -1400 . csv " ,
232         "k -Matrix -Mo -Tu -1400 . csv " )
233 k [30] = Find _k ( "Estimator -Monday -1430 . csv " ,
234         "Estimator -Tuesday -1430 . csv " ,
235         "Variance -Monday -1430 . csv " ,
236         "Variance -Tuesday -1430 . csv " ,
237         "Number -Monday -1430 . csv " ,
238         "Number -Tuesday -1430 . csv " ,
239         "k -Matrix -Mo -Tu -1430 . csv " )
240 k [31] = Find _k ( "Estimator -Monday -1500 . csv " ,

```

```

241         "Estimator - Tuesday - 1500. csv" ,
242         "Variance - Monday - 1500. csv" ,
243         "Variance - Tuesday - 1500. csv" ,
244         "Number - Monday - 1500. csv" ,
245         "Number - Tuesday - 1500. csv" ,
246         "k - Matrix - Mo - Tu - 1500. csv")
247 k [32] = Find_k ("Estimator - Monday - 1530. csv" ,
248                "Estimator - Tuesday - 1530. csv" ,
249                "Variance - Monday - 1530. csv" ,
250                "Variance - Tuesday - 1530. csv" ,
251                "Number - Monday - 1530. csv" ,
252                "Number - Tuesday - 1530. csv" ,
253                "k - Matrix - Mo - Tu - 1530. csv")
254 k [33] = Find_k ("Estimator - Monday - 1600. csv" ,
255                "Estimator - Tuesday - 1600. csv" ,
256                "Variance - Monday - 1600. csv" ,
257                "Variance - Tuesday - 1600. csv" ,
258                "Number - Monday - 1600. csv" ,
259                "Number - Tuesday - 1600. csv" ,
260                "k - Matrix - Mo - Tu - 1600. csv")
261 k [34] = Find_k ("Estimator - Monday - 1630. csv" ,
262                "Estimator - Tuesday - 1630. csv" ,
263                "Variance - Monday - 1630. csv" ,
264                "Variance - Tuesday - 1630. csv" ,
265                "Number - Monday - 1630. csv" ,
266                "Number - Tuesday - 1630. csv" ,
267                "k - Matrix - Mo - Tu - 1630. csv")
268 k [35] = Find_k ("Estimator - Monday - 1700. csv" ,
269                "Estimator - Tuesday - 1700. csv" ,
270                "Variance - Monday - 1700. csv" ,
271                "Variance - Tuesday - 1700. csv" ,
272                "Number - Monday - 1700. csv" ,
273                "Number - Tuesday - 1700. csv" ,
274                "k - Matrix - Mo - Tu - 1700. csv")
275 k [36] = Find_k ("Estimator - Monday - 1730. csv" ,
276                "Estimator - Tuesday - 1730. csv" ,
277                "Variance - Monday - 1730. csv" ,
278                "Variance - Tuesday - 1730. csv" ,
279                "Number - Monday - 1730. csv" ,
280                "Number - Tuesday - 1730. csv" ,
281                "k - Matrix - Mo - Tu - 1730. csv")
282 k [37] = Find_k ("Estimator - Monday - 1800. csv" ,
283                "Estimator - Tuesday - 1800. csv" ,
284                "Variance - Monday - 1800. csv" ,
285                "Variance - Tuesday - 1800. csv" ,
286                "Number - Monday - 1800. csv" ,

```

```

287         "Number - Tuesday - 1800. csv" ,
288         "k - Matrix - Mo - Tu - 1800. csv" )
289 k [ 38 ] = Find _ k ( " Estimator - Monday - 1830. csv " ,
290         " Estimator - Tuesday - 1830. csv " ,
291         " Variance - Monday - 1830. csv " ,
292         " Variance - Tuesday - 1830. csv " ,
293         " Number - Monday - 1830. csv " ,
294         " Number - Tuesday - 1830. csv " ,
295         " k - Matrix - Mo - Tu - 1830. csv " )
296 k [ 39 ] = Find _ k ( " Estimator - Monday - 1900. csv " ,
297         " Estimator - Tuesday - 1900. csv " ,
298         " Variance - Monday - 1900. csv " ,
299         " Variance - Tuesday - 1900. csv " ,
300         " Number - Monday - 1900. csv " ,
301         " Number - Tuesday - 1900. csv " ,
302         " k - Matrix - Mo - Tu - 1900. csv " )
303 k [ 40 ] = Find _ k ( " Estimator - Monday - 1930. csv " ,
304         " Estimator - Tuesday - 1930. csv " ,
305         " Variance - Monday - 1930. csv " ,
306         " Variance - Tuesday - 1930. csv " ,
307         " Number - Monday - 1930. csv " ,
308         " Number - Tuesday - 1930. csv " ,
309         " k - Matrix - Mo - Tu - 1930. csv " )
310 k [ 41 ] = Find _ k ( " Estimator - Monday - 2000. csv " ,
311         " Estimator - Tuesday - 2000. csv " ,
312         " Variance - Monday - 2000. csv " ,
313         " Variance - Tuesday - 2000. csv " ,
314         " Number - Monday - 2000. csv " ,
315         " Number - Tuesday - 2000. csv " ,
316         " k - Matrix - Mo - Tu - 2000. csv " )
317 k [ 42 ] = Find _ k ( " Estimator - Monday - 2030. csv " ,
318         " Estimator - Tuesday - 2030. csv " ,
319         " Variance - Monday - 2030. csv " ,
320         " Variance - Tuesday - 2030. csv " ,
321         " Number - Monday - 2030. csv " ,
322         " Number - Tuesday - 2030. csv " ,
323         " k - Matrix - Mo - Tu - 2030. csv " )
324 k [ 43 ] = Find _ k ( " Estimator - Monday - 2100. csv " ,
325         " Estimator - Tuesday - 2100. csv " ,
326         " Variance - Monday - 2100. csv " ,
327         " Variance - Tuesday - 2100. csv " ,
328         " Number - Monday - 2100. csv " ,
329         " Number - Tuesday - 2100. csv " ,
330         " k - Matrix - Mo - Tu - 2100. csv " )
331 k [ 44 ] = Find _ k ( " Estimator - Monday - 2130. csv " ,
332         " Estimator - Tuesday - 2130. csv " ,

```



```

333         "Variance -Monday -2130.csv",
334         "Variance -Tuesday -2130.csv",
335         "Number -Monday -2130.csv",
336         "Number -Tuesday -2130.csv",
337         "k-Matrix -Mo -Tu -2130.csv")
338 k[45]=Find_k("Estimator -Monday -2200.csv",
339             "Estimator -Tuesday -2200.csv",
340             "Variance -Monday -2200.csv",
341             "Variance -Tuesday -2200.csv",
342             "Number -Monday -2200.csv",
343             "Number -Tuesday -2200.csv",
344             "k-Matrix -Mo -Tu -2200.csv")
345 k[46]=Find_k("Estimator -Monday -2230.csv",
346             "Estimator -Tuesday -2230.csv",
347             "Variance -Monday -2230.csv",
348             "Variance -Tuesday -2230.csv",
349             "Number -Monday -2230.csv",
350             "Number -Tuesday -2230.csv",
351             "k-Matrix -Mo -Tu -2230.csv")
352 k[47]=Find_k("Estimator -Monday -2300.csv",
353             "Estimator -Tuesday -2300.csv",
354             "Variance -Monday -2300.csv",
355             "Variance -Tuesday -2300.csv",
356             "Number -Monday -2300.csv",
357             "Number -Tuesday -2300.csv",
358             "k-Matrix -Mo -Tu -2300.csv")
359 k[48]=Find_k("Estimator -Monday -2330.csv",
360             "Estimator -Tuesday -2330.csv",
361             "Variance -Monday -2330.csv",
362             "Variance -Tuesday -2330.csv",
363             "Number -Monday -2330.csv",
364             "Number -Tuesday -2330.csv",
365             "k-Matrix -Mo -Tu -2330.csv")
366
367 # Do the whole process for every ensemble of
368     #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 program code is skipped.

```

1 library(dplyr)

```

```

2 library(exps)
3
4 Pooling=function(k_Matrix11, k_Matrix22, k_Matrix33,
5                 k_Matrix44, k_Matrix55, k_Matrix66,
6                 Day1_mean1, Day2_mean2, Day3_mean3,
7                 Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
8                 Day1_error1, Day1_Data1, k, Pool1,
9                 Pool2, Pool3, Pool4, Pool5, Pool6){
10
11 m=65
12 n=64
13 Step_Lat=0.003703
14 Step_Long=0.004206
15 Lat_min=41.786
16 Lat_max=42.02122
17 Long_min=-87.80453
18 Long_max=-87.53139
19 mn=m*n
20 Error_time_day1=0
21 d=length(Pool1)
22 if(d>1){
23   for (c in 2:d) {
24     i1=Pool1[c]%%mn
25     j1=(Pool1[c]%/%mn)+1
26     if(is.na(Day1_mean[i1, j1])==TRUE){next}
27     if(is.na(Day2_mean[i1, j1])==TRUE){
28       ticker =1
29     }
30     else{
31       Day1_mean[i1, j1]=Day1_mean[i1, j1]+Day2_mean[i1, j1]
32       ticker=2
33     }
34     if(Pool1[c]%in%Pool2==TRUE){
35       if(is.na(Day3_mean[i1, j1])==FALSE){
36         Day1_mean[i1, j1]=Day1_mean[i1, j1]+Day3_mean[i1, j1]
37         ticker=ticker+1
38       }
39       v=match(Pool1[c], Pool2)
40       Pool2=Pool2[-v]
41     }
42     if(Pool1[c]%in%Pool3==TRUE){
43       if(is.na(Day4_mean[i1, j1])==FALSE){
44         Day1_mean[i1, j1]=Day1_mean[i1, j1]+Day4_mean[i1, j1]
45         ticker=ticker+1
46       }
47       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)&

```

```

94         Day1_Data$Dropoff.Centroid.Longitude <
95             (Long_min + l * Step_Long))
96     if (nrow(as.matrix(Day1_help)) != 0) {
97         Daten = as.vector(Day1_help[, 1])
98         Daten2 = as.numeric(sub
99             ("", "", "", Daten, fixed = TRUE))
100         n=length(Daten2)
101         Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
102         Day1_error[(i - 1) * m + j, (k - 1) * m + 1]
103         = sum(Daten2)/n
104     }
105 }
106 }
107 d=length(Pool2)
108 if(d>1){
109     for (c in 2:d) {
110         i1=Pool2[c]%%mn
111         j1=(Pool2[c]%/%mn)+1
112         if(is.na(Day1_mean[i1,j1])==TRUE){next}
113         if(is.na(Day3_mean[i1,j1])==TRUE){
114             ticker =1
115         }
116         else{
117             Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day3_mean[i1,j1]
118             ticker=2
119         }
120         if(Pool2[c]%in%Pool3){
121             if(is.na(Day4_mean[i1,j1])==FALSE){
122                 Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day4_mean[i1,j1]
123                 ticker=ticker+1
124             }
125             v=match(Pool2[c],Pool3)
126             Pool3=Pool3[-v]
127         }
128         if(Pool2[c]%in%Pool4){
129             if(is.na(Day5_mean[i1,j1])==FALSE){
130                 Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
131                 ticker=ticker+1
132             }
133             v=match(Pool2[c],Pool4)
134             Pool4=Pool4[-v]
135         }
136         if(Pool2[c]%in%Pool5){
137             if(is.na(Day6_mean[i1,j1])==FALSE){
138                 Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
139                 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 + 1]
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) &

```

```

232     Day1_Data$Pickup.Centroid.Longitude <
233         (Long_min + j * Step_Long) &
234     Day1_Data$Dropoff.Centroid.Latitude >=
235         (Lat_min + (k - 1) * Step_Lat) &
236     Day1_Data$Dropoff.Centroid.Latitude <
237         (Lat_min + k * Step_Lat) &
238     Day1_Data$Dropoff.Centroid.Longitude >=
239         (Long_min + (l - 1) * Step_Long) &
240     Day1_Data$Dropoff.Centroid.Longitude <
241         (Long_min + l * Step_Long))
242     if (nrow(as.matrix(Day1_help)) != 0) {
243         Daten = as.vector(Day1_help[, 1])
244         Daten2 = as.numeric(sub
245             ("", "", "", Daten, fixed = TRUE))
246         n=length(Daten2)
247         Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
248         Day1_error[(i - 1) * m + j, (k - 1) * m + l]
249         = sum(Daten2)/n
250     }
251 }
252 }
253 d=length(Pool4)
254 if(d>1){
255     for (c in 2:d) {
256         i1=Pool4[c]%%mn
257         j1=(Pool4[c]%%mn)+1
258         if(is.na(Day1_mean[i1,j1])==TRUE){next}
259         if(is.na(Day5_mean[i1,j1])==TRUE){
260             ticker =1
261         }
262         else{
263             Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day5_mean[i1,j1]
264             ticker=2
265         }
266         if(Pool4[c]%in%Pool5){
267             if(is.na(Day6_mean[i1,j1])==FALSE){
268                 Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
269                 ticker=ticker+1
270             }
271             v=match(Pool4[c],Pool5)
272             Pool5=Pool5[-v]
273         }
274         if(Pool4[c]%in%Pool6){
275             if(is.na(Day7_mean[i1,j1])==FALSE){
276                 Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
277                 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 + 1]
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         }

```



```

324     else{
325         Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day6_mean[i1,j1]
326         ticker=2
327     }
328     if(Pool5[c]%in%Pool6){
329         if(is.na(Day7_mean[i1,j1])==FALSE){
330             Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
331             ticker=ticker+1
332         }
333         v=match(Pool5[c],Pool6)
334         Pool6=Pool6[-v]
335     }
336     Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
337     i=(i1%%m)+1
338     j=i1%%m
339     k=(j1%%m)+1
340     l=j1%%m
341     Day1_help=filter(Day1_Data,
342                     Day1_Data$Pickup.Centroid.Latitude >=
343                         (Lat_min + (i - 1) * Step_Lat)&
344                     Day1_Data$Pickup.Centroid.Latitude <
345                         (Lat_min + i * Step_Lat) &
346                     Day1_Data$Pickup.Centroid.Longitude >=
347                         (Long_min + (j - 1) * Step_Long) &
348                     Day1_Data$Pickup.Centroid.Longitude <
349                         (Long_min + j * Step_Long) &
350                     Day1_Data$Dropoff.Centroid.Latitude >=
351                         (Lat_min + (k - 1) * Step_Lat) &
352                     Day1_Data$Dropoff.Centroid.Latitude <
353                         (Lat_min + k * Step_Lat) &
354                     Day1_Data$Dropoff.Centroid.Longitude >=
355                         (Long_min + (l - 1) * Step_Long)&
356                     Day1_Data$Dropoff.Centroid.Longitude <
357                         (Long_min + l * Step_Long))
358     if (nrow(as.matrix(Day1_help)) != 0) {
359         Daten = as.vector(Day1_help[, 1])
360         Daten2 = as.numeric(sub
361             ("", "", "", Daten, fixed = TRUE))
362         n=length(Daten2)
363         Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
364         Day1_error[(i - 1) * m + j, (k - 1) * m + l]
365         = sum(Daten2)/n
366     }
367 }
368 }
369 d=length(Pool6)

```

```

370   if(d>1){
371     for (c in 2:d) {
372       i1=Pool6[c]%%mn
373       j1=(Pool6[c]%/%mn)+1
374       if(is.na(Day1_mean[i1,j1])==TRUE){next}
375       if(is.na(Day7_mean[i1,j1])==TRUE){
376         ticker =1
377       }
378       else{
379         Day1_mean[i1,j1]=Day1_mean[i1,j1]+Day7_mean[i1,j1]
380         ticker=2
381       }
382       Day1_mean[i1,j1]=Day1_mean[i1,j1]/ticker
383       i=(i1%%m)-1
384       j=i1%%m+1
385       k=(j1%%m)-1
386       l=j1%%m
387       Day1_help=filter(Day1_Data,
388         Day1_Data$Pickup.Centroid.Latitude >=
389           (Lat_min + (i - 1) * Step_Lat)&
390         Day1_Data$Pickup.Centroid.Latitude <
391           (Lat_min + i * Step_Lat) &
392         Day1_Data$Pickup.Centroid.Longitude >=
393           (Long_min + (j - 1) * Step_Long) &
394         Day1_Data$Pickup.Centroid.Longitude <
395           (Long_min + j * Step_Long) &
396         Day1_Data$Dropoff.Centroid.Latitude >=
397           (Lat_min + (k - 1) * Step_Lat) &
398         Day1_Data$Dropoff.Centroid.Latitude <
399           (Lat_min + k * Step_Lat) &
400         Day1_Data$Dropoff.Centroid.Longitude >=
401           (Long_min + (l - 1) * Step_Long)&
402         Day1_Data$Dropoff.Centroid.Longitude <
403           (Long_min + l * Step_Long))
404       if (nrow(as.matrix(Day1_help)) != 0) {
405         Daten = as.vector(Day1_help[, 1])
406         Daten2 = as.numeric(sub
407           ("", "", "", Daten, fixed = TRUE))
408         n=length(Daten2)
409         Daten2=abs(Daten2-rep(Day1_mean[i1,j1],n))
410         Day1_error[(i - 1) * m + j, (k - 1) * m + l]
411         = sum(Daten2)/n
412       }
413     }
414   }
415   Error_i=rep(0,mn)

```

```

416   for (e in 1:mn) {
417     Help=Day1_error[,e]
418
419     Error_i[e]=mean(Help, na.rm = TRUE)
420     if(is.nan(Error_i[e])==TRUE){Error_i[e]=NA}
421   }
422   Error_time_day1=mean(Error_i, na.rm = TRUE)
423   return(Error_time_day1)
424 }
425 Error_Mo_16=rep(0,48)
426 k=16
427
428 k_Matrix1=read.csv("k-Matrix-Mo-Tu-0000.csv")
429 k_Matrix1=k_Matrix1[, -1]
430 k_Matrix2=read.csv("k-Matrix-Mo-We-0000.csv")
431 k_Matrix2=k_Matrix2[, -1]
432 k_Matrix3=read.csv("k-Matrix-Mo-Th-0000.csv")
433 k_Matrix3=k_Matrix3[, -1]
434 k_Matrix4=read.csv("k-Matrix-Mo-Fr-0000.csv")
435 k_Matrix4=k_Matrix4[, -1]
436 k_Matrix5=read.csv("k-Matrix-Mo-Sa-0000.csv")
437 k_Matrix5=k_Matrix5[, -1]
438 k_Matrix6=read.csv("k-Matrix-Mo-Su-0000.csv")
439 k_Matrix6=k_Matrix6[, -1]
440 Day1_mean=read.csv("Estimator-Monday-0000.csv")
441 Day1_mean=Day1_mean[, -1]
442 Day2_mean=read.csv("Estimator-Tuesday-0000.csv")
443 Day2_mean=Day2_mean[, -1]
444 Day3_mean=read.csv("Estimator-Wednesday-0000.csv")
445 Day3_mean=Day3_mean[, -1]
446 Day4_mean=read.csv("Estimator-Thursday-0000.csv")
447 Day4_mean=Day4_mean[, -1]
448 Day5_mean=read.csv("Estimator-Friday-0000.csv")
449 Day5_mean=Day5_mean[, -1]
450 Day6_mean=read.csv("Estimator-Saturday-0000.csv")
451 Day6_mean=Day6_mean[, -1]
452 Day7_mean=read.csv("Estimator-Sunday-0000.csv")
453 Day7_mean=Day7_mean[, -1]
454 Day1_error=read.csv("Error-ij-Monday-0000.csv")
455 Day1_error=Day1_error[, -1]
456 Day1_Data=read.csv("Monday-0000-2019.csv")
457 Day1_Data=Day1_Data[, 5:9]
458 Pool1=which(k_Matrix1>=0 &k_Matrix1<=k)
459 Pool1=c(0, Pool1)
460 Pool2=which(k_Matrix2>=0 &k_Matrix2<=k)
461 Pool2=c(0, Pool2)

```

```

462 Pool3=which(k_Matrix3>=0 &k_Matrix3 <=k)
463 Pool3=c(0,Pool3)
464 Pool4=which(k_Matrix4>=0 &k_Matrix4 <=k)
465 Pool4=c(0,Pool4)
466 Pool5=which(k_Matrix5>=0 &k_Matrix5 <=k)
467 Pool5=c(0,Pool5)
468 Pool6=which(k_Matrix6>=0 &k_Matrix6 <=k)
469 Pool6=c(0,Pool6)
470 Error_Mo_16[1]=Pooling(k_Matrix11, k_Matrix22, k_Matrix33,
471     k_Matrix44, k_Matrix55, k_Matrix66,
472     Day1_mean1, Day2_mean2, Day3_mean3,
473     Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
474     Day1_error1, Day1_Data1, k ,Pool1, Pool2,
475     Pool3, Pool4, Pool5, Pool6)
476 k_Matrix1=NULL
477 k_Matrix2=NULL
478 k_Matrix3=NULL
479 k_Matrix4=NULL
480 k_Matrix5=NULL
481 k_Matrix6=NULL
482 Day1_mean=NULL
483 Day2_mean=NULL
484 Day3_mean=NULL
485 Day4_mean=NULL
486 Day5_mean=NULL
487 Day6_mean=NULL
488 Day7_mean=NULL
489 Day1_error=NULL
490 Day1_Data=NULL
491 Pool1=NULL
492 Pool2=NULL
493 Pool3=NULL
494 Pool4=NULL
495 Pool5=NULL
496 Pool6=NULL
497
498 k_Matrix1=read.csv("k-Matrix-Mo-Tu-0030.csv")
499 k_Matrix1=k_Matrix1[,-1]
500 k_Matrix2=read.csv("k-Matrix-Mo-We-0030.csv")
501 k_Matrix2=k_Matrix2[,-1]
502 k_Matrix3=read.csv("k-Matrix-Mo-Th-0030.csv")
503 k_Matrix3=k_Matrix3[,-1]
504 k_Matrix4=read.csv("k-Matrix-Mo-Fr-0030.csv")
505 k_Matrix4=k_Matrix4[,-1]
506 k_Matrix5=read.csv("k-Matrix-Mo-Sa-0030.csv")
507 k_Matrix5=k_Matrix5[,-1]

```

```

508 k_Matrix6=read.csv("k-Matrix-Mo-Su-0030.csv")
509 k_Matrix6=k_Matrix6[, -1]
510 Day1_mean=read.csv("Estimator-Monday-0030.csv")
511 Day1_mean=Day1_mean[, -1]
512 Day2_mean=read.csv("Estimator-Tuesday-0030.csv")
513 Day2_mean=Day2_mean[, -1]
514 Day3_mean=read.csv("Estimator-Wednesday-0030.csv")
515 Day3_mean=Day3_mean[, -1]
516 Day4_mean=read.csv("Estimator-Thursday-0030.csv")
517 Day4_mean=Day4_mean[, -1]
518 Day5_mean=read.csv("Estimator-Friday-0030.csv")
519 Day5_mean=Day5_mean[, -1]
520 Day6_mean=read.csv("Estimator-Saturday-0030.csv")
521 Day6_mean=Day6_mean[, -1]
522 Day7_mean=read.csv("Estimator-Sunday-0030.csv")
523 Day7_mean=Day7_mean[, -1]
524 Day1_error=read.csv("Error-ij-Monday-0030.csv")
525 Day1_error=Day1_error[, -1]
526 Day1_Data=read.csv("Monday-0030-2019.csv")
527 Day1_Data=Day1_Data[, 5:9]
528 Pool1=which(k_Matrix1>=0 &k_Matrix1<=k)
529 Pool1=c(0, Pool1)
530 Pool2=which(k_Matrix2>=0 &k_Matrix2<=k)
531 Pool2=c(0, Pool2)
532 Pool3=which(k_Matrix3>=0 &k_Matrix3<=k)
533 Pool3=c(0, Pool3)
534 Pool4=which(k_Matrix4>=0 &k_Matrix4<=k)
535 Pool4=c(0, Pool4)
536 Pool5=which(k_Matrix5>=0 &k_Matrix5<=k)
537 Pool5=c(0, Pool5)
538 Pool6=which(k_Matrix6>=0 &k_Matrix6<=k)
539 Pool6=c(0, Pool6)
540 Error_Mo_16[1]=Pooling(k_Matrix11, k_Matrix22, k_Matrix33,
541     k_Matrix44, k_Matrix55, k_Matrix66,
542     Day1_mean1, Day2_mean2, Day3_mean3,
543     Day4_mean4, Day5_mean5, Day6_mean6, Day7_mean7,
544     Day1_error1, Day1_Data1, k , Pool1, Pool2,
545     Pool3, Pool4, Pool5, Pool6)
546 k_Matrix1=NULL
547 k_Matrix2=NULL
548 k_Matrix3=NULL
549 k_Matrix4=NULL
550 k_Matrix5=NULL
551 k_Matrix6=NULL
552 Day1_mean=NULL
553 Day2_mean=NULL

```

```

554 Day3_mean=NULL
555 Day4_mean=NULL
556 Day5_mean=NULL
557 Day6_mean=NULL
558 Day7_mean=NULL
559 Day1_error=NULL
560 Day1_Data=NULL
561 Pool1=NULL
562 Pool2=NULL
563 Pool3=NULL
564 Pool4=NULL
565 Pool5=NULL
566 Pool6=NULL
567
568 #Do the same process for every time interval t and
569     #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.

```

1
2 #Find the error E(t,d) and E(d)
3
4 library(dplyr)
5 library(exps)
6 Times=c("12am", "12.30am", "1am", "1.30am", "2am", "2.30am",
7         "3am", "3.30am", "4am", "4.30am", "5am", "5.30am",
8         "6am", "6.30am", "7am", "7.30am", "8am", "8.30am",
9         "9am", "9.30am", "10am", "10.30am", "11am", "11.30am",
10        "12pm", "12.30pm", "1pm", "1.30pm", "2pm", "2.30pm",
11        "3pm", "3.30pm", "4pm", "4.30pm", "5pm", "5.30pm",
12        "6pm", "6.30pm", "7pm", "7.30pm", "8pm", "8.30pm",
13        "9pm", "9.30pm", "10pm", "10.30pm", "11pm", "11.30pm")
14
15 Error_i_Monday_0000=read.csv("Error-i-Monday-0000.csv")
16 Error_i_Monday_0000=Error_i_Monday_0000[, -1]
17 Error_i_Monday_0030=read.csv("Error-i-Monday-0030.csv")
18 Error_i_Monday_0030=Error_i_Monday_0030[, -1]
19 Error_i_Monday_0100=read.csv("Error-i-Monday-0100.csv")
20 Error_i_Monday_0100=Error_i_Monday_0100[, -1]
21 Error_i_Monday_0130=read.csv("Error-i-Monday-0130.csv")

```

```
22 Error_i_Monday_0130=Error_i_Monday_0130[, -1]
23 Error_i_Monday_0200=read.csv("Error-i-Monday-0200.csv")
24 Error_i_Monday_0200=Error_i_Monday_0200[, -1]
25 Error_i_Monday_0230=read.csv("Error-i-Monday-0230.csv")
26 Error_i_Monday_0230=Error_i_Monday_0230[, -1]
27 Error_i_Monday_0300=read.csv("Error-i-Monday-0300.csv")
28 Error_i_Monday_0300=Error_i_Monday_0300[, -1]
29 Error_i_Monday_0330=read.csv("Error-i-Monday-0330.csv")
30 Error_i_Monday_0330=Error_i_Monday_0330[, -1]
31 Error_i_Monday_0400=read.csv("Error-i-Monday-0400.csv")
32 Error_i_Monday_0400=Error_i_Monday_0400[, -1]
33 Error_i_Monday_0430=read.csv("Error-i-Monday-0430.csv")
34 Error_i_Monday_0430=Error_i_Monday_0430[, -1]
35 Error_i_Monday_0500=read.csv("Error-i-Monday-0500.csv")
36 Error_i_Monday_0500=Error_i_Monday_0500[, -1]
37 Error_i_Monday_0530=read.csv("Error-i-Monday-0530.csv")
38 Error_i_Monday_0530=Error_i_Monday_0530[, -1]
39 Error_i_Monday_0600=read.csv("Error-i-Monday-0600.csv")
40 Error_i_Monday_0600=Error_i_Monday_0600[, -1]
41 Error_i_Monday_0630=read.csv("Error-i-Monday-0630.csv")
42 Error_i_Monday_0630=Error_i_Monday_0630[, -1]
43 Error_i_Monday_0700=read.csv("Error-i-Monday-0700.csv")
44 Error_i_Monday_0700=Error_i_Monday_0700[, -1]
45 Error_i_Monday_0730=read.csv("Error-i-Monday-0730.csv")
46 Error_i_Monday_0730=Error_i_Monday_0730[, -1]
47 Error_i_Monday_0800=read.csv("Error-i-Monday-0800.csv")
48 Error_i_Monday_0800=Error_i_Monday_0800[, -1]
49 Error_i_Monday_0830=read.csv("Error-i-Monday-0830.csv")
50 Error_i_Monday_0830=Error_i_Monday_0830[, -1]
51 Error_i_Monday_0900=read.csv("Error-i-Monday-0900.csv")
52 Error_i_Monday_0900=Error_i_Monday_0900[, -1]
53 Error_i_Monday_0930=read.csv("Error-i-Monday-0930.csv")
54 Error_i_Monday_0930=Error_i_Monday_0930[, -1]
55 Error_i_Monday_1000=read.csv("Error-i-Monday-1000.csv")
56 Error_i_Monday_1000=Error_i_Monday_1000[, -1]
57 Error_i_Monday_1030=read.csv("Error-i-Monday-1030.csv")
58 Error_i_Monday_1030=Error_i_Monday_1030[, -1]
59 Error_i_Monday_1100=read.csv("Error-i-Monday-1100.csv")
60 Error_i_Monday_1100=Error_i_Monday_1100[, -1]
61 Error_i_Monday_1130=read.csv("Error-i-Monday-1130.csv")
62 Error_i_Monday_1130=Error_i_Monday_1130[, -1]
63 Error_i_Monday_1200=read.csv("Error-i-Monday-1200.csv")
64 Error_i_Monday_1200=Error_i_Monday_1200[, -1]
65 Error_i_Monday_1230=read.csv("Error-i-Monday-1230.csv")
66 Error_i_Monday_1230=Error_i_Monday_1230[, -1]
67 Error_i_Monday_1300=read.csv("Error-i-Monday-1300.csv")
```

```

68 Error_i_Monday_1300=Error_i_Monday_1300[, -1]
69 Error_i_Monday_1330=read.csv("Error-i-Monday-1330.csv")
70 Error_i_Monday_1330=Error_i_Monday_1330[, -1]
71 Error_i_Monday_1400=read.csv("Error-i-Monday-1400.csv")
72 Error_i_Monday_1400=Error_i_Monday_1400[, -1]
73 Error_i_Monday_1430=read.csv("Error-i-Monday-1430.csv")
74 Error_i_Monday_1430=Error_i_Monday_1430[, -1]
75 Error_i_Monday_1500=read.csv("Error-i-Monday-1500.csv")
76 Error_i_Monday_1500=Error_i_Monday_1500[, -1]
77 Error_i_Monday_1530=read.csv("Error-i-Monday-1530.csv")
78 Error_i_Monday_1530=Error_i_Monday_1530[, -1]
79 Error_i_Monday_1600=read.csv("Error-i-Monday-1600.csv")
80 Error_i_Monday_1600=Error_i_Monday_1600[, -1]
81 Error_i_Monday_1630=read.csv("Error-i-Monday-1630.csv")
82 Error_i_Monday_1630=Error_i_Monday_1630[, -1]
83 Error_i_Monday_1700=read.csv("Error-i-Monday-1700.csv")
84 Error_i_Monday_1700=Error_i_Monday_1700[, -1]
85 Error_i_Monday_1730=read.csv("Error-i-Monday-1730.csv")
86 Error_i_Monday_1730=Error_i_Monday_1730[, -1]
87 Error_i_Monday_1800=read.csv("Error-i-Monday-1800.csv")
88 Error_i_Monday_1800=Error_i_Monday_1800[, -1]
89 Error_i_Monday_1830=read.csv("Error-i-Monday-1830.csv")
90 Error_i_Monday_1830=Error_i_Monday_1830[, -1]
91 Error_i_Monday_1900=read.csv("Error-i-Monday-1900.csv")
92 Error_i_Monday_1900=Error_i_Monday_1900[, -1]
93 Error_i_Monday_1930=read.csv("Error-i-Monday-1930.csv")
94 Error_i_Monday_1930=Error_i_Monday_1930[, -1]
95 Error_i_Monday_2000=read.csv("Error-i-Monday-2000.csv")
96 Error_i_Monday_2000=Error_i_Monday_2000[, -1]
97 Error_i_Monday_2030=read.csv("Error-i-Monday-2030.csv")
98 Error_i_Monday_2030=Error_i_Monday_2030[, -1]
99 Error_i_Monday_2100=read.csv("Error-i-Monday-2100.csv")
100 Error_i_Monday_2100=Error_i_Monday_2100[, -1]
101 Error_i_Monday_2130=read.csv("Error-i-Monday-2130.csv")
102 Error_i_Monday_2130=Error_i_Monday_2130[, -1]
103 Error_i_Monday_2200=read.csv("Error-i-Monday-2200.csv")
104 Error_i_Monday_2200=Error_i_Monday_2200[, -1]
105 Error_i_Monday_2230=read.csv("Error-i-Monday-2230.csv")
106 Error_i_Monday_2230=Error_i_Monday_2230[, -1]
107 Error_i_Monday_2300=read.csv("Error-i-Monday-2300.csv")
108 Error_i_Monday_2300=Error_i_Monday_2300[, -1]
109 Error_i_Monday_2330=read.csv("Error-i-Monday-2330.csv")
110 Error_i_Monday_2330=Error_i_Monday_2330[, -1]
111
112 Error_Monday=rep(0,48)
113 Error_Monday[1]=mean(Error_i_Monday_0000, na.rm = TRUE)

```



```
114 Error_Monday [2]=mean(Error_i_Monday_0030, na.rm = TRUE)
115 Error_Monday [3]=mean(Error_i_Monday_0100, na.rm = TRUE)
116 Error_Monday [4]=mean(Error_i_Monday_0130, na.rm = TRUE)
117 Error_Monday [5]=mean(Error_i_Monday_0200, na.rm = TRUE)
118 Error_Monday [6]=mean(Error_i_Monday_0230, na.rm = TRUE)
119 Error_Monday [7]=mean(Error_i_Monday_0300, na.rm = TRUE)
120 Error_Monday [8]=mean(Error_i_Monday_0330, na.rm = TRUE)
121 Error_Monday [9]=mean(Error_i_Monday_0400, na.rm = TRUE)
122 Error_Monday [10]=mean(Error_i_Monday_0430, na.rm = TRUE)
123 Error_Monday [11]=mean(Error_i_Monday_0500, na.rm = TRUE)
124 Error_Monday [12]=mean(Error_i_Monday_0530, na.rm = TRUE)
125 Error_Monday [13]=mean(Error_i_Monday_0600, na.rm = TRUE)
126 Error_Monday [14]=mean(Error_i_Monday_0630, na.rm = TRUE)
127 Error_Monday [15]=mean(Error_i_Monday_0700, na.rm = TRUE)
128 Error_Monday [16]=mean(Error_i_Monday_0730, na.rm = TRUE)
129 Error_Monday [17]=mean(Error_i_Monday_0800, na.rm = TRUE)
130 Error_Monday [18]=mean(Error_i_Monday_0830, na.rm = TRUE)
131 Error_Monday [19]=mean(Error_i_Monday_0900, na.rm = TRUE)
132 Error_Monday [20]=mean(Error_i_Monday_0930, na.rm = TRUE)
133 Error_Monday [21]=mean(Error_i_Monday_1000, na.rm = TRUE)
134 Error_Monday [22]=mean(Error_i_Monday_1030, na.rm = TRUE)
135 Error_Monday [23]=mean(Error_i_Monday_1100, na.rm = TRUE)
136 Error_Monday [24]=mean(Error_i_Monday_1130, na.rm = TRUE)
137 Error_Monday [25]=mean(Error_i_Monday_1200, na.rm = TRUE)
138 Error_Monday [26]=mean(Error_i_Monday_1230, na.rm = TRUE)
139 Error_Monday [27]=mean(Error_i_Monday_1300, na.rm = TRUE)
140 Error_Monday [28]=mean(Error_i_Monday_1330, na.rm = TRUE)
141 Error_Monday [29]=mean(Error_i_Monday_1400, na.rm = TRUE)
142 Error_Monday [30]=mean(Error_i_Monday_1430, na.rm = TRUE)
143 Error_Monday [31]=mean(Error_i_Monday_1500, na.rm = TRUE)
144 Error_Monday [32]=mean(Error_i_Monday_1530, na.rm = TRUE)
145 Error_Monday [33]=mean(Error_i_Monday_1600, na.rm = TRUE)
146 Error_Monday [34]=mean(Error_i_Monday_1630, na.rm = TRUE)
147 Error_Monday [35]=mean(Error_i_Monday_1700, na.rm = TRUE)
148 Error_Monday [36]=mean(Error_i_Monday_1730, na.rm = TRUE)
149 Error_Monday [37]=mean(Error_i_Monday_1800, na.rm = TRUE)
150 Error_Monday [38]=mean(Error_i_Monday_1830, na.rm = TRUE)
151 Error_Monday [39]=mean(Error_i_Monday_1900, na.rm = TRUE)
152 Error_Monday [40]=mean(Error_i_Monday_1930, na.rm = TRUE)
153 Error_Monday [41]=mean(Error_i_Monday_2000, na.rm = TRUE)
154 Error_Monday [42]=mean(Error_i_Monday_2030, na.rm = TRUE)
155 Error_Monday [43]=mean(Error_i_Monday_2100, na.rm = TRUE)
156 Error_Monday [44]=mean(Error_i_Monday_2130, na.rm = TRUE)
157 Error_Monday [45]=mean(Error_i_Monday_2200, na.rm = TRUE)
158 Error_Monday [46]=mean(Error_i_Monday_2230, na.rm = TRUE)
159 Error_Monday [47]=mean(Error_i_Monday_2300, na.rm = TRUE)
```

```

160 Error_Monday[48]=mean(Error_i_Monday_2330, na.rm = TRUE)
161 write.csv(Error_Monday,"Error-Monday.csv")
162
163 #Do the same process for every day.
164
165 #Plots
166 par(mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
167 plot(seq(1,48,1),Error_Monday,
168      main = "The Error depending on the time interval t
169             for the several days", type = "l",
170      xaxt="n", xlab = "Time Interval Starting Point",
171      ylab = "Error E(t,d)", col="red", lwd=2,
172      ylim = c(0,2200), pch=19)
173 axis(1,at = seq(1,48,1),label=Times)
174 lines(Error_Wednesday, type="l", col="yellow", lwd=2,pch=19)
175 lines(Error_Thursday, type="l", col="green", lwd=2,pch=19)
176 lines(Error_Saturday, type="l", col="brown", lwd=2,pch=19)
177 legend("topright",inset = c(-0.25,0),
178      legend=c("Monday","Wednesday", "Thursday", "Saturday"),
179      col=c("red", "yellow", "green", "brown"),cex=0.8,lty=1)
180
181 plot(seq(1,48,1),Error_Tuesday,
182      main = "The Error depending on the time interval t
183             for the several days", type = "l",
184      xaxt="n", xlab = "Time Interval Starting Point",
185      ylab = "Error E(t,d)", col="orange", lwd=2,
186      ylim = c(0,9000))
187 axis(1,at = seq(1,48,1),label=Times)
188 lines(Error_Friday, type = "l", col="blue", lwd=2)
189 lines(Error_Sunday,type = "l", col="black", lwd=2)
190 legend("topright",inset = c(-0.22,0),
191      legend=c("Tuesday","Friday", "Sunday"),
192      col=c("orange", "blue", "black"),cex=0.8,lty=1)
193
194 Error_Tuesday2=Error_Tuesday
195 Error_Tuesday2[Error_Tuesday2>1500]=1500
196 Error_Friday2=Error_Friday
197 Error_Friday2[Error_Friday2>1500]=1500
198 Error_Sunday2=Error_Sunday
199 Error_Sunday2[Error_Sunday2>1500]=1500
200
201 plot(seq(1,48,1),Error_Tuesday2,
202      main = "The Error depending on the time interval t
203             for the several days", type = "l",
204      xaxt="n", xlab = "Time Interval Starting Point",
205      ylab = "Error E(t,d)", col="orange", lwd=2,

```

```

206     ylim = c(0,1500))
207 axis(1, at = seq(1,48,1), label=Times)
208 lines(Error_Friday2, type = "l", col="blue", lwd=2)
209 lines(Error_Sunday2, type = "l", col="black", lwd=2)
210 legend("topright", inset = c(-0.22,0),
211       legend=c("Tuesday", "Friday", "Sunday"),
212       col=c("orange", "blue", "black"), cex=0.8, lty=1)
213 mean(Error_Monday)
214
215 plot(seq(1,48,1), Error_Tuesday,
216      main = "The Error depending on the time interval t
217            for the day Tuesday", type = "l", xaxt="n",
218            xlab = "Time Interval Start", ylab = "Error E(t,d)")
219 axis(1, at = seq(1,48,1), label=Times)
220 mean(Error_Tuesday)
221
222 plot(seq(1,48,1), Error_Wednesday,
223      main = "The Error depending on the time interval t
224            for the day Wednesday", type = "l", xaxt="n",
225            xlab = "Time Interval Start", ylab = "Error E(t,d)")
226 axis(1, at = seq(1,48,1), label=Times)
227 mean(Error_Wednesday)
228
229 plot(seq(1,48,1), Error_Thursday,
230      main = "The Error depending on the time interval t
231            for the day Thursday", type = "l", xaxt="n",
232            xlab = "Time Interval Start", ylab = "Error E(t,d)")
233 axis(1, at = seq(1,48,1), label=Times)
234 mean(Error_Thursday)
235
236 plot(seq(1,48,1), Error_Friday,
237      main = "The Error depending on the time interval t
238            for the day Friday", type = "l", xaxt="n",
239            xlab = "Time Interval Start", ylab = "Error E(t,d)")
240 axis(1, at = seq(1,48,1), label=Times)
241 mean(Error_Friday)
242
243 plot(seq(1,48,1), Error_Saturday,
244      type = "l", xaxt="n", xlab = "Time Interval Start",
245      ylab = "Error E(t,d)")
246 axis(1, at = seq(1,48,1), label=Times)
247 mean(Error_Saturday)
248
249 plot(seq(1,48,1), Error_Sunday,
250      type = "l", xaxt="n", xlab = "Time Interval Start",
251      ylab = "Error E(t,d)")

```

```

252 axis(1, at = seq(1,48,1), label=Times)
253 mean(Error_Sunday)
254
255 #Boxplots
256 par(mar=c(5.1, 4.1, 4.1, 2.1), mgp=c(3, 1, 0), las=0)
257 boxplot(Error_i_Friday_1400[Error_i_Friday_1400>0],
258         Error_i_Friday_1730[Error_i_Friday_1730>0],
259         Error_i_Friday_2200[Error_i_Friday_2200>0],
260         horizontal = TRUE,
261         main = "Boxplot of the error E(i,t,d) for
262               selected days and time intervals",
263         names = c("Friday 2 pm", "Friday 5.30 pm",
264                  "Friday 10 pm"))
265
266 boxplot(Error_i_Friday_1400[Error_i_Friday_1400>0],
267         Error_i_Friday_1730[Error_i_Friday_1730>0],
268         Error_i_Friday_2200[Error_i_Friday_2200>0],
269         horizontal = TRUE, outline = FALSE,
270         main = "Boxplot of the error E(i,t,d) for
271               selected days and time intervals without outlines",
272         names = c("Friday 2pm", "Friday 5.30 pm", "Friday 10 pm"))
273 max(Error_i_Friday_1400[Error_i_Friday_1400>0])
274
275 boxplot(Error_i_Sunday_0500[Error_i_Sunday_0500>0],
276         horizontal = TRUE,
277         main = "Boxplot of the error E(i,t,d)",
278         ylab = "Sunday 5 am")
279 boxplot(Error_i_Tuesday_2100[Error_i_Tuesday_2100>0],
280         Error_i_Friday_0300[Error_i_Friday_0300>0],
281         horizontal = TRUE,
282         main= "Boxplot of the error E(i,t,d) with outliers",
283         names=c("Tuesday 9 pm", "Friday 3 am"))
284
285 boxplot(Error_i_Tuesday_2100[Error_i_Tuesday_2100>0],
286         Error_i_Friday_0300[Error_i_Friday_0300>0],
287         horizontal = TRUE,
288         main= "Boxplot of the error E(i,t,d) without outliers",
289         names=c("Tuesday 9 pm", "Friday 3 am"), outline = FALSE)
290
291 Error_Tuesday
292
293 #Find number of error E(i,t,d)
294
295 number_Monday=rep(0,48)
296 number_Monday[1]= length(which(Error_i_Monday_0000>0))
297 number_Monday[2]= length(which(Error_i_Monday_0030>0))

```

```
298 number_Monday[3]= length(which(Error_i_Monday_0100>0))
299 number_Monday[4]= length(which(Error_i_Monday_0130>0))
300 number_Monday[5]= length(which(Error_i_Monday_0200>0))
301 number_Monday[6]= length(which(Error_i_Monday_0230>0))
302 number_Monday[7]= length(which(Error_i_Monday_0300>0))
303 number_Monday[8]= length(which(Error_i_Monday_0330>0))
304 number_Monday[9]= length(which(Error_i_Monday_0400>0))
305 number_Monday[10]= length(which(Error_i_Monday_0430>0))
306 number_Monday[11]= length(which(Error_i_Monday_0500>0))
307 number_Monday[12]= length(which(Error_i_Monday_0530>0))
308 number_Monday[13]= length(which(Error_i_Monday_0600>0))
309 number_Monday[14]= length(which(Error_i_Monday_0630>0))
310 number_Monday[15]= length(which(Error_i_Monday_0700>0))
311 number_Monday[16]= length(which(Error_i_Monday_0730>0))
312 number_Monday[17]= length(which(Error_i_Monday_0800>0))
313 number_Monday[18]= length(which(Error_i_Monday_0830>0))
314 number_Monday[19]= length(which(Error_i_Monday_0900>0))
315 number_Monday[20]= length(which(Error_i_Monday_0930>0))
316 number_Monday[21]= length(which(Error_i_Monday_1000>0))
317 number_Monday[22]= length(which(Error_i_Monday_1030>0))
318 number_Monday[23]= length(which(Error_i_Monday_1100>0))
319 number_Monday[24]= length(which(Error_i_Monday_1130>0))
320 number_Monday[25]= length(which(Error_i_Monday_1200>0))
321 number_Monday[26]= length(which(Error_i_Monday_1230>0))
322 number_Monday[27]= length(which(Error_i_Monday_1300>0))
323 number_Monday[28]= length(which(Error_i_Monday_1330>0))
324 number_Monday[29]= length(which(Error_i_Monday_1400>0))
325 number_Monday[30]= length(which(Error_i_Monday_1430>0))
326 number_Monday[31]= length(which(Error_i_Monday_1500>0))
327 number_Monday[32]= length(which(Error_i_Monday_1530>0))
328 number_Monday[33]= length(which(Error_i_Monday_1600>0))
329 number_Monday[34]= length(which(Error_i_Monday_1630>0))
330 number_Monday[35]= length(which(Error_i_Monday_1700>0))
331 number_Monday[36]= length(which(Error_i_Monday_1730>0))
332 number_Monday[37]= length(which(Error_i_Monday_1800>0))
333 number_Monday[38]= length(which(Error_i_Monday_1830>0))
334 number_Monday[39]= length(which(Error_i_Monday_1900>0))
335 number_Monday[40]= length(which(Error_i_Monday_1930>0))
336 number_Monday[41]= length(which(Error_i_Monday_2000>0))
337 number_Monday[42]= length(which(Error_i_Monday_2030>0))
338 number_Monday[43]= length(which(Error_i_Monday_2100>0))
339 number_Monday[44]= length(which(Error_i_Monday_2130>0))
340 number_Monday[45]= length(which(Error_i_Monday_2200>0))
341 number_Monday[46]= length(which(Error_i_Monday_2230>0))
342 number_Monday[47]= length(which(Error_i_Monday_2300>0))
343 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)

```

```

390 quantile(Monday[,7],0.05)
391 max(Monday[,7])
392 min(Monday[,8])
393 quantile(Monday[,8],0.01)
394 quantile(Monday[,8],0.05)
395 quantile(Monday[,8],0.1)
396 max(Monday[,8])
397
398
399 Mo_1=filter(Monday,
400     Monday$Pickup.Centroid.Latitude>41.786 &
401     Monday$Dropoff.Centroid.Latitude>41.786 &
402     Monday$Pickup.Centroid.Longitude>-87.80453 &
403     Monday$Dropoff.Centroid.Longitude>-87.80453)
404 plot(Mo_1[,6], Mo_1[,5],main="The pickup places in terms
405     of latitude and longitude after the cut",
406     xlab="Longitude", ylab="Latitude")
407 min(Mo_1[,5])
408 max(Mo_1[,5])
409 min(Mo_1[,6])
410 max(Mo_1[,6])
411
412 plot(Mo_1[,8], Mo_1[,7],main="The dropoff places in terms
413     of latitude and longitude after the cut",
414     xlab="Longitude", ylab="Latitude")
415 min(Mo_1[,7])
416 max(Mo_1[,7])
417 min(Mo_1[,8])
418 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.

```

1 library(dplyr)
2 library(exps)
3
4 Error_Monday=read.csv("Error-Monday.csv")
5 Error_Monday=Error_Monday[, -1]
6 Error_Tuesday=read.csv("Error-Tuesday.csv")
7 Error_Tuesday=Error_Tuesday[, -1]
8 Error_Wednesday=read.csv("Error-Wednesday.csv")

```

```

9 | Error_Wednesday=Error_Wednesday[, -1]
10 | Error_Thursday=read.csv("Error-Thursday.csv")
11 | Error_Thursday=Error_Thursday[, -1]
12 | Error_Friday=read.csv("Error-Friday.csv")
13 | Error_Friday=Error_Friday[, -1]
14 | Error_Saturday=read.csv("Error-Saturday.csv")
15 | Error_Saturday=Error_Saturday[, -1]
16 | Error_Sunday=read.csv("Error-Sunday.csv")
17 | Error_Sunday=Error_Sunday[, -1]
18 |
19 | Error_Mo_1=read.csv("Error-Mo-1.csv")
20 | Error_Mo_1=Error_Mo_1[, -1]
21 | Error_Tu_1=read.csv("Error-Tu-1.csv")
22 | Error_Tu_1=Error_Tu_1[, -1]
23 | Error_We_1=read.csv("Error-We-1.csv")
24 | Error_We_1=Error_We_1[, -1]
25 | Error_Th_1=read.csv("Error-Th-1.csv")
26 | Error_Th_1=Error_Th_1[, -1]
27 | Error_Fr_1=read.csv("Error-Fr-1.csv")
28 | Error_Fr_1=Error_Fr_1[, -1]
29 | Error_Sa_1=read.csv("Error-Sa-1.csv")
30 | Error_Sa_1=Error_Sa_1[, -1]
31 | Error_Su_1=read.csv("Error-Su-1.csv")
32 | Error_Su_1=Error_Su_1[, -1]
33 |
34 | Error_Mo_2=read.csv("Error-Mo-2.csv")
35 | Error_Mo_2=Error_Mo_2[, -1]
36 | Error_Tu_2=read.csv("Error-Tu-2.csv")
37 | Error_Tu_2=Error_Tu_2[, -1]
38 | Error_We_2=read.csv("Error-We-2.csv")
39 | Error_We_2=Error_We_2[, -1]
40 | Error_Th_2=read.csv("Error-Th-2.csv")
41 | Error_Th_2=Error_Th_2[, -1]
42 | Error_Fr_2=read.csv("Error-Fr-2.csv")
43 | Error_Fr_2=Error_Fr_2[, -1]
44 | Error_Sa_2=read.csv("Error-Sa-2.csv")
45 | Error_Sa_2=Error_Sa_2[, -1]
46 | Error_Su_2=read.csv("Error-Su-2.csv")
47 | Error_Su_2=Error_Su_2[, -1]
48 |
49 | Error_Mo_4=read.csv("Error-Mo-4.csv")
50 | Error_Mo_4=Error_Mo_4[, -1]
51 | Error_Tu_4=read.csv("Error-Tu-4.csv")
52 | Error_Tu_4=Error_Tu_4[, -1]
53 | Error_We_4=read.csv("Error-We-4.csv")
54 | Error_We_4=Error_We_4[, -1]

```



```

55 | Error_Th_4=read.csv("Error-Th-4.csv")
56 | Error_Th_4=Error_Th_4[, -1]
57 | Error_Fr_4=read.csv("Error-Fr-4.csv")
58 | Error_Fr_4=Error_Fr_4[, -1]
59 | Error_Sa_4=read.csv("Error-Sa-4.csv")
60 | Error_Sa_4=Error_Sa_4[, -1]
61 | Error_Su_4=read.csv("Error-Su-4.csv")
62 | Error_Su_4=Error_Su_4[, -1]
63 |
64 | Error_Mo_8=read.csv("Error-Mo-8.csv")
65 | Error_Mo_8=Error_Mo_8[, -1]
66 | Error_Tu_8=read.csv("Error-Tu-8.csv")
67 | Error_Tu_8=Error_Tu_8[, -1]
68 | Error_We_8=read.csv("Error-We-8.csv")
69 | Error_We_8=Error_We_8[, -1]
70 | Error_Th_8=read.csv("Error-Th-8.csv")
71 | Error_Th_8=Error_Th_8[, -1]
72 | Error_Fr_8=read.csv("Error-Fr-8.csv")
73 | Error_Fr_8=Error_Fr_8[, -1]
74 | Error_Sa_8=read.csv("Error-Sa-8.csv")
75 | Error_Sa_8=Error_Sa_8[, -1]
76 | Error_Su_8=read.csv("Error-Su-8.csv")
77 | Error_Su_8=Error_Su_8[, -1]
78 |
79 | Error_Mo_16=read.csv("Error-Mo-16.csv")
80 | Error_Mo_16=Error_Mo_16[, -1]
81 | Error_Tu_16=read.csv("Error-Tu-16.csv")
82 | Error_Tu_16=Error_Tu_16[, -1]
83 | Error_We_16=read.csv("Error-We-16.csv")
84 | Error_We_16=Error_We_16[, -1]
85 | Error_Th_16=read.csv("Error-Th-16.csv")
86 | Error_Th_16=Error_Th_16[, -1]
87 | Error_Fr_16=read.csv("Error-Fr-16.csv")
88 | Error_Fr_16=Error_Fr_16[, -1]
89 | Error_Sa_16=read.csv("Error-Sa-16.csv")
90 | Error_Sa_16=Error_Sa_16[, -1]
91 | Error_Su_16=read.csv("Error-Su-16.csv")
92 | Error_Su_16=Error_Su_16[, -1]
93 |
94 | Monday_k=rep(0,6)
95 | Monday_k[1]=mean(Error_Monday)
96 | Monday_k[2]=mean(Error_Mo_1)
97 | Monday_k[3]=mean(Error_Mo_2)
98 | Monday_k[4]=mean(Error_Mo_4)
99 | Monday_k[5]=mean(Error_Mo_8)
100 | 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)

```

```

147 Sunday_k[5]=mean(Error_Su_8)
148 Sunday_k[6]=mean(Error_Su_16)
149
150 #Plots of overall error depending on k
151 par(mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
152 k_s=c(0,1,2,4,8,16)
153 plot(seq(1,6,1),Monday_k, type = "l", ylim = c(250,700),
154      main = "The overall error E(d,k) of the different days",
155      col="red", lwd=2, xlab= "The values of k",
156      ylab = "Error E(d,k)", xaxt="n")
157 lines(Tuesday_k, col="orange",lwd=2)
158 lines(Wednesday_k, col="yellow",lwd=2)
159 lines(Thursday_k, col="green",lwd=2)
160 lines(Friday_k, col="blue",lwd=2)
161 lines(Saturday_k, col="brown",lwd=2)
162 lines(Sunday_k,col="black",lwd=2)
163 axis(1, at=seq(1,6,1),labels = k_s)
164 legend("topright",inset = c(-0.22,0),
165      legend=c("Monday", "Tuesday", "Wednesday",
166      "Thursday","Friday", "Saturday", "Sunday"),
167      col=c("red","orange","yellow","green","blue",
168      "brown", "black"),cex=0.8,lty=1)
169
170 Monday_k_rel=Monday_k/Monday_k[1]
171 Tuesday_k_rel=Tuesday_k/Tuesday_k[1]
172 Wednesday_k_rel=Wednesday_k/Wednesday_k[1]
173 Thursday_k_rel=Thursday_k/Thursday_k[1]
174 Friday_k_rel=Friday_k/Friday_k[1]
175 Saturday_k_rel=Saturday_k/Saturday_k[1]
176 Sunday_k_rel=Sunday_k/Sunday_k[1]
177
178 k_s=c(0,1,2,4,8,16)
179 plot(seq(1,6,1),Monday_k_rel, type = "l", ylim = c(0.4,1),
180      main = "The ratio E(d,k)/E(d,0) for the respective days",
181      col="red", lwd=2, xlab= "The values of k",
182      ylab = "Error E(d,k)/E(d)", xaxt="n")
183 lines(Tuesday_k_rel, col="orange",lwd=2)
184 lines(Wednesday_k_rel, col="yellow",lwd=2)
185 lines(Thursday_k_rel, col="green",lwd=2)
186 lines(Friday_k_rel, col="blue",lwd=2)
187 lines(Saturday_k_rel, col="brown",lwd=2)
188 lines(Sunday_k_rel,col="black",lwd=2)
189 axis(1, at=seq(1,6,1),labels = k_s)
190 legend("topright",inset = c(-0.22,0),
191      legend=c("Monday", "Tuesday", "Wednesday",
192      "Thursday","Friday", "Saturday", "Sunday"),

```

```

193     col=c("red","orange","yellow","green",
194         "blue", "brown","black"),cex=0.8,lty=1)
195
196 par(mar=c(5.1, 4.1, 4.1, 6.1), xpd=TRUE)
197 Times=c("12am", "12.30am", "1am","1.30am","2am","2.30am",
198         "3am","3.30am", "4am","4.30am", "5am","5.30am",
199         "6am","6.30am", "7am","7.30am", "8am","8.30am",
200         "9am","9.30am", "10am","10.30am", "11am","11.30am",
201         "12pm", "12.30pm", "1pm","1.30pm","2pm","2.30pm",
202         "3pm","3.30pm", "4pm","4.30pm", "5pm","5.30pm",
203         "6pm","6.30pm", "7pm","7.30pm", "8pm","8.30pm",
204         "9pm","9.30pm", "10pm","10.30pm", "11pm","11.30pm")
205 plot(seq(1,48,1), Error_Monday, col="red",
206     main = "The error E(t,d,k) for the day Monday
207         for different values of k",
208     xlab = "The time intervals",
209     ylab = "The Error E(t,d,k)", xaxt="n", type = "l",
210     lwd=2, ylim = c(0,1500))
211 lines(Error_Mo_1, col="orange", lwd=2)
212 lines(Error_Mo_2, col="yellow", lwd=2)
213 lines(Error_Mo_4, col="green", lwd=2)
214 lines(Error_Mo_8, col="blue", lwd=2)
215 lines(Error_Mo_16, col="brown", lwd=2)
216 axis(1,at = seq(1,48,1),label=Times)
217 legend("topright",inset = c(-0.16,0),
218     legend=c("k=0", "k=1", "k=2","k=4","k=8", "k=16"),
219     col=c("red","orange","yellow","green","blue",
220         "brown", "black"),cex=0.8,lty=1)
221
222 plot(seq(1,48,1), Error_Wednesday, col="red",
223     main = "The error E(t,d,k) for the day Wednesday
224         for different values of k",
225     xlab = "The time intervals",
226     ylab = "The Error E(t,d,k)", xaxt="n",
227     type = "l", lwd=2, ylim = c(0,1500))
228 lines(Error_We_1, col="orange", lwd=2)
229 lines(Error_We_2, col="yellow", lwd=2)
230 lines(Error_We_4, col="green", lwd=2)
231 lines(Error_We_8, col="blue", lwd=2)
232 lines(Error_We_16, col="brown", lwd=2)
233 axis(1,at = seq(1,48,1),label=Times)
234 legend("topright",inset = c(-0.16,0),
235     legend=c("k=0", "k=1", "k=2","k=4","k=8", "k=16"),
236     col=c("red","orange","yellow","green","blue","brown"),
237     cex=0.8,lty=1)

```