GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

TRAVEL TIME PREDICTION IN PUBLIC TRANSPORTATION

Betül BOYLU

Supervisor
Assist. Prof. Dr. Ali BOYACI

M.Sc. THESIS
DEPARTMENT OF COMPUTER ENGINEERING
ISTANBUL - 2021

ACCEPTANCE AND APPROVAL PAGE

On 24/02/2021, **Betül BOYLU** successfully defended the thesis entitled "**Travel Time Prediction In Public Transportation**" which she prepared after fulfilling the requirements specified in the associated legislations, before the jury members whose signatures are listed below. This thesis is accepted as a **Master's Thesis** by Istanbul Commerce University, Graduate School of Natural and Applied Sciences, **Computer Engineering Department**.

Supervisor	**Assist. Prof. Dr. Ali BOYACI** Istanbul Commerce University	.....
Jury Member	**Assist. Prof. Dr. M. Alper ÖZPINAR** Istanbul Commerce University	.....
Jury Member	**Assist. Prof. Dr. Özgür Can TURNA** Istanbul University - Cerrahpaşa	.....

Approved Date: 15/03/2021

Istanbul Commerce University, Graduate School of Natural and Applied Sciences, accordance with the 1st article of the Board of Directors Decision dated 15.03.2021 and numbered 2021/308, "Betül BOYLU" (TC:37072726194) who has determined to fulfill the course load and thesis obligation was unanimously decided to graduated.

Prof. Dr. Necip ŞİMŞEK
Head of Graduate School of Natural and Applied Science

DECLERATION OF ACADEMIC AND ETHIC INTEGRITY

I hereby declare that,

- I have obtained the all information and documents within the academic and ethical rules,

- I have presented all visual and written information and results in accordance with academic ethics,

- I refer to the relevant studies in case the studies of others are used,

- neither whole nor any part of this thesis is not presented in this university or any other university, previously.

15.03.2021

Betül BOYLU

TABLE OF CONTENTS

ABSTRACT

M.Sc. Thesis

TRAVEL TIME PREDICTION IN PUBLIC TRANSPORTATION

Betül BOYLU

Istanbul Commerce University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Assist. Prof. Dr. Ali BOYACI
2021, 56 pages

Today, travel time prediction is essential for passengers who can easily access information and want to be able to plan their journeys as well as their daily activities. Travel time varies due to some unpredictable external factors especially in big cities. Therefore, this paper proposes a powerful but simple Machine Learning model by using data collected by GPS devices. The model uses a Multiple Linear Regression algorithm that learns from historic data and predicts future values for each bus stop interval by considering external factors such as; weather conditions, peak hours, busy week days and busy days of year. A validation model was developed to measure the accuracy of the prediction model. Then the validation model was compared to average of historic data and real data. Results show that the prediction model outperforms the average model and calculates closest travel times to the real data.

Keywords: Machine learning, multiple linear regression, travel time prediction.

ÖZET

Yüksek Lisans Tezi

TOPLU ULAŞIM ARAÇLARINDA ULAŞIM SÜRESİNİN TAHMİNİ

Betül BOYLU

İstanbul Ticaret Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Ana Bilim Dalı

Danışman: Dr. Öğr. Üyesi Ali BOYACI
2021, 56 sayfa

Günümüzde toplu ulaşımda, ulaşım süresinin tahmini, bilgiye kolayca erişebilen ve günlük aktivitelerini planladıkları gibi yolculuklarını da planlamak isteyen yolcular için oldukça önemlidir. Büyük şehirlerde ulaşım süresi bazı öngörülemeyen dış faktörler nedeniyle çeşitlilik göstermektedir. Bu nedenle bu çalışma, GPS cihazları ile toplanan veriyi kullanarak, güçlü ancak sade bir Makine Öğrenmesi tekniği sunmaktadır. Teknik, geçmiş veriden öğrenerek, gelecek verisini hava durumu, yoğun saatler, haftanın yoğun günleri ve yıllın yoğun günleri gibi dış etkenleri göz önünde bulundurarak tahmin eden Çoklu Düzlemsel Regresyon algoritmasını kullanmaktadır. Tekniği doğrulamak amacı ile bir doğrulama modeli oluşturulmuştur. Doğrulama modeli geçmiş verinin ortalaması ve gerçek veri ile kıyaslanarak modelin doğruluğu ölçülmüştür. Sonuçlar tahmin tekniğinin ortalama modele göre daha iyi performans gösterdiğini ve gerçek veriye en yakın tahmini yaptığını göstermiştir.

Anahtar Kelimeler: Çoklu doğrusal regresyon, makine öğrenmesi, ulaşım süresi tahmini.

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor Assist. Prof. Dr. Ali BOYACI for his help with his knowledge during the development of the research.

I would also like to take this opportunity to thank IETT for providing the data I needed to develop the work.

Last but never least, I would like to thank my family and friends for their never-ending support, prayers, and patience.

Betül BOYLU
İSTANBUL, 2021

LIST OF FIGURES

LIST OF TABLES

SYMBOLS AND ABBREVIATION

AI	Artificial Intelligence
ANN	Artificial Neural Network
APC	Automated Passenger Counters
AVL	AVL
BISN	Bayesian Inference of High-Dimensional Sparse Networks framework
BP	Back Propagation
BRT	Bus Rapid Transit
BTMS	Bluetooth Traffic Monitoring System
CORSIM	Comprehensive Microscopic Traffic Simulation
DIAC	Dedicated Inquiry Access Code
DLMs	Dynamic Linear Models
FCN	Fully Connected Neural Network
GA-SVM	Genetic Algorithm-Support Vector Machine
GBRT	Gradient Boosting Regression Tree
GMM	Gaussian Mixture Model
GPS	Global Positioning System
HA	Historical Average
IETT	Istanbul Elektrik Tramvay ve Tünel İşletmeleri Genel Müdürlüğü
ITS	Intelligent Transportation Systems
K-NN	K-nearest Neighbor
LR	Linear Regression
LS	Least Squares
LSTM	Long Short-Term Memory
MAC	Media Access Control
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MARE	Mean Absolute Relative Error
ML	Machine Learning
MTT	Measured Travel Time
OBD	Onboard Devices
PECM	Partial Empirical Covariance Matrix
PIS	Passenger Information System
PT	Public Transportation
RBF	Radial Basis Function
RF	Random Forest
RMSE	Root Mean Square Error
RVM	Relevance Vector Machine
SSNN	State Space Neural Network
SVR	Support Vector Regression
TFTS	Tensor Flow Time Series
TSP	Transit Signal Priority

1. INTRODUCTION

Efficient time management has become an indispensable requirement in today's busy world. Every day, new technologies are developed to provide time efficiency for individuals. People demand easily accessible, affordable and reliable public transportation. Therefore travel time prediction has an utmost importance by providing time saving and personal planning in public transportation as a part of comprehensive passenger information systems (PIS).

Passenger information systems aim to inform passengers about their journey via several platforms hence increase customer satisfaction. These systems reduce call center work load for transportation operators by making the information available and easily accessible. Passengers are able to learn about line, travel time, delays, closest bus on map, fees, cancelled journeys, incidents and local events via these services. In addition, journey planner tools allow passengers to find best option in terms of time, fee and accessibility. PIS provide services through widely used platforms such as smart bus stops with displays, screens onboard with audio annunciation, mobile phones and websites with map applications. Recently most public transportation operators use these services to maintain positive customer experience especially in big cities.

Istanbul is a commercial and historical center of Turkey with thousands of years of history. It is a transcontinental city connecting Europe to Asia via 3 bridges; Bosphorus Bridge, Fatih Sultan Mehmet Bridge and Yavuz Sultan Selim Bridge and 2 underground tunnels; Avrasya Tunnel and Marmaray. Everyday thousands of people use these connections to cross other side in addition to ferries. The city has a population of 15 million and hosts an average of 9 million visitors every year. Public transportation (PT) in this colorful and vibrant city is carried out by various modes such as bus, metrobus, subway, tram, ferry, minibus and taxi. Bus transportation constitutes 30% of PT. Istanbul Elektrik Tramvay ve Tünel İşletmeleri Genel Müdürlüğü (IETT) is the transportation operator in Istanbul since 1871 that started

business with horse drawn trams. Today IETT operates bus and metrobus transportation and has about 7000 buses, 6000 drivers and 5 million journeys daily. IETT installed screens onboard in every bus and 933 smart bus stops in the city to display current location and expected arrival time of the buses to prevent time loss in PT. Additionally, a mobile app and a website helps individuals to plan their journey and view expected travel times of the buses.

Predicting travel time is a challenging process for Istanbul since it directly depends on number of vehicles in traffic and there are various factors determining the number of vehicles. First of all, during school periods more vehicles join traffic because while schools are located in central points of the city, the settlement is towards the outside of the city. Therefore, students need to be transported to the city center by shuttle buses and public transportation. Second, every day of week shows different characteristics as a result of local bazaars, weekend events and workdays. For example, when local bazaars set up, pedestrian traffic increases dramatically. Next, in mornings and evenings there is too much traffic congestion as people go to work and get back home, these times called peak hours and during peak hours number of vehicles in traffic is maximum. Last, weather conditions have a huge impact on traffic flow. When it is rainy, less people go out thereby less traffic occurs in some parts of the city. When building a prediction model for Istanbul, these 4 major factors are needed to be considered. Moreover, the effect of the factors might change according to parts of the city. This inconsistency causes obstacles in the prediction and indicates need of a stronger model.

The subsequent chapter 2 addresses different types of work done in this research area and gives detailed explanation. In chapter 3, Preparation process with Parameter Selection, Line Selection, Field Observation and Data operations (collection, cleaning and processing) are given. Then, Travel Time Prediction model and Validation step are described in detail. In chapter 4 Results are shown and validity of the model is explained. Lastly in chapter 6 the model is summarized in Conclusion part with a possible future work.

2. LITERATURE REVIEW

There are 3 main prediction model types as seen on Figure 2.1; historical data based models, statistical models and Machine Learning models. Historical data based models calculates current travel time based on average of previous travel times for the same time span. Time Series and Kalman Filter are types of statistical models. Time series models suggest that historic data will be same in future and suggest to convert data into multidimensional collection in order to consider all the factors (Zhu et al., 2010). Kalman Filter models are used when there is ambiguous data about the system. It is a mathematical method to use data observed over time and calculates values that are closer to the true values of the data (Yang, 2005). This type of models predicts current and future states of a system (Choudhary et al., 2016). Machine Learning models can learn from existing data and predict the future data. Multiple Linear Regression and Artificial Neural Networks are some of the Machine Learning algorithms. Regression models calculates travel time with a Linear Function formed by some independent variables.

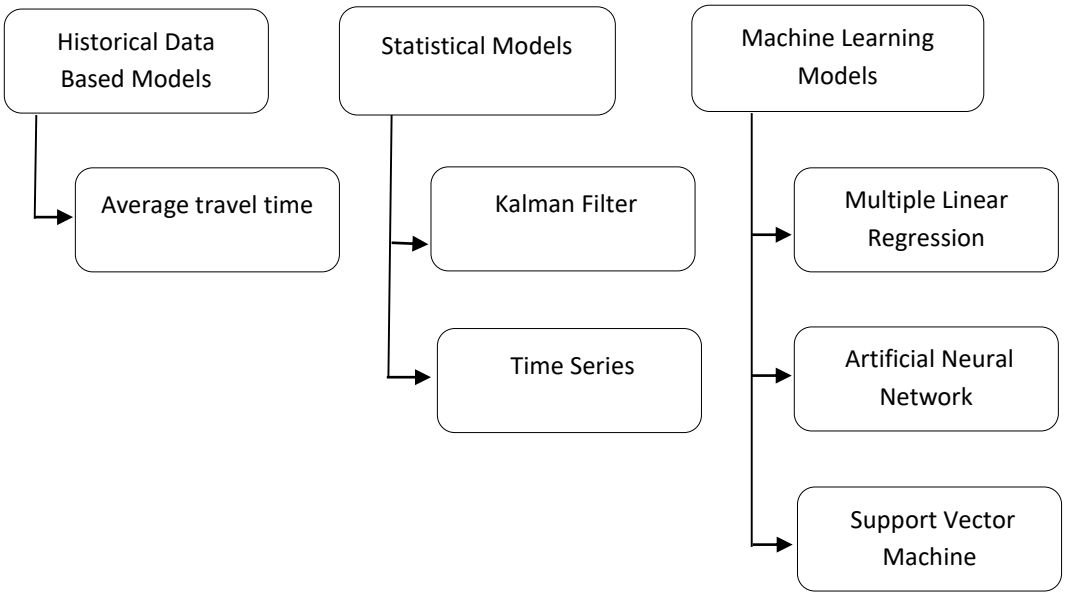

Figure 2.1. Types of prediction models

Some studies work on multiple prediction models to find a better solution. Lin et al. (2019), studied 3 models to find the best solution for travel time prediction in Chiayi City, Taiwan. They applied Gradient Boosting Regression Tree, K-Nearest Neighbor and Linear Regression. They believed that previous works gained limited success for predicting time on urban roadways because there were not enough vehicle detectors on these roads. Consequently, they used 3 types of units as detectors. Traffic data was collected by vehicle detectors, speed data was obtained from Global Positioning System (GPS) and Onboard Devices (OBD), and cellular-based travel data was obtained from telecom companies. They collected data between 1 October 2018 and 4 March 2019 and grouped data into two categories to use for training and validation. They developed a weekday model and a weekend model by using three methods. The results showed that all three models performed well for prediction on urban roadways. Cellular-based vehicle probe data provided the best source of data because with this type there was no missing records when compared to Vehicle Detector data and eBus data. Overall study showed 12% Mean Absolute Percentage Error (MAPE) which was accepted good forecasting.

Kwak and Geroliminis (2020), proposed dynamic linear models (DLMs) for speed and time prediction, because DLMs assume parameters are changing in time. They inferred that the model derived temporal velocity by means of dynamic characteristics and described a linear relationship between velocity of specific time and velocity of future time. In addition to travel time prediction, velocity prediction was also conducted. The regular velocity was collected by 80+ loop detectors for every 30 seconds. By using velocity at each sensor with 5 minutes intervals, continuous velocity was generated. They gathered traffic data for 2012 and 2015, used 70% of it for model training, 15% for validation and remaining 15% for all experiments. They tested the models by using freeway data in California and compared results to four different algorithms; Instantaneous Travel Time Forecaster, K-Nearest Neighbor (KNN), Support Vector Regression (SVR) and the Artificial Neural Network (ANN). The proposed model, used both historic and real data, showed better performance for short term prediction (0-15 minutes) under any traffic situation by providing 56% prediction accuracy improvement with 0-minutes horizon.

Some studies focused on specific types of travel time. For example, Yang (2005), developed a Kalman Filter model for arterial travel time prediction during a graduation ceremony assuming that an event can cause sudden congestion. He believed that by providing right traffic information, drivers can be directed to other routes thus congestion might be eliminated. He used GPS data from test vehicles for a period of 30-45 minutes considering the length of the ceremony. He collected both total and section travel times for the ceremony. By using Kalman Filter, he formulated a recursive procedure that used result of current step to estimate result of next step. He calculated total travel time as well as link travel time which was from one intersection to another. For overall testing Mean Absolute Relative Error (MARE) was 17.61%. The results for three segments showed 24.98%, 25.17% and 21.25% MARE rate. Then he tested the model for two concerts performed on 25 April and 22 May 2004 by using 3 minute and 5 minute time intervals. He found that 3 minute interval performed better with 21.20% MARE. Later he used data interpolation method to increase the number of intervals and reduce the error. With interpolation, he reduced the error to 4.40%.

Likewise, Hapsari et al. (2018), focused on touristic travelling and developed a regression model to predict visiting time of destinations in Indonesia to help tourists. They agreed that although Google Map provided accurate data for travel time of the touristic attractions, there were still many locations without information about visiting time. They found out that only 35.78% of the touristic places had visiting time information. Therefore, they used Multiple Linear Regression method to predict the visiting time for each location. Six parameters were considered in the model for prediction; access, government, rating, number of reviews, number of pictures, and other information. The data was collected from Google. They compared their model with four other popular prediction models; K-Nearest Neighbors, Decision Tree, Support Vector Regression, and Multi-Layer Perceptron and accepted that the one with lowest Root Mean Square Error (RMSE) and highest coefficient correlation was the best model. They obtained the least error ratio from Linear Regression model with 48% (RMSE). They predicted visiting time for 402 destinations in the city. This work found total amount of time to reach the destination.

Some of the previous studies focused on different vehicle types while predicting travel times. For example Tan et al. (2008), targeted Bus Rapid Transit (BRT) vehicles and worked on transit signal priority (TSP) concept where transit vehicles move through signal controlled intersections and delay time in intersections were aimed to be reduced. They collected both historical and real-time traffic data from GPS devices for their work. First they developed a historic model to obtain an average travel time, the historic model used constant average speed for an entire section. Although the model gave well result, because the convergence was slow, they developed an adaptive model to compensate convergence by using real time data. The adaptive model used adaptive average speed formulated by a Least Squares (LS) algorithm. Then they put two models in a weighted average. Assuming that there was a relationship between section length and travel time, Linear Regression algorithm was used to predict the travel time. The algorithm of the combination model worked well when congestion is less and traffic flow is normal. They believed that including length of queue and queue discharge rate might improve the work for highly congested traffic.

Rice and van Zwet (2001), built a model to estimate travel time between two points of a freeway. They presumed that there is a linear relationship between current travel time and future travel time and developed a Linear Regression Model. They collected data via 116 single loop detectors detected with 5 minute intervals from California freeways between 16 June and 8 September 2000. Bu using collected velocity data, they predicted travel times and noticed that there was a huge variance during morning and afternoon congestion. They calculated RMSE for the current status predictor and historic average. The model had better RMSE than both. After comparing their model to Principal Components (PC) and Nearest Neighbor Predictor, they experienced that the model showed better performance. The model gives an RMSE below 10 minutes for an average of one hour travel time. They also developed an online tool for users to choose a start and destination and see the predicted travel time and the best route.

Pan et al. (2012), developed a self-learning algorithm based on historic data that was collected by GPS sensors. The historic speed data was classified according to seasons, holidays, and peak hours and recorded into a database. Location of the bus was also recorded periodically. Then a Back Propagation (BP) neural network was used to train the data and to correct the speed based on the average historic travel time. A BP Network is a system with one input layer, one output layer and some hidden layers that is used to train the network. When number of layers increases, accuracy of results increases also. However, this makes the network more complicated and training time longer. After extracting distance between stops, they used speed of the bus as an input for BP neural network and speed at next moment as an output and they experienced that after training huge data, the network predicted the speed. The algorithm had less overall prediction error. However, when congestion is heavier, the prediction error grows accordingly.

Yu et al. (2013), developed a model for Beijing City in China and used real world historic GPS data to develop the model based on cluster analysis and polynomial fitting. They assumed that for an accurate prediction, road condition, bus velocity, traffic flow, density of crowd and traffic lights should be considered. The model used the GPS data monitored every 15-20 seconds. The data was classified by using average distance method and two nearest classes were merged together until there was only one class. The method generated a historic traffic pattern based on bus line, period time and day type. They assumed that traffic flow was similar for a week thus velocity of the bus was consistent in the same week. The model employed a hierarchical cluster analysis using Euclidean distance to maximize the effects of similar patterns. The model was considered as a simple prediction model without need of extensive computation. MAPE rate on all lines was 22.57% for all distances, 29.47% for distances less than 700m and 16.29% for distances more than or equal to 700m.

Achar et al. (2020), developed a model that learnt the spatial patterns of traffic. They rewrote the predictive model in a linear state space form and applied Kalman Filter. They split the line into sections and used running time of the bus, dwell time and

unexpected stoppages to build the model. They compared their results to Historical Average (HA), Random Forest (RF) and Space Discretization (SD). HA provides the average of historic data where RF is another ML method that learns from existing data and predict the future data. SD on the other hand, assumes a relationship between consecutive sections and predicts travel time for a section based on travel time of the previous section. They compared results for one section, over sections and over trips. For one section, their model showed 16% better performance than HA, 4% better than RF and similar performance to SD. For over sections, the model showed 16%, 14% and 7% better performance respectively for most of the sections. For trips, the model performed between 14% to 37% for SD, 13% to 27% for RF and around 26% for HA.

He et al. (2019), indicated the need for a prediction system that covers multiple bus trips rather than a single one. They considered multiple journeys of a passenger as well as waiting time of the passenger at transfer points to predict the travel time. They predicted riding and waiting time of a journey based on different datasets including historical data and combined the results. They used Long Short-Term Memory (LSTM) for riding time prediction of each segment by using some external factors like length of route, number of bus stops, number of intersections, directions and etc. They used historical average method for waiting time prediction. Mean Absolute Error (MAE) and MAPE of the model were compared to six other algorithms; HA, KNN, Tensor Flow Time Series (TFTS), Fully Connected Neural Network (FCN), Linear Regression (LR) and Support Vector Regression (SVR). Their model showed better results than the baselines with 55.2% improvement on MAE.

Reddy et al. (2016), focused on high variance problem at their study. In the first stage of the study, they used an existing model based approach and Kalman Filter for the high variance. In the second stage, they used Support Vector Machine (SVM) assuming that SVM is better than ANNs and other techniques when variability is high. SVM is used to transfer the two sets of nonlinearly separable data into a higher dimension space where they can be linearly separated. Location data of the bus was obtained via Global Positioning System (GPS) units and stored into an SQL database.

Haversine formula was used to calculate the distance between two stops. Then the travel time variation was extracted for a period of one week. Their study required large data sets. An advanced nonlinear model could be more accurate for the system and they believed that SVM can be better if traffic factor, driver's age and vision and vehicle age and characteristics were considered.

Another Machine Learning Algorithm is Extremely Randomize Trees. An Extremely Randomize Tree is a type of randomize trees where the tree is built by using whole training set. Input variables and splitting values are randomly selected. Splitting the nodes random reduces the prediction variance. Garcia and Retamar (2016), used this method in their work. They concerned about economic losses caused by long travel times because Philippine economy highly depends on land transportation. They developed a prediction system for Metro Manila where there is no scheduled bus operations. And bus schedules depend on traffic flow, time, vehicle availability and number of passengers. They extracted 2015 GPS data and processed data to add hour, minute, second, year, month, day and day of week and calculated travel time for 10 selected drop off points by subtracting arrival time of one point from arrival time of next point. They generated number of trees and developed a regression prediction by averaging the total number of the trees. For testing step, they used 2 sets of data. They compared predicted travel time to measured travel time and received R^2 between 0.9 and 1. The first test set showed 0.97% R^2 and second test set showed 0.94 R^2. The model was tested on a single route.

Li and Bai (2016), focused on freight vehicles to help freight transportation companies with better planning. They believed that there was a big amount of data collected by transportation authorities but not shared with freight companies thus not utilized fully. Such data could help these companies to make better plans and task scheduling. Therefore they built a Gradient Boosting Regression Tree (GBRT) for travel time prediction on 3 routes by using basic, historic data and mean speed sequence. The trajectory data between 16 March and 30 April 2014 was used. For the missing trips in historic data some rules were applied at preprocessing step. Mean travel time for same interval from all other vehicles was used for a missing trip. And mean travel time of previous interval was used in case there was no trip from all

vehicles. The parameters used in the model were departure time, day of week, month, day in month, day in year, weekday, workday and public holiday. To reduce over fitting, they used Bayesian optimization. They performed a pre-start and a post-start prediction. The last 200 trips were used as test data set. They obtained 80% and above prediction accuracy for both pre-start and post-start prediction. By adding more speed sequence, the performance was improved by 2%. This work shows good performance when there is a speed data. For systems with huge amount of data, and routes closer to the city center, the results might show difference.

Chien and Ding (2002), believed that the models developed based on historic data cannot consider dynamic factors such as dwell time and delay at intersections and therefore an advanced model was needed. They developed two Artificial Neural Network (ANN) models to predict the bus travel time. These two models were trained with link-based and stop-based data. A designed algorithm integrated two ANNs. For link based ANN, they assumed that there were links between two stops and prediction of time was calculated by sum of travel times on each links. Stop based ANN was developed by using stop based data such as volumes, speeds and delays between two stops. They used comprehensive microscopic traffic simulation (CORSIM) program to simulate the model. CORSIM is a traffic simulation program that can simulate even lanes and flow conditions. After measuring performance of two ANNs they decided that stop based ANN was more suitable for the two stops with more intersections in between, while link based was preferred by the ones with less intersections.

Another study that used ANN was developed by Turchenko and Demchuk (2006), by considering date, day of week, departure time of the bus, holidays, weather, events and accidents as inputs for the network. They also analyzed quality of road, location of road and type of vehicle factors. They categorized weather under 5 types; very nice (sunny), nice (nice), satisfactory (heavy cloudy), bad (snow, fog) and very bad (ice, heavy rain, snow, fog). Departure time was also categorized as; morning, late morning, noon, afternoon, evening, late evening and night. Date is divided into 5 categories; work day, weekend, holiday, unexpected event and holiday time. Quality

of road was classified as very nice, nice, satisfactory, bad and very bad. Traffic load was divided as very low, low, average, high and very high. Two types for road location were city and out of city and finally type of car was categorized as car, racing car, minibus and truck. In the model development, a sigmoid function was used for hidden layer neurons, a linear function was used for output neuron and a back propagation error algorithm was used for training. The model had seven input neurons and an output neuron. By using C programming language, they developed a software to test the results. The model showed enough prediction accuracy by 9 - 3% error rate.

One more example that used Neural Networks based on single segment for Netherlands was developed by Liu et al. (2009). They focused on urban streets in spite of other works focusing on highways. They categorized prediction into direct and indirect approaches. Indirect prediction started with prediction of factors like volume, speed etc. while direct prediction works on previous data. They stated that main difference of two concepts were the inputs. However, travel time is independent on historic travel data although it is dependent on factors like volume and speed. They picked an indirect approach with volume and signal timing inputs and data driven approach with neural networks. They developed a State Space Neural Network (SSNN) by using incoming volume and green light time. The inputs and a context layer were used to calculate a hidden layer vector. And an output layer produced output by using hidden layer outputs. They used 82 days of data saved in 2004, data for 62 days were used as training set and 20 days were used for validation. The test was conducted on a road that connected two motorways. The model performed better when the prediction time was less than 15 minutes.

Meng et al. (2017), practiced automatic vehicle location (AVL) data saved every 30 seconds on arterial links in Edmonton, Canada. The model consisted of three elements; free flow, dwell time at intersections and congestion time. They suggested that travel time was allocated from a certain point of one link to certain point of another link. Therefore, they calculated travel time based on three types of links. First, when starting and ending points are on the same link, second, when the

reported positions were on different links and third, when there is at least one full link between two points. The model predicted three types of travel time; free flow, congestion time and stopping time. MAE and MAPE were calculated for each link types after the model was tested. The model showed 5.08 MAE for all links, 4.23 MAE for bus stops and 6.65 MAE for intersections. It also showed 26.83% MAPE for all links, 12.04% MAPE for bus stops and 10.35% MAPE for intersections. The results presented that prediction did not work well when links were divided by intersections. The results were better if links were divided by bus stops.

Lin and Zeng (1999), also used GPS data in San Francisco and stored it into a database every 45 seconds. In order to calculate the distance between two points based on the location data, they did not use Euclidean distance, they designed a time-distance graph instead. They developed four different algorithms each using previous travel data. The first one was based on only bus location so calculating arrival time at destination point was not needed. Second one was based on bus location and bus schedule table. Third algorithm used bus location, bus schedule table and delay where it was assumed that drivers were aware of the delay so they would adjust their speed accordingly. Fourth algorithm was based on bus location, bus schedule table, delay and time check point. They examined the performance of the algorithms with overall precision to measure the average deviation between predicted travel time and real time, robustness to decide any prediction rate far off the real one and stability to see if any algorithm produces a prediction time that was instable and found out that the algorithm that concerned time-check data had the best performance.

Cheng et al. (2010), developed a model by using historic data produced by the Automated Passenger Counters (APC). These devices record the time that a bus arrives at a stop, bus stop number, time that the bus leaves the stop, date, direction, route number and route name. The model had two main components, prediction of the travel time between two stops and searching for the next bus. To predict the travel time, clustering method and K-nearest algorithms were used. K-nearest algorithm calculated distance by time of day and day of week parameters, found

historic records with shortest distance and determined the travel time class where most of historic records appeared. To measure the performance of the model, they calculated variation between predicted travel time and real travel time by using APC data of Harbin City in China. Results showed that there was a maximum 64 seconds difference between the actual arrival time and predicted one. They also claimed that the clustering method can be applied at irregular schedules to increase the reliability.

Zhu et al. (2011), believed that number of the travel time prediction models with dwell times and intersection delay times were limited. That was why they developed a prediction algorithm based on travel times between two stops and these factors. They extracted travel data of Beijing City by Global Positioning Systems that obtain latitude and longitude every 20 seconds. By Haversine Formula, distance for each 20 seconds was calculated. The model was divided into two sections; first the current location of the bus was its next location; second the bus was two or more bus stops from the current location. For the first section, it was assumed that traffic jam condition is the same and bus speed did not change. So distance was calculated with constant speed. For second section, total travel time was divided into three parts; running time of the bus, dwell time at the bus stops and intersection delay time. Each of these times was calculated separately and total travel time was estimated accordingly. When the current location of the bus was the objective location, the maximum error was 45 seconds. For the second section, when the bus was two or more stops away from the stop, the prediction results were very close to the measured results.

Yildirimoglu and Geroliminis (2012), developed an estimation model that used both historic and real time data in addition to shockwave analysis and bottleneck identification. They used identified bottleneck locations to restore the traffic events. And historic data was used for this identification. They used clustering method in order to get days with similar traffic patterns. They reduced dimensions of the dataset and used Gaussian Mixture Model (GMM) to obtain results and created stochastic congestion maps for each cluster. To predict the speed profile of the bottlenecks, they used average speed. They found that holidays and weekends have

no significant level of congestion. Weekdays except Fridays have significant level of congestion while Fridays have the highest level of congestion. In this work, if traffic conditions were less congested than expected, then some bottleneck point was occurred, therefore this model experienced some time lag.

Li et al. (2017), introduced a mixed model for arrival time prediction. They believed that traffic incidents caused similar time delays. Instead of using regular travel times for training, they used delay fluctuation. The model had three stages. First one was pattern training where K-Nearest Neighbor and K-means methods were used to mine the traffic delay data based on traffic incidents. Second one was a single step prediction of travel time by Kalman Filter. And third one was the combination of single step prediction with Markov transfer model. They believed that because of high buildings GPS data was not correct. That was why the data was preprocessed by noise alteration, backward error adjustment and error modification to be used for KNN training. KNN finds the most similar records to the current one and combines their future values to estimate the next value. The nearest neighbor value which was demonstrated as K, was very important for KNN and they set it between 5 and 15. K-means is a common approach to partition values into clusters where the values belong to a cluster with nearest mean. They stated that there were three time patterns; time without delay, light delay and strong delay. Therefore, arrival times were clustered into three groups with K=3. Single-step prediction was accomplished by using real time traffic data and historical fluctuation. Kalman Filter was applied for real time prediction, it was adjusted with every new measurement. Euclidian distance was used to align similarity and by using weighted prediction, KNN correction for single step was completed. For multi-step prediction, single-step prediction was merged with Markov chains. Markov chain defends that the probability of next state relies on the current state rather than depending on sequence of previous events. Multi-step prediction was completed in four steps. First, max Markov transition probability of arrival time found and next arrival time was calculated. Second, bus travel time was calculated by using historical data. Third, single-step prediction was combined with dynamically adjusted model. And fourth, step one, two and three were repeated until the multi-step prediction

provides the given steps. Two coefficients used in the model were historical pattern and current traffic flow. The results were tested for PT buses in Hefei, China. Actual travel time was compared to four algorithms with MAPE values; single-step, Kalman Filter, KNN and short-term traffic flow. Both Kalman Filter and short-term prediction were performed well during a traffic jam but bad during non-peak hours. In case of an accident, single step and short-term performed better than KNN. Multi-step prediction outperformed other three algorithms. MAPE of multi-step prediction was between 10% and 25%. When the number of bus stops increased MAPE rate of multi-step prediction raised as well.

Deng et al. (2013), proposed a Bayesian Network model to predict the time travel based on road traffic state. They believed that predicting travel time and informing passenger about it reduces their anxiety when they wait for a bus. They used Markov transfer matrix to get the traffic state. Bayesian Network is a graph consisting nodes and directed edges where conditional probability between child and parent nodes decides the strength of associations between the nodes. In this model, road average speed was accepted as parent node and predicted bus travel time is accepted as child node. They obtained bus travel time for every 10 minutes interval and in case there were several travel times in 10 minutes, average of them was taken. They did a historic data training for 9 days and calculated average speed. The model had 0.196 MAPE, 39.09 MAE and 49.14 RMSE. They indicated that, the reasons behind the high error rate might be; the stochastic variance of bus travel time, the effect of intersections and traffic lights, dwell time at the bus stops and the less number of variables to be considered in the model.

Zhou et al. (2014), used a rare method to predict the travel time in Singapore. Their system relied on passenger's mobile phone and thus it was not dependent on bus operating companies. Instead of GPS based data collection, they used energy efficient sensing resources. They recorded some cell tower IDs and whenever a user's mobile phone connected to a tower, the location of that user was known. To determine if the connected user was on bus or not, they used audio detection. In Singapore travel cards are used in the buses to pay the fee. Each bus has a card

reader inside and readers respond by a beep sound to travel cards. In this system, mobile phones only with Android operating system, detected the beep sound. This way passengers on other vehicles like cars and taxis were separated. However, some similar beep sounds may occur in other environments such as cash cards and employee's cards. Rapid Train System also can cause such problems. To distinguish the rapid trains from the buses, they assumed that trains have more consistent speed than buses as they are not affected by any traffic jam. In addition to that, buses have frequent acceleration and deceleration. So they set an accelerometer threshold to distinguish the buses. A backend server calculated arrival time based on historic data and route state. After a 7-week testing, this flexible model provided an accurate travel time prediction. However, the model always required travel cards as fee collection system and a backend server in order to work in other cities. Moreover, the system accuracy was highly effected by the number of passengers who participate. They might need to promote the system so that at least one passenger in the bus was willing to report the bus status.

Yu et al. (2017), proposed a relevance vector machine (RVM) model to estimate bus headway. A RVM algorithm tries to find a relationship between an input and output value. It was maintained based on SVM but builds a method by using Bayesian framework. They aimed to make a probabilistic estimation on bus headway. The RVM was used to make a single point prediction for a bus headway. Instead of GPS data, they used smart card data based on stop-level passenger movement from Beijing. Distance based fare buses data was used because there were a lot of data on distance based journeys and this data was more detailed. Data between 1 July 2012 and 1 November 2012 was extracted and preprocessed to calculate arrival time and average speed between two bus stops. They picked several factors effecting the traffic flow such as; dwell time, traffic condition, boarding and alighting time. They extracted these factors by using travel card data. Next, by using the historic data they utilized RVM to predict the bus headway for next stop. RMSE and MAPE were calculated for results and compared to SVM, Genetic Algorithm-Support Vector Machine (GA-SVM), Kalman Filter, KNN and ANN models for one-step ahead bus headway and two-step ahead bus headway estimations. RVM performed better

results for both steps by 14.63% - 15.39% MAPE for one-step prediction and 19.80% - 23.76% MAPE for two-steps prediction.

O'Sullivan et al. (2016), drew attention to uncertainty with bus travel time predictions and instead of generating a prediction algorithm, they took an existing algorithm and treated it as a black box. In order to improve the existing algorithm, they applied quantile regression to set up bounds on error rate. They used real travel time data between March and May 2014, collected from two routes in Boston, one being a popular route and other was a simple one. They calculated mean, median, skewness and kurtosis values in order to obtain an evidence of heteroscedasticity in the data and found a characteristic uncertainty. In order to solve this problem, they decided to design prediction intervals with upper and lower bounds. They believed that providing upper and lower bounds to passengers rather than giving an exact travel time would be more proper because of the uncertainty in prediction. After developing a Gaussian Process Quantile Regression algorithm, results showed that the algorithm changed uncertainty levels and contributed expected coverage on unseen data.

Vinagre Díaz et al. (2016), brought a different perspective to travel time prediction. They used Bluetooth technology in order to collect travel data to see if Bluetooth traffic monitoring system (BTMS) was able to make proper estimations. A Bluetooth is a low cost short range device that requests media access control (MAC) address to connect other devices. It is a part of Personal Area Network. They declared that BTMS could detect a vehicle this way and anonymously store the timespan and the MAC address. When a second detector captures the same MAC address, BTMS can calculate travel time between two detectors. However, there were some problems with the system. When traffic flow is low because of traffic congestion or intersections, signals from bicycles and pedestrians could cause confusion. Moreover, when multiple mobile phones correspond to same vehicle, uncertainty increases. They filtered devices by Dedicated Inquiry Access Code (DIAC) to overcome these problems. They believed that using this technology causes a spatial error in the measurement effecting distance, velocity and travel time. They

calculated min travel time value by dividing the distance between two detectors by velocity. Then a max travel time was calculated in order to eliminate any outliers. Later, travel time was predicted based on collected MAC addresses. They installed Bluetooth devices on a 6km route in Madrid, Spain to test the system on real traffic. The data was collected on 26 June 2013 for 24 hours. Every detector recorded a separate LOG file and 45673 devices were detected with 54% being hands free devices. Travel time was estimated with 5 min time difference, 90 km/h speed limit and 2.4 – 3.6 km distance for each direction. Max travel time was calculated by multiplying the estimated travel time by 2.5. Any value beyond this was accepted as an outlier. The results showed that 89% of estimates had error rates lower than 10%. Although the model performed well, BTMS can be difficult to implement in a crowded city with high congestion levels like Istanbul as it might cause a lot of outliers.

Prokhorchuk et al. (2020), focused on travel time distributions of paths instead of estimating expected travel times, because they believed that travel times showed much variability in urban areas. They aimed to predict travel time distributions by using sparse GPS data. In order to use different number of observations for variables, they combined Gaussian Copulas and Bayesian Network algorithms. They collected 15.000 taxis' data for August 2011 in Singapore by using GPS devices. The data contained coordinates, taxi identifier, timestamp, and state of taxi that can be; free, on call or passenger on board. By focusing on passenger on board state in order to get real traffic conditions, they divided the data into 1-hour intervals and used 70% of the paths to obtain travel time distributions. Then they computed Kullback-Leibler divergence and Hellinger distance by using 30% of the paths. They tested 50 paths with 1-hour intervals in order to get sufficient coverage. The model was compared to several other models such as; covariance matrix with where all links are independent, a Partial Empirical Covariance Matrix (PECM) as covariance matrix, a PECM where non-neighboring links were zero, a PECM with graphical lasso, a Bayesian Inference of High-Dimensional Sparse Networks framework (BISN) on entire network and a BISN on each path. The results showed that path based BISN performed better than other methods. However, it required more computational

time to proceed than other models. Therefore, it can be difficult to implement BISN when the amount of data is big.

Support Machine Regression is another type of Machine Learning Algorithm. Wang et al. (2009), proposed a model with this method that used departure time of a bus as input. Wu et al. (2004), studied on Support Vector Machines (SVM) and proved that the method had high performance difference when compared to others. Support Vector Regression is an application of SVM, it is an improved SVR (Schölkopf et al., 2000). SVM uses a kernel function to map the data in the input space to a higher dimensional space. In the model departure time, length of link, number of intersections were used as input parameters. It was assumed that traffic conditions were similar in weekdays. Therefore, they replaced the traffic condition parameter with the departure time. Bus travel time depends on link length and intersection delays, so these were also accepted as input parameters. Additionally, historic travel data was used as input. After deciding the inputs, they followed a five step guide to build the model.

- The route was separated into links.
- v-SVR (a modified Support Vector Regression) was used as basic algorithm and Radial Basis Function (RBF) Kernel was used as Kernel function.
- The data was divided into two, first data trained the SVM and second data was used to calculate the predicting matrix.
- LIBSVM (Chang et al., 2001) was used to optimize the parameters.
- Training of the SVR and calculation of travel time.

The results showed that the model could estimate the travel time well. However, the model should be compared to ANN models in order to decide the performance of SVM. Obtaining parameters of the Kernel function took a lot of time in the model, even they were optimized, time problem still couldn't be solved.

Although there are numerous studies on travel time prediction, Istanbul still needs a reliable estimation system because some of the previous works focus on special events but Istanbul needs a solution that works anytime. Some works use neural

networks or vector machines, although they perform well, Istanbul demands a faster algorithm that responds in a very short time period. Most of the models focused on short term prediction and they will not perform well because there are really long lines in Istanbul with longer intervals. Some works used limited amount of data and this provided better computation time. However, traffic data is extensive in Istanbul and processing the data as well as using it to build a system requires a robust and agile model. Kalman Filter algorithm was used earlier in Istanbul and because accuracy of the method fell below 60% it is not in use anymore. The models that work with mobile phone data instead of GPS data, might not be efficient for the city because roads are not isolated from pedestrian traffic and traffic congestion is a big part of traffic flow.

The solution needs to have specific parameters for the city to simplify the complexity, keeping in mind flexible transport habits and the constant movement of 5 million passengers between districts per day. The model should also be easily implementable in terms of hardware installation because there is a complicated infrastructure and a big traffic network with a lot of bus stops in the city.

In this study, a Multiple Linear Regression model was developed to predict travel time for each bus stop interval by taking into account specific factors.

3. MODEL

3.1. Machine Learning Approach and Types of ML Algorithms

Artificial Intelligence (AI) is a technological concept that support building automated systems free from human control. Machine Learning (ML) is a part of AI focusing on building applications that learn from experience and complete some assigned tasks accordingly (Ray, 2019). For example a robot vacuum cleaner can record amount of dust when it is started and learn from this data, then can decide when to work next time by itself based on the data it learnt. Likewise, an online meeting software recording attendees' facial expressions by their permission, can decide if any attendant is distracted on topic and it can alert the speaker to be more interactive.

Recent technologies allow us to produce huge amount of data because almost everything is online now. Cows in a farm are online for tracking purposes, buses in transportation are connected for the same purpose, mobile phones, digital glasses and smart houses are online and they produce data almost every second. The areas where this big data is used are quite wide. For example, data collected from online buses are used by Intelligent Transportation Systems (ITS) for planning transportation, deciding on settlements and making infrastructural arrangements. In addition, this data plays an important role in analyzing travelers' habits, identifying new needs and seeing problems. Using this data correctly to develop new systems, big cities can offer their people a comfortable living space.

Formerly, collected data were analyzed by human. However, as volume of data increased, need of a powerful tool was born, so computers are started to be used to evaluate the big data (Ray, 2019). Machine Learning algorithms use this big data to learn and maintain automated solutions. Prediction of anything is one of the common tasks run by ML by using massive amount of data.

Basically, ML algorithms are categorized into four types; supervised, unsupervised, semi-supervised and reinforcement learning (Portugal et al., 2015). In supervised ML,

model learns from labelled data where in unsupervised ML unlabeled data is used. Linear Regression, Decision Trees, Support Vector Machine (SVM) are examples of supervised types while Gaussian Mixture and K-Nearest Neighbor (KNN) are examples for unsupervised types. In this paper a Multiple Linear Regression algorithm is used to make a prediction model.

3.2. Multiple Linear Regression

Linear Regression algorithm is used to predict a dependent variable based on an independent variable. It sets a relationship between the two variables and tries to draw a line that is closest to the real data. When number of independent variables more than one, then the algorithm is called Multiple Linear Regression.

In this work, dependent variable is the travel time to be calculated while independent variables are weather conditions, time of day, day of week and day of year as seen on Figure 3.1. The logic of regression is to develop a relationship between these input variables and the output variable. (Zhang et al., 2019) The independent input variables are the only factors determining the travel time.

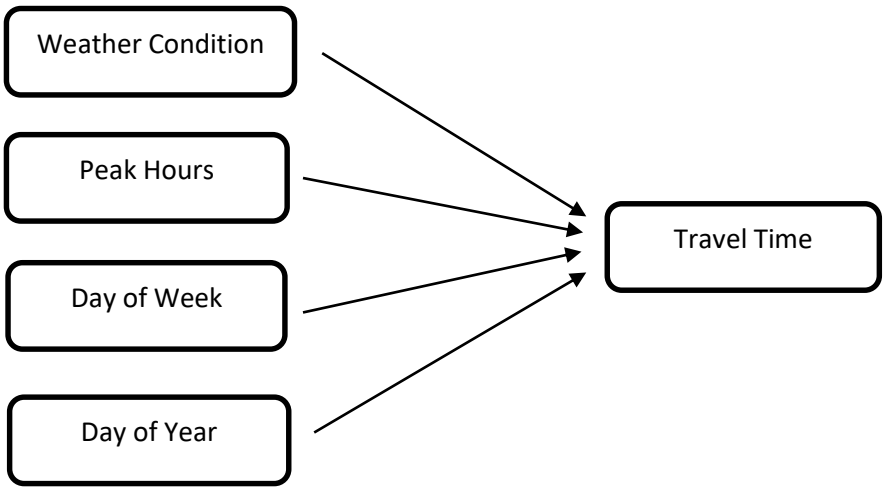

Figure 3.1. Multiple Linear Regression input and output variables for Travel Time Prediction Model

3.3. Line Information

There are approximately 7000 public transportation buses in Istanbul. Each bus has a Line Code and a unique Door Number. Line Codes consist of numbers/numbers and letters e.g. 90, 16C, 18K. Every journey taken by a bus has a unique Journey Id along with a Stream Code showing the direction as every line has 2 directions. There are 933 smart bus stops showing predicted travel time of these buses to the passengers. In addition to that, a mobile app has the same feature that allows passengers planning their journey before leaving their house. Therefore, showing the right time to the passenger is important.

3.4. Preparation

3.4.1. Parameter selection

There are various factors effecting the travel flow in Istanbul including driver's behavior, speed of vehicles, age of vehicle, infrastructure, traffic lights, and weather. For example, too many traffic lights on a line interrupts the flow and causes constant ups and downs at the speed of the bus. Furthermore, some events like football games, marathons, graduation ceremonies, openings and celebrations cause traffic congestion. It is because every district of the city is designed with activity areas, educational institutions and shopping centers to serve the people of the region. People here prefer to spend time outside and participate in outdoor activities, as they are generally people of temperate climate. Although it is possible to include most of these factors into a prediction method, results might not be accurate as expected in this case. It is because usage of too many variables increases complexity and decreases the performance.

In this study four very effective variables were selected. First, school periods and summer holidays are known to affect public transportation on a large scale in context of traffic density. There are 57 universities and 9103 schools in Istanbul. Every year, the city hosts large number of foreign students, especially from Europe. Students

who come to Istanbul both as exchange students and to complete their higher education here, besides making a great contribution to the city's economy, play a role in promoting Istanbul abroad. (Üniversite Şehri İstanbul, 2020) Total number of students is about 4 million including university students in the city. The students use shuttle buses and public transportation to go to school. IETT provides an affordable student transportation card for students to encourage usage of public transportation. Therefore, during school periods usually high density is observed.

In summer more people tend to spend time outside. However, traffic density is not high because schools are closed and shuttle buses for students are not in traffic. As a result, there is a visible relief on traffic in summer. In the model, day of year parameter was used to include the school and holidays. It is because day of year parameter covers seasons, school periods and holiday seasons. Thus same days of years tend to show similar results.

Second, traffic pattern on weekdays and weekends diversifies. People go to work on weekdays and schools are open on weekdays. On some days of week, regular events take place at miscellaneous points of the city. For example, local bazaars are set on some streets and these streets are reserved to only pedestrians for a certain period of time. Since there is no vehicle entrance to these streets, the traffic flow is transferred to other directions. This transfer can double the traffic jam. In addition to that, historic places receive a lot of visitors on weekends. Especially museums, old mosques and other type of structures are very common and close to each other in Eminönü. Tourists can visit these places by walking. Although most parts are closed to traffic in Eminönü, a lot of people come from other districts by buses and their cars. Likewise, other districts have many attractions on weekends. Therefore, day of week parameter was used in the model by taking into account that every day might have different effect.

Third, traffic flow changes at different times of a day and congestion is quite heavy during peak hours. In the past, peak hours were known as 2 early hours in morning (07:00 - 09:00) and 2 hours in evening (17:00 - 19:00), but these periods recently

expanded, and the traffic density increased at noon compared to other hours because people spend time outside during lunch break by taking a walk or doing little shopping after lunch. As a result of this, any hour can be a peak hour for a specific line. Thus, minute of day parameter was selected to show peak hours. Minute of day is more sensitive than hour of day, this provides more factual peak hours definition as peak hours might be 2.5 hours in morning and 3.5 hours in evening or vice versa.

Finally, rainy weather significantly affects the traffic in Istanbul. Unexpectedly, it was observed that the rainfall on the examined lines affected the traffic positively. In other words, buses arrive from one stop to another in a shorter time in rainy weather. The major reason is that, these lines are located on places with heavy pedestrian traffic. Since people do not go out in rainy weather, pedestrian traffic and therefore the number of passengers using the bus are decreasing. Accordingly, the waiting time of the bus at the station is also reduced. Although drivers are extra careful to prevent any accident and use vehicles with lower speed, travel times of the buses are still less. While this situation occurs for some lines, the effect of precipitation on different lines might be different. For example, it has been observed that the traffic slows down on some of longer lines and high ways and the travel time is prolonged due to accidents caused by rain. To observe the exact effect of the weather on traffic, weather data was included to the model.

3.4.2. Line selection

Istanbul is a metropolis with thousands of years of history, thus it has an infrastructure that differs for each part of the city. The historical texture is protected by the state and roads are not intervened in these areas. On the other hand, in newly constructed districts, roads are wider and transportation is planned. For example, recently, city planners ensure that new settlements are close to metro to provide faster transportation and to prevent traffic jam.

Bus lines in the city can be categorized into 5 different types; touristic lines, urban lines, suburban lines, relatively short lines and combination of all types. Initially the

line 90 was selected for this model because it is a combination of almost all line types. It starts at a small but crowded region. In this region, the roads are relatively narrow due to the historical structure. There is a local bazaar on Wednesdays and it causes pedestrian traffic. The line continues in a very crowded street with shopping centers and historical places receiving many shoppers from other parts of the city every day. It ends in the touristic centers Karaköy and Eminönü as seen on Figure 3.2 where usually high congestion is observed (IETT, 2020).

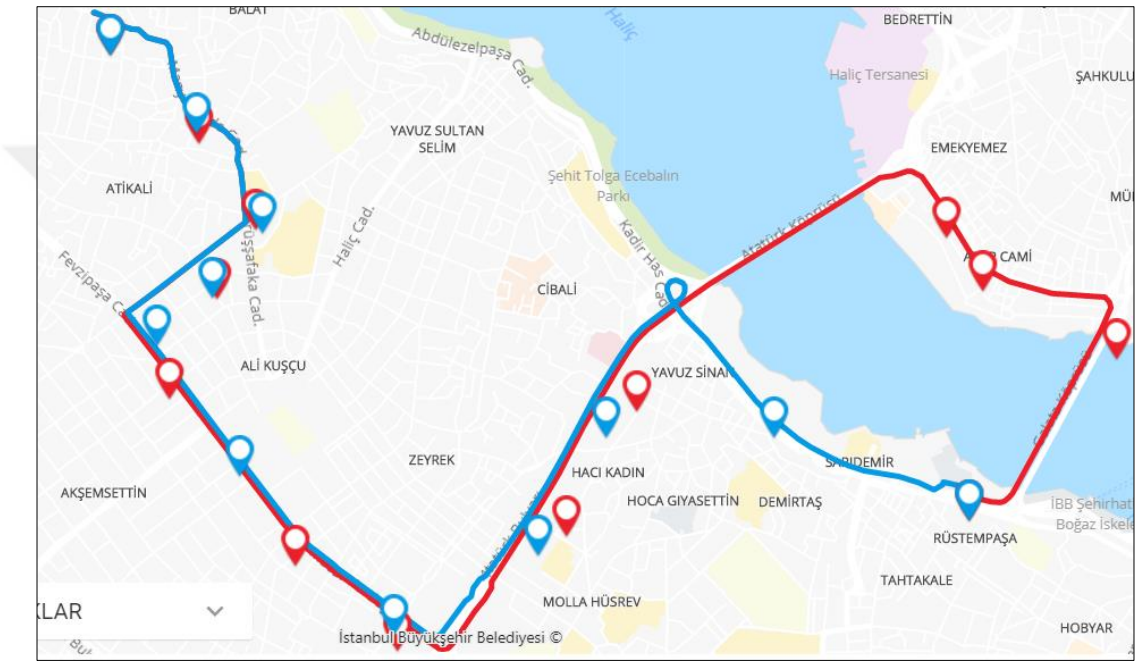

Figure 3.2. Line 90 route (IETT, 2020)

Later, 6 more lines were added to the work to ensure that the model shows accuracy for any lines; 11ÜS, 14, 16C, 17, 18K and 252. These lines have different characteristics. The line 252 operates between Asia and Europe, starts in Pendik district, passing through the Bosphorus Bridge ends in Şişli. The line 17 uses the coastal route, starts in Pendik and reaches Kadıköy by passing through Bağdat Street, which is a very busy street with many famous brands' shops. The line 14 was selected because it has an irregular infrastructure and we wanted to test irregularities. 18K and 11ÜS are the lines of Sultanbeyli district where the settlement density is high. These lines end in Kadıköy and Üsküdar districts where settlement is even more intense.

Table 3.1. Number of bus stops, length in km and total Travel Time in min for selected lines

Line	Bus Stops	Length	Total Travel Time
11ÜS	55 - 58	35 km	83 min
14	71 - 73	26.5 km	66.5 min
16C	72 - 71	35 km	108 min
17	68 – 67	28 km	67 min
18K	31 – 31	32 km	76.5 min
252	57 – 57	34.6 km	83 min
90	13 – 11	5.4 km	12 min

Table 3.1 shows number of bus stops on the lines for each direction, length of the line in km and measured total travel time in minutes. Although some lines are really long with 73 bus stops and 35 km length, there are even longer lines in other parts. And some are pretty shorter like the line 90.

Table 3.2. Starting and ending districts with populations for selected lines

Line	Starting District	Population	Ending District	Population
11ÜS	Sultanbeyli	336.021	Üsküdar	535.916
14	Ümraniye	710.280	Kadıköy	482.713
16C	Pendik	711.894	Kadıköy	482.713
17	Pendik	711.894	Kadıköy	482.713
18K	Sultanbeyli	336.021	Kadıköy	482.713
252	Kartal	470.676	Şişli	279.817
90	Fatih	443.090	Fatih	443.090

Table 3.2 shows where the lines start and end. Some selected lines operate in Asia and some in Europe. Some of the lines operate in districts with more than half million population. For example, Pendik has a population of more than 710.000 and others are very close to a half million. These numbers are almost same as population of some cities in the world. Finding proper solution for such populations is an arduous task. Planning a suitable transportation that works for everyone requires a nonstop and perceptive work ethic.

3.4.3. Field observations

Before starting development of the model, we decided to make a fieldwork and experience several journeys. A field observation is experiencing real life and collecting data out of laboratory. This type of work provides having a guide during the research and a validation mechanism to compare research data to the real data. As stated earlier, weather conditions, school time and peak hours are the most common components that slows the traffic flow significantly. The fieldwork was done to observe the effects of these components for the line 90. At different times of days, the journeys were monitored and whenever the bus arrived to a bus stop, timestamp was recorded.

Table 3.3. An observation result for line 90 for first direction

Bus Stop	Order	Time Recorded	Travel Time in sec
Eminönü	1	18:20:00	0
Haliç Metro	2	18:24:30	150
Unkapanı	3	18:27:00	180
Vefa	4	18:28:20	80
Fatih İtfaiye	5	18:31:44	144
Fatih	6	18:35:25	221
Yavuz Selim	7	18:40:26	301
Nişanca	8	18:41:30	64
Çarşamba	9	18:42:50	80
Şehit İbrahim Yılmaz	10	18:45:10	190
Draman	11	18:46:40	90
		Total Travel Time	**25 min**

Name of bus stop, order of it, the time the bus reached to the stop was recorded as seen on Table 3.3 and Table 3.4. Then travel time was calculated by subtracting simultaneous recorded times and converted to seconds. The same process repeated for both directions.

Table 3.4. An observation results for line 90 for second direction.

Bus Stop	Order	Time Recorded	Travel Time in sec
Draman	1	18:57:00	0
Şehit İbrahim Yılmaz	2	18:58:30	90
Çarşamba	3	18:59:55	85

Nişanca	4	19:01:57	122
Yavuz Selim	5	19:04:45	168
Fatih	6	19:06:37	112
İtfaiye	7	19:08:33	116
Vefa	8	19:11:15	162
Unkapanı	9	19:11:55	40
Azapkapı - Haliç Metro	10	19:13:30	155
Perşembe Pazarı	11	19:14:25	55
Karaköy	12	19:15:32	67
Eminönü	13	19:16:55	83
		Total Travel Time	**19,25 min**

Field observations were repeated 18 times on different days and hours of the days, under different weather conditions and during the local bazaar. The travel time for first direction ranges between 13 min and 29 min and for second direction between 17 min and 48 min. In short, a journey takes min 12 min and max 60 min. Table 3.5 shows the total travel time for each direction on different days. There were times where traffic flow was very low and a lot of congestion observed because of pedestrian traffic. Travel time for the same interval might take between 30 seconds to 5 minutes during the congestion. Moreover, traffic lights were another reason for traffic congestion.

Table 3.5. Comparison of travel times for each journey observed

	Date	**Day of Week**	**Total Travel Time in sec**	
			First Direction	**Second Direction**
1	15.01.2020	Wednesday	1500	1255
2	16.01.2020	Thursday	1085	1518
3	21.01.2020	Tuesday	1005	1015
4	22.01.2020	Wednesday	1020	1233
5	22.01.2020	Wednesday	1217	1568
6	22.01.2020	Wednesday	1355	2157
7	22.01.2020	Wednesday	1564	2515
8	23.01.2020	Thursday	840	1259
9	23.01.2020	Thursday	796	1315
10	23.01.2020	Thursday	968	1597
11	23.01.2020	Thursday	1533	1425
12	24.01.2020	Friday	966	1207
13	24.01.2020	Friday	940	1390
14	24.01.2020	Friday	1559	1327
15	24.01.2020	Friday	1265	1454

16	25.01.2020	Saturday	1410	2460
17	25.01.2020	Saturday	1756	2880
18	01.02.2020	Saturday	1360	1315

In addition, a survey was conducted with the drivers on their travel experience. Some general questions asked to the drivers are:

- How many years have you been operating?
- What is the longest time you waited between 2 stops?
- What days is congestion highest?
- What time is congestion highest during the day?
- How does rain/snow effect a journey?

The first question was asked to the drivers to ensure that they have enough experience on facing irregularities. Most of the drivers have been operating on the line at least for 2 years.

The average longest time they waited in traffic is 10 – 12 min because of an incident, a bad congestion and even a concrete mixer blocking the road. During the field observation, we experienced a similar situation, a car blocked the road for about 7 minutes and because the roads are narrow in the area, the bus had to wait until the road was open again.

All drivers agreed that congestion was highest on weekends and Wednesdays. On weekends people go out and travel a lot for shopping or local events. On Wednesdays there is a bazaar attracting many people. Speed of the buses is relatively low on these days.

Most of the drivers agreed that cold weather and rain caused less congestion because people do not prefer to be outside and roads are open. This was experimented on the line 11ÜS also, during 2 heavily rainy days, roads were open and there was no traffic jam so travel time was short. This was confirmed in the prediction model also.

Travel time calculated from one stop to another is less when it is rainy on the line 90. However, rain still might show opposite results on some other areas.

The drivers said that during a day, peak hours are the busiest, which are between 06:00 – 09:00 when people go to work and school in morning, and 17:00 – 20:00 in evening when they turn back home. Some traffic engineers claim that peak hours in the city are even wider than 3 hours and some believe that afternoon breaks should also be defined as peak hours which are between 12:00 and 13:00. In this work, hours with busiest traffic as declared by the drivers were accepted as peak hours.

Traffic accidents are one of the reason for congestion according to the drivers. Although the line is short, there are many traffic lights on the line and journeys are interrupted by the lights often. As a result, the general reason most agreed for congestion is narrow roads and infrastructure.

Last of all, they expressed that there was a time limit to be complied for each direction. The drivers can adjust the total travel time of a journey to complete within the given time limit. The velocity of the bus does not depend on the traffic flow but the time limit. Therefore, they do not speed up to reach the bus stop as soon as possible. This causes fluctuation in the travel time for the same interval.

Field observations and the survey with the drivers showed that there are new things to learn from experience in spite of having a huge data in hand. It also ensured that data could be interpreted correctly during data analysis.

3.4.4. Data collection

IETT provides an uninterrupted public transportation service in Istanbul for 24/7. Approximately 50.000 journeys are made every day by buses on about 814 bus lines. The buses are tracked by on board computers. Every piece of journey between two stops is recorded, and this big data is used by central management to develop a live system for transportation planning, needs analysis and problem detection. Travel

data is obtained from Global Positioning System (GPS) via 3 types of In Vehicle Computers located in the buses. The devices are regularly checked and updated to newer versions in order to gather right information and prevent missing data. Whenever a bus reaches to a bus stop, the on board computers send data shown in Table 3.6 to the center and the data is stored into a database simultaneously.

Table 3.6. Columns of raw data gathered from GPS devices

Travel Data	**Type**
JourneyId	Int
Line	String
BusDoorNo	String
BusStopId	Int
BusStopOrder	Int
StreamCode	String
TravelDate	DateTime

In order to build the prediction model, every journey from one bus stop to another recorded in 2019 for 7 lines was extracted from the database. There are 3.667.944 records for the line 11ÜS, 1.957.388 records for the line 14, 502.384 rows for 16C, 1.532.065 records for 17, 1.402.836 records for 18K, 1.072.926 records for 252 and 336.256 records for the line 90 for 2019.

After development of the model, 2020 data were extracted to use on validation step and to make sure that the model works on new data. However, because of the Covid-19 curfews there were irregularities on the transportation data after March. Some days, there were no public transportation. Therefore, only data that belongs to first two months were extracted for the line 11US, 17, 14, 252, 16C and 18K. For the first two months of 2020, there are 560.668 records for 11ÜS, 212.763 records for 14, 68.459 records for 16C, 186.357 records for 17, 183.782 records for 18K, and 111.094 records for 252.

The weather archive was downloaded from the weather forecast web site (Weather archive in Istanbul (airport), METAR, 2020) for Istanbul. The archive file contains detailed weather information for every 30 min of a day for the years 2019 and 2020.

3.4.5. Data cleaning

When the data was analyzed for the model, some missing records, negative values and duplicate records were detected due to disconnections on the GPS. This data has been filtered out.

Z-score was applied to remove any outliers from the data files. Z-score is a method that finds the distance between a value and the mean, it is also called standard score. It is calculated by subtracting the mean μ from a value x and dividing it to the standard deviation σ as seen on Equation 1. (Z Table, 2020)

$$z = \frac{x-\mu}{\sigma} \quad (1)$$

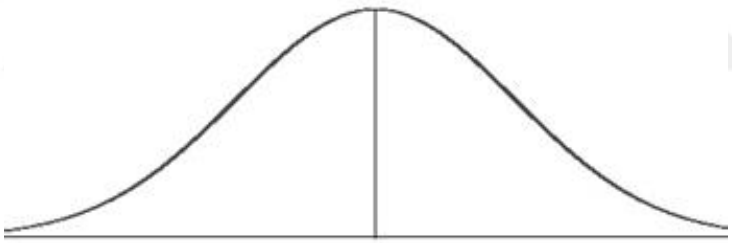

Figure 3.3. Normally distributed travel data graph (Using the Z Table, 2020)

We expect that the real travel time values normally distributed as seen on Figure 3.3 (Using the Z Table, 2020). A normal distribution is a bell shaped curve consisting two symmetrical pieces. The total piece under the curve equals to 1. Most types of data are naturally normally distributed. However, data analysis on travel times showed that there are many outliers that fluctuates the distribution of the data as seen on Figure 3.4.

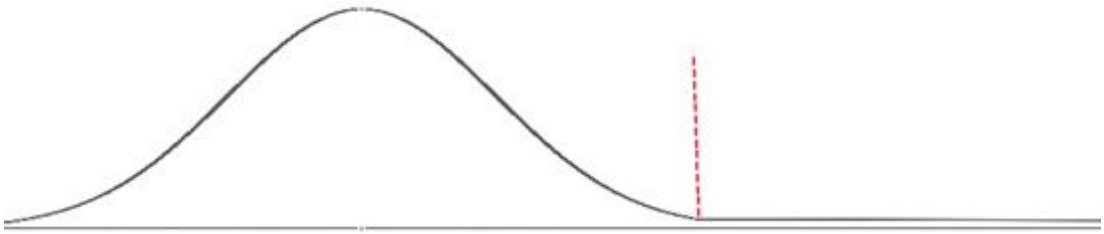

Figure 3.4. Real time travel data distribution

In order to gain a normally distributed data set, we applied Z-score tables and cut the data at the point red dotted line shows on Figure 3.4. There are two types of Z-score tables, negative and positive. The tables define corresponding z-values with a percentage to show an interval for a better distribution. We wanted to use at least 95% of the data. By using the tables below, a filter was built. To obtain 95%, first intersection of -1.9 from first column (-z) and -0.06 column is extracted which corresponds to 0.025 from negative z-score table, Table 3.7 (Z Table, 2020). So –z value is -1.96. Then intersection of 1.9 and 0.06 column is extracted from Table 3.8 (Z Table, 2020), which is 0.975. So +z value is 1.96. Then in the model these filters were used.

Table 3.7. Negative z-score table (Z Table, 2020)

-z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.50000	0.49601	0.49202	0.48803	0.48405	0.48006	0.47608	0.47210	0.46812	0.46414
0.1	0.46017	0.45621	0.45224	0.44828	0.44433	0.44038	0.43644	0.43251	0.42858	0.42466
0.2	0.42074	0.41683	0.41294	0.40905	0.40517	0.40129	0.39743	0.39358	0.38974	0.38591
0.3	0.38209	0.37828	0.37448	0.37070	0.36693	0.36317	0.35942	0.35569	0.35197	0.34827
0.4	0.34458	0.34090	0.33724	0.33360	0.32997	0.32636	0.32276	0.31918	0.31561	0.31207
0.5	0.30854	0.30503	0.30153	0.29806	0.29460	0.29116	0.28774	0.28434	0.28096	0.27760
0.6	0.27425	0.27093	0.26763	0.26435	0.26109	0.25785	0.25463	0.25143	0.24825	0.24510
0.7	0.24196	0.23885	0.23576	0.23270	0.22965	0.22663	0.22363	0.22065	0.21770	0.21476
0.8	0.21186	0.20897	0.20611	0.20327	0.20045	0.19766	0.19489	0.19215	0.18943	0.18673
0.9	0.18406	0.18141	0.17879	0.17619	0.17361	0.17106	0.16853	0.16602	0.16354	0.16109
1.0	0.15866	0.15625	0.15386	0.15151	0.14917	0.14686	0.14457	0.14231	0.14007	0.13786
1.1	0.13567	0.13350	0.13136	0.12924	0.12714	0.12507	0.12302	0.12100	0.11900	0.11702
1.2	0.11507	0.11314	0.11123	0.10935	0.10749	0.10565	0.10384	0.10204	0.10027	0.09853
1.3	0.09680	0.09510	0.09342	0.09176	0.09012	0.08851	0.08692	0.08534	0.08379	0.08226
1.4	0.08076	0.07927	0.07780	0.07636	0.07493	0.07353	0.07215	0.07078	0.06944	0.06811
1.5	0.06681	0.06552	0.06426	0.06301	0.06178	0.06057	0.05938	0.05821	0.05705	0.05592
1.6	0.05480	0.05370	0.05262	0.05155	0.05050	0.04947	0.04846	0.04746	0.04648	0.04551
1.7	0.04457	0.04363	0.04272	0.04182	0.04093	0.04006	0.03920	0.03836	0.03754	0.03673
1.8	0.03593	0.03515	0.03438	0.03363	0.03288	0.03216	0.03144	0.03074	0.03005	0.02938
1.9	0.02872	0.02807	0.02743	0.02680	0.02619	0.02559	0.02500	0.02442	0.02385	0.02330
2.0	0.02275	0.02222	0.02169	0.02118	0.02068	0.02018	0.01970	0.01923	0.01876	0.01831
2.1	0.01786	0.01743	0.01700	0.01659	0.01618	0.01578	0.01539	0.01500	0.01463	0.01426
2.2	0.01390	0.01355	0.01321	0.01287	0.01255	0.01222	0.01191	0.01160	0.01130	0.01101
2.3	0.01072	0.01044	0.01017	0.00990	0.00964	0.00939	0.00914	0.00889	0.00866	0.00842
2.4	0.00820	0.00798	0.00776	0.00755	0.00734	0.00714	0.00695	0.00676	0.00657	0.00639
2.5	0.00621	0.00604	0.00587	0.00570	0.00554	0.00539	0.00523	0.00509	0.00494	0.00480

2.6	0.00466	0.00453	0.00440	0.00427	0.00415	0.00403	0.00391	0.00379	0.00368	0.00357
2.7	0.00347	0.00336	0.00326	0.00317	0.00307	0.00298	0.00289	0.00280	0.00272	0.00264
2.8	0.00256	0.00248	0.00240	0.00233	0.00226	0.00219	0.00212	0.00205	0.00199	0.00193
2.9	0.00187	0.00181	0.00175	0.00170	0.00164	0.00159	0.00154	0.00149	0.00144	0.00140
3.0	0.00135	0.00131	0.00126	0.00122	0.00118	0.00114	0.00111	0.00107	0.00104	0.00100
3.1	0.00097	0.00094	0.00090	0.00087	0.00085	0.00082	0.00079	0.00076	0.00074	0.00071
3.2	0.00069	0.00066	0.00064	0.00062	0.00060	0.00058	0.00056	0.00054	0.00052	0.00050
3.3	0.00048	0.00047	0.00045	0.00043	0.00042	0.00040	0.00039	0.00038	0.00036	0.00035
3.4	0.00034	0.00033	0.00031	0.00030	0.00029	0.00028	0.00027	0.00026	0.00025	0.00024
3.5	0.00023	0.00022	0.00022	0.00021	0.00020	0.00019	0.00019	0.00018	0.00017	0.00017

Table 3.8. Positive z-score table (Z Table, 2020)

z	**0.00**	**0.01**	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
0.0	0.50000	0.50399	0.50798	0.51197	0.51595	0.51994	0.52392	0.52790	0.53188	0.53586
0.1	0.53983	0.54380	0.54776	0.55172	0.55567	0.55962	0.56356	0.56749	0.57142	0.57535
0.2	0.57926	0.58317	0.58706	0.59095	0.59483	0.59871	0.60257	0.60642	0.61026	0.61409
0.3	0.61791	0.62172	0.62552	0.62930	0.63307	0.63683	0.64058	0.64431	0.64803	0.65173
0.4	0.65542	0.65910	0.66276	0.66640	0.67003	0.67364	0.67724	0.68082	0.68439	0.68793
0.5	0.69146	0.69497	0.69847	0.70194	0.70540	0.70884	0.71226	0.71566	0.71904	0.72240
0.6	0.72575	0.72907	0.73237	0.73565	0.73891	0.74215	0.74537	0.74857	0.75175	0.75490
0.7	0.75804	0.76115	0.76424	0.76730	0.77035	0.77337	0.77637	0.77935	0.78230	0.78524
0.8	0.78814	0.79103	0.79389	0.79673	0.79955	0.80234	0.80511	0.80785	0.81057	0.81327
0.9	0.81594	0.81859	0.82121	0.82381	0.82639	0.82894	0.83147	0.83398	0.83646	0.83891
1.0	0.84134	0.84375	0.84614	0.84849	0.85083	0.85314	0.85543	0.85769	0.85993	0.86214
1.1	0.86433	0.86650	0.86864	0.87076	0.87286	0.87493	0.87698	0.87900	0.88100	0.88298
1.2	0.88493	0.88686	0.88877	0.89065	0.89251	0.89435	0.89617	0.89796	0.89973	0.90147
1.3	0.90320	0.90490	0.90658	0.90824	0.90988	0.91149	0.91308	0.91466	0.91621	0.91774
1.4	0.91924	0.92073	0.92220	0.92364	0.92507	0.92647	0.92785	0.92922	0.93056	0.93189
1.5	0.93319	0.93448	0.93574	0.93699	0.93822	0.93943	0.94062	0.94179	0.94295	0.94408
1.6	0.94520	0.94630	0.94738	0.94845	0.94950	0.95053	0.95154	0.95254	0.95352	0.95449
1.7	0.95543	0.95637	0.95728	0.95818	0.95907	0.95994	0.96080	0.96164	0.96246	0.96327
1.8	0.96407	0.96485	0.96562	0.96638	0.96712	0.96784	0.96856	0.96926	0.96995	0.97062
1.9	0.97128	0.97193	0.97257	0.97320	0.97381	0.97441	0.97500	0.97558	0.97615	0.97670
2.0	0.97725	0.97778	0.97831	0.97882	0.97932	0.97982	0.98030	0.98077	0.98124	0.98169
2.1	0.98214	0.98257	0.98300	0.98341	0.98382	0.98422	0.98461	0.98500	0.98537	0.98574
2.2	0.98610	0.98645	0.98679	0.98713	0.98745	0.98778	0.98809	0.98840	0.98870	0.98899
2.3	0.98928	0.98956	0.98983	0.99010	0.99036	0.99061	0.99086	0.99111	0.99134	0.99158
2.4	0.99180	0.99202	0.99224	0.99245	0.99266	0.99286	0.99305	0.99324	0.99343	0.99361
2.5	0.99379	0.99396	0.99413	0.99430	0.99446	0.99461	0.99477	0.99492	0.99506	0.99520
2.6	0.99534	0.99547	0.99560	0.99573	0.99585	0.99598	0.99609	0.99621	0.99632	0.99643
2.7	0.99653	0.99664	0.99674	0.99683	0.99693	0.99702	0.99711	0.99720	0.99728	0.99736
2.8	0.99744	0.99752	0.99760	0.99767	0.99774	0.99781	0.99788	0.99795	0.99801	0.99807
2.9	0.99813	0.99819	0.99825	0.99831	0.99836	0.99841	0.99846	0.99851	0.99856	0.99861
3.0	0.99865	0.99869	0.99874	0.99878	0.99882	0.99886	0.99889	0.99893	0.99896	0.99900
3.1	0.99903	0.99906	0.99910	0.99913	0.99916	0.99918	0.99921	0.99924	0.99926	0.99929
3.2	0.99931	0.99934	0.99936	0.99938	0.99940	0.99942	0.99944	0.99946	0.99948	0.99950
3.3	0.99952	0.99953	0.99955	0.99957	0.99958	0.99960	0.99961	0.99962	0.99964	0.99965
3.4	0.99966	0.99968	0.99969	0.99970	0.99971	0.99972	0.99973	0.99974	0.99975	0.99976
3.5	0.99977	0.99978	0.99978	0.99979	0.99980	0.99981	0.99981	0.99982	0.99983	0.99983

2020 data was used for validation step. Therefore, z-score method was not applied on 2020 data. Instead of trimming the data to have a normal distribution, some simpler methods applied to filter 2020 data. Under normal circumstances, a travel time from one bus stop to another takes about max 15 minutes on the selected lines, therefore any outlier not in the range of 0-2000 sec for a single interval, removed from the test data. Although 2000 seconds is about half an hour, we wanted to

consider any unexpected situation that might have effect the traffic flow. Moreover, any record without direction information was also removed.

The weather condition that has a huge effect on traffic flow when compared to other conditions is rainy weather. Rainy/Not rainy information was needed to use in the work. Therefore, weather data was filtered by removing records of days without rain for both years 2019 and 2020.

3.4.6. Data processing

3.4.6.1. Travel data processing

After required amount of data was extracted and properly cleaned, data processing phrase was started in order to generate required data columns for model development. A Python program was developed by importing Datetime and Collections libraries. Then data files containing cleaned travel times for each interval and for weather conditions were included to the program. Journeys were grouped for each Journey Id and by looping through the intervals, travel time was calculated for each bus stop. By using Travel Date in the raw data, measured travel time (MTT) from bus stop t_i to t_j was calculated by using Equation 2 and Equation 3.

$$MTT = t_j - t_i \quad (2)$$

$$j = i + 1 \quad (3)$$

t_j is the travel time required to arrive to a bus stop and t_i is the travel time for previous bus stop. MTT shows the time difference between two bus stops, in other words for one interval.

Each journey was a group of data containing travel times for corresponding intervals. The journey groups were assigned to a dictionary data structure in order to iterate between intervals. Minute of day parameter was calculated as integer data type by using Travel Date. In order to check with rain data, minute of year value was

calculated. If minute of year value was in weather data file, then rain column value was assigned to 1. Day of year and day of week parameters were also retrieved from Travel Date. Considering direction information was not an independent column but merged in Stream Code column, it was taken out as two string output values. All calculated values were united to conceive a data row for each record of journey groups as seen on Figure 3.5. The data rows were printed into another file to use in the model. This process was repeated for 7 lines.

```
formatted_time = time.strptime(travelTime, "%d.%m.%Y %H:%M:%S")

#Calculates minute of day
minuteOfDay = formatted_time.tm_hour * 60 + formatted_time.tm_min

#Calculates minute of year
minuteOfYear = minuteOfDay + 60 * 24 * (int(formatted_time.tm_yday) - 1)

#Append rain data
isRainy = 1 if minuteOfYear in rainFile else 0

#Calculates day of year and day of week
dayOfYear = formatted_time.tm_yday
dayOfWeek = formatted_time.tm_wday

#Extracts direction from StreamCode
if "_G_" in streamCode :
       direction ='G'
elif "_D_" in streamCode:
       direction = 'D'

#Data for output file
dataRow= "%d;%d;%d;%d;%d;%d;%s;%s;%d;%s\n" % (isRainy, dayOfYear, dayOfWeek,
minuteOfDay,journeyId,order,previousBusStop,nextBusStop,travelTime,direction)

outputFile.write(dataRow)
```

Figure 3.5. Python code for travel data processing

3.4.6.2. Weather data processing

Weather data file contained 13 columns including local time and weather condition for every 30 minutes. Another python program was written by importing weather file to extract the date and times only. All datetime columns corresponding the weather conditions such as; rain, snow, and thunderstorm were recorded into the second file in day of year, day of week, minute of day and time format. Since 30 min space corresponded to a big gap, we decided to produce appropriate time intervals

by including every minutes. The second file was imported to the code and by splitting data on tab spaces, time was calculated in minutes. Then, an interval was created for 15 minutes before and 15 minutes after each minute value and this interval was filled by one minute. The result set was saved into an output file as seen on Figure 3.6. This way, rainy days were obtained in minute of year format which was a high resolution data. The process was repeated for both 2019 and 2020 weather files.

```
#Reads data from weather file
weatherFile = open("weather.csv")

#A new file for output
rainFile = open("rain-2020.txt","w")

w = {}

#For every record interval in weather file, generates minutes
for line in weatherFile.readlines():
	line = line.strip()
	f = line.split("\t")

	rDay = int(f[0]) - 1
	rMin = int(f[2])
	x = rDay * 24 * 60 + rMin
	for tmp in range(x-15,x+15):
		w[tmp] = 1


for x in w.keys():
	rainFile.write("%s\n" % x)


weatherFile.close()
```

Figure 3.6. Python code for processing weather data

3.5. Model Development

When the required data was collected and processed after cleaning step, the development of the model was started. The model was built by developing a Python program on Anaconda Navigator, Jupiter Network editor. Pandas, Numpy, Sklearn, Matplotlib, Random, DateTime and Collections libraries were imported to the program.

According to Linear Regression models there is a linear relationship between the input variables $X=\{X_1, X_2,....X_n\}$ and an output variable Y as shown in Equation 4.

$$Y = T_0 + T_1X_1 + T_2X_2 + .. + T_nX_n \quad (4)$$

Where $T_0 \ldots T_n$ are the coefficients to be calculated and $X_1 \ldots X_n$ are the parameters selected and Y is the value being predicted.

Total travel time T was calculated as sum of predicted travel time (P) for each bus stop interval by using Equation 5 and Equation 6. Every component used in the formulas are explained in Table 3.9.

$$T = \sum_{i=1}^{n-1} P(X,Y)_i \quad (5)$$

$$P(X,Y) = Y_0 + Y_1X_1 + Y_2X_2 + \cdots + Y_4X_4 \quad (6)$$

Table 3.9. Symbol table for Predicted Travel Time equation

Symbol	**Explanation**
P	Predicted Travel Time
Y0	Coefficient
Y1	Rain Coefficient
X1	Rainy/Not Rainy Data
Y2	Day of Year Coefficient
X2	Day Of Year Data
Y3	Day Of Week Coefficient
X3	Day Of Week Data
Y4	Minute Of Day Coefficient
X4	Minute Of Day Data

The formula was used to build the model and by using Python programming language and Sklearn machine learning library as seen on Figure 3.7, a linear regression model was developed.

```
intervalGroups = df3.groupby(by=[previousBusStop, nextBusStop, direction])

for interval in intervalGroups:
    key = "_".join([str(x) for x in interval[0]])
    d = interval[1]
    X = d[['rain','dayOfYear','dayOfWeek','minuteOfDay']]
    Y = d['travelTime']

    #Using %20 of the data for testing, and remaining for training
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.2,random_state= 0)
    #training the model
    model = linear_model.LinearRegression()
    model.fit(X_train, Y_train)

    y_pred = model.predict(X_test)

    #Evaluation with RMSE and R2
    rmse = np.sqrt(mean_squared_error(Y_test, y_pred))
    r2_value = r2_score(Y_test, y_pred)

    results[key] = np.append(model.coef_, [model.intercept_])

results
```

Figure 3.7. Python code to calculate coefficients

First, to indicate every bus stop interval for each direction, data was grouped based on previous bus stop, next bus stop and direction. For instance, 114781_214671_D demonstrates that the interval starts at bus stop 114781, and ends at 214671 while 114781_214671_G shows reverse, the interval is the same but it is in other direction. The lengths of these two intervals are different as well as the infrastructure. Then for each group of data, rain, day of year, day of week and minute of day parameters were used as inputs and travel time as output. By using Sklearn library Linear Regression function the model was trained. 20% of the data was used to test the model while 80% was used to train the model. Then the coefficients were calculated and printed as a key set for each interval as seen on Table 3.10.

Table 3.10. A coefficients result set

200531_261641_G	3.09808069e+00, 1.61771258e-02, -2.87244001e-01, 2.12285246e-03, 6.70748294e+01]
201221_218862_D	-3.13399470e+00, 5.57489975e-02, -5.36025881e-01, 1.44333392e-02, 6.06033967e+01
202921_222342_D	-1.18893144e+01, 8.25585590e-02, -2.22430751e+00, 5.96444210e-03, 7.04156831e+01
205801_222931_D	-4.46143357e+00, 3.53739245e-02, -1.77177929e-01, -5.40890353e-03, 7.86549242e+01
213701_414031_G	-7.87610211, 0.37313132, -0.93674523, -0.0453326 , 43.47584336
215181_225572_G	-1.39166761e+00, 1.42915899e-02, -3.32246253e+00, -2.53399091e-02, 1.12823236e+02
215891_215901_D	-1.33253404e+00, 2.37359875e-02, 1.22344541e-02, 8.86592401e-03, 4.46955661e+01
215892_216171_G	1.11554288e+00, 7.05237753e-03, -4.12318171e-01, 1.49289909e-02, 9.79768545e+01
215901_215911_D	-3.03530804e+00, 2.04760481e-02, 5.22426078e-02, 7.95418004e-03, 3.23571193e+01
215902_215892_G	-9.81743925e-01, 1.42549941e-02, -7.23984046e-02, -2.14535381e-03, 5.32244003e+01
215911_215921_D	-3.90009956e-01, -1.43918911e-01, -1.82219317e-01, 1.13707927e-02, 6.28904807e+01
215912_215902_G	5.03124229e-01, -7.19125837e-03, -1.22714606e-01, -1.87044808e-03, 5.81536421e+01

215921_215931_D	-3.64107311e+00, 2.32991839e-02, 3.06080514e-01, 5.65036515e-03, 2.42887766e+01
215922_215912_G	4.06860925e-01, 7.70725314e-03, 4.38816310e-02, -2.08486179e-03, 4.55111949e+01
215931_217742_D	-5.35204104e-01, 9.27904884e-03, -2.49079811e-02, 1.16485782e-02, 6.07017288e+01
215932_215922_G	-1.01046869e+00, 5.55037313e-03, -2.21103752e-02, 3.62066718e-04, 4.07341836e+01
216021_216032_D	-4.11820186e+00, 3.17889441e-02, -1.36371769e-01, 1.12724339e-02, 3.73322260e+01
216022_218861_G	-2.23780307e+00, 1.10051477e-01, -3.02533034e-01, 6.38993300e-03, 2.50882755e+01
216031_216022_G	-2.51162701e-01, -2.16005074e-02, -2.39073155e-01, 7.05682124e-03, 6.91460214e+01
216032_216042_D	-1.08488380e+01, 8.82390925e-02, -2.38508212e-01, 7.41699679e-03, 4.15434973e+01
216041_216031_G	-9.69089691e-01, 1.70505781e-02, -2.52570188e-01, -8.05271554e-03, 8.85559346e+01
216042_216052_D	7.42924319e+00, -3.73828731e-02, -2.39971177e-02, 8.71476128e-03, 3.15965027e+01
216051_216041_G	-2.08868526e+00, 2.85807007e-03, -1.19731153e-01, -3.80381537e-03, 3.79910737e+01
216052_216062_D	7.48012925e+00, -3.14234631e-02, -1.50599547e-01, 8.91443885e-03, 3.18468899e+01
216061_216051_G	-7.57834005e-01, 2.58686148e-02, 3.54819925e-02, -6.16303226e-03, 4.28372716e+01
216062_216072_D	-5.11501394e-01, 6.43373959e-02, -5.96204481e-01, 3.04668467e-02, 7.59908935e+01
216071_216061_G	-5.91669087e-01, 1.22365258e-02, -8.53410085e-02, -3.38292412e-03, 4.05260684e+01
216072_216811_D	3.30070396e+00, -2.22688086e-02, -6.63989410e-02, 1.42035199e-02, 5.01472606e+01
216101_216822_G	-7.00186080e+00, 7.16487399e-02, -1.23816818e-01, 4.09611375e-03, 8.22852817e+01
216102_403081_D	-3.35754489e+01, 4.27789160e-02, 1.96158849e+00, -9.28048030e-03, 3.18416324e+01
216171_216812_G	1.59198255e+00, 4.62145359e-03, 6.52628690e-02, 1.83258895e-02, 5.98999514e+01
216172_215891_D	-1.13871381e+00, 7.22162795e-03, -5.83040987e-02, 8.15349528e-03, 6.17515885e+01
216192_222172_G	-2.62579699e+00, 3.75917507e-03, -1.40685304e+00, 2.77626643e-02, 7.68821279e+01
216202_216192_G	3.76369308e+00, 1.51272236e-02, -8.96132871e-02, 2.53279529e-02, 4.00228647e+01
216492_216202_G	1.14330892e+00, -1.73514494e-02, -3.78916816e-01, 1.70557208e-02, 5.79283521e+01
216511_222151_D	1.60933655e+01, -9.46788809e-02, -3.18153797e-01, 2.16396356e-03, 4.57893495e+01
216811_216172_D	-1.12788872e+01, 1.08158645e-01, -3.65085471e-01, 2.23131917e-02, 5.96206210e+01
216812_216071_G	-1.77521588e-01, 3.40102900e-02, -1.11688354e+00, 7.85118576e-02, 8.01970094e+01
216821_216102_D	-1.80972330e+01, 2.73331006e-02, -1.90357278e+00, 1.79683842e-02, 5.65643752e+01
216822_216832_G	1.07560846e-01, 1.12914038e-02, 3.07919784e-01, -3.03810516e-03, 7.24037985e+01
216831_216821_D	-5.56426619e-01, 1.74090720e-02, 1.67621642e-01, 7.23341064e-03, 5.64667238e+01
216832_229812_G	8.42481632e+00, -3.85242500e-02, 2.10876842e-01, -1.14634486e-02, 6.07021778e+01
216841_229811_D	1.45298678e+00, -1.35421173e-01, -2.70462025e-01, 8.82634004e-03, 5.50585565e+01
216842_217741_G	3.68620080e-01, 5.72443354e-03, -1.17276605e-01, -3.75323313e-03, 7.25174675e+01
217061_260112_G	-1.88766366e+00, 1.10885865e-02, -4.16650721e-01, 2.65834615e-02, 5.14810768e+01
217131_222262_G	3.85679424e+00, -2.65120574e-03, -1.62016447e+00, -2.29690142e-03, 6.44636674e+01
217741_215932_G	2.69026691e-01, 3.85916790e-03, -1.69103861e-01, -2.67012461e-03, 4.61955496e+01
217742_216841_D	-1.64057545e+00, -1.38722617e-01, -1.83547543e-01, 1.00845659e-02, 6.45156274e+01
217941_222311_D	-1.03719244e+00, 2.43021411e-02, -1.19993489e+00, 3.51148776e-03, 6.21057704e+01
217942_222302_G	-7.64720360e-02, 1.90209880e-03, -2.52197663e-01, -4.29257498e-03, 3.31619494e+01
218601_222112_G	4.67888592e+00, -3.24015600e-02, -7.94872717e-01, -3.76058411e-03, 8.20571960e+01
218801_222322_G	1.57226127e+00, -2.01872213e-03, -3.66346032e-01, -3.47671250e-03, 3.81182829e+01
218802_222331_D	-5.38216392e+00, 4.14692077e-02, -1.76504442e+00, 3.78398441e-02, 8.20491232e+01
218842_201221_D	-1.00216310e+01, 1.04036581e-01, 2.28812838e-01, 2.55770637e-02, 3.32467338e+01
218851_222422_G	3.07844764e+00, 6.06358765e-03, -2.11947297e+00, 1.55587964e-02, 7.38469485e+01
218861_222432_G	8.20544038e+00, 2.86630441e-02, -1.55497112e+00, 2.99236685e-02, 7.86231310e+01
218862_216021_D	2.03849250e+00, -6.29100351e-03, -2.05149388e-01, 1.19471981e-02, 3.14101713e+01
222061_222972_G	-1.67909567e-01, -2.02010671e-03, -1.24270853e+00, 4.38149488e-03, 5.52201668e+01
222072_222082_D	2.94219875e+00, 2.79756498e-02, -1.41604689e+00, 7.04539158e-03, 5.48170266e+01
222081_222061_G	6.20920171e+00, -1.22144723e-02, -5.21492666e+00, -4.71823361e-03, 1.74104809e+02
222082_222092_D	1.16160288e+01, -6.13035694e-02, -1.18720596e-01, 4.98872831e-03, 7.97939137e+01
222091_224051_G	-3.82647161e+00, -3.56971814e-02, -8.77666828e+00, 1.46269981e-02, 1.69292942e+02
222092_222111_D	-1.98073931e+01, 1.07452754e-01, -3.03212766e-01, 2.44746759e-02, 9.12266246e+01
222111_222122_D	-8.73259364e+00, 5.18810080e-02, -5.14326449e-01, 2.97441722e-03, 6.05893244e+01
222112_285631_G	2.14885812e+00, 9.98223659e-02, -1.09141164e+00, -1.71856393e-02, 7.31665634e+01
222122_223201_D	2.15287429e+00, -4.83166818e-03, -7.95407809e-01, 9.57343514e-04, 4.49651773e+01
222151_222161_D	2.90052946e+00, -6.40494025e-03, -1.82846849e+00, 9.04524070e-03, 9.14719350e+01
222152_261492_G	2.52743453e+00, 4.33049212e-04, -1.77920234e+00, -3.56220844e-03, 6.43359697e+01
222161_222171_D	-9.01089690e+00, 1.11816059e-01, -1.74814419e+00, 1.39751302e-02, 6.13784368e+01
222162_222152_G	1.37924129e+00, -4.62581756e-03, -9.79169239e-01, -2.50026459e-03, 7.70328796e+01
222171_222181_D	1.26961938e+01, -6.31579554e-02, -4.16462681e-01, 1.67252662e-02, 4.77843808e+01

Every interval for each direction has different coefficient values including negative numbers. The coefficient tables were obtained for 6 lines. These values later used to predict travel time of 2020 in the validation step.

3.6. Validation

To validate the model, different types of lines were used in addition to the line 90.

On selected lines, a random bus stop was chosen as a destination point, assuming that a passenger was waiting at the stop for a bus. Then closest bus to the passenger from the real data was picked out. In case the bus was not exactly at the stop but in between two stops, a time difference was obtained from the exact location to the nearest stop. From where the bus was located, travel time for each interval was calculated. By using coefficient values calculated for each interval before, travel time of each intersect was calculated by using Equation 6 until the bus arrived to the destination point. Then the process was repeated for all six lines.

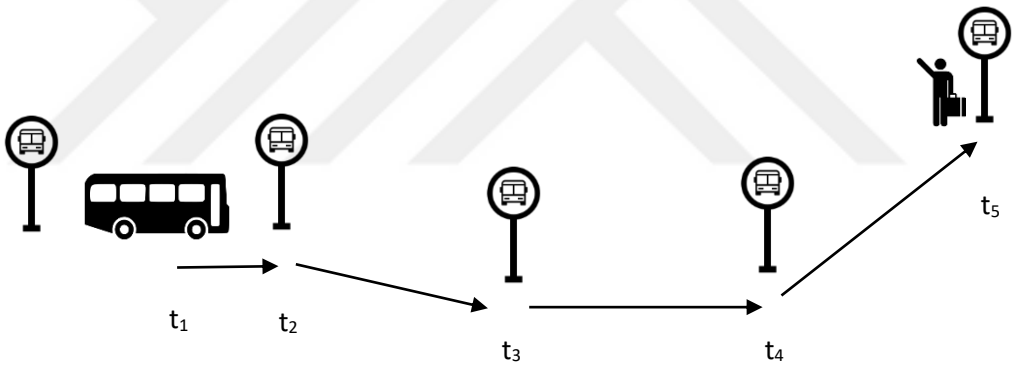

Figure 3.8. Validation of travel time prediction

Travel time t_i as seen on Figure 3.8 for each bus stop was added to each other and a total travel time T was calculated as in Equation 7 where n is number of bus stops and n-1 is number of intervals.

$$T = \sum_{i=1}^{n-1} ti \quad (7)$$

More than 10.000 journeys were generated for each line in this way. As seen on Table 3.11 4th row below, a passenger is at 43rd bus stop and the bus is at first bus

stop. The model predicted travel time as 99.6 minutes while real time is 96 minutes. For this 42 intervals, total time difference is 3.6 minutes.

Table 3.11. Results with real time and predicted time comparison for 11ÜS

Date	Bus Location	Passenger Location	Real Time	Predicted Time	Error Rate
2020-02-06 00:11:00	2	5	25	25.794	0.030787
2020-01-10 10:47:00	2	30	193	187.603	-0.028768
2020-01-21 05:13:00	1	31	43	47.651	0.097608
2020-01-25 21:32:00	1	43	96	99.600	0.036146
2020-01-09 00:43:00	4	20	303	306.819	0.012446
2020-02-02 00:37:00	4	6	281	281.697	0.002474
2020-01-05 08:16:00	15	31	16	16.569	0.034356
2020-01-24 19:31:00	1	16	135	135.147	0.001095
2020-02-04 04:03:00	8	35	810	797.231	-0.016016
2020-01-05 03:47:00	2	24	164	167.519	0.021007

Moreover, the parameters were updated to test different cases. To validate a prediction during peak hours, minute of day parameter was given with an upper and lower bound. This way the program selected certain hours. The location of passenger and the bus was chosen as far as possible to test longest distances. For example, the passenger was located at 70^{th} -75^{th} bus stop where the bus was located at 1^{st} – 5^{th} bus stop. This way the behavior of the model on longer intervals was also tested. Weather condition was defined as rainy for many cases in order to see the rain effect at different times.

After completing validation for all lines, the results were saved and error rates were calculated to compare the results and measure the performance.

4. RESULTS

In order to obtain the results of the system, Root Mean Square Error (RMSE) was calculated with each prediction result. RMSE is a method to measure the error in prediction systems. It is widely used because it reports how far the predicted value is from the real value. There is a cumulative error rate in the model that increases when the bus moves from one interval to another. It is because, if there is 20 seconds deviation on first interval, 15 seconds on second interval and 30 seconds on third interval, when the bus moves from first interval to third one, the total error will be 65 seconds. That is why RMSE was selected. The model generated 10,000 and above predictions for each line. The predicted travel time $\boldsymbol{x_2}$ for each of this set of records was subtracted from the actual travel time $\boldsymbol{x_1}$ and result was squared, the value was divided by the total number of records $\boldsymbol{n}$. All results were summed together and the square root was taken and the RMSE was calculated in this way by using Equation 8.

$$\text{RMSE} = \sqrt{\sum_{i=1}^{n} \frac{(x_2 - x_1)^2}{n}} \tag{8}$$

RMSE was calculated for the predicted journeys of the 6 lines; 11ÜS, 14, 16C, 17, 18K, 252 as seen on Table 4.1.

Table 4.1. RMSE for 6 bus lines; 11ÜS, 14, 16C, 17, 18K and 252

LINE	NUMBER OF GENERATED ROWS	RMSE
11ÜS	13109	0.059
14	11159	0.021
16C	13455	0.020
17	14772	0.016
18K	11002	0.088
252	13561	0.153

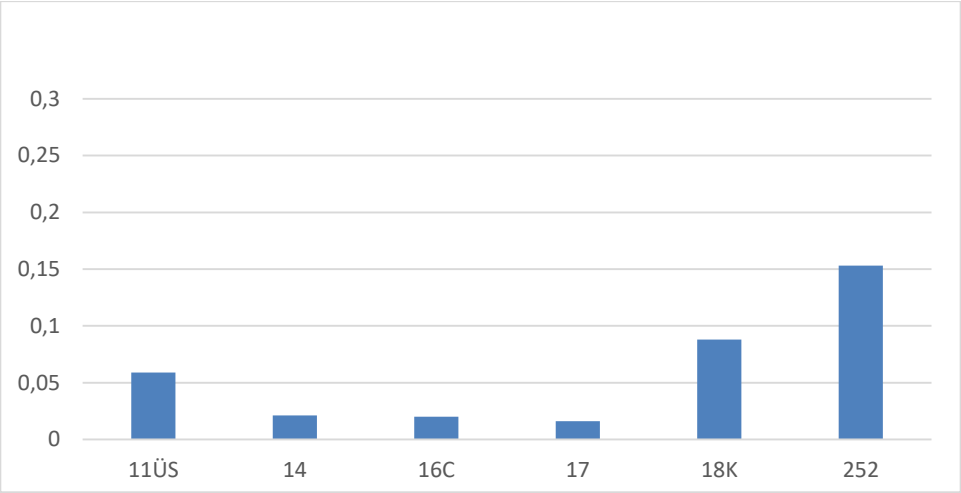

Figure 4.1. RMSE comparison for 11US, 14, 16C, 17, 18K and 252 lines

The RMSE values of the 6 lines resulted as Figure 4.1. For 11ÜS, the validation model estimated 13,109 records and the RMSE ratio was 0.059. 11,159 prediction records were generated for line 14 and the error rate was 0.021. For the line 16C, 13.455 validation records were generated and the error was 0.020 for these records. For the 17 line, the validation found 14,772 records, and the RMSE output 0.016. 11,002 records were processed on the 18K line, and RMSE was 0.088. 13.561 records were found for the 252 line and the RMSE value was 0.153 for this line. The best result belonged to 17 line, the least successful line was 252. The reason why these numbers differ so much depends on the historical data being processed and the four basic parameters (day of the year, day of the week, minute of the day and rain) included in the model. Even the line 252 shows sufficient accuracy and these rates are accepted as successful in public transportation.

Although the regression model was developed based on a single line at the beginning, it showed good performance on all tested lines.

In addition to RMSE calculation, Mean Absolute Error (MAE) was also calculated for each line with 10000+ predicted records. Table 4.2 shows MAE ratio on each line. The difference between the real travel time and predicted travel time could be negative for some records because the predicted value could be smaller or bigger

than the real value. That is why, MAE was calculated to get absolute difference for a clearer value.

Table 4.2. MAE for 6 bus lines; 11ÜS, 14, 16C, 17, 18K and 252

LINE	NUMBER OF GENERATED ROWS	MAE
11ÜS	13162	0.3297
14	11249	0.3598
16C	13337	0.1294
17	14770	0.3343
18K	11108	0.6382
252	13162	0.3297

4.1. Comparison of Prediction Model with Historical Average and Real Travel Time

In order to compare the results with other models and measure the accuracy, firstly Historical Average model was established. For the line 16C, there are about 72 intervals, average travel time was calculated for each interval as seen on Table 4.3 and recorded into a file.

Table 4.3. Historical Average data for line 16C for first direction

Bus Stop Interval	Average Travel Time	Bus Stop Interval	Average Travel Time
206031_227451_G	38.92	224391_227351_G	87.21
206042_224391_G	89.48	225562_227851_G	59.35
206331_225761_G	111.34	225572_213701_G	95.24
206491_225621_G	76.45	225602_225562_G	48.66
206532_229432_G	57.18	225621_229681_G	76.94
206542_206532_G	62.08	225631_206491_G	200.31
206552_206542_G	33.17	225641_225631_G	161.11
206562_206552_G	29.57	225652_225641_G	65.36
206572_206562_G	50.25	225661_225652_G	112.44
206582_206572_G	43.17	225671_225661_G	128.59
206592_210352_G	51.45	225681_225671_G	102.39
206602_206592_G	60.20	225691_225681_G	69.35
206612_206602_G	53.17	225711_261611_G	93.24
206622_206612_G	61.56	225721_225711_G	53.21
206631_209331_G	38.64	225731_208552_G	58.12
206801_210362_G	49.38	225741_225731_G	102.97
206811_208072_G	70.63	225751_225741_G	76.26

207021_206811_G	55.54	225761_227461_G	77.09
207031_225572_G	50.63	227333_225751_G	88.57
208072_206801_G	63.50	227351_227333_G	78.77
208311_206331_G	64.46	227451_206042_G	52.30
208552_225721_G	70.92	227461_211841_G	27.63
208671_206031_G	52.29	227815_260132_G	68.04
209331_206622_G	38.47	227822_228162_G	68.44
209672_227842_G	96.28	227832_227822_G	116.44
210352_213521_G	48.77	227842_227832_G	38.82
210362_206631_G	63.15	227851_207031_G	55.95
211841_208671_G	25.10	228162_227815_G	60.83
212841_208311_G	38.39	229432_213481_G	34.00
212851_285551_G	20.57	229681_266891_G	91.13
213272_212851_G	126.00	260132_212841_G	53.82
213281_207021_G	80.26	261611_225691_G	73.82
213481_263331_G	45.39	263331_209672_G	57.39
213521_206582_G	37.41	266891_225602_G	77.17
213701_408031_G	104.02	285551_213281_G	91.99

Table 4.4. Selected prediction model result for one journey

Bus Stop Interval	Order	Average Travel Time	Bus Stop Interval	Order	Average Travel Time
213281_207021_G	7	77,81	227842_227832_G	31	32,98
207021_206811_G	8	55,02	227832_227822_G	32	98,57
206811_208072_G	9	70,4	227822_228162_G	33	59,64
208072_206801_G	10	60,42	228162_227815_G	34	57,48
206801_210362_G	11	45,44	227815_260132_G	35	46,42
210362_206631_G	12	62,1	260132_212841_G	36	44,59
206631_209331_G	13	39	212841_208311_G	37	34,98
209331_206622_G	14	38,17	208311_206331_G	38	53,73
206622_206612_G	15	59,81	206331_225761_G	39	97,25
206612_206602_G	16	53,38	225761_227461_G	40	70,77
206602_206592_G	17	61,61	227461_211841_G	41	24,13
206592_210352_G	18	48,16	211841_208671_G	42	21,83
210352_213521_G	19	48,68	208671_206031_G	43	43,67
213521_206582_G	20	36,53	206031_227451_G	44	30,6
206582_206572_G	21	41,79	227451_206042_G	45	44,14
206572_206562_G	22	47,23	206042_224391_G	46	69,58
206562_206552_G	23	29,72	224391_227351_G	47	61,18
206552_206542_G	24	21,48	227351_227333_G	48	61,96
206542_206532_G	25	57,7	227333_225751_G	49	66,69
206532_229432_G	26	53,39	225751_225741_G	50	63,71
229432_213481_G	27	32,41	225741_225731_G	51	85,32
213481_263331_G	28	44,48	225731_208552_G	52	44,83

263331_209672_G	29	47,33	208552_225721_G	53	52,14
209672_227842_G	30	88,05	225721_225711_G	54	39,08

Then a prediction result was selected. As seen on Table 4.4, the prediction program chose a passenger who was waiting at 54th bus stop and the bus was at 7th bus stop. Every journey from one bus stop to another was calculated one by one by prediction algorithm and results were recorded.

Then from the real data, a record was found with the same day of year, weather condition, day of week and minute of day parameters. These three records were compared as seen on Figure 4.2.

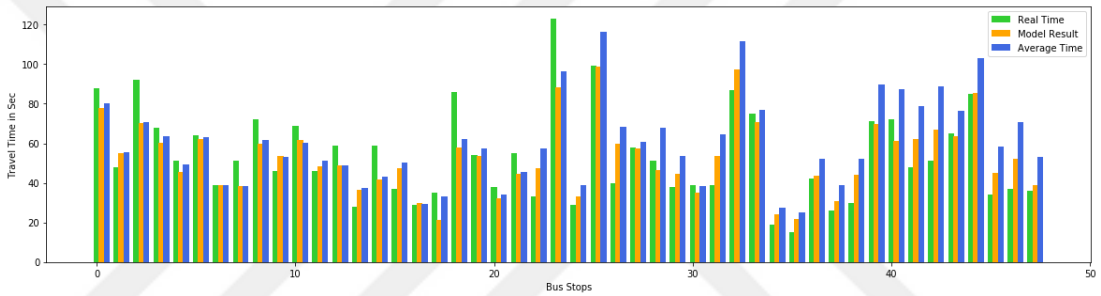

Figure 4.2. Comparison of results for 16C; Real Travel Time, Average Travel Time and Predicted Travel Time (Model Result) for 47 bus stop intervals

The prediction model showed 8.85 Mean Absolute Error (MAE) for the selected journey while average model showed 13.45 MAE as seen on Figure 4.3.

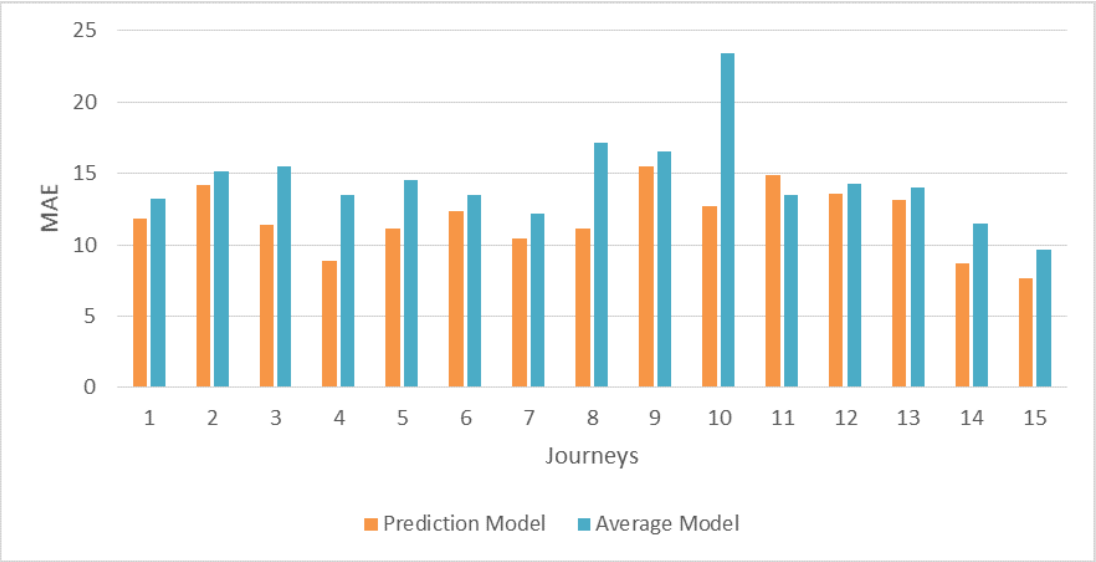

Figure 4.3. MAE of Prediction Model and Average Model for 15 journeys on different days including sunny and rainy weather, short and long journeys, peak and non-peak hours.

Same approach was tested for 15 random days under different weather conditions, with different number of intervals and for both peak and non-peak hours. For example, journey 8, 9, 10 and 15 in Figure 4.3 were generated on rainy days while others were generated when there was no precipitation. The prediction model performed better with smaller error rate, under tested conditions for most of the journeys.

5. CONCLUSION

5.1. Conclusions

This work answered the research question "How can travel time of public transportation buses be predicted in Istanbul?" The work was started with a compelling preparation step. After choosing the proper algorithm, parameters to be used in the work were selected carefully for the city. Although there are many factors effecting the travel flow, only day of year, day of week, time of day and weather condition were considered in order to reduce model complexity. Then by considering most common line types, seven lines were selected. The lines had extensive amount of travel data. Next, a field observation was conducted on a selected line. The observation was done by travelling 18 times by buses and interviewing drivers for a survey. The data logged during the field observation was used to see effects of the selected factors on the line and it was compared to real data during the model development. The drivers' responds to survey questions changed our vision about the effect of rainy weather on traffic flow. The survey also confirmed real time data analysis. A long data operations step was then initiated. 2019 and 2020 data was extracted for six lines; 11ÜS, 14, 16C, 17, 18K and 252. 2019 data was cleaned by using z-score table and data of both years was processed to have a useful result set for the model development.

A machine learning algorithm was used to build the prediction model. Multiple Linear Regression was chosen to maintain a solution for Istanbul's extraordinary traffic pattern. The model was trained with 2019 travel data of 6 lines. Then a validation model was proposed by using 2020 data of the selected lines. The validation model demonstrated more than 10.000 predictions for each line. Performance of the model was measured in two steps. First, error rate of prediction was calculated for each line and it ranged between 0.016 and 0.153. Then a Historic Average model was developed and predicted results were compared to Historic Average model result and real travel time for 15 journeys taken on different days. The proposed model outperformed the average travel time for each bus stop interval for different

journeys. Results show that the model is a prominent solution to predict travel time in public transportation in Istanbul.

5.2. Further Work

This work proposes a prediction system by using some specific parameters. A further improvement could be adding some other factors such as bus type and driver's speed habit. Moreover, in this work, on training step, travel times from one bus stop to another was used. This distance was called an interval. In the future, interval can be divided into sections in order to obtain the locations with longest travel time. Such improvement can even help to spot any structural problems that cause traffic jam and to maintain a solution.

REFERENCES

Achar, A., Bharathi, D., Kumar, B. A., & Vanajakshi, L., 2020. Bus Arrival Time Prediction: A Spatial Kalman Filter Approach. IEEE Transactions on Intelligent Transportation Systems, 1298-1307.

Chang, C., & Lin, C., 2001. LIBSVM : a library for support vector machines. Date Of Access: 01.01.2021. http://www.csie.ntu.edu.tw/~cjlin/libsvm

Cheng, S., Liu, B., & Zhai, B., 2010. Bus Arrival Time Prediction Model Based on APC Data. School of Transportation Science and Engineering, Harbin Institute of Technology.

Chien, J., & Ding, Y., 2002. Dynamic bus arrival time prediction with Artificial Neural Networks. Journal of Transportation Engineering, 128(5), 429-438.

Choudhary, R., Khamparia, A., & Gahier, A. K., 2016. Real time prediction of bus arrival time: A review. 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), 25-29. Dehradun.

Deng, L., He, Z., & Zhong, R., 2013. The Bus Travel Time Prediction Based On Bayesian Networks. International Conference on Information Technology and Applications.

IETT, 2020. Hat - Hareket Saati, Hat Güzergahı. Date Of Access: 01.01.2021. https://iett.istanbul/tr/main/hatlar/90/DRAMAN%20-%20EM%C4%B0N%C3%96N%C3%9C-%C4%B0ETT-Otob%C3%BCs-Sefer-Saatleri-ve-Duraklar%C4%B1

Garcia, F. C., & Retamar, A. E., 2016. Towards building a bus travel time prediction model for Metro Manila. 2016 IEEE Region 10 Conference (TENCON), 3805-3808. Singapore.

Hapsari, I., Surjandari, I., & Komarudin., 2018. Visiting Time Prediction Using Machine Learning Regression Algorithm. 2018 6th International Conference on Information and Communication Technology (ICoICT), 495-500. Bandung.

He, P., Jiang, G., Lam, S., & Tang, D., 2019. Travel-Time Prediction of Bus Journey With Multiple Bus Trips. IEEE Transactions on Intelligent Transportation Systems, 4192-4205.

Kwak, S., & Geroliminis, N., 2020. Travel Time Prediction for Congested Freeways With a Dynamic Linear Model. IEEE Transactions on Intelligent Transportation Systems.

Li, J., Gao, J., Yang , Y., & Wei, H., 2017. Bus arrival time prediction based on mixed model. China Communications, 14(5), 38-47.

Li, X., & Bai, R., 2016. Freight Vehicle Travel Time Prediction Using Gradient Boosting Regression Tree. 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), 1010-1015. Anaheim, CA.

Lin, D., Tsao, W., Yu, C., Liu, H., & Chang, Y., 2019. The Travel Time Prediction by Machine Learning Methods with Traffic Data in Chiayi City, Taiwan. 2019 4th International Conference on Electromechanical Control Technology and Transportation (ICECTT), 257-260. Guilin, China: ICECTT.

Lin, W., & Zeng, J., 1999. An experimental study of real-time bus arrival time prediction with GPS data. Journal of the Transportation Research Board, 101-109.

Liu, H., Zhang, K., He , R., & Li, J., 2009. A neural network model for travel time prediction. 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, 752-756. Shanghai.

Meng, Z., Wang, C., Peng, L., Teng , A., & Qiu, T. Z., 2017. Link travel time and delay estimation using transit AVL data. 2017 4th International Conference on Transportation Information and Safety (ICTIS), 67-72. Banff.

O'Sullivan, A., Pereira, F. C., Zhao , J., & Koutsopoulos, H. N., 2016. Uncertainty in Bus Arrival Time Predictions: Treating Heteroscedasticity With a Metamodel Approach. IEEE Transactions on Intelligent Transportation Systems, 3286-3296.

Pan, J., Dai, X., Xu, X., & Li, Y., 2012. A Self-learning algorithm for predicting bus arrival time based on historical data model. 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, 1112-1116. Hangzhou.

Portugal, I., Alencar, P., & Cowan, D., 2015. The Use of Machine Learning Algorithms in Recommender Systems: A Systematic Review. Expert Systems with Applications, 97.

Prokhorchuk, A., Dauwels, J., & Jaillet, P., 2020. Estimating Travel Time Distributions by Bayesian Network Inference. IEEE Transactions on Intelligent Transportation Systems, 1867-1876.

Ray, S., 2019. A Quick Review of Machine Learning Algorithms. 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), 35-39. Faridabad, India.

Reddy, K. K., Kumar, B. A., & Vanajakshi, L., 2016. Bus travel time prediction under high variability conditions. Current Science (00113891).

Rice, J., & van Zwet, E., 2001. A simple and effective method for predicting travel times on freeways. ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585), 227-232. Oakland, CA.

Schölkopf, B., Smola, J., Williamson, C., & Bartlett, L., 2000. New support vector algorithms. J. Neural Computation, No. 12, 1207-1245 .

Tan, C., Park, S., Liu, H., Xu, Q., & Lau, P., 2008. Prediction of Transit Vehicle Arrival Time for Signal Priority Control: Algorithm and Performance. IEEE Transactions on Intelligent Transportation Systems, 688-696.

Turchenko, V., & Demchuk, V., 2006. Neural-Based Vehicle Travel Time Prediction Noised by Different Influence Factors. 2006 International Conference - Modern Problems of Radio Engineering, Telecommunications, and Computer Science, 195-198.

Using the Z Table, 2020. Date Of Access: 01.09.2020. https://www.superprof.co.uk/resources/academic/maths/probability/normal-distribution/using-the-z-table.html

Üniversite Şehri İstanbul., 2020. Istanbul Valiliği. Date Of Access: 01.01.2021. http://www.istanbul.gov.tr/universite-sehri-istanbul

Vinagre Díaz, J. J., Rodríguez González, A. B., & Wilby, M. R., 2016. Bluetooth Traffic Monitoring Systems for Travel Time Estimation on Freeways. IEEE Transactions on Intelligent Transportation Systems, 123-132.

Wang, J., Chen, X., & Guo, S., 2009. Bus Travel Time Prediction Model with ν – Support Vector Regression., (p. 12th International IEEE Conference on Intelligent Transportation Systems). St. Louis, MO, USA.

Weather archive in Istanbul (airport), METAR., 2020. Weather for 243 countries of the world. Date Of Access : 01.06.2020. https://rp5.ru/Weather_archive_in_Istanbul_(airport),_METAR

Wu, C., Ho, J., & Lee, D., 2004. Travel-time prediction with support vector regression. IEEE Transactions on Intelligent Transportation Systems 5(4), 276 - 281.

Yang, J.-S., 2005. Travel time prediction using the GPS test vehicle and Kalman filtering techniques. Proceedings of the 2005, American Control Conference, 3, 2128-2133. Portland, USA.

Yildirimoglu, M., & Geroliminis, N., 2012. Experienced travel time prediction for freeway systems. 15th International IEEE Conference on Intelligent Transportation Systems.

Yu, H., Wu, Z., Chen, D., & Ma, X., 2017. Probabilistic Prediction of Bus Headway Using Relevance Vector Machine Regression. IEEE Transactions on Intelligent Transportation Systems, 1772-1781.

Yu, H., Xiao, R., Du, Y., & He, Z., 2013. A Bus-Arrival Time Prediction Model Based on Historical Traffic Patterns. International Conference on Computer Sciences and Applications.

Z Table, 2020. Z Table. Date Of Access: 01.09.2020. https://www.ztable.net/

Zhang, Z., Li, Y., Li, L., Li, Z., & Liu, S., 2019. Multiple Linear Regression for High Efficiency Video Intra Coding. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1832-1836. Brighton, United Kingdom.

Zhou, P., Zheng, Y., & Li, M., 2014. How Long to Wait? Predicting Bus Arrival Time With Mobile Phone Based Participatory Sensing. IEEE Transactions on Mobile Computing, 1228-1241.

Zhu, T., Kong, X., Lv, W., Zhang, Y., & Du, B., 2010. Travel Time Prediction for Float Car System Based on Time Series. 2010 The 12th International Conference on Advanced Communication Technology (ICACT). Phoenix Park, South Korea.

Zhu, T., Ma, F., Ma, T., & Li, C., 2011. The Prediction of Bus Arrival Time Using Global Positioning System Data And Dynamic Traffic Information. 4th Joint IFIP Wireless and Mobile Networking Conference .

RESUME

Name Surname : Betül BOYLU

Foreign Language : English

E-mail :

Education

High School : Pendik Imam Hatip High School, 2000

Bachelor Degree : European University of Lefke, Faculty of Engineering, Department of Computer Engineering

Post Graduate : Istanbul Commerce University, Graduate School of Natural and Applied Sciences, Department of Computer Engineering

Work Experience

European University of Lefke	2004 - 2005
Astra Computer	2005 - 2006
Eastern Corner	2006 - 2008
Teksan Generator	2008 - 2010
IETT General Management	2010-… (Continues)

Publications

Boylu, B., Boyacı, A., 2021. Travel Time Prediction In Public Transportation. Istanbul Commerce University Journal of Technologies and Applied Sciences.