PUBLIC BUS ARRIVAL TIME PREDICTION USING MACHINE LEARNING: IN CASE OF ADDIS ABABA

A MASTER'S THESIS

BY

HABTU REDA

DEPARTMENT OF SOFTWARE ENGINEERING COLLEGE OF ELECTRICAL AND MECHANICAL ENGINEERING

ADDIS ABABA SCIENCE AND TECHNOLOGY UNIVERSITY

October, 2021

PUBLIC BUS ARRIVAL TIME PREDICTION USING MACHINE LEARNING: IN CASE OF ADDIS ABABA

By

HABTU REDA

A Thesis Submitted as a Partial Fulfillment to the Requirements for the Award of the Degree of Master of Science in Software Engineering

to

DEPARTMENT OF SOFTWARE ENGINEERING

ADDIS ABABA SCIENCE AND TECHNOLOGY UNIVERSITY

October, 2021

Declaration

I hereby declare that this thesis entitled "**Public Bus Arrival Time Prediction Using Machine Learning: In Case of Addis Ababa**" was prepared by me, with the guidance of my advisor. The work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted, in whole or in part, for any other degree or professional qualification.

Author: Habtu Reda

Signature, Date: 16/10/21

Witnessed by:

Name of student advisor: Beakal Gizachew (PhD)

Signature, Date: 18-NOV-2021

Name of student co-advisor: Tarikwa Tesfa (Msc)

Signature, Date:

iii

Certification

This is to certify that the thesis prepared by **Mr. Habtu Reda Behrhe** entitled **"Public Bus Arrival Time Prediction Using Machine Learning: In Case of Addis Ababa"** and submitted as partial fulfillment for the Degree of Master of Science complies with the regulations of the University and meets the accepted standards concerning originality, content, and quality.

Signed by Examining Board:

External Examiner: Ketema Ader (phD) Signature, Date: 17/11/21

Internal Examiner: Dr. Hussien Seid Signature, Date:

Chairperson: Tulu Tilahun (PhD) Signature, Date:

DGC Chairperson: Felix Edesa Signature, Date:

College Dean/Associate Dean for GP: Muluneh Mekonnen Tulu (PhD) Associate Dean for College of Electrical and Mechanical Engineering Signature, Date: 22/11/21

College of Electrical & Mechanical Engineering
Science & Technology University

Dedication

This research work is dedicated to my wife, Abadit Aberha, for her continuous support and encouragement.

ABSTRACT

Estimating public bus arrival times and delivering accurate arrival time information to passengers are critical for making public transportation more user-friendly and thereby increasing its competitiveness among various forms of transportation. However public bus arrival time prediction remains major bottlenecks With traffic heterogeneity in composition and diversity of vehicles, as well as a big pedestrian population combined with inadequate lane use, predicting the arrival time of public buses at stations is a severe concern.. The main objective of this study is to apply machine learning algorithms to predict bus arrival time. The data was collected from Addis Ababa Sheger Public Bus Transport. Random Forest, Gradient Boosting, Artificial Neural Network, K-Nearest Neighbors and Support Vector Machine algorithms are applied to build the models and to compare and choose the best model to predict the bus arrival time. After selecting the features and algorithms, different data preprocessing tasks like checking outliers, missing values and data reduction are done. Finally, 140,000 instances of dataset are used to train and build the model. The prepared dataset is partitioned into 90% training and 10% testing set. Beginning Date, Beginning Time, End Date, Time Range, Mileage, Duration, Initial latitude, Initial longitude, Final latitude, Final longitude, and End Time were used as input features for developing the model. Based on the experiment result the Random Forest algorithm achieved a better performance with R-squared score of 0.994, MAE of 0.812, RMSE of 3.780 and MSE of 14.28.

Keyword: Bus Arrival Time, Intelligent Transportation, Machine Learning, Random Forest, Artificial Neural Network

Acknowledgment

I would love to express my gratitude to a few people who have been kind, motivating, and encouraging during this thesis and in the years proceeding up to this point.

First and foremost, I want to express my gratitude to my supervisor Dr. Beakal Gizachew Thank you for allowing me to do my thesis and for taking the time to share your vast knowledge. Thank you to Tarikwa Tesfa for giving the thesis foundation and inspiration. Thank you to the entire TransLink team for their encouragement and open-handed for research data I'd want to express my gratitude to all of my friends and classmates for their effort and contributions to the study's success, especially my friend Adem Guluma. Finally, I'd want to thank Naeb Yeman for always being a support system for me.

Table of Contents

Abbreviations and Acronyms

AdaBoost	Adaptive Boosting
AVL	Automatic Vehicle Locater
ANN	Artificial Neural Network
APTS	Advanced Public Transit Systems
ATIS	Advanced Traveler Information Systems
ATMS	Advanced Traffic Management Systems
AVCS	Advanced Vehicle Control Systems
CSV	Comma Separated
CVO	Commercial Vehicle Operations
DT	Decision Tree
GUI	Graphical User Interface
ITS	Intelligent transport system
KNN	K nearest Neighbor
LR	Logistic Regression
ML	Machine Learning
MSE	Mean Squared Error
NB	Naive Bayes
RF	Random Forest

List of Tables

List of Figures

Chapter 1: Introduction

1.1. Background

For a long time, public transportation services like buses and railroads relied only on static timetables to plan their trips. Passengers and operators had no means of knowing if their bus was running late, early, or on schedule. In the 1960s, the first vehicle tracking systems were trailed in Germany and the United States of America [1].

Early systems relied on sensors placed along the route that the bus could detect, allowing it to supply the service provider with real-time location information [2] [3].

Modernization of bus operations and management to focus on supply efficiency and effectiveness in satisfying user needs will be a critical aspect of any strategy to improve public transportation services and to use public transportation as a tool for the City's green growth-oriented initiatives.

Intelligent transportation systems (ITS) are transportation services and technology that aim to improve the efficiency, safety, dependability, and environmental sustainability of transportation systems while avoiding the construction of new infrastructure. Advanced Traffic Management Systems (ATMS), Advanced Vehicle Control Systems (AVCS), Advanced Public Transit Systems (APTS), Commercial Vehicle Operations (CVO) and Advanced Traveler Information Systems (ATIS) are only a few of the ITS subsystems [4].

1.2. Public Transport in Addis Ababa

The capital city of Ethiopia, Addis Ababa, is a thriving and quickly changing urban metropolis with a population of about 4.42 million people.

The City's relatively rapid economic boom and population explosion in recent years have brought significant challenges and pressures to its socioeconomic infrastructure, particularly its transportation network. Addis Ababa is home to several international organizations, notably the African Union and the United Nations Economic Commission for Africa. Several international conferences are held in the city on a regular basis. Addis Ababa is frequently referred to be Africa's political capital. [5] [6].

The city has a low level of mobility, with an estimated per capita trip rate of 1.08, with 60% of excursions made on foot. Nearly half of these excursions are for educational purposes, which correspond to the average age of local residents (80 percent below 40). The average travel length is predicted to be 3.3 kilometers, with walk trips being 1.49 kilometers [7].Even by Sub-Sahara African standards; Addis Ababa has a small number of registered motor cars. The number of registered vehicles in Addis Ababa was 630,440 by the middle of 2020 [8]. Addis Ababa is also reported to have more than 70% of the country's registered vehicles. Despite the city's enormous expansion of its road and highway networks in recent years, frequent and long traffic congestion has become the cities daily experience and face. As a result, the City's traffic congestion, traffic accidents, and greenhouse gas emissions have increased and are continuing to rise.

However, the sector has been unable to match the increased demand. As a result, additional ways to control this demand are required. Adoption of Intelligent Transportation Systems (ITS) is a promising strategy.

Hence, this research is aimed at developing machine learning models to predict bus arrival times of buses in in Addis Ababa, using (AVL) data. The models will enable public transit operators to provide more accurate information to passengers to improve reliability and consequently increase bus ridership.

Figure 1.1 Sheger public bus transport

1.3. Motivation

Bus arrival time prediction in public transportation systems is a direct measure of their efficiency and usefulness. Arrival time information is also important in planning operations and route assignments.

The design and implementation of ITS solutions rely on accurate arrival forecasts based on current travel time data. The composition of traffic in Addis Ababa is heterogeneous and urban road is shared by a wide range of vehicles, including two, three, and four-wheelers, as well as a considerable pedestrian population. This heterogeneity, along with a lack of lane discipline, makes trip time prediction more difficult than convex models can manage. Machine learning is an emerging technique and technology that holds the promise of making things easy, suitable and process huge amounts of datasets for organizations and industries. Machine learning aims at developing algorithms that can learn and create statistical models for data analysis and prediction. The ML algorithms should be able to learn by themselves based on data provided and make accurate prediction bus arrival time, without being specifically programmed for a given task [9].

Statement of the Problem

Bus arrival prediction is a direct indicator of a public transportation system's efficiency and usability. Arrival time information is also useful for operations planning and route assignments. Transportation, particularly bus transportation, is subjected to many problems in Addis Ababa city. In particular, the arrival time of buses at stations cannot be predicted accurately. And this serious problem is not a well-researched problem.

In addition to this, researchers conducted elsewhere couldn’t be directly applied in Addis Ababa city because of the exceptional nature of transportation problems such as the traffic heterogeneity in composition and variety of vehicles and a large pedestrian population coupled with poor lane utilization. Also, the transportation management system of Addis Ababa city involves disintegrated manual systems which resulted in poor delivery of transportation services.

To overcome these problems, machine learning algorithms together with real-time GPS data to provide a better solution to the existing problem. Hence, this thesis machine learning models to solve bus arrival prediction problem in Addis Ababa city.

1.4. Research Question

In this study, a machine learning algorithms is used in a knowledge discovery process to predict bus arrival prediction. Therefore, this research study addresses the following research questions:

RQ1: What are the key factors or feature that helps to develop and predict bus arrival time prediction model?

RQ2: Which machine learning algorithm performs better?

1.5. Objectives of the Study

1.5.1. General Objective

The general objective of the study is to develop a model that predicts bus arrival time by using machine learning regression techniques.

1.5.2. Specific Objectives

The specific objective that needs to be carried out to achieve the above general objective are:

- Conduct a literature review on the different machine learning techniques and the applicability of the algorithms to GPS dataset.
- Training and testing the models that will be developed.
- Build bus arrival predication models using the selected algorithms.
- Evaluate the performance of the trained model for comparison to select the best model

1.6. Scope and Limitations

This research work focuses on the development of model that can predict bus arrival time. The model is designed by analyzing the data collected from Trans-link which is GPS tacking company. The activities include preparation of dataset, selection of appropriate algorithms, feature selection, and regression approaches on the GPS dataset of 20 Sheger buses. Due to lack of well-structured data and time to gather road traffic congestion and also weather information; as a result, this study used only derived data features from historically obtained GPS datasets to predict bus arrival time.

1.7. Significance of the Research

Bus arrival prediction model is believed to have the ability to make a major contribution to transport operators' and road and transport authorities. Bus arrival time estimation could aid transportation operators in offline management by allowing them to optimize their transportation schedules and gain a better grasp of their systems' resilience. And also predict bus arrival time could be used to evaluate the transportation system effectiveness on the other hand, might be a significant part of a real-time traveler information system, and it could also help passengers minimize anxiety and stress by guiding them to bus routes with the least amount of waiting time.

The study for predict bus arrival time could provide an estimate of how long the buses will be on the road and when they will arrive. This information supports passengers in determining the best mode of transportation and route, particularly in congested areas, by providing real-time information on various travel options. Furthermore, predicting bus arrival times at station could make public transportation more efficient.

In addition, the Addis Ababa Transport Office can also utilize it to take proper and appropriate action against public transportation, such as evaluating existing norms and regulations. Citizens, non-governmental organizations and media will be beneficiary from study.

The facts and results of the study could suggest transport authorities and operators to develop and evaluate public transport. It's also used to improve all aspects of the public transport control system by building public policies and strategies.

1.8. Organization of the Research Report

This research work report is divided into five chapters, each of which addresses a different aspect of the research. The first chapter covers the introduction and context of the study, as well as the problem statement, study objectives, scope, constraints, and importance. The available literatures and associated work on bus arrival time prediction are briefly described. The study's methodology, as well as data preparation mechanisms such as data collection, comprehending GPS datasets, data formatting, data preprocessing, and feature selection, are all described in Chapter 3. The outcomes of a complete data analysis presented and discussed in Chapter 2, and finally in chapter 5 conclusion and recommendation.

Chapter 2: Literature Review

2.1. Overview of Bus Arrival Prediction

Over the years, the public transportation system has gone through numerous phases of renovation in order to better serve the public's needs while remaining profitable. The AVL system has been introduced as part of the improvements in Intelligent Transport Systems as part of numerous research initiatives to improve transportation. This enables the collecting of real-time data at bus stops and aboard buses, which can then be shared with riders and transit management. Several bus arrival prediction models have been built using various methodologies based on the literature. [8].

This section mainly focuses on the background information and review of literature of the domain of this thesis. It includes a detail explanation about bus arrival time prediction and different machine learning algorithms and related works. Finally, the section is concluded with the summaries of related works and the main gaps which should be solved in this thesis.

2.2. Machine Learning

Machine learning (ML) is a field of Artificial Intelligence that is used to train and teach machines how to handle and manage data more efficiently by recognizing patterns and correlations between data characteristics automatically. ML uses statistical tools and algorithms to learn from data and extract information. It can be used to extract important and previously unknown patterns and information from data that is difficult to comprehend.

Training/testing and prediction are the two main tasks that ML is used for most of the time. Machine learning's main goal is to create a model that can accurately predict or classify target outputs based on testing previously unseen or unknown data [9]

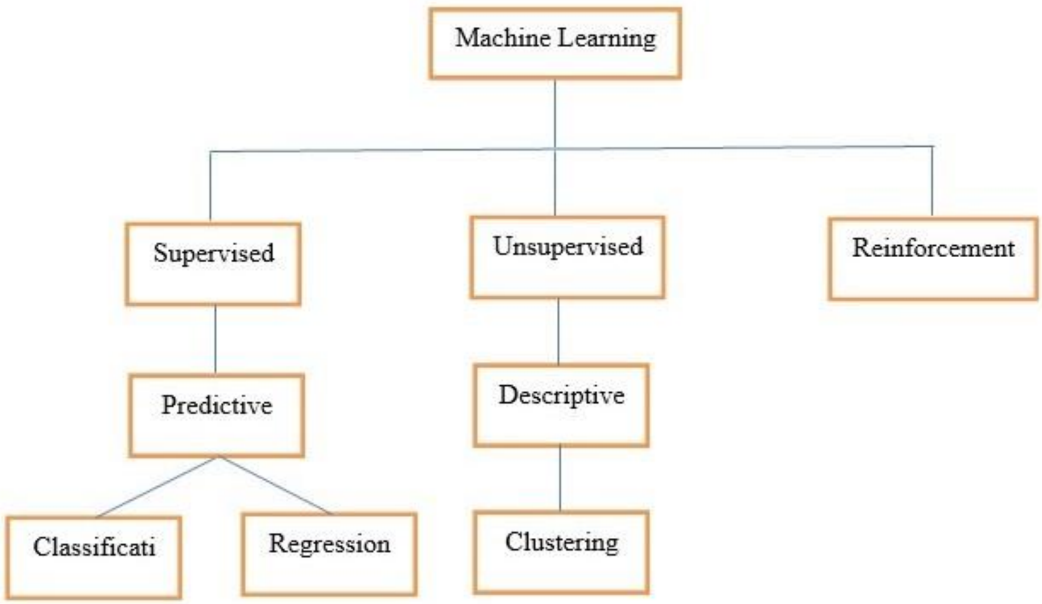

Figure 0.1: Machine Learning Prediction Algorithms

2.2.1. Supervised Machine Learning

It is the task of applying machine learning to infer a function from labeled training data. The training data is made up of a sequence of training examples. Each example in supervised learning is made up of a pair of descriptions, labels, and target outputs, and it employs labeled data. The datasets are divided into two sections by Supervise ML. There are two types of sets: training and testing. The training set contains characteristics that will be predicted and utilized to train the model. The performance of the model will then be tested by the test set, which consists of previously unseen or untrained data to the model, following training or learning. When a model is trained using a training dataset, it learns patterns that will aid it in labeling fresh data [10].

The dataset contains features, which are separated into dependent, or target features that the model must predict and independent features that assist the model in making predictions about the target output. By mapping and determining the link between the dependent and independent properties, the algorithm generates various functions. Model training is when the mapping is done. The training operations will continue until the model is well-trained or the performance has been tuned. There are two types of supervised machine learning approaches. The first is a classification method that has a discrete value as its target output. It's usually used to sort dependent variables into target groups. The other is a regression method, which has a continuous value as its target output. It maps the independent features to the dependent feature to forecast the quantities of a particular problem [11].

2.2.1.1. Classification

Classification is a form of supervised machine learning method with a discrete value target output. Its most commonly used to sort dependent variables into target classes. The dependent variables in binary classification issues have just two classes to predict, but the dependent variables in multi-class classification issues have more than two classes to be classed or predicted [12].

2.2.1.2. Regression

Another sort of supervised algorithm is a regression algorithm, which has a continuous value as its target output. It maps the independent features to the dependent feature to forecast the quantities of a specific problem [13].

It's typically used to forecast pricing and time spent on a specific activity. The supervised learning regression approach was employed in this investigation. This sort of algorithm aids in the construction of the most accurate and efficient model since the learning data comes with labels or intended outputs, and the goal is to establish a general method of mapping input to output. It requires creating a machine learning model based on labeled data with a continuous output as the goal value.

2.2.2. Unsupervised Learning

Unsupervised learning is a type of machine learning that finds patterns in data without the use of supervision. There are no intended outputs or outcomes to forecast in this algorithm. It is commonly used to group or cluster features without any prior knowledge based on similarities and patterns [14]. There is no need to partition the dataset into training and testing for this methodology to train and test the model. As a result, the computer should learn and discover patterns among the data features on its own, clustering or associating them into groups of similarity. It uses unlabeled data to uncover hidden patterns. As a result, the unsupervised learning algorithm seeks to efficiently group a given dataset into a particular number of clusters, and it is a very strong tool for studying and discovering hidden patterns and trends in datasets [15].

2.3. Machine Learning Algorithms

The following subsections provide a detailed discussion of various machine learning algorithms that are extensively used and regularly applied to practically any data problem are presented in the following subsections.

2.3.1. Logistic Regression (LR)

Linear regression is a machine learning regression approach that shows the link between data features to approximate based on continuous variables the dependent feature is the data characteristic that needs to be predicted, and the independent features are the features that are utilized to predict the continuous values of the dependent features. By fitting the optimal line, the straight-line method assesses the relationships between the dependent and independent features [16].

2.3.2. Decision Tree (DT)

Decision trees are another supervised machine learning algorithm that uses hierarchical and statistical models to classify the dataset into various classes. The regression and classification models created by DT have tree structure [14].

Statistical measures were used to select the root node attributes and experimentation on the data. Then, recursively going through other attributes until no attributes remains the end node or leaf node. Entropy and information gain used to get the statistical measure for building the tree while constructing the algorithm’s tree structure. The entropy of the attributes calculated by Equation 2.1 after the data split additionally, a decision tree is used to create models that predict the value of dependent features having information about the simple decision rules which inferred from the data features [15].

$$H(s) = -\sum P(x) log_2 P(x) \qquad (0.1)$$

Where,

- H(S): Entropy of the dataset

- S: The current dataset for which entropy is being calculated
- x: Set of classification in S
- P(x): The probability of x

Information gain is a statistical measure which helps to select a decision and root node. Attributes that are having higher information gain are taken as a root node. Then the hierarchy of the tree will be built by calculating the information gain of the other attributes recursively. The information gain of the attribute is calculated using Equation 2.2.

$$IG(A,S) = H(S) - H_A(S) \quad (0.2)$$

Where,

- IG (A, S): Information gain of a specific attribute 'A'
- H(S): Entropy of the dataset
- H_A (S): Entropy of the specific attribute in the given dataset.

2.3.3. Support Vector Machines (SVM)

A support vector machine is a popular supervised machine learning technique that splits data into different categories using hyperplane [17] [18]. SVM divides n-dimensional space into classes by creating a decision boundary or line known as a hyperplane. The new data points are then simply assigned to the appropriate classifications. Support vectors are points that the SVM selects to form the hyperplane. Its mission is to select and locate the ideal hyperplane. The distance between the vectors and the hyperplane is called the margin. As a result, SVM is utilized to optimize this margin or distance. The hyperplane with the highest points of margin is the optimal hyperplane [13].

The SVM employs the input weight and the bias values on the training set to separate the two classes. Examples of class one are isolated to one side of the hyperplane based on the value and result of S(x), whereas instances of class two are isolated to the other side of the hyperplane *[19]*. As demonstrated in Equation 2.4, a kernel trick is used to generate a smooth separating nonlinear decision boundary for nonlinearly separable datasets, resulting in a smooth separating nonlinear decision border *[20]*.

$$S(x) = w^T x + b \quad (0.3)$$

$$K(x_i, x_j) = e^{-(||xi - xj||2 / \sigma 2)} \quad (0.4)$$

1.1.1 Artificial Neural Network

The Artificial Neural Network (ANN) is a mathematical function designed to mimic the basic function of a biological neuron. It's been used in a range of applications, including data filtering, prediction, and input classification.

The network is trained using a back propagation algorithm in which the actual output is calculated in the forward pass and the weights between the output layer, hidden layer, and hidden layer and input layer are adjusted in the backward path, then the steps of this algorithm are repeated until the error is reduced, and the significance of the sigmoid transfer function is also discussed in detail. [13].

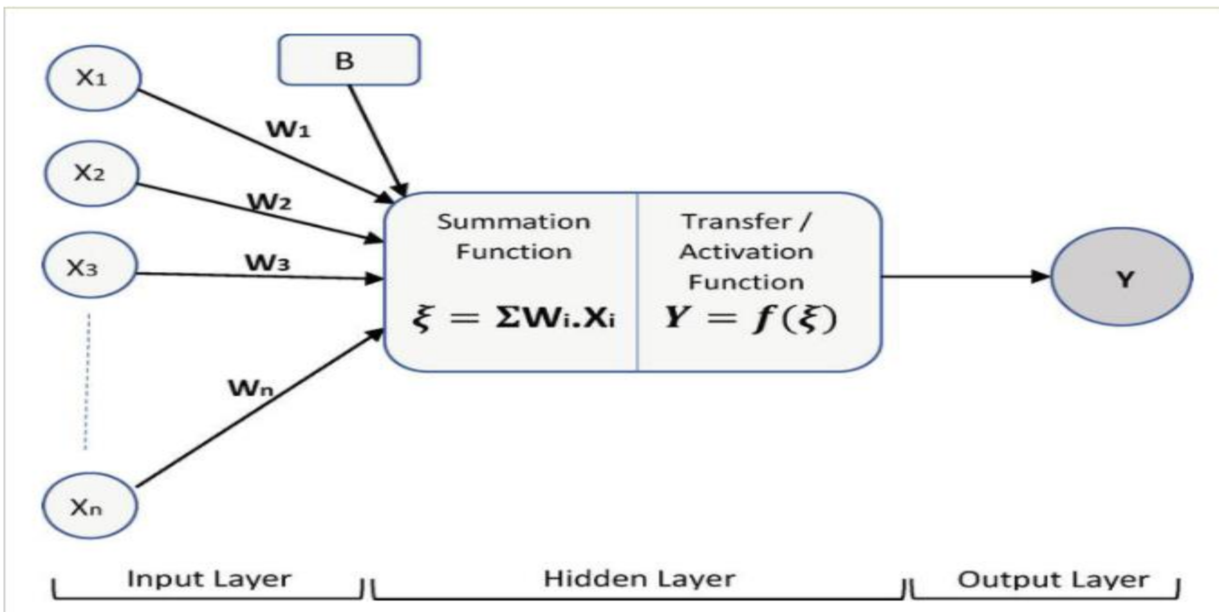

Figure 0.2 General model of ANN [13]

2.3.4. Naive Bayes (NB)

The Bayes theorem [14] drives the Nave Bayes algorithm, which is a simple probabilistic supervised learning system. It's a probabilistic classifier that calculates a set of probabilities by computing the frequency of values in a dataset. It thinks all features are independent or unrelated, and it doesn't learn the link between them. In numerous categorization challenges, NB performs well and learns quickly. When utilizing Nave Bayes to categorize a class, two types of probabilities are used. Priori and posterior probability are the two types of probability. The following are the steps and formulas for calculating the Nave Bayes formula:

Step 1: Create a frequency table from the dataset.

Step 2: Find the probabilities and create a Likelihood table.

Step 3: Compute the posterior probability for each class using the Nave Bayesian formula, and the prediction is based on the class with the highest posterior probability.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{0.5}$$

Where,

• P(c|x) denotes the target class's posterior probability given a predictor characteristic.

• P(c) is the class prior probability.

• The probability of predictor given class is P(x|c).

• P(x) is the predictor's prior probability.

NB is a straightforward and quick approach that may be applied to both binary and multiclass classification problems.

2.3.5. Time Series Prediction (TSP)

Time series forecasting is a crucial aspect of machine learning that is sometimes overlooked. Because there are so many prediction issues with a temporal component, it's critical. These issues are overlooked since it is the time component of time series problems that makes them more difficult to solve [21].

The selection of a suitable probability model for the data is an important component of time series analysis. To account for the unpredictability of future observations, it's natural to assume that each observation xt is a realized value of a random variable Xt. A time series model for observed data xt is a specification of the joint of a sequence of random variables Xt, of which xt is a realization [22].

2.3.6. Random Forest (RF)

Arbitrary Timberland could be a stowing ensemble-learning computation based on a number of different choice tree expectation models [18]. Gathering is a strategy that combines various classifiers to create a single solid indicator that may be used to improve show performance. It appears as hastily actualized choice trees. The correlations and individual strength of the trees determine RF's generalization error. Both regression and classification applications can benefit from RF. The algorithm is broken down into four phases.

Step 1. Select a set of random data points from a dataset.

Step 2: For each data point, create a decision tree and receive a forecast result from each decision tree.

Step 3. Make a vote for each expected outcome.

Step 4. As a forecast outcome, use the highest number of votes

2.3.7. Lasso Regression (LR)

Lasso regression algorithm employs the "shrinkage" technique, in which the coefficients of determination are reduced to zero. The regression coefficients seen in the dataset are returned by linear regression. To avoid over fitting and make them operate better on diverse datasets, you can use the lasso regression to reduce or regularize these coefficients. This type of algorithm is used when the data shows high multicollinearity or when you want to automate variable elimination and feature selection [23].

$$\sum_{i=1}^{n}(y_i - \sum_{j} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \qquad (2.6)$$

This is equivalent to minimizing the sum of squares with constraint |Bj s (= summatio notation). Some of the s is reduced to zero, resulting in an easier-to-understand regression model.
The intensity of the L1 penalty is controlled by a tuning parameter. is the amount of shrinkage in terms of?

- When $\lambda = 0$, There are no parameters that are removed. The result is the same as the one obtained using linear regression.
- As λ increases, as time goes on, more and more coefficients are set to zero and removed. when $\lambda = \infty$, all coefficients are eliminated).

- As λ increases, bias increases.
- As λ decreases, variance increases.

2.3.8. K-Nearest Neighbors (KNN)

K-Nearest Neighbors is a supervised learning technique that retains all available examples during training and uses a majority vote of its k neighbors to classify a new data point category that is very comparable to the new data points [17] [18]. Because it makes no assumptions about the original data, KNN is a non-parametric and lazy learner algorithm. The distance between the K closest neighbors is calculated using the Euclidean, Manhattan, Minkowski, or Weighted distance functions. If K = 1, the example is simply assigned to the nearest neighbor's class. The following are the steps taken by the KNN:

Step 1: Determine the number of neighbors (k).

Step 2: Determine the distance between two points.

Step 3: Compiles a list of the k closest neighbors;

Step 4: Count the number of times each class appears with the shortest distance between them;

Step 5: Attends the class that has been mentioned numerous times;

Step 6: Sorts the new data into the same class as in step 5;

2.3.9. Gradient Boosting Algorithm (GBA)

Gradient Boosting is one of boosting ensemble learning algorithm that helps to build predictive models. It uses decision tree specifically regression tree as weak learners that output real values for splits. The trees are designed in a greedy manner to aid in the selection of the best split points based on purity scores. When adding the trees, it uses a gradient decent approach to minimize the loss function [35].

Gradient Boost has three main components.

- Loss Function: it estimates how best the model in making predictions with the given data and it varies depending on the type of the problem.
- Weak Learner: is one that categorizes the data so poorly when compared to random guessing. The weak learners are mostly decision tree.
- Additive Model: Adding the decision tree one step at a time in an iterative and sequential process. The value of the loss function should decrease with each iteration and fixed number of trees is introduced, or training is stopped until failure reaches an appropriate level or the external validation dataset no longer improves [24].

2.3.10. Linear Regression (LR)

Linear regression is a machine learning regression approach that shows the link between data characteristics to approximate based on a continuous variable(s) [18]. Dependent features are data features that need to be anticipated, whereas independent features are features that are utilized to forecast the continuous values of dependent features. The regression line is the best fit line and is represented by the following linear equation.

$$Y = A + Bx \tag{0.7}$$

Where:

• A, B: Constants that describe the line's y-axis intercept and slope.

• Y stands the predicted value

2.3.11. K-Means (KM)

Unsupervised machine learning algorithms such as K-Means clustering [17] are a sort of unsupervised machine learning algorithm. Using unlabeled data, this method is used to solve the clustering problem, which splits data into k number of clusters. Without training the algorithms, it iteratively groups each data point into one of the k clusters based on the specified attributes. The similarity of the features is used to classify each cluster. It employs a centroid and reduces the distance between data points as well as data grouping. The k-means algorithm is an iterative process that continues until the algorithm no longer finds the optimal clusters to yield a final output. It accepts k clusters as input, each of which has a set of variables for each occurrence in the dataset. The clustering steps for K- Means are as follows:

Step 1: Determine the number of k clusters to be used.

Step 2: Calculate the k centroid

Step 2: Using the squared Euclidian distance, assign each sample to the closest cluster centroid.

Step 3: Repeat the procedures until all of the samples have been grouped [17]

2.4. Related Work

In [25] different models and approaches were proposed to predict the arrival time of the bus. The scholar presented a model by taking one month GPS data which included bus line name, direction, date, coordinate and speed as variables from Pingdingshan Urban City Bus Company. The model utilized three different evaluation techniques; a maximum error, a minimum error of and an average error by dividing the prediction into two parts: road travel time and stop time. The results showed that, both support vector regression and k-nearest neighbor (KNN) methods predict better with a maximum error, a minimum error of and an average error of 84 s, 10 s 42.5 s and an average relative error of 5.74% respectively.

Researcher studied [26] a model using a dataset obtained from Sao Paulo City, which includes bus fleet position data, real-time traffic data, and traffic forecast from Google Maps. Collect geographic data (extensions, bus stops) by cleaning, formatting, and interpreting the data, as well as configuring and training data sets to determine which data sets and hyper parameters to use and give best result. Finally, deliver bus journey time forecasts after finishing the training and validation procedure. With the influence of traffic congestion, the correlations of the data and the most relevant data for ANN training could be assessed correctly 8.97 present (in comparison, the Nave technique resulted in 12.47 MAPE errors). The stronger the dataset's correlation, the more precise the prediction with an error rate of roughly 9%.

In [27]. Proposed artificial neural network model to forecast arrival time by computing three distinct models from the dataset found from GPS date which had time, coordinates, driver id, bus id, routes, and bus stops between one year with additional variables, date of travel, time of trip, weekday, source bus stop, and destination bus stop, length between source and destination bus stop, and weather condition .By applying traditional measurements like mean absolute error and root mean squared error were used to compare the results. Finally, the artificial neural network was shown to be the better model in quantitative errors less than 1 minute.

In [28] applied five different frequently used prediction models of machine learning algorithms and feature techniques for prediction like k nearest-neighbor, kernel regression, additive model, and neural network on the data obtained from the bus GPS city of Dublin, Ireland, covers approximately two months' worth of real-time data. To examine the models, superiority of stop-based trip interpolation method for traditional and data pre-processing method for distance-based trip. The results demonstrate the size of the input data grows larger; "stop-based trip" method reduces the amount of computing time required by up to 5 times when compared to the distance-based trip technique.

In [29] by comparing multiple neural network architectures using historical data and training three models: multilayer perceptron, convolutional neural network, and recurrent neural network, we were able to predict Boston municipal buses. Furthermore, model traffic networks employ historical, statistical, and learning-based models, with learning-based models employing GPS data from buses, which includes latitude, longitude, route, and direction. As a result, the MLP and CNN architectures achieve similar accuracy of 0.88 and 0.85, respectively, but the RNN architecture achieves the best coefficient of determination of around 0.89.

In models were studied to predict bus arrival time throughout various peak seasons (AM, Mid-Day and PM) with different factors like bus stops, the length of the route between bus stops, dwell time, and the number of junctions between bus stops. With six months' worth of AVL and APC data for six bus routes in DC, divided into AM Peak (7:00 AM–9:30 AM), Mid-Day Peak (10:00 AM–2:30 PM), and PM Peak (4:00 PM–6:30 PM). For each route, the study used ANN models with a sample size of 500 origin-to-destination trips per peak period, resulting in a total minimum sample size of 1,500 for the ANN model. According to the findings, ANN models can accurately predict bus travel times on selected routes were 95% [20].

The researcher [30] relied on the use Support Vector Machine (SVM) models to predict bus arrival times tested with the real data and also compared with other three models: k-NN (k-nearest neighbors), ANN, and LR. The raw data were collected from public transport providers, including bus Id, time, current location (latitude and longitude), heading, average speed, number of boarding/alighting passengers at bus stops, and so on from a bus route in Dalian, China. The results showed that SVM models have high accuracy in predicting bus arrival times by making the reliability of bus increase from 0.5 to 0.45.

In [31] Studied a model by collecting rom single bus with 48 stations which had one month GPS data included bus line name, direction, date, coordinate, speed. Then, the study prediction was divided into two parts: road travel time and stop time. Two different models were compared SVR plus KNN method and historical data-based prediction method. The testing findings showed that the SVR plus KNN model achieved enhanced performance than the historical data-based prediction method in estimating the arrival time of a bus on the same road section. The results showed that the method used in this study was more accurate, and the result was closer to the actual value, with a maximum error of 84 seconds, a minimum error of 10 seconds, and a maximum error of 84 seconds an average relative inaccuracy of 5.74 % and an average error of 42.5 seconds.

The author in [32] the study looked at numerous ways for predicting journey time, including historical and real-time approaches, statistical techniques, machine learning techniques, and model-based techniques. However, the researcher developed a bus arrival time prediction model using raw data from Public Transportation Corporation in Jinan, China, using various traffic data. Support Vector Machine, Kalman filter, and Multilayer Perceptron were all examined for better prediction using three distinct evaluation indexes Finally, RNN have better prediction in both Static Factors RMSE (min) 0.6670 MAE (min) 0.5229 MAPE (%) 22.32% & Dynamic Factors RMSE (min) 0.5261, MAE (min) 0.4258 and MAPE (%) 19.07.

In [33] the model comprised ten independent variables with different models like linear regression, artificial neural networks, decision trees, and gene expression programming models. Data from the AVL and passenger traffic were combined into a single dataset for the investigation Inter-stop distance, route completion, dwell time, number of passengers boarding and/or alighting, day and week of the week, time of day, route number, and other variables were found to be significant variables for the models. The ANN time series model, according to the findings, was the best-performing model in terms of computing effort and accuracy. The inter-stop distance i.e., the distance between two stops was also found to be the most important variable or factor across all regression models, while dwell time and the number of persons boarding had no effect on bus journey time.

In [34] different bus arrival/running time prediction models have been developed during the previous few decades. The study used data from Shenyang, China's bus routes 232 and 249, which included location and time. Linear regression, k-nearest neighbors, support vector machine, and classic random forest were the four prediction models used. Finally, the findings show that the random forest model has a high level of accuracy with mean absolute errors value of 13.65, mean absolute percentage error of 6.90, and root mean squared error of 26.37 respectively.

In [35] four predicting bus travel times using GPS data, three distinct models were developed and compared: Historical Averaging, Linear Regression, and Gradient Boosting. The preceding charts show that Gradient Boosting provides superior forecasts; more closely approximates real journey times, and outperforms the other two proposed models. Gradient Boosting beats the other models while predicting arrival times, according to the results. Finally, the result of the Gradient boosting model shows a mean absolute percentage error of 9.62.

Table 0.1: Related Work Summary

Cite	GPS Data (>50,000	ML Algorithm used	Best Algorithm Performance	Performance	Drawback
[28]	-	SVM and KNN	SVM and KNN	Maximum error 84 s Minimum error 10 s Average error 42.5s relative error 5.74%	• Uses same road segment • Compare two models only
[28]	-	KNN, k regression, additive model, and RNN	NN	R-squared 0.87 %	• Used small testing data
[10]	-	MLP, CNN and RNN	RNN	R-squared 0.89 %	• Only one model compared
[33]	-	RNN	RNN	RMSE 0.5261m MAE 0.4258m MAPE 19.07%	• useless variables were used like passenger information
[34]	-	LR, KNN, SVM and RF	RF	MAE 13.65, m MAPE 6.90 m RMSE 26.37 %	• small data were used training and testing models
[35]	-	HV, LR and GB	GB	MAE 0.80 m, MAPE 9.62m and RMSE 1.16%	• small data were used training and testing models

Chapter 3: Research Methodology

The researcher constructs a methodology or experimental process that includes all machine learning steps to achieve the research objectives. Under the upcoming sections details of how the data were collected, understanding the collected data, feature selection, preprocessing such as data cleaning, data formatting, and methodologies that have been used for the model building will be discussed.

2.1. Research Flow

In order to achieve the objectives of the research, the research flow embraced with four stages. The first stage comprises identifying and understanding the domain of the problem by reviewing different literature and then objectives of the study are formulated. The second stage was data gathering and preparations of the dataset to make it suitable for the development of the model, at this stage of the research following procedures were adopted: feature selection and data preprocessing (including: data cleaning, data reduction, handling missing values) and data transformation. At the third stage, the prepared dataset is partitioned into training and testing set and algorithm selection is carried out. In the final stage, a model is designed and compared based on the evaluation metric and choose the best model. Then, based on the result and findings of the study a report has been written.

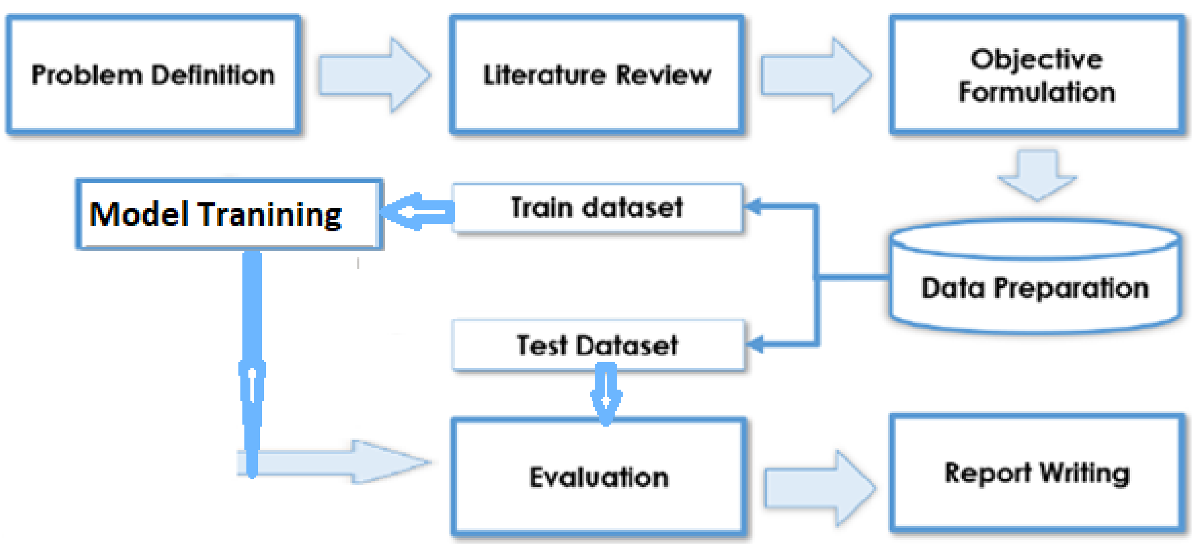

Figure 2.1: Research Flow

2.2. Data Collection

In this study, a one-year GPS data of each day, from 6:00 a.m. to 7:30 p.m. was collected from Sheger Public Transport Addis Ababa, Ethiopia. The dataset has around 140,000 records of initial and destination of bus trip dataset.

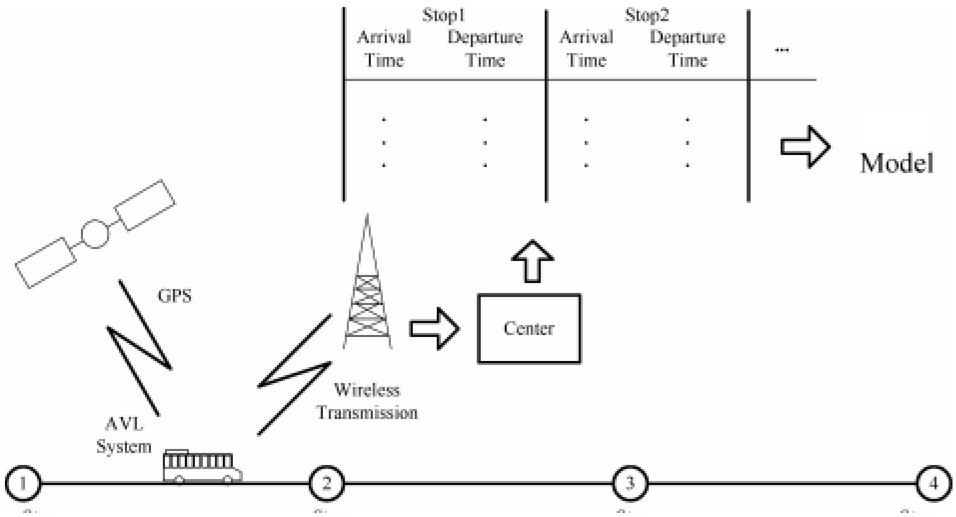

Figure 2.2: Data collection scheme in the study area [36]

1	Date	Month	)eak	Grouping	ginning D	ginning Ti	End Date	End Time	Mileage	uratio	tial latitu	tial longitu	nal latitu	nal longitu	Initial location	Final location
2	Monday	February	Not	AA-3-37781	021-01-04	06:00:00	2021-01-25	20:59:56	8762 km	527	9.002048	38.876694	8.922403	38.764191	የከ በሬ በታ, Et	4, Addis Ababa, Ethiopia
3	Monday	February	Not	AA-3-37781	021-01-04	06:00:14	2021-01-04	06:12:34	7.26 km	0.2	9.020035	38.852539	9.066685	38.864788	Yk_13_0605 :	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
4	Monday	February	Not	AA-3-37781	021-01-04	06:19:26	2021-01-04	07:36:44	24 km	1.28	9.066675	38.864799	9.066710	38.865120	Fikre Mariam	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
5	Monday	February	Not	AA-3-37781	021-01-04	07:47:34	2021-01-04	08:47:20	12.01 km	0.99	9.066731	38.865108	9.020809	38.802269	Fikre Mariam	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
6	Monday	February	Not	AA-3-37781	021-01-04	09:03:10	2021-01-04	09:33:24	11.49 km	0.5	9.020815	38.802258	9.066730	38.865070	Fikre Mariam	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
7	Monday	February	Not	AA-3-37781	021-01-04	09:49:38	2021-01-04	10:37:22	11.57 km	0.79	9.066575	38.864635	9.021213	38.801983	Fikre Mariam	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
8	Monday	February	Not	AA-3-37781	021-01-04	10:54:28	2021-01-04	11:25:18	5.79 km	0.51	9.021235	38.801979	8.983000	38.780258	Fikre Mariam	Ring Road, Addis Ababa, Ethiopia
9	Monday	February	Not	AA-3-37781	021-01-04	11:35:32	2021-01-04	12:07:50	11.05 km	0.53	8.983003	38.780247	8.912194	38.784325	Ring Road, A	Yerer Road, Addis Ababa, Ethiopia
10	Monday	February	Not	AA-3-37781	021-01-04	12:14:39	2021-01-04	13:34:58	17.97 km	1.33	8.912234	38.784348	9.046308	38.762741	Yerer Road, A	Gulele_02_256 St., Addis Ababa, Ethiopia
11	Monday	February	Not	AA-3-37781	021-01-04	13:59:52	2021-01-04	14:25:14	5.99 km	0.42	9.046222	38.762779	9.021878	38.801315	Gulele_02_25	Diaspora Square, Addis Ababa, Ethiopia
12	Monday	February	Not	AA-3-37781	021-01-04	14:44:54	2021-01-04	15:26:48	11.74 km	0.69	9.021881	38.801311	9.066817	38.865078	Diaspora Squ	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia
13	Monday	February	Not	AA-3-37781	021-01-04	16:14:02	2021-01-04	16:45:16	11.69 km	0.52	9.066817	38.865078	9.021930	38.801426	Fikre Mariam	Diaspora Square, Addis Ababa, Ethiopia
14	Monday	February	Not	AA-3-37781	021-01-04	17:11:16	2021-01-04	17:16:02	0.27 km	0.07	9.021924	38.801430	9.020652	38.802570	Diaspora Squ	Fikre Mariam Aba Techan Street, Addis Ababa, Ethiopia

Figure 2.3: Screenshot dataset

2.3. Data Preparation

The process of making data suitable and adequate for the machine learning tool and algorithms to be utilized for model construction is known as data preparation. To go on to the next task, every machine learning tool and software package need some kind of data format.

As a result, the data must be transformed into a format that the tool and software can understand. The initial stage in every machine learning activity is to understand the issue domain and the data that will be used in the process.

Data preparation processes such as data collecting, data and business understanding, dataset description, and data formatting are covered in the sections below. The data cleaning stage is then applied after feature selection and data pre-processing [37].

2.3.1. Understand GPS Data

Before taking any actions, the main activity in any machine learning work is to understand the data. It all starts with gaining a basic understanding of the data and the problem domain. It then moves on to other duties such as evaluating the data quality, missing values, and outliers that could affect the machine learning final output. Understanding the data clearly aids in the identification of some useful and fascinating data insights that will be used to develop a hypothesis for the hidden or unknown information.

2.3.2. Business Process Understanding

Transportation, particularly bus transportation, is subjected to many problems in Addis Ababa city. In particular, the arrival time of buses at stations cannot be predicted accurately. And this serious problem is not a well-researched problem. Also, the transportation management system of Addis Ababa city involves disintegrated manual systems which resulted in poor delivery of transportation services.

In many cities, public bus service is still the most common mode of transportation, and millions of people rely on it every day to go to their destinations. However, in many places, the service is known for its unpredictable scheduling and substantial time differences.

While the system is inherently dependent on external elements it cannot control, such as traffic and weather, some infrastructure enhancements can be done in an attempt to reduce concerns and provide a better experience for riders.

Several cities across the globe have begun to implement real-time monitoring systems that track the locations of each bus in their fleet and make that information public via the internet. This study describes the process for converting real-time vehicle data into a format suitable for machine learning, as well as the strategies utilized to generate precise vehicle arrival time predictions from the GPS data.

The data used in this study is collected from the Sheger bus services in Addis Ababa, Ethiopia, which operate and manage both public and private transportation. TransLink operates a fleet of 20 buses on around 123 routes, with many of them fitted with the AVL system (Automatic Vehicle Location), which tracks each bus's precise GPS coordinates. However, alternative public transportation options, such as Anbessa bus and the other sheger buses have no fleet management and still they used manual system.

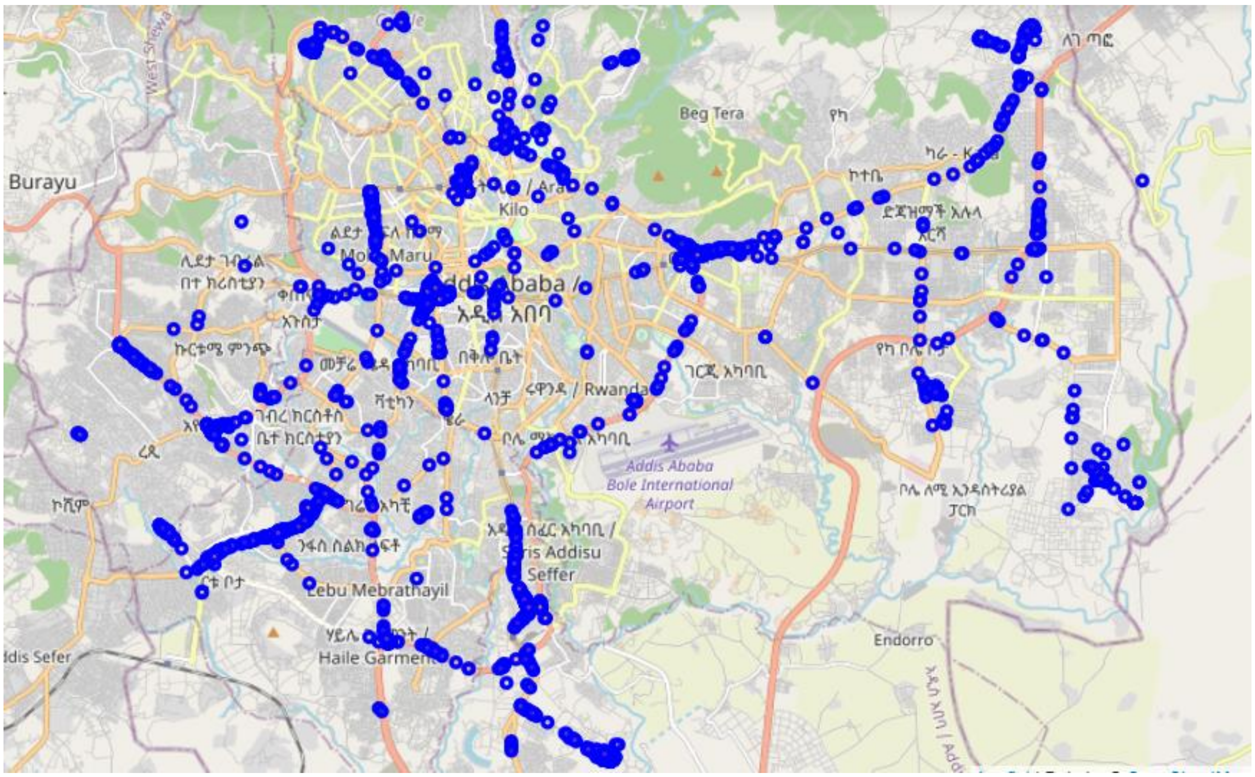

Figure 2.4: Bus Route

2.3.3. Data Formatting

Data formatting was required because the gathered GPS data was in Excel format. Accordingly the Excel file is converted to a comma-separated (CSV) file format that was used later by python programming language for further analysis.

2.4. Feature Selection

The process of producing a subset from an initial feature set according to a feature selection criterion, which picks the relevant features of the dataset, is known as feature selection. It aids in the compression of data processing scales by removing superfluous and extraneous characteristics. Learning algorithms can be pre-processed using feature selection techniques, and effective feature selection results can enhance learning accuracy, shorten learning time, and simplify learning outcomes [36].

In this study, more relevant features that might help develop the bus arrival time prediction model are identified. Features from the GPS data that were not related to the research purpose were deleted in order to train the machine learning algorithms based on domain expertise's consult.

According to domain experts, the 11 variables given in Table 3.1, including the dependent feature, are identified as the most important features for bus arrival time prediction. These features were believed to play a role in predicting actual bus arrival times. As a result, the model building has been based on the 11 most important features.

Table 2.1: Description of the Selected Features

No.	Features Name	Data Type	Description
1.	Beginning Date	Numerical	Initial Date of the trip
2.	Beginning Time	Numerical	Initial Time of the trip
3.	End Date	Numerical	End Date of the trip
4.	Time Range	Numerical	Shows the time of crowding/ traffic congestion
5.	Mileage	Numerical	Total distance in kilometer of the trip
6.	Duration	Numerical	The difference between the projected arrival time and the actual arrival time
7.	Initial latitude	Numerical	Source Latitude of vehicle
8.	Initial longitude	Numerical	Source Latitude of vehicle
9.	Final latitude	Numerical	Destination Latitude of vehicle
10.	Final longitude	Numerical	Destination Latitude of vehicle
11.	End Time	Numerical	End Time of the trip

2.5. Data Preprocessing

2.5.1. Data Cleaning

In any machine learning project, data cleaning is a vital stage. In tabular data, a variety of statistical analysis and data visualization approaches are used to find the place to make data cleaning activities. Before moving to more advanced methodologies, we should definitely undertake some fundamental data cleaning activities on any machine learning task [38]. Finding and repairing (or removing) faulty or inaccurate entries from a record set, table, or database is the process of replacing, modifying, or deleting the dirty or noisy data. The collected dataset, on the other hand, contains no missing data or outline.

```
## Checking for nan values
data.isnull().sum()
```

```
Date                  0
peak/off peak         0
Beginning Time        0
End Time              0
Mileage               0
Initial latitude      0
Initial longitude     0
Final latitude        0
Final longitude       0
dtype: int64
```

Figure 2.5: Missing Values

2.5.2. Data Reduction

Tasks like reducing the number of features attribute values, and tuples are integrated into this technique. These activities can be performed in a range of different ways. Some of these tasks have been carried out in this study. For example, removing irrelevant feature was done in the feature selection section. After applying data cleaning and getting the cleaned dataset a significant dependency or association between the independent variables or the predictor variables must be considered to evidence the presence or absence of multicollinearity. From the below heat-map, it can be concluded that there is no multicollinearity in the features. Therefore, all the features are taken for building the model.

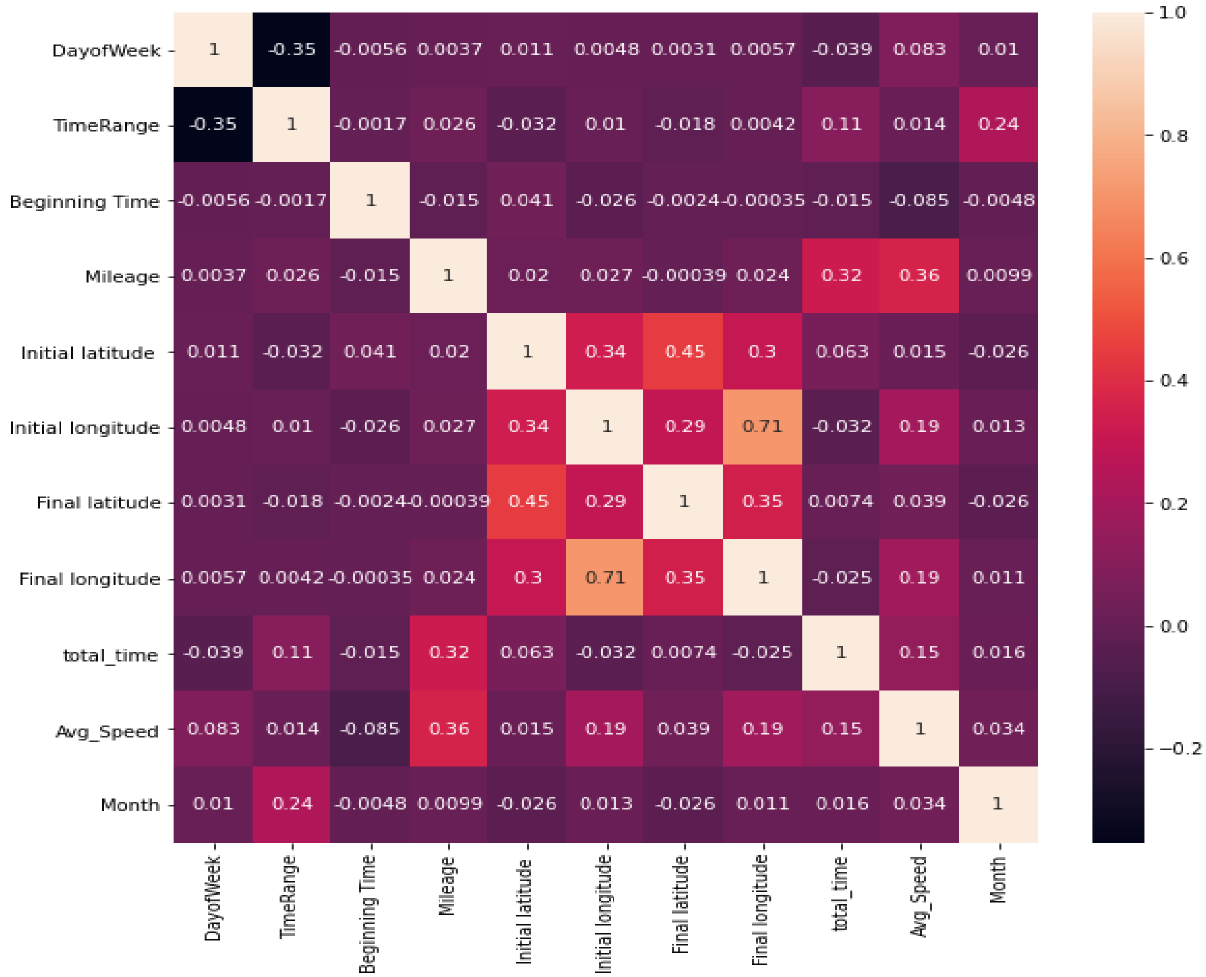

Figure 2.6: Feature Correlation

2.6. Data Transformation

The process of converting data into a format suitable for model building is known as data transformation (data consolidation) [39]. It is the section in which the selected data features are transformed into the most suitable format for the machine learning algorithms. All the selected features were numerical.

The dataset also contains features which have date and time format. These features should also be changed into more suitable format for the model. Hence, the features are converted into numeric value by using python's 'datetime' module. The module used to transform the date into weekday, month and year and the time into hour numerical value.

```
data['Month'] = data['Date'].dt.month
data['Beginning Time'] = data['Beginning Time'].dt.hour
data['End Time'] = data['End Time'].dt.hour
data['Date'] = data['Date'].dt.weekday
```

Figure 2.7: Date and Time Transformation

2.7. Algorithm Selection

Researchers would find it difficult to choose the appropriate machine-learning algorithms to tackle their challenges because there are so many. It is well understood that a single machine-learning method will not provide the optimum results in all issue scenario.

Machine-learning algorithms' performance can be influenced by a number of factors, including data volume, diversity, and velocity, as well as the sort of task at hand.

Managing different trade-offs between accuracy and interpretability, size of the training data, speed or training time, and numbers of features are all challenges in choosing the optimum algorithms [40].

2.7.1. Train/Test Split

One of the most crucial processes in machine learning task is the Train/Test split. It's critical since the model must be reviewed before it can be deployed. And that evaluation must be done on unseen data because all incoming data is unseen when it is distributed.

The train/test split's fundamental idea is to separate the original data set into two sets.

- Training set
- Testing set

This is done by using sklearn library, which includes a built-in function called train_test_split. In this study, the X is used to store all the independent features and y the dependent feature (End Time) and these datasets are split into 10% testing and 90% training set as X_train, X_test, y_train, and y_test.

2.7.2. Performance Evaluation Metrics

After training the model, the model should be tested with never seen (unseen) data to asset the performance or goodness of the model. This allows checking whether the model is good at predicting new data, or whether it is good at predicting only with the data that has been fed previously but the data that hasn't been seen before. The model should be carefully assessed to ensure that the designed model predicts properly. The most commonly used performance matrix for regression prediction problems is mean squared error, the mean absolute error, determination coefficients, and root mean squared error.

R-squared is statistical measure that indicates how well the data is fit to a statistical model. And RMSE is used as a measure of the differences between predicted values of a model and the observed values. Normally, R-squared takes values between 0 and 1. Closer values to 1 are better because the model explains more variance. The formula to calculate each evaluation metrics depicted in the following equations

$$\boldsymbol{MAE = \frac{1}{n}\sum_{i,j=1}^{n} |y_j - y_i|} \tag{3.1}$$

$$\boldsymbol{MSE = \frac{1}{n}\sum_{i,j=1}^{n} \left(y_j - y_i\right)^2} \tag{3.2}$$

Where y_i is the predicted value of instance j, y_j is real target value of instance j and n is the total number of instances.

$$\boldsymbol{RMSE = \sqrt{\frac{1}{n}\sum_{i,j=1}^{n} \left(y_j - y_i\right)^2}} \tag{3.3}$$

$$\boldsymbol{R^2 = \frac{SS_{reg}}{SS_{tot}} = \frac{\sum_j \left(y_j - \bar{y}\right)^2}{\sum_i (y_i - \bar{y})^2}} \tag{3.4}$$

Where predicted labels are denoted by y_j, while genuine labels are denoted by $y_{i.}$. SS_{tot} is the total square value, which is proportional to the variance of the data, and SS_{reg} is the regression sum of squares.

2.7. Implementation Tool and Environment

Different tools and libraries are used to perform the data preparation, data preprocessing and data analysis to develop the prediction model. In this research, the python programming language and libraries are used for the machine learning algorithms implementation with different data processing tools since python is a widely used programming language by many developers, researchers, and data scientists. The following are lists of libraries and tools that are used in this research.

Anaconda: It is a graphical user interface (GUI) of desktop that is included in Anaconda® distribution to allow us to manage conda packages easily, launch applications, environments, and channels without using command-line commands. It is used to put the model into action. It's a free and open-source Python and R programming language distribution for data science and machine learning applications that aims to make package management and deployment easier. It includes a variety of integrated development environments (IDEs) for writing code, such as Jupyter Notebook which has been used to implement the coding part.

Jupyter Notebook: Jupyter Notebook is an open-source web application and interactive computational environment that allows us that allows us to create and share documents that contain visualizations, live code, equations, and narrative text. Uses include numerical simulation, data visualization, statistical modeling, data cleaning, and transformation. It is easy and runs in a web browser.

Pandas: High performance, easy to use data structures, and data analysis tools. It is used for data reading, manipulation, writing, and handling the dataset.

Sklearn: It is a python free machine learning library which is used to work on the different classification and regression algorithms.

Numpy: Array processing for number, strings, and objects. We use it to handling our data set features for training and testing of the model.

Matplotlib and Seaborn: Python libraries that are used for publication of quality figures. They are used for data visualization.

Haversine: It is a python package that used to calculate the distance between two geolocations using latitude and longitude.

Figure 2.8: Software tools and libraries

2.7.1. Testing Environment

The tools used for implementation were installed on a personal computer Hp Laptop with Intel® Core™ i7-8750H CPU @2.20GHz, 2208Mhz, 6 Core(s), 12 Logical Processor (s), 16 Gigabyte of RAM, 18.8 Gigabyte of total virtual memory, and 500 Gigabyte hard disk storage capacities. The operating system is Windows 10 Enterprise, 64 bits.

Chapter 4: Experimental Result and Discussion

The practical implementation and experimental outcomes of the research are described and discussed in this chapter. It includes information on the computer that was used for the implementation, as well as the programming language, software tools, and libraries that were used. The chapter then goes over all of the experiments that went into creating the model, before concluding by analyzing the performance of categorization models and reporting the results.

3.1. System Architecture

The system architecture, as illustrated in Figure 4.1, begins with GPS data collection and subsequently converting the collected Excel file data to CSV file format. The data was then prepared for analysis. The data preparation phase comprises picking the best features and preprocessing. The prepared dataset is then divided into training and testing sets, with the training set being used to train the algorithm and the testing set being used to test the model's performance. The model trained using the selected algorithms is assessed and compared in the model evaluation step. After that, the algorithm with the best results is chosen. Finally, the best model chosen is utilized to forecast bus arrival time.

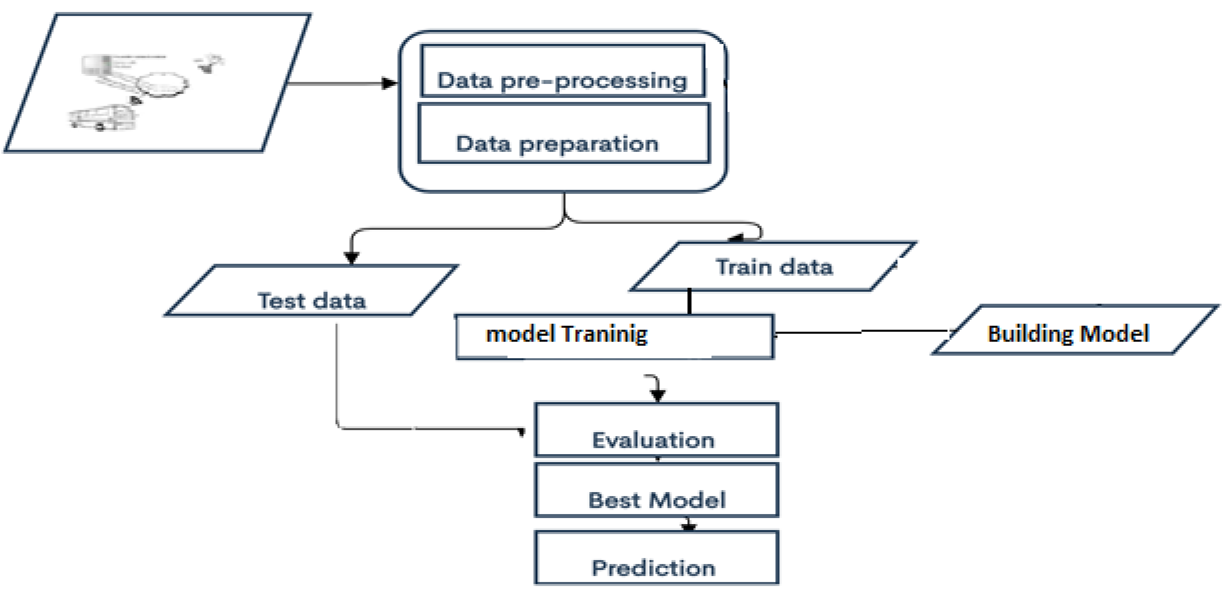

Figure 3.1: Proposed Architecture

3.2. Preliminary Analysis

Before building the machine learning model, additional features that helps for the model are derived from the collected GPS data. In this section the derived features are discussed as follows.

3.2.1. Time Series Analysis

In this study, time is an important feature, since it reveals how the data changes over time, as well as the final outcomes is also time feature. Hence, all the time and date features in the dataset are handled by using python datetime function. It helps to transform the date into month, day and year and time to minute, hour and second.

```
data['Month'] = data['Date'].dt.month
data['Beginning Time'] = data['Beginning Time'].dt.hour
data['End Time'] = data['End Time'].dt.hour
data['Date'] = data['Date'].dt.weekday
```

Figure 3.2: Date and Time Transformation

As can be seen from the above figure, the date is changed into weekday and month. In addition to that beginning time and end time are changed into hour.

3.2.2. Deriving Average Speed of the Bus

Average speed is one of the factors that might influence the final arrival time of the bus. It is derived by subtracting start time from end time and divide mileage by total time. The following code shows the derivation of average speed of the bus.

$$\text{AVERAGE SPEED} = \frac{\text{total distance traveled}}{\text{total time}}$$

$$\text{Total time} = \text{End Time} - \text{Beginning Time}$$

$$\text{Average Speed} = \text{Mileage}/\text{total time}$$

Figure 3.3: Deriving Average Speed of a Bus

After deriving the average speed, the dataset had increased with a new column called Avg_Speed as follows.

```
## shows sample 5 data randomly
data.sample(5)
```

	DayofWeek	TimeRange	Beginning Time	Mileage	Initial latitude	Initial longitude	Final latitude	Final longitude	End Time	total_time	Avg_Speed	Month
7583	5	1	52	0.20	9.068649	38.875397	9.068598	38.875240	53	0.017222	11.612903	6
77865	1	4	9	6.90	9.010457	38.746960	9.020719	38.802418	42	0.533889	12.924037	2
53856	1	2	52	8.46	9.009475	38.744522	8.960463	38.711857	29	0.605556	13.970642	3
135208	4	5	11	13.59	8.912875	38.783379	8.960844	38.713139	53	0.702222	19.352848	10
72134	2	3	20	12.03	8.960485	38.712681	9.035555	38.752689	4	0.741111	16.232384	10

Figure 3.4: Dataset after Adding Average Speed

3.2.3. Calculating Distance between Two Locations

Since one bus travels from one initial location to a predetermined destination, it's possible to calculate the distance it travels by using haversine distance formula. Haversine distance formula determines the great-circle distance between two locations on sphere by using geolocation information (longitude and latitude). This has been done by using a python module called haversine. The following code shows the function that calculate the distance a bus travels from initial location (using initial latitude and longitude) to final location (using final latitude and longitude).

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right)$$

$$c = 2 \cdot \operatorname{atan2}\left(\sqrt{a}, \sqrt{(1-a)}\right)$$

$$d = R \cdot c$$

Φ – latitude, λ – longitude, R – radius of the earth (R ≈ 6.371 km)

Figure 3.5Haversine formula

```python
# haversine function
from haversine import haversine
from haversine import haversine_vector, Unit
from math import radians


def haversine(lat1, lon1, lat2, lon2, to_radians=True, earth_radius=6371):
    """
    Calculate the great circle distance between two points
    on the earth (specified in decimal degrees or in radians)

    All (lat, lon) coordinates must have numeric dtypes and be of equal length.

    """
    lat1 = data['Initial latitude']
    lon1 = data['Initial longitude']
    lat2 = data['Final latitude']
    lon2 = data['Final longitude']
    if to_radians:
        lat1, lon1, lat2, lon2 = np.radians([lat1, lon1, lat2, lon2])

   # a = np.sin((lat2-lat1)/2.0)**2 + \
   #     np.cos(lat1) * np.cos(lat2) * np.sin((lon2-lon1)/2.0)**2

    data['distance'] = haversine_vector([lat1, lon1], [lat2, lon2], Unit.KILOMETERS)
```

Figure 3.6: Calculate distance from initial to final location

The following Figure shows the dataset after calculating the haversine distance from the initial location and final location of the bus trip.

```python
data.sample(5)
```

	DayofWeek	peak/off peak	Beginning Time	End Time	Mileage	Initial latitude	Initial longitude	Final latitude	Final longitude	total_time	Avg_Speed	Month	distance
74950	2	1	14	15	13.64	8.964787	38.744450	9.035839	38.751144	1.299444	10.496794	11	9.945651
17351	0	0	9	11	34.00	9.024541	38.822395	9.064456	38.718704	2.087222	16.289593	2	17.188942
124117	1	0	6	6	10.78	9.036055	38.751984	8.948624	38.764683	0.447778	24.074442	8	4.639621
120996	1	0	10	11	24.00	8.912086	38.784225	8.991082	38.855087	1.236111	19.415730	7	4.309398
70884	0	1	17	17	11.99	9.020754	38.802948	9.066708	38.864803	0.614444	19.513562	10	8.559532

Figure 3.7: Dataset after Distance Calculation

The dataset covers areas with the minimum latitude and longitude range of (8.74698, 38.668259) and maximum rage (9.07393, 38.990913).

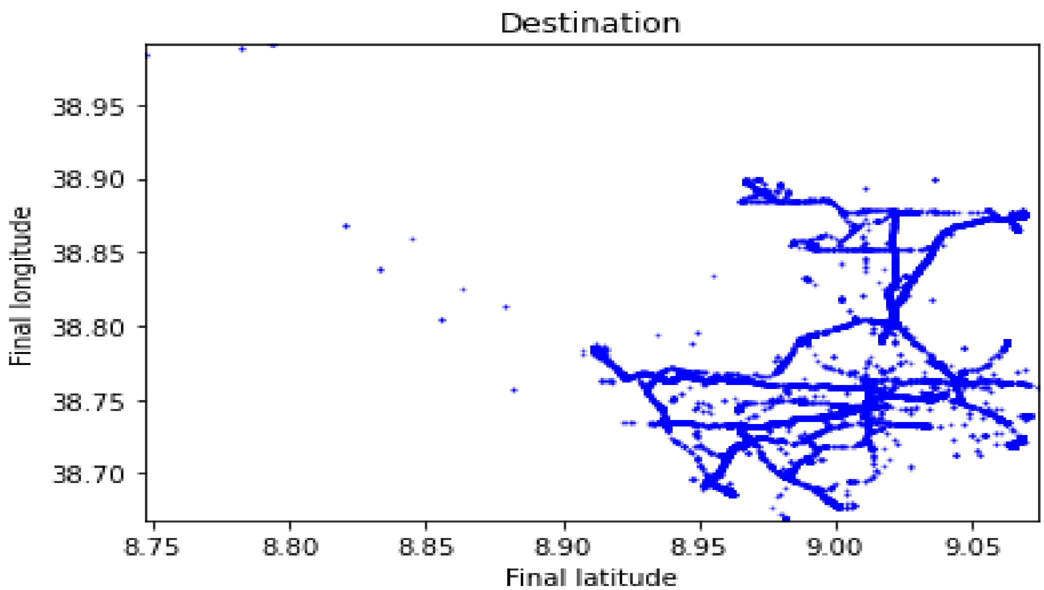

Figure 3.8: Bus trip locations

Based on the GPS data, the buses covers and travels to more 123 routes as it can be shown in Figure 4.7.

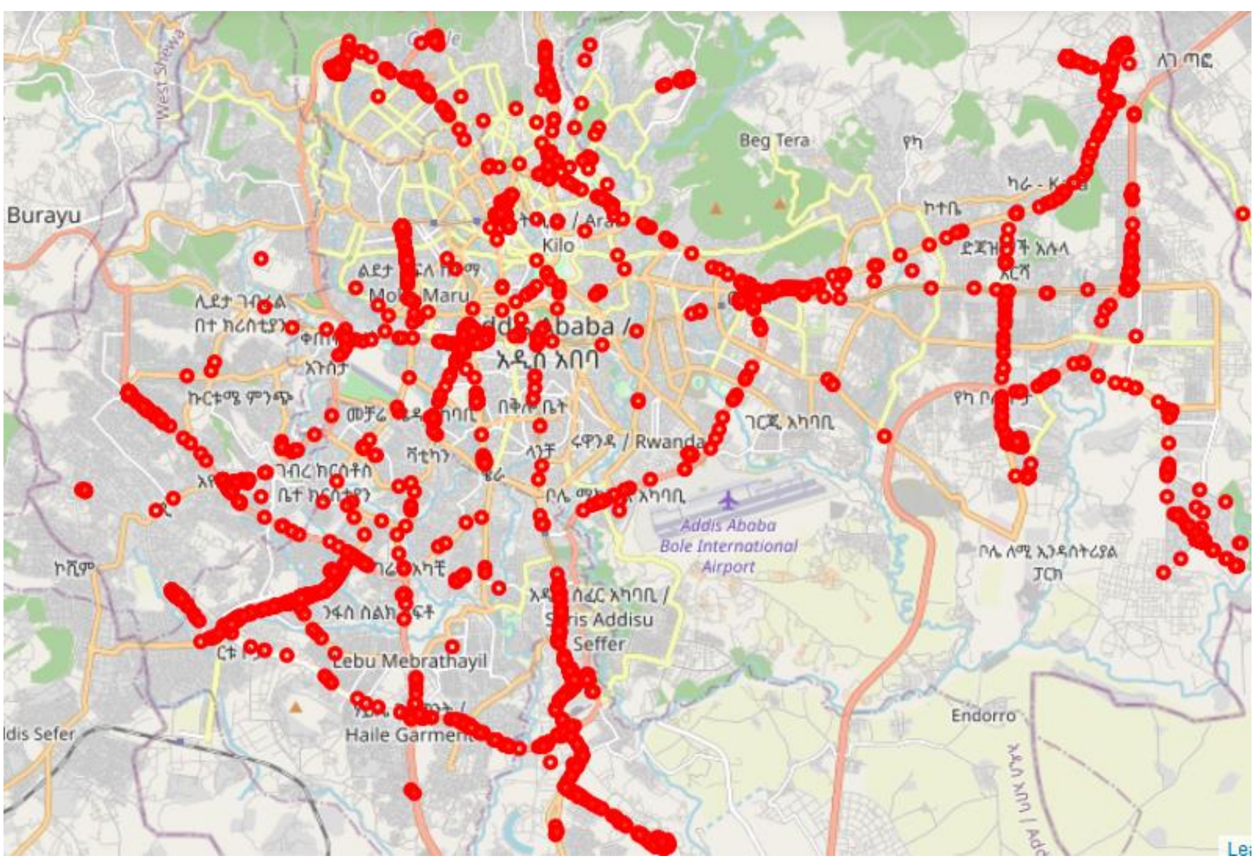

Figure 3.9: Bus Trip Routes

3.3. Feature Scaling

Due to the range of each input feature can be highly diverse, the data must be scaled before being fed into the model. For example, a duration range can be tens to thousands of seconds, with 1 and 0 representing peak and off-peak hours, respectively. Huge differences in the input features might lead to a fitting problem, since a feature with a large number may be given different weights than a feature with a tiny number. As a technique of improving the prediction model, min-max scaling is utilized to scale the value of the data between 0 and 1 are used in this study. This can be done by using the sklearn StandardScaler built-in function.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

```
print("Mean of the dataset: ", np.mean(X).round(8))
print("Standard deviation of the dataset: ", np.std(X).round(8))
```

```
Mean of the dataset:  -0.0
Standard deviation of the dataset:  1.0
```

Figure 3.10: Feature Scaler

3.1. Train/Test Split

Before diving into model building, the dataset must be split into training set and testing set to make the model more efficient by training the model by using the training set and then evaluate it the performance of the model by using testing set. Therefore, from the total 140589 instances of the data 90% is used for training and 10% for testing purpose.

```
# print the shapes of training and test set
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(126530, 12)
(14059, 12)
(126530,)
(14059,)
```

Figure 3.11: Train/Test Split

3.2. Model Building

After doing all the necessary data preparation and preprocessing steps, the next step is building the model which is the main aim of this study. As discussed in previous chapter, five different machine learning regression algorithms are selected to develop the models and finally one best model which achieved the best performance will be selected as the final model. The following Figures illustrate the model training and evaluation result of each model developed by random forest, gradient boosting, k nearest neighbor, artificial neural network and support vector regression algorithms.

```
## Model building with Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
import time
start_time = time.time()
RF=RandomForestRegressor()
# feeding the training data into the model
RF.fit(X_train, y_train)
# predicting the values for x-test
predictions=RF.predict(X_test)
print("Execution time: " + str((time.time() - start_time)) + ' sec')
```

```
Execution time: 87.10375809669495 sec
```

```
from sklearn import metrics
## Testing score
print('Random Forest MAE Result:', metrics.mean_absolute_error(y_test, predictions))
print('Random Forest MSE Result:', metrics.mean_squared_error(y_test, predictions))
print('Random Forest RMSE Result:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
print('Random Forest R2 Score Result:', RF.score(X_train, y_train))
```

```
Random Forest MAE Result: 0.8129646489793017
Random Forest MSE Result: 14.289671591151576
Random Forest RMSE Result: 3.7801681961457185
Random Forest R2 Score Result: 0.9947617860321322
```

Figure 3.12: Random Forest Regressor Model

The model built with random forest algorithm has fitted the training dataset and has achieved R^2 score of 0.994, MAE of 0.812, RMSE of 3.780 and MSE of 14.28. It took around 87 sec to execute.

```python
from sklearn.ensemble import GradientBoostingRegressor
start_time = time.time()
model = GradientBoostingRegressor()
model.fit(X_train, y_train)
# predicting the values for x-test
predictions_gb=model.predict(X_test)

print("Execution time: " + str((time.time() - start_time)) + ' sec')
```

```
Execution time: 24.389780044555664 sec
```

```python
## Testing score
print('Gradient Boosting MAE Result:', metrics.mean_absolute_error(y_test, predictions_gb))
print('Gradient Boosting MSE Result:', metrics.mean_squared_error(y_test, predictions_gb))
print('Gradient Boosting RMSE Result:', np.sqrt(metrics.mean_squared_error(y_test, predictions_gb)))
print('Gradient Boosting R2 Score Result:', model.score(X_train, y_train))
```

```
Gradient Boosting MAE Result: 9.429548166756563
Gradient Boosting MSE Result: 170.90294173187442
Gradient Boosting RMSE Result: 13.072985188237398
Gradient Boosting R2 Score Result: 0.4928544398585749
```

Figure 3.13: Gradient Boosting Regressor Model

The model built with gradient boosting algorithm has fitted the training dataset and has achieved R^2 score of 0.492, MAE of 9.429 RMSE of 13.072 and MSE of 170.902 It took around 24 sec to execute. It has less execution time and the performance is less than random forest.

```python
from sklearn.neural_network import MLPRegressor
ann_model = MLPRegressor()
# Fitting the ANN to the Training set
ann_model.fit(X_train, y_train)
pred_ann=ann_model.predict(X_test)

print("Execution time: " + str((time.time() - start_time)) + ' sec')
```

```
Execution time: 4902.285028934479 sec
```

```python
## Testing score
print('ANN MAE Result:', metrics.mean_absolute_error(y_test, pred_ann))
print('ANN MSE Result:', metrics.mean_squared_error(y_test, pred_ann))
print('ANN RMSE Result:', np.sqrt(metrics.mean_squared_error(y_test, pred_ann)))
print('ANN R2 Score:', ann_model.score(X_train, y_train))
```

```
ANN MAE Result: 2.471835352306847
ANN MSE Result: 39.629977182572375
ANN RMSE Result: 6.295234481937299
ANN R2 Score: 0.8882106258813406
```

Figure 3.14: ANN Model

The model built with ANN algorithm has fitted the training dataset and has achieved R^2 score of 0.888, MAE of 2.471, RMSE of 6.295 and MSE of 39.629. It took around 4902 sec to execute.

```
from sklearn.neighbors import KNeighborsRegressor
model3 = KNeighborsRegressor(n_neighbors=5)
model3.fit(X_train, y_train)
# predicting the values for x-test
pred_knn=model3.predict(X_test)

print("Execution time: " + str((time.time() - start_time)) + ' sec')
```

```
Execution time: 198.89184165000916 sec
```

```
## Testing score
print('KNN MAE Result:', metrics.mean_absolute_error(y_test, pred_knn))
print('KNN MSE Result:', metrics.mean_squared_error(y_test, pred_knn))
print('KNN RMSE Result:', np.sqrt(metrics.mean_squared_error(y_test, pred_knn)))
print('KNN R2 Score:', model3.score(X_train, y_train))
```

```
KNN MAE Result: 7.808393200085355
KNN MSE Result: 165.00410840031296
KNN RMSE Result: 12.84539249693496
KNN R2 Score: 0.6777786001563577
```

Figure 3.15: K Nearest Neighbor Model

The model built with k nearest neighbor algorithm has fitted the training dataset and has achieved R^2 score of 0.677, MAE of 7.808, RMSE of 12.845 and MSE of 165.004. It took around 198 sec to execute.

```
from sklearn.svm import SVR
model5 = SVR(kernel='rbf')
model5.fit(X_train, y_train)

# predicting the values for x-test
pred_svm=model5.predict(X_test)

print("Execution time: " + str((time.time() - start_time)) + ' sec')
```

```
Execution time: 7651.031231164932 sec
```

```
## Testing score
print('SVM MAE Result:', metrics.mean_absolute_error(y_test, pred_svm))
print('SVM MSE Result:', metrics.mean_squared_error(y_test, pred_svm))
print('SVM RMSE Result:', np.sqrt(metrics.mean_squared_error(y_test, pred_svm)))
print('SVM R2 Score:', model5.score(X_train, y_train))
```

```
SVM MAE Result: 0.28813773591764036
SVM MSE Result: 0.16580833662801858
SVM RMSE Result: 0.4071956981943923
SVM R2 Score: 0.989458099170198
```

Figure 3.16: Support Vector Regressor Model

The model built with support vector regression algorithm has fitted the training dataset and has achieved R^2 score of 0.989, MAE of 0.288, RMSE of 0.407 and MSE of 0.166. It took around 7651 sec to execute. It has taken more execution time and the performance is less compared to random forest.

3.3. Model Comparison and Discussion

The fact that, the dataset used in this study were not having any missing values and outliers. However, form the selected five algorithms; model built with random forest achieved the best result compared to the rest five algorithms by achieving less errors values and very high R^2 score because the random forest adds extra randomization to the model by separating a node from a random subset of features rather than the most important feature while building the trees. When the determinant factor approaching to 1 indicates that the model can predict the value by having less error rate.

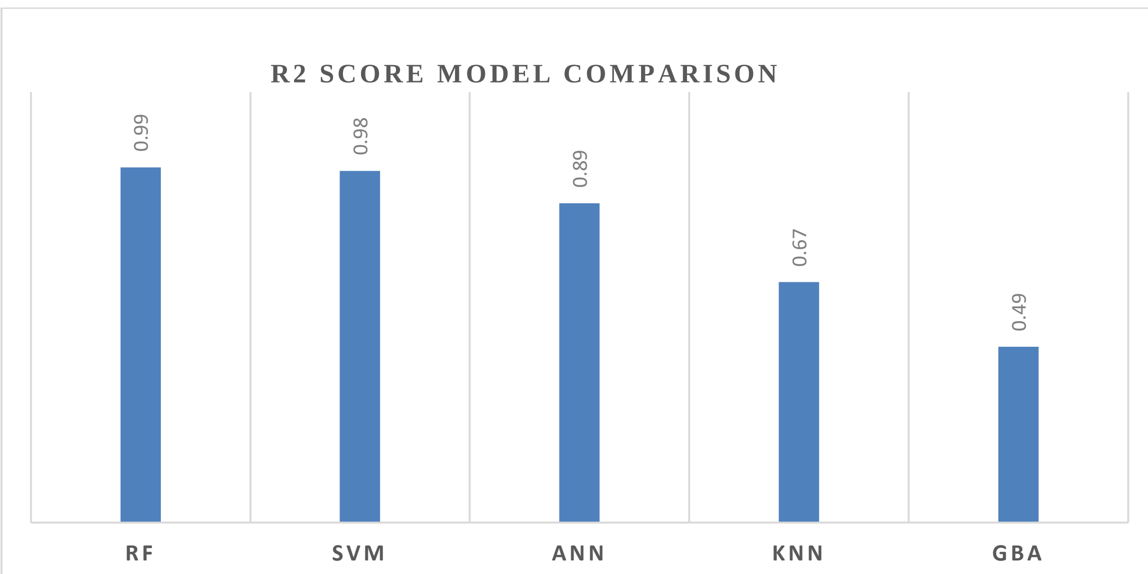

Figure 3.17: Model performance comparison

3.4. Model Testing

As illustrated in the Figure, the random forest regressor model outperformed the other models by achieving less error. Therefore, in this phase the model is tested or prediction is made by using unseen/test data.

```
result_dict = {'test_arrival_time': y_test,'predicted_arrival_time': predictions}
results = pd.DataFrame(result_dict)
results.head(10)
```

	test_arrival_time	predicted_arrival_time
108921	51	51.62
48776	21	21.05
129140	25	25.08
82308	59	58.42
97480	4	4.09
91554	5	5.98
31219	4	3.70
41340	34	34.04
97282	34	33.91
119769	14	14.02

Figure 3.18: Model Testing Result

The model testing result depicted in Figure 4.17, demonstrate the model built with random forest regressor model predicts the arrival time with less error. Therefore, it can be concluded that random forest algorithm is a very promising and can be used to predict the arrival time of buses in Addis Ababa city.

3.5. Finding and Discussion

The general objective of this study was to develop a machine learning model that predicts bus arrival time by using regression technique by formulating two research questions. The first research question for this thesis was: "What are the key factors or features that help to develop and predict bus arrival time prediction model?" Hence this research question is answered by analyzing the features gained from the collected GPS dataset. And day of week, time range, beginning time, initial latitude, initial longitude, final latitude, final longitude, total time, average speed, month and distance are considered as the key factors/features that help to build the prediction model and proven by getting best performance score.

Based on the result of the experiment, all the algorithms which were used in this research can be used to predict bus arrival time. From all the algorithms, Random Forest attained the highest score compared to other algorithms used in this study and this could answer research question two. "Which machine learning algorithm performs better?" Therefore, algorithms like Random Forest can be used for better prediction of bus arrival time.

Chapter 5: Conclusion and Recommendations

4.1. Conclusion

It is becoming increasingly important to have good public transportation in order to sustain and improve quality of life by providing mobility and accessibility. It also helps to safeguard the environment, promote economic development, and foster social solidarity. Various public transportation improvement ideas have been presented in the past. One of them is utilizing the Advanced Public Transportation System to give reliable trip and travel information like bus arrival time for travelers/public transport users many tourists in Ethiopia use public transportation services such as buses, particularly in Addis Ababa, and they face challenges due to a lack of travel information.

The use of machine learning regression techniques for bus arrival time prediction was the focus of this study. It's part of the pre-construction work for Addis Ababa's Advanced Public Transportation System. To the researchers' knowledge, no attempt has ever been made before. For such an APTS system, real-time data gathering and techniques are required for better arrival time prediction. To create the models, the GPS data, which was relatively new at the time, and five machine learning regression algorithms, including random forest, gradient boosting, artificial neural network, support vector regression, and k nearest neighbor are employed. Then, the performance of each model is tested and compared to get the model that can predict the arrival time of buses efficiently.

As one might assume, traffic conditions in developing countries differ from those in developed countries, with more variety and a lack of lane discipline. As a result, compared to short-term trip arrival time prediction using homogeneous data in most earlier reported experiments, the prediction system requires greater attention during development. The issues are exacerbated by a lack of historical data, traffic congestion data, meteorological data, and long-term data gathering strategies. As a result, this study is one of the first to use an urban route in Addis Ababa to predict bus arrival time in a mixed traffic situation.

The performance metric determines whether an algorithm will occasionally make a forecast that is more accurate The overall findings of this study is very encouraging, and shows that the proposed random forest model can be used to Advance and modernize the public transportation system by predicting bus arrival times at bus stops at bus stops in the study area with undisciplined traffic flow, particularly since the fleeting monitoring system is new in the area.

To conclude, the introduction of this machine learning applications will expand and enhance the public transportation system's reliability, attracting more public transportation like buses and reducing traffic congestion or crowded in Addis Ababa.

4.2. Recommendation

Based on the result and study made in this research study the following recommendations are made for further research and advance in the intelligent transportation system of Ethiopia.

- Weather and traffic were not taken into consideration during this research work because of limitation of time. Accordingly, in addition to using current GPS data features as an input to the model, another study may look into how these factors affect the prediction results in the future.
- Future research can also be done by including and adding data of the bus trip information of the bus in each bus stop and route.

References

[1] L. Vanajakshi, "Development of a real time bus arrival time prediction system under Indian traffic conditions," *Center of Excellence in Urban Transport,* 2011.

[2] T. R. Board, "The Federal Transit Administration TCRP Synthesis 34 Data Analysis for Bus Planning and Monitoring A Synthesis of Transit Practice," 28 8 2021. [Online]. Available: http://onlinepubs.trb.org/onlinepubs/tcrp/tsyn34.pdf.

[3] ""PASSIV E" ITS PROBE Travel Time Data Collection Handbook 5-1 5 ITS PROBE VEHICLE TECHNIQUES," 28 8 2021. [Online]. Available: https://www.fhwa.dot.gov/ohim/handbook/chap5.pdf.

[4] B. a. K. B. A. a. V. L. Dhivyabharathi, Real time bus arrival time prediction system under Indian traffic condition, 2016.

[5] "ADDIS ABABA 2017 THE ADDIS ABABA WE WANT," [Online]. Available: https://www.urbanafrica.net/wp-content/uploads/2017/07/State-of-Addis-Ababa-2017-Report-web-1.pdf.

[6] "Why should Ethiopians care about urbanization? Jobs, infrastructure, and formal land and housing," 27 3 2019. [Online]. Available: https://blogs.worldbank.org/africacan/why-should-ethiopians-care-about-urbanization-jobs-infrastructure-and-formal-land-and-housing.

[7] Chibani, S., & Coudert, F. X, "Machine learning approaches for the prediction of materials properties," APL Materials, vol. 8, p. 080701, 2020.

[8] Arhin, Stephen, et al, "Predicting Bus Travel Times in Washington, DC Using Artificial Neural Networks (ANNs)," 2021.

[9] S. a. C. F.-X. Chibani, "Machine learning approaches for the prediction of materials properties," *APL Materials,* vol. 8, p. 080701, 2020.

[10] R. Turner, Python Machine Learning: 3 books in 1-The Ultimate Beginners, Intermediate and Expert Guide to Master Python Machine Learning, 2020.

[11] Guido, Sarah, and Andreas C. Müller. "Introduction to machine learning with Python, 2016."

[12] R. Turner, Python Machine Learning: 3 books in 1-The Ultimate Beginners, Intermediate and Expert Guide to Master Python Machine Learning, 2020.

[13] Osisanwo, F. Y., et al. , "upervised machine learning algorithms: classification and comparison," *International Journal of Computer Trends and Technology (IJCTT),* vol. 48, pp. 128--138, 2017.

[14] P. D. A. U. a. S. M. Python, Mastering Machine Learning with Python in Six Steps.

[15] Mishra, Arvind K., et al, "Random forest tree based approach for blast design in surface mine," *Geotechnical and Geological Engineering,* vol. 36, pp. 1647--1664, 2018.

[16] P. D. A. U. a. S. M. Python, Mastering Machine Learning with Python in Six Steps.

[17] P. D. A. U. a. S. M. ython, Mastering Machine Learning with Python in Six Steps.

[18] Mishra, Arvind K., et al, "Random Forest Tree Based Approach for Blast Design in Surface Mine," *Geotechnical and Geological Engineering,* vol. 36, pp. 1647--1664, 2018.

[19] E. Alpaydin, Introduction to Machine Learning Second Edition, 2020.

[20] Mohammed, M., Khan, M. B., & Bashier, E. B. M, Machine learning: algorithms and applications, 2016.

[21] "What Is Time Series Forecasting?," 12 2016. [Online]. Available: https://machinelearningmastery.com/time-series-forecasting/.

[22] Brockwell, Peter J., et al., Introduction to time series and forecasting, 2016.

[23] "Lasso Regression: Simple Definition," 24 9 2015. [Online]. Available:

https://www.statisticshowto.com/lasso-regression/.

[24] A. a. K. A. Natekin, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotic,* vol. 7, p. 21, 2013.

[25] Y. Wang, "Intellectualization of the urban and rural bus: The arrival time prediction method," *Journal of Intelligent Systems,* pp. 689--697, 2021.

[26] G. H. a. Y. L. R. a. M. C. L. Larsen, Bus Travel Times Prediction based on Real-Time Traffic Data Forecast using Artificial Neural Networks, 2020, pp. 1--6.

[27] N. a. M. K. Jargalsaikhan, An investigation of machine learning methods for prediction bus travel time of Mongolian public transportation, vol. 11515, 2020, p. 115151S.

[28] D. a. S. J. a. W. S. Liu, Bustime: Which is the right prediction model for my bus arrival time?, 2020, pp. 180--185.

[29] C. S. Sloan, Neural bus networks, 2018.

[30] Gu, Xiaoning, et al., "Reliability prediction of further transit service based on support vector machine," *Measurement and Control,* 2020.

[31] Y. Wang, "Intellectualization of the urban and rural bus: The arrival time prediction method," *Journal of Intelligent Systems,* pp. 689--697, 2021.

[32] Zhou, Xin, et al., "Learning dynamic factors to improve the accuracy of bus arrival time prediction via a recurrent neural network," Future Internet, p. 247, 2019.

[33] Ranjitkar, Prakash, et al., "Bus arrival time modeling based on auckland data," Transportation Research Record, pp. 1--9, 2019.

[34] Yu, Bin, et al., "Prediction of bus travel time using random forests based on near neighbors," Computer-Aided Civil and Infrastructure Engineering, vol. 33, pp. 333--350, 2018.

[35] A. a. B. M. aparia, Bus Journey and Arrival Time Prediction based on Archived AVL/GPS

data using Machine Learning}, 2021, pp. 1--6.

[36] Z. K. Gurmu, A dynamic prediction of travel time for transit time for transit vehicles in Brazil using GPS data, 2010.

[37] T. a. H. S. Beshah, "Mining road traffic accident data to improve safety: role of road-related factors on accident severity in Ethiopia," 2010.

[38] J. Brownlee, "https://machinelearningmastery.com/basic-data-cleaning-for-machine-learning/," 19 3 2020. [Online]. Available: https://machinelearningmastery.com/basic-data-cleaning-for-machine-learning/.

[39] Koller, Daphne, et al, "Introduction to statistical relational learning," 2007.

[40] M. Stewart, "The Limitations of Machine Learning," 29 7 2019. [Online]. Available: https://towardsdatascience.com/the-limitations-of-machine-learning-a00e0c3040c6.

[41] Cai, Jie, et al., "Feature selection in machine learning: A new perspective," *Neurocomputing,* pp. 70--79, 2018.

[42] C.-W. a. L. C.-J. Hsu, "A comparison of methods for multiclass support vector machines," *IEEE transactions on Neural Networks,* pp. 415--425, 2002.

[43] W. S. a. P. W. McCulloch, "A logical calculus of the ideas immanent in nervous activity," pp. 115--133, 1943.

[44] D. O. Hebb, "The first stage of perception: growth of the assembly," *The Organization of Behavior,* pp. 60--78, 1949.

[45] H. L. a. D. S. E. Dreyfus, Making a mind versus modelling the brain: artificial intelligence back at the branchpoint, 1991, pp. 33--54.

[46] Ren, Shaoqing, et al., "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems,* pp. 91--99, 2015.

[47] Ren, Shaoqing, et al., "Faster r-cnn: Towards real-time object detection with region proposal

networks," *Advances in neural information processing systems,* pp. 91--99, 2015.

[48] C. S. Sloan, "neural bus networks," 2018.

[49] J. Y. a. C. T. W. Yam, "A weight initialization method for improving training speed in feedforward neural network," *Neurocomputing,* pp. 219--232, 2000.

[50] Welch, Richard L., Stephen M. Ruffing, and Ganesh K. Venayagamoorthy, Comparison of feedforward and feedback neural network architectures for short term wind speed prediction, 2009, pp. 3335--3340.

[51] X. a. W. Q. Hu, "Estimating impervious surfaces from medium spatial resolution imagery using the self-organizing map and multi-layer perceptron neural networks," *Remote Sensing of Environment,* pp. 2089--2102, 2009.

[52] S. a. E. V. Savalia, "Cardiac arrhythmia classification by multi-layer perceptron and convolution neural networks," *Bioengineering,* p. 35, 2018.

[53] P. J. Werbos, Neural networks for control and system identification, 1989, pp. 260--265.

[54] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural networks,* pp. 339--356, 1988.

[55] Chauvin, Yves, and David E. Rumelhart, "Backpropagation: Theory, architectures and applications," *Backpropagation: Theory, architectures and applications,* pp. 1--34, 1995.

[56] P. J. Werbos, Neural networks for control and system identification, 1989, pp. 260--265.

[57] Su, Hong Te, Thomas J. McAvoy, and Paul Werbos, "Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach," *Industrial \& engineering chemistry research,* pp. 1338--1352, 1992.

[58] "https://www.heflo.com/blog/process-mapping/business-process-analysis-methodology/," 24 3 2019. [Online].

[59] "URBAN ACCESSIBILITY / MOBILITY INDEX FEASIBILITY STAGE REPORT," 28

8 2021. [Online]. Available: https://documents1.worldbank.org/curated/en/264131468336679123/pdf/699330ESW0P1030asibilityStageReport.pdf.

[60] "Addis Ababa - Ethiopia," 2021. [Online]. Available: http://www.uemi.net/addis-ababa---ethiopia.html.

[61] A. a. T. N. a. S. A. Singh, A review of supervised machine learning algorithms, 2016, pp. 1310--1315.

[62] Ren, Shaoqing, et al., "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91--99, 2015.