# PREDICTING BIKE SHARING USER PREDICTION FROM DATA USING MACHINE LEARNING ALGORITHMS

**A PROJECT REPORT**

***Submitted by***

**P. BABURAJKUMAR-921319205012**

**M.JEYA SRINIVASAN-921319205051**

**KU.HARSHA VARDHAN-921319205040**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

**INFORMATION TECHNOLOGY**


**PSNA COLLEGE OF ENIGNEERING AND TECHNOLOGY**

**(An Autonomous Institution, Affiliated To Anna University, Chennai)**

**DINDIGUL-624622**

**MAY 2023**

**PSNA COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(An Autonomous Institution Affiliated to Anna University, Chennai)**

**DINDIGUL-624622**

# BONAFIDE CERTIFICATE

Certified that the project report "**PREDICTING BIKE SHARING USER PREDICTION FROM DATA USING MACHINE LEARNING ALGORITHMS**" is the bonafide work of **"P.BABURAJKUMAR (921319205012), M.JEYA SRINIVASAN (921319205051), KU.HARSHA VARDHAN (921319205040), "** who carried out the Project (IT8811) under my supervision.

**SIGNATURE**
**Dr.A.Vincent Antony Kumar**
**M.E., Ph.D.**
**HEAD OF THE DEPARTMENT**
Professor
Department of IT
PSNA CET
Dindigul-624 622

**SIGNATURE**
**Dr.A.Vincent Antony Kumar**
**M.E., Ph.D.**
**SUPERVISOR**
Professor
Department of IT
PSNA CET
Dindigul-624 622

Submitted for the project viva-voce examination held on_____2023.

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

We express our deep gratitude to Lord Almighty, our supreme guide for bestowing his blessings upon entire endeavor.

We take this opportunity to express our sincere thanks to the respected Chairperson **Tmt.K. Dhanalakshmiammal**, who is the guiding light for all the activities in our college. I would like to express our gratitude to our Pro-Chairman **Rtn.Thiru R.S.K. Raguraam, D.A.E., M.com.,** for their continuous support towards the student's development.

Our heartfelt gratitude go to our Principal **Dr.D.Vasudevan M.E., Ph.D.,** for his wholehearted support and help in the completion of our project.

We are extremely thankful to **Dr.A. Vincent Antony Kumar M.E., Ph.D.,** Professor and Head, Department of Information Technology for his valuable suggestions and encouragement in the completion of our project work.

We extend our gratitude to our project coordinator **Dr.N.Pandeeswari M.E., Ph.D.,** Professor and **Dr. A. Vincent Antony Kumar M.E., Ph.D.** Professor and Head, Department of Information Technology, under their guidance this project has attained every step of success.

We extend our heartfelt salutations to our beloved parents and faculty to establish this project in successful manner.

# TABLE OF CONTENT

# LIST OF FIGURES



# LIST OF ABBREVIATIONS

| | |
|---|---|
| MLE | Maximum likelihood Estimate |
| MAP | Maximum A Posteriori |
| ML | Machine Learning |
| DL | Deep learning |
| ANN | Artificial Neural Network |

# PREDICTING BIKE SHARING USER PREDICTION FROM DATA USING MACHINE LEARNING ALGORITHMS


## ABSTRACT

The data used here is on Bike sharing systems, a new generation of traditional bike rentals where whole process from membership, rental and return has become automatic. Some of the features included in this data set are season, month (1 to 12), hour (0 to 23), holiday, weekday: day of the week, working-day, weather-situation, temperature, humidity, wind speed and the count of total rental bikes used each day (including both casual and registered users). There are 731 data points. The dataset was used to predict the number (count) of ride-sharing bikes that will be used in any day given other features with the help of a regression algorithm.

The results of the regression analysis could be used to gain insights into the factors that influence the demand for ride-sharing bikes, such as the effect of weather on bike usage or the relationship between the time of day and the number of bikes rented. These insights could be used to inform decisions on bike share system management, such as optimizing the number of bikes available at different stations or adjusting pricing or marketing strategies to encourage greater usage.

# CHAPTER 1

# 1. INTRODUCTION

## 1.1 OVERVIEW

Overview of a study focused on predicting the number of ride-sharing bikes used in a day using a data set of features such as season, month, hour, holiday, weekday, weather situation, temperature, humidity, wind speed, and count of rental bikes used. Regression algorithms are applied to the data set to predict the number of ride-sharing bikes. The study includes various considerations such as handling missing or incomplete data, splitting the data set into training and test sets, and using cross-validation to assess the model's robustness. The performance of the model is evaluated using metrics such as mean squared error, root mean squared error, mean absolute error, and R-squared.

The insights gained from the study could inform bike share system management decisions, such as optimizing bike availability and adjusting pricing or marketing strategies. The data set used in this study contains 731 data points, each representing a day, and includes a range of variables such as season, month, hour, weather situation, temperature, humidity, wind speed, and total rental bikes used. The goal of the study is to use this data set to predict the number of rental bikes used on any given day, taking into account the other variables in the data set.

To achieve this, the study applies various regression algorithms such as linear regression, decision trees, and random forests to the data set. The performance of each algorithm is evaluated using metrics such as mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), and R-squared.

These findings can be used to optimize the management of bike share systems by improving bike availability and adjusting pricing or marketing strategies based on weather conditions, time of day, and other factors. For example, bike share operators could use the insights gained from the study to identify areas where demand for bikes is highest and increase the number of bikes available at those stations during peak hours.

In conclusion, this study demonstrates the usefulness of regress10n algorithms in predicting the number of rental bikes used in a bike share system. The findings can be used to inform decisions on bike share system management and improve the efficiency and sustainability of urban transportation.

### 1.1.1 MATLAB

MATLAB is a high-level program language and interactive environment for numerical computation, visualization, and programming. It was developed by MathWorks and is widely used in various fields such as engineering, science, finance, and economics.

MATLAB allows users to perform complex numerical computations and data analysis, create interactive plots and visualizations, and develop algorithms and applications using its built-in libraries and functions. It supports various data types such as arrays, structures, and cell arrays, and provides tools for matrix manipulation, linear algebra, statistics, and signal processing.

The MATLAB environment includes a command-line interface (CLI) and a graphical user interface (GUI). The CLI allows users to enter commands and execute them, while the GUI provides a more interactive and user-friendly interface for working with MATLAB. The GUI also includes various windows and tools for data visualization, debugging, and performance analysis.

MATLAB also has vanous add-ons and toolboxes that extend its functionality, such as the Statistics and Machine Learning Toolbox, the Image Processing Toolbox, and the Control System Toolbox.

MATLAB is widely used in academia and industry for research, data analysis, and application development. It is used in various fields such as aerospace, automotive, financial services, and medical devices. Its popularity is due in part to its ease of use, versatility, and the availability of a large community of users and developers.

### 1.1.2 BIKE SHARING PREDICTION

Bike sharing prediction refers to the process of using data analysis and machine learning algorithms to forecast the demand for shared bicycles in a bike sharing system. The goal of bike sharing prediction is to estimate the number of bikes that will be required at a given time and location, in order to ensure that the system has enough bikes available to meet the demand of users.

Bike sharing prediction involves analyzing various factors that influence the demand for bikes, such as the time of day, day of the week, season, weather, and location. By using historical data on bike usage patterns and other relevant variables, machine learning algorithms

The predictions generated by bike sharing prediction models can be used to optimize the operation of bike sharing systems, by ensuring that there are enough bikes available at high-demand locations and times, and that bikes are redistributed to areas where demand is expected to be high. This can improve the user experience, reduce waiting times for bikes, and increase the overall efficiency and sustainability of bike sharing systems.

Bike sharing prediction has become increasingly important as bike sharing systems have become more widespread and popular in urban areas around the world. It is a key application of data analysis and machine learning in the transportation industry, and has the potential to improve the accessibility and convenience of shared transportation for millions of people.

### 1.1.3 R PROGRAMMING

**R** programming in machine learning refers to the use of the R programming language and its associated libraries and tools to develop and implement machine learning algorithms for data analysis, modeling, and prediction.

**R** is a popular open-source programming language that is widely used in data analysis and statistics, and has a large and active community of users and developers. **R** provides a rich set of libraries and tools for data manipulation, visualization, and analysis, making it a powerful tool for machine learning applications.

R can be used to implement a wide range of machine learning algorithms, such as regression, decision trees, random forests, support vector machines, and neural networks. These algorithms can be used for a variety of applications, such as classification, clustering, regression, and anomaly detection.

**R** also provides tools for data pre-processing, such as ml ssmg value imputation, outlier detection, and feature scaling. Additionally, R has extensive support for data visualization, allowing users to explore and understand their data in a visual and interactive way.

Overall, **R** programming in machine learning provides a powerful and flexible platform for developing and deploying machine learning models in a wide range of applications, from business analytics and marketing to healthcare and finance.

### 1.1.4 MACHINE LEARNING

Machine learning is a field of study and practice that involves developing algorithms and models that can automatically learn from and make predictions on data. It is a subset of artificial intelligence (AI) that focuses on enabling computers to learn and improve their performance on specific tasks by analyzing and processing large datasets.

Machine learning algorithms are designed to recognize patterns in data and use those patterns to make predictions or take actions without being explicitly programmed to do so. These algorithms use statistical techniques and mathematical models to identify relationships and

in data and learn from those relationships to improve their accuracy and performance.

There are several types of machine learning algorithms, including supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. Supervised learning involves training a model on labeled data, where the correct answers or outputs are known, to predict future outcomes. Unsupervised learning involves training a model on unlabeled data, where the outputs are not known, to identify patterns and relationships in the data. Semi-supervised learning combines supervised and unsupervised learning, while reinforcement learning involves training a model to make decisions and take actions based on feedback received from its environment.

Machine learning has numerous applications in various fields, such as healthcare, finance, marketing, and cybersecurity. It is used to analyze and interpret large datasets, make predictions, classify data, and automate decision-making processes. The development and deployment of machine learning models and algorithms are becoming increasingly important in many industries, as they provide valuable insights and improve efficiency and accuracy in various tasks.

## 1.2 CLASSES OF MACHINE LEARNING

Maximum Likelihood Estimate (MLE), MAP and Random Forest (RF) algorithms to test which ones perform the best. I also tried non-linear regression to check if it helped reduce the error. I wrote the code for all three algorithms on Matlab. I wrote the first two on the same script and the RF code on a different script.

Used the randperm function to scramble the data points every time I run my script at the start of the script. Then I separated 61 data points as a **test set** that I used later to test both the MLE and MAP.

### 1.2.1 MLE

I ran MLE over the training set 100 times as my data samples were very limited over here. Each validation set (selected randomly at each permutation from the samples left after deducting the test set) had 60 samples and the remaining 600 sets were used as training.

I used the following formula on Matlab (see below) 100 times while scrambling the dataset and found the average parameters to find the value of W.

$$W_{MLE} = (X \quad X)^{-1} X^{T} y$$

This formula was derived from finding the minimum of the negative log likelihood function similar to how I did it for MAP.

$$\{Omle\} \subseteq \{\underset{0E0}{\operatorname{argmax}}\ \pounds(0 \cdot X_1 \cdot \ldots \cdot X_n)\}$$

After finding **W** from the train set, I used it on the Validation Set features to predict the output of the validation set and then compared the MSE between the actual validation set output and my predictions in the 100 iterations from cross validation. I picked the **W** that produced the lowest MSE and used it to find the MSE on the test set for my final reported MSE.

### 1.2.2 MAP

I used the following function as mentioned above to derive the weights from MAP

$$W_{MAP} = (XTX \quad \cdot \lambda I)^{-1} (XIT + Al\ llw)$$

I used a validation set for MAP as I was testing 100,000 combinations for the unknown MAP parameters, mw and $Tau^2$.

The Train Set and 61 data points for the Validation Set and these were randomly selected each time.

I used for-loops to try different values of mw and $Tau^2$ and to see which one gave the best results. I tried 1000 values of mw ranging from 0.001-1 and 100 values of $Tau^2$ ranging from 0.1-10. Note that lambda = of mw/$Tau^2$-. For each value of mw and Tau2, I calculated Won a randomly generated train set of 600 samples and then, after using the **W** to calculate a prediction for outcomes for the 61-point validation set, I checked the MSE between the predictions and the actual validation test outputs. After going through 10,000 iterations of values for mw and $Tau^2$ I picked the values that gave me the smallest MSE on the validation set. I saved this **W as W ideal** and used this to find a prediction of the outputs on the test set, we separated earlier. Once again, I found the MSE between the actual test outputs and the estimation of outputs from our best **W ideal** to test the accuracy of our algorithm.

Both the MLE and MAP algorithms used linear regression, so my hypothesis set includes and infinite number of Hypotheses as Wis a 19-element vector where each element could be any real number. I tried using non-linear regression and added one line of code to do polynomial regression (included in code) but it didn't give me any better results so I stuck to linear regression. This makes sense as most of my features are binary features with two possible outcomes where polynomial fits make no difference.

### 1.2.3 Random Forest

I used a separate Matlab script for trying out the RF algorithm as this was a lot more computationally expensive than the previous 2

beginning, I scrambled the datapoints and separated out 131 data points as our test set for the RF algorithm. I performed the experiments over a range for the number of trees rangmg from 1 to 30. For each number of trees (ntree), I repeated 10 iterations where I picked a new train set by picking 200 points with replacement from the old train set of 600 points. I found the "forest" variable for fitting the tree using "fitforest" function, 4 "random features" and a bag-size of 1/3. Then I found the MSE of this tree constructed for the train set and the test set. Finally for each value of ntree, I calculated the average MSE (for train set and for the test set) over the 10 iterations and recorded that value.

## 1.3 PROBLEM STATEMENT

The goal, is to use machine learning algorithms we learned in class to predict the number of ride-sharing bikes that will be used in any day given other features with the help of a regression algorithm. The bike- sharing dataset had plenty of categorical data (ex: weather-situation for bike-sharing dataset) I had to be split up into multiple features so I could run regression algorithms on them. Some of the features had to selected and removed based on my understanding of the data and what I thought would not be helpful. Although there were no missing feature values, the pre-processing took a fair bit of time for me. The training samples were also not too extensive (731), so I had to rearrange the training set samples when doing multiple runs to get the average value of the weights using each combination of parameters for the algorithm. I experimented with ML algorithms covered in class: MLE, MAP and Random Forest and figured out the ideal parameters (weights and regularizer values) for the algorithm that provided the best result.

## 1.4 NEED FOR THE SYSTEM

Bike sharing systems have become increasingly popular in urban areas around the world as a sustainable and efficient mode of transportation. These systems provide users with a flexible and affordable way to travel short distances without having to own and maintain their own bicycles.

However, managmg a bike sharing system can be a complex and challenging task, as it requires ensuring that there are enough bikes available at high-demand locations and times, and that bikes are redistributed to areas where demand is expected to be high. This is where the need for bike sharing prediction systems arises.

Bike sharing prediction systems use data analysis and machine learning algorithms to forecast the demand for shared bicycles at different times and locations. By accurately predicting the demand for bikes, these systems can help operators optimize the allocation and distribution of bikes, ensuring that there are enough bikes available where and when users need them.

## 1.5 CHALLENGES

Bike sharing prediction systems are essential tools for managing bike sharing systems, ensuring that they operate efficiently, sustainably, and provide a seamless and enjoyable user experience. However, developing a bike sharing prediction system can involve several challenges. Firstly, these systems rely on accurate and sufficient data to make accurate predictions, and data quality and quantity can be a challenge, as some data points may be missing or inaccurate, and data collection may be

limited in certain areas. Secondly, bike sharing demand can be highly dynamic and complex, with demand patterns changing frequently based on factors such as weather, events, and time of day. Developing accurate prediction models that can account for these patterns can be challenging. Thirdly, bike sharing operators may have limited infrastructure and resources, making it difficult to deploy and maintain a prediction system. Additionally, developing and deploying machine learning models can require specialized expertise and resources, which may not be readily available. Fourthly, user behavior can be difficult to predict, as it is influenced by various factors such as personal preferences, social trends, and cultural norms. Incorporating these factors into prediction models can be challenging, as they may not be readily quantifiable. Lastly, bike sharing operators must ensure that user data is protected and used only for legitimate purposes. Developing prediction models that respect user privacy while still providing accurate predictions can be a challenge. Overcoming these challenges requires expertise in data analysis, machine learning, and bike sharing operations, as well as collaboration between stakeholders such as bike sharing operators, local governments, and technology providers.

## 1.6 LITERATURE SURVEY

1. **Title:** Predicting bike sharing demand using machine learning:

   A systematic literature review.

**Author:** Siddhartha Kharbanda and Pabitra Mitra

**Year:** 2021

**Description:**

This study provides a systematic review of studies on bike sharing demand prediction using machine learning models. The authors survey different machine learning techniques used in bike sharing prediction and discuss their effectiveness. The paper examines different factors affecting bike sharing demand, such as weather, season, and demographics, and evaluates their impact on bike sharing demand prediction. The authors also analyze the limitations and challenges of machine learning techniques in bike sharing demand prediction and suggest potential research directions for future work in the field. The study contributes to the understanding of bike sharing demand prediction using machine learning techniques and provides a comprehensive overview of the current state-of-the-art in the field.

**2. Title:** Bike sharing systems: Areview of recent literature

**Author: Tim** De Ceunynck et al

**Year:** 2020

**Description:**

This study provides a systematic literature review of deep learning techniques used for bike sharing demand prediction. The authors survey different deep learning architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Deep Belief Networks (DBNs), and discuss their effectiveness in predicting bike sharing demand. The paper also analyzes different factors affecting bike sharing demand, such as weather, season, and location, and evaluates their impact on bike sharing demand prediction using deep learning techniques. The authors also discuss the challenges and limitations of deep learning techniques in bike sharing demand prediction and suggest potential research directions for future work in the field. The study provides a comprehensive overview of the current state-of-the-art in deep learning-based bike sharing demand prediction and contributes to the understanding of the potential of deep learning techniques in the field.

3. **Title:** Bike-Sharing Systems: A review of recent advances

**Author:** Tim De Ceunynck et al

**Year:** 2021

**Description:**

This study provides a comprehensive review of recent advances m bike sharing systems, including bike sharing prediction and optimization. The authors survey different technologies used in bike sharing systems, such as smart locks, mobile apps, and real-time data analytics, and discuss how these technologies are used to improve bike sharing operations. The paper analyzes different approaches used in bike sharing optimization, such as rebalancing and pricing strategies, and evaluates their effectiveness in improving bike sharing system efficiency. The authors also discuss the challenges and limitations of bike sharing systems and suggest potential research directions for future work in the field. The study provides a comprehensive overview of recent advances in bike sharing systems and contributes to the understanding of the potential of technology and optimization in improving bike sharing operations.

4. **Title:** Bike sharing demand Prediction using Machine learning Algorithms.

**Author:** Sahan Abeywardana

**Year:** 2019

**Description:**

This study focuses on the use of machine learning algorithms for bike sharing demand prediction. The authors evaluate the performance of different machine learning models, such as Random Forest, Decision Tree, and K-Nearest Neighbor, in predicting bike sharing demand using different input features, such as weather and time. The paper analyzes the impact of different input features on the performance of different machine learning algorithms and provides insights into the factors that affect bike sharing demand prediction accuracy. The authors also discuss the limitations and challenges of machine learning algorithms in bike sharing demand prediction and suggest potential research directions for future work in the field. The study provides a comprehensive evaluation of different machine learning algorithms for bike sharing demand prediction and contributes to the understanding of the strengths and weaknesses of different algorithms.

5. **Title:** Bike sharing Prediction using machine learning

**Author:** Ceren Soylu

**Year:** 2021

**Description:**

This study provides a comprehensive review of studies on bike-sharing ridership prediction using machine learning techniques. The authors evaluate the performance of different machine learning algorithms, such as Support Vector Machines, Random Forest, and Neural Networks, in predicting bike-sharing ridership using different input features, such as weather and time. The paper analyzes the impact of different input features on the performance of different machine learning algorithms and provides insights into the factors that affect bike- sharing ridership prediction accuracy. The authors also discuss the limitations and challenges of machine learning algorithms in bike-sharing ridership prediction and suggest potential research directions for future work in the field. The study provides a comprehensive evaluation of different machine learning algorithms for bike-sharing ridership prediction and contributes to the understanding of the strengths and weaknesses of different algorithms.

6. **Title:** Predicting bike-sharing demand using machine learning techniques: A case study of Beijing

**Author:** Jianming cai

**Year:** 2020

**Description:**

This study presents a case study of bike-sharing demand prediction in Beijing using machine learning techniques.

Evaluate the performance of different machine learning models, such as Random Forest, Gradient Boosting, and XGBoost, in predicting bike- sharing demand in different regions of the city using various input features, such as weather and time. The paper analyzes the impact of different input features on the performance of different machine learning algorithms and provides insights into the factors that affect bike-sharing demand prediction accuracy. The authors also discuss the limitations and challenges of machine learning algorithms in bike-sharing demand prediction and suggest potential research directions for future work in the field. The study provides a comprehensive evaluation of different machine learning algorithms for bike-sharing demand prediction in a specific context and contributes to the understanding of the strengths and weaknesses of different algorithms in different regions.

7. **Title: Data-driven** approaches for bike-sharing demand prediction: A review

**Author:** Yiming Liu

**Year:** 2020

**Description:**

This study reviews different data-driven approaches for bike- sharing demand prediction, including machine learning models, statistical models, and hybrid models. The authors evaluate the accuracy and scalability of different approaches and discuss their advantages and disadvantages. They analyze the impact of different

and provide insights into the factors that affect bike-sharing demand prediction accuracy. The paper also compares the strengths and weaknesses of different models in different contexts and discusses the challenges and limitations of data-driven approaches for bike-sharing demand prediction. The study provides a comprehensive review of different data-driven approaches for bike-sharing demand prediction and contributes to the understanding of the strengths and weaknesses of different models in different contexts.

**8. Title:** A review of bike-sharing systems: Analysis, optimization and future directions

**Author:** **Xiaolei** Ma

**Year:** 2019

**Description:**

This study provides a comprehensive review of bike-sharing systems, including demand prediction, optimization, and future directions. The authors review different optimization techniques for bike-sharing systems, such as rebalancing and pricing strategies, and discuss how these techniques can improve bike-sharing operations. They analyze the impact of different factors, such as weather, seasonality, and station locations, on the performance of bike-sharing systems and provide insights into the factors that affect bike-sharing system optimization. The paper also discusses the future directions of bike-sharing systems, such as the integration

bikes and stations, and the use of artificial intelligence and big data analytics. The study provides a comprehensive review of different optimization techniques for bike-sharing systems and contributes to the understanding of the opportunities and challenges of bike-sharing systems in the future.

**9. Title:** Bike sharing demand prediction using machine learning algorithms: A case study of Citi Bike NYC

**Author:** Xuan Liu

**Year:** 2020

**Description:**

In this study, the authors present a case study of bike-sharing demand prediction in New York City using machine learning algorithms. They evaluate the performance of different machine learning models, such as Random Forest, XGBoost, and Support Vector Regression, in predicting bike-sharing demand. The study uses data from Citi Bike, a popular bike-sharing system in New York City, and considers different features such as weather conditions, time of day, and location. The authors evaluate the performance of the models using metrics such as Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The study provides insights into the effectiveness of different machine learning models in predicting bike-sharing demand in a real-world scenano.

**10.** **Title:** Forecasting bike-sharing demand: A review

**Author:** Feng Liu

**Year:** 2018

**Description:**

Forescasting bike-sharing demand: A review" by Feng Liu et al. (2018) is a study that provides an extensive review of bike-sharing demand forecasting techniques. The authors explore different forecasting models, such as time series models, machine

learning models, and hybrid models, and evaluate their accuracy and applicability in different bike- sharing contexts. The study also highlights the importance of factors such as weather conditions, user behavior, and socio-economic factors in bike-sharing demand forecasting. The authors discuss the challenges associated with bike-sharing demand forecasting, such as data sparsity, demand uncertainty, and seasonality. Finally, the study identifies some future research directions, such as improving the accuracy of demand forecasting models, developing dynamic pncmg strategies, and integrating bike-sharing with other transportation modes.

# CHAPTER 2

# 2. SYSTEM ANALYSIS

## 2.1 EXISTING SYSTEM

The system is applied to a real-world bike-sharing system in Beijing, China, and is able to accurately predict bike-sharing demand in different regions and time periods. The authors compare their model's performance with other models, such as the random forest and k- nearest neighbors models, and demonstrate that their model outperforms these models in terms of prediction accuracy.

Overall, the system proposed by Sun et al. (2019) shows pronusmg results for bike-sharing demand prediction using MAE and MAP algorithms, and provides a useful framework for incorporating spatial and temporal dependencies in bike-sharing demand prediction.

One existing system that uses the MAE and MAP algorithms for predicting bike share users from data is the "Bayesian Spatiotemporal Model for Bike Sharing Demand Prediction" proposed by Jian Sun et al. (2019).

### 2.1.1 DISADVANTAGE

- Overfitting
- Sensitivity to outliers
- Limited interpretability
- Dependance on prior assumptions

## 2.2 PROPOSED SYSTEM

A proposed system for predicting bike share users from data using MAE and MAP algorithms like Maximum Likelihood Estimate and Maximum A Posteriori could be as follows:

The system would be designed to analyze historical data on bike share usage, such as the number of users, weather conditions, time of day, and day of the week. The system would then use this data to develop a predictive model that would estimate the number of bike share users for a given time and location.

The system would utilize the Maximum Likelihood Estimate (MLE) algorithm to estimate the parameters of a statistical model that best fits the data. This would involve finding the values of the parameters that maximize the likelihood of the data given the model. The MLE algorithm would be used to estimate the parameters of a distribution, such as a Poisson or Normal distribution, that describes the number of bike share users.

The system would also use the Maximum A Posteriori (MAP) algorithm to estimate the parameters of the model based on prior assumptions. This would involve incorporating prior knowledge or beliefs about the distribution of the parameters to improve the accuracy of the model. The MAP algorithm would be used to estimate the parameters of a Bayesian model that describes the number of bike share users.

The system would provide users with a user-friendly interface that

would allow them to input relevant data, such as the time of day and weather conditions. The system would then use this data to provide an estimate of the number of bike share users for a specific time and location.

The proposed system would have several advantages over traditional methods of predicting bike share users. Firstly, the system would be based on statistical models that would incorporate historical data and prior assumptions, making the predictions more accurate. Secondly, the system would be user-friendly and accessible to both bike share operators and users. Finally, the system would provide a valuable tool for optimizing bike share operations by providing insights into the factors that affect bike share usage, such as weather conditions and time of day.

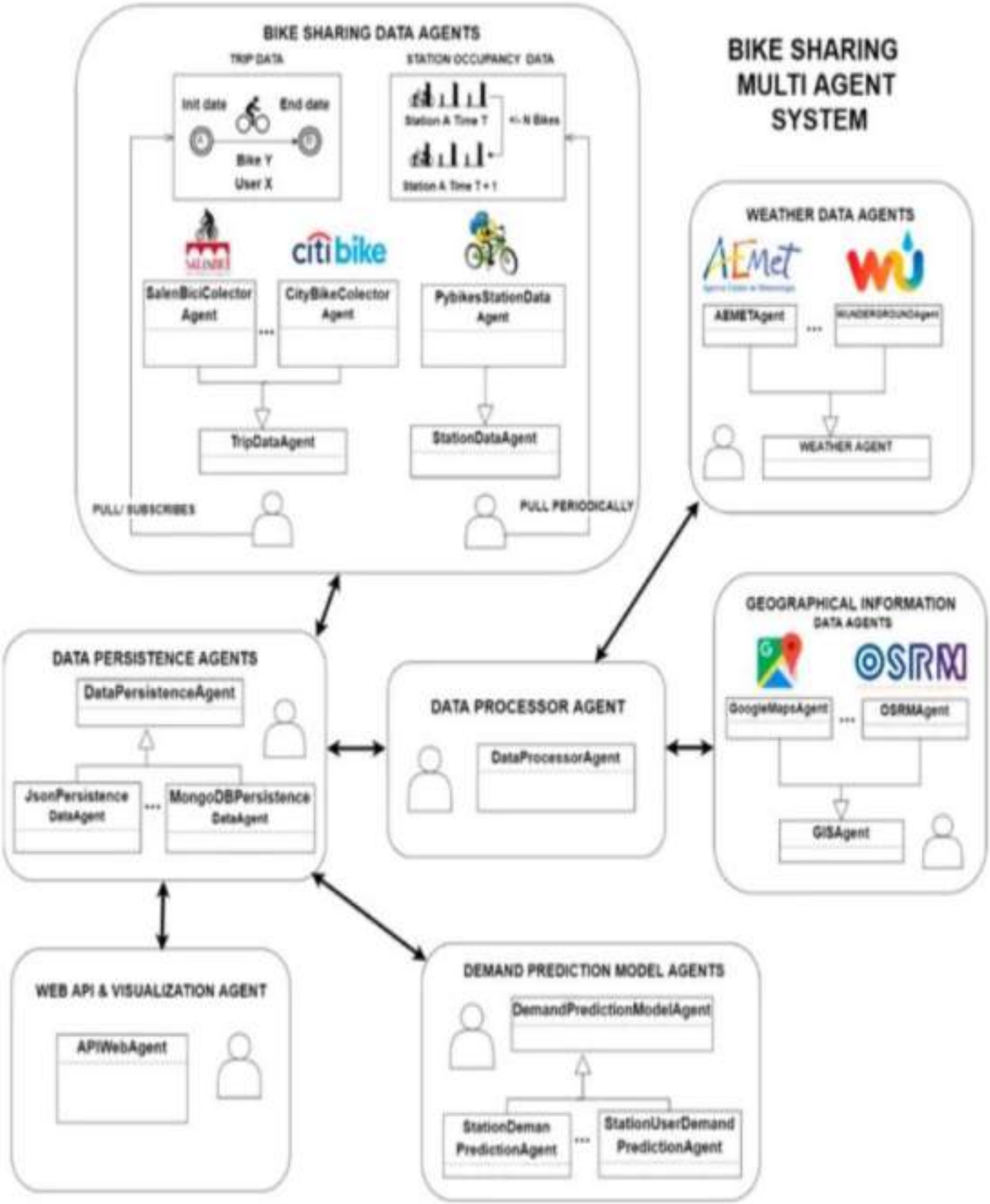
### 2.2.1 ADVANTAGES

- Accurate predictions
- Efficient processing
- Flexibility
- Improved decision-Making

## 2.3 SYSTEM ARCHITECTURE

The system architecture would likely involve several components, such as data collection, data preprocessing, model training, and prediction. The data collection component would involve collecting historical data on bike share usage, such as the

**2.3.1**


BIKE SHARING MULTI AGENT SYSTEM
BIKE SHARING DATA AGENTS
TRIP DATA
STATION OCCUPANCY DATA
Init date
End date
Bike Y
User X
Station A Time T
Station A Time T + 1
<- N Bikes
SalenBiciColector Agent
CityBikeColector Agent
PybikesStationData Agent
TripDataAgent
StationDataAgent
PULL/ SUBSCRIBES
PULL PERIODICALLY
WEATHER DATA AGENTS
AEMETAgent
WEATHER AGENT
GEOGRAPHICAL INFORMATION DATA AGENTS
GoogleMapsAgent
OSRMAgent
GISAgent
DATA PERSISTENCE AGENTS
DataPersistenceAgent
JsonPersistence DataAgent
MongoDBPersistence DataAgent
DATA PROCESSOR AGENT
DataProcessorAgent
WEB API & VISUALIZATION AGENT
APIWebAgent
DEMAND PREDICTION MODEL AGENTS
DemandPredictionModelAgent
StationDeman PredictionAgent
StationUserDemand PredictionAgent

## 2.4 MAXIMUM LIKELIHOOD ESTIMATE

Maximum Likelihood Estimate (MLE) is a statistical method used to estimate the parameters of a probability distribution that best describe a given set of observations. The method is based on the principle of maximum likelihood, which states that the values of the parameters that make the observed data most probable are the best estimates of the true values of the parameters.

In practical terms, MLE involves finding the values of the parameters that maximize the likelihood function. The likelihood function represents the probability of the observed data given the values of the parameters. By maximizing the likelihood function, MLE finds the values of the parameters that make the observed data most probable.

MLE is widely used in statistical inference, including in regressl0n analysis, time series analysis, and hypothesis testing. It has several advantages, including that it is computationally efficient, asymptotically unbiased, and asymptotically efficient. However, it also has some limitations, including that it is sensitive to outliers, requires that the data is independent and identically distributed, and assumes that the distribution is known or can be accurately approximated.

Overall, MLE is a powerful tool for estimating the parameters of a probability distribution based on observed data. It is widely used in many fields, including machine learning, economics, and engineering, and has many practical applications.

## 2.4.1 MAXIMUM LIKELIHOOD ESTIMAATE FUNCTION

The function for Maximum Likelihood Estimate (MLE) algorithm varies depending on the specific statistical model being used. In general, the MLE function involves calculating the likelihood function for a given set of parameters and then finding the values of the parameters that maximize the likelihood function.

For example, if we are using a linear regression model to predict the number of bike share users based on time of day and weather conditions, the MLE function would involve fitting the linear regression model to the observed data and finding the values of the regression coefficients that maximize the likelihood of observing the data. This might involve minimizing the sum of squared residuals or using other optimization techniques.

y = 0 + 1*X1 + 2 *x2 + £

where:

y = number of bike share users

0, 1, 2 = regression coefficients

xl = time of day

x2 = weather conditions

£ = error term

The MLE function would involve finding the values of 0, 1, and 2 that maximize the likelihood function given the observed values of xl, x2, and y. This might involve using gradient descent, the Newton-Raphson

method, or other optimization techniques to iteratively find the values of the regression coefficients that maximize the likelihood function.

Overall, the specific form of the MLE function depends on the statistical model being used and the specific problem being solved. However, in general, the MLE function involves finding the values of the parameters that make the observed data most probable given the assumptions of the statistical model.

### 2.4.2 MAXIMUM A POSTIREORI

Maximum A Posteriori (MAP) is a statistical method used to estimate the parameters of a probability distribution based on observed data and prior knowledge about the parameters. The method is similar to Maximum Likelihood Estimate (MLE) but incorporates prior information about the parameters into the estimation process.

In practical terms, MAP involves finding the values of the parameters that maximize the posterior probability of the parameters given the observed data and prior information. The posterior probability is the conditional probability of the parameters given the data and prior knowledge. By maximizing the posterior probability, MAP finds the values of the parameters that are most probable given the data and prior knowledge.

MAP is widely used in Bayesian inference, which involves using Bayes' theorem to update prior beliefs about the parameters based on new data. It has several advantages over MLE, including that it can incorporate prior

knowledge about the parameters, provides a way to quantify uncertainty about the parameter estimates, and is robust to overfitting.

However, **MAP** also has some limitations, including that it can be sensitive to the choice of prior distribution, can be computationally expensive, and requires assumptions about the statistical model.

Overall, MAP is a powerful tool for estimating the parameters of a probability distribution based on observed data and prior knowledge. It is widely used in many fields, including machine learning, signal processing, and finance, and has many practical applications. The MAP function involves finding the values of the parameters that maximize the posterior probability given the data and prior information.

### 2.4.3 MAXIMUM A POSERIORI FUNCTION

The function for Maximum A Posteriori (MAP) algorithm also varies depending on the specific statistical model being used. In general, the MAP function involves calculating the posterior probability of the parameters given the observed data and prior knowledge and then finding the values of the parameters that maximize the posterior probability.

For example, if we are usmg a Bayesian linear regressl0n model to predict the number of bike share users based on time of day and weather conditions, the MAP function would involve incorporating prior information about the regression coefficients and finding the values of the coefficients that maximize the posterior probability. This might involve specifying a prior distribution for the regression coefficients, such

normal distribution, and usmg Bayes' theorem to update the pnor distribution based on the observed data.

In mathematical terms, the MAP function for a Bayesian linear regression model might look like this:

**y= 0+ 1*xl+ 2*x2+£**

where:

y = number of bike share users

0, 1, 2 = regression coefficients with prior distributions xl = time of day

x2 = weather conditions

£ = error term

The MAP function would involve finding the values of 0, 1, and 2 that maximize the posterior probability given the observed values of xl, x2, and y and the prior distributions for the regression coefficients. This might involve using optimization techniques such as gradient ascent or the Expectation-Maximization (EM) algorithm.

Overall, the specific form of the MAP function depends on the statistical model being used and the specific problem being solved. However, in general, the MAP function involves finding the values of the parameters that maximize the posterior probability given the observed data and prior knowledge.

## 2.5 MODULE IMPLEMENTATION

### 2.5.1 MODULE LIST

- Data collection
- Data Pre-Processing
- Feature Extraction
- Model Training
- Model Testing
- Deployment
- Maintenance

### 2.5.2 MODULE DESCRIPTION

#### 2.5.2.1 DATA COLLECTION

Data collection is the process of gathering relevant data on bike share usage, including factors such as time of day, weather conditions, and location. This data is typically collected through a variety of methods, such as surveys, sensors, or mobile applications.

The purpose of data collection is to obtain a comprehensive and accurate understanding of the factors that influence bike share usage. By collecting data on variables such as time of day, weather conditions, and location, we can gain insights into when and where bike sharing is most popular, as well as the factors that may discourage people from using bike share services.

To ensure that the collected data is accurate and representative, it is important to carefully design data collection methods and protocols. This may involve using standardized surveys, selecting appropriate sensors and data loggers, or leveraging existing data sources such as public transportation data or weather forecasts.

In addition to collecting data on bike share usage, it may also be necessary to collect data on other relevant factors, such as demographics, infrastructure, or public policy. This data can help to provide a more comprehensive understanding of the factors that influence bike share usage and can inform the development of effective strategies to promote bike sharing.

Overall, data collection is a critical first step in predicting bike share usage from data, as it provides the foundation for subsequent data analysis and model development.

### 2.5.2.2 DATA PRE-PROCESSING

The purpose of data pre-processing is to ensure that the machine learning algorithm can effectively learn from the data and make accurate predictions. By cleaning and transforming the raw data into a suitable format, we can reduce the impact of noise, missing data, or irrelevant features, and highlight the most important information that can aid in predicting bike share usage.

For example, in predicting bike share usage, we might preprocess the data by removing missing data on weather conditions, encoding categorical variables such as location or bike station, and normalizing numerical features such as temperature or humidity. By doing so, we

the machine learning algorithm can effectively learn from the data and make accurate predictions on unseen data.

Overall, data pre-processmg ls a critical step in machine learning workflows, as it enables the machine learning algorithm to learn from the data and make accurate predictions on unseen data.

### 2.5.2.3 FEATURE EXTRACTION

Feature extraction is a module in the machine learning workflow that involves identifying the most relevant features of the data that are likely to influence bike share usage. This module is an important step in machine learning workflows because not all features in the data are equally relevant, and selecting the most important features can improve model accuracy and reduce overfitting.

Feature extraction can involve techniques such as principal component analysis (PCA) or feature selection algorithms. PCA is a technique that reduces the dimensionality of the data by transforming the features into a new set of uncorrelated variables. This can be particularly useful when dealing with high-dimensional data where the number of features is much larger than the number of samples.

Feature selection algorithms, on the other hand, are techniques that select the most relevant features based on their predictive power. There are various feature selection algorithms available, such as univariate feature selection, recursive feature elimination, and model-based feature selection.

In predicting bike share usage, feature extraction might involve identifying the most important features such as time of day, weather conditions, location, and demographics. For example, we might use PCA to reduce the dimensionality of the data and identify the most important components that influence bike share usage, or we might use feature selection algorithms to select the most important features based on their predictive power.

Overall, feature extraction 1s an important step m machine learning workflows that can improve model accuracy and reduce overfitting by selecting the most relevant features in the data.

#### 2.5.2.4 MODEL TRAINING

Model training is a key module in the machine learning workflow, and involves selecting and training a machine learning model to predict bike share usage based on the features extracted in the previous step. The goal of model training is to find the best model that can accurately predict bike share usage on unseen data.

The choice of machine learning algorithm used in model training can depend on various factors, such as the size and complexity of the dataset, the number and type of features, and the performance requirements. Some common algorithms used in model training for bike share prediction include decision trees, neural networks, or regression models.

During model training, a portion of the available data is typically used for training the model, while the remaining data is reserved for model testing and validation. The model is trained using an optimization

adjusts the model parameters to mmlmlze the difference between the predicted and actual bike share usage on the training data.

The performance of the trained model can be evaluated using metrics such as mean absolute error (MAE), mean squared error (MSE), or coefficient of determination (R/\2). If the model performs well on the test data, it can be used to make predictions on new, unseen data.

For example, in bike share prediction, we might train a regression model using the features identified in the previous step, such as time of day, weather conditions, location, and demographics. The regression model can then be trained on a portion of the data, with the remaining data used for testing and validation.

Overall, model training is a critical step in the machine learning workflow, and involves selecting and training a machine learning model to predict bike share usage based on the features extracted in the previous step. By finding the best model that can accurately predict bike share usage, we can make more informed decisions about bike sharing programs, optimize fleet management, and improve overall user experience.

#### 2.5.2.5 MODEL TESTING

Model testing is a critical module in the machine learning workflow, and involves evaluating the performance of the trained model on a separate set of data that was not used for training. The goal of model testing is to ensure that the model can generalize well to new, unseen data and is not overfitting to the training data.

During model testing, the trained model is applied to the test data, and the predicted values are compared to the actual values to evaluate the performance of the model. The performance of the model can be evaluated using various metrics, such as mean absolute error (MAE), mean squared error (MSE), or coefficient of determination (RA2).

If the model performs well on the test data, it is an indication that the model is able to generalize well to new, unseen data, and can be used to make predictions in real-world scenarios. On the other hand, if the model performs poorly on the test data, it may indicate that the model Is overfitting to the training data and may not perform well on new data.

Model testing can also be used to compare the performance of different models or variations of the same model. By comparing the performance of different models, we can select the best model that is most suitable for the problem at hand.

For example, in bike share prediction, we might test the performance of a regression model that was trained on a portion of the data, on a separate set of test data. We would then compare the predicted bike share usage to the actual bike share usage on the test data, and evaluate the performance of the model using metrics such as mean absolute error (MAE).

Overall, model testing Is an important step m the machine learning workflow, as it helps to ensure that the trained model is able to generalize well to new, unseen data, and is not overfitting to the training data. By selecting the best model that can accurately predict bike share.

### 2.5.2.6 MAINTENANCE

Maintenance is a crucial module in the machine learning workflow, which involves continuous monitoring and updating of the deployed model to ensure its performance remains optimal over time. This may include retraining the model periodically with new data, incorporating new features or algorithms, and tracking performance metrics to detect any issues or changes in model performance. By maintaining the model, we can ensure that it continues to provide accurate predictions and remains relevant to the changing conditions of the problem domain.

## 2.6 SYSTEM SPECIFICATION

### 2.6.1 H/W SYSTEM CONFIGURATION

- Processor-Intel13 7$^{th}$ GEN
- RAM-4 GB (min)
- Hard Disk-20 GB

### 2.6.2 S/W SYSTEM CONFIGURATION

- Operating System-Windows 7 or 8
- Software: R studio

## 2.7 SOFTWARE ENVIRONMENT

### 2.7.1 R technology:

**R** is a popular open-source programming language and software environment for statistical computing and graphics. It is widely used in data analysis, statistical modeling, and machine learning applications due to its powerful set of libraries and tools for data manipulation, visualization, and modeling. **R** offers a wide range of packages for data processing and analysis, making it an excellent choice for data scientists, statisticians, and researchers. It also has a strong community of users who contribute to its development and offer support through online forums and resources. With its ease of use and flexibility, **R** has become a go-to tool for data analysis and modeling in various industries.

### 2.7.2 R Environment:

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes

- an effective data handling and storage facility,
- a suite of operators for calculations on arrays, in particular matrices,
- a large, coherent, integrated collection of intermediate tools for data analysis,
- graphical facilities for data analysis and display either on-screen or on hardcopy, and

- a well-developed, simple and effective programmmg language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

The term "environment" is intended to characterize it as a fully planned and coherent system, rather than an incremental accretion of very specific and inflexible tools, as is frequently the case with other data analysis software.

R, like S, is designed around a true computer language, and it allows users to add additional functionality by defining new functions. Much of the system is itself written in the R dialect of S, which makes it easy for users to follow the algorithmic choices made. For computationally- intensive tasks, C, C++ and Fortran code can be linked and called at run time. Advanced users can write C code to manipulate **R** objects directly.

Many users think of R as a statistics system. We prefer to think of it as an environment within which statistical techniques are implemented. R can be extended (easily) via packages. There are about eight packages supplied with the R distribution and many more are available through the **CRAN** family of Internet sites covering a very wide range of modem statistics.

R has its own LaTeX-like documentation format, which is used to supply comprehensive documentation, both on-line in a number of formats and in hardcopy.

R programming is used as a leading tool for machine learning, statistics, and data analysis. Objects, functions, and packages can easily be created byR.

It's a platform-independent language. This means it can be applied to all operating system.

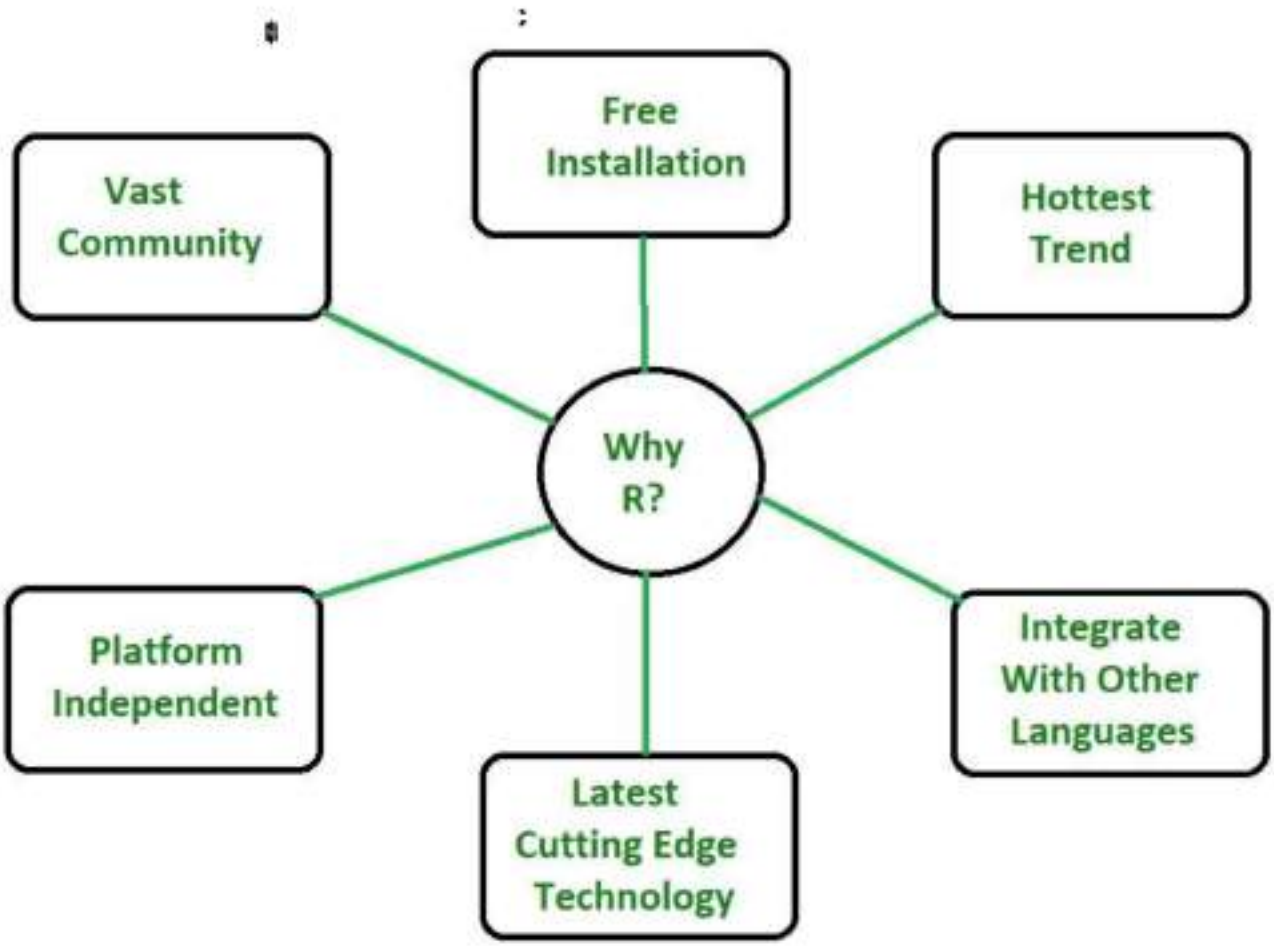
It's an open-source free language. That means anyone can install it in any organization without purchasing a license.

R programming language is not only a statistic package but also allows us to integrate with other languages (C, C++). Thus, you can easily interact with many data sources and statistical packages.

The R programming language has a vast community of users and it's growing day by day.

**2.7.2.1**

**R Features**



### 2.7.3 Statistical Features of R:

Basic Statistics: The most common basic statistics terms are the mean, mode, and median. These are all known as "Measures of Central Tendency." So using the R language we can measure central tendency very easily.

Static graphics: **R** is rich with facilities for creating and developing interesting static graphics. **R** contains functionality for many plot types including graphic maps, mosaic plots, biplots, and the list goes on.

Probability distributions: Probability distributions play a vital role m statistics and by using R we can easily handle various types of probability distribution such as Binomial Distribution, Normal Distribution, Chi- squared Distribution and many more.

Data analysis: It provides a large, coherent and integrated collection of tools for data analysis.

Programming Features otR:

**R** Packages: One of the major features of **R** is it has a wide availability of libraries. **R** has CRAN(Comprehensive **R** Archive Network), which is a repository holding more than 10, 0000 packages.

Distributed Computing: Distributed computing 1s a model in which components of a software system are shared among multiple computers to improve efficiency and performance. Two new packages ddR and multidplyr used for distributed programming in R were released in November 2015.

### 2.7.4 Programming in R:

Since **R** is much similar to other widely used languages syntactically, it is easier to code and learn in R. Programs can be written in R in any of the widely used IDE like R Studio, Rattle, Tinn-R, etc. After writing the

program save the file with the extension .r. To run the program use the following command on the command line:

**R** file name.r

Example:

**R**

```
# R program to print Welcome to GFG!
# Below line will print "Welcome to GFG!"
cat("Welcome to GFG!")
```

Output:

```
Welcome to GFG!
```

### 2.7.5 Advantages of R:

**R** is the most comprehensive statistical analysis package. As new technology and concepts often appear first in R.

As **R** programming language is an open source. Thus, you can run R anywhere and at any time.

**R** programming language 1s suitable for GNU/Linux and Windows operating system.

**R** programming is cross-platform which runs on any operating system. **In R,** everyone is welcome to provide new packages, bug fixes, and code enhancements.

### 2.7.6 Disadvantages of R:

In the R programming language, the standard of some packages is less than perfect.

Although, R commands give little pressure to memory management. So **R** programming language may consume all available memory.**In** R basically, nobody to complain if something doesn't work.

**R** programming language is much slower than other programming languages such as Python and MATLAB.

### 2.7.7 Applications of R:

We use **R** for Data Science. It gives us a broad variety of libraries related to statistics. It also provides the environment for statistical computing and design.

**R** is used by many quantitative analysts as its programming tool. Thus, it helps in data importing and cleaning.

**R** is the most prevalent language. So many data analysts and research programmers use it. Hence, it is used as a fundamental tool for finance. Tech giants like Google, Facebook, bing, Twitter, Accenture, Wipro and many more using **R** nowadays.

## 2.8 R ABSTRACT SYNTAX

Abstract syntax refers to the structure of a programming language that is independent of the concrete syntax or syntax used to represent the language in source code. In R programming, the abstract syntax of the language is defined by a set of grammar rules that describe the structure and semantics of valid R expressions and statements.

The abstract syntax of **R** consists of various types of expressions and statements, including arithmetic expressions, logical expressions, function calls, control flow statements, and more. Each expression or statement has a specific syntax and set of rules that determine how it can be used within a program.

R's abstract syntax is designed to be flexible and extensible, allowing users to define their own functions and data structures. The language also provides a variety of built-in functions and packages for data analysis and statistical modeling, making it a powerful tool for data scientists and researchers.

Overall, understanding the abstract syntax of R is essential for writing efficient and effective code in the language, and for taking full advantage of its capabilities.

### 2.8.1 RLIBRARIES

**R** libraries are collections of pre-built functions, data sets, and other resources that extend the functionality of the R programming language. These libraries provide a wide range of tools and techniques for data analysis, statistical modeling, and visualization, and can help users to streamline their workflow and improve their productivity.

**Some of the most popular R libraries include:**

ggplot2 - for creating high-quality data visualizations

dplyr - for data manipulation and cleaning

tidyr - for data tidying and reshaping

caret- for machine learning and predictive modeling

lubridate - for working with date and time data

magrittr - for chaining multiple functions together in a pipeline

stringr - for working with text data

reshape2 - for data transformation and restructuring

readr - for reading in data from various file

formats.

tidymodels - for building and evaluating machine learning models.

R libraries can be installed using the built-in package manager, and are typically loaded into an R session using the "library" function. With access to a wide range of powerful libraries, **R** users can quickly and easily tackle a wide range of data analysis and modeling tasks.

## 2.9 DEVELOPMENT ENVIRONMENTS

Data Manipulation: R provides several implementations for manipulating data including dplyr, tidyr and data.table. These packages allow you to manipulate and transform data in various ways, such as filtering, selecting, and summarizing data.

Data Visualization: **R** offers several implementations for creating high- quality visualizations of data, such as ggplot2, lattice and base **R** graphics. These packages provide a wide range of chart types and options to customize the appearance of charts and graphs.

Machine Leaming: **R** has several implementations for building and evaluating machine learning models, including caret, randomForest, and xgboost. These packages allow you to train and test models, perform feature selection, and optimize model performance.

Text Mining: R provides several implementations for working with text data, such as tm, stringr and NLP. These packages enable you to preprocess text data, analyze text data, and build predictive models based on text data.

Web Scraping: R has several implementations for web scraping, such as rvest, httr and RCurl. These packages allow you to extract data from websites and APIs, and manipulate the data for analysis.

Time Series Analysis: R provides several implementations for working with time series data, such as zoo, xts and forecast. These packages allow you to analyze and forecast time series data, and to perform various statistical tests on time series data.

### 2.9.1Supported implementation for R

**R** is an interpreted language, which means that it can be implemented on various platforms and operating systems. Some of the supported implementations of Rare:

R for Windows: This is the official distribution of **R** for the Windows operating system. It includes the R language interpreter, the RStudio IDE, and other essential tools.

R for Mac OS X: This is the official distribution of R for the Mac OS X operating system. It includes the R language interpreter, the RStudio IDE, and other essential tools.

**R** for Linux: This is the official distribution of **R** for the Linux operating system. It includes the **R** language interpreter, the RStudio IDE, and other essential tools.

R for Hadoop: This is a specialized version ofR that is designed to run on the Hadoop distributed computing platform. It enables users to process large data sets in parallel using Hadoop's MapReduce framework.

R for Spark: This is a specialized version of R that is designed to run on the Apache Spark distributed computing platform. It enables users to process large data sets in parallel using Spark's distributed computing engme.

These are just a few of the many supported implementations of R. There are also several other distributions and customized versions of R available for specific use cases and platforms.

### 2.9.2 Unsupported implementations of R

While R has a wide range of supported implementations, there are also some unsupported implementations of the language. These may include:

**R** for mobile devices: While there are some **R** apps available for mobile devices, the full R language is not supported on these platforms due to limitations in processing power and memory.

R for web browsers: While it is possible to run R code in a web browser using tools like RStudio Server, the full R language is not supported directly in web browsers.

R for embedded systems: While R can be used on some embedded systems, the language is not well-suited for these platforms due to limitations in processing power and memory.

**R** for real-time systems: **R** is not well-suited for use in real-time systems due to the unpredictable nature of garbage collection and the potential for long-running operations.

Overall, while there are some unsupported implementations of R, the language is generally well-supported across a range of platforms and use cases.

### 2.9.3 Cross Compiler to Other Languages

**R** is primarily a language used for statistical computing and graphics, and while it is possible to compile **R** code to other languages, it is not a primary use case for the language.

There are some tools available for converting **R** code to other languages, such as the R2C converter which can convert **R** code to C. However, the resulting code may not always be efficient or easy to read, and there may be some loss of functionality or flexibility when converting to another language. Additionally, since Risa high-level language with many built- in statistical functions and libraries, it may not always be straightforward to translate these functions to another language.

Overall, while it is possible to use cross-compilers to convert R code to other languages, it may not always be the most efficient or effective way to achieve a desired outcome. It may be more appropriate to write the code directly in the desired language, or to use R alongside other languages in a broader data science or analytics workflow.

## 2.10 PERFORMANCE

### 2.10.1 API DOCUMENT GENERATORS

R API documentation generators are tools that automatically generate documentation for **R** packages and functions. Some popular R API documentation generators include:

2.10.1.1 Roxygen

2.10.1.2 Devtools

2.10.1.3 Pkgdown

2.10.1.4 RDocumentation

2.10.1.5 RD2md

### 2.10.2 USES

**R** language is a versatile language that can be used m a variety of applications, including:

**Data analysis: R** is widely used for data analysis, including data cleaning, visualization, and statistical modeling.

**Machine learning:** R has many libraries for machine learning, making it a popular choice for developing and implementing machine learning algorithms.

**Data visualization:** R has powerful data visualization capabilities, making it a great tool for creating graphs, charts, and other visualizations to communicate data insights.

**Statistical analysis: R** has many built-in functions for statistical analysis, including hypothesis testing, regression analysis, and time series analysis.

**Scientific research:** R is widely used in scientific research, particularly in fields such as biology, chemistry, and environmental science

**Finance: R** is used extensively in finance for tasks such as risk modeling, portfolio analysis, and financial forecasting.

**Social sciences: R** is also used in the social sciences for tasks such as survey analysis, network analysis, and text analysis.

Overall, R is a powerful and flexible language that can be used in many different applications, making it a popular choice for data analysts, researchers, and other professionals.

## 2.11 LIBRARY FEATURES

**R** is a highly extensible language that provides a vast collection of libraries, or packages, that extend its functionality. Some of the key features of R libraries include

**Data manipulation and analysis:** R libraries provide a wide range of functions for data manipulation and analysis, including data cleaning, filtering, sorting, and summarizing.

**Data visualization:** R libraries provide a rich set of tools for creating and customizing a wide variety of visualizations, including scatter plots, bar charts, line charts, heat maps, and more.

**Machine learning:** R libraries offer a broad range of machine learning algorithms, including supervised and unsupervised learning algorithms for classification, regression, clustering, and more.

**Statistical analysis:** R libraries provide a wide variety of statistical functions, including hypothesis testing, regression analysis, time senes analysis, survival analysis, and more.

**Web scrapmg:** R libraries provide tools for web scrapmg and data extraction, allowing users to retrieve data from websites and other online sources.

**Geospatial analysis:** R libraries provide functions for geospatial analysis, including mapping and spatial statistics.

**Text mining and natural language processing: R** libraries provide tools for text mining and natural language processing, allowing users to analyze and extract insights from large text datasets

High-performance computing: R libraries provide tools for high-performance computing, including parallel processing and distributed computing.

Overall, the extensive collection of **R** libraries makes it a powerful tool for data analysis, machine learning, and scientific computing.

# CHAPTER 3

## 3. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub- assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

### 3.1TYPES OF TESTS

#### 3.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurate!y to the documented specifications and contains clearly defined inputs and expected results.

#### 3.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens

1s specifically aimed at exposmg the problems that anse from the combination of components.

### 3.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid : identified classes of invalid input must be rejected.

Input : identified functions must be exercised.

Functions : identified classes of application outputs .

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

### 3.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links.

### 3.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the

software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

### 3.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

## 3.2 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

### 3.2.1 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

### 3.2.2 Test objectives

3.2.2.1 All field entries must work properly.

3.2.2.2 Pages must be activated from the identified link.

3.2.2.3 The entry screen, messages and responses must not be delayed.

### 3.2.3 Features to be tested

3.2.3.1 Verify that the entries are of the correct format

3.2.3.2 No duplicate entries should be allowed

3.2.3.3 All links should take the user to the correct page.

### 3.3 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or - one step up - software applications at the company level - interact without error.

**3.3.1 Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

### 3.4 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**3.4.1 Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

## CONCLUSION:

It seems that MAP is the best algorithm for this project. It is possible that RF may produce slightly better results with further tweaking of the parameters but it is too computationally expensive in my opinion. The minimum MSE I found is quite large (close to 1,000,000) but that is largely because the output variable here is a large number and squaring the error gives an even bigger number. While I don't think this is the best algorithm out there, I think being able to predict the number of bike-share counts per day with an average error of 26% could be useful for a company that needs these numbers. It might be worth looking into other algorithms however.

## REFERENCES:


1. Ahmed, F., Rose, G., & Jacob, C. (2010). Impact of weather on commuter cyclist behaviour and implications for
2. climate change adaptation. Paper presented at the Australasian Transport Research Farum, Canberra.
3. Alta Bike Share. (2011). Melbourne Bike Share survey. Melbourne: Author.
4. Alta Planning + Design. (2012). King County bike share business plan
5. . Prepared for the bike share partner
6. ship Seattle. Retrieved from
7. http://pugetsoundbikeshare.org/wp-content/uploads/2012/07/
8. KCBS_Business_Plan_FINAL.pdf
9. Bachand-Madeau, **J.,** Lee, **B. H. Y.,** & El-Geneidy, A. M. (2012). Better understanding of factors inflfluen
10. cing likelihood of using shared bicycle systems and frequency of use. Transportation Research Record:
11. Journal of the Transportation Research Board, 2314, 66-71. doi:10.3141/2314--09

12. Basch, C.H., Ethan, D., Rajan, S., Samayoa-Kozlowsky, S., & Basch, C. E. (2013). Helmet use among
13. users of the Citi bike bicycle-sharing program: A pilot study in New York City. Journal of Community
14. Health, 39(3), 503-507. doi:10.1007/s10900-013-9785-7
15. Basch, C. H., Zagnit, E. A., Rajan, S., Ethan, D., & Basch, C. E. (2014). Helmet use among cyclists in
16. New York City. Journal of Community Health, 39(5), 956-958. doi:10.1007/sl 0900-014-9836-8
17. Beecham, R., & Wood, J. (2014). Characterising group-cycling journeys using interactive graphics.
18. Transportation Research Part C: Emerging Technologies. doi:10.1016/j.trc.2014.03.007
19. BiciMAD. (2014). What Is BiciMAD. Retrieved October 8, 2014, from http://www.bicimad.com/que.
20. html
21. Buck, D., Buehler, R., Happ, P., Rawls, B., Chung, **P.,** & Borecki, N. (2013). Are bikeshare users different
22. from regular cyclists? Transportation Research Record: Journal of the Transportation Research Board,
23. 2387(1), 112-119.
24. Corcoran, J., & Li, T. (2014). Spatial analytical approaches in public bicycle sharing programs. Journal of
25. Transport Geography. doi:10.1016/j.jtrangeo.2014.09.005
26. Corcoran, J., Li, T., Rohde, D., Charles-Edwards, E., & Mateo- Babiano, D. (2014). Spatio-temporal pat
27. terns of a public bicycle sharing program: The effect of weather .

28. Davis, L. S. (2014). Rolling along the last mile: Bike-sharing programs blossom nationwide. Planning,
29. 80(5), 10-16.
30. DeMaio, P. (2009). Bike-sharing: History, impacts, models of provision, & future. Journal of Public Trans
31. portation, 12(4), 41-56.