VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JnanaSangama, Belagavi-590 018, Karnataka

Project Report on

"OLA RIDE REQUEST FORECAST USING ML"

Submitted in Partial fulfillment of the Requirements for the award of the Degree of

Bachelor of Engineering in Artificial Intelligence and Machine Learning

By:

DHRUV VASHISHTHA (1BI20AI012)

BHOOGARV MAHESHWARY (1BI20AI008)

Advanced Machine Learning Project

Under the Guidance of

Mrs. Subha Meenakshi S

Project Lead

Department of Artificial Intelligence and Machine Learning

BANGALORE INSTITUTE OF TECHNOLOGY

K. R. Road, V. V. Puram, Bengaluru-560 004

BANGALORE INSTITUTE OF TECHNOLOGY

K. R. Road, V. V. Puram, Bengaluru-560 004

Department of Artificial Intelligence and Machine Learning

CERTIFICATE

Certified that the Project work entitled "**OLA RIDE REQUEST FORECAST USING ML**" carried out by **Dhruv Vashishtha (1BI20AI012) & Bhoogarv Maheshwary (1BI20AI008)** are bonafide students of Bangalore Institute of Technology, Bangalore in partial fulfillment of the requirement of VII semester (Advanced Machine Learning Project) Bachelor of Engineering in Artificial Intelligence and Machine Learning of Visvesvaraya Technological University, Belagavi during the year 2023 – 2024. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Project report has been approved as it satisfies the academic requirements concerning the Mini Project work prescribed for the said degree.

Dr. Jyothi D. G.	**Dr. M. U. Aswath**
HOD, Dept. of AI&ML	Principal, BIT

ACKNOWLEDGEMENT

While presenting this Advanced Machine Learning Project on "**OLA Ride Request Forecast Using Python and ML**", We feel that we must acknowledge the help rendered to us by various people.

We would like to thank our Principal **Dr. M. U. Aswath**, Bangalore Institute of Technology for his support throughout this project.

We express our wholehearted gratitude to **Dr. Jyothi**, **D. G.** who is our respectable Head of Dept. of Artificial Intelligence and Machine Learning. We wish to acknowledge her valuable help and encouragement.

We sincerely acknowledge the Guidance and Constant Encouragement of our Project Guide **Ms. Subha Meenakshi S, Project Lead** for her guidance and valuable advice at every stageof our project which helped us in the successful completion of the project.

Dhruv Vashishtha
Bhoogarv Maheshwary

ABSTRACT

This project focuses on addressing the business challenges faced by Ola Bikes, particularly the issue of operational losses and competition due to inefficient allocation of drivers to meet ride requests. The goal is to predict ride demand for a specified region and future time window, allowing Ola to intelligently allocate drivers and optimize their service.

The dataset includes essential information such as unique user IDs, ride request timestamps in IST, and geographical coordinates (latitude and longitude) for pickup and drop locations. To ensure the accuracy of the predictions, certain guidelines have been established by Ola's business team. These guidelines include considerations for multiple bookings from the same location within a specific time frame, handling ride requests within a short time interval, detecting potential fraud rides based on geodesic distance, and identifying system errors or rides outside the serviceable area.

The objective is to develop a demand forecasting model that takes into account these guidelines and accurately predicts the number of ride requests for a given region and future time window. The model should effectively filter out invalid or fraudulent requests while considering the temporal and spatial aspects of ride bookings.

By successfully addressing these challenges, Ola aims to enhance its operational efficiency, reduce losses, and improve its competitive edge by providing a more reliable and responsive service to users, especially in key regions such as Karnataka. The model's predictions will enable Ola to strategically allocate resources, ensuring optimal coverage and responsiveness to user demands within the specified geographical constraints.

CONTENTS

CHAPTER 1

INTRODUCTION

Arthur Samuel, a pioneer in the field of artificial intelligence and computer gaming, coined the term “Machine Learning”. He defined machine learning as – a “Field of study that gives computers the capability to learn without being explicitly programmed”. The process starts with feeding good quality data and then training our machines(computers) by building machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data do we have and what kind of task we are trying to automate.

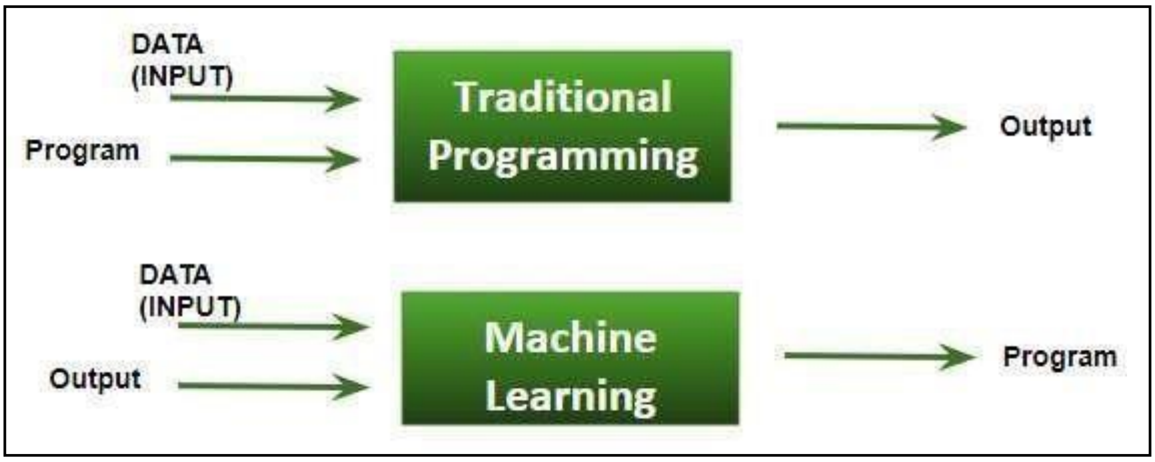


Figure 1.1: Difference between the traditional and machine learning

How does ML work?

- Gathering past data in any form suitable for processing. The better the quality of data, the more suitable it will be for modeling
- Data Processing – Sometimes, the data collected is in raw form and it needs to be pre-processed. Example: Some tuples may have missing values for certain attributes, and, in this case, it has to be filled with suitable values in order to perform machine learning or any form of data mining. Missing values for numerical attributes such as the price of the house may be replaced with the mean value of the attribute whereas missing values for categorical attributes

may be replace with the attribute with the highest mode. This invariably depends on the types of filters we use. If data is in the form of text or images then converting it to numerical form will be required, be it a list or array or matrix.

- Divide the input data into training, cross-validation, and test sets. The ratio between the respective sets must be 6:2:2
- Building models with suitable algorithms and techniques on the training set.
- Testing our conceptualized model with data that was not fed to the model at the time of training and evaluating its performance using metrics such as F1 score, precision, and recall.
  - Linear Algebra
  - Statistics and Probability
  - Calculus
  - Graph theory
  - Programming Skills – Languages such as Python, R, MATLAB, C++, or Octave.

Types of Machine Learning

Based on the methods and way of learning, machine learning is divided into mainly four types, which are:

1. Supervised Machine Learning
2. Unsupervised Machine Learning
3. Semi-Supervised Machine Learning
4. Reinforcement Learning

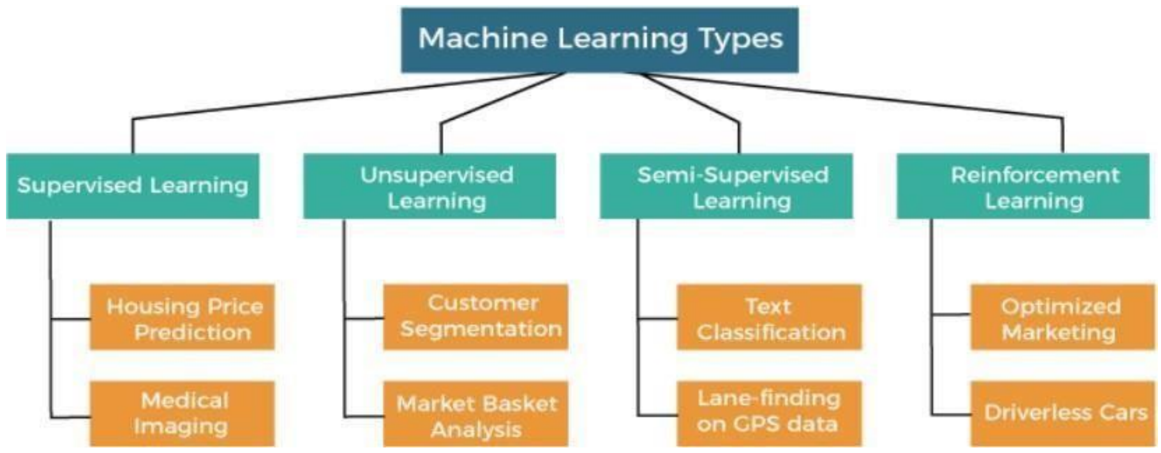


Figure 1.2: Types of Machine Learning

Supervised Machine Learning

As its name suggests, Supervised machine learning is based on supervision. It means in the supervised learning technique, we train the machines using the "labelled" dataset, and based on the training, the machine predicts the output. Here, the labeled data specifies that some of the inputs are already mapped to the output. More preciously, we can say; first, we train the machine with the input and corresponding output, and then we ask the machine to predict the output using the test dataset.

The main goal of the supervised learning technique is to map the input variable(x) with the output variable(y). Some real-world applications of supervised learning are Risk Assessment, Fraud Detection, Spam filtering, **etc.**

Categories of Supervised Machine Learning

Supervised machine learning can be classified into two types of problems, which are given below:

- Classification
- Regression

a) Classification

Classification algorithms are used to solve the classification problems in which the output variable is categorical, such as "**Yes" or No,** Male or Female, Red or Blue, etc. The classification algorithms predict the categories present in the dataset. Some real-world examples of classification algorithms are Spam Detection, Email filtering, etc.

Some popular classification algorithms are given below:

- Random Forest Algorithm
- Decision Tree Algorithm
- Logistic Regression Algorithm
- Support Vector Machine Algorithm

b) Regression

Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables. These are used to predict continuous output variables, such as market trends, weather prediction, etc.

Some popular Regression algorithms are given below:

- Simple Linear Regression Algorithm
- Multivariate Regression Algorithm
- Decision Tree Algorithm
- Lasso Regression

Advantages and Disadvantages of Supervised Learning

Advantages:

- Since supervised learning work with the labelled dataset so we can have an exact idea about the classes of objects.
- These algorithms are helpful in predicting the output on the basis of prior experience.

Disadvantages:

- These algorithms are not able to solve complex tasks.
- It may predict the wrong output if the test data is different from the training data.
- It requires lots of computational time to train the algorithm.

Unsupervised Machine Learning

Unsupervised learning is different from the Supervised learning technique; as its name suggests, there is no need for supervision. It means, in unsupervised machine learning, the machine is trained using the unlabeled dataset, and the machine predicts the output without any supervision.

In unsupervised learning, the models are trained with the data that is neither classified nor labeled, and the model acts on that data without any supervision.

The main aim of the unsupervised learning algorithm is to group or categories the unsorted dataset according to the similarities, patterns, and differences**.** Machines are instructed to find the hidden patterns from the input dataset.

Categories of Unsupervised Machine Learning

Unsupervised Learning can be further classified into two types, which are given below:

- Clustering
- Association

1) Clustering

The clustering technique is used when we want to find the inherent groups from the data. It is a way to group the objects into a cluster such that the objects with the most similarities remain in one group and have fewer or no similarities with the objects of other groups. An example of the clustering algorithm is grouping the customers by their purchasingbehavior.

Some of the popular clustering algorithms are given below:

- K-Means Clustering algorithm
- Mean-shift algorithm
- DBSCAN Algorithm
- Principal Component Analysis
- Independent Component Analysis

2) Association

Association rule learning is an unsupervised learning technique, which finds interesting relations among variables within a large dataset. The main aim of this learning algorithm is to find the dependency of one data item on another data item and map those variables accordingly so that it can generate maximum profit. This algorithm is mainly applied in Market Basket analysis, Web usage mining, continuous production**,** etc.

Some popular algorithms of Association rule learning are Apriori Algorithm, Eclat, FP-growth algorithm.

Advantages and Disadvantages of Unsupervised Learning Algorithm

Advantages:

- These algorithms can be used for complicated tasks compared to the supervised ones because these algorithms work on the unlabeled dataset.
- Unsupervised algorithms are preferable for various tasks as getting the unlabeled dataset is easier as compared to the labelled dataset.

Disadvantages:

- The output of an unsupervised algorithm can be less accurate as the dataset is not labelled, and algorithms are not trained with the exact output in prior.
- Working with Unsupervised learning is more difficult as it works with the unlabelled dataset that does not map with the output.

Semi-Supervised Learning

Semi-Supervised learning is a type of Machine Learning algorithm that lies between Supervised and Unsupervised machine learning. It represents the intermediate ground between Supervised (With Labeled training data) and Unsupervised learning (with no labeled training data) algorithms and uses the combination of labeled and unlabeled datasets during the training period.

Although Semi-supervised learning is the middle ground between supervised and unsupervised learning and operates on the data that consists of a few labels, it mostly consists of unlabeled data. As labels are costly, but for corporate purposes, they may have few labels. It is completely different from supervised and unsupervised learning as they are based on the presence & absence of labels.

To overcome the drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced. The main aim of semi-supervised learning is to effectively use all the available data, rather than only labeled data like in supervised learning. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabeled data into labelled data. It is because labeled data is a comparatively more expensive acquisition than unlabeled data.

Advantages and disadvantages of Semi-supervised Learning

Advantages:

- It is simple and easy to understand the algorithm.
- It is highly efficient.
- It is used to solve drawbacks of Supervised and Unsupervised Learning algorithms.

Disadvantages:

- Iterations results may not be stable.
- We cannot apply these algorithms to network-level data.
- Accuracy is low.

Reinforcement Learning

Reinforcement learning works on a feedback-based process, in which an AI agent (A software component) automatically explore its surrounding by hitting & trail, taking action, learning from experiences, and improving its performance**.** Agent gets rewarded for each good action and get punished for each bad action; hence the goal of reinforcement learning agent is to maximize the rewards.

In reinforcement learning, there is no labeled data like supervised learning, and agents learn from their experiences only.

The reinforcement learning process is similar to a human being; for example, a child learns various things by experiences in his day-to-day life. An example of reinforcement learning is to play a game, where the Game is the environment, moves of an agent at each step define states, and the goal of the agent is to get a high score. Agent receives feedback in terms of punishment and rewards.

Due to its way of working, reinforcement learning is employed in different fields such as Game theory, Operation Research, Information theory, multi-agent systems**.**

A reinforcement learning problem can be formalized using Markov Decision Process(MDP)**.** In MDP, the agent constantly interacts with the environment and performs actions; at each action, the environment responds and generates a new state.

Categories of Reinforcement Learning

Reinforcement learning is categorized mainly into two types of methods/algorithms:

- **Positive Reinforcement Learning:** Positive reinforcement learning specifies increasing the tendency that the required behaviour would occur again by adding something. It enhances the strength of the behaviour of the agent and positively impacts it.
- **Negative Reinforcement Learning:** Negative reinforcement learning works exactly opposite to the positive RL. It increases the tendency that the specific behaviour would occur again by avoiding the negative condition.

Real-world Use cases of Reinforcement Learning

- **Video Games:** RL algorithms are much popular in gaming applications. It is used to gain super-human performance. Some popular games that use RL algorithms are AlphaGO and AlphaGO Zero.
- **Resource Management:** The "Resource Management with Deep Reinforcement Learning" paper showed that how to use RL in computer to automatically learn and schedule resources to wait for different jobs in order to minimize average job slowdown.
- **Robotics:** RL is widely being used in Robotics applications. Robots are used in the industrial and manufacturing area, and these robots are made more powerful with reinforcement learning. There are different industries that have their vision of building intelligent robots using AI and Machine learning technology.
- **Text Mining:** Text-mining, one of the great applications of NLP, is now being implemented with the help of Reinforcement Learning by Salesforce company.

Advantages and Disadvantages of Reinforcement Learning

Advantages

- It helps in solving complex real-world problems which are difficult to be solved by general techniques.
- The learning model of RL is similar to the learning of human beings; hence most accurate results can be found.
- Helps in achieving long term results.

Disadvantage

- RL algorithms are not preferred for simple problems.
- RL algorithms require huge data and computations.
- Too much reinforcement learning can lead to an overload of states which can weaken the results.

The curse of dimensionality limits reinforcement learning for real physical systems.

Comparison of Machine Learning Algorithms

Comparing machine learning algorithms is important in itself, but there are some not-so-obvious benefits of comparing various experiments effectively.

- **Better performance**

  The primary objective of model comparison and selection is definitely better performance of the machine learning software/solution. The objective is to narrow down on the best algorithms that suit both the data and the business requirements.

- **Longer lifetime**

  High performance can be short-lived if the chosen model is tightly coupled with the training data and fails to interpret unseen data. So, it’s also important to find the model that understands underlying data patterns so that the predictions are long-lasting and the need for re-training is minimal.

- **Easier retraining**

  When models are evaluated and prepared for comparisons, minute details, and metadata get recorded which come in handy during retraining. For example, if a developer can clearly retrace the reasons behind choosing a model, the causes of model failure will immediately pop out and re-training can start with equal speed.

- **Speedy production**

  With the model details available at hand, it’s easy to narrow down on models that can offer high processing speed and use memory resources optimally. Also during production, several parameters are required to configure the machine learning solutions. Having production-level data can be useful for easily aligning with the production engineers. Moreover, knowing the resource demands of different algorithms, it will also be easier to check their compliance and feasibility with respect to the organization’s allocated assets.

Loss Functions and Metrics for Regression:

- **Mean Square Error:** measures the average of the squares of the errors or deviations, that is, the difference between the estimated and true value. It aids in imposing higher weights on outliers, thus reducing the issue of overfitting.

- **Mean Absolute Error:** It’s the absolute difference between the estimated value and true value. It decreases the weight for outlier errors when compared to the mean squared error.
- **Smooth Absolute Error:** It’s the absolute difference between the estimated value and true value for the predictions lying close to the real value, and it’s the square of the difference between the estimated and the true values of the outliers (or points far off from predicted values). Essentially, it’s a combination of MSE and MAE.

Metrics for Classification:

For every classification model prediction, a matrix called the confusion matrix can be constructed which demonstrates the number of test cases correctly and incorrectly classified. It looks something like this (considering 1 – Positive and 0 – Negative are the target classes):

Table 1.1: Confusion Matrix

	Actual 0	**Actual 1**
Predicted 0	**True Negatives (TN)**	**False Negatives (FN)**
Predicted 1	**False Positives (FP)**	**True Positives (TP)**

- TN: Number of negative cases correctly classified
- TP: Number of positive cases correctly classified
- FN: Number of positive cases incorrectly classified as negative
- FP: Number of negative cases correctly classified as positive

Accuracy

Accuracy is the simplest metric and can be defined as the number of test cases correctly classified divided by the total number of test cases.

Accuracy = (TP + TN)/(TP + FP + TN + FN)

It can be applied to most generic problems but is not very useful when it comes to unbalanced datasets. For instance, if we’re detecting fraud in bank data, the ratio of fraud to non-fraud cases can be 1:99. In such cases, if accuracy is used, the model will turn out to be 99% accurate by predicting all test cases as non-fraud.

This is why accuracy is a false indicator of model health, and for such a case, a metric is required that can focus on the fraud data points.

Precision

Precision is the metric used to identify the correctness of classification.

Precision = TP / (TP + FP)

Intuitively, this equation is the ratio of correct positive classifications to the total number of predicted positive classifications. The greater the fraction, the higher the precision, which means better ability of the model to correctly classify the positive class.

Recall

Recall tells us the number of positive cases correctly identified out of the total number of positive cases.

Recall = TP / (TP + FN)

F1 Score

F1 score is the harmonic mean of Recall and Precision, therefore it balances out the strengths of each. It’s useful in cases where both recall and precision can be valuable – like in the identification of plane parts that might require repairing. Here, precision will be required to save on the company’s cost (because plane parts are extremely expensive) and recall will be required to ensure that the machinery is stable and not a threat to human lives.

F1 Score = 2 * ((Precision * Recall) / (Precision + Recall))

- To predict the price of the laptops.
- An approach to receive higher accuracy.
- To build a machine learning model to classify the given problem statement.

Preprocessing

When we talk about data, we usually think of some large datasets with a huge number of rows and columns. While that is a likely scenario, it is not always the case — data could be in so many different forms: Structured Tables, Images, Audio files, Videos, etc..

Machines don’t understand free text, image, or video data as it is, they understand 1s and 0s. So it probably won't be good enough if we put on a slideshow of all our images and expect our machine learning model to get trained just by that.

A dataset can be viewed as a collection of data objects, which are often also called as a records, points, vectors, patterns, events, cases, samples, observations, or entities.

Data objects are described by a number of features, that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred, etc.. Features are often called as variables, characteristics, fields, attributes, or dimensions.

For instance, color, mileage and power can be considered as features of a car. There are different types of features that we can come across when we deal with data.

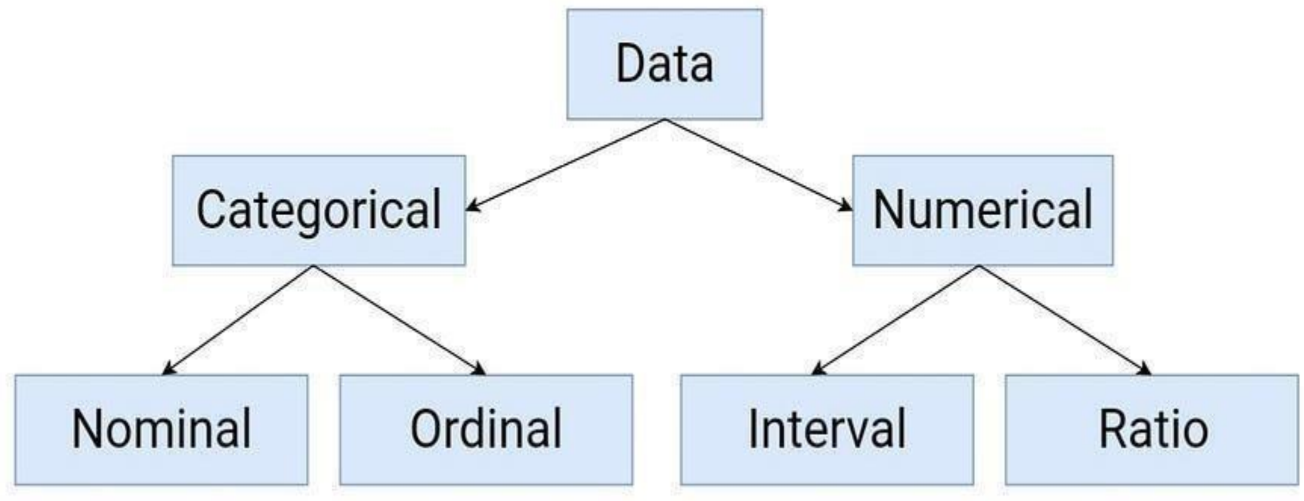
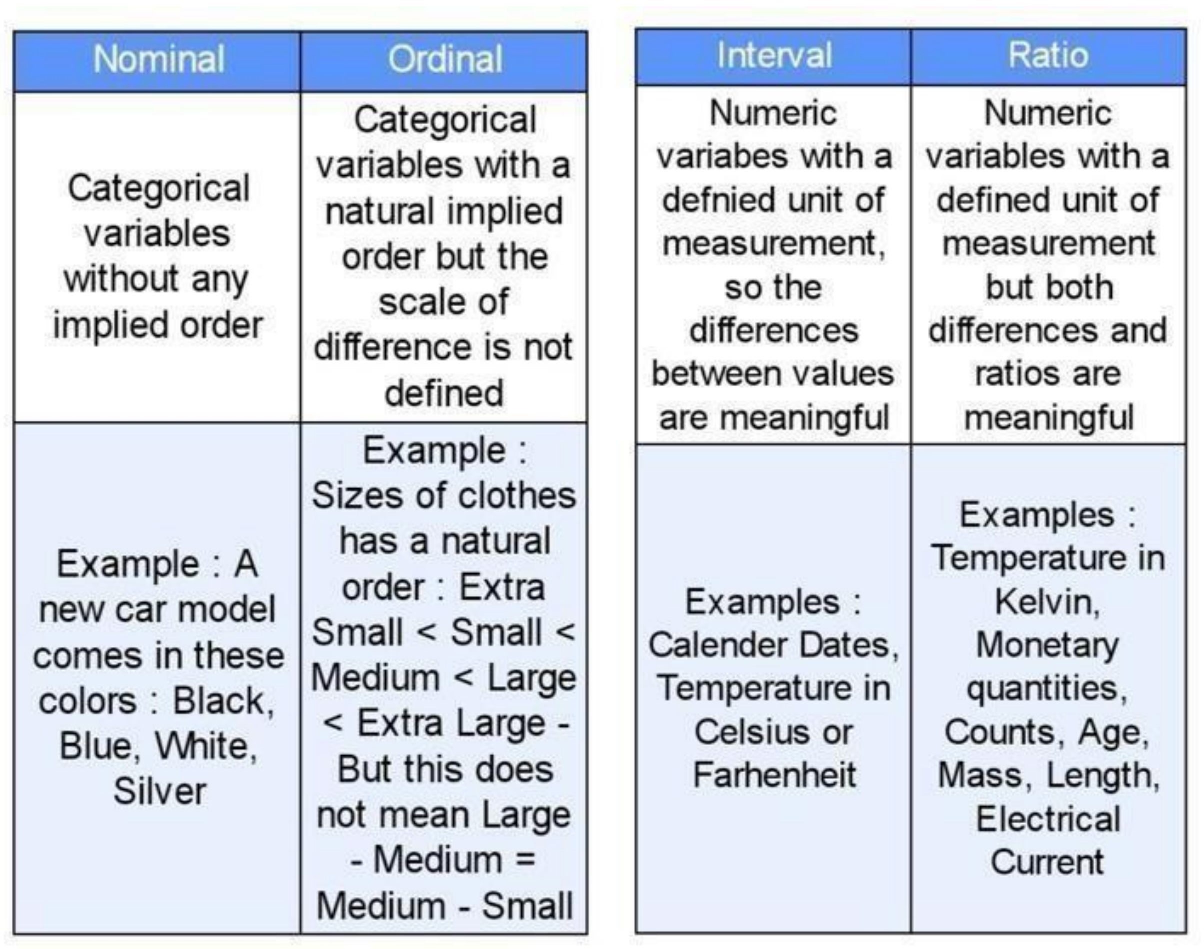


Figure 1.3: Statistical Data Types

Features can be:

- **Categorical:** Features whose values are taken from a defined set of values. For instance, days in a week: {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday,Sunday} is a category because its value is always taken from this set. Another examplecould be the Boolean set : {True, False}
- **Numerical:** Features whose values are continuous or integer-valued. They are represented by numbers and possess most of the properties of numbers. For instance, number of steps you walk in a day, or the speed at which you are driving your car at.



Nominal	Ordinal	Interval	Ratio
Categorical variables without any implied order	Categorical variables with a natural implied order but the scale of difference is not defined	Numeric variabes with a defnied unit of measurement, so the differences between values are meaningful	Numeric variables with a defined unit of measurement but both differences and ratios are meaningful
Example : A new car model comes in these colors : Black, Blue, White, Silver	Example : Sizes of clothes has a natural order : Extra Small < Small < Medium < Large < Extra Large - But this does not mean Large - Medium = Medium - Small	Examples : Calender Dates, Temperature in Celsius or Farhenheit	Examples : Temperature in Kelvin, Monetary quantities, Counts, Age, Mass, Length, Electrical Current

Figure 1.4: Feature Types

The steps of Data Preprocessing: Not all the steps are applicable for each problem, it is highly dependent on the data we are working with, so maybe only a few steps might be required with the dataset. Generally, they are:

- Data Quality Assessment
- Feature Aggregation
- Feature Sampling
- Dimensionality Reduction
- Feature Encoding

Data Quality Assessment

Because data is often taken from multiple sources which are normally not too reliable and that too in different formats, more than half our time is consumed in dealing withdata quality issues when working on a machine learning problem. It is simply unrealistic to expect that the data will be perfect. There may be problems due to human error, limitations of measuring devices, or flaws in the data collection process. The methods to deal with the problem :

1. Missing values

It is very much usual to have missing values in your dataset. It may have happened during data collection, or maybe due to some data validation rule, but regardless missing values must be taken into consideration.

- **Eliminate rows with missing data**

  Simple and sometimes effective strategy. Fails if many objects have missing values. If a feature has mostly missing values, then that feature itself can also be eliminated.

- **Estimate missing values**

  If only a reasonable percentage of values are missing, then we can also run simple interpolation methods to fill in those values. However, most common method of dealing with missing values is by filling them in with the mean, median or mode value of the respective feature.

2. Inconsistent values

The data can contain inconsistent values. For instance, the ‘Address’ field contains the ‘Phone number’. It may be due to human error or maybe the information was misread while being scanned from a handwritten form.

It is therefore always advised to perform data assessment like knowing what the data type of the features should be and whether it is the same for all the data objects.

3. Duplicate values

A dataset may include data objects which are duplicates of one another. It may happen when say the same person submits a form more than once. The term deduplication is often used to refer to the process of dealing with duplicates.

In most cases, the duplicates are removed so as to not give that particular data object an advantage or bias, when running machine learning algorithms.

Feature Aggregation

Feature Aggregations are performed so as to take the aggregated values in order to put the data in a better perspective. Think of transactional data, suppose we have day-to-day transactions of a product from recording the daily sales of that product in various store locations over the year. Aggregating the transactions to single store-wide monthly or yearly transactions will help us reducing hundreds or potentially thousands of transactions that occurdaily at a specific store, thereby reducing the number of data objects.

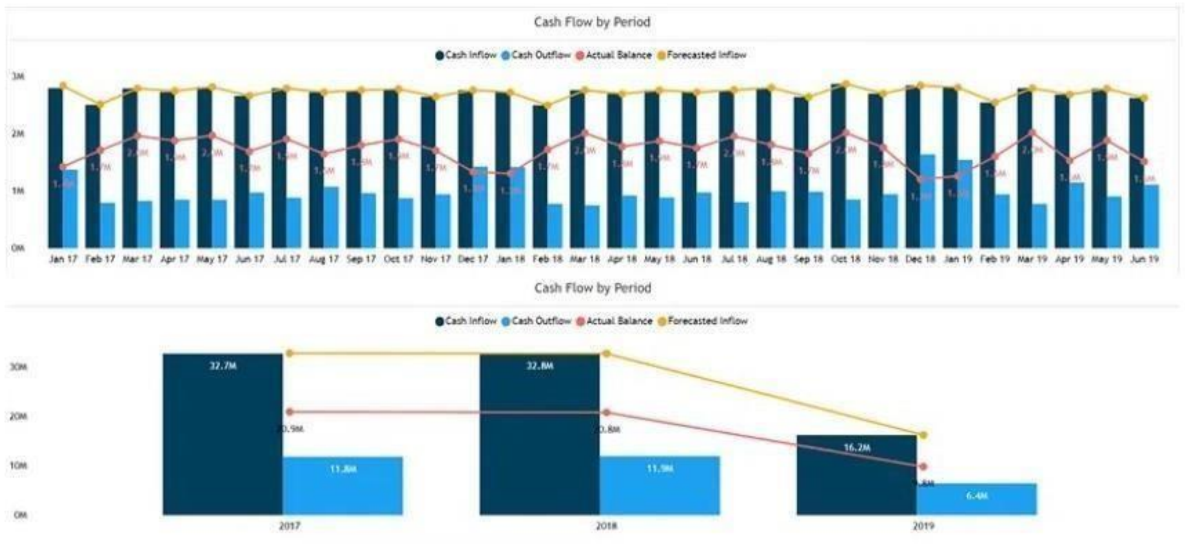


Figure 1.5: Aggregation from monthly to yearly

- This results in reduction of memory consumption and processing time.
- Aggregations provide us with a high-level view of the data as the behavior of groups or aggregates is more stable than individual data objects.

Feature Sampling

Sampling is a very common method for selecting a subset of the dataset that we are analyzing. In most cases, working with the complete dataset can turn out to be too expensive considering the memory and time constraints. Using a sampling algorithm can help us reduce the size of the dataset to a point where we can use a better, but more expensive, machine learning algorithm.

The key principle here is that the sampling should be done in such a manner that the sample generated should have approximately the same properties as the original dataset, meaning that the sample is representative. This involves choosing the correct sample size and sampling strategy.

Simple Random Sampling dictates that there is an equal probability of selecting any particular entity. It has two main variations as well:

- Sampling without Replacement : As each item is selected, it is removed from the set of all the objects that form the total dataset.
- Sampling with Replacement : Items are not removed from the total dataset after getting selected. This means they can get selected more than once.

Dimensionality Reduction

Most real world datasets have a large number of features. For example, consider an image processing problem, we might have to deal with thousands of features, also called as dimensions. As the name suggests, dimensionality reduction aims to reduce the number of features - but not simply by selecting a sample of features from the feature-set, which is something else — Feature Subset Selection or simply Feature Selection.

Conceptually, dimension refers to the number of geometric planes the dataset lies in, which could be high so much so that it cannot be visualized with pen and paper. More the number of such planes, more is the complexity of the dataset.

The Curse of Dimensionality

This refers to the phenomena that generally data analysis tasks become significantly harder as the dimensionality of the data increases. As the dimensionality increases, the number planes occupied by the data increases thus adding more and more sparsity to the data which is difficult to model and visualize.

What dimension reduction essentially does is that it maps the dataset to a lower-dimensional space, which may very well be to a number of planes which can now be visualized, say 2D. The basic objective of techniques which are used for this purpose is to reduce the dimensionality of a dataset by creating new features which are a combination of the old features. In other words, the higher-dimensional feature-space is mapped to a lower-dimensional feature-space. Principal Component Analysis and Singular Value Decomposition are two widely accepted techniques.

A few major benefits of dimensionality reduction are :

- Data Analysis algorithms work better if the dimensionality of the dataset is lower. This is mainly because irrelevant features and noise have now been eliminated.
- The models which are built on top of lower-dimensional data are more understandable and explainable.
- The data may now also get easier to visualize.

Feature Encoding

The whole purpose of data preprocessing is to encode the data in order to bring it to such a state that the machine now understands it.

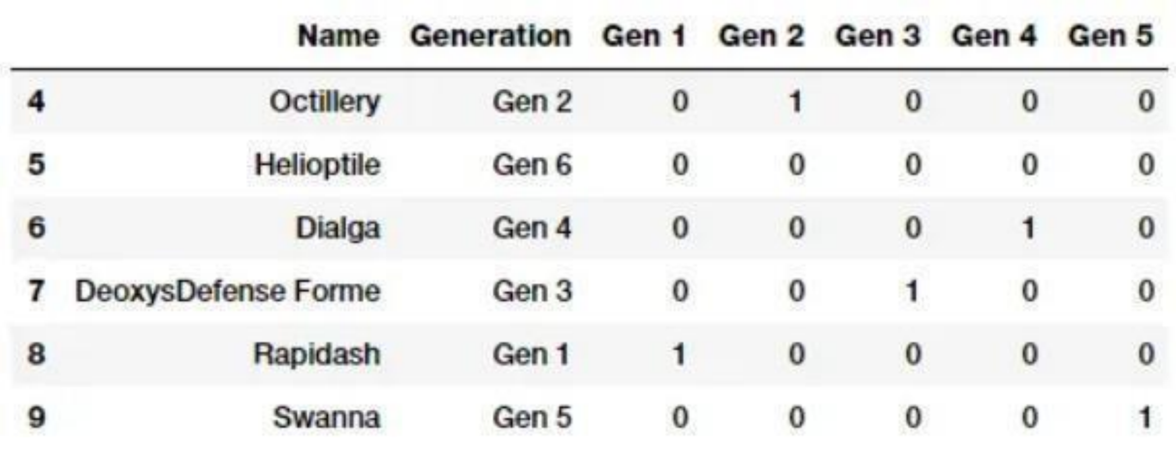
Feature encoding is basically performing transformations on the data such that it can be easily accepted as input for machine learning algorithms while still retaining its original meaning.

There are some general norms or rules which are followed when performing feature encoding.

For Continuous variables

- Nominal: Any one-to-one mapping can be done which retains the meaning. For instance, a permutation of values like in One-Hot Encoding.
- Ordinal: An order-preserving change of values. The notion of small, medium and large can be represented equally well with the help of a new function, that is, <new_value = f(old_value)> - For example, {0, 1, 2} or maybe {1, 2, 3}.

Example of One-hot encoding



	Name	Generation	Gen 1	Gen 2	Gen 3	Gen 4	Gen 5
4	Octillery	Gen 2	0	1	0	0	0
5	Helioptile	Gen 6	0	0	0	0	0
6	Dialga	Gen 4	0	0	0	1	0
7	DeoxysDefense Forme	Gen 3	0	0	1	0	0
8	Rapidash	Gen 1	1	0	0	0	0
9	Swanna	Gen 5	0	0	0	0	1

Figure 1.6: One-hot encoding of the data

For Numeric variables

- Interval: Simple mathematical transformation like using the equation <new_value = a*old_value + b>, a and b being constants. For example, Fahrenheit and Celsius scales, which differ in their Zero values size of a unit, can be encoded in this manner.
- Ratio : These variables can be scaled to any particular measures, of course while still maintaining the meaning and ratio of their values. Simple mathematical transformations work in this case as well, like the transformation <new_value = a*old_value>. For, length can be measured in meters or feet, money can be taken in different currencies.

Train / Validation / Test Split

After feature encoding is done, our dataset is ready for the exciting machine learning algorithms. But before we start deciding the algorithm which should be used, it is always advised to split the dataset into 2 or sometimes 3 parts. Machine Learning algorithms, or any algorithm for that matter, has to be first trained on the data distribution available and then validated and tested, before it can be deployed to deal with real-world data.

- **Training data:** This is the part on which your machine learning algorithms are actually trained to build a model. The model tries to learn the dataset and its various characteristics and intricacies, which also raises the issue of Overfitting v/s Underfitting.

Validation data: This is the part of the dataset which is used to validate our various model fits. In simpler words, we use validation data to choose and improve our model hyperparameters. The model does not learn the validation set but uses it to get to a better state of hyperparameters.

Test data: This part of the dataset is used to test our model hypothesis. It is left untouched and unseen until the model and hyperparameters are decided, and only after that the model is applied on the test data to get an accurate measure of how it would perform when deployed on real-world data.

Training Set
Validation Set
Testing Set

Figure 1.7: Data Split into parts

Split Ratio: Data is split as per a split ratio which is highly dependent on the type of model we are building and the dataset itself. If our dataset and model are such that a lot of training is required, then we use a larger chunk of the data just for training purposes (usually the case) For instance, training on textual data, image data, or video data usually involves thousands of features.

If the model has a lot of hyperparameters that can be tuned, then keeping a higher percentage of data for the validation set is advisable. Models with less number of hyperparameters are easy to tune and update, and so we can keep a smaller validation set.

Like many other things in Machine Learning, the split ratio is highly dependent on the problem we are trying to solve and must be decided after taking into account all the various details about the model and the dataset in hand.

Exploratory Data Analysis

Exploratory Data Analysis is a process of examining or understanding the data and extracting insights or main characteristics of the data. EDA is generally classified into two methods, i.e. graphical analysis and non-graphical analysis.

EDA is very essential because it is a good practice to first understand the problem statement and the various relationships between the data features before getting your hands dirty.

Technically, The primary motive of EDA is to

- Examine the data distribution
- Handling missing values of the dataset(a most common issue with every dataset)
- Handling the outliers
- Removing duplicate data
- Encoding the categorical variables
- Normalizing and Scaling

1.1 Problem Statement

Ola Bikes is facing operational challenges, resulting in losses and a decline in competitiveness due to inefficient driver allocation to meet user ride requests. The company seeks a solution to accurately predict ride demand for a specific region and future time window, aiming to optimize driver allocation and enhance overall service efficiency.

1.2 Objectives

- Develop an Accurate Demand Forecasting Model
- Optimize Driver Allocation
- Mitigate Fraudulent Ride Requests
- Enhance Operational Efficiency and Competitiveness

1.3 Future Scope

The future scope for Ola Bikes' predictive demand forecasting initiative includes continuous refinement of the machine learning model through the integration of real-time data sources and advanced algorithms. Exploring dynamic pricing strategies, personalized user experiences, and customer segmentation will contribute to a more responsive and user-centric service. Collaborations with urban planning authorities and the adoption of emerging technologies, such as IoT and sensors, offer avenues for improved city-wide transportation planning. Implementing a robust feedback mechanism and extending the model's applicability to global expansion strategies will ensure ongoing adaptability and scalability.

CHAPTER 2

REQUIREMENTS SPECIFICATION

2.1 SOFTWARE REQUIREMENTS

- Operating system – Windows 7/8/10/11
- Jupyter Notebook Environment
- Libraries – NumPy, Pandas, Matplotlib, and seaborn
- Language used is Python

2.2 HARDWARE REQUIREMENTS

- Processor – AMD Ryzen 7
- Processor Speed – 3201 MHz
- Memory – 16 GB RAM
- Mouse or any other pointing device
- Keyboard
- Display device: Color Monitor

CHAPTER 3

SYSTEM DEFINITION

PROJECT DESCRIPTION

Supervised Machine Learning algorithm can be broadly classified into Regression and Classification Algorithms. Regression algorithms are used to predict the output for continuous values whereas the classification algorithms are used to predict discrete values. In this project regression algorithms are used to predict the price of the laptops.

Regression analysis is often used in finance, investing, and others, and finds out the relationship between a single dependent variable(target variable) dependent on several independent ones. For example, predicting house price, stock market or salary of an employee, etc are the most common regression problems. Here, the target variable is the priceof the Laptops.

Regression Algorithms

1. Linear Regression
2. Decision Tree
3. Support Vector Regression
4. Random Forest

Linear regression

Linear Regression is an ML algorithm used for supervised learning. Linear regression performs the task to predict a dependent variable(target) based on the given independent variable(s). So, this regression technique finds out a linear relationship between a dependent variable and the other given independent variables. Hence, the name of this algorithm is Linear Regression.

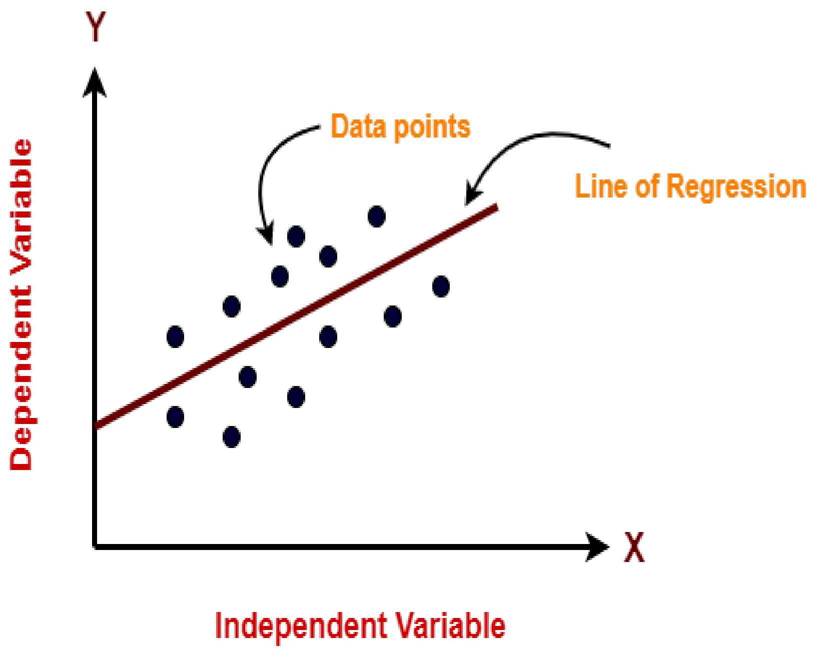


Figure 3.1: Linear Regression Algorithm

In the figure above, on X-axis is the independent variable and on Y-axis is the output. The regression line is the best fit line for a model. And our main objective in this algorithm is to find this best fit line.

Pros:

- Linear Regression is simple to implement.
- Less complexity compared to other algorithms.
- Linear Regression may lead to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization techniques, and cross-validation.

Cons:

- Outliers affect this algorithm badly.
- It over-simplifies real-world problems by assuming a linear relationship among the variables, hence not recommended for practical use-cases.

Decision Tree

The decision tree models can be applied to all those data which contains numerical features and categorical features. Decision trees are good at capturing non-linear interaction between the features and the target variable. Decision trees somewhat match human-level thinking so it’s very intuitive to understand the data.

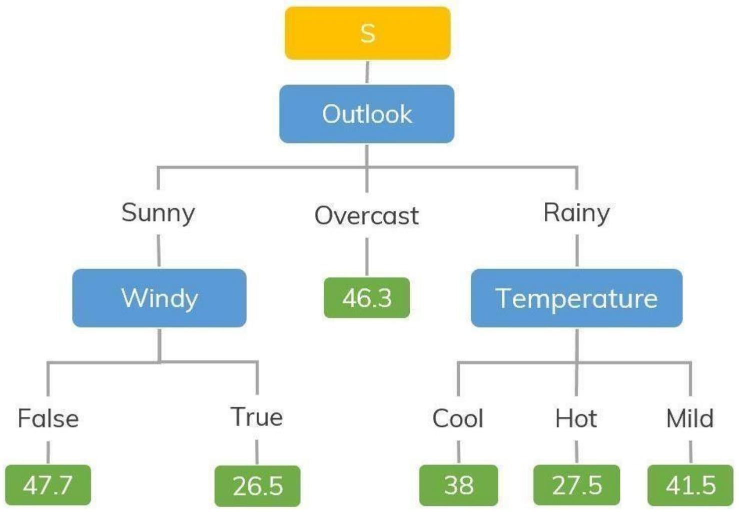


Figure 3.2: Illustration of Decision Tree

For example, if we are classifying how many hours a kid plays in particular weather then the decision tree looks like somewhat this above in the image.

So, in short, a decision tree is a tree where each node represents a feature, each branch represents a decision, and each leaf represents an outcome(numerical value for regression).

How does the Decision Tree algorithm Work?

Step 1. Begin the tree with the root node, says S, which contains the complete dataset. Step 2. Find the best attribute in the dataset using Attribute Selection Measure (ASM). Step 3. Divide the S into subsets that contains possible values for the best attributes.

Step 4. Generate the decision tree node, which contains the best attribute.

Step 5. Recursively make new decision trees using the subsets of the datasets of the dataset created in step 3.

Continue this process until a stage is reached where you cannot further classify the nodes and calledthe final node as a leaf node.

Pros:

- Easy to understand and interpret, visually intuitive.
- It can work with numerical and categorical features.
- Requires little data preprocessing: no need for one-hot encoding, dummy variables, etc.

Cons:

- It tends to overfit.
- A small change in the data tends to cause a big difference in the tree structure, which causes instability.

Support Vector Regression

SVR uses the same idea of SVM but here it tries to predict the real values. This algorithm uses hyperplanes to segregate the data. In case this separation is not possible then it uses kernel trick where the dimension is increased and then the data points become separable by a hyperplane.

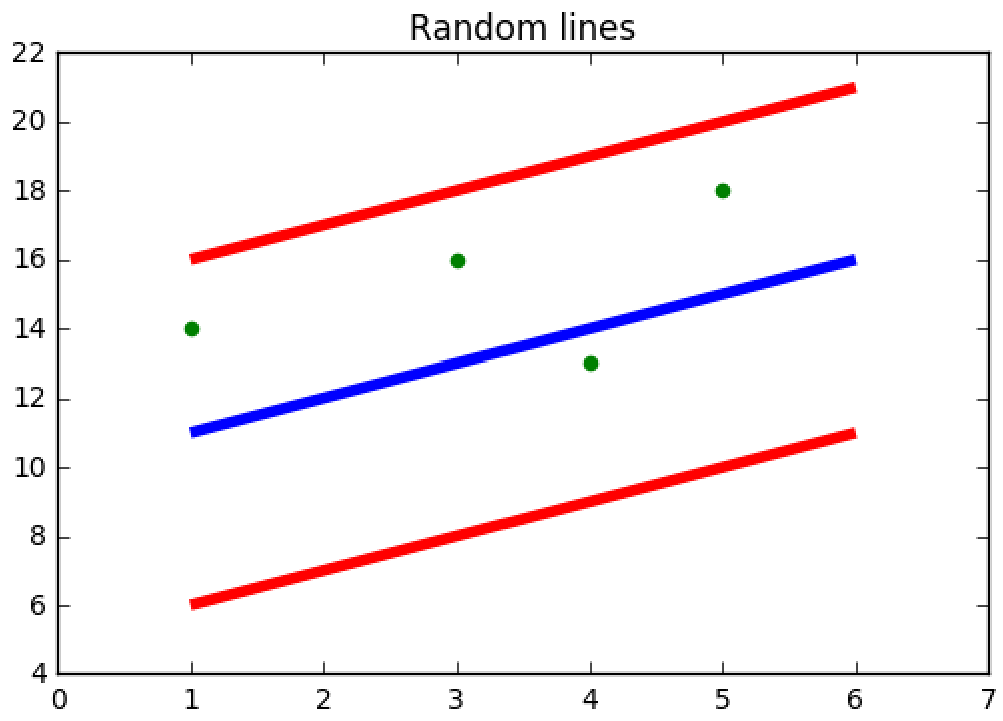


Figure 3.3: Support Vector Regression

In the figure above, the Blue line is the Hyper Plane; Red Line is the Boundary Line

All the data points are within the boundary line(Red Line). The main objective of SVR is to basically consider the points that are within the boundary line.

Pros:

- Robust to outliers.
- Excellent generalization capability
- High prediction accuracy.

Cons:

- Not suitable for large datasets.
- They do not perform very well when the data set has more noise.

Random Forest Regression

Random Forests are an ensemble(combination) of decision trees. It is a Supervised Learning algorithm used for classification and regression. The input data is passed through multiple decision trees. It executes by constructing a different number of decision trees at training time and outputting the class that is the mode of the classes (for classification) or mean prediction (for regression) of the individual trees.

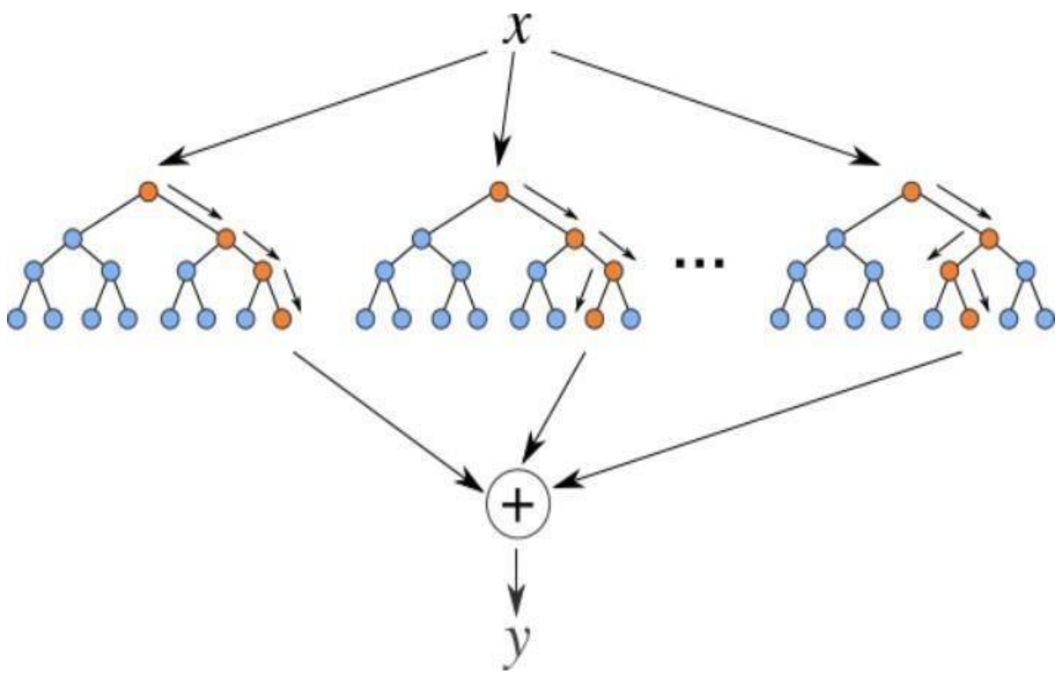


Figure 3.4: Random Forest Regression

How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps:

Step 1. Select random K data points from the training set.

Step 2. Build the decision trees associated with the selected data points.

Step 3. Choose the number N for decision trees that you want to build.

Step 4. Repeat Step 1 & 2.

Step 5. For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Pros:

- Good at learning complex and non-linear relationships
- Very easy to interpret and understand

Cons:

- They are prone to overfitting
- Using larger random forest ensembles to achieve higher performance slows down their speed and then they also need more memory.

WORKING DESCRIPTION

Credit Card Fraud Detection is the collection of data on the transactions made by credit cards. We have extracted the dataset from Kaggle.

OLA RIDE REQUEST FORECAST

The project revolves around addressing the operational challenges faced by Ola Bikes, a prominent ride-sharing service, by developing a demand forecasting model. The primary issue is the inability to efficiently fulfill ride requests, resulting in operational losses and increased competition. To tackle this, the project aims to predict ride demand for a specific region and time window, providing Ola with insights to intelligently allocate drivers and optimize service delivery. The dataset contains crucial information, including unique user IDs, timestamps for ride requests in Indian Standard Time (IST), and geographical coordinates for both pickup and drop locations. Ola's business team has provided guidelines to enhance the accuracy of the predictions and improve the overall effectiveness of the model. These guidelines cover scenarios such as handling multiple bookings from the same location within a specified time frame, managing ride requests within a short time interval, identifying potential fraud based on geodesic distance, and detecting system errors or rides outside the serviceable area.

The primary objective is to build a demand forecasting model that incorporates these guidelines, enabling accurate predictions of ride requests for a defined region and future time window. The model's capabilities will extend to filtering out invalid or fraudulent requests while considering both temporal and spatial dimensions of ride bookings.

Context of Dataset

Raw Data contains a number (unique for every user), ride request Date Time (IST time), pickup and drop location latitude, and longitude.

Data Preprocessing

Data preprocessing is an important step before using it. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific model to train.

1. Data Cleaning:
   - Removing duplicates, handle missing values, and address any inconsistencies or errors in the dataset.
2. Feature Engineering:
   - Creating relevant features like transaction amount normalization, time-based features,

and identify potential indicators of fraud.

3. Balancing the Dataset:

- Addressing class imbalance by employing techniques such as oversampling, undersampling, or using algorithms like SMOTE to balance the number of fraud and non-fraud cases.

4. Dimensionality Reduction:

- Utilizing techniques like PCA (Principal Component Analysis) to reduce the number of features and retain essential information for model training and evaluation.

Training and Testing Split

Before splitting the data for training and testing, we have to assign the response variable and predictor variable to Y and X respectively. Now we have to split the data inan 80:20 ratio. 80% of the data will be used for training the models and 20%of the datawill be used for testing.

Performing Regression

Prepare the model using the X_train and y_train (training data) using the following algorithms:

- Decision Tree
- Random Forest
- Support Vector Regression

With the prepared model ,test that with the 20% (x_test) testing data and assign that to the y_pred variable Now test the performance of the model using root squared mean error and r2 score.

Libraries Used:

- **NumPy**

  Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

- **Pandas**

  Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance & productivity for users.

- **Matplotlib**

  Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

- **Seaborn**

  Seaborn is a library mostly used for statistical plotting in Python. It is built on top of Matplotlib and provides beautiful default styles and color palettes to make statistical plots more attractive.

TECHNOLOGY USED

Machine Learning

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and using data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data,known as "training data”, in order to make predictions or decisions without being explicitly programmed to do so.

Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

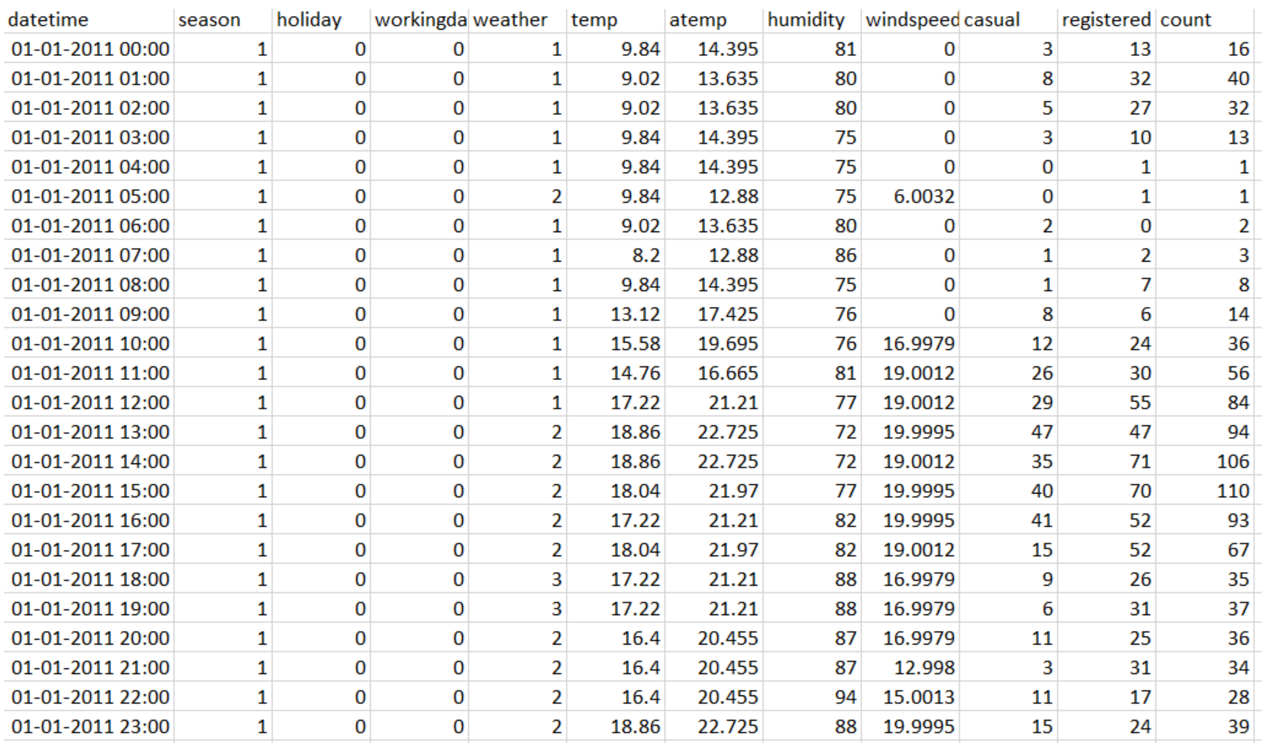
LANGUAGE USED

Python

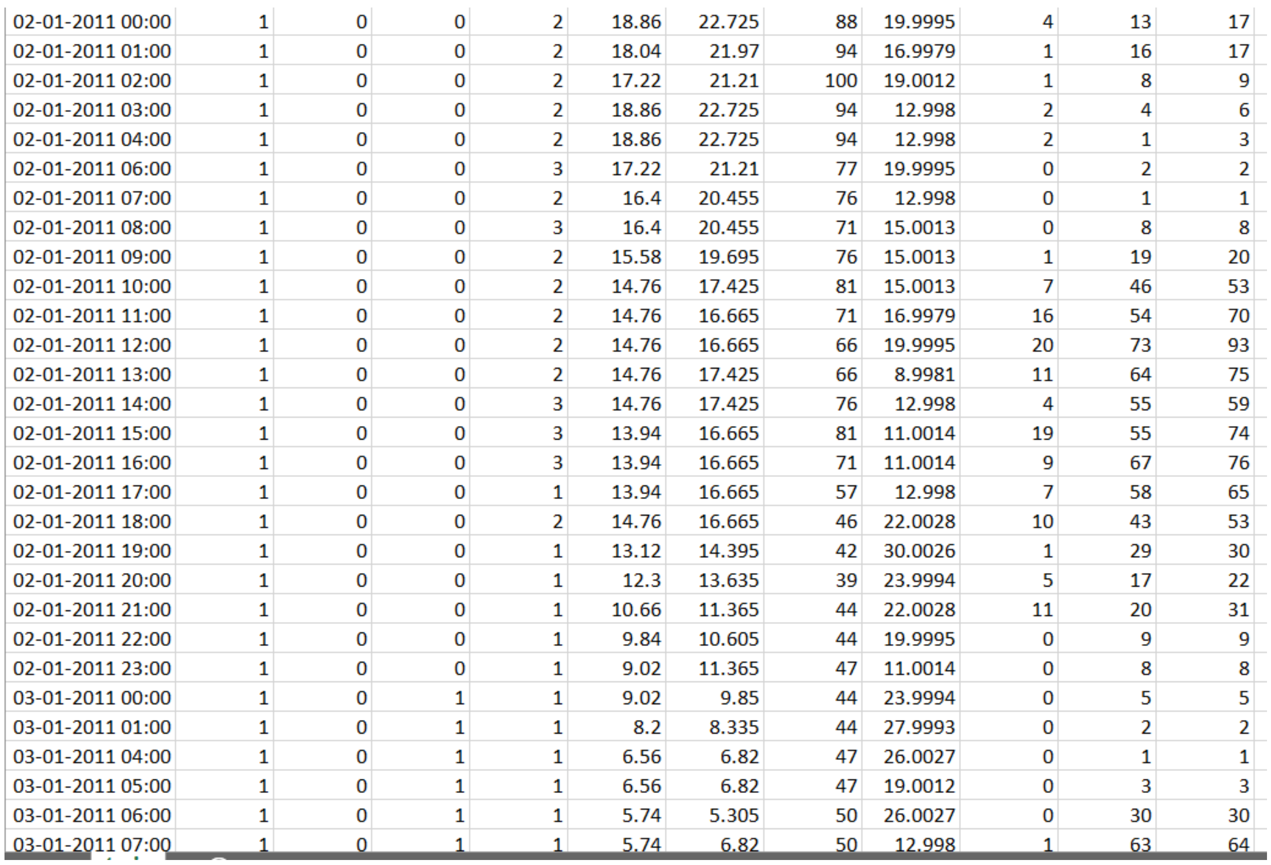
Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Python Programming Language is very well suited for Beginners, also for experienced programmers with other programming languages like C++ and Java.

DATASET

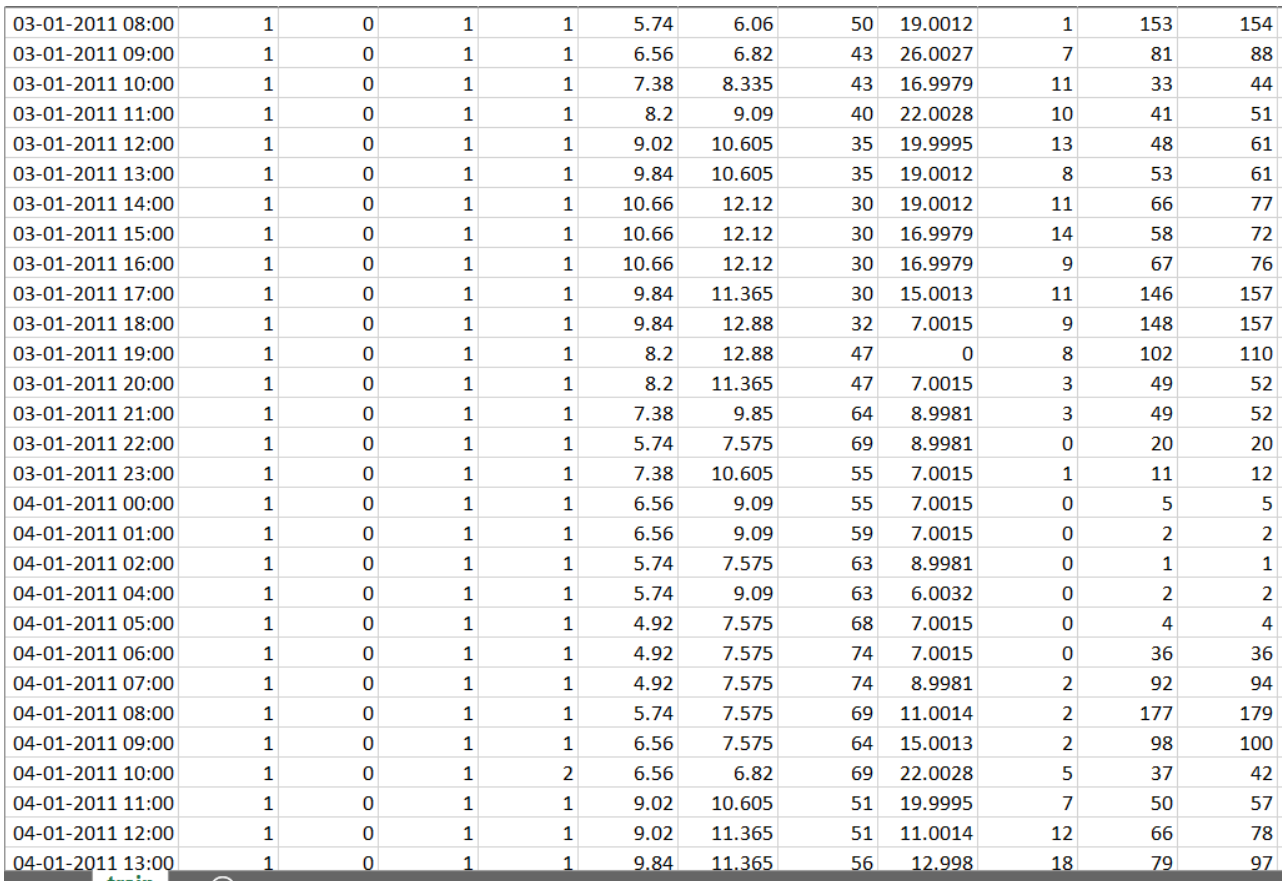
For this project, I have used the dataset extracted from kaggle. The dataset given by the source is fairly accurate and is taken from European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. Snapshot of part of the dataset is given below



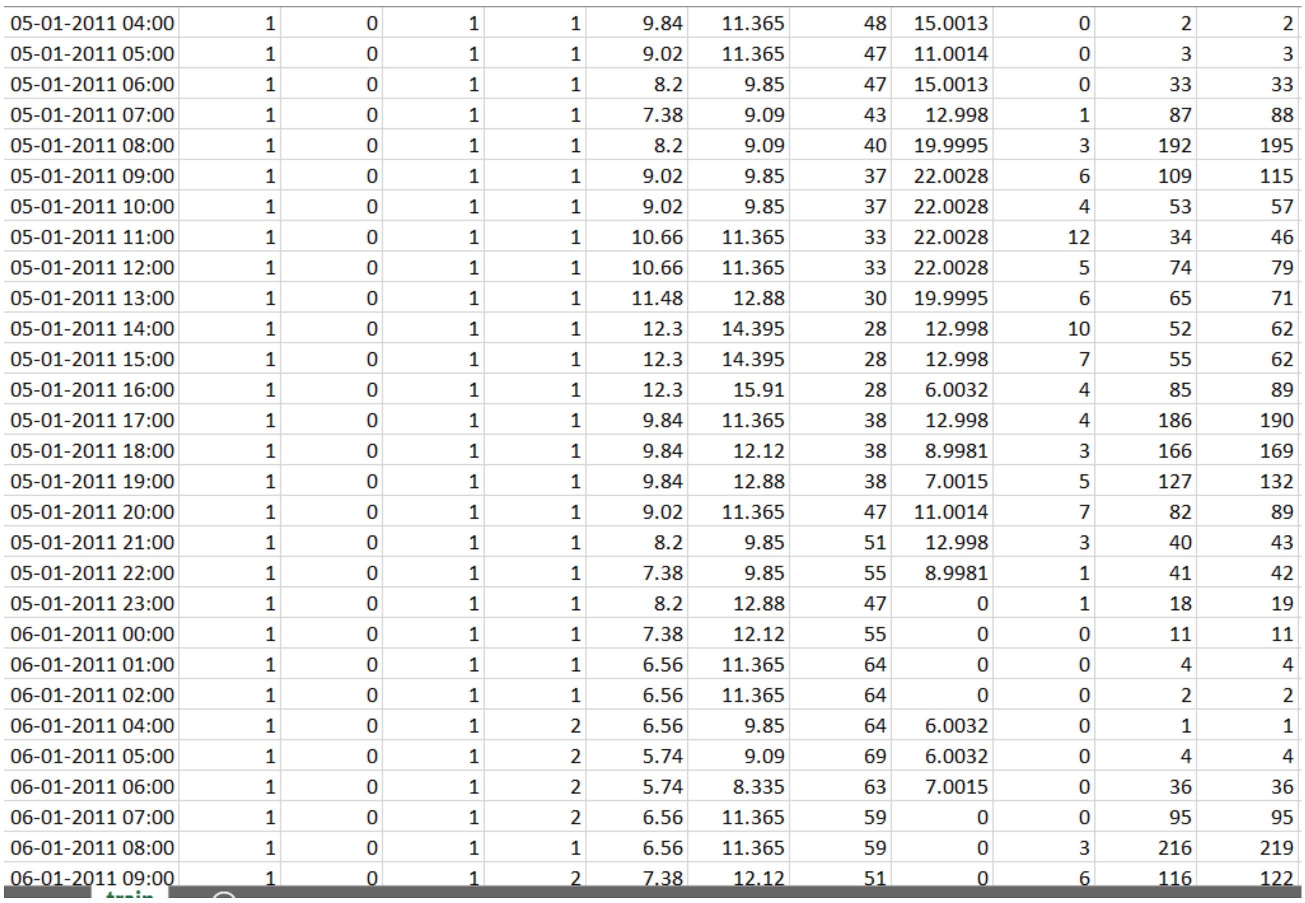
datetime	season	holiday	workingda	weather	temp	atemp	humidity	windspeed	casual	registered	count
01-01-2011 00:00	1	0	0	1	9.84	14.395	81	0	3	13	16
01-01-2011 01:00	1	0	0	1	9.02	13.635	80	0	8	32	40
01-01-2011 02:00	1	0	0	1	9.02	13.635	80	0	5	27	32
01-01-2011 03:00	1	0	0	1	9.84	14.395	75	0	3	10	13
01-01-2011 04:00	1	0	0	1	9.84	14.395	75	0	0	1	1
01-01-2011 05:00	1	0	0	2	9.84	12.88	75	6.0032	0	1	1
01-01-2011 06:00	1	0	0	1	9.02	13.635	80	0	2	0	2
01-01-2011 07:00	1	0	0	1	8.2	12.88	86	0	1	2	3
01-01-2011 08:00	1	0	0	1	9.84	14.395	75	0	1	7	8
01-01-2011 09:00	1	0	0	1	13.12	17.425	76	0	8	6	14
01-01-2011 10:00	1	0	0	1	15.58	19.695	76	16.9979	12	24	36
01-01-2011 11:00	1	0	0	1	14.76	16.665	81	19.0012	26	30	56
01-01-2011 12:00	1	0	0	1	17.22	21.21	77	19.0012	29	55	84
01-01-2011 13:00	1	0	0	2	18.86	22.725	72	19.9995	47	47	94
01-01-2011 14:00	1	0	0	2	18.86	22.725	72	19.0012	35	71	106
01-01-2011 15:00	1	0	0	2	18.04	21.97	77	19.9995	40	70	110
01-01-2011 16:00	1	0	0	2	17.22	21.21	82	19.9995	41	52	93
01-01-2011 17:00	1	0	0	2	18.04	21.97	82	19.0012	15	52	67
01-01-2011 18:00	1	0	0	3	17.22	21.21	88	16.9979	9	26	35
01-01-2011 19:00	1	0	0	3	17.22	21.21	88	16.9979	6	31	37
01-01-2011 20:00	1	0	0	2	16.4	20.455	87	16.9979	11	25	36
01-01-2011 21:00	1	0	0	2	16.4	20.455	87	12.998	3	31	34
01-01-2011 22:00	1	0	0	2	16.4	20.455	94	15.0013	11	17	28
01-01-2011 23:00	1	0	0	2	18.86	22.725	88	19.9995	15	24	39



02-01-2011 00:00	1	0	0	2	18.86	22.725	88	19.9995	4	13	17
02-01-2011 01:00	1	0	0	2	18.04	21.97	94	16.9979	1	16	17
02-01-2011 02:00	1	0	0	2	17.22	21.21	100	19.0012	1	8	9
02-01-2011 03:00	1	0	0	2	18.86	22.725	94	12.998	2	4	6
02-01-2011 04:00	1	0	0	2	18.86	22.725	94	12.998	2	1	3
02-01-2011 06:00	1	0	0	3	17.22	21.21	77	19.9995	0	2	2
02-01-2011 07:00	1	0	0	2	16.4	20.455	76	12.998	0	1	1
02-01-2011 08:00	1	0	0	3	16.4	20.455	71	15.0013	0	8	8
02-01-2011 09:00	1	0	0	2	15.58	19.695	76	15.0013	1	19	20
02-01-2011 10:00	1	0	0	2	14.76	17.425	81	15.0013	7	46	53
02-01-2011 11:00	1	0	0	2	14.76	16.665	71	16.9979	16	54	70
02-01-2011 12:00	1	0	0	2	14.76	16.665	66	19.9995	20	73	93
02-01-2011 13:00	1	0	0	2	14.76	17.425	66	8.9981	11	64	75
02-01-2011 14:00	1	0	0	3	14.76	17.425	76	12.998	4	55	59
02-01-2011 15:00	1	0	0	3	13.94	16.665	81	11.0014	19	55	74
02-01-2011 16:00	1	0	0	3	13.94	16.665	71	11.0014	9	67	76
02-01-2011 17:00	1	0	0	1	13.94	16.665	57	12.998	7	58	65
02-01-2011 18:00	1	0	0	2	14.76	16.665	46	22.0028	10	43	53
02-01-2011 19:00	1	0	0	1	13.12	14.395	42	30.0026	1	29	30
02-01-2011 20:00	1	0	0	1	12.3	13.635	39	23.9994	5	17	22
02-01-2011 21:00	1	0	0	1	10.66	11.365	44	22.0028	11	20	31
02-01-2011 22:00	1	0	0	1	9.84	10.605	44	19.9995	0	9	9
02-01-2011 23:00	1	0	0	1	9.02	11.365	47	11.0014	0	8	8
03-01-2011 00:00	1	0	1	1	9.02	9.85	44	23.9994	0	5	5
03-01-2011 01:00	1	0	1	1	8.2	8.335	44	27.9993	0	2	2
03-01-2011 04:00	1	0	1	1	6.56	6.82	47	26.0027	0	1	1
03-01-2011 05:00	1	0	1	1	6.56	6.82	47	19.0012	0	3	3
03-01-2011 06:00	1	0	1	1	5.74	5.305	50	26.0027	0	30	30
03-01-2011 07:00	1	0	1	1	5.74	6.82	50	12.998	1	63	64



03-01-2011 08:00	1	0	1	1	5.74	6.06	50	19.0012	1	153	154
03-01-2011 09:00	1	0	1	1	6.56	6.82	43	26.0027	7	81	88
03-01-2011 10:00	1	0	1	1	7.38	8.335	43	16.9979	11	33	44
03-01-2011 11:00	1	0	1	1	8.2	9.09	40	22.0028	10	41	51
03-01-2011 12:00	1	0	1	1	9.02	10.605	35	19.9995	13	48	61
03-01-2011 13:00	1	0	1	1	9.84	10.605	35	19.0012	8	53	61
03-01-2011 14:00	1	0	1	1	10.66	12.12	30	19.0012	11	66	77
03-01-2011 15:00	1	0	1	1	10.66	12.12	30	16.9979	14	58	72
03-01-2011 16:00	1	0	1	1	10.66	12.12	30	16.9979	9	67	76
03-01-2011 17:00	1	0	1	1	9.84	11.365	30	15.0013	11	146	157
03-01-2011 18:00	1	0	1	1	9.84	12.88	32	7.0015	9	148	157
03-01-2011 19:00	1	0	1	1	8.2	12.88	47	0	8	102	110
03-01-2011 20:00	1	0	1	1	8.2	11.365	47	7.0015	3	49	52
03-01-2011 21:00	1	0	1	1	7.38	9.85	64	8.9981	3	49	52
03-01-2011 22:00	1	0	1	1	5.74	7.575	69	8.9981	0	20	20
03-01-2011 23:00	1	0	1	1	7.38	10.605	55	7.0015	1	11	12
04-01-2011 00:00	1	0	1	1	6.56	9.09	55	7.0015	0	5	5
04-01-2011 01:00	1	0	1	1	6.56	9.09	59	7.0015	0	2	2
04-01-2011 02:00	1	0	1	1	5.74	7.575	63	8.9981	0	1	1
04-01-2011 04:00	1	0	1	1	5.74	9.09	63	6.0032	0	2	2
04-01-2011 05:00	1	0	1	1	4.92	7.575	68	7.0015	0	4	4
04-01-2011 06:00	1	0	1	1	4.92	7.575	74	7.0015	0	36	36
04-01-2011 07:00	1	0	1	1	4.92	7.575	74	8.9981	2	92	94
04-01-2011 08:00	1	0	1	1	5.74	7.575	69	11.0014	2	177	179
04-01-2011 09:00	1	0	1	1	6.56	7.575	64	15.0013	2	98	100
04-01-2011 10:00	1	0	1	2	6.56	6.82	69	22.0028	5	37	42
04-01-2011 11:00	1	0	1	1	9.02	10.605	51	19.9995	7	50	57
04-01-2011 12:00	1	0	1	1	9.02	11.365	51	11.0014	12	66	78
04-01-2011 13:00	1	0	1	1	9.84	11.365	56	12.998	18	79	97



05-01-2011 04:00	1	0	1	1	9.84	11.365	48	15.0013	0	2	2
05-01-2011 05:00	1	0	1	1	9.02	11.365	47	11.0014	0	3	3
05-01-2011 06:00	1	0	1	1	8.2	9.85	47	15.0013	0	33	33
05-01-2011 07:00	1	0	1	1	7.38	9.09	43	12.998	1	87	88
05-01-2011 08:00	1	0	1	1	8.2	9.09	40	19.9995	3	192	195
05-01-2011 09:00	1	0	1	1	9.02	9.85	37	22.0028	6	109	115
05-01-2011 10:00	1	0	1	1	9.02	9.85	37	22.0028	4	53	57
05-01-2011 11:00	1	0	1	1	10.66	11.365	33	22.0028	12	34	46
05-01-2011 12:00	1	0	1	1	10.66	11.365	33	22.0028	5	74	79
05-01-2011 13:00	1	0	1	1	11.48	12.88	30	19.9995	6	65	71
05-01-2011 14:00	1	0	1	1	12.3	14.395	28	12.998	10	52	62
05-01-2011 15:00	1	0	1	1	12.3	14.395	28	12.998	7	55	62
05-01-2011 16:00	1	0	1	1	12.3	15.91	28	6.0032	4	85	89
05-01-2011 17:00	1	0	1	1	9.84	11.365	38	12.998	4	186	190
05-01-2011 18:00	1	0	1	1	9.84	12.12	38	8.9981	3	166	169
05-01-2011 19:00	1	0	1	1	9.84	12.88	38	7.0015	5	127	132
05-01-2011 20:00	1	0	1	1	9.02	11.365	47	11.0014	7	82	89
05-01-2011 21:00	1	0	1	1	8.2	9.85	51	12.998	3	40	43
05-01-2011 22:00	1	0	1	1	7.38	9.85	55	8.9981	1	41	42
05-01-2011 23:00	1	0	1	1	8.2	12.88	47	0	1	18	19
06-01-2011 00:00	1	0	1	1	7.38	12.12	55	0	0	11	11
06-01-2011 01:00	1	0	1	1	6.56	11.365	64	0	0	4	4
06-01-2011 02:00	1	0	1	1	6.56	11.365	64	0	0	2	2
06-01-2011 04:00	1	0	1	2	6.56	9.85	64	6.0032	0	1	1
06-01-2011 05:00	1	0	1	2	5.74	9.09	69	6.0032	0	4	4
06-01-2011 06:00	1	0	1	2	5.74	8.335	63	7.0015	0	36	36
06-01-2011 07:00	1	0	1	2	6.56	11.365	59	0	0	95	95
06-01-2011 08:00	1	0	1	1	6.56	11.365	59	0	3	216	219
06-01-2011 09:00	1	0	1	2	7.38	12.12	51	0	6	116	122

CHAPTER 4

IMPLEMENTATION

The implementation is done using python language and executed in Jupyter Notebook.

Import Libraries

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sb

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn import metrics

from sklearn.svm import SVC

from xgboost import XGBRegressor

from sklearn.linear_model import LinearRegression, Lasso, Ridge

from sklearn.ensemble import RandomForestRegressor
```

Load and Prepare Data

```
data=pd.read_csv('ola.csv')

data.head()
print(data.shape)
print(data.describe())
```

Feature Engineering parts =
df["datetime"].str.split(" ", n=2,
expand=True)

```
df["date"] = parts[0]

df["time"] = parts[1].str[:2].astype('int')

df.head()

parts = df["date"].str.split("-", n=3,
expand=True)

df["day"] = parts[0].astype('int')

df["month"] = parts[1].astype('int')

df["year"] = parts[2].astype('int')

df.head()
from datetime import datetime
import calendar

def weekend_or_weekday(year, month, day):
    try:
        # Check if the day is within the valid range for the month
        _, max_day = calendar.monthrange(int(year), int(month))
        day = min(int(day), max_day)

        # Create datetime object
        d = datetime(int(year), int(month), day)

        # Check if it's a weekend (Saturday or Sunday)
        if d.weekday() >= 5:
            return 0
        else:
            return 1
    except ValueError:
        return 0  # Handle invalid date values gracefully

# Apply the function to create the 'workingday' column
df['workingday'] = df.apply(lambda x: weekend_or_weekday(x['year'], x['month'], x['day']), axis=1)
df.head()
```

```
def am_or_pm(x):
    if x > 11:
        return 1
    else:
        return 0
df['am_or_pm'] =
df['time'].apply(am_or_pm)
df.head()
from datetime import date
import holidays
def is_holiday(x):
    india_holidays =
holidays.country_holidays('IN')
    if india_holidays.get(x):
        return 1
    else:
        return 0
df['holidays'] = df['date'].apply(is_holiday)
df.head()
df.drop(['datetime', 'date'],
        axis=1,
        inplace=True)
```

Exploratory Data Analysis

```
features = ['day', 'time', 'month']


plt.subplots(figsize=(15, 10))
for i, col in enumerate(features):
   plt.subplot(2, 2, i + 1)
   df.groupby(col).mean()['count'].plot()
plt.show()
features = ['season', 'weather', 'holidays',\
            'am_or_pm', 'year', 'workingday']


plt.subplots(figsize=(20, 10))
for i, col in enumerate(features):
   plt.subplot(2, 3, i + 1)
   df.groupby(col).mean()['count'].plot.bar()
plt.show()

features = ['temp', 'windspeed']

plt.subplots(figsize=(15, 5))

for i, col in enumerate(features):

   plt.subplot(1, 2, i + 1)

   sb.distplot(df[col])

plt.show()
```

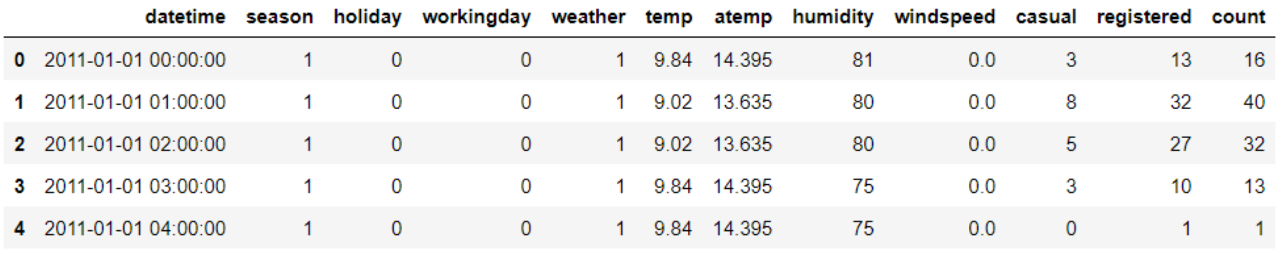
```
features = ['temp', 'windspeed']

plt.subplots(figsize=(15, 5))

for i, col in enumerate(features):

    plt.subplot(1, 2, i + 1)

    sb.boxplot(df[col])

plt.show()
num_rows = df.shape[0] -
df[df['windspeed']<32].shape[0]
print(f'Number of rows that will be lost if we
remove outliers is equal to {num_rows}.')
features = ['humidity', 'casual', 'registered', 'count']


plt.subplots(figsize=(15, 10))
for i, col in enumerate(features):
    plt.subplot(2, 2, i + 1)
    sb.boxplot(df[col])
plt.show()


 sb.heatmap(df.corr() > 0.8,

         annot=True,

         cbar=False)

 plt.show()

 df.drop(['registered', 'time'], axis=1, inplace=True)
 df = df[(df['windspeed'] < 32) & (df['humidity'] > 0)])
```

```
features = df.drop(['count'], axis=1)
target = df['count'].values


X_train, X_val, Y_train, Y_val = train_test_split(features,
                                                  target,
                                                  test_size = 0.1,
                                                  random_state=22)
X_train.shape, X_val.shape
```

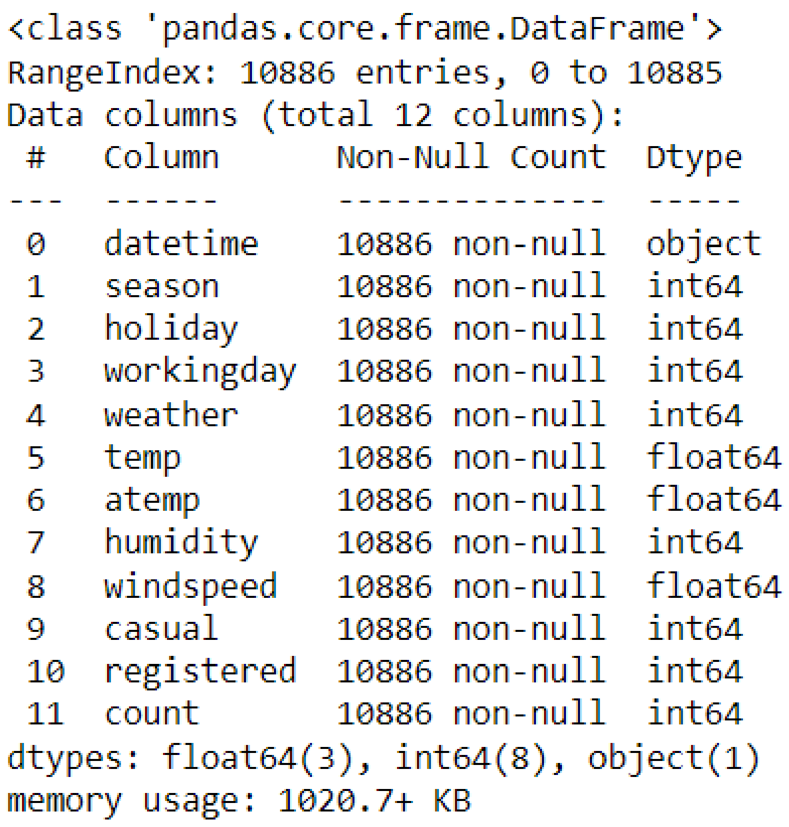
```
from sklearn.metrics import mean_absolute_error as mae
models = [LinearRegression(), XGBRegressor(), Lasso(),
        RandomForestRegressor(), Ridge()]

for i in range(5):
    models[i].fit(X_train, Y_train)

    print(f'{models[i]} : ')

    train_preds = models[i].predict(X_train)
    print('Training Error : ', mae(Y_train, train_preds))

    val_preds = models[i].predict(X_val)
    print('Validation Error : ', mae(Y_val, val_preds))
    print()
```

```
from sklearn.metrics import mean_absolute_error as mae
models = [LinearRegression(), XGBRegressor(), Lasso(),
        RandomForestRegressor(), Ridge()]

for i in range(5):
    models[i].fit(X_train, Y_train)

    print(f'{models[i]} : ')

    train_preds = models[i].predict(X_train)
    print('Training Error : ', mae(Y_train, train_preds))

    val_preds = models[i].predict(X_val)
    print('Validation Error : ', mae(Y_val, val_preds))
    print()
```

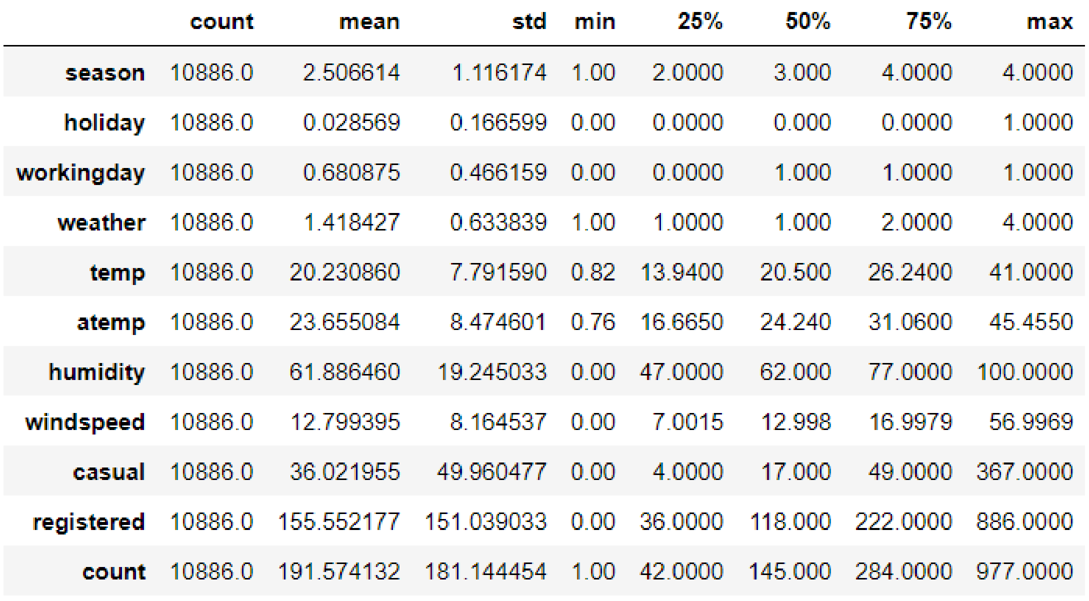
SNAPSHOTS



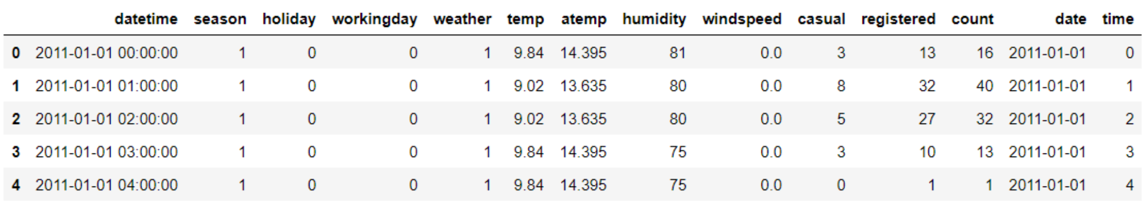
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1



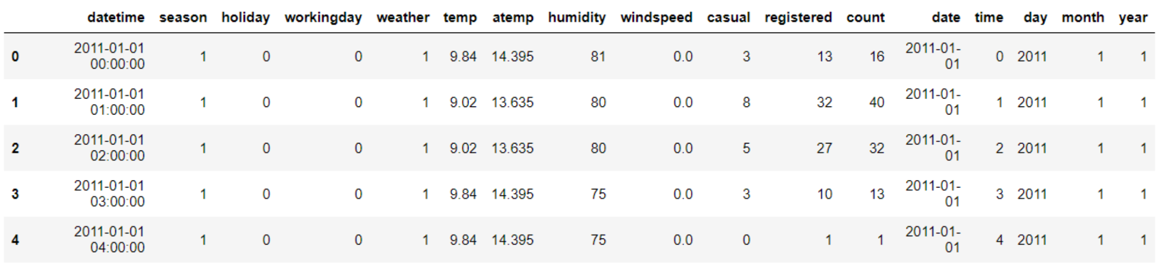
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   datetime    10886 non-null  object
 1   season      10886 non-null  int64
 2   holiday     10886 non-null  int64
 3   workingday  10886 non-null  int64
 4   weather     10886 non-null  int64
 5   temp        10886 non-null  float64
 6   atemp       10886 non-null  float64
 7   humidity    10886 non-null  int64
 8   windspeed   10886 non-null  float64
 9   casual      10886 non-null  int64
 10  registered  10886 non-null  int64
 11  count       10886 non-null  int64
dtypes: float64(3), int64(8), object(1)
memory usage: 1020.7+ KB
```



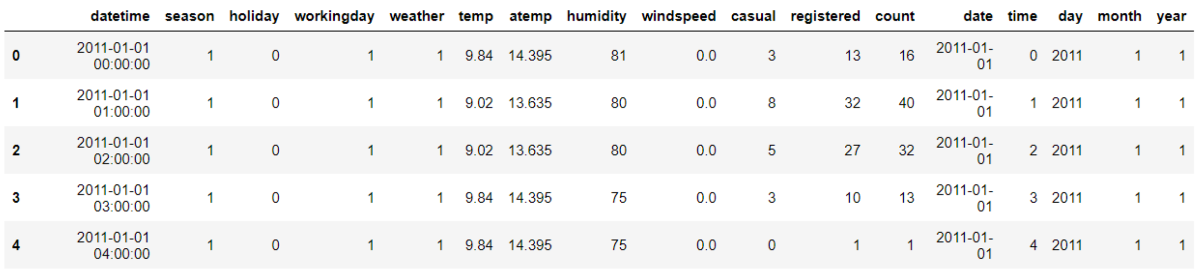
	count	mean	std	min	25%	50%	75%	max
season	10886.0	2.506614	1.116174	1.00	2.0000	3.000	4.0000	4.0000
holiday	10886.0	0.028569	0.166599	0.00	0.0000	0.000	0.0000	1.0000
workingday	10886.0	0.680875	0.466159	0.00	0.0000	1.000	1.0000	1.0000
weather	10886.0	1.418427	0.633839	1.00	1.0000	1.000	2.0000	4.0000
temp	10886.0	20.230860	7.791590	0.82	13.9400	20.500	26.2400	41.0000
atemp	10886.0	23.655084	8.474601	0.76	16.6650	24.240	31.0600	45.4550
humidity	10886.0	61.886460	19.245033	0.00	47.0000	62.000	77.0000	100.0000
windspeed	10886.0	12.799395	8.164537	0.00	7.0015	12.998	16.9979	56.9969
casual	10886.0	36.021955	49.960477	0.00	4.0000	17.000	49.0000	367.0000
registered	10886.0	155.552177	151.039033	0.00	36.0000	118.000	222.0000	886.0000
count	10886.0	191.574132	181.144454	1.00	42.0000	145.000	284.0000	977.0000



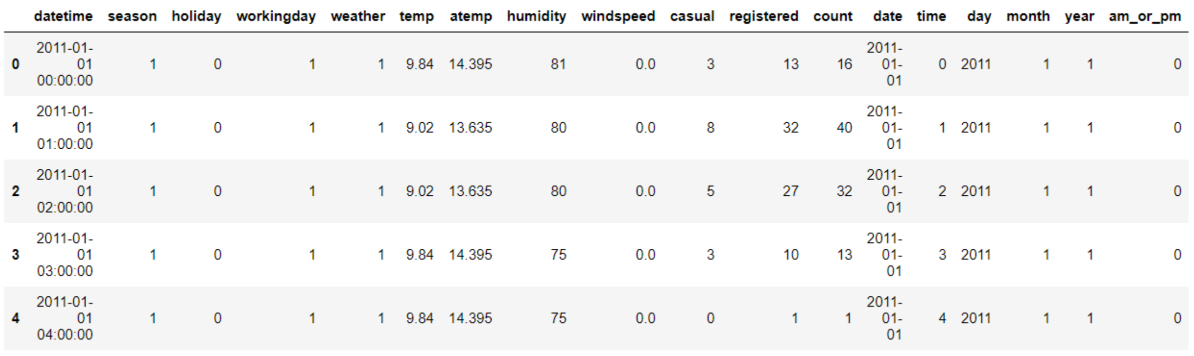
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	0
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	1
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	2
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	3
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	4



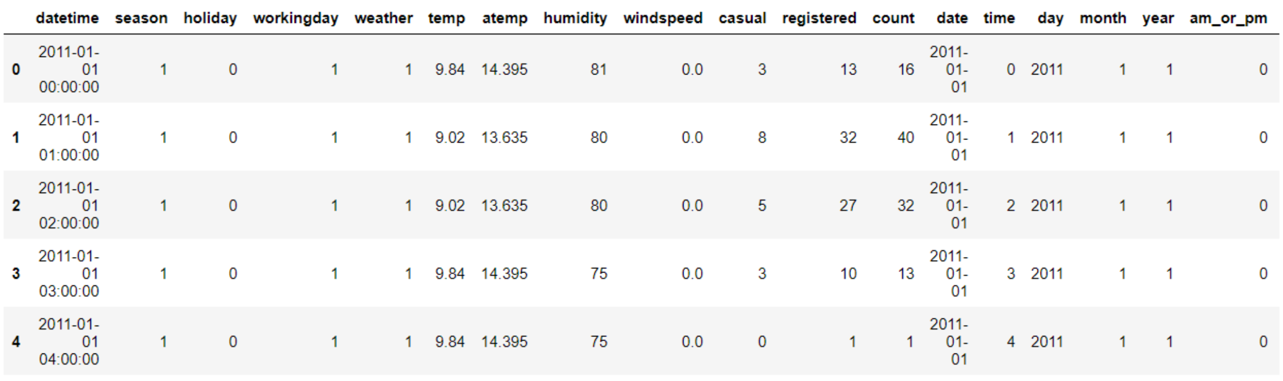
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	day	month	year
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	0	2011	1	1
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	1	2011	1	1
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	2	2011	1	1
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	3	2011	1	1
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	4	2011	1	1



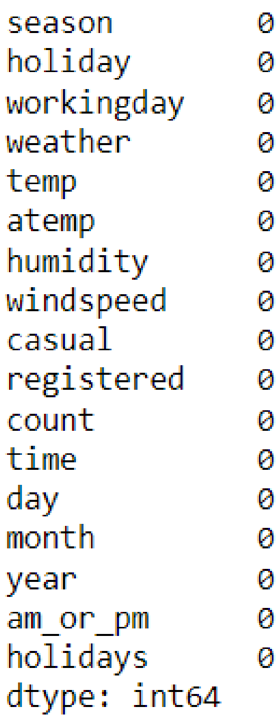
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	day	month	year
0	2011-01-01 00:00:00	1	0	1	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	0	2011	1	1
1	2011-01-01 01:00:00	1	0	1	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	1	2011	1	1
2	2011-01-01 02:00:00	1	0	1	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	2	2011	1	1
3	2011-01-01 03:00:00	1	0	1	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	3	2011	1	1
4	2011-01-01 04:00:00	1	0	1	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	4	2011	1	1



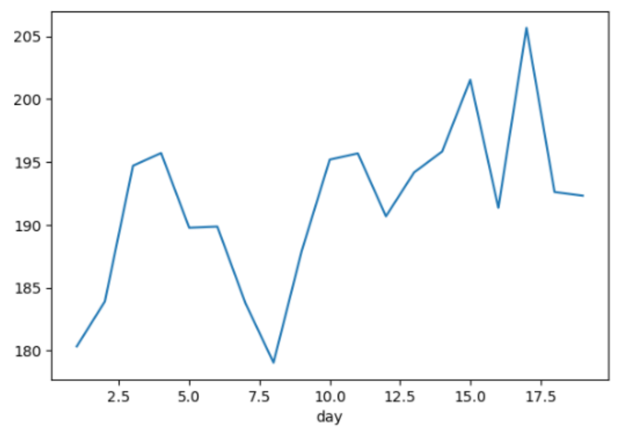
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	day	month	year	am_or_pm
0	2011-01-01 00:00:00	1	0	1	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	0	2011	1	1	0
1	2011-01-01 01:00:00	1	0	1	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	1	2011	1	1	0
2	2011-01-01 02:00:00	1	0	1	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	2	2011	1	1	0
3	2011-01-01 03:00:00	1	0	1	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	3	2011	1	1	0
4	2011-01-01 04:00:00	1	0	1	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	4	2011	1	1	0

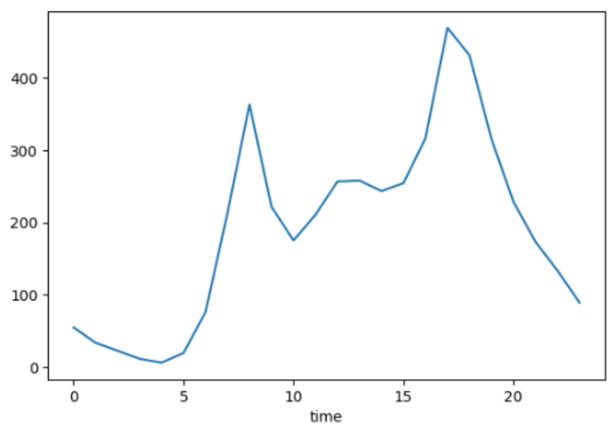


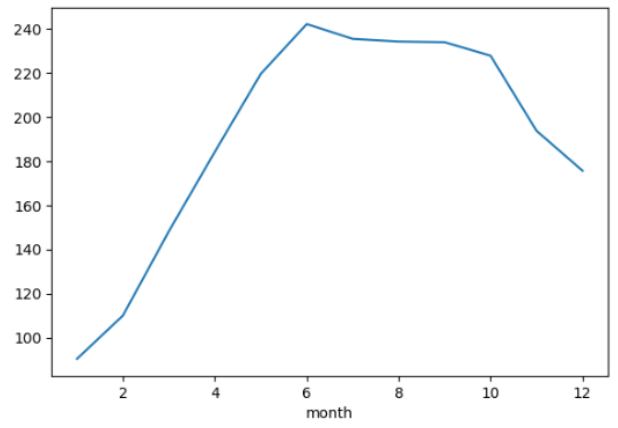
	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	date	time	day	month	year	am_or_pm
0	2011-01-01 00:00:00	1	0	1	1	9.84	14.395	81	0.0	3	13	16	2011-01-01	0	2011	1	1	0
1	2011-01-01 01:00:00	1	0	1	1	9.02	13.635	80	0.0	8	32	40	2011-01-01	1	2011	1	1	0
2	2011-01-01 02:00:00	1	0	1	1	9.02	13.635	80	0.0	5	27	32	2011-01-01	2	2011	1	1	0
3	2011-01-01 03:00:00	1	0	1	1	9.84	14.395	75	0.0	3	10	13	2011-01-01	3	2011	1	1	0
4	2011-01-01 04:00:00	1	0	1	1	9.84	14.395	75	0.0	0	1	1	2011-01-01	4	2011	1	1	0

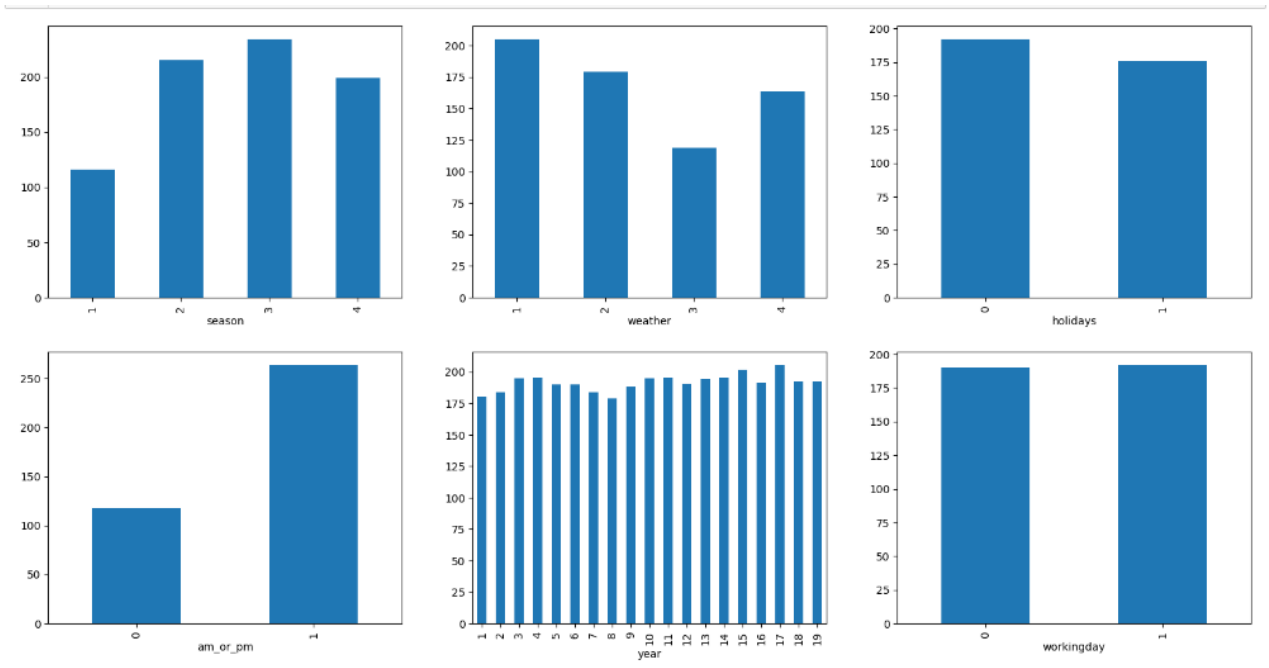


```
season         0
holiday        0
workingday     0
weather        0
temp           0
atemp          0
humidity       0
windspeed      0
casual         0
registered     0
count          0
time           0
day            0
month          0
year           0
am_or_pm       0
holidays       0
dtype: int64
```

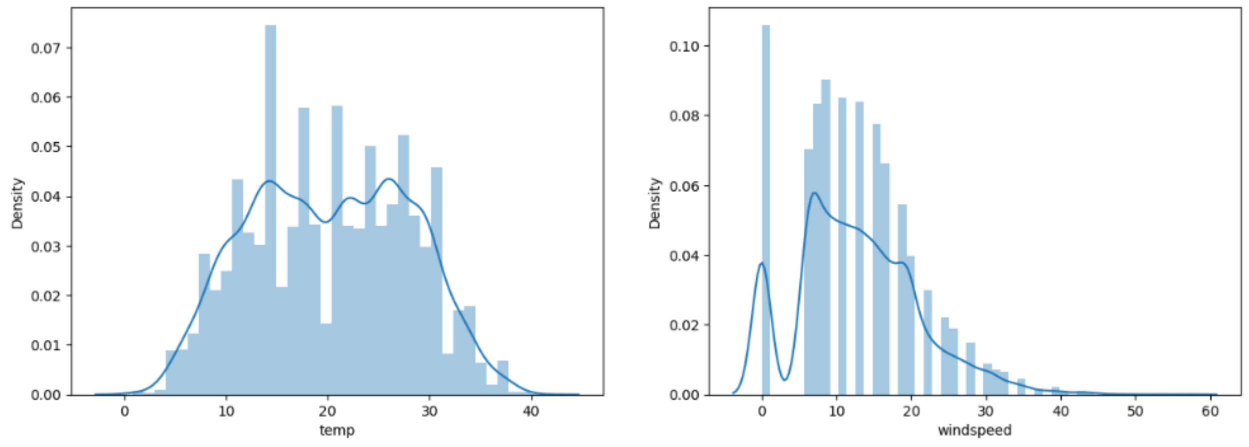

day

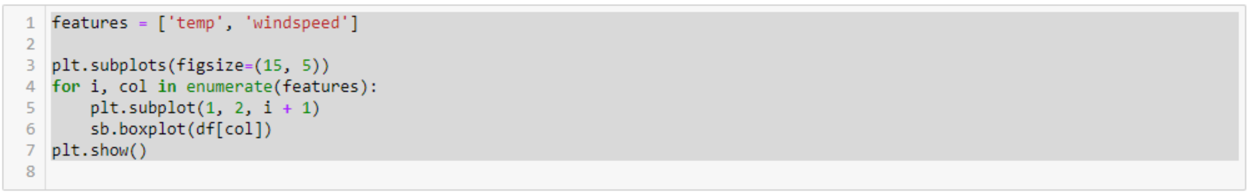

time


month

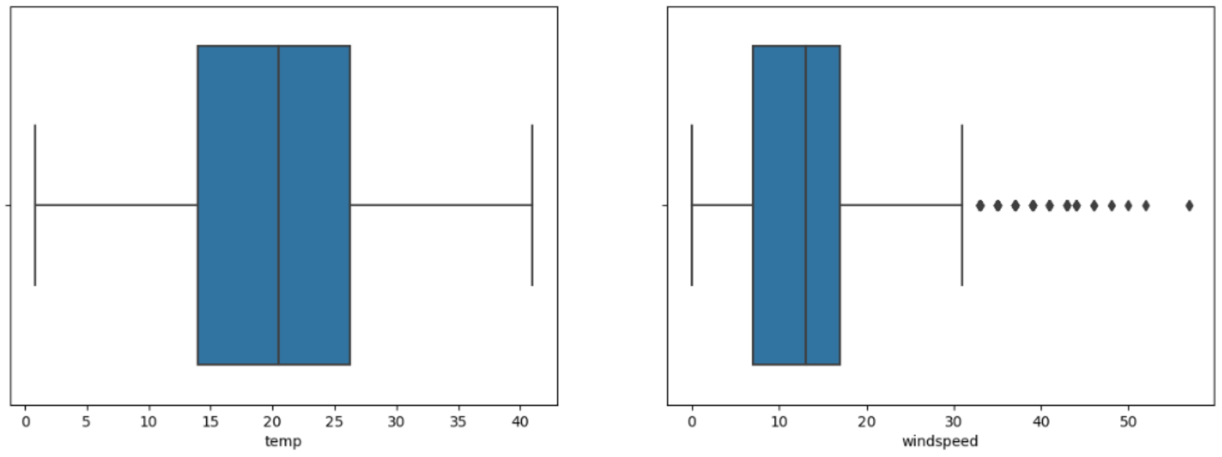

season
weather
holidays
am_or_pm
year
workingday

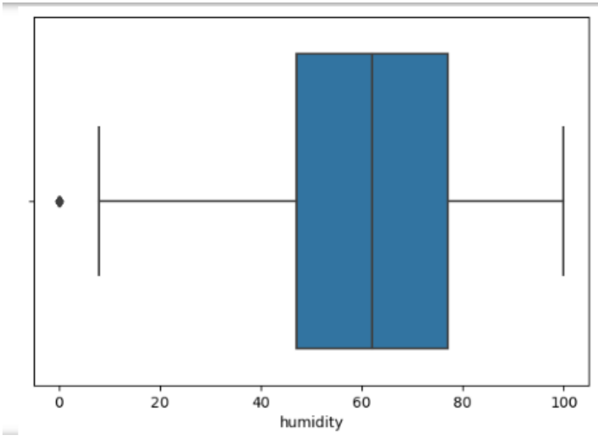
```python
features = ['temp', 'windspeed']

plt.subplots(figsize=(15, 5))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.distplot(df[col])
plt.show()
```

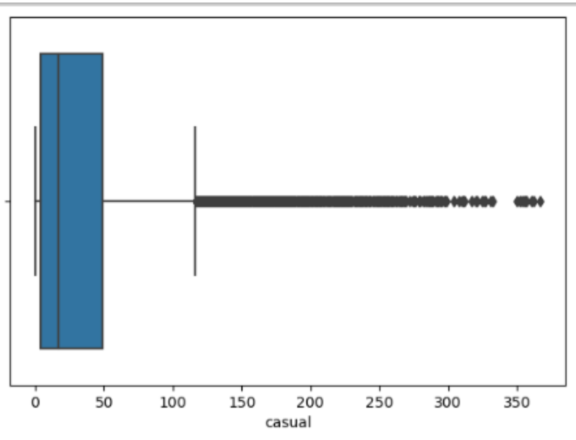


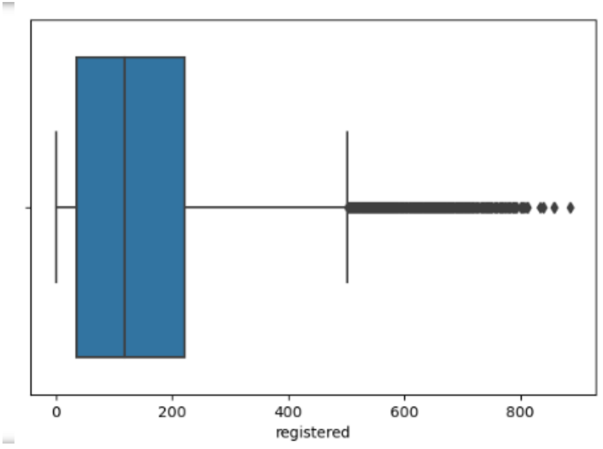


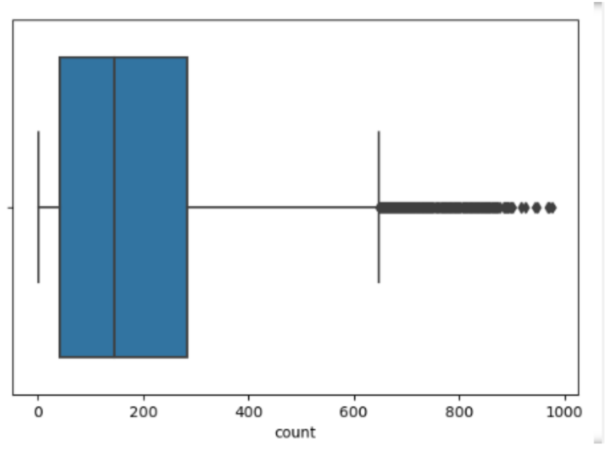
```
features = ['temp', 'windspeed']

plt.subplots(figsize=(15, 5))
for i, col in enumerate(features):
    plt.subplot(1, 2, i + 1)
    sb.boxplot(df[col])
plt.show()
```

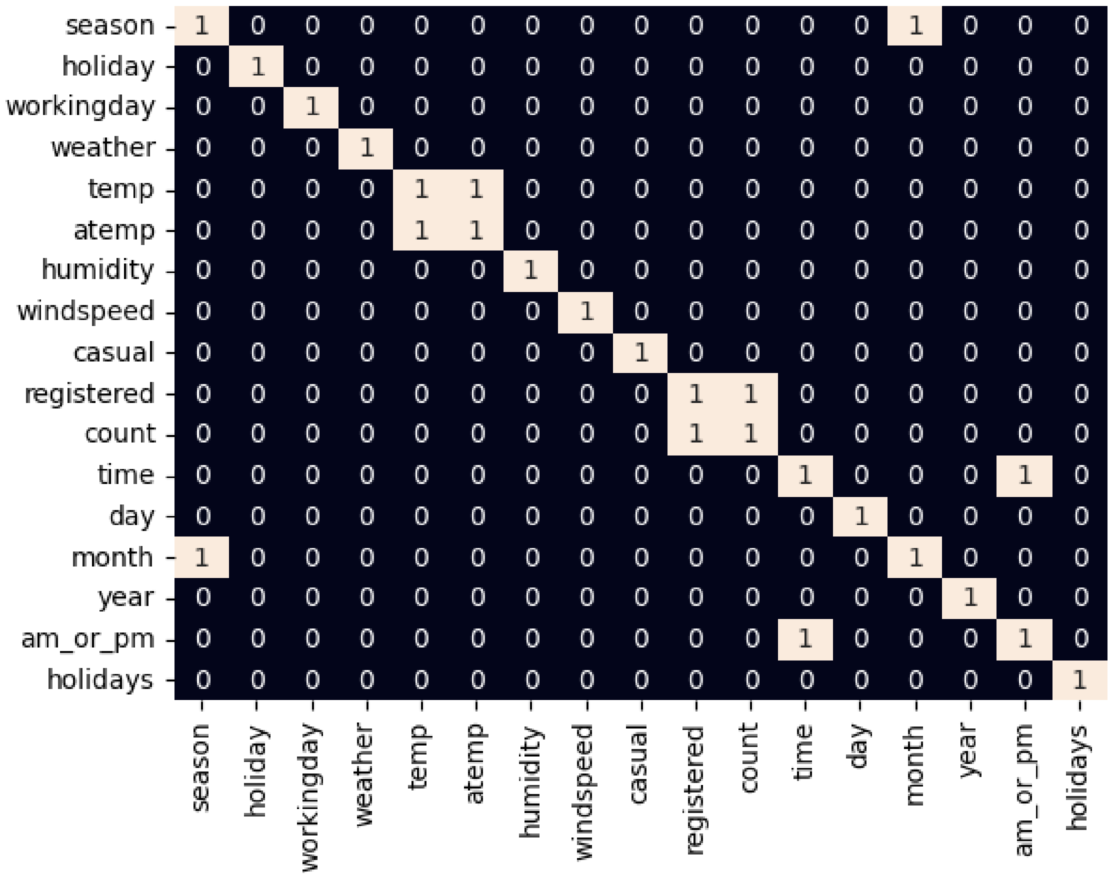



0
20
40
60
80
100
humidity

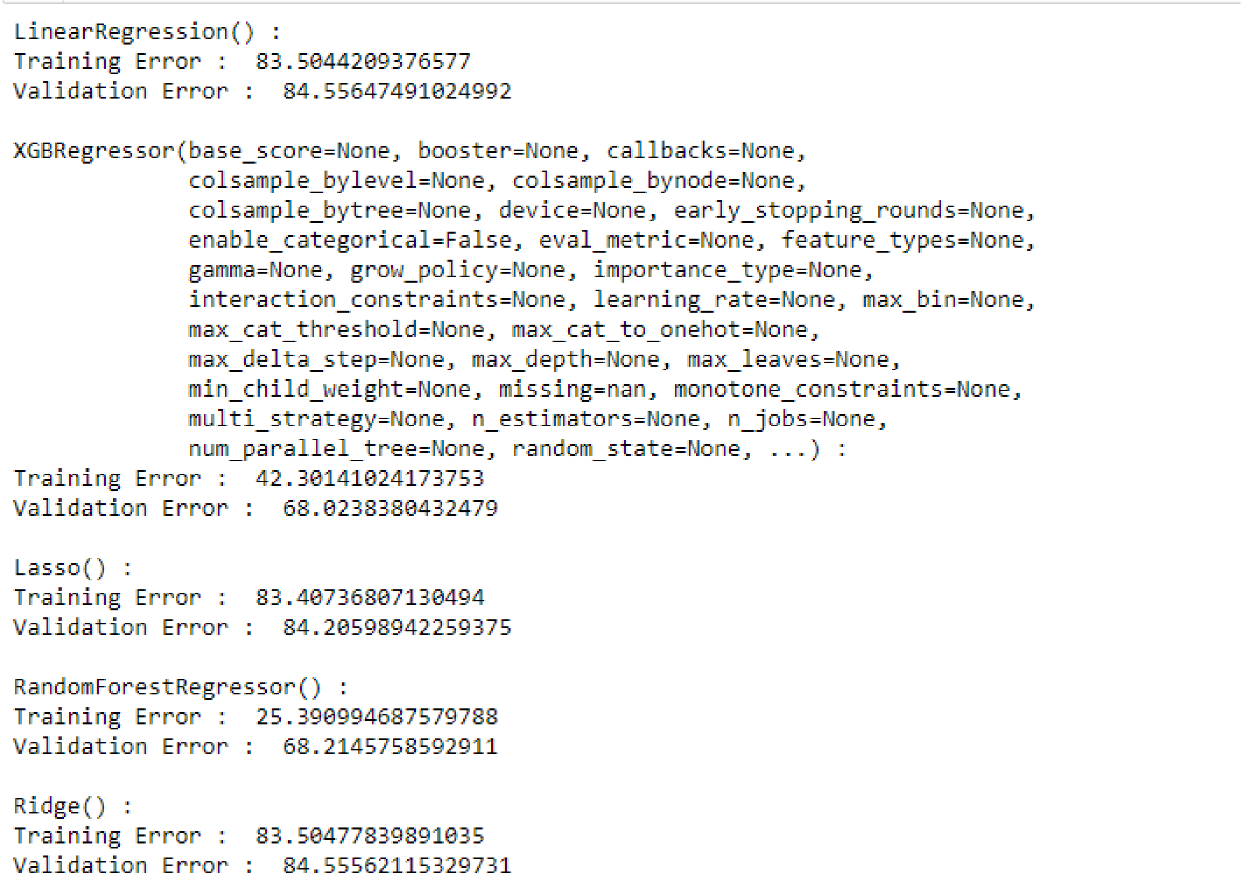

0
50
100
150
200
250
300
350
casual


0
200
400
600
800
registered


0
200
400
600
800
1000
count



	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count	time	day	month	year	am_or_pm	holidays
season	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
holiday	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
workingday	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
weather	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
temp	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
atemp	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
humidity	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
windspeed	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
casual	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
registered	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
count	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
time	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
day	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
month	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
year	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
am_or_pm	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
holidays	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1



```
LinearRegression() :
Training Error :  83.5044209376577
Validation Error :  84.55647491024992

XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...) :
Training Error :  42.30141024173753
Validation Error :  68.0238380432479

Lasso() :
Training Error :  83.40736807130494
Validation Error :  84.20598942259375

RandomForestRegressor() :
Training Error :  25.390994687579788
Validation Error :  68.2145758592911

Ridge() :
Training Error :  83.50477839891035
Validation Error :  84.55562115329731
```

CHAPTER 5

DECLARATION

We, Dhruv Vashishtha & Bhoogarv Maheshwary students of 7th semester BE, Artificial Intelligence and Machine Learning department, Bangalore Institute of Technology, Bengaluru hereby declare that AML project work entitled "OLA RIDE REQUEST FORECAST" has been carried out and submitted in partial fulfillment of the course requirement for the award of the degree of Bachelor of Engineering in Artificial Intelligence and Machine Learning of Visvesvaraya Technological University, Belagavi, during the academic year 2023-2024.

We also declare that, to the best of our knowledge and belief, the work reported here is not from the part of dissertation based on which a degree or award was conferred on an earlier occasion on this by any other student.

Dhruv Vashishtha
Bhoogarv Maheshwary

CHAPTER 6

CONCLUSION/FUTURE ENHANCEMENT

Conclusion:

The “OLA Ride Request Demand Forecast” project successfully tackled the challenge of optimizing ride demand forecasting for Ola Bikes. The developed model incorporates guidelines provided by Ola's business team, effectively filtering out invalid and fraudulent ride requests. By considering temporal and spatial aspects, the model provides accurate predictions for a specified region and future time window, enabling Ola to allocate drivers more intelligently. The implementation of this model is expected to significantly improve Ola's operational efficiency, reduce losses, and enhance its competitive position in the bike-sharing market.

Future Enhancements:

1. **Dynamic Geographical Constraints:** Implementing a dynamic system for geographical constraints could enhance the model's adaptability to changing demand patterns and emerging service areas.
2. **Integration of External Factors:** Incorporating external factors such as weather conditions, local events, or traffic data could further refine the accuracy of demand predictions and improve resource allocation.
3. **Real-time Data Updates:** Implementing a mechanism for real-time data updates would enable the model to continuously adapt to evolving user behaviors, ensuring ongoing accuracy in ride demand predictions.
4. **Machine Learning Explainability:** Enhancing model interpretability and explainability features would enable Ola to better understand the factors influencing predictions, facilitating more informed decision-making and strategic planning for future service expansions.

CHAPTER 7

REFERENCES

- https://www.kaggle.com/competitions/bike-sharing-demand/data
- https://www.irjet.net/archives/V10/i3/IRJET-V10I342.pdf
- https://www.academia.edu/104358866/Taxi_Demand_Prediction_using_Machine_Learning
- Google