A Project Report

On

Airline Price Prediction Using ML

(Submitted in partial fulfilment of requirements for the award of degree of)

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE &ENGINEERING

by

Deepti Bansal

(1750810023)

Under the Guidance of

Mr. Sumit Kumar Mishra

Assistance Professor

BABU BANARASI DAS ENGINEERING COLLEGE

Affiliated to Dr. APJ Abdul Kalam Technical University, Lucknow
College Code: 508

June,2021

CERTIFICATE

Certified that **Deepti Bansal** has carried out the Project work presented in this report entitled

"Airline Price Prediction Using ML"

for the B.Tech. (Computer Science & Engineering) Fourth Year (Eight Semester) from

Babu Banarasi Das Engineering College, Lucknow under my supervision. The report embodies result of original work and studies carried out by Student himself and the contents of the Project do not form the basis for the award of any other degree to the candidate or to anybody else.

Head Computer Science & Engineering

Project Guide Name
Mr. Sumit Kumar Mishra
Designation: Assistant Professor

Date: 31-07-2021

ABSTRACT

Air travel has become an important part of modern life as more and more people are choosing the fastest travel options. Prices for airline tickets go up or down from time to time depending on a variety of factors such as flight arrangements, destination, flight length. various occasions such as holiday or holiday season. Therefore, having a basic idea of the cost of air travel before planning a trip will help many people to save money and time. In this case the prediction model will be built using machine learning algorithms on the collected historical data of the aircraft. This program will give people an idea of the price-tracking trends and also provide a predictable price to which they can refer before booking their airline tickets to save money. This type of prediction can be provided to customers by airline booking companies that will help customers book their tickets properly.

Some of the currently used approaches for the prediction of price of flights are:

- Linear Regression
- Decision Tree
- K-Nearest Neighbors
- Random Forest Regressor

ACKNOWLEDGMENT

I take this occasion to thank God, the Almighty for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our esteemed guide, guide name, for providing us with the right guidance and advice at the crucial junctures and for showing us the right way.

I extend my sincere thanks to our respected Head of the Department Prof. (Dr.) Avinash Gupta, for allowing me to use the facilities available.

I am also thankful to the entire department who were helpful in providing their thorough insight which helped me in enhancing the various modules and features of this project.

I would also like to thank my family and friends for being a constant source of motivation throughout the project work.

Thankfully,
Deepti Bansal
(175080023)

TABLE OF CONTENT

CONTENTS	PAGE NO.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Machine Learning is used everywhere from self-fulfilling common responsibilities to providing intelligent understanding, industries in every region attempt to benefit from it. You can already share the use of a tool it uses. For example, a wearable health tracker like a little game, or a smart home assistant like google home. However, there are too many models used.

• **Predictability** - Machine learning can also be used within predictive systems. Considering the loan model, calculating the probability of error, the gadget will want to differentiate that it should be realistic in companies.

• **Photo Recognition** - Study gadget can be used to find the face in a photo properly. There is a separate category for everyone in the database of several people.

• **Speech Recognition** - Interpretation of cited phrases in the text. These are miles used for voice search and more. Voice user communication methods include voice dialing, word processing, and performance management. It can also be used to include simple information and instruction for organized files.

• **Clinical Diagnosis** - ML has the ability to hold cancerous tissue.

1.1 History of Machine Learning

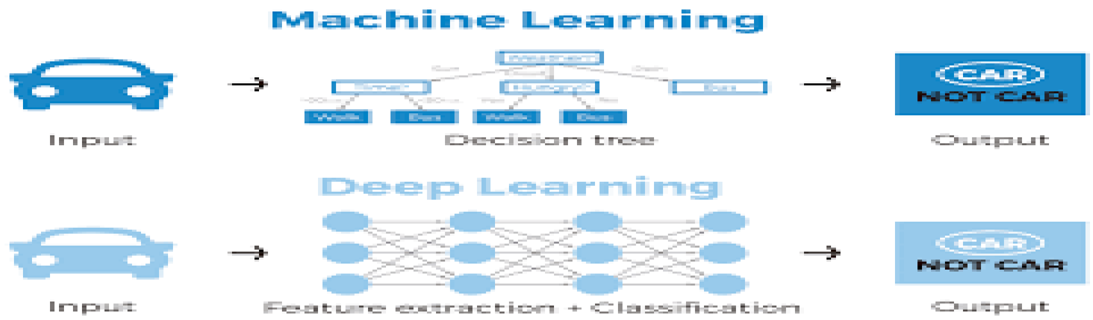


Fig 1.1 Machine Learning vs. Deep Learning

It was in the 1940s when the first manually operated computer system, ENIAC (Electronic Numerical Integrator and Computer), was invented.

Fig 1.2 EIMC — Electronic Numerical Integrator and Computer

In the 1950s, we see the first computer game program claiming to be able to beat the checkers world champion.

Thanks to statistics, machine learning became very famous in the 1990s. The intersection of computer science and statistics gave birth to probabilistic approaches in AI. This shifted the field further toward data-driven approaches. Having large-scale data available, scientists started to build intelligent systems that were able to analyze and learn from large amounts of data. As a highlight, IBM's Deep Blue system beat the world champion of chess, the grand-master Garry Kasparov.

1.2 What is Machine Learning?

According to Arthur Samuel, Machine Learning algorithms enable computers to learn from data, and even to improve themselves, without explicit programming. Machine learning (ML) is an algorithm component that allows software applications to be more accurate in predicting results without explicit programming.

Machine learning uses an algorithm and data to create a model. The algorithm is a code written in Python, R, or in the language of your choice, and describes how the computer will start learning from training data. In supervised reading these data and labels used to train the system to predict the label according to input. The predictability of a production model depends on how well the distribution of production data is similar to the distribution of training data. As these distributions flow separately, or overflow, the performance of the model deteriorates, and the predictions become less accurate. For example, if you use the training data for a tree identification program from the summer, when the leaves are full and green, the system will be more accurate as the color of the leaves changes or when the trees lose their leaves. To keep your system accurate, you will need to update your model as the seasons change to keep data sharing synced.

1.1 Categorization of Machine Learning:

Machine learning are broadly classified into three categories:

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

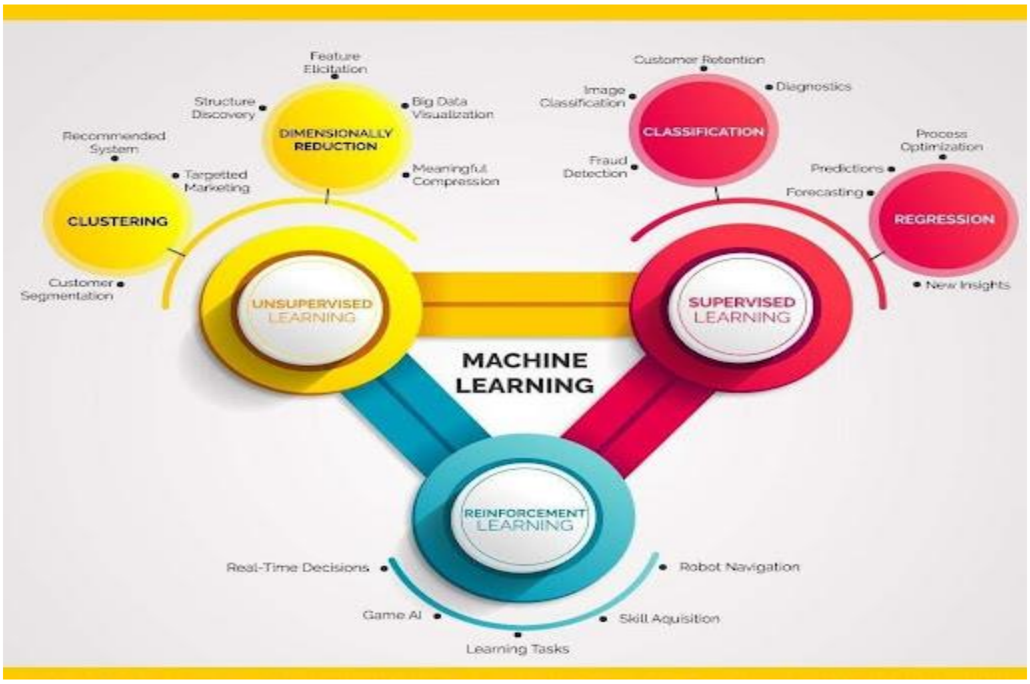


Fig 1.3 Machine Learning Categories

1.1.1 Supervised Learning Algorithm

Supervised learning is a form of machine learning in which machines are taught using well-written training data, and according to that data, machines predict the outcome. Labeled data means that some input data is already tagged with the correct output. In supervised learning, the training data provided by the machines serves as the equipment manager to predict the outcome accordingly. The same principle applies as a student learns from the teacher's guidance. Supervised learning is the process of providing input data and output data relevant to a machine learning model. The purpose of the supervised learning algorithm is to find a mapping function to map the input variable (x) with the output output (y).In the real world, supervised learning can be used for Risk Assessment, Image Sharing, Fraud Detection, Spam Filtering, etc.

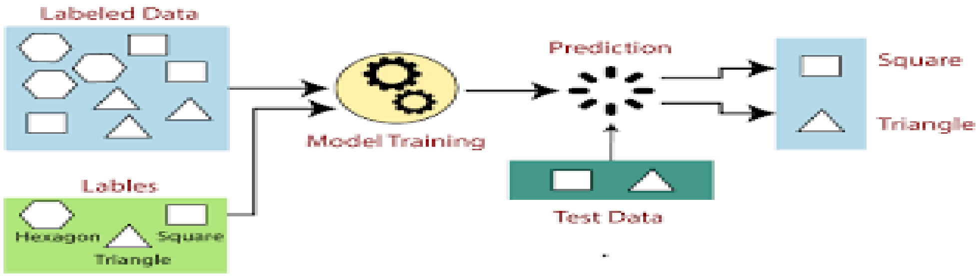


Fig 1.4 Process of Supervised Learning

1.3.3.1 Types of Supervised learning

- **Classification**: a classification hassle is whilst the output variable is a category, inclusive of "red" or "blue" or "ailment" and "no ailment".
- **Regression:** a regression hassle is when the output variable is a real cost, together with "dollars" or "weight".

1.1.2 Unsupervised Learning Algorithm

As the name suggests, unsupervised learning is a form of machine learning where models can be directed using a training database. Instead, the models themselves acquire hidden patterns and insights from the data provided. It can be compared to learning about the human brain while learning new things. It can be described as:

"**Unsupervised learning is a form of machine learning in which models are trained using a unlabeled database and are allowed to process that data without supervision.**"

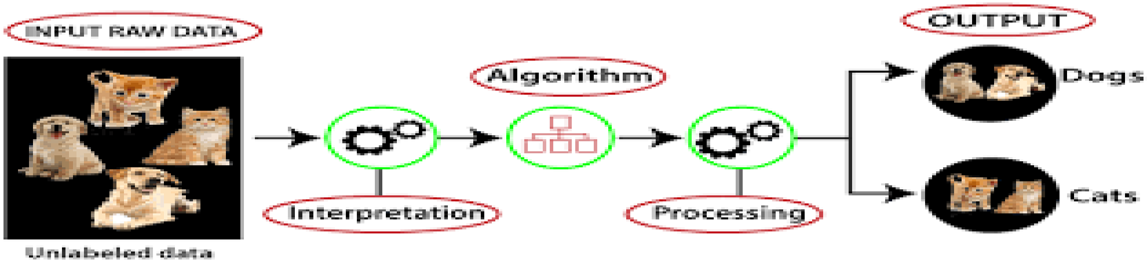


Fig 1.5 Process of Unsupervised Learning

Unattended readings cannot be applied directly to the back or split problem because unlike supervised readings, we have input data but no corresponding output data. The purpose of unsupervised learning is to find the basic structure of a database, group that data accordingly, and represent that database in a compressed format

1.1.2.1 Types of Unsupervised learning

- **Clustering**: A clustering hassle is in which you want to find out the inherent groupings within the statistics, such as grouping customers by using buying behavior.

- **Association**: an affiliation rule mastering hassle is where you need to discover rules that describe massive quantities of your records, such as people that purchase x additionally have a tendency to buy Y.

1.1.3 Reinforcement Learning

A reinforcement gaining knowledge of set of rules, or agent, learns with the aid of interacting with its surroundings. The agent receives rewards via acting efficiently and penalties for performing incorrectly. The agent learns without intervention from a human by way of maximizing its reward and minimizing its penalty. It's far a type of dynamic programming that trains algorithms the use of a machine of praise and punishment.

Fig 1.6 Process of Reinforcement Learning

It is basically leveraging the rewards obtained, the agent improves its environment information to select the subsequent motion.

It is basically leveraging the rewards obtained, the agent improves its environment knowledge to select the next action.

1.2 Models Used

1.2.1 Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. Do the back-up function. Regression models are targeted predictor values based on independent variables. It is widely used to find relationships between variables and predictions. The different types of regression vary depending on the type of relationship between dependent and independent variables, taking into account the number of independent variables used.

The linear regression enables the function to predict a different amount of dependence (y) depending on the given independent variation (x).

Therefore, this regression process finds an equal relationship between x (input) and y (output). Therefore, the name is Linear Regression.

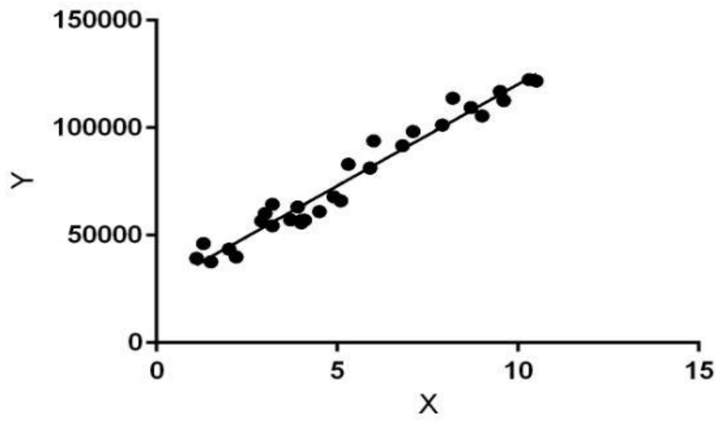


Fig 1.7 Linear Regression

In the above figure, X (input) is a work experience and Y (output) is a personal salary. The return line is the most appropriate line in our model.Hypothesis function for Linear Regression:

$$y = \theta_{1} + \theta_{2} * X$$

While training the model we are given:

x: input training data (univariate – one input variable(parameter))

y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.

θ1: intercept

θ2: coefficient of x

Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

Cost Function (J):

By achieving a well-proportioned regression line, the model aims to predict the degree to which the error difference between the predicted value and the actual value be minimal. Therefore, it is very important to update the values of θ1 and θ2, to achieve the leading value that reduces the error between the predicted value y (pred) and the real value (y).

$$minimize \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

Cost function(J) of Linear Regression is the Root Mean Squared Error (RMSE) between predicted y value (pred) and true y value (y).

1.2.2 Random forest

Random Forest is a combination of learning algorithm, which can be used to reverse each type of task. In particular, it is a long way from bagging. We explained that the fundraising process involves the integration of many vulnerable novices. In the wild jungle, these endangered visitors are beautiful woods. Therefore, before going into the details of the random forest, we can try to understand the basics of the selected trees. The selected tree is a set of guided rules, which can be used to go back and forth. However, it is commonly used in category matters. It is the miles that make up the number of internal nodes where each node represents a check in value (e.g., whether the next day's weather is sunny or full or wet). Each branch inside the tree represents the test result and the leaf nodes represent the final result (beauty label). It involves violating training set to multiple repositories.

The development of the decision tree includes the division of all educational information into subsets, 16 of which are made in each internal area according to a specific requirement. A set of determining rules determines the optimal classification of all nodes in a metrics perspective that includes pollution and information gain. Pollution acquisition is a method always an option that is randomly selected from a set that can be incorrectly written randomly in accordance with the distribution of labels within the subset. Record acquisition is used to determine which work should be cut at every step in the construction of the tree. The separation process lasts until the internal node has a category label fee.

Apart from the fact that cutting trees are smooth to see and do well in a few data sets, they tend to have high variability due to the algorithm capture method where the tree tends to always choose high quality partitions at all levels and cannot see far behind the modern stage. For this reason, there may be an opportunity for over-qualifying, where the version only works more efficiently within the training set and fails to perform well in test sets. In simple language, a random forest forms a few trees of choice and combines them to improve the performance of the whole version.

As we have seen before, bootstrapping is the process of sampling school records from time to time otherwise. The random forest uses bootstrapping as per the tree of choice can be taught with unfamiliar passages of information. In addition, the random forest uses random subsets of elements.

For example, if there are 50 elements within the records, the random forest will specifically select their different species, allowing 10, to train on each tree. Therefore, each tree can have 10 random elements that will be used for training including 17 to get a good cut for all tree nodes. Once we have collected a collection of decision trees, the results of each tree can be combined to get the final result (final vote). A competent form in one of these methods will ensure the creation of happiness because there is no longer one, but more than one decision tree is used in selection, and in addition, each tree has the capabilities of different data categories.

1.2.3 Random Forest Regressor

All decision trees have high variability, but when we combine them all the same then the resulting difference is low as each decision tree is well trained in that sample data so the result does not depend on one decision tree but determines many trees. In the case of a segregation problem, the final result is determined by using a majority vote. In the case of a relapse problem, the end result is the goal of all outcomes. This part is Consolidation.

Unplanned Forest is a combination method that is able to perform retrofitting and separation tasks using multiple decision-making trees and a process called Bootstrap and Aggregation, more commonly known as bagging. The basic premise of this is to combine multiple decision trees in determining the final outcome rather than relying on individual decision trees.

The Random Forest has many trees for decision-making as basic learning models. We randomly process the sample and then extract the sample from the database that creates the data samples for all models. This section is called Bootstrap.

We need to look at the process of deforestation in a random forest like any other machine learning process:

Create a specific query or data and find a source to determine the required data.

- Make sure the data is in an accessible format and convert it to the required format.
- Specify all visible faults and missing data points that may be required to complete the required data.
- Create a machine learning model
- Set the basic model you want to achieve
- Train data machine learning model.
- Provide model understanding with test details
- Now compare performance metrics for both test data and predicted data from the model.

- If expectations do not meet your expectations, you can try to improve your model appropriately or fall in love with your data or use another data modeling process.
- In this section you translate the data you receive and report it accordingly.

1.2.4 Gradient Boosting Regressor

Gradient Promotion is a popular promotion algorithm. In a gradient extension, each predictor corrects its previous error. In contrast to Adaboost, training weight weights are not used, instead, each predictor is trained using the following remaining errors such as labels.

There is a process called Gradient Boosted Trees with its basic student CART (Classification and Regression Trees).

The diagram below illustrates how advanced gradient trees are trained for retreat problems.

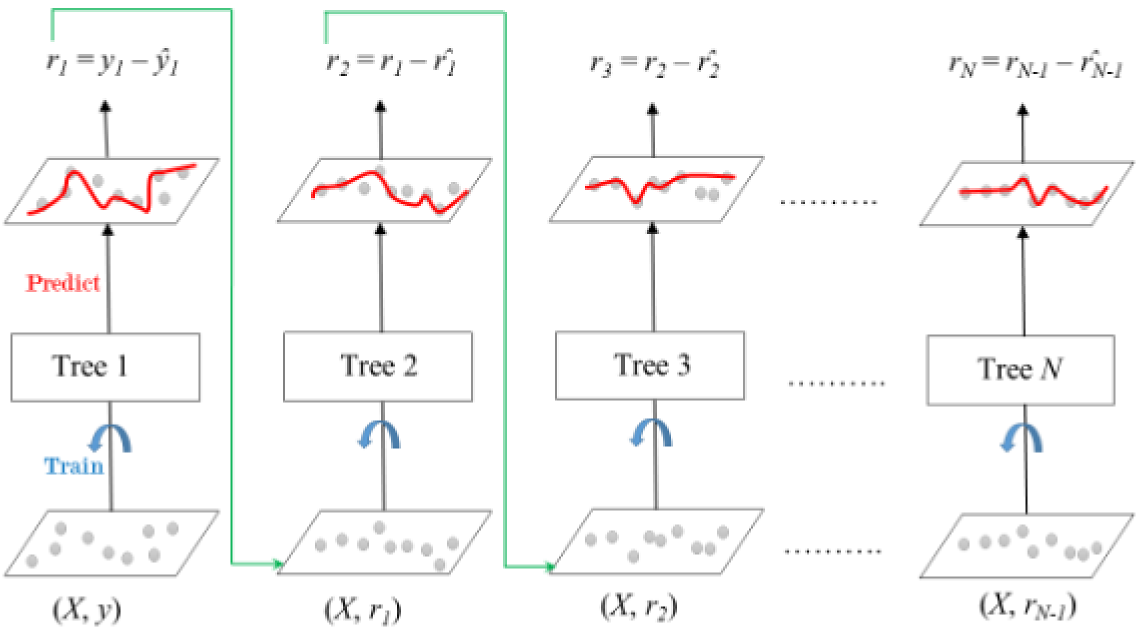


Fig 1.8 Gradient Boosting Regressor

The collection contains N trees. Tree1 is trained using the element matrix X and labels y. The predictions with the label y1 (hat) are used to determine if errors are a remnant of training r1 training. Tree2 is then trained using the matrix X feature and the remaining r1 of Tree1 r1 as labels. Results predicted r1 (hat) and used to find residue r2. This process is repeated until all N trees are trained including.

There is an important parameter used in this process known as Shrinkage.

The decrease refers to the fact that the prediction of each tree in the ensemble is reduced after repeated with a reading level (eta) of between 0 and 1. Instead of transactions between eta and number of speculators, the declining learning level requires compensation with increasing proportions in order to achieve certain model performance. Now that all the trees have been trained now, predictions can be made.

Each tree predicts the label and the final prediction is given by a formula,

Y(pred) = y1+(eta * r1) + (eta * r2) +-----------------+ (eta * rN)

The category of gradient boosting regression in scikit-learn is the Gradient Boosting Regressor. The same algorithm is used for partitions known as Gradient Boosting Classifier.

1.2.5 SVM

Support Vector Machine is a discriminatory algorithm that attempts to find a suitable hyperplane that accurately separates data points in the N-dimensional space (N - number of features). In a two-dimensional space, a hyperplane line separates data points into two separate categories. In a larger space, the hyperplane would have a different shape than the line.

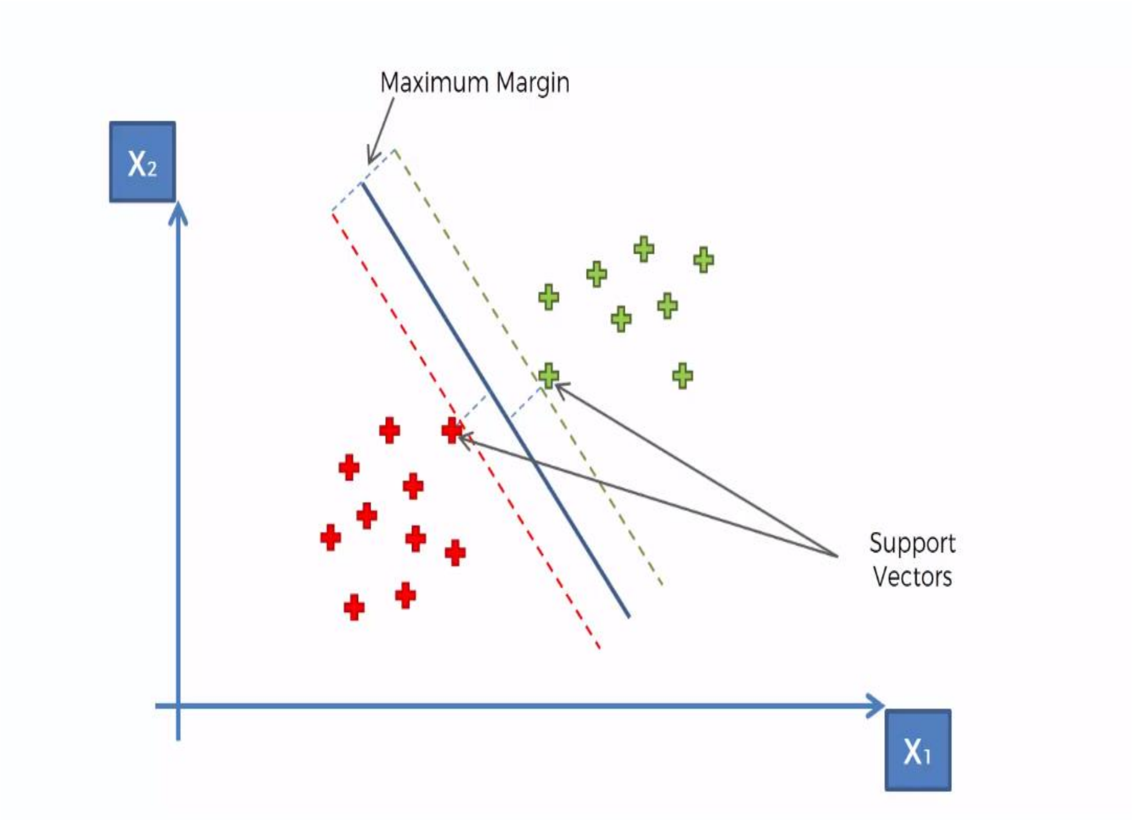


Fig 1.9 SVM

In search of a suitable hyperplane, SVM attempts to obtain parameters of the boundary data or support vectors. The support vectors are selected in such a way that the hyperplane is at the highest point from both support carriers.

The supporting vectors of those two data points support the decision limit (data points have higher limits from hyperplane). SVM always tries on those two data points from different classes that are very close to each other. These support vectors are the key to drawing a suitable hyperplane with SVM. In SVM, the input and output data set is treated as vectors. This is because when data is a high-volume space (more than twice the size), classes cannot be represented as single data points, so they should be represented as vectors. And that's why it's called "Support Vector Machine".

CHAPTER 2

PROBLEM DEFINITION

These days, airline ticket prices can vary considerably and significantly on the same plane, even in nearby seats within the same cabin. Customers want to get the lowest price while the airlines try to keep their total cost as high as possible and make their profits more. Airlines use a variety of computer techniques to increase revenue such as demand estimates and price discrimination.

On the customer side, two types of models are proposed by various investigators to save money for customers: models predicting the right time to buy a ticket and models predicting the low number of tickets. We might have often heard travelers saying that flight ticket prices are so unpredictable.

As a student of computer science & engineering, we have the knowledge of machine learning So, we are going to prove that if given the right data anything can be predicted.

CHAPTER 3

OBJECTIVE

We attempt is to gather different datasets investigated by researchers, categorize them into real and synthetized groups and extract the common attributes affects the price of flight.

- The main purpose of the project is to predict the cost of a flight using different algorithms for the study of different machines.
- Anyone who regularly buys a plane ticket, will be able to predict the exact amount to buy a ticket
- Doing well saves time / money.

CHAPTER 4

LITERATURE REVIEW

Reference	Addressed Problem	Dataset	Features	Computational Techniques Used	Performance Result	Remark
• Y. Chen et al., (2015)	Minimum Ticket Price Prediction	More than 3 months (110 days) data for 5 international routes.	Prices of the same itinerary, prices of recent itineraries before the target day, prices of itineraries with the same day of week, price of itineraries with the same day of month	An ensemble-based learning algorithm Learn++.NSE is modified and used	Mean absolute percentage error (MAPE) of 10.7% as compared to KNN (12.58) and PA (15.41%).	Not possible to predict price for a flight Does not consider multi-stop flights
• Anastasia Lantseva et al., (2015)	Ticket Price per kilometer Prediction	Ticket price data collected for 75 days and 90 days for local and international flights.	City of departure, destination, ticket purchase date, departure date, ticket options with the price	Regression Model	Not given	No performance evaluation presented. The dataset set is limited

• (K. Lazaridis et al., (2017)	Comparing regression machine learning models for predicting airline ticket prices.	A dataset consisting of 1814 flights for a single international route	Departure time, arrival time, number of free luggage, days before departure, number of intermediate stops, holiday, time of day and day of week	Eight regression machine learning models used.	Bagging Regression: 87.42%, accuracy and Random Forest Regression Tree: 85.91%. accuracy	NA
• T. Liu et al., (2017)	Predicting the lowest price available before departure date	Data consisting of 19 different routes and spans three months period (92 days).	Historical ticket prices, a signal indicating whether the departure date is holiday or not and number of days before departure	Ensemble model that uses techniques such as K-Nearest Neighbours, Random Forest and Bayesian	Improved the MAPE from (7% −12%) to (3.7% − 6%).as compared to the single model	NA

Fig 4.1 Literature Review Table

CHAPTER 5

MACHINE LEARNING METHODOLOGY

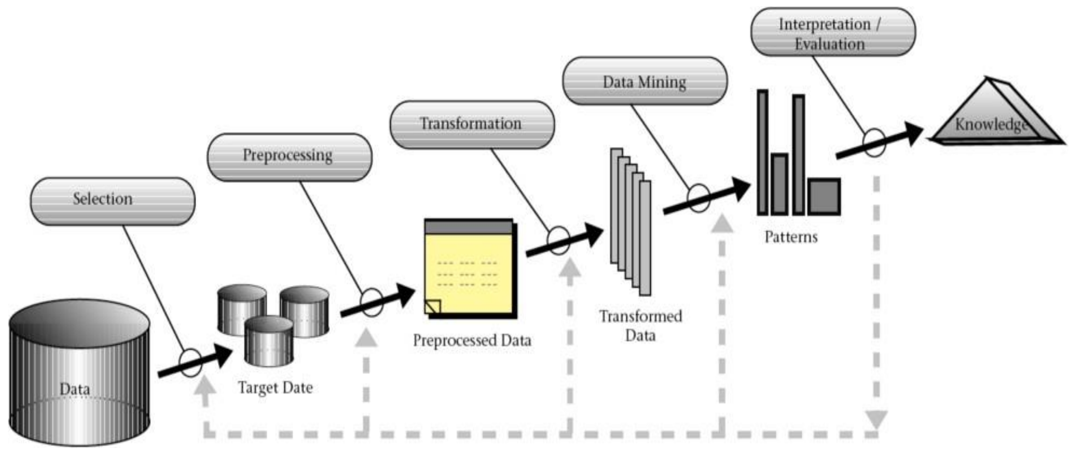


Fig 5.1 Proposed Methodology Structure

5.1 INPUT DATA

For the prediction of price of airline, the more data we have gives us the better results. For supervised machine learning, the data must be labelled.

5.2 DATA CLEANING

Data cleaning is the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated, or improperly formatted. This data is usually not necessary or helpful when it comes to analyzing data because it may hinder the process or provide inaccurate results

5.3 .DATA PRE-PROCESSING

Data Preprocessing is that step in which the data gets transformed, or encoded, to bring it to such a state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm.

5.4 TRAIN ALGORITHM

The algorithm is a set of rules to follow when solving complex problems, such as a mathematical equation or recipe. The algorithm uses the customer data defined by our features to learn how to make predictions. Initially, we will train an algorithm in historical data, this we call a training set. The extra trick in this training sets better, so that the machine has more examples to learn from.

5.5 CREATING A MODEL

When the training is completed, you have a model that is specific to your business, which can detect fraud in milliseconds.

We are always looking for a model to make sure it behaves properly, and we are always looking for ways to improve it. We are constantly updating, updating and uploading a new model for all clients so that the system is always up to date with the latest fraud measures.

CHAPTER 6

AGILE METHODOLOGY IN MACHINE LEARNING

As a framework and approach, agile is one of the most popular formats for major growing building software models/applications. Through the interaction and engagement woven into the development process, agile is guaranteed to provide greater efficiency across all parameters. This will increase for development teams who are focused on ongoing solutions to complex challenges. The pilot project is a growing number of agile approaches. From voice assistants to real-time forecasts, agile is used to continuously improve the feedback system. More and more activities tend to be more aware of day-to-day effort and character development, which creates a basic need for agile as a comprehensive team process.

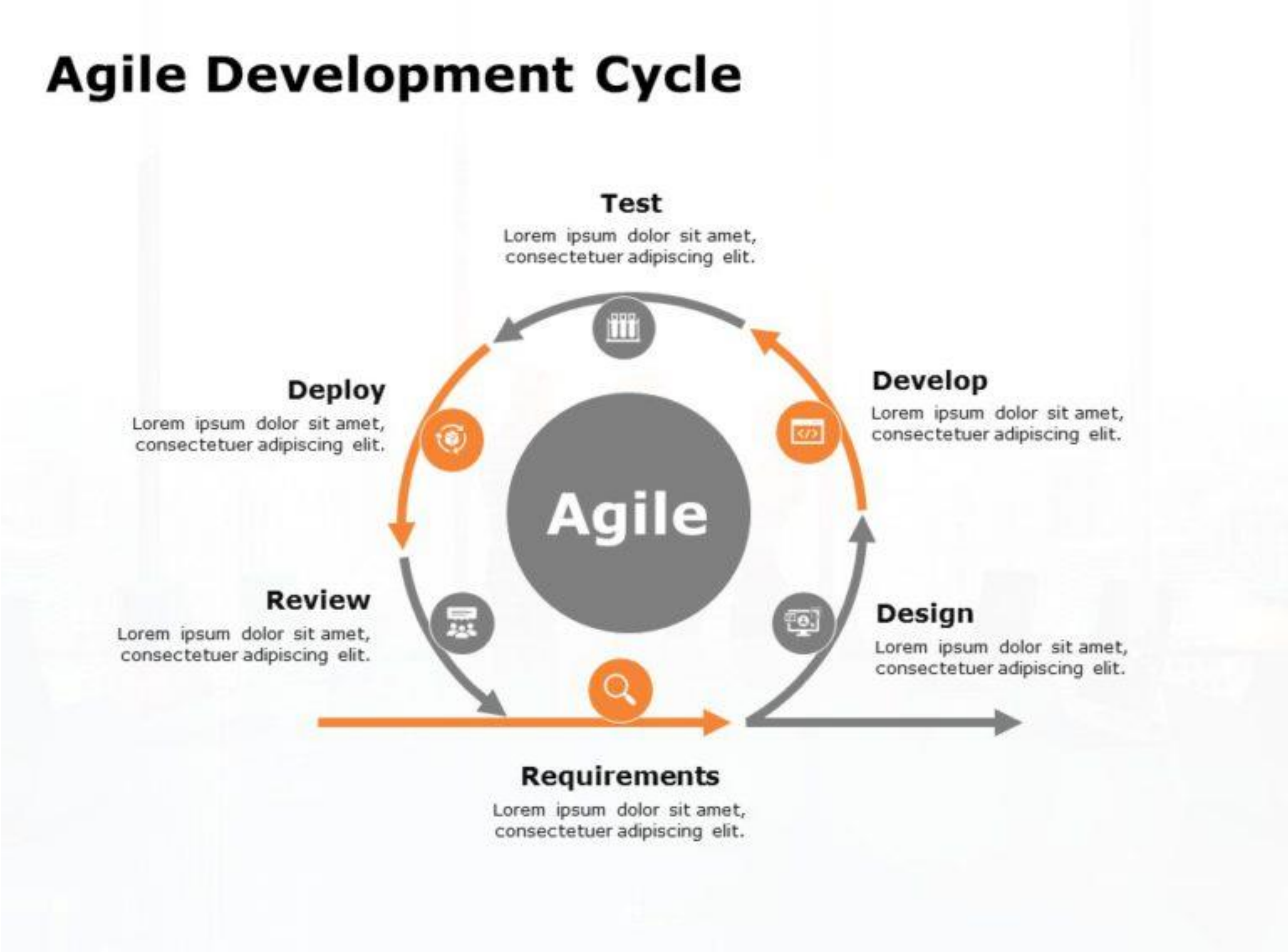


Fig 6.1 Agile development Cycle

6.1 Project Management

Agile usually involves multiple layers of participant input, with rotating experimentation and fast prototyping. This requires the process of fingerprinting on behalf of the project manager, and teams should be a key element of consistent communication. Agile resources for improving conversations within the challenge, creating more bonds between group providers. This leads to a larger green management structure, which allows for understanding to be accompanied by a free flow.

Ideas, skills, and comments can be added to the loop at any time, making the path more dynamic and focused. This ends up in the notion that markets are becoming increasingly digital and keeping up with the latest trends is very important. Agile allows machine learning projects to be market-targeted and achieve challenging objectives in a timely manner.

Jobs were also undertaken by an increasing number of transparent exhibitors, with all levels of development gaining universal access. This results in a holistic approach to development, in which all employees have access to key statistics. Openness also allows managers to be more efficient, which allows them to keep track of institutional improvements.

6.2 Decision-Making in Design

A global survey of more than 1,300 decision-makers found that speed plays an important role in speeding up decision-making. This approach has been shown to improve interpersonal communication, statistical understanding, and record processing. The data confirmed that agile-following groups saw a 60% increase in sales technology. General benefits are growing, focusing on greater acceptance in all industries. Significant change is driven by the help of extended decision-making across all technologies. In terms of the knowledge system, agile allows teams to meet sponsorship demands at a faster rate. It creates better generation solutions that can have measured results. While agile is actually part of the engineering, design and testing of domains, it can revitalize the company significantly.

Businesses are looking for ways to improve their product portfolios and make a decent project using a computer. Accelerating the decision-making process is just one of many ways in which agile will control machine understanding in the years to come.

6.3 Optimize Core Resources

Agile allows agencies to upgrade their assets in the form of skills and technology. The groups were also assigned in accordance with the desired final results, with repeated improvements being in the middle of the assignment. Teams can then join to find a different solution for that particular task. This increases the effort and time of each resource, providing significant competitive advantage to companies.

While engineering is directly linked to R&D, there is interaction across the board. Agile ensures that every effort given to the job learning gadget is made from scratch. There is no ambiguity within the process and every engineer knows his position in the system.

Whether that’s a facelift or a Chatbot development, agile creates a powerful environment for all resources to participate. This lets the device know its operating time, which leads to better performance by increasing the allocation of useful resources. Teams can also be transferred to new jobs or projects after their work has been completed.

The business cycle is maintained in the right place, where all stakeholders have the right resources. Performance is very important, which makes service delivery one of the key areas where agile will be central to all ml activities.

6.4 Rapid Validation of Data Models

Agile is particularly powerful in the rapid validation of hypotheses, specifically in the healthcare domain. It allows developers to test one-of-a-kind fashions and information scientists to have greater accurate data. When handling massive statistics sets, it's high-quality to have a method that provides flexibility and scale. That's where agile comes in.

Agile permits information groups to validate their models at a miles quicker rate. Teams can then iterate on various models and statistics sets to deliver higher results. They also can layout new fashions based on clearer facts factors, and rapid-check them until fulfillment. Agile allows teams to have a unique awareness, whilst continuously imparting key insights which could decorate the overall undertaking.

This blends into the artificial intelligence domain, with improved validation. Ai fashions can also be evolved with the help of machine studying and agile method, to create greater commercial enterprise effect. All areas inside excessive-tech domain names, consisting of finance, healthcare, and production, can leverage agile to create real-time speedy validation.

6.5 Increment in Adoption of Machine Learning

With the help of agile, big companies may want to know about working with a program to get to know teams and areas for improvement. This increases the reception of the gadget for information as a generation, in terms of providing a more efficient process to improve performance. Organizations can set up a gadget to get information while working within the agile paradigm.

Compared to conventional models, understanding a gadget can also grow extremely difficult thus enhancing the understanding burden. Businesses can also rely heavily on the use of the knowledge resource because they may now be less comfortable working with the season. However, within agile, organizations can reap the benefits of learning about the device without feeling crushed or challenged.

Agile takes data and records from all domains to create additional transparent solutions. This will increase the readability of device readings, giving a higher priority to greater demand across all industries.

CHAPTER 7
DATA FLOW DIAGRAM (DFD)

DFD is an acronym for Data Flow. The flow of system or process data is specified by DFD. It also provides insight into each enterprise and outbound business. DFD has no control flow and no rules or decision rules exist. Specific tasks depending on the type of data can be defined by flowchart. Data Flow diagram can be displayed in several ways. DFD is a systematic analysis tool. Data Flow diagrams are very popular because they help us visualize the major steps and data involved in software programs.

As with all large drawings and charts, DFD often can "tell" stories that can be difficult to explain in words, and draw a technical and non-technical audience, from an engineer to a CEO. That is why DFDs remain very popular despite everything over the years. At the same time as they work well on Statistics flow software and structures, they are less effective these days in looking at interactive, real-time applications or data-driven programs.

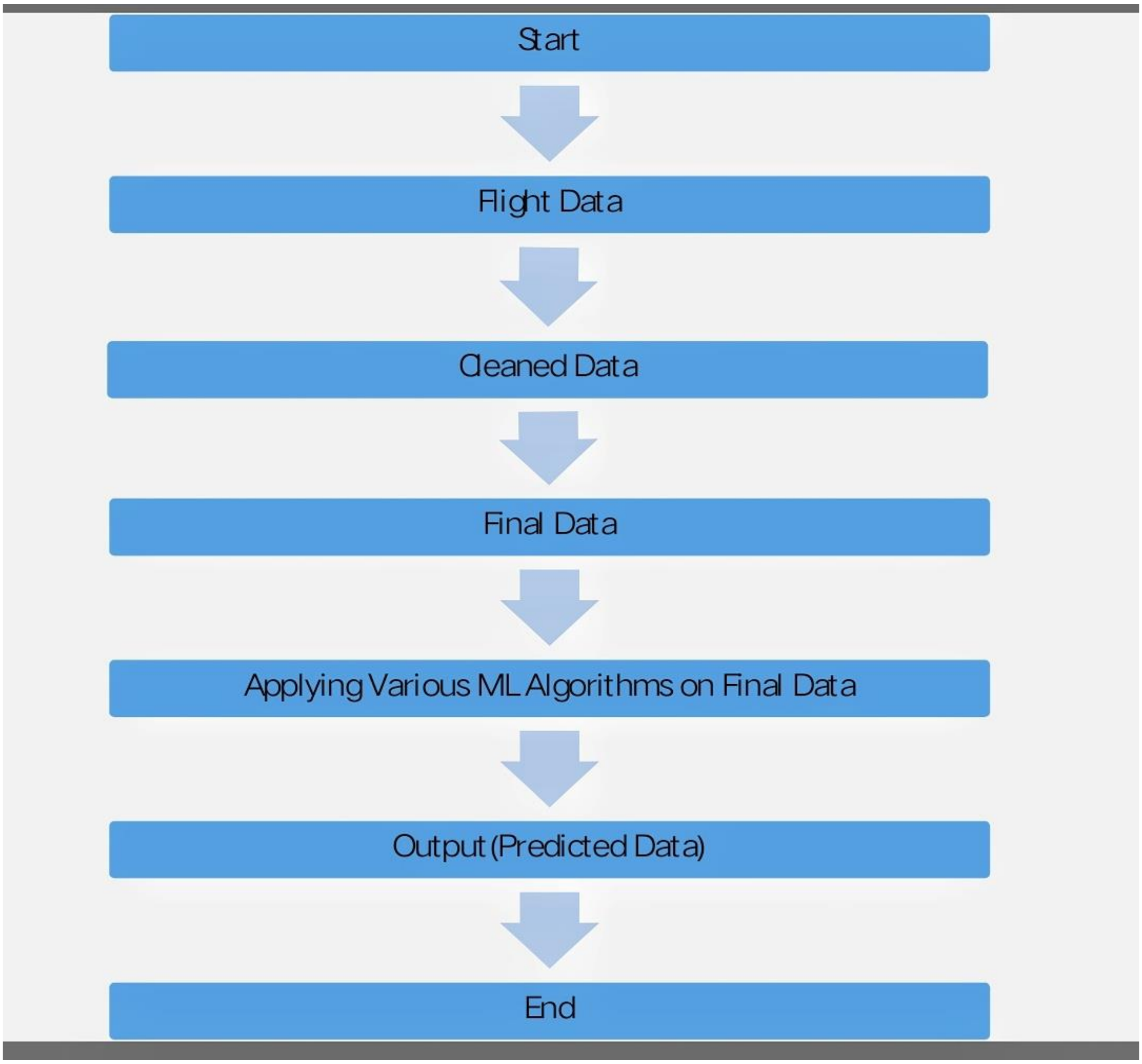


Fig 7.1 Data Flow Diagram

7.1 History of the DFD

The DFD text draws on a graph concept, which was originally used in performance research to move work flow across organizations. DFD came up with a diagram of the work used in formal planning and roadmaking in the late 1970s. DFD celebrities include Edward Yourdon, Larry Constantine, Tom DeMarco, Chris Gane and Trish Sarson.

Data flow diagrams (DFD) quickly became a popular way to visualize the major steps and data involved in software programs. DFDs were often used to demonstrate the flow of data on a computer system, although it could conceptually be used to model business processes. DFDs were useful for recording large data flows or exploring new high-level formats based on data flow

7.2 Components of DFD

Using any convention's DFD regulations or hints, the symbols depict the 4 components of records go with the flow diagrams:

The process

Output conversion in the system occurs due to process function. The process signs are rectangular with round, oval, rectangular or circular corners. This process is called a short sentence, with one word or phrase to express its context

Data Flow

Data mobility describes data that is transmitted between different parts of systems. An arrow icon is a data flow symbol. The associated name must be given to the flow to retrieve deleted information. The flow of data also represents objects and information that is moved. Property change is followed by programs that are not limited to education. The flow provided should convey only one type of information. Flow direction must be an arrow that can also be on both sides.

Storage space

The information is stored in a repository for later use. The two horizontal lines represent the store sign. Storage is not limited to a data file instead it can be anything like a folder with a document, an optical disc, a locker. The data repository can be viewed as independent of its use. When data flows from a repository is stored as data readings and when data flows to the store it is called data entry or data renewal.

Shortcut

Terminator is a foreign business that stands out of the system and communicates with the system.

It could be, for example, organizations such as banks, groups of people as clients or various departments of the same organization, which are not part of the modeling system and are outsourced. The corresponding systems also communicate with the terminator.

7.3 Rules for creating DFD

The name of the entity should be easy and understandable without any extra assistance (like comments).

- The processes should be numbered or put in ordered list to be referred easily.
- The DFD should maintain consistency across all the DFD levels.
- A single DFD can have maximum processes up to 9 and minimum 3 processes.

7.4 Levels of DFD

DFD uses hierarchy to maintain transparency thus multilevel DFD's can be created. Levels of DFD are as follows:

0-level DFD

1-level DFD:

2-level DFD:

7.5 Notations

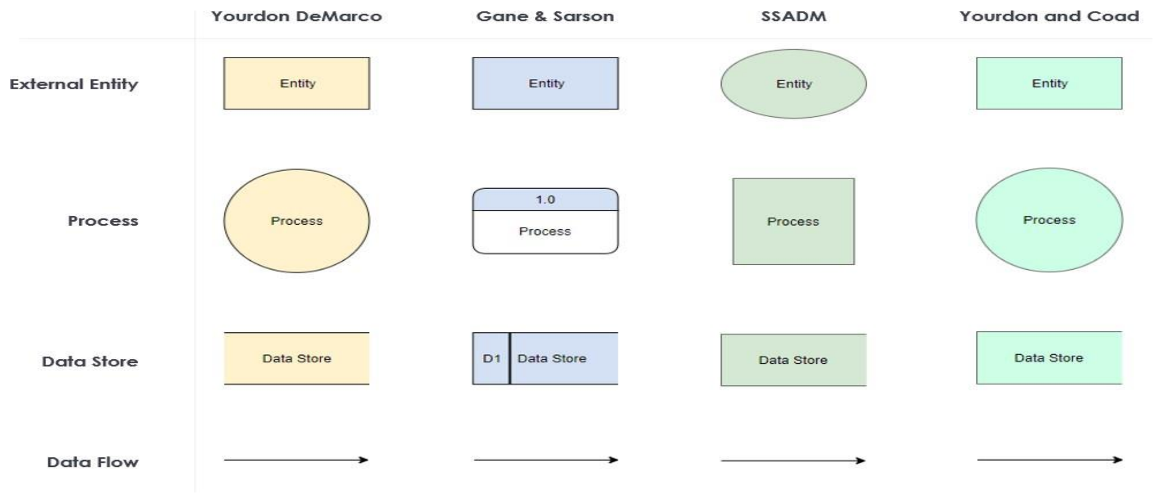


Fig 7.2 DFD Notations

7.6 Advantages of DFD

- It helps us understand the functionality and limitations of the system.
- It is an easy-to-understand presentation as it helps visualize content.
- The Data Flow diagram represents a detailed and well-defined diagram of system objects.
- It is used as part of a program document file.
- Data Flow Drawings can be understood by both professional or non-technical people because they are easy to understand

7.7 Disadvantages Of DFD

- DFD can sometimes confuse organizers with regard to the program.
- Data Flow diagram takes a long time to process, and often for this reason analysts are denied permission to work on it

CHAPTER 8
SOFTWARE/HARDWARE REQUIREMENTS

HARDWARE REQUIREMENTS

- RAM- 4 GB Minimum, 8 GB recommended
- Memory -512 GB or above
- Processor 64 bit
- Disk Space -2GB of available minimum disk spaces.

SOFTWARE REQUIREMENTS

- Editor can be Jupyter Notebook or Pycharm or Spyder (Used Jupyter)
- Python Programming Language
- Python compiler
- Matplotlib
- Sklearn
- Numpy
- Pandas

CHAPTER 9

ADVANTAGES AND DISADVANTAGES

ADVANTAGES OF MACHINE LEARNING

1. Handling mundane tasks

One sizable advantage of gadget mastering is its capacity to carry out mundane responsibilities with the assist of elaborate automation on the way to enhance productiveness. Theoretically, this could even take away "boring" responsibilities from humans and unfastened them up to be extra innovative.

2. Faster choices

The usage of system getting to know except cognitive technologies can aid in making quicker choices and take moves faster. You could analyze system getting to know by using gadget studying on line training magnificence.

3. Avoiding errors

The expression "human errors" became born because humans, surely, make errors every now and then. Computer systems, though, do no longer make these mistakes – this is, of course, thinking about they're programmed efficaciously. With gadget gaining knowledge of, information could be handled error-loose, regardless of how big the dataset might be.

4. Taking risks on behalf of people

With system studying you may arguably reduce the uncertainties you divulge humans to in the call of experimentation. Take, as an example, area exploration and the mars rover, known as interest. It may travel throughout the landscape of mars, inspecting it and determining the satisfactory routes to take, even as getting to assume for itself. Using synthetic intelligence in this manner ought to result in massive benefits in areas consisting of call for forecasting, scientific diagnosis, and oil exploration.

DISADVANTAGES OF MACHINE LEARNING

1. Job losses

There's no doubt that artificial intelligence & system studying will displace many low-skilled jobs. Arguably, robots have already taken many jobs at the meeting line – however now this could enlarge to new degrees.

Take, for instance, the concept of driverless cars, that could displace the want to have millions of human drivers, from taxi drivers to chauffeurs, right away. Of path a few might argue that synthetic intelligence will create more wealth than it destroys – but there is real risk that this may not be allotted evenly, in particular during its early expansion.

2. Distribution of energy

Artificial intelligence includes the hazard, inside the minds of a few, of taking manage away from human beings – de-humanizing movements in many approaches. Nations that are in possession of synthetic intelligence ought to theoretically kill human beings without needing to tug a trigger.

3. Lack of judgement calls

Humans can take precise situations and judgment calls into account once they make their selections, something that device learning can also by no means be able to do. One instance came about in Sydney, Australia, in 2014 when a shooting drama within the downtown area caused humans to make severe calls to an effort to break out the area. The result turned into that Uber’s ride fees surged primarily based on its deliver and call for algorithm – there was no consideration involved for the situations wherein the riders determined themselves

CHAPTER 10

CODING RESULTS

```
[2]:
# ....................................................Cleaning The Dataset.............................................

import pandas as pd
import numpy as np
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', None)
```

```
[3]: df = pd.read_excel("Data_Train.xlsx")
```

```
[4]: df.shape
```

```
[4]: (10683, 11)
```

```
[5]: df.head()
```

[5]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302

```
[6]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          10683 non-null  object
 1   Date_of_Journey  10683 non-null  object
 2   Source           10683 non-null  object
 3   Destination      10683 non-null  object
 4   Route            10682 non-null  object
 5   Dep_Time         10683 non-null  object
 6   Arrival_Time     10683 non-null  object
 7   Duration         10683 non-null  object
 8   Total_Stops      10682 non-null  object
 9   Additional_Info  10683 non-null  object
 10  Price            10683 non-null  int64
dtypes: int64(1), object(10)
memory usage: 918.2+ KB
```

```
[7]: df.describe()
```

[7]:

	Price
count	10683.000000
mean	9087.064121
std	4611.359167
min	1759.000000
25%	5277.000000
50%	8372.000000
75%	12373.000000

max	79512.000000

[8]:
```
df.isnull().sum()
```

[8]:
```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              1
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        1
Additional_Info    0
Price              0
dtype: int64
```

[9]:
```
df.shape
```

[9]:
```
(10683, 11)
```

[10]:
```
df1 = df.dropna()
df1.shape
```

[10]:
```
(10682, 11)
```

[11]:
```
df1['Day'],df1['Month'],df1['Year'] = df1['Date_of_Journey'].str.split('/',3).str
```

[16]:
```
df1.head()
```

[16]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Day	Month	Year	Dep_Hour	Dep_Minute
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897	24	03	2019	22	20
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662	1	05	2019	05	50
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882	9	06	2019	09	25
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218	12	05	2019	18	05
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302	01	03	2019	16	50

[17]:
```
df1['Dep_Hour'],df1['Dep_Minute'] = df1['Dep_Time'].str.split(':',2).str
```

[18]:
```
df1.head
```

[18]:
```
<bound method NDFrame.head of           Airline Date_of_Journey    Source Destination  \
0          IndiGo      24/03/2019  Banglore   New Delhi
1       Air India       1/05/2019   Kolkata    Banglore
2     Jet Airways       9/06/2019     Delhi      Cochin
3          IndiGo      12/05/2019   Kolkata    Banglore
4          IndiGo      01/03/2019  Banglore   New Delhi
...           ...             ...       ...         ...
10678    Air Asia       9/04/2019   Kolkata    Banglore
10679   Air India      27/04/2019   Kolkata    Banglore
10680 Jet Airways      27/04/2019  Banglore       Delhi
10681     Vistara      01/03/2019  Banglore   New Delhi
10682   Air India       9/05/2019     Delhi      Cochin
```

```
[19]: df1['Arrival_Time'].unique()

[19]: array(['01:10 22 Mar', '13:15', '04:25 10 Jun', ..., '06:50 10 Mar',
             '00:05 19 Mar', '21:20 13 Mar'], dtype=object)

[20]: df1['Arrival_Time'],_ = df['Arrival_Time'].str.split(' ',1).str

[21]: df1['Arr_Hour'],df1['Arr_Minute'] = df1['Arrival_Time'].str.split(':',2).str

[22]: df1['Duration'] = df1['Duration'].str.replace('h ',':').str.replace('m','')
      df1['Duration_Hour'],df1['Duration_Minute'] = df1['Duration'].str.split(':',2).str

[23]: df1['Total_Stops'].unique()

[23]: array(['non-stop', '2 stops', '1 stop', '3 stops', '4 stops'],
            dtype=object)

[24]: df1['Total_Stops'],_ = df1['Total_Stops'].str.split(' stops').str
      df1['Total_Stops'],_ = df1['Total_Stops'].str.split(' stop').str
      df1['Total_Stops'] = df1['Total_Stops'].apply(lambda x: 0 if 'non-stop' in x else x)

[25]: df1['Additional_Info'].unique()

[25]: array(['No info', 'In-flight meal not included',
             'No check-in baggage included', '1 Short layover', 'No Info',
             '1 Long layover', 'Change airports', 'Business class',
             'Red-eye flight', '2 Long layover'], dtype=object)
```

```
[26]: df1['Additional_Info'] = df1['Additional_Info'].str.replace('No info','No Info')
      df1['Price'].describe()

[26]: count    10682.000000
      mean      9087.214567
      std       4611.548810
      min       1759.000000
      25%       5277.000000
      50%       8372.000000
      75%      12373.000000
      max      79512.000000
      Name: Price, dtype: float64

[27]: df1.to_csv('Cleaned.csv', index=False)
```

[1]:
```
# _______________________________________________________________ANALYSIS AND VISUAL REPRESENTATION OF DATA_______________________________________________________________

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', None)
```

[2]:
```
df = pd.read_csv('Cleaned.csv')
df.shape
```

[2]:
```
(10682, 20)
```

[3]:
```
df.head(10)
```

[3]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Day	Month	Year	Dep_Hour	Dep_Minute	Arr_Hour	Arr_Minute	Duration_Hour	Duration_Mi
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10	2:50	0	No Info	3897	24	3	2019	22	20	1	10	2	
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7:25	2	No Info	7662	1	5	2019	5	50	13	15	7	

[4]:
```
df.info()
```

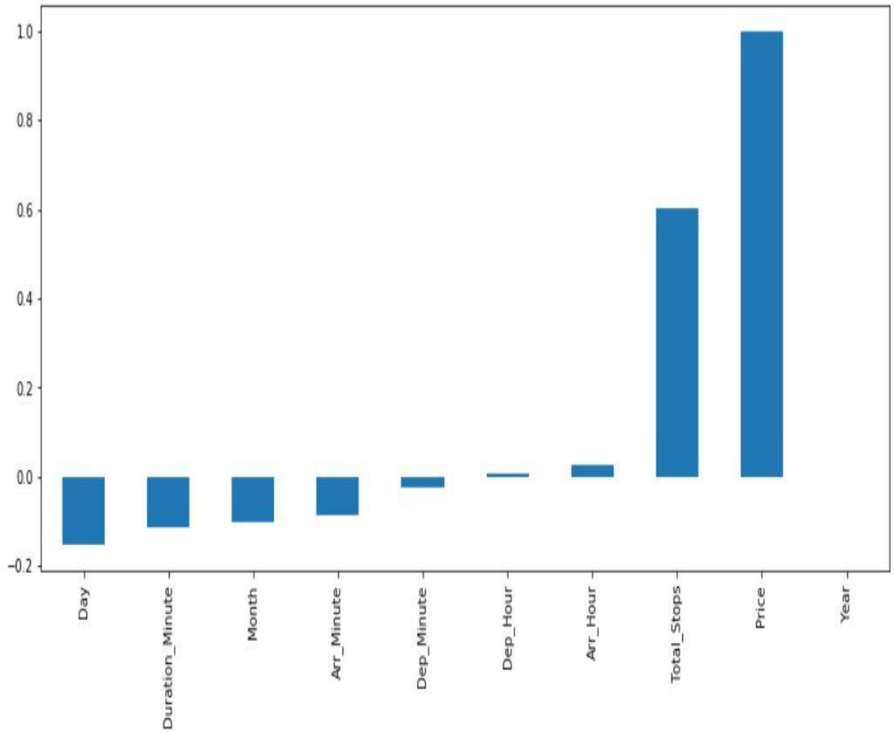
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10682 entries, 0 to 10681
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          10682 non-null  object
 1   Date_of_Journey  10682 non-null  object
 2   Source           10682 non-null  object
 3   Destination      10682 non-null  object
 4   Route            10682 non-null  object
 5   Dep_Time         10682 non-null  object
 6   Arrival_Time     10682 non-null  object
 7   Duration         10682 non-null  object
 8   Total_Stops      10682 non-null  int64
 9   Additional_Info  10682 non-null  object
 10  Price            10682 non-null  int64
 11  Day              10682 non-null  int64
 12  Month            10682 non-null  int64
 13  Year             10682 non-null  int64
 14  Dep_Hour         10682 non-null  int64
 15  Dep_Minute       10682 non-null  int64
 16  Arr_Hour         10682 non-null  int64
 17  Arr_Minute       10682 non-null  int64
 18  Duration_Hour    10682 non-null  object
 19  Duration_Minute  9650 non-null   float64
dtypes: float64(1), int64(9), object(10)
memory usage: 1.6+ MB
```

[5]:
```
df.describe()
```

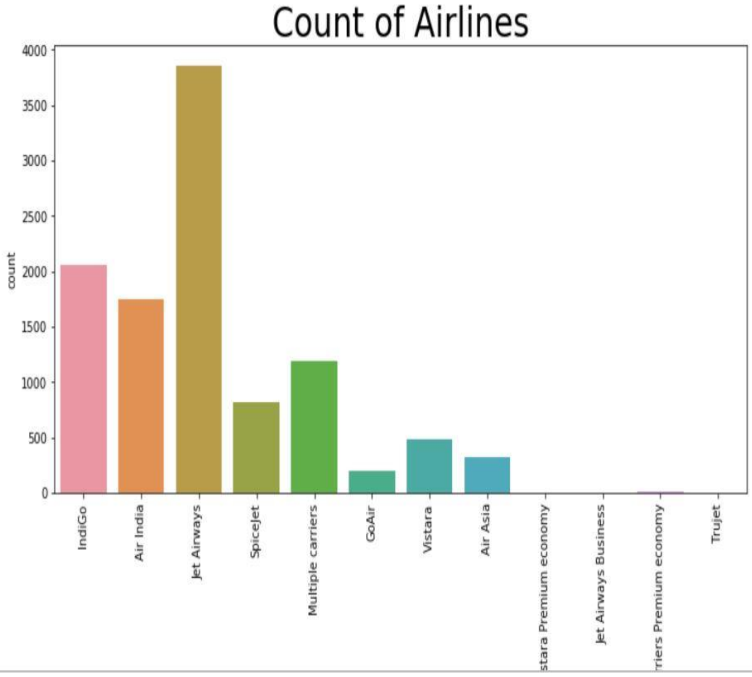
[5]:

	Total_Stops	Price	Day	Month	Year	Dep_Hour	Dep_Minute	Arr_Hour	Arr_Minute	Duration_Minute
count	10682.000000	10682.000000	10682.000000	10682.000000	10682.0	10682.000000	10682.000000	10682.000000	10682.000000	9650.00000
mean	0.824190	9087.214567	13.509081	4.708575	2019.0	12.491013	24.409287	13.349186	24.690601	31.35544
std	0.675229	4611.548810	8.479363	1.164408	0.0	5.748820	18.767801	6.859317	16.506808	14.93004
min	0.000000	1759.000000	1.000000	3.000000	2019.0	0.000000	0.000000	0.000000	0.000000	5.00000
25%	0.000000	5277.000000	6.000000	3.000000	2019.0	8.000000	5.000000	8.000000	10.000000	20.00000
50%	1.000000	8372.000000	12.000000	5.000000	2019.0	11.000000	25.000000	14.000000	25.000000	30.00000
75%	1.000000	12373.000000	21.000000	6.000000	2019.0	18.000000	40.000000	19.000000	35.000000	45.00000
max	4.000000	79512.000000	27.000000	6.000000	2019.0	23.000000	55.000000	23.000000	55.000000	55.00000

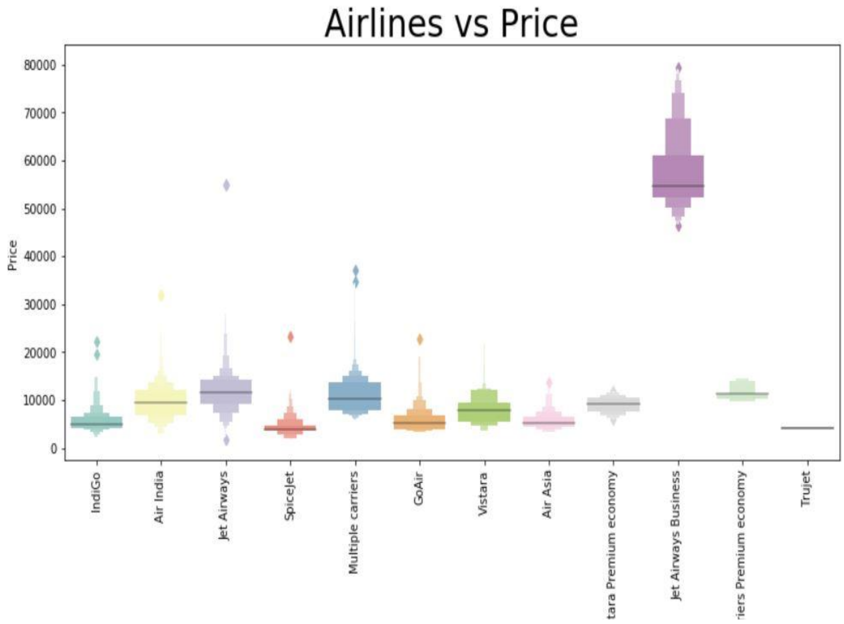
[6]:
```
plt.figure(figsize=(14,6))
df.corr()['Price'].sort_values().plot(kind='bar');
```



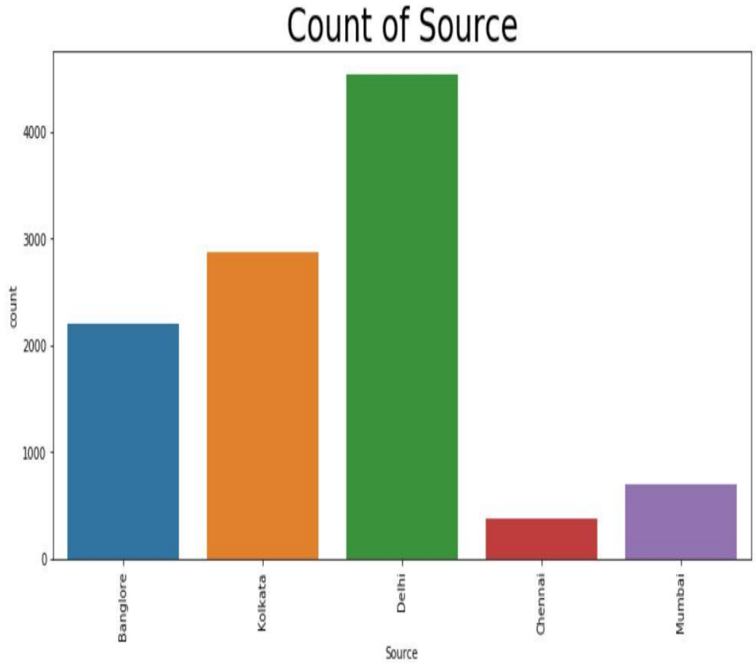
[7]:
```
plt.figure(figsize=(12,6))
sns.countplot(df['Airline'])
plt.title('Count of Airlines', size=30)
plt.xticks(rotation=90)
plt.show()
```



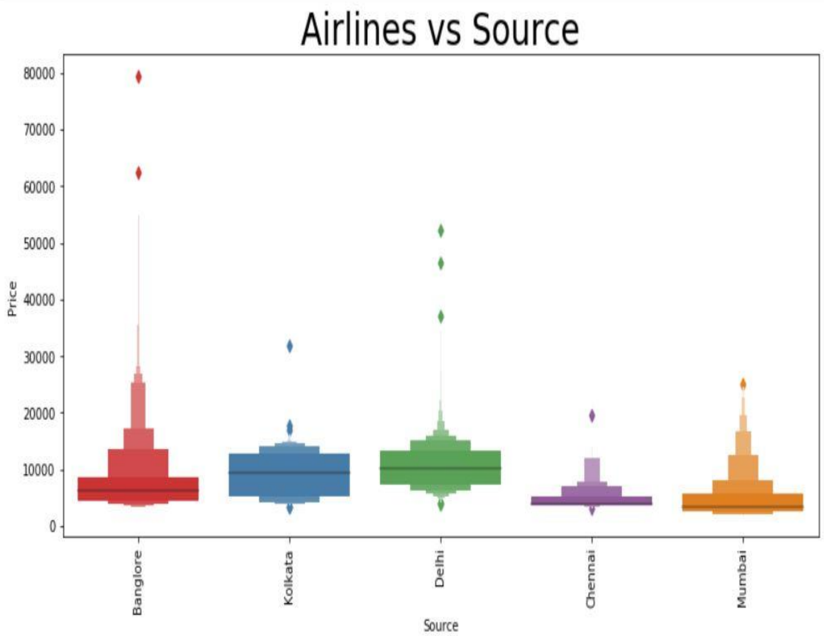
[8]:
```
plt.figure(figsize=(13,6))
sns.boxenplot(df['Airline'], df['Price'], palette='Set3')
plt.title('Airlines vs Price', size=30)
plt.xticks(rotation=90)
plt.show()
```



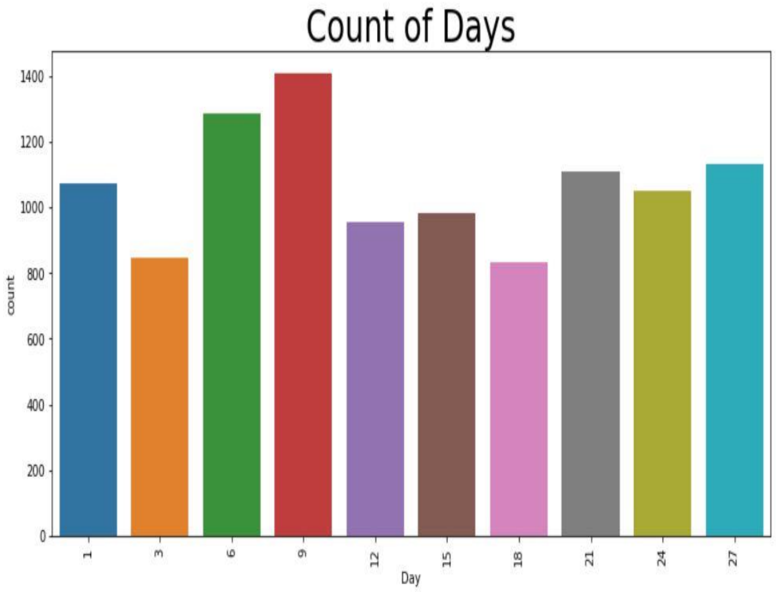
```
[9]: plt.figure(figsize=(13,6))
     sns.countplot(df['Source'])
     plt.title('Count of Source', size=30)
     plt.xticks(rotation=90)
     plt.show()
```



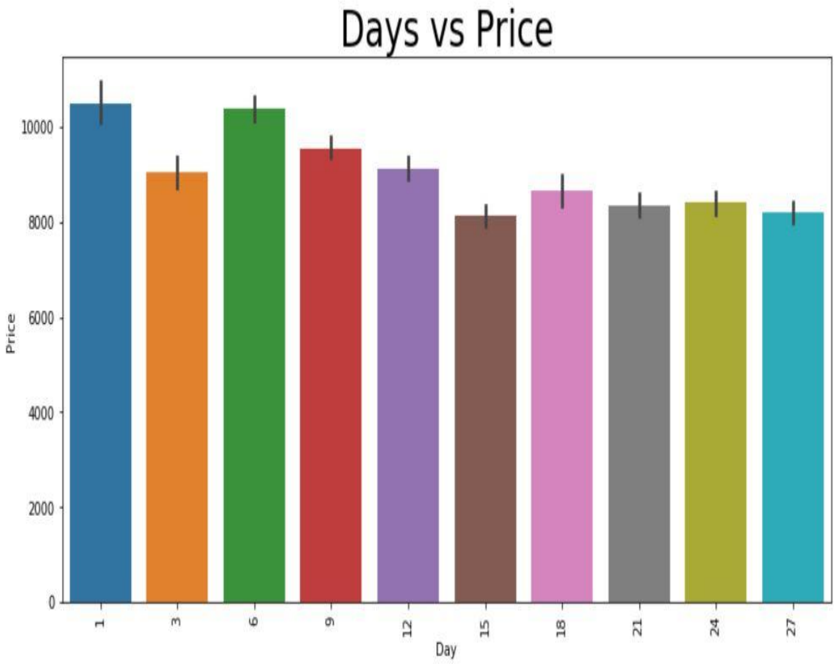

```
[10]: plt.figure(figsize=(13,6))
      sns.boxenplot(df['Source'], df['Price'], palette='Set1')
      plt.title('Airlines vs Source', size=30)
      plt.xticks(rotation=90)
      plt.show()
```



[11]:
```
plt.figure(figsize=(13,6))
sns.countplot(df['Day'])
plt.title('Count of Days', size=30)
plt.xticks(rotation=90)
plt.show()
```



[12]:
```
plt.figure(figsize=(13,6))
sns.barplot(df['Day'], df['Price'])
plt.title('Days vs Price', size=30)
plt.xticks(rotation=90)
plt.show()
```



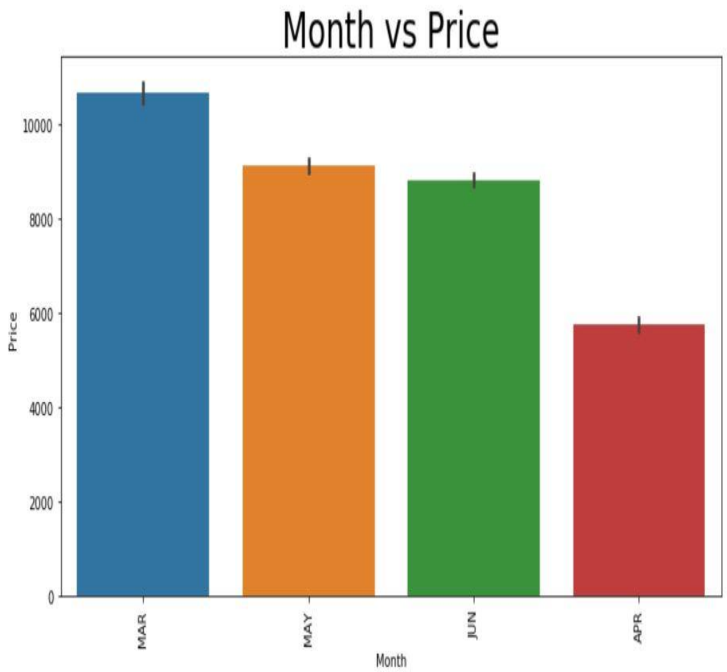
```
[13]: df['Month'] = df['Month'].map({
    1:'JAN',
    2:'FEB',
    3:'MAR',
    4:'APR',
    5:'MAY',
    6:'JUN',
    7:'JUL',
    8:'AUG',
    9:'SEP',
    10:'OCT',
    11:'NOV',
    12:'DEC'
})
```

```
[14]: df.head(2)
```

[14]:

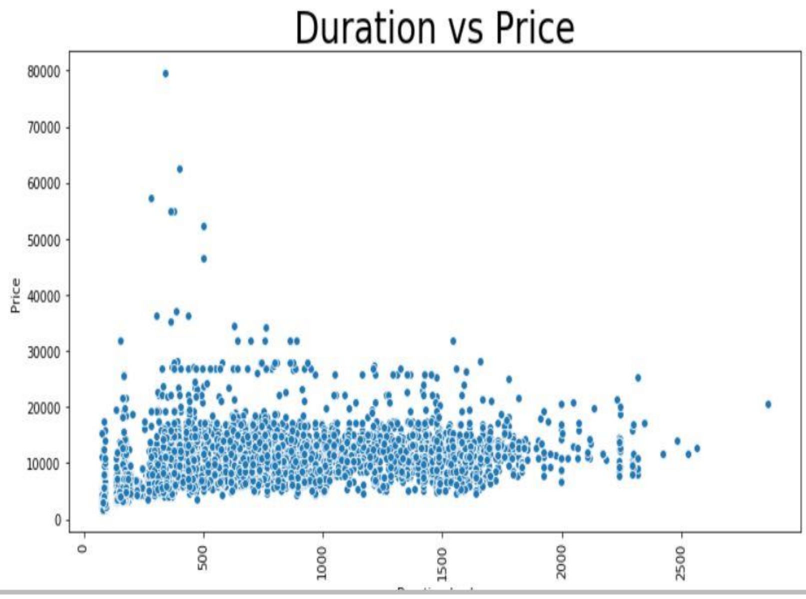
	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Day	Month	Year	Dep_Hour	Dep_Minute	Arr_Hour	Arr_Minute	Duration_Hour	Duration_Mi
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10	2:50	0	No Info	3897	24	MAR	2019	22	20	1	10	2	
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7:25	2	No Info	7662	1	MAY	2019	5	50	13	15	7	
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25	19h	2	No Info	13882	9	JUN	2019	9	25	4	25	19h	

```
[15]: plt.figure(figsize=(12,6))
sns.barplot(df['Month'], df['Price'])
plt.title('Month vs Price', size=30)
plt.xticks(rotation=90)
plt.show()
```



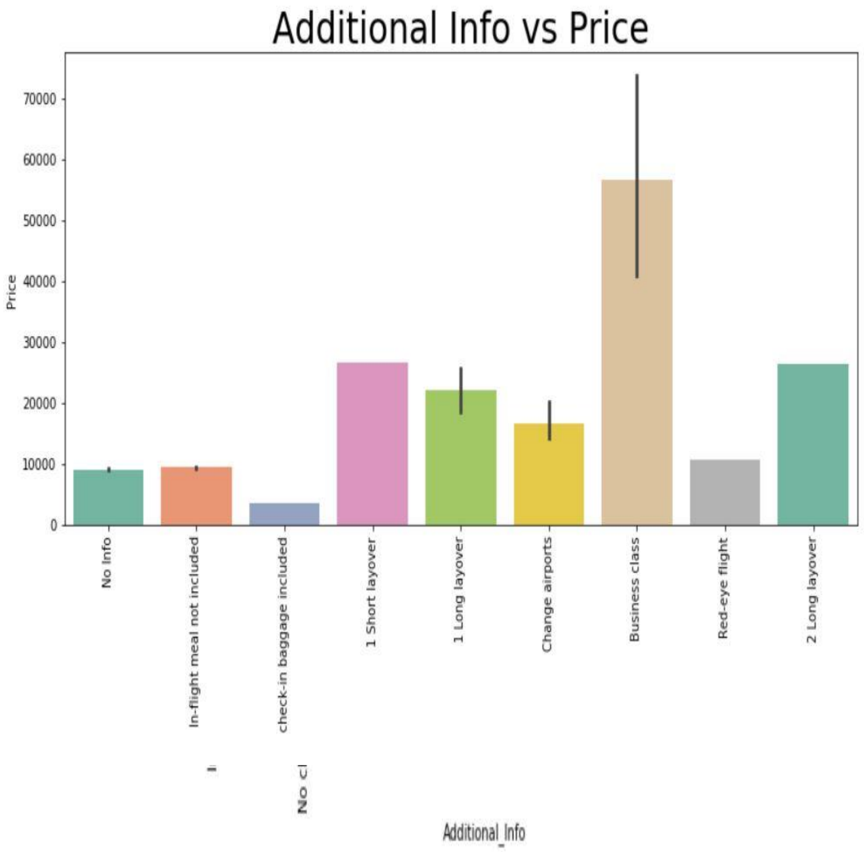
[16]:
```
df['Duration_Hour'],_ = df['Duration_Hour'].str.split('h',1).str
df['Duration_Hour'] = df['Duration_Hour'].astype(int)
df['Duration_bool'] = (df['Duration_Hour']*60)+df['Duration_Minute']
```

[17]:
```
plt.figure(figsize=(12,6))
sns.scatterplot(df['Duration_bool'], df['Price'], palette='Set2')
plt.title('Duration vs Price', size=30)
plt.xticks(rotation=90)
plt.show()
```



[18]:
```
plt.figure(figsize=(12,6))
sns.barplot(df['Total_Stops'], df['Price'], palette='Set2')
plt.title('Stops vs Price', size=30)
plt.xticks(rotation=90)
plt.show()
```

```
[19]: plt.figure(figsize=(13,6))
      sns.barplot(df['Additional_Info'], df['Price'], palette='Set2')
      plt.title('Additional Info vs Price', size=30)
      plt.xticks(rotation=90)
      plt.show()
```




```
[20]: ncol=["Duration_bool"]
      for i in ncol:
          q75, q25 = np.percentile(df.loc[:,i], [75 ,25])
          iqr = q75 - q25
          min = q25 - (iqr*1.5)
          max = q75 + (iqr*1.5)
          df = df.drop(df[df.loc[:,i] <= min].index)
          df = df.drop(df[df.loc[:,i] >= max].index)
```

```
[21]: df = df.dropna()
```

```
[22]: df.to_csv('Final.csv', index=None)
```

[25]:

```
#_______________________________________________________________MODELS USED FOR PREDICTION OF FARE________________________________________________________________________________

import pandas as pd
import numpy as np

from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score, RandomizedSearchCV
#from lazypredict.Supervised import LazyRegressor
from sklearn.linear_model import LinearRegression, ElasticNet, Lasso, Ridge, HuberRegressor, LogisticRegression, BayesianRidge
from sklearn.tree import DecisionTreeRegressor, ExtraTreeRegressor
from sklearn.ensemble import RandomForestRegressor, AdaBoostRegressor, GradientBoostingRegressor, VotingRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error, mean_squared_error

import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', None)
```

[6]:

```
df = pd.read_csv('Final.csv')
df.shape
```

[6]: (9650, 21)

[7]:

```
df.head(5)
```

[7]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Day	Month	Year	Dep_Hour	Dep_Minute	Arr_Hour	Arr_Minute	Duration_Hour	Duration_Mi
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10	2:50	0	No Info	3897	24	MAR	2019	22	20	1	10	2	
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7:25	2	No Info	7662	1	MAY	2019	5	50	13	15	7	
2	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5:25	1	No Info	6218	12	MAY	2019	18	5	23	30	5	
3	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4:45	1	No Info	13302	1	MAR	2019	16	50	21	35	4	
4	SpiceJet	24/06/2019	Kolkata	Banglore	CCU → BLR	09:00	11:25	2:25	0	No Info	3873	24	JUN	2019	9	0	11	25	2	

[8]:
```
df.columns
```

[8]:
```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info', 'Price', 'Day', 'Month', 'Year', 'Dep_Hour',
       'Dep_Minute', 'Arr_Hour', 'Arr_Minute', 'Duration_Hour',
       'Duration_Minute', 'Duration_bool'],
      dtype='object')
```

[9]:
```
df1 = df[['Airline', 'Source', 'Destination', 'Total_Stops',
         'Additional_Info', 'Price', 'Day', 'Month', 'Duration_bool']]
df1.shape
```

[9]:
```
(9650, 9)
```

10]:
```
df1.head()
```

10]:

	Airline	Source	Destination	Total_Stops	Additional_Info	Price	Day	Month	Duration_bool
0	IndiGo	Banglore	New Delhi	0	No Info	3897	24	MAR	170.0
1	Air India	Kolkata	Banglore	2	No Info	7662	1	MAY	445.0
2	IndiGo	Kolkata	Banglore	1	No Info	6218	12	MAY	325.0
3	IndiGo	Banglore	New Delhi	1	No Info	13302	1	MAR	285.0
4	SpiceJet	Kolkata	Banglore	0	No Info	3873	24	JUN	145.0

[11]:
```
df1 = df1.rename(columns={'Duration_bool': 'Duration'})
```

[13]:
```
df1.isnull().any().any()
```

[13]:
```
False
```

[14]:
```
df1['Month'] = df1['Month'].map({
    'JAN':1,
    'FEB':2,
    'MAR':3,
    'APR':4,
    'MAY':5,
    'JUN':6,
    'JUL':7,
    'AUG':8,
    'SEP':9,
    'OCT':10,
    'NOV':11,
    'DEC':12
})
```

[15]:
```
df1['Additional_Info'] = df1['Additional_Info'].map({
    'No Info':0,
    'In-flight meal not included':1,
    'No check-in baggage included':1,
    '1 Short layover':3,
    '1 Long layover':4,
    'Change airports':5,
    'Business class':6,
    'Red-eye flight':7,
    '2 Long layover':8
})
```

```
[16]: dummies = pd.get_dummies(df1[['Airline', 'Source', 'Destination']])
```

```
[17]: df2 = pd.concat([df1,dummies], axis=1)
      df2.shape
```

```
[17]: (9650, 32)
```

```
[18]: df2 = df2.drop(['Airline', 'Source', 'Destination'], axis=1)
      df2.shape
```

```
[18]: (9650, 29)
```

```
[19]: df2.head()
```

[19]:

	Total_Stops	Additional_Info	Price	Day	Month	Duration	Airline_Air Asia	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJet	Airline_Trujet	Airline_Vistara	Airline P e
0	0	0	3897	24	3	170.0	0	0	0	1	0	0	0	0	0	0	0	
1	2	0	7662	1	5	445.0	0	1	0	0	0	0	0	0	0	0	0	
2	1	0	6218	12	5	325.0	0	0	0	1	0	0	0	0	0	0	0	
3	1	0	13302	1	3	285.0	0	0	0	1	0	0	0	0	0	0	0	
4	0	0	3873	24	6	145.0	0	0	0	0	0	0	0	0	1	0	0	

```
[20]:
      df2['Additional_Info'].unique()
```

```
[20]: array([0, 1, 3, 4, 5, 6, 7, 8], dtype=int64)
```

```
[21]: df2.columns
```

```
[21]: Index(['Total_Stops', 'Additional_Info', 'Price', 'Day', 'Month', 'Duration',
             'Airline_Air Asia', 'Airline_Air India', 'Airline_GoAir',
             'Airline_IndiGo', 'Airline_Jet Airways', 'Airline_Jet Airways Business',
             'Airline_Multiple carriers',
             'Airline_Multiple carriers Premium economy', 'Airline_SpiceJet',
             'Airline_Trujet', 'Airline_Vistara', 'Airline_Vistara Premium economy',
             'Source_Banglore', 'Source_Chennai', 'Source_Delhi', 'Source_Kolkata',
             'Source_Mumbai', 'Destination_Banglore', 'Destination_Cochin',
             'Destination_Delhi', 'Destination_Hyderabad', 'Destination_Kolkata',
             'Destination_New Delhi'],
            dtype='object')
```

```
[22]: X = df2.drop('Price', axis=1)
      y = df2['Price']
```

```
[23]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
      X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
[23]: ((6755, 28), (2895, 28), (6755,), (2895,))
```

```
[26]: models = [['LinearRegression : ', LinearRegression()],
                ['ElasticNet :', ElasticNet()],
                ['Lasso : ', Lasso()],
                ['Ridge : ', Ridge()],
                ['KNeighborsRegressor : ', KNeighborsRegressor()],
                ['DecisionTreeRegressor : ', DecisionTreeRegressor()],
                ['RandomForestRegressor : ', RandomForestRegressor()],
                ['SVR : ', SVR()],
                ['AdaBoostRegressor : ', AdaBoostRegressor()],
                ['GradientBoostingRegressor : ', GradientBoostingRegressor()],
                ['ExtraTreeRegressor : ', ExtraTreeRegressor()],
                ['HuberRegressor : ', HuberRegressor()],
                ['BayesianRidge : ', BayesianRidge()]]
```

```
[27]: for name, model in models:
          model=model
          model.fit(X_train, y_train)
          predictions = model.predict(X_test)
          print(name, (np.sqrt(mean_squared_error(y_test, predictions))))
```

```
LinearRegression :  2779.0455708889162
ElasticNet : 3379.6819876610443
Lasso :  2759.449381312224
Ridge :  2710.847612774103
KNeighborsRegressor :  3249.005561971264
DecisionTreeRegressor :  2076.9502626135522
RandomForestRegressor :  1667.3153160359643
SVR :  4246.460099935076
AdaBoostRegressor :  3338.219679937957
GradientBoostingRegressor :  1904.9193087392046
ExtraTreeRegressor :  2052.902735812673
HuberRegressor :  3127.29660899842
BayesianRidge :  2773.2755615168767
```

```
[28]: algorithms = {
          'RandomForestRegressor' : {
              'model' : RandomForestRegressor(),
              'param' : {
                  'n_estimators' : [300, 500, 700, 1000, 2100],
                  'max_depth' : [3, 5, 7, 9, 11, 13, 15],
                  'max_features' : ["auto", "sqrt", "log2"],
                  'min_samples_split' : [2, 4, 6, 8]
              }
          },
          'GradientBoostingRegressor' : {
              'model' : GradientBoostingRegressor(),
              'param' : {
                  'learning_rate' : [0.5, 0.8, 0.1, 0.20, 0.25, 0.30],
                  'n_estimators' : [300, 500, 700, 1000, 2100],
                  'criterion' : ['friedman_mse', 'mse']
              }
          }
      }
```

```
[29]: score = []

      for name, mp in algorithms.items() :
          rs = RandomizedSearchCV(estimator = mp['model'], param_distributions = mp['param'], cv = 10, n_jobs=-1, verbose=3)
          rs.fit(X_train, y_train)
          score.append({
              'model': name,
              'score' : rs.best_score_,
              'params' : rs.best_params_
          })
```

```
Fitting 10 folds for each of 10 candidates, totalling 100 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  16 tasks      | elapsed:   31.0s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:  2.6min finished
```

```
[30]: final = pd.DataFrame(score, columns=['model', 'score', 'params'])
      final
```

[30]:

	model	score	params
0	RandomForestRegressor	0.856524	{'n_estimators': 300, 'min_samples_split': 2, ...
1	GradientBoostingRegressor	0.864565	{'n_estimators': 300, 'learning_rate': 0.5, 'c...

```
[32]: final['params'][1]

[32]: {'n_estimators': 300, 'learning_rate': 0.5, 'criterion': 'friedman_mse'}

[33]: regressor = GradientBoostingRegressor(n_estimators = 500, learning_rate = 0.3, criterion = 'friedman_mse')
      regressor.fit(X_train, y_train)
      prediction = regressor.predict(X_test)
      print('RMSE : {}'.format(np.sqrt(mean_squared_error(y_test, prediction))))

      RMSE : 1649.1882310608464

[34]: regressor.score(X_train, y_train), regressor.score(X_test, y_test)

[34]: (0.917514528615212, 0.8690424873855073)

[35]: prediction[0]    # Predicted price for  first entry  in the data

[35]: 4742.384163142352

[36]: df2['Price'][0]    #Original price of the first entry in the data

[36]: 3897
```

```
[37]: print('MAE:', mean_absolute_error(y_test, prediction))
      print('MSE:', mean_squared_error(y_test, prediction))
      print('RMSE:', np.sqrt(mean_squared_error(y_test, prediction)))

      MAE: 961.4355665887581
      MSE: 2719821.8214696036
      RMSE: 1649.1882310608464

[41]: prediction [2]

[41]: 13440.153441011267

[43]: df2['Price'][2]

[43]: 6218
```

CHAPTER 11
CONCLUSION

Choosing the right system gaining knowledge of technique depends on the hassle type, size of a dataset, assets, and many others. A great practice is to use several fashions to both streamline assessment and reap higher accuracy.

With this project, we are developing a machine learning model that can predict flight prices. The Gradient Boosting Regression model and the Random Tree Regressor are giving the best accuracy. Most accurate results are given by Gradient boost Regressor. Also, it was observed that from the data collected and through exploratory data analysis, we can determine the following:

The trend of flight prices varies over various months and across the holiday

There are two groups of airlines: the economical group and the luxurious group. SpiceJet, AirAsia, IndiGo, Go Air are in the economical class, whereas Jet Airways and Air India in the other. Vistara has a more spread-out trend.

The airfare varies depending on the time of departure, making timeslot used in analysis an important parameter.

The airfare increases during a holiday season. In our time period, during Diwali the fare remained high for all the values of days to departure. We haven’t considered holiday season as a parameter now, since we are looking at data for a few months.

Airfare varies according to the day of the week of travel. It is higher for weekends and Monday and slightly lower for the other days

CHAPTER 11

FUTUREWORK

● More routes can be added and similar analysis can be extended to major airports and routes in India.

● Analysis can be done by increasing the data points and increasing the historical data used. That will train the model by better providing relevant details and more savings.

● Many rules can be added to Rule-based education based on our understanding of the industry, including the delivery times provided by airlines.

● Creating an easy-to-use interface for various routes that provides additional flexibility for users.

REFERENCES

[1] Abhilash, Ranjana,shilpa and ZubedaSurvey on Air Price Prediction using Machine Learning Algorithm, IJIREEICE 2019 .

[2] B. Smith, J. Leimkuhler, R. Darrow, and Samuels,―Yield management at american airlines,‖Interfaces, vol.22, pp. 8–31, 1992.

[3] Bingchuan Liu, Yudone Tan and Humine Zhou, A Baysian predictor of Airline class Seats Based on Multinomial Event Model, International conference on Big Data 2016.

[4] C. Koopmans and R. Lieshout, "Airline cost changes: To whatextent are they passed through to the passenger?" Journal ofAir Transport Management, vol. 53, pp. 1–11, 2016

[5] D. P. Kingma and J. Ba, "Adam: A method forstochastic optimization," in the 3rd international conferenceon learning representations, 2015. [Online]. Available:http://arxiv.org/abs/1412.698

[6] Dominguez-Menchero, J.Santo, Riviera, ‖optimal purchase timing in airline markets‖ ,2014

[7] G. Francis, A. Fidato, and I. Humphreys, "Airport–airlineinteraction: the impact of low-cost carriers on two europeanairports," Journal of Air Transport Management, vol. 9, no. 4,pp. 267–273, 2003.

[8] H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola and V. Vapnik, ‖Support vector regression machines,‖ Advances in neural information processing systems, vol. 9, pp. 155-161, 1997.

[9] International Civil Aviation Organization, "List of low-cost-carriers (LCCs)," cited July 2018. [Online]. Available:https://www.icao.int/sustainability/Documents/LCC-List.pdf

[10] K. Tziridis, K.I. Diamantaras,Airfare Prices Prediction Using machine Learning Technique,European signal processing conference 2017.

[11] L. Breiman, ―Random forests,‖ Machine Learning, vol. 45, pp. 5- 32 , 2001.

[12] S. Lee, K. Seo, and A. Sharma, "Corporate social respon-sibility and firm performance in the airline industry: Themoderating role of oil prices," Tourism Management, vol. 38,pp. 20–30, 2013.

[13] S.B. Kotsiantis, ―Decision trees: a recent overview,‖ Artificial Intelligence Review, vol. 39, no. 4, pp. 261-283, 2013.

[14] S. Haykin, Neural Networks – A Comprehensive Foundation. Prentice Hall, 2nd Edition, 1999.

[15] T. Janssen, ―A linear quantile mixed regression model for prediction of airline ticket prices,‖ Bachelor Thesis, Radboud University, 2014.

[16] V. Nair and G. E. Hinton, "Rectified linear units improverestricted boltzmann machines," in the 27th internationalconference on machine learning, 2010, pp. 807–814.

[17] Viet Hoang Vu, Quang Tran Minh and Phu H. Phung,‖An Airfare Prediction Model for Developing Markets‖, IEEE paper 2018.

[18] W. Groves and M. Gini, ―An agent for optimizing airline ticket purchasing,‖ 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), St. Paul, MN, May 06 - 10, 2013 , pp. 1341-1342.

[19] William Grooves and Maria Gini,A regression model for predicting optimal purchase timing for airline tickets,University of Minnesota 2011.[i]

CV

DEEPTI BANSAL

EDUCATION

- **BABU BANARASI DAS ENGINEERING COLLEGE**
  B. Tech in Computer Science & Engineering (2017-2021)
  [80.50% (Till 6th Semester)]
- **Navayuga Radiance Senior Secondary School**
  Completed Intermediate (2016)
  [60.8%]
- **Navayuga Radiance Senior Secondary School**
  Completed High School (2014)
  [74.1%]

ACHIEVEMENTS & CERTIFICATIONS

- Cleared Test for MTA Introduction To Python Programming.
- Created a homepage for the website using HTML/CSS.
- Won 1 prize for Nukkad in college Annual Fest UTKARSH in 2019 &2020
- Won 1 prize for Nukkad in IIT Kanpur Road Trip 2019.

TRAINING / PROJECT

PYTHON Training - Completed

Airlines data Analysis Using Data Science With Python

- **From:** ICT ,IITKanpur
- **Tools Used:**Pycharm
- **Start Date:** June 2019
- **End Date:** July 2019

MACHINE LEARNING USING PYTHON Training - Completed

- **From:** APTRON Solutions Pvt. Ltd.
- **Tools Used:** pycharm and Anaconda
- **Start Date:** June 2020
- **End Date**: July 2020

PROJECT USING PYTHON WITH ML Project - Ongoing

- **Tools Used**: Jupyter and Spyder
- **Start Date** : Nov 2020

PROFILE

Self-Motivated and Hardworking Graduate seeking an opportunity to work in a challenging environment to prove my Coding Skills and utilize my knowledge of various Databases for the growth of the Organization.

CONTACT

PHONE:
7985372601

EMAIL:
Deeptibansal3910@gmail.com

SKILLS

Programming Skills

- Python
- C
- HTML/CSS

Database

- MySQL

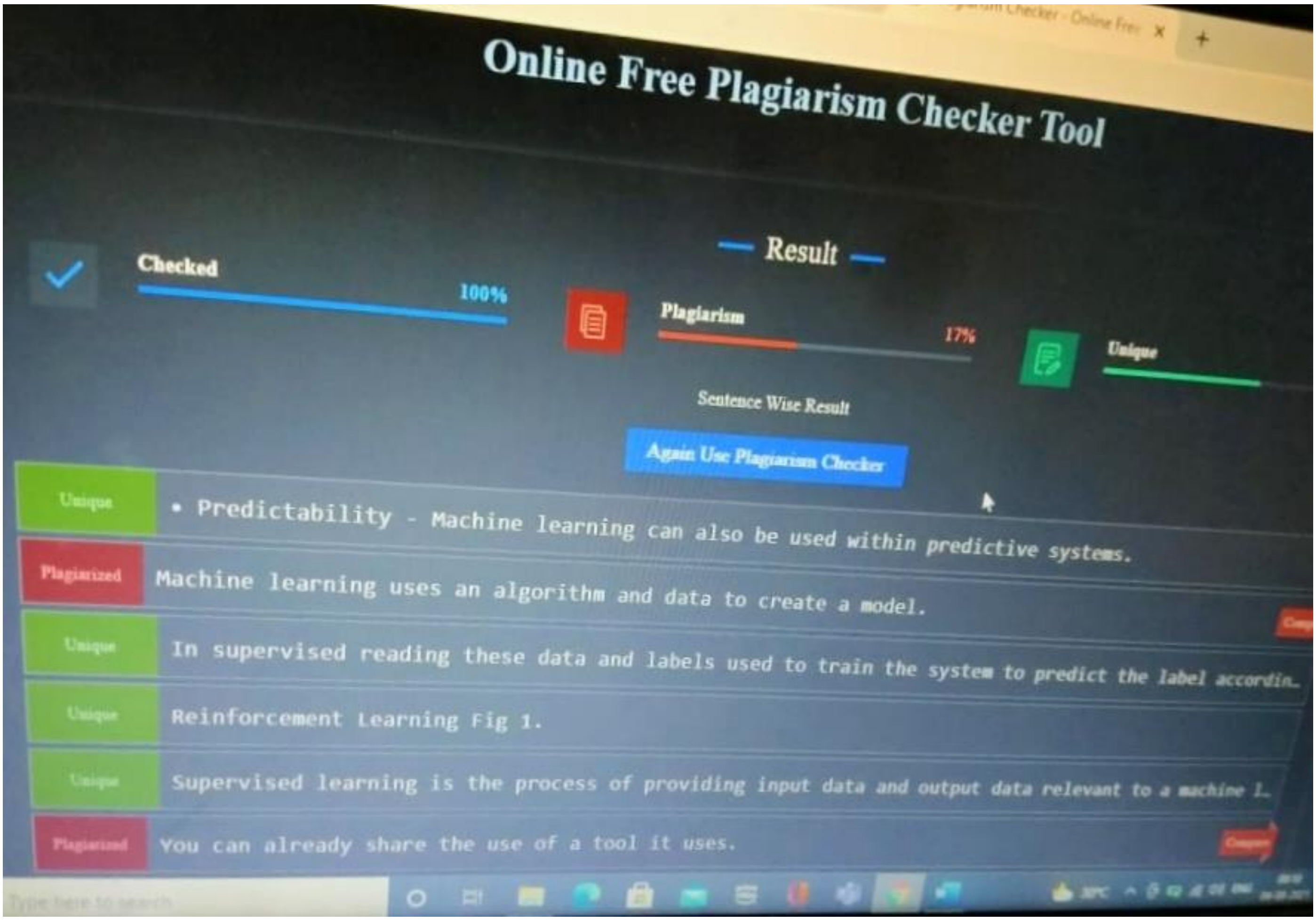
Operating System

- Windows
- Linux(Still Learning)

LANGUAGES KNOWN

- Hindi (Mother Tongue)
- English

PLAG REPORT



i