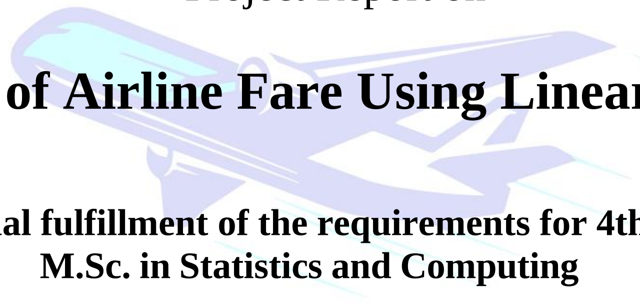
BANARAS HINDU UNIVERSITY

DST-CIMS

DEPARTMENT OF STATISTICS & COMPUTING

Project Report on



Prediction of Airline Fare Using Linear Regression

In partial fulfillment of the requirements for 4th semester
M.Sc. in Statistics and Computing

Under the guidance of
Prof. Piyush Kant Rai
Department of Statistics
Institute of Science
Banaras Hindu University

Submitted by
GARGI MAJUMDER
Exam Roll No.: 22419STC014
Enrollment No.: 453276
Statistics and Computing
DST-CIMS, Institute of Science
Banaras Hindu University

Acknowledgements

I am happy to acknowledge the help I received encouragement from many in preparing this dissertation. Completion of this dissertation was not possible without the support of them.

First and Foremost, I would like to express my sincere gratitude to the Head of Course Coordinator of Statistics and Computing Dr. Rakesh Ranjan,DST-CIMS, Institute of Science, B.H.U. Varanasi, for his valuable guidance, scholarly inputs and consistent encouragement I received throughout the dissertation work.

Secondly, I wish to put my profound sense of gratitude to my research guide Prof. Piyush Kant Rai, Department of Statistics, Institute of Science, B.H.U. Varanasi, for giving supervision and persistence guidance to me. His motivation and constant interest throughout my work has facilitated me to conduct the present work in an uninterrupted and timely manner.

One's gratefulness to parents and family cannot be expressed in words. I am highly indebted to Mr. Goutam Majumder; my father and Mrs. Manjari Majumder; my mother for their love, patience and motivation, which enabled me to pursue my studies without any hindrance whatsoever. I shall be failing in my duty if I do not express my gratitude to all my family members.

In the end, I would like to express my heartfelt thanks to all who have directly and indirectly supported me throughout my work.

Date:16.05.2024
Place: BHU, Varanasi

GARGI MAJUMDER
M.Sc. Statistics & Computing, 2nd Year
Enrollment No.: 453276
Exam Roll No.: 22419STC014

DECLARATION

I **Gargi Majumder** hereby declare that the dissertation, titled **"Prediction of Airline Fare Using Linear Regression"** is a record of research work undertaken by me in partial fulfillment of the requirements for 4th semester M.Sc. in Statistics and Computing. I have completed this study under the supervision of Prof. Piyush Kant Rai, Department of Statistics, Banaras Hindu University, Varanasi.

I also declare that this dissertation has not been submitted for the award of any degree, diploma, fellowship or other title. It has not been sent for any publication or presentation purpose. I hereby confirm the originality of the work and that there is no plagiarism in any part of the dissertation.

Place: Varanasi

Date: 16.05.2024

Signature of the candidate

GARGI MAJUMDER

Exam Roll No.: 22419STC014

Enrollment No.: 453276

Statistics and Computing

DST-CIMS, Institute of Science

Banaras Hindu University

काशी हिन्दू विश्वविद्यालय

BANARAS HINDU UNIVERSITY

AN INSTITUTION OF NATIONAL IMPORTANCE ESTABLISHED BY AN ACT OF PARLIAMENT

CERTIFICATE

This is to certify that the dissertation submitted by **GARGI MAJUMDER**, Roll No. **22419STC014**, Enrollment No. **453276**, Department of Statistics and Computing, DST-CIMS, Institute of Science, Banaras Hindu University, Varanasi, titled **"Prediction of Airline Fare Using Linear Regression"** is a record of research work done by her during 4th semester under my/our supervision in partial fulfillment for the award of the degree of M.Sc. in Statistics and Computing.

This dissertation has not been submitted for the award of any degree, diploma, fellowship or other title. It has not been sent for any publication or presentation purpose. I hereby confirm the originality of the work and that there is no plagiarism in any part of the dissertation.

Place: Varanasi
Date: 16.05.2024

Signature of Supervisor
Prof. Piyush Kant Rai
Department of Statistics
Institute of Science
Banaras Hindu University, Varanasi
Date:

Contents

Chapter1. Introduction

In today's fast-paced world, where time is of the essence, effective travel planning has become essential for people looking for savings and convenience. Imagine a scenario where we have an upcoming trip and need to book a flight. We have various options with different departure times, layovers, and prices. In such a scenario, having the ability to accurately predict flight prices can greatly aid in decision-making and ensure an optimal travel experience.

This project aims to create a model to predict trip costs using information from reliable sources, like airline databases or historical flight records. To ensure accuracy and relevance, the dataset used for this project will be carefully selected and pre-processed. The prediction model will be trained using important parameters including departure time, destination, airline, and past pricing data.

In today's digital age, where travel booking platforms and airline websites provide real-time price estimates, understanding the underlying mechanisms behind flight price prediction is crucial. By employing supervised machine learning techniques, we aim to unravel the patterns and trends within the data to create a reliable model capable of forecasting flight prices with precision.

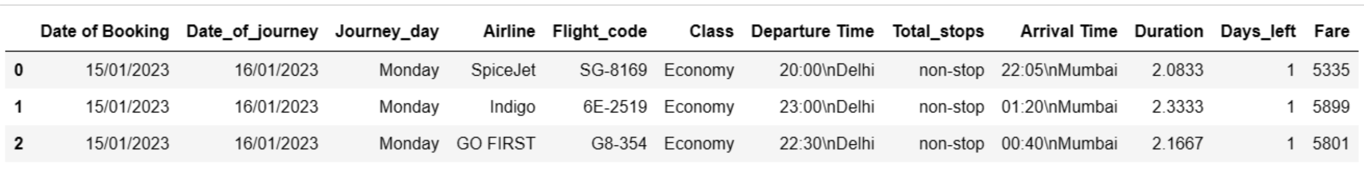
1.1 Objective

My objective is to use linear regression to analyze and predict airfare data based on various factors including total stops, duration, journey date, journey day, airline, class, source, destination, and departure time.

1.2 Dataset Description

The data is taken from the "Ease My Trip" website. 'Easemytrip' is an internet platform for booking flight tickets, and hence a platform that potential passengers use to buy tickets.

The data comprises of 12 columns namely Date of Booking, Date of journey, Journey day, Airline, Flight code, Class, Departure Time, Total stops, Arrival Time, Duration, Days left, Fare.



	Date of Booking	Date_of_journey	Journey_day	Airline	Flight_code	Class	Departure Time	Total_stops	Arrival Time	Duration	Days_left	Fare
0	15/01/2023	16/01/2023	Monday	SpiceJet	SG-8169	Economy	20:00\nDelhi	non-stop	22:05\nMumbai	2.0833	1	5335
1	15/01/2023	16/01/2023	Monday	Indigo	6E-2519	Economy	23:00\nDelhi	non-stop	01:20\nMumbai	2.3333	1	5899
2	15/01/2023	16/01/2023	Monday	GO FIRST	G8-354	Economy	22:30\nDelhi	non-stop	00:40\nMumbai	2.1667	1	5801

The dataset includes data recorded between January and March of 2023.

Date of Booking: The date when the booking was made.

Date of Journey: Denotes the date when the journey is scheduled to take place.

Journey day: The day of the week for the journey date (e.g., Monday, Tuesday, etc.).

Airline: The name of the airline providing the flight service.

Flight code: This is the alphanumeric code assigned to a specific flight by the airline.

Class: This column denotes the class of service booked for the flight (e.g., Economy, Business).

Departure Time: Indicates the time at which the flight is scheduled to depart.

Total stops: The total number of stops the flight will make during the journey.

Arrival Time: The time at which the flight is scheduled to arrive at its destination.

Duration: Represents the duration of the flight, usually in hours and minutes.

Days Left: This could be a derived column indicating the number of days left between the booking date and the journey date.

Fare: This column represents the cost or fare associated with the booking.

o Flight Price Prediction Using Machine Learning

Ankita Panigrahi, Rakesh Sharma, Sujata Chakravarty, Bijay K. Paikaray and Harshvardhan Bhoyar

123Dept. of CSE, Centurion University of Technology and Management, Odisha, India. 4School of Information & Communication Technology, Medhavi Skills University, Sikkim, India 5Faculty. of Management Studies, Sri Sri University, Odisha, India

travelling through a plane change now and then which also includes the day and night time. Additionally, it changes with special times of the year or celebration seasons. There are a few unique elements upon which the cost of air transport depends. The salesperson has data regarding each of the variables, however, buyers can get confined information which is not sufficient to foresee the airfare costs. Considering the provisions, for example, time of the day, the number of days remaining and the time of take-off this will provide the perfect time to purchase the plane ticket. The motivation behind this paper is to concentrate on every component that impacts the variations in the costs of this means of transport and how these are connected with the diversity in the airfare. Subsequently, at that point, utilizing this data, construct a framework that can help purchasers when to purchase a ticket. Machine Learning algorithms prove to be the best solution for the above-discussed problems. In this project, there is an implementation of Artificial Neural Network (ANN), LR (Linear Regression), DT (Decision Tree), and RF (Random Forest).

o A Framework for Airfare Price Prediction: A Machine Learning Approach

Tianyi Wang , Samira Pouyanfar , Haiman Tian , Yudong Tao , Miguel Alonso Jr. , Steven Luis and Shu-Ching Chen School of Computing and Information Sciences Florida International University

The price of an airline ticket is affected by a number of factors, such as flight distance, purchasing time, fuel price, etc. Each carrier has its own proprietary rules and algorithms to set the price accordingly. Recent advance in Artificial Intelligence (AI) and Machine Learning (ML) makes it possible to infer such rules and model the price variation. This paper proposes a novel application based on two public data sources in the domain of air transportation: the Airline Origin and Destination Survey (DB1B) and the Air Carrier Statistics database (T-100). The proposed framework combines the two databases, together with macroeconomic data, and uses machine learning algorithms to model the quarterly average ticket price based on different origin and destination pairs, as known as the market segment. The framework achieves a high prediction accuracy with 0.869 adjusted R squared score on the testing dataset.

Firstly, I shall check for null values and try to replace or remove them. Also, I need to see if there are any duplicate values in my dataset. Then, I shall check for outliers using boxplots and try to replace or remove them. Thirdly, I shall get some visualizations to get an idea of the variables in hand.

➢ Feature Engineering

Feature engineering is the process of transforming or creating new features from existing data variables (features) to improve the performance and effectiveness of machine learning models. It involves selecting, extracting, or manipulating features to capture relevant information, enhance predictive power, and increase the model's ability to generalize and make accurate predictions.

The goal of feature engineering is to represent the data in a format that maximizes the model's ability to learn patterns and relationships within the data. It requires domain knowledge, creativity, and an understanding of the underlying data and problem domain. Some common techniques and tasks involved in feature engineering include:

- Feature Selection: Choosing the most relevant and informative features from the available dataset. This helps reduce dimensionality, eliminate noisy or redundant features, and improve model efficiency.
- Feature Extraction: Creating new features by applying mathematical transformations, statistical measures, or domain-specific knowledge to the existing dataset. This can involve techniques like Principal Component Analysis (PCA), text vectorization, or Fourier transforms.
- Handling Missing Values: Addressing missing values in the dataset by imputing or creating new features that capture information about the missingness.
- Binning or Discretization: Grouping continuous or numerical features into categories or bins based on specific criteria. This can help capture non-linear relationships or simplify complex patterns.
- One-Hot Encoding: Converting categorical variables into binary indicator variables, allowing them to be effectively utilized in machine learning algorithms.
- Feature Scaling: Normalizing or scaling features to ensure they are on a similar scale. This prevents certain features from dominating the learning process and helps models converge faster.
- Feature Interactions: Creating new features by combining or interacting existing features. This can capture complex relationships and interactions between variables.

The effectiveness of feature engineering can significantly impact the performance of machine learning models. Well-engineered features can lead to improved accuracy, reduced overfitting, faster training, and more interpretable models.

The first feature breakdown I've performed is to get Journey_date, Journey_month, Journey_year from Date_of_journey.

Similarly we performed the breakdown for Dep_Time_hour, Dep_Time_minute and Arrival_Time_hour, Arrival_Time_minute from Dep_time and Arrival_time.

➢ Missing Value

Missing values in a dataset refer to the absence or lack of data for certain variables or observations. They occur when no data is recorded or available for a specific attribute or instance within the dataset. Missing values can occur for various reasons, such as data collection errors, data corruption, non-response in surveys, or technical issues.

- Missing values can be represented in different forms, depending on the dataset and the software used. Some common representations of missing values include:
- Blank or empty cells: The missing value is indicated by leaving the corresponding cell empty or blank.
- Null or NaN (Not a Number): The missing value is denoted by a specific placeholder value, such as "null" or "NaN," which signifies the absence of data.
- Placeholder codes: Some datasets may use specific codes or symbols to represent missing values, such as "9999" or "-999."

Handling Missing values in a dataset:

Handling missing values is an essential step in data preprocessing and analysis. Ignoring or mishandling missing values can lead to biased or erroneous results. Several strategies can be employed to address missing values, including:

- **Removal:** Removing observations or variables with missing values. This approach is suitable when the missingness is random and does not significantly affect the analysis.
- **Imputation:** Replacing missing values with estimated or imputed values based on statistical techniques. Common imputation methods include mean imputation, regression imputation, or multiple imputation.
- **Indicator variable:** Creating a binary indicator variable to explicitly identify the presence or absence of missing values for a specific variable.
- **Modeling-based imputation:** Using machine learning algorithms or statistical models to predict missing values based on the available data.

The choice of how to handle missing values depends on the nature of the data, the reason for missingness, the amount of missing data, and the specific analysis goals.

➢ Basic EDA of the Categorical Variables

Categorical variables are variables that represent qualitative characteristics or attributes rather than numerical values. They divide data into distinct groups or categories based on specific characteristics or labels. Categorical variables can take on a limited number of possible values, often represented by text or labels rather than numerical values.

There are two main types of categorical variables:

Nominal variables: These variables have categories with no inherent order or ranking. Examples include Airline (categories: 'SpiceJet', 'Indigo', 'GO FIRST', 'Air India',other), Class(categories: 'Economy', 'Business')

Ordinal variables: These variables have categories with a specific order or ranking. The categories represent different levels or degrees of a characteristic but do not necessarily have equal intervals between them. Examples include Total stops (categories: 'non-stop' '1-stop' '2+-stop')

➢ Outliers

An outlier in statistics refers to a data point that significantly deviates from the rest of the observations. It can occur due to various reasons such as measurement variability, the presence of new or unusual data, or experimental errors, which are sometimes excluded from the dataset. Outliers can provide valuable insights or indicate interesting patterns in the data, but they can also pose challenges when conducting statistical analyses. Outliers can occur randomly in any distribution, but they can signify unique behaviors or structures within the dataset, measurement inaccuracies, or the presence of heavy-tailed distributions in the population. In the case of measurement errors, it is often desirable to remove them or utilize statistical methods that are resistant to the influence of outliers. On the other hand, if the data follows a heavy-tailed distribution, outliers indicate significant skewness, and one should exercise caution when applying tools or assumptions based on a normal distribution. Outliers can frequently emerge when there is a mixture of two distributions, representing distinct sub-populations or differentiating between "correct trials" and "measurement errors." This scenario can be modeled using a mixture model.

• **Outliers detection and removal:**

- ❑ **Trimming-** This technique removes any outlier values from our analysis, resulting in a narrower dataset when there are more outliers present. Its primary benefit is its speediness.
- ❑ **Capping-** In this approach, we set a limit or threshold for outlier values, considering all values above or below that threshold as outliers. The count of outliers in the dataset indicates the number of values that exceed this limit. For instance, if we are analyzing the income feature and notice that individuals with income above a certain level exhibit similar behavior to those with lower income, we can cap the income value at a level that preserves this pattern and handle the outliers accordingly. Another method is treating outliers as missing values. We consider outliers as observations that are missing and apply the same imputation techniques used for handling missing values.
- ❑ **Discretization-** also referred to as binning, is a technique where we create groups or bins and assign outliers to a specific bin, thereby forcing them to exhibit similar behavior as the other data points within that bin.

There are several techniques commonly used to remove outliers from data. Here are a few of them:

- ❑ **Z-Score or Standard Deviation Method:** This method involves calculating the z score for each data point and removing those that fall outside a certain threshold (typically a z-score of 3 or -3).
- ❑ **Percentile Method:** In this approach, the data points are sorted in ascending order, and the values at specified percentiles (such as the 1st percentile and 99th percentile) are considered as the threshold. Any values below the lower percentile or above the upper percentile are treated as outliers and removed.
- ❑ **Box Plot Method:** A box plot visually represents the distribution of data and identifies outliers based on the position of data points outside the whiskers (the lines extending from the box). Outliers can be identified and removed based on certain criteria, such as being above or below a specific multiple of the interquartile range.
- ❑ **Robust Statistical Methods:** These methods, such as Median Absolute Deviation (MAD) or Huber's M-estimator, are less sensitive to outliers and can be used to estimate robust measures of central tendency and dispersion, which can then be used to identify and remove outliers.
- ❑ **Winsorization:** It is a technique used to handle outliers in data by capping or limiting extreme values. Instead of removing outliers completely, winzorization replaces them with less extreme values. The process involves setting a threshold or limit for the outlier values. Any data points that exceed this threshold are replaced with the nearest non-outlier values.

The replacement values are determined by either the maximum or minimum non-outlier values.

- ❑ **IQR method:** The IQR (Interquartile Range) method is a technique used to identify and remove outliers from a dataset based on the spread of the data. It involves using the quartiles, which divide the data into four equal parts, to calculate the range within which most of the data points lie.

  Here's how the IQR method works:
  Calculate the first quartile (Q1) and third quartile (Q3) of the dataset.
  The first quartile represents the 25th percentile, while the third quartile represents the 75th percentile.
  Compute the interquartile range (IQR) by subtracting Q1 from Q3: IQR = Q3 - Q1.
  Define the lower bound as Q1 - 1.5 * IQR and the upper bound as Q3 + 1.5 * IQR.

Identify any values in the dataset that fall below the lower bound or above the upper bound. These values are considered outliers. Remove the identified outliers from the dataset.

The IQR method assumes that the data follows a roughly symmetric distribution. It considers values outside the range defined by the lower and upper bounds as potential outliers. The advantage of using the IQR method is its robustness to extreme values while preserving a larger portion of the dataset compared to other methods.

- ❑ **Domain Knowledge:** Sometimes, expert knowledge about the domain can help in identifying outliers. By understanding the context and characteristics of the data, outliers that are genuine anomalies or errors can be identified and removed. It's important to note that the choice of which technique to use depends on the nature of the data, the specific problem at hand, and the underlying assumptions and requirements of the analysis.

➢ Multicollinearity

Multicollinearity refers to a high correlation or linear relationship between two or more predictor variables (also known as independent variables) in a statistical model. It occurs when the predictor variables are not independent of each other and can make it challenging to interpret the individual effects of each variable on the dependent variable accurately. Multicollinearity arises when there is a strong correlation between two or more predictor variables. High correlation means that the variables are providing similar or redundant information, making it difficult to determine their unique contribution to the model.

In regression analysis, multicollinearity can have several effects. It can lead to unstable or unreliable parameter estimates, high variability in the estimated coefficients, inflated standard errors, and decreased predictive accuracy of the model. Additionally, multicollinearity can make it challenging to interpret the importance and significance of individual predictor variables.

Detecting multicollinearity: Multicollinearity can be assessed using various statistical techniques, such as correlation matrices, variance inflation factor (VIF), tolerance, or eigenvalues. These methods help identify the presence and severity of multicollinearity in the dataset.

Dealing with multicollinearity: To address multicollinearity, several strategies can be employed:

- Removing or combining correlated variables: If two variables are highly correlated, one of them can be removed from the model or combined into a single variable to eliminate redundancy.

- Feature selection: Selecting a subset of variables based on their relevance and importance can help mitigate multicollinearity.
- Regularization techniques: Applying regularization methods like Ridge regression or Lasso regression can handle multicollinearity by penalizing the coefficients of highly correlated variables.
- Collecting more data: Increasing the sample size can sometimes alleviate multicollinearity issues by providing a wider range of values and reducing the correlation between variables.

It is important to address multicollinearity to ensure accurate and reliable results in regression analysis. By understanding and mitigating multicollinearity, one can improve the interpretability and stability of the statistical model and make sound inferences about the relationships between predictor variables and the dependent variable.

Variance Inflation Factor (VIF):

The Variance Inflation Factor (VIF) is a commonly used diagnostic approach to detect multicollinearity. In our case, we employed the VIF approach to identify and address multicollinearity issues. The VIF measures the extent to which the variance of a regression coefficient is inflated due to multicollinearity.

The VIF for the j-th explanatory variable is calculated as:

$$VIF_j = \frac{1}{1 - R_j^2}$$

where R_j^2 represents the coefficient of determination when regressing the j-th variable on the remaining (k - 1) variables, excluding the j-th variable itself.

In practice, a VIF value greater than 5 is often considered an indication of poorly estimated regression coefficients due to multicollinearity. To handle multicollinearity, we implemented an iterative algorithm that repeatedly drops the variable with the highest VIF and checks the VIF again. This process is repeated until the VIF of all variables is less than 5, indicating that multicollinearity has been addressed.

➢ Data Preprocessing

o Train Test Split:

In machine learning, it is a normal practice to divide a given dataset into two subsets for further model training:

- the training set
- the test set

The training set is used to train the machine learning model by fitting it to the available data and learning from the outcomes. On the other hand, the test set consists of new instances that were not used during training. The purpose of the test set is to assess how well the trained model performs on unseen data and to evaluate its generalization capabilities.

The split between the training set and the test set is typically achieved by specifying a ratio, indicating the desired proportion for each subset. This split can be implemented in the code by randomly assigning instances to either the training set or the test set according to the specified ratio.

By evaluating the model's performance on the test set, we can determine if the algorithm is only effective on the training set or if it also performs well on new, unseen data. This evaluation helps in assessing the model's ability to generalize and make accurate predictions outside the context of the training data.

It is crucial to have a reliable evaluation of the model's performance on the test set as it provides insights into how the model is likely to perform in real-world scenarios. This evaluation helps in making informed decisions about deploying the model and assessing its usefulness.

To summarize, the training set is used to train the machine learning model on existing observations, while the test set is used to evaluate the model's performance on new, unseen observations. The split between the two sets can be specified by a ratio, allowing for a comprehensive assessment of the model's capabilities and its ability to generalize to new data.

➢ Feature Scaling

The process of feature scaling entails scaling each of our variables to ensure that they all accept values from the same scale and we do this to avoid having one trait predominate over another, which would lead to machine learning models ignoring it. summary, feature scaling is a preprocessing step in machine learning that aims to bring all variables to a common scale. It prevents any particular feature from dominating the learning algorithm and ensures fair consideration of all features. This process is crucial for maintaining the integrity and effectiveness of machine learning models.

Do we have to apply feature scaling before or after splitting the dataset into the training set and test set?

As feature scaling will take the mean and standard deviation of the entire set, including the "test set," which is not expected and will be referred to as information leaking on the test set, feature scaling should be carried out after splitting. This is because the test set should have fresh data.

➢ Model Training and Testing

o Linear Regression:

When describing the relationship between a scalar answer and one or more explanatory factors (also known as dependent and independent variables) in statistics, linear regression is a linear approach. Simple linear regression is used when there is only one explanatory variable, and multiple linear regression is used when there are numerous variables.As opposed to multivariate linear regression, which predicts numerous correlated dependent variables as opposed to a single scalar variable, this phrase is more specific.

In linear regression, linear predictor functions are used to model relationships, with the model's unknown parameters being estimated from the data. These models are referred to as linear models. The conditional mean of the response is typically considered to be an affine function of the values of the explanatory variables (or predictors); the conditional median or another quantile is occasionally employed. In common with all other types of regression analysis, linear regression concentrates on the conditional probability distribution of the response given the values of the predictors rather than the joint probability distribution of all these variables, which is the purview of multivariate analysis.

○ What is OLS?

The statistical technique known as OLS, or ordinary least squares, is used to estimate a linear regression model's parameter. Finding the line or hyperplane that minimizes the sum of the squared differences between the observed data points and the model's projected values is the objective of linear regression.

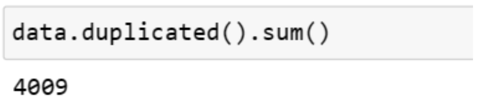
The OLS method works by minimizing the sum of the squared residuals, which are the differences between the actual values and the predicted values of the dependent variable. The estimated parameters are chosen to minimize this sum, hence the name "least squares." The OLS method assumes that the errors or residuals follow a normal distribution with mean zero and constant variance. In summary, OLS is a statistical method used for estimating the parameters of a linear regression model by minimizing the sum of squared differences between observed and predicted values. It provides a way to find the best-fitting line or hyperplane that describes the relationship between variables in a linear regression framework.

Assumptions of Regression Model:

- Regression model is linear in parameters, $Y = \beta_0 + \beta_1 X + e_i$.
- The X values are fixed in repeated sampling i.e., X is non-Stochastic.
- Zero mean of error
- Homoscedasticity or equal variance of e_i.
- Non-Auto correlation between errors.
- Zero Covariance between e_i and x_i.
- No of observations n is greater than the number of parameters to be estimated p, n>p.
- All x values in a given sample must not be same. Thus V(X) must be a finite positive number.
- The regression model should be correctly specified.
- There should be no multicollinearity between explanatory variables.

To find the accuracy of the model we obtained the RMSE and R2 Score, and also drew a scatter plot between actual and predicted values of the target variable.

In first step, I want to do some exploration on the data. Before that, I need to see whether there is any duplicate values in the dataset.

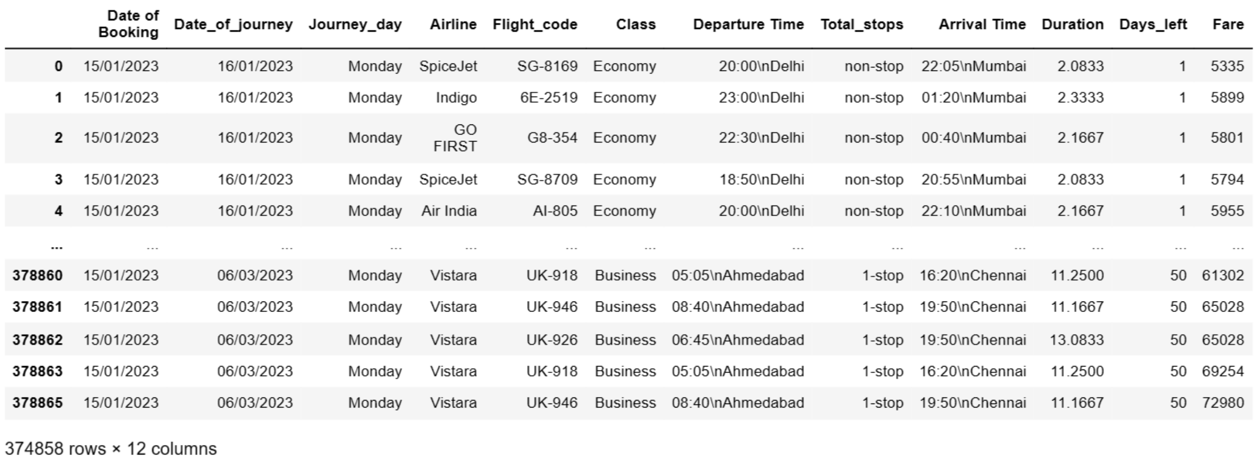


```
data.duplicated().sum()

4009
```

So there is 4009 duplicate values and we need to remove them from the dataset.

So the final dataset becomes:

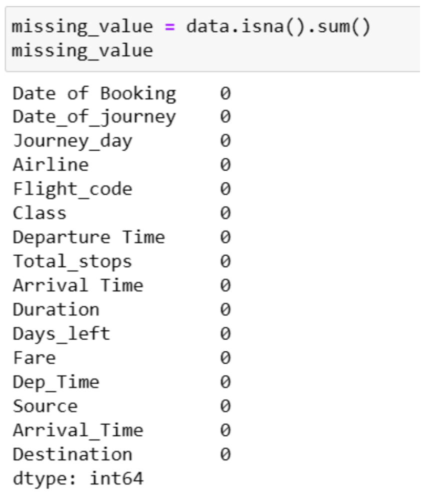


	Date of Booking	Date_of_journey	Journey_day	Airline	Flight_code	Class	Departure Time	Total_stops	Arrival Time	Duration	Days_left	Fare
0	15/01/2023	16/01/2023	Monday	SpiceJet	SG-8169	Economy	20:00\nDelhi	non-stop	22:05\nMumbai	2.0833	1	5335
1	15/01/2023	16/01/2023	Monday	Indigo	6E-2519	Economy	23:00\nDelhi	non-stop	01:20\nMumbai	2.3333	1	5899
2	15/01/2023	16/01/2023	Monday	GO FIRST	G8-354	Economy	22:30\nDelhi	non-stop	00:40\nMumbai	2.1667	1	5801
3	15/01/2023	16/01/2023	Monday	SpiceJet	SG-8709	Economy	18:50\nDelhi	non-stop	20:55\nMumbai	2.0833	1	5794
4	15/01/2023	16/01/2023	Monday	Air India	AI-805	Economy	20:00\nDelhi	non-stop	22:10\nMumbai	2.1667	1	5955
...	...	...	...	...	...	...	...	...	...	...	...	...
378860	15/01/2023	06/03/2023	Monday	Vistara	UK-918	Business	05:05\nAhmedabad	1-stop	16:20\nChennai	11.2500	50	61302
378861	15/01/2023	06/03/2023	Monday	Vistara	UK-946	Business	08:40\nAhmedabad	1-stop	19:50\nChennai	11.1667	50	65028
378862	15/01/2023	06/03/2023	Monday	Vistara	UK-926	Business	06:45\nAhmedabad	1-stop	19:50\nChennai	13.0833	50	65028
378863	15/01/2023	06/03/2023	Monday	Vistara	UK-918	Business	05:05\nAhmedabad	1-stop	16:20\nChennai	11.2500	50	69254
378865	15/01/2023	06/03/2023	Monday	Vistara	UK-946	Business	08:40\nAhmedabad	1-stop	19:50\nChennai	11.1667	50	72980

374858 rows × 12 columns

- Now I shall check for null values and try to replace or remove them.
- Then, I shall check for outliers using boxplots and try to replace or remove them.
- Thirdly, I shall get some visualizations to get an idea of the variables in hand.

Is there any missing values in our data?

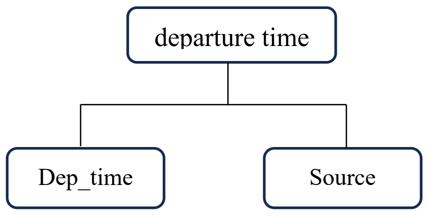


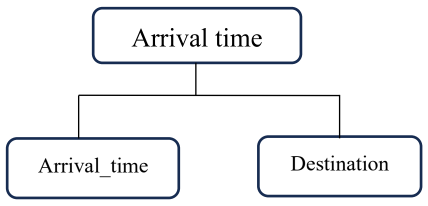
```
missing_value = data.isna().sum()
missing_value

Date of Booking    0
Date_of_journey    0
Journey_day        0
Airline            0
Flight_code        0
Class              0
Departure Time     0
Total_stops        0
Arrival Time       0
Duration           0
Days_left          0
Fare               0
Dep_Time           0
Source             0
Arrival_Time       0
Destination        0
dtype: int64
```

So, we can see there is no missing values in the dataset so we do not need to impute missing values.

Now we need to split departure time and source from departure time and same for arrival time.





And then we should split the Hour min form Dep_time and Arrival_time and also the journey date,month and year from Date_of_journey.

We can form some questions for this EDA part as follows:

1) when will most of the flights take-off?

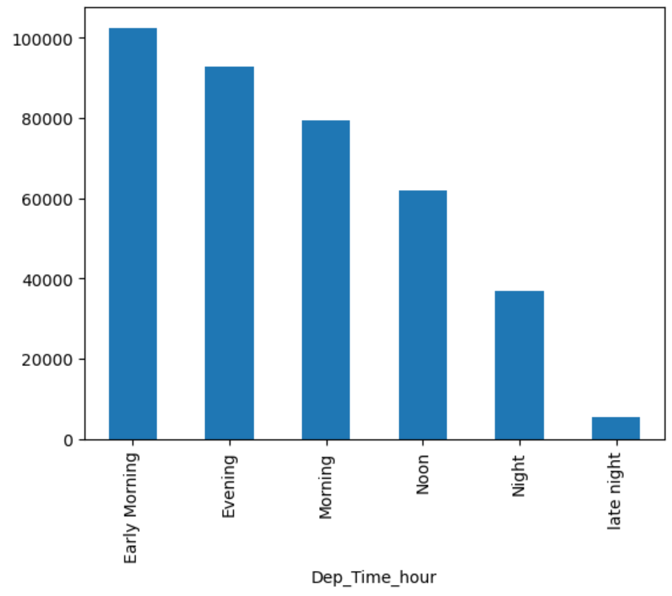


Fig1: Visualization of Flight Departure Times

From the above figure we can see that most flights depart in the early morning hours, with fewer departures occurring late at night.

2) when will most of the flights take-off?

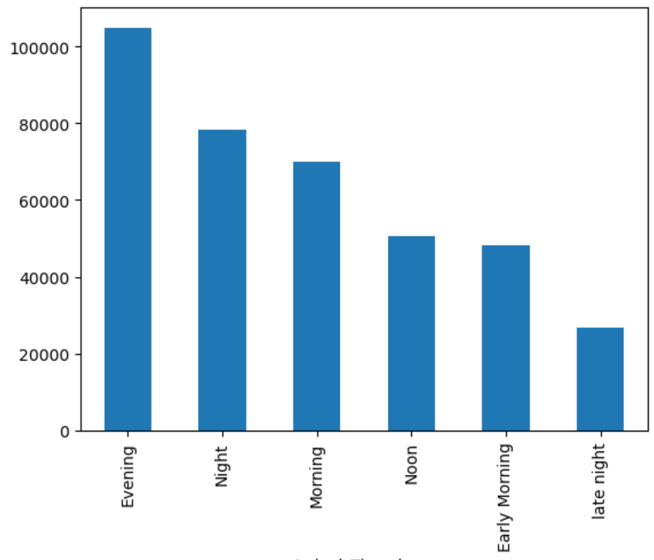


Fig2: Visualization of Flight Arrival Times

From the above figure we can see that most flights arrived in the early morning hours.

3) What are number of flights operated by each airline?

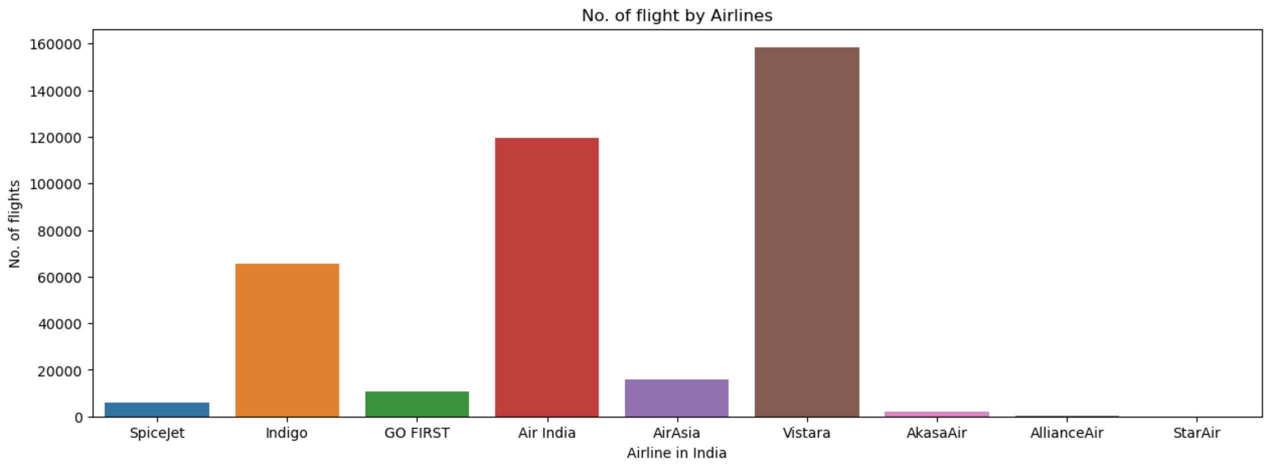


Fig3: Visualization of Number of Flights Operated by Different Airlines

From the above figure, we can see 'Vistara' has maximum no. of fights followed by 'Air India', 'Indiogo' while 'Spice Jet' , 'AkasaAir' ,'AllianceAir','StarAir' has least no. of flights.

4) *What is price range according to class and days left of travel?*

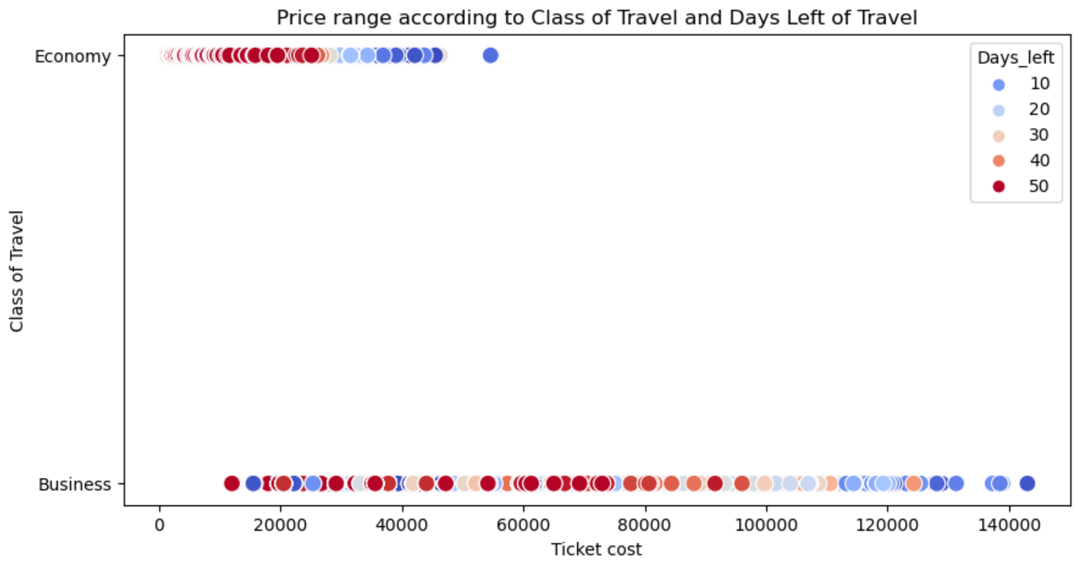


Fig4: Visualization of Price Range According to Class and Days Left Before Travel

From the above figure, it is evident that 'Economy' class tickets typically range in price from 1300 to 50000, whereas 'Business' class tickets typically range from 12000 to 150000. Additionally, it appears that as the days remaining until the departure of the flight decrease, the flight prices tend to increase.

5) *How do ticket prices vary across different airlines and class of travel?*

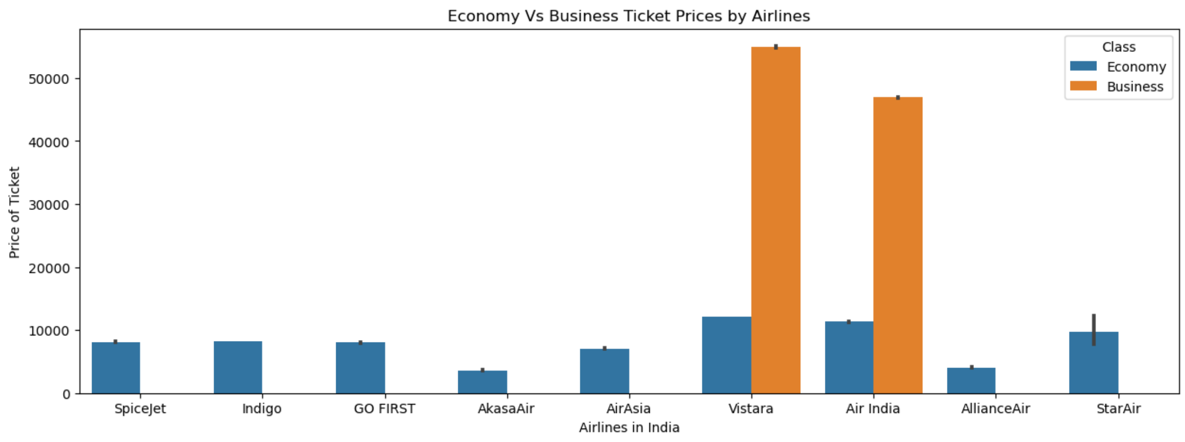


Fig5: Visualization of Ticket Prices Across Different Airlines and Classes of Travel

From the above figure, we can conclude that 'AkasaAir','AllianceAir' offers the cheapest 'Economy' class tickets following 'Air Asia', 'Indigo, 'Go First', 'Spice Jet','StarAir'. Meanwhile 'Air India' and 'Vistara' are priced much higher than other 7 airlines.'Business' class tickets for 'Vistara' cost much higher than 'Air India' which can be due to better service, quality of seats available on 'Vistara' as compared to 'Air India'.

6) What is the availability of Tickets according to class of travel?

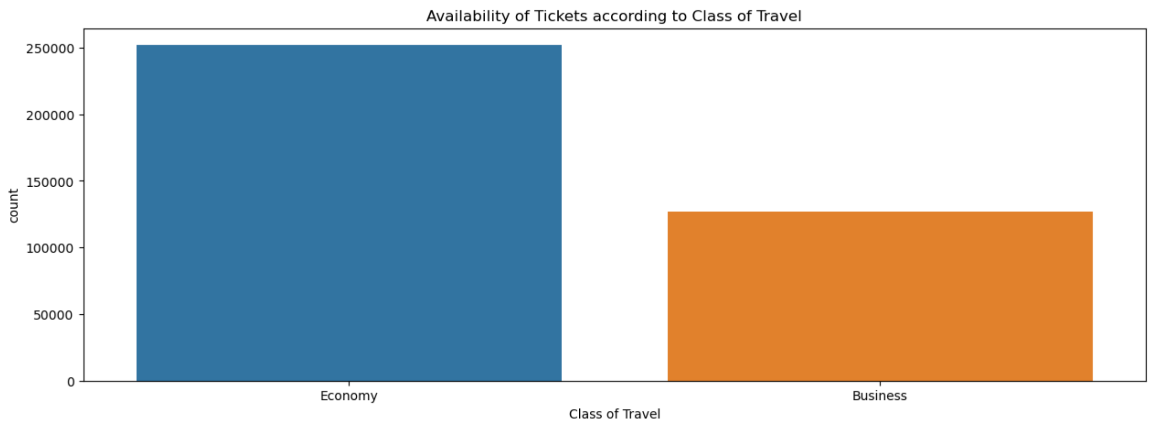


Fig6: Visualization of Ticket Availability According to Class of Travel

From the above figure, we can see that availabilty of 'Economy' tickets is almost twice than availibitly of 'Business' class tickets which is explained by the fact that only 2 airlines - 'Air India', 'Vistara' offer 'Business' class tickets while all airlines offer 'Economy' class tickets.

7) What is price of ticket for different airlines based on duration of flight?

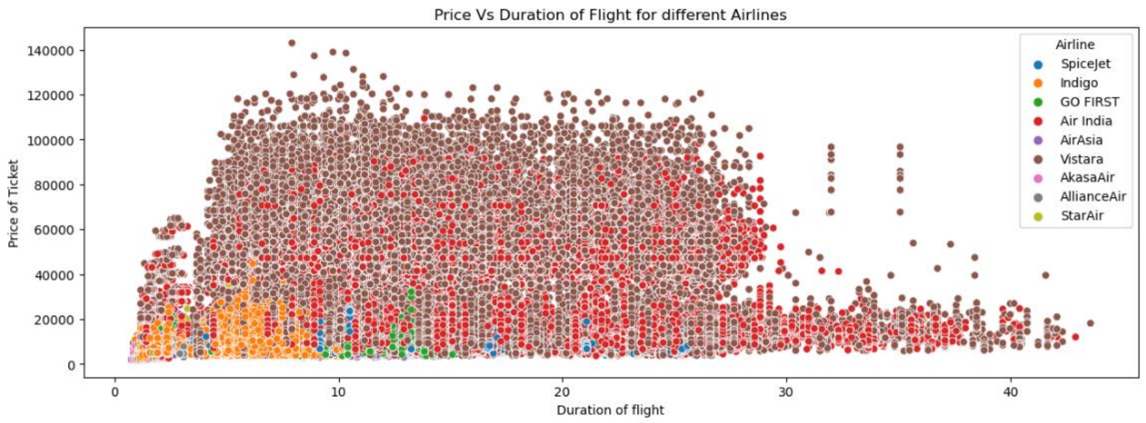


Fig7: Visualization of Ticket Prices for Different Airlines Based on Flight Duration

From the above figure, we see the distribution of ticket price varying with duration of flight. More no. of brown and red points in the figure is explained by the fact that 'Vistara' and 'Air India' have maximum no. of flights.

8) How does price of tickets vary based on no. of stops and airline?

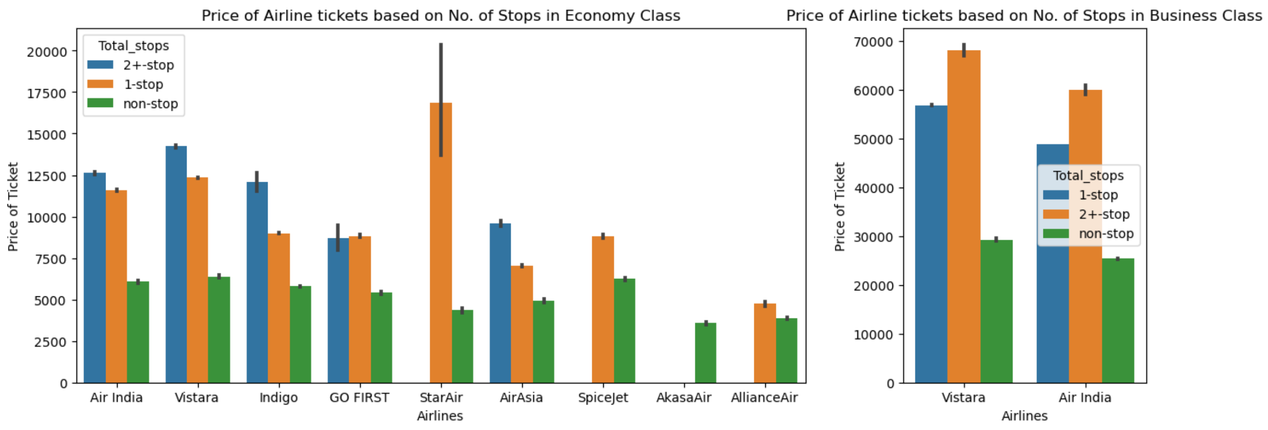


Fig8: Visualization of Ticket Prices Based on Number of Stops and Airline

From the above figure, we can conclude that Non-Stop flights are generally the cheapest while One-Stop flights are more expensive and 2+ stop flights are most expensive which can be explained on basis that as one undertakes more flights to fly to destination, it costs more. 'Allianceair' seems to be an exception in this case which shows little variation in prices between its Non-Stop and One Stop flights. For 'Gofirst' there is little variation in prices between its One Stop and 2+ stop flights.

9) How do airline ticket prices vary depending on when you buy them?

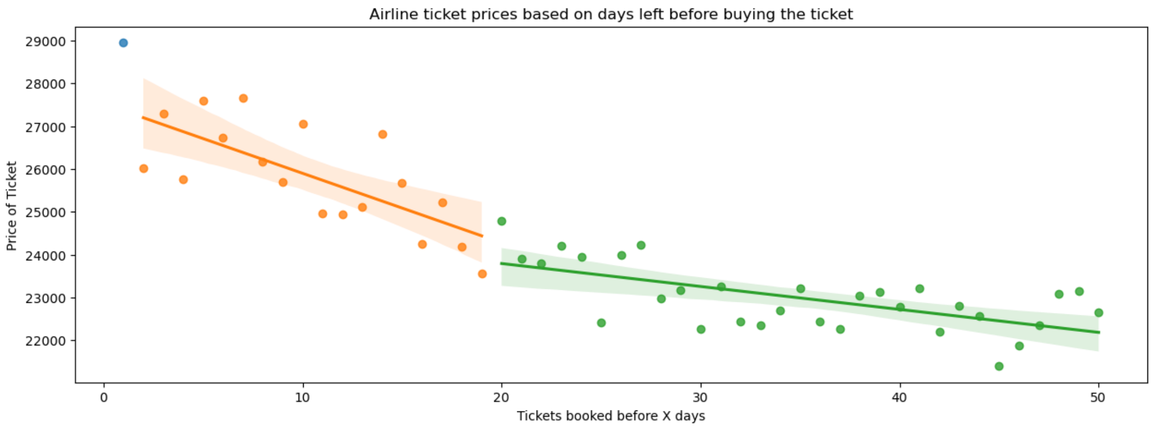


Fig9: Visualization of Ticket Prices Variation Depending on Time of Purchase

From the above figure, we can conclude that ticket price rise till 20 days from the date of flight, then rise sharply till the last day, while suddenly increase just 1 day before the date of flight. This can be explained by the fact that people usually buy flight tickets before 6-7 weeks of flight which generates more profits for airlines.

10) How does price of ticket vary depending on duration of flight?

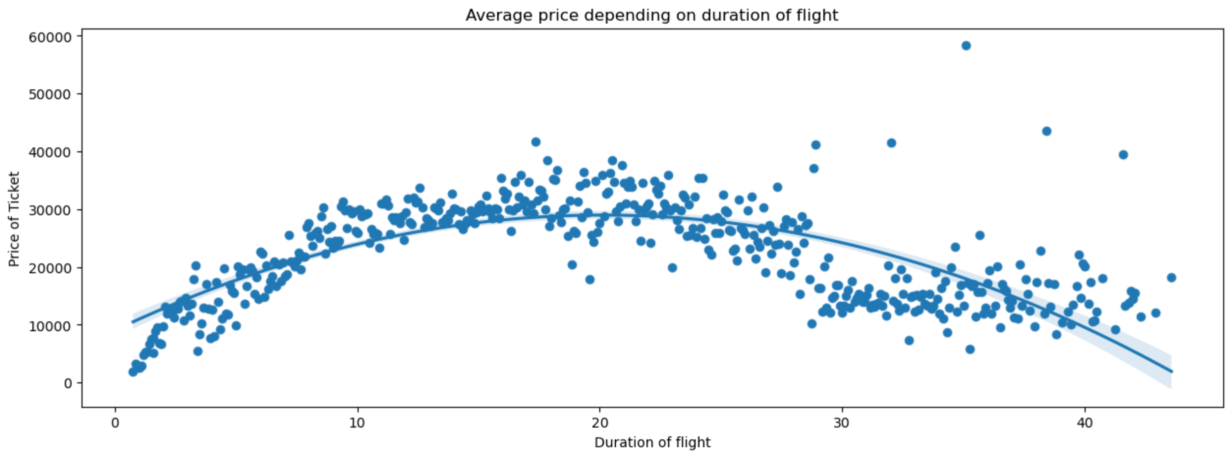


Fig10: Visualization of Ticket Price Variation Depending on Duration of Flight

From the above figure, we can see that the relationship is not linear but can be approximated by second degree curve. We can see linear growth in prices as duration of flight increases till 20 and then lowering again.

Some outliers may be affecting the curve.

Are there any outliers in our data?

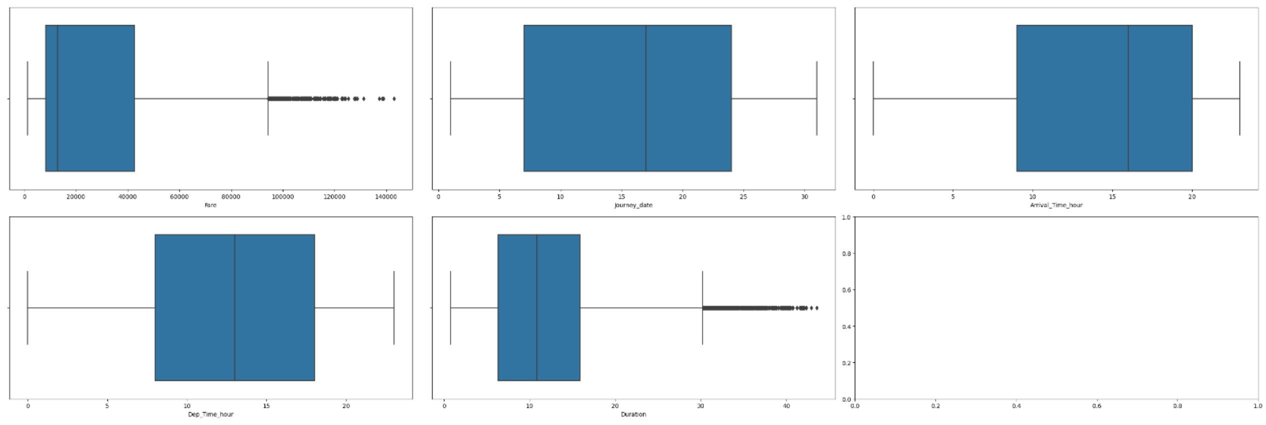


Fig11: Boxplot Showing Outliers in the Data

From the box plot we can see that there are several points as outliers in Fare and Duration. But from the data we have seen the price is high for those flights of Business class, and those which have longer duration such as the international flights. So, we do not remove these data points as they carry out meaningful information.

Now we should observe the distribution of fare.

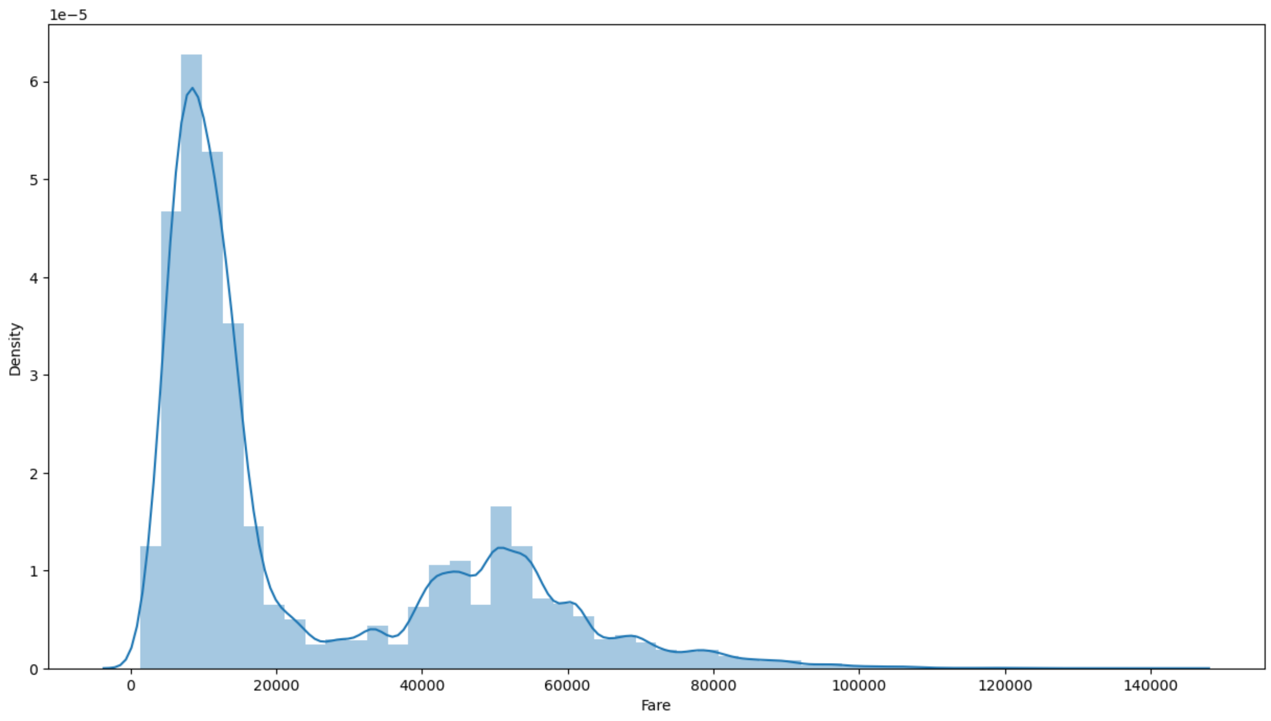


Fig12: Distribution for Fare

From the above graph, we can see that the distribution of Fare is positively skewed. So we take log transformation and it will become as follows:

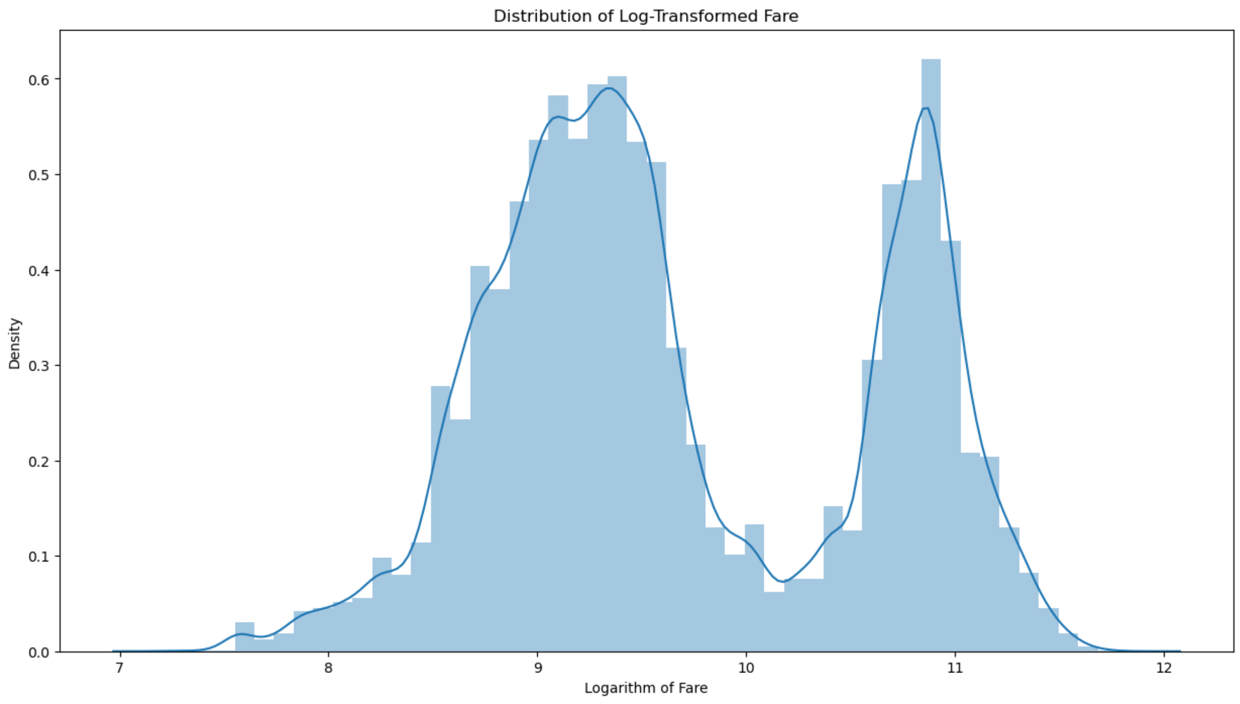
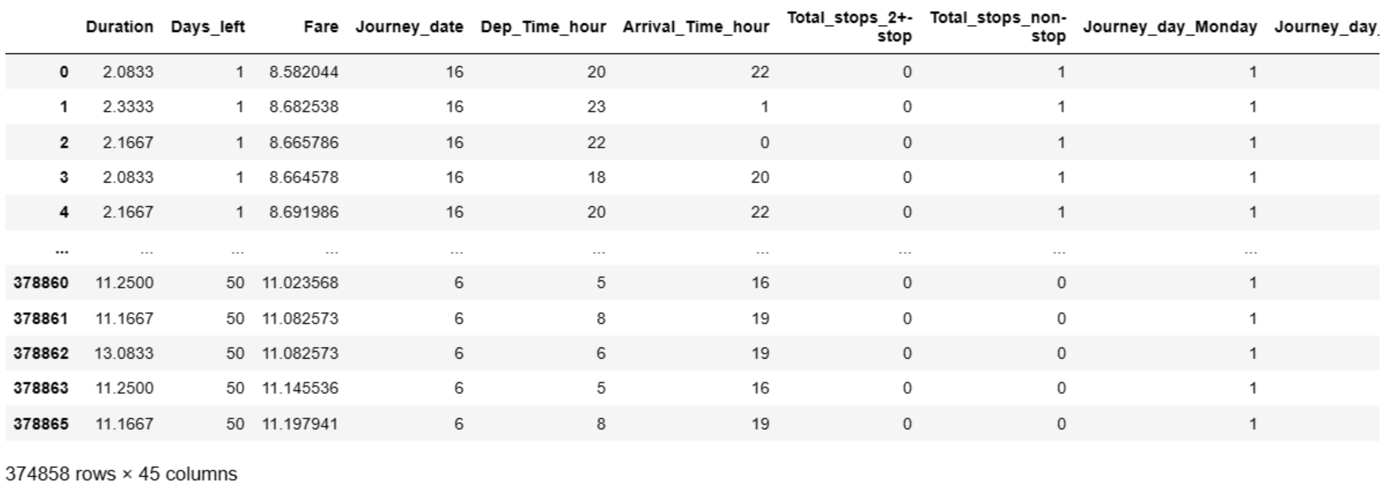


Fig12: Distribution for Logarithm of Fare

Now we can drop the unnecessary columns such as 'Date of Booking', 'Date_of_journey', 'Dep_Time_minute', 'Arrival_Time_minute', 'Journey_year', 'Journey_month', and 'Flight_code'.

I want to observe the categorical variables, so I introduced them as dummy variables for ‘Total_stops’,’Journey_day’,’Airline’,’Class’,’Source’,’Destination’,’depart’,’Arrival’.

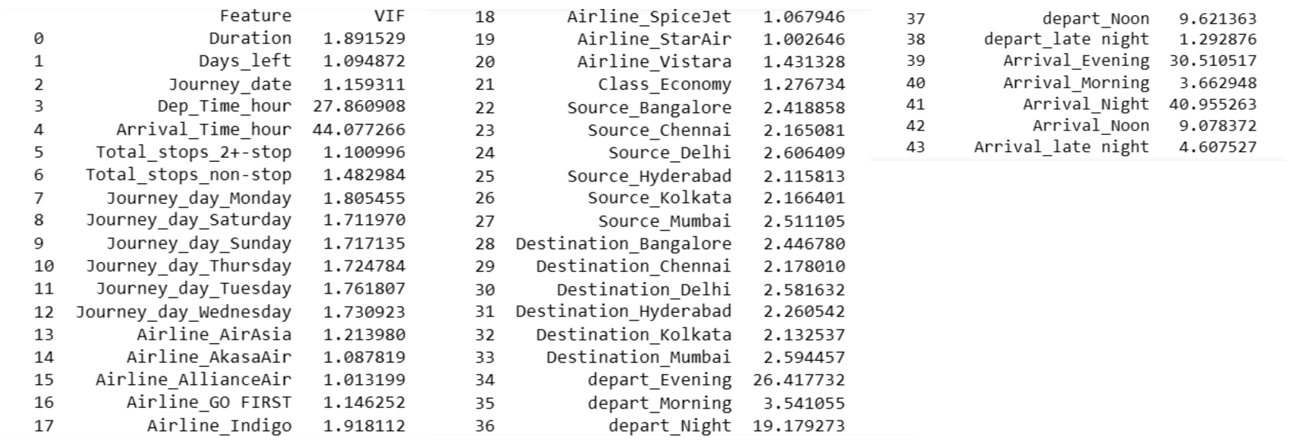
So our final dataset is as follows:



	Duration	Days_left	Fare	Journey_date	Dep_Time_hour	Arrival_Time_hour	Total_stops_2+-stop	Total_stops_non-stop	Journey_day_Monday	Journey_day
0	2.0833	1	8.582044	16	20	22	0	1	1	
1	2.3333	1	8.682538	16	23	1	0	1	1	
2	2.1667	1	8.665786	16	22	0	0	1	1	
3	2.0833	1	8.664578	16	18	20	0	1	1	
4	2.1667	1	8.691986	16	20	22	0	1	1	
...	...	...	...	...	...	...	...	...	...	
378860	11.2500	50	11.023568	6	5	16	0	0	1	
378861	11.1667	50	11.082573	6	8	19	0	0	1	
378862	13.0833	50	11.082573	6	6	19	0	0	1	
378863	11.2500	50	11.145536	6	5	16	0	0	1	
378865	11.1667	50	11.197941	6	8	19	0	0	1	

374858 rows × 45 columns

Now we should check for multicollinearity:



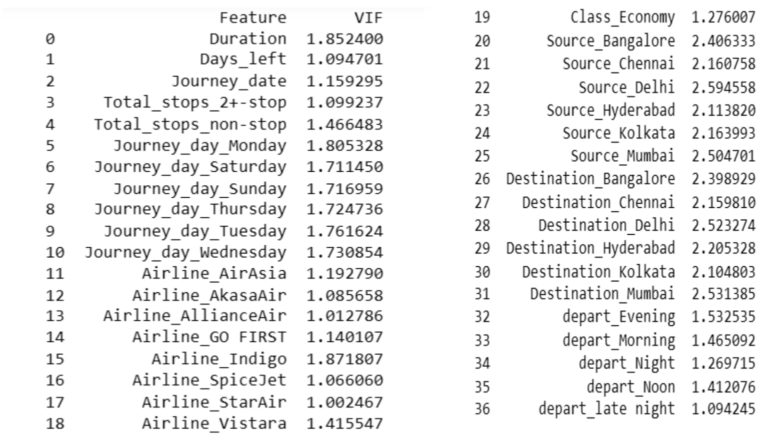
```
                Feature        VIF
0              Duration   1.891529
1             Days_left   1.094872
2          Journey_date   1.159311
3         Dep_Time_hour  27.860908
4     Arrival_Time_hour  44.077266
5    Total_stops_2+-stop   1.100996
6   Total_stops_non-stop   1.482984
7     Journey_day_Monday   1.805455
8   Journey_day_Saturday   1.711970
9     Journey_day_Sunday   1.717135
10  Journey_day_Thursday   1.724784
11   Journey_day_Tuesday   1.761807
12 Journey_day_Wednesday   1.730923
13       Airline_AirAsia   1.213980
14      Airline_AkasaAir   1.087819
15   Airline_AllianceAir   1.013199
16      Airline_GO FIRST   1.146252
17        Airline_Indigo   1.918112
```

```
18       Airline_SpiceJet   1.067946
19        Airline_StarAir   1.002646
20        Airline_Vistara   1.431328
21          Class_Economy   1.276734
22       Source_Bangalore   2.418858
23         Source_Chennai   2.165081
24           Source_Delhi   2.606409
25       Source_Hyderabad   2.115813
26         Source_Kolkata   2.166401
27          Source_Mumbai   2.511105
28  Destination_Bangalore   2.446780
29    Destination_Chennai   2.178010
30      Destination_Delhi   2.581632
31  Destination_Hyderabad   2.260542
32    Destination_Kolkata   2.132537
33     Destination_Mumbai   2.594457
34         depart_Evening  26.417732
35         depart_Morning   3.541055
36           depart_Night  19.179273
```

```
37            depart_Noon   9.621363
38      depart_late night   1.292876
39        Arrival_Evening  30.510517
40        Arrival_Morning   3.662948
41          Arrival_Night  40.955263
42           Arrival_Noon   9.078372
43     Arrival_late night   4.607527
```

We can see that the VIF scores of the variables are less than 5 in most of the cases, but some are greater than 5. So we can say that there is high multicollinearity in the data set. Now we simply drop the columns with high vif. So basically we are not including the departure time in hour and arrival time in hour and also the arrival time slots because arrival time is dependent on departure time slots.

Now we again check the VIF:



```
                Feature       VIF
0              Duration  1.852400
1             Days_left  1.094701
2          Journey_date  1.159295
3    Total_stops_2+-stop  1.099237
4   Total_stops_non-stop  1.466483
5     Journey_day_Monday  1.805328
6   Journey_day_Saturday  1.711450
7     Journey_day_Sunday  1.716959
8   Journey_day_Thursday  1.724736
9    Journey_day_Tuesday  1.761624
10 Journey_day_Wednesday  1.730854
11       Airline_AirAsia  1.192790
12      Airline_AkasaAir  1.085658
13   Airline_AllianceAir  1.012786
14      Airline_GO FIRST  1.140107
15        Airline_Indigo  1.871807
16      Airline_SpiceJet  1.066060
17       Airline_StarAir  1.002467
18       Airline_Vistara  1.415547
```

```
19          Class_Economy  1.276007
20       Source_Bangalore  2.406333
21         Source_Chennai  2.160758
22           Source_Delhi  2.594558
23       Source_Hyderabad  2.113820
24         Source_Kolkata  2.163993
25          Source_Mumbai  2.504701
26  Destination_Bangalore  2.398929
27    Destination_Chennai  2.159810
28      Destination_Delhi  2.523274
29  Destination_Hyderabad  2.205328
30    Destination_Kolkata  2.104803
31     Destination_Mumbai  2.531385
32         depart_Evening  1.532535
33         depart_Morning  1.465092
34           depart_Night  1.269715
35            depart_Noon  1.412076
36      depart_late night  1.094245
```

Finally we get scores of the variables as less than 5, so we can say that very little multicollinearity is present in the data.

Now we fit the Linear Regression Model in our dataset and find the R^2 values, Mean Absolute Error, Mean Squared Error for training and testing dataset and we get the result as follows:

```
MAE for training set: 0.2229847635553437
MSE for training set: 0.0837307957972198
RMSE for training set: 0.28936274085863195
R2 score for training set: 0.8962554829467508
```

```
MAE for testing set: 0.22332471664004097
MSE for testing set: 0.08367868477782886
RMSE for testing set: 0.28927268239124976
R2 score for testing set: 0.8961777035649338
```

Now let us see the residual plot:

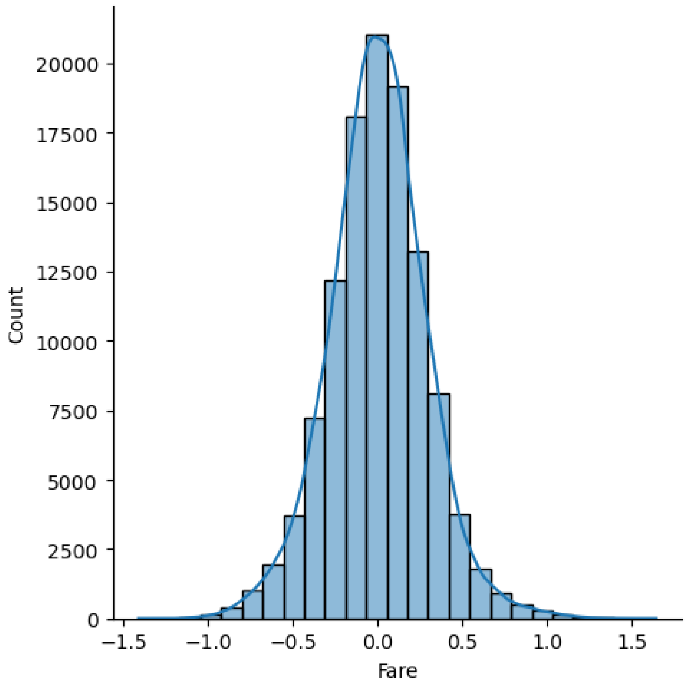
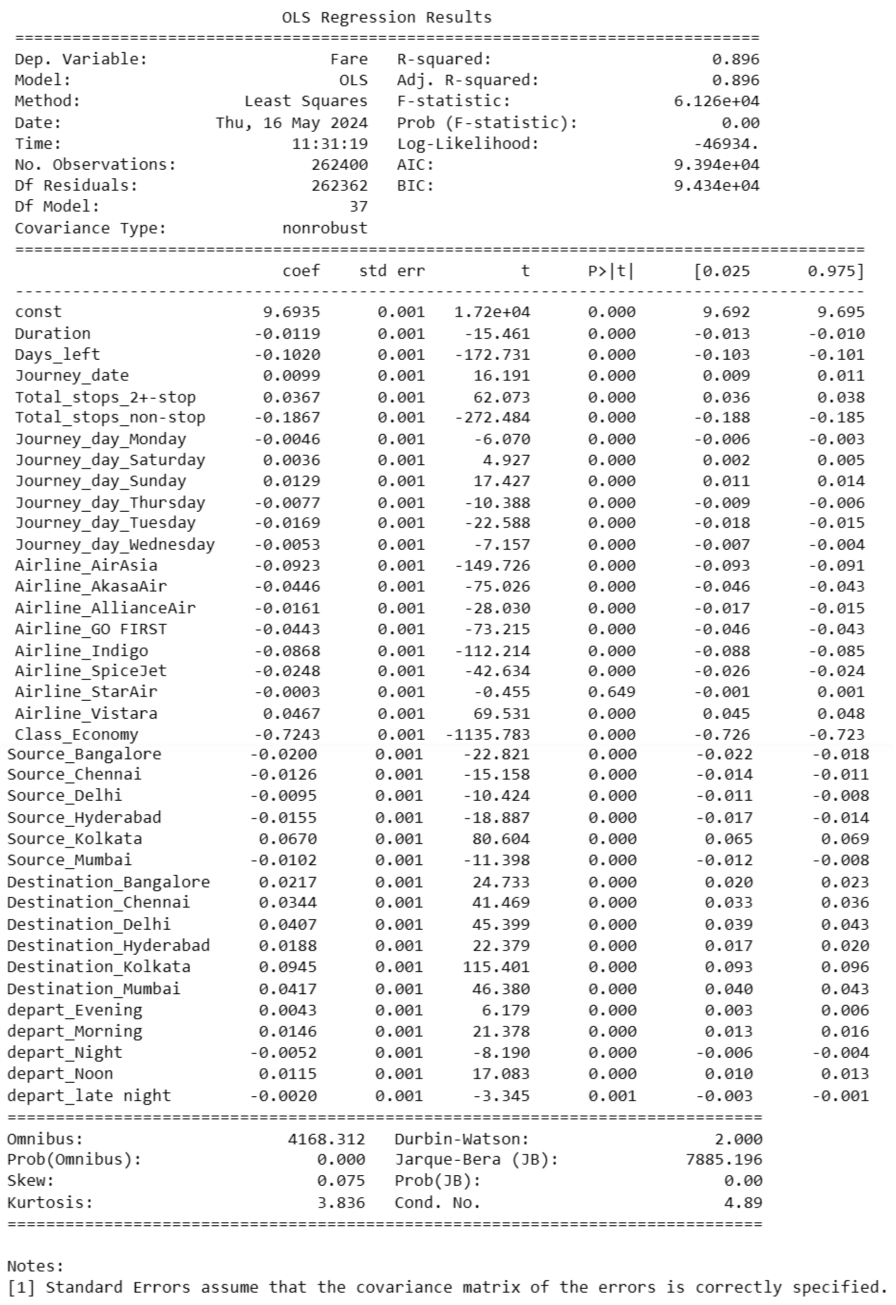


Fig13: Residual plot

So we can see that the residual plot is normally distributed and therefore the fitted linear regression satisfies the condition of homoscedasticity.

So we can get a summary for this linear regression model as follows:



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Fare   R-squared:                       0.896
Model:                            OLS   Adj. R-squared:                  0.896
Method:                 Least Squares   F-statistic:                 6.126e+04
Date:                Thu, 16 May 2024   Prob (F-statistic):               0.00
Time:                        11:31:19   Log-Likelihood:                -46934.
No. Observations:              262400   AIC:                         9.394e+04
Df Residuals:                  262362   BIC:                         9.434e+04
Df Model:                          37
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                     9.6935      0.001   1.72e+04      0.000       9.692       9.695
Duration                 -0.0119      0.001    -15.461      0.000      -0.013      -0.010
Days_left                -0.1020      0.001   -172.731      0.000      -0.103      -0.101
Journey_date              0.0099      0.001     16.191      0.000       0.009       0.011
Total_stops_2+-stop       0.0367      0.001     62.073      0.000       0.036       0.038
Total_stops_non-stop     -0.1867      0.001   -272.484      0.000      -0.188      -0.185
Journey_day_Monday       -0.0046      0.001     -6.070      0.000      -0.006      -0.003
Journey_day_Saturday      0.0036      0.001      4.927      0.000       0.002       0.005
Journey_day_Sunday        0.0129      0.001     17.427      0.000       0.011       0.014
Journey_day_Thursday     -0.0077      0.001    -10.388      0.000      -0.009      -0.006
Journey_day_Tuesday      -0.0169      0.001    -22.588      0.000      -0.018      -0.015
Journey_day_Wednesday    -0.0053      0.001     -7.157      0.000      -0.007      -0.004
Airline_AirAsia          -0.0923      0.001   -149.726      0.000      -0.093      -0.091
Airline_AkasaAir         -0.0446      0.001    -75.026      0.000      -0.046      -0.043
Airline_AllianceAir      -0.0161      0.001    -28.030      0.000      -0.017      -0.015
Airline_GO FIRST         -0.0443      0.001    -73.215      0.000      -0.046      -0.043
Airline_Indigo           -0.0868      0.001   -112.214      0.000      -0.088      -0.085
Airline_SpiceJet         -0.0248      0.001    -42.634      0.000      -0.026      -0.024
Airline_StarAir          -0.0003      0.001     -0.455      0.649      -0.001       0.001
Airline_Vistara           0.0467      0.001     69.531      0.000       0.045       0.048
Class_Economy            -0.7243      0.001  -1135.783      0.000      -0.726      -0.723
Source_Bangalore         -0.0200      0.001    -22.821      0.000      -0.022      -0.018
Source_Chennai           -0.0126      0.001    -15.158      0.000      -0.014      -0.011
Source_Delhi             -0.0095      0.001    -10.424      0.000      -0.011      -0.008
Source_Hyderabad         -0.0155      0.001    -18.887      0.000      -0.017      -0.014
Source_Kolkata            0.0670      0.001     80.604      0.000       0.065       0.069
Source_Mumbai            -0.0102      0.001    -11.398      0.000      -0.012      -0.008
Destination_Bangalore     0.0217      0.001     24.733      0.000       0.020       0.023
Destination_Chennai       0.0344      0.001     41.469      0.000       0.033       0.036
Destination_Delhi         0.0407      0.001     45.399      0.000       0.039       0.043
Destination_Hyderabad     0.0188      0.001     22.379      0.000       0.017       0.020
Destination_Kolkata       0.0945      0.001    115.401      0.000       0.093       0.096
Destination_Mumbai        0.0417      0.001     46.380      0.000       0.040       0.043
depart_Evening            0.0043      0.001      6.179      0.000       0.003       0.006
depart_Morning            0.0146      0.001     21.378      0.000       0.013       0.016
depart_Night             -0.0052      0.001     -8.190      0.000      -0.006      -0.004
depart_Noon               0.0115      0.001     17.083      0.000       0.010       0.013
depart_late night        -0.0020      0.001     -3.345      0.001      -0.003      -0.001
==============================================================================
Omnibus:                     4168.312   Durbin-Watson:                   2.000
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7885.196
Skew:                           0.075   Prob(JB):                         0.00
Kurtosis:                       3.836   Cond. No.                         4.89
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Chapter 5. Conclusion

Model Fit: The R-squared value indicates that approximately 89.6% of the variance in the dependent variable (Fare) is explained by the independent variables included in the model. This suggests that the model provides a reasonably good fit to the data.

Significance of Predictors: The coefficients (coef) of the independent variables represent the estimated change in the dependent variable (Fare) for a one-unit change in the respective predictor variable, holding all other predictors constant. The p-values (P>|t|) associated with each coefficient indicate whether the predictor is statistically significant in predicting the dependent variable. In this case, all predictors have p-values close to zero, indicating that they are statistically significant.

Interpretation of Coefficients: The coefficient for the variable 'Class_Economy' is -0.7226. This suggests that, on average, for each instance where the class of the flight is Economy (compared to other classes), the Fare decreases by $0.7226, holding all other variables constant. This implies that Economy class flights tend to have lower fares compared to flights in other classes.

Categorical Variables: Some predictors are categorical variables represented as dummy variables. For example, "Journey_day_Monday" is a dummy variable representing whether the journey day is Monday or not. A coefficient value of -0.0059 for this variable suggests that, on average, the Fare is lower on Mondays compared to other days, holding all other variables constant.

So, the regression model demonstrates strong performance on both training and testing sets, as indicated by low mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE) values. Additionally, the high R-squared scores indicate that the model explains approximately 89.5% of the variance in the target variable. Overall, this regression model suggests that several factors, including duration, days left before the journey, journey date, number of stops, airline, class, source, destination, and departure time, are significant predictors of the fare.

REFERENCES

Books:

- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Springer, 2001.
- G.J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, Wiley, 1992.

Online Resources:

- https://karansinghreen.medium.com/creating-airline-price-predictor-using-linear-regression-52c68206a599
- https://medium.com/analytics-vidhya/regression-flight-price-prediction-6771fc4d1fb3
- https://github.com/notrichbish/airline-ticket-price-prediction/blob/main/README.md
- https://www.ijraset.com/research-paper/flight-price-prediction
- https://ceur-ws.org/Vol-3283/Paper90.pdf
- https://ieeexplore.ieee.org/abstract/document/8081365
- https://www.eurchembull.com/uploads/paper/a760b593b2939991ad58269e9ca36479.pdf
- https://www.jsr.org/hs/index.php/path/article/view/5303

Research Articles:

- FLIGHT FARE PREDICTION USING MACHINE LEARNING K.D.V.N.Vaishnavi, L. Hima Bindu, M. Satwika, K. Udaya Lakshmi4, M. Harini, N. Ashok
- International flight fare prediction and analysis of factors impacting flight fare Tianyun Deng
- Flight Price Prediction Using Machine Learning Ankita Panigrahi1, Rakesh Sharma, Sujata Chakravarty, Bijay K. Paikaray and Harshvardhan Bhoyar
- Flight Fare Prediction System Using Machine Learning
- Prediction of Flight-fare using machine learning Conference Paper · November 2022
- FLIGHT TICKET PREDICTION USING XGBREGRESSION COMPARED WITH KNEIGHBOUR REGRESSION ALGORITHM N. Sri Sai Venkata Subba Rao , S. John Justin Thangaraj
- FLIGHT TICKET PREDICTION USING XGBREGRESSION COMPARED WITH KNEIGHBOUR REGRESSION ALGORITHM N. Sri Sai Venkata Subba Rao, S. John Justin Thangaraj

APPENDIX

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

```
data = pd.read_csv(r'C:\Users\GARGI\Desktop\project\M\final dataset1.csv')
```

```
data.shape
```

```
(378867, 12)
```

```
data.duplicated().sum()
```

```
4009
```

```
data = data.drop_duplicates() ; data
```

```
data[['Dep_Time', 'Source']] = data['Departure Time'].str.split('\n', expand=True)

data[['Arrival_Time', 'Destination']] = data['Arrival Time'].str.split('\n', expand=True)
```

```
category=['Journey_day','Airline','Class','Source','Total_stops','Destination']
#checking for uniques values
for i in category:
    print(i,data[i].unique())
    print('---------------------------------------------------------------------------------------------')
```

```
Journey_day ['Monday' 'Tuesday' 'Wednesday' 'Thursday' 'Friday' 'Saturday' 'Sunday']
---------------------------------------------------------------------------------------------
Airline ['SpiceJet' 'Indigo' 'GO FIRST' 'Air India' 'AirAsia' 'Vistara' 'AkasaAir'
 'AllianceAir' 'StarAir']
---------------------------------------------------------------------------------------------
Class ['Economy' 'Business']
---------------------------------------------------------------------------------------------
Source ['Delhi' 'Mumbai' 'Bangalore' 'Hyderabad' 'Kolkata' 'Chennai' 'Ahmedabad']
---------------------------------------------------------------------------------------------
Total_stops ['non-stop' '1-stop' '2+-stop']
---------------------------------------------------------------------------------------------
Destination ['Mumbai' 'Bangalore' 'Hyderabad' 'Kolkata' 'Chennai' 'Ahmedabad' 'Delhi']
---------------------------------------------------------------------------------------------
```

```
def change_into_Datetime(col):
    data[col] = pd.to_datetime(data[col])
```

```
import warnings
from warnings import filterwarnings
filterwarnings("ignore")
```

```
data.columns
```

```
Index(['Date of Booking', 'Date_of_journey', 'Journey_day', 'Airline',
       'Flight_code', 'Class', 'Departure Time', 'Total_stops', 'Arrival Time',
       'Duration', 'Days_left', 'Fare', 'Dep_Time', 'Source', 'Arrival_Time',
       'Destination'],
      dtype='object')
```

```
for feature in ['Dep_Time', 'Arrival_Time' , 'Date_of_journey']:
    change_into_Datetime(feature)
```

```python
data["Journey_date"] = data['Date_of_journey'].dt.day
data["Journey_month"] = data['Date_of_journey'].dt.month
data["Journey_year"] = data['Date_of_journey'].dt.year
```

```python
data.head(3)
```

	Date of Booking	Date_of_journey	Journey_day	Airline	Flight_code	Class	Departure Time	Total_stops	Arrival Time	Duration	Days_left	Fare	Dep_Time	So
0	15/01/2023	2023-01-16	Monday	SpiceJet	SG-8169	Economy	20:00\nDelhi	non-stop	22:05\nMumbai	2.0833	1	5335	2024-05-16 20:00:00	
1	15/01/2023	2023-01-16	Monday	Indigo	6E-2519	Economy	23:00\nDelhi	non-stop	01:20\nMumbai	2.3333	1	5899	2024-05-16 23:00:00	
2	15/01/2023	2023-01-16	Monday	GO FIRST	G8-354	Economy	22:30\nDelhi	non-stop	00:40\nMumbai	2.1667	1	5801	2024-05-16 22:30:00	

```python
def extract_hour_min(df , col):
    data[col+"_hour"] = data[col].dt.hour
    data[col+"_minute"] = data[col].dt.minute
    return data.head(3)
```

```python
extract_hour_min(data , "Dep_Time")
```

```python
extract_hour_min(data , "Arrival_Time")
```

```python
cols_to_drop = ['Arrival Time' , "Departure Time","Dep_Time","Arrival_Time"]

data.drop(cols_to_drop , axis=1 , inplace=True )
```

1. when will most of the flights take-off

```python
#### Converting the flight Dep_Time into proper time i.e. mid_night, morning, afternoon and evening.

def flight_dep_time(x):


    if (x>4) and (x<=8):
        return "Early Morning"

    elif (x>8) and (x<=12):
        return "Morning"

    elif (x>12) and (x<=16):
        return "Noon"

    elif (x>16) and (x<=20):
        return "Evening"

    elif (x>20) and (x<=24):
        return "Night"

    else:
        return "late night"
```

```python
data['depart'] = data['Dep_Time_hour'].apply(flight_dep_time)
```

```python
depart = data['Dep_Time_hour'].apply(flight_dep_time)
```

```python
depart.value_counts().plot(kind="bar")
```

```python
plt.figure(figsize=(15, 5))
PD = sns.scatterplot(x='Duration', y='Fare', hue='Airline', data=data)
PD.set(xlabel='Duration of flight', ylabel='Price of Ticket', title='Price Vs Duration of Flight for different Airlines')
plt.show()
```

```python
plt.figure(figsize=(15, 5))
PD = sns.scatterplot(x='Duration', y='Fare', hue='Airline', data=data)
PD.set(xlabel='Duration of flight', ylabel='Price of Ticket', title='Price Vs Duration of Flight for different Airlines')
plt.show()
```

```
fig, axs = plt.subplots(1,2, gridspec_kw= {'width_ratios': [3,1]}, figsize = (15,5))
sns.barplot(y = 'Fare', x = 'Airline', hue = 'Total_stops', data = data.loc[data['Class'] == 'Economy'].sort_values('Fare', ascer
axs[0].set(xlabel='Airlines', ylabel='Price of Ticket', title='Price of Airline tickets based on No. of Stops in Economy Class')
sns.barplot(y='Fare', x='Airline', hue='Total_stops', data= data.loc[data['Class'] == 'Business'].sort_values('Fare', ascending=
axs[1].set(xlabel='Airlines', ylabel='Price of Ticket', title='Price of Airline tickets based on No. of Stops in Business Class')

plt.show(fig, axs)
```

```
fig, axs = plt.subplots(2,3, figsize = (30,10))
plt1 = sns.boxplot(x= data['Fare'], ax = axs[0,0])
plt2 = sns.boxplot(x= data['Journey_date'], ax = axs[0,1])
plt3 = sns.boxplot(x= data['Arrival_Time_hour'], ax = axs[0,2])
plt1 = sns.boxplot(x= data['Dep_Time_hour'], ax = axs[1,0])
plt2 = sns.boxplot(x= data['Duration'], ax = axs[1,1])
plt.tight_layout()
```

```
plt.figure(figsize=(15,8))
sns.distplot(df.Fare)
```

```
<Axes: xlabel='Fare', ylabel='Density'>
```

```
# Transform 'Fare' data into logarithmic scale
df['Fare'] = np.log(df['Fare'])

# Plot the distribution of log-transformed 'Fare' data
plt.figure(figsize=(15, 8))
sns.distplot(df['Fare'])
plt.xlabel('Logarithm of Fare')
plt.title('Distribution of Log-Transformed Fare')
plt.show()
```

```
df = pd.get_dummies(df,columns = ['Total_stops','Journey_day','Airline','Class','Source','Destination','depart','Arrival'],drop_f
```

```
df
```

```
X = df.drop('Fare',axis = 1)
Y=df['Fare']
```

```
X
```

```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
```

```
df_train=sc.fit_transform(X)
x = pd.DataFrame(df_train,columns=X.columns)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
vif_data = pd.DataFrame()
vif_data["Feature"] = x.columns
vif_data["VIF"] = [variance_inflation_factor(x.values, i) for i in range(len(x.columns))]

print(vif_data)
```

```
x = x.drop(['Dep_Time_hour','Arrival_Time_hour','Arrival_Evening','Arrival_Morning','Arrival_Night','Arrival_Noon','Arrival_late
```

```
x
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, Y, test_size=0.3)
```

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train, y_train)
```

```
LinearRegression()
```

```
from sklearn import metrics
from sklearn.metrics import mean_squared_error, r2_score
```

```
y_pred_train = lm.predict(X_train)
y_pred_test = lm.predict(X_test)

print("MAE for training set:", metrics.mean_absolute_error(y_train, y_pred_train))
print("MSE for training set:", metrics.mean_squared_error(y_train, y_pred_train))
print("RMSE for training set:", np.sqrt(metrics.mean_squared_error(y_train, y_pred_train)))
print("R2 score for training set:", metrics.r2_score(y_train, y_pred_train))

print("MAE for testing set:", metrics.mean_absolute_error(y_test, y_pred_test))
print("MSE for testing set:", metrics.mean_squared_error(y_test, y_pred_test))
print("RMSE for testing set:", np.sqrt(metrics.mean_squared_error(y_test, y_pred_test)))
print("R2 score for testing set:", metrics.r2_score(y_test, y_pred_test))
```

```
MAE for training set: 0.2229847635553437
MSE for training set: 0.0837307957972198
RMSE for training set: 0.28936274085863195
R2 score for training set: 0.8962554829467508
MAE for testing set: 0.22332471664004097
MSE for testing set: 0.08367868477782886
RMSE for testing set: 0.28927268239124976
R2 score for testing set: 0.8961777035649338
```

```
lm.intercept_
```

```
9.693504215190181
```

```
import statsmodels.api as sm

# Reset indices for X_train and y_train
X_train.reset_index(drop=True, inplace=True)
y_train.reset_index(drop=True, inplace=True)

# Add constant to X_train
X_train_sm = sm.add_constant(X_train)

# Fit the OLS model
model_sm = sm.OLS(y_train, X_train_sm).fit()

# Print summary
print(model_sm.summary())
```

```
test_residuals = y_test - y_pred_test
sns.displot(test_residuals,bins=25,kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x2bf18787760>
```