**Flight Ticket Price Prediction**

Submitted by

SHASHANKA

**Problem Definition**

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow and it will be a different story. We might have often heard travellers saying that flight ticket prices are so unpredictable.

Someone who purchases flight tickets frequently would be able to predict the right time to procure a ticket to obtain the best deal. Many airlines change ticket prices for their revenue management. The airline may increase the prices when the demand is to be expected to increase the capacity.

In this article, we will use the machine learning regression methods to predict the prices of Air tickets with the use of our dataset.

To estimate the airfare, we have the dataset with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities.

Features:

**Airline**: The name of the airline.

**Date of Journey**: The date of the journey

**Source**: The source from which the service begins.

**Destination**: The destination where the service ends.

**Route**: The route taken by the flight to reach the destination.

**Departure Time**: The time when the journey starts from the source.

**Arrival Time**: Time of arrival at the destination.

**Duration**: Total duration of the flight.

**Total Stops**: Total stops between the source and destination.

**Additional Info**: Additional information about the flight

**Price**: The price of the ticket

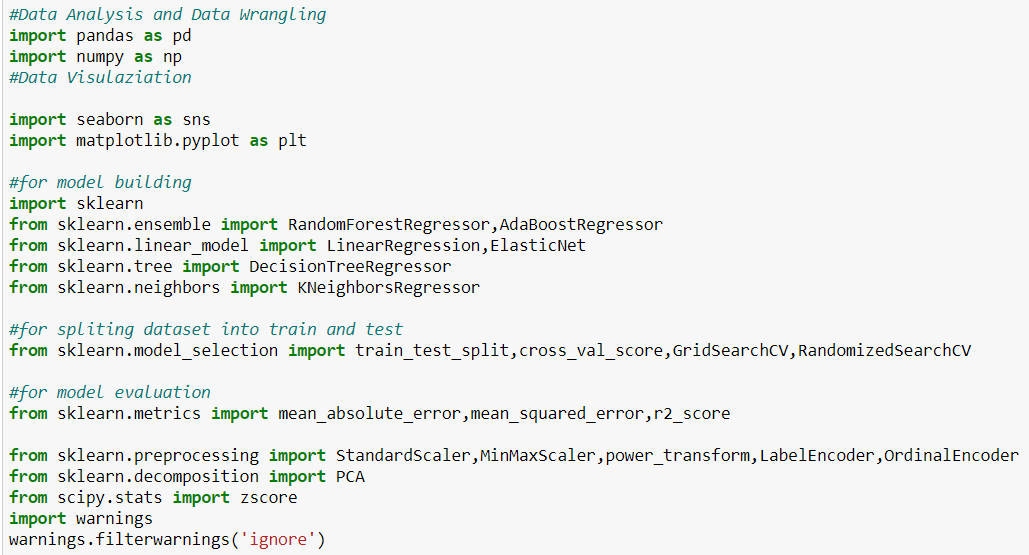
**Conceptual Background of the Flight Ticket Price Prediction**

The flight ticket buying system is to purchase a ticket many days prior to flight take-off so as to stay away from the effect of the most extreme charge. Mostly, aviation routes don’t agree with this procedure. Plane organizations may diminish the cost at the time they need to build the market and at the time when the tickets are less accessible. They may maximize the costs. So the cost may rely upon different factors. To foresee the costs this venture uses AI to exhibit the ways of flight tickets after some time. All organizations have the privilege and opportunity to change its ticket costs at any time. Explorers can set aside cash by booking a ticket at the least cost.

People who travel by flight frequently are aware of price fluctuations. The airlines use complex policies of Revenue Management for execution of distinctive evaluating systems. The evaluating system as a result changes the charge depending on time, season, and festive days to change the header or footer on successive pages. The ultimate aim of the airways is to earn profit whereas the customer searches for the minimum rate. Customers usually try to buy the ticket well in advance of the departure date so as to avoid a hike in airfare as the date comes closer. But actually this is not the fact. The customer may wind up by giving more than they ought to for the same seat.

It is hard for the client to buy an air ticket at the lowest cost. For this few procedures are explored to determine time and date to grab air tickets with minimum fare rate. The majority of these systems are utilizing the modern computerized system known as Machine Learning.

**Importing the Required Libraries**



I have imported all the regression algorithms and metrics of regression, since we know that we are predicting the numerical value which is the price of an air ticket. Along with regression algorithms I have imported all the required libraries for data pre-processing.

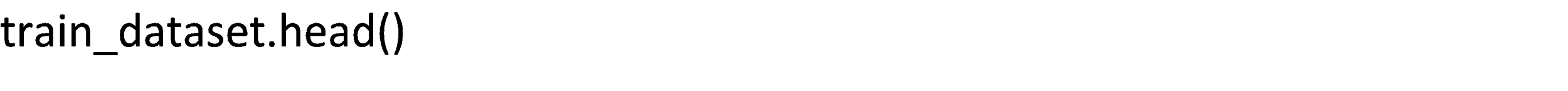
**Loading the dataset**

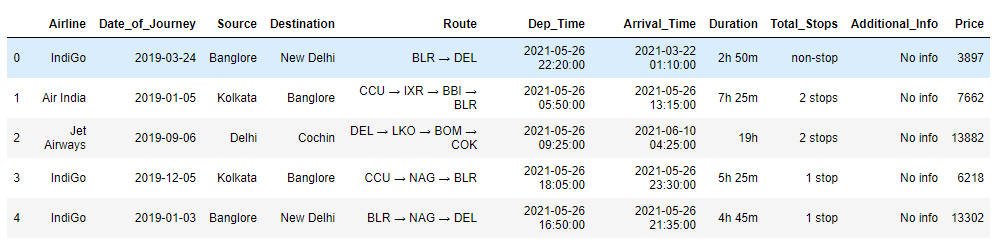
Let’s import the dataset and convert into a data frame using pandas.

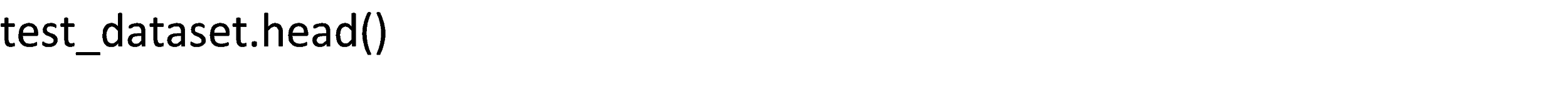
We have two datasets, one for training and one for testing.

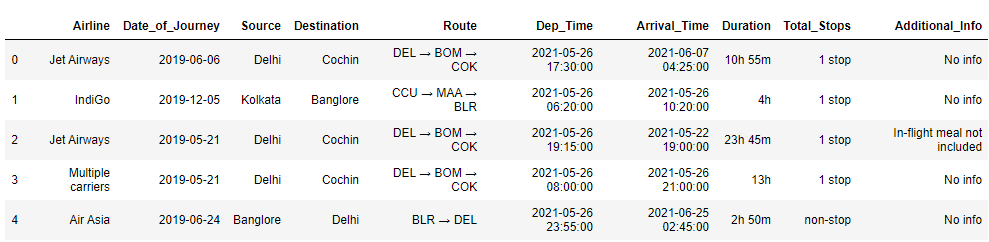
Since we already know that "Date of Journey"," Departure Time" and "Arrival Time" are date columns, loading them as dates.

Loading both test and train dataset for data pre-processing.









We can see that our test dataset has no target column (Price).

## **EDA (Exploratory Data Analysis)**

Let’s check the shape of both the train and test dataset.



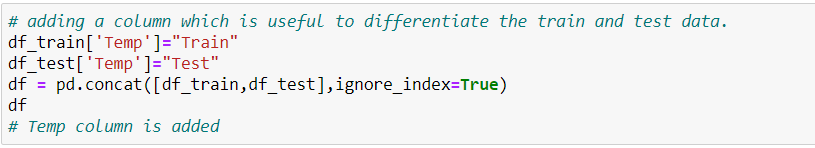
We have 10,643 rows and 11 columns in the train dataset including target column (Price).



We have 2,671 rows and 10 columns in the test dataset excluding the target column.

**Let’s combine these datasets for data pre-processing:**

And also I will make a temporary column called temp which will show the train and test dataset in future.

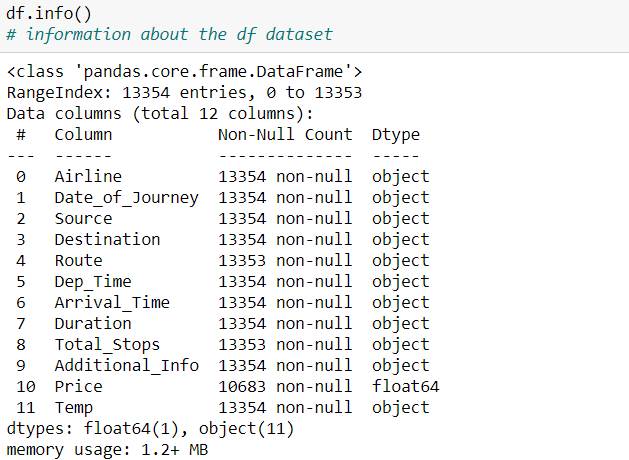


We can see the size of the combined dataset



Now we have 13354 rows and 12 columns including temporary column, which will be removed later. We don't have a target variable in the rows where the Test data is present in the temporary column.

Let’s get the information of the dataset using info () method:



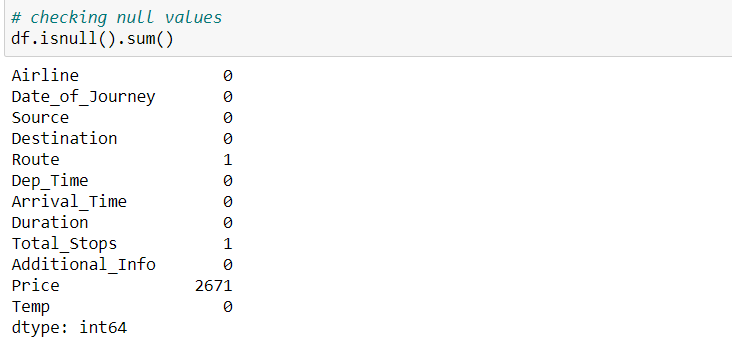
Price is the target column, Temp is Temporary column and rest all columns are independent columns.

Now the dataset has 13,354 rows and 12 columns, in which ‘Price’ column is our target variable.

We can also see that there are no null values in any of the columns except for the Price column which was not provided. That is what we need to predict after building the model.

Only the Price column is float data type, all other columns are object data type which has to be converted before model building.

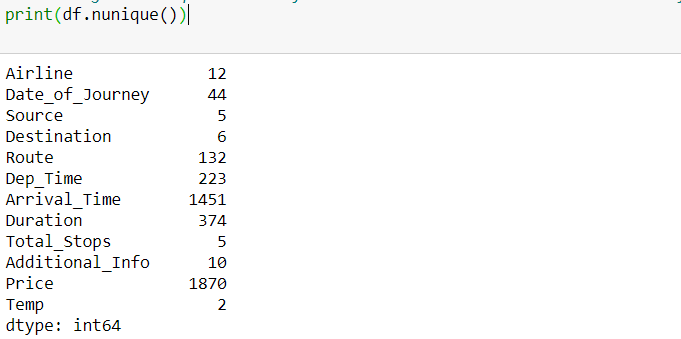
**Checking for Null Values:**

****

We have 2671 null values in Price column, which we can ignore.

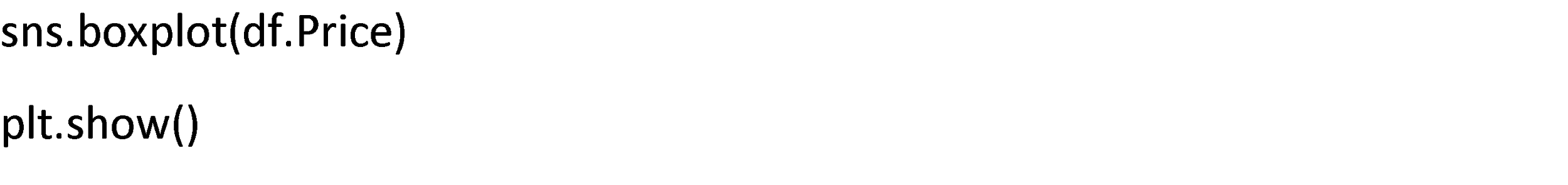
Route and Total stops columns have 1 null values each.

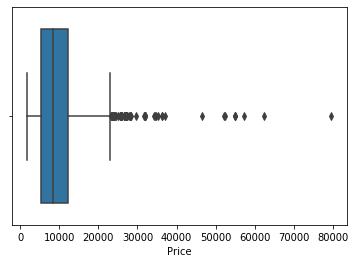
**Checking for unique values in each object columns:**

****

**Data Visualization**

Since Price is the only numerical column let’s check the box plot of our target column, to check if there are outliers.

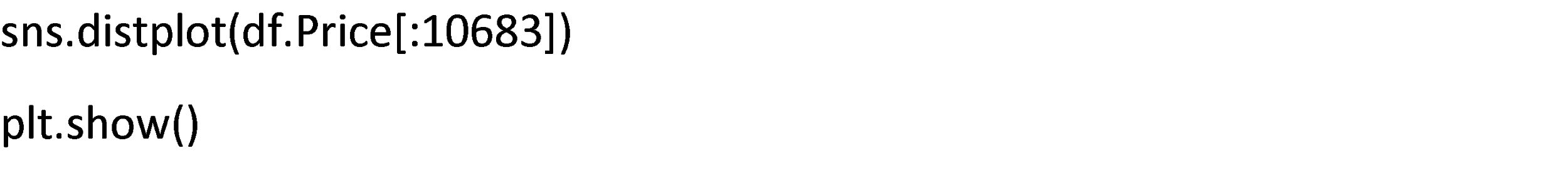


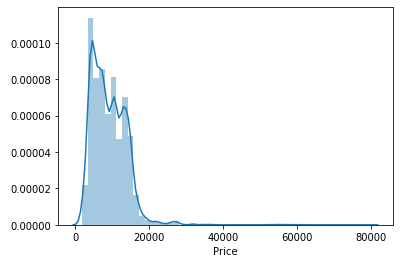


From the above plot we can see there are many outliers in the target column.

Let’s check the distribution of the target column:

I am taking the 0th row to 10,683rd row because all the values are Nan after that.





From the above plot we can see that data is right skewed in the target column.

**EDA Concluding Remarks**

Exploratory data analysis is a way of visualizing, summarizing and interpreting the information that is hidden in rows and column format. Once EDA is complete and insights are drawn, its feature can be used for supervised and unsupervised machinelearning modelling.

Below are insights obtained after performing EDA:

**1. Shape of the dataset:** 13,354 rows and 12 columns including Target column and Temporary column.

**2. Null values:** There are 2 null values in the present Route and Total stops each.

**3. Column DataTypes:** There are 11 object DataTypes columns and 1 numerical column in this dataset, which is our target column (Price).

**4. Bivariate Analysis with Target Column:** We will perform bivariate analysis of columns with our target variable in the further steps along with data pre-processing.

**5. Outliers:**  We saw that there are many outliers in our target column, which has to be removed.

**6. Distribution:** We saw that our target column (price) is Right skewed and most of the data is concentrated at 0 to 20,000.

**7. Skewness:** Once the outlier’s removal is done, then we can handle skewness. Because once the outliers are removed, skewness will be reduced automatically.

Let’s continue the EDA process along with the data pre-processing.

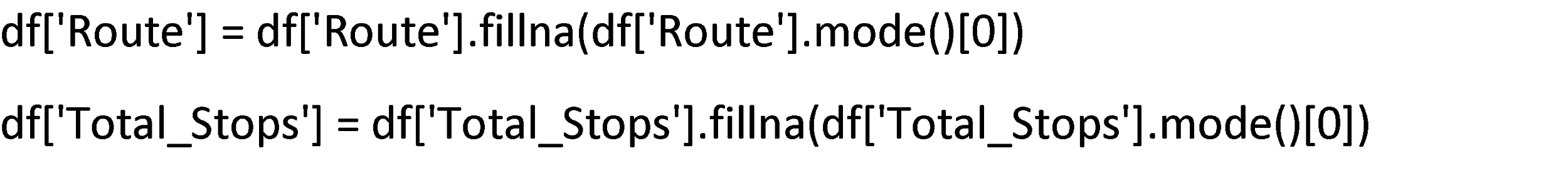
**Data Pre-processing**

Since we have some null values and all the columns are object DataTypes. Let’s continue the data visualization process along with the data pre-processing.

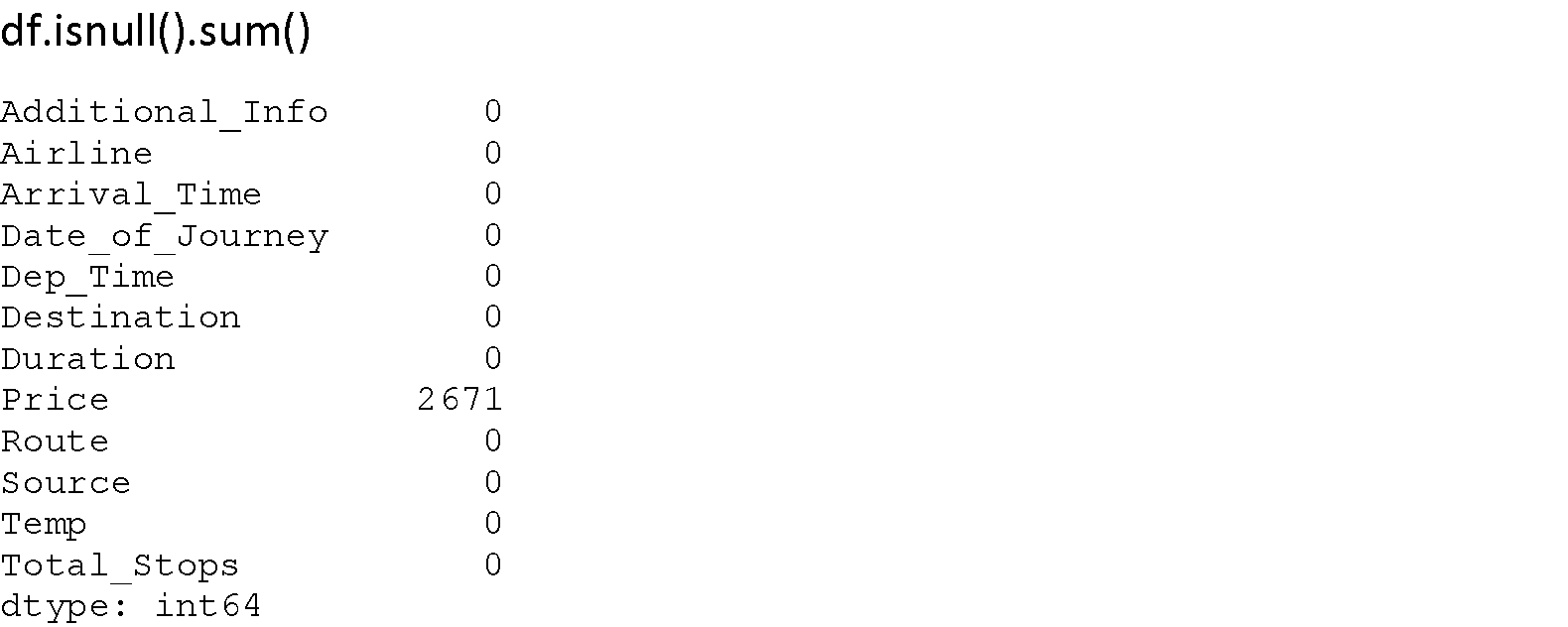
**Let’s Begin by handling Null values**:

We already know that there are only two null values in this dataset and both the columns are object DataTypes, so we can replace them with the mode.

Replacing the Nan value with mode:



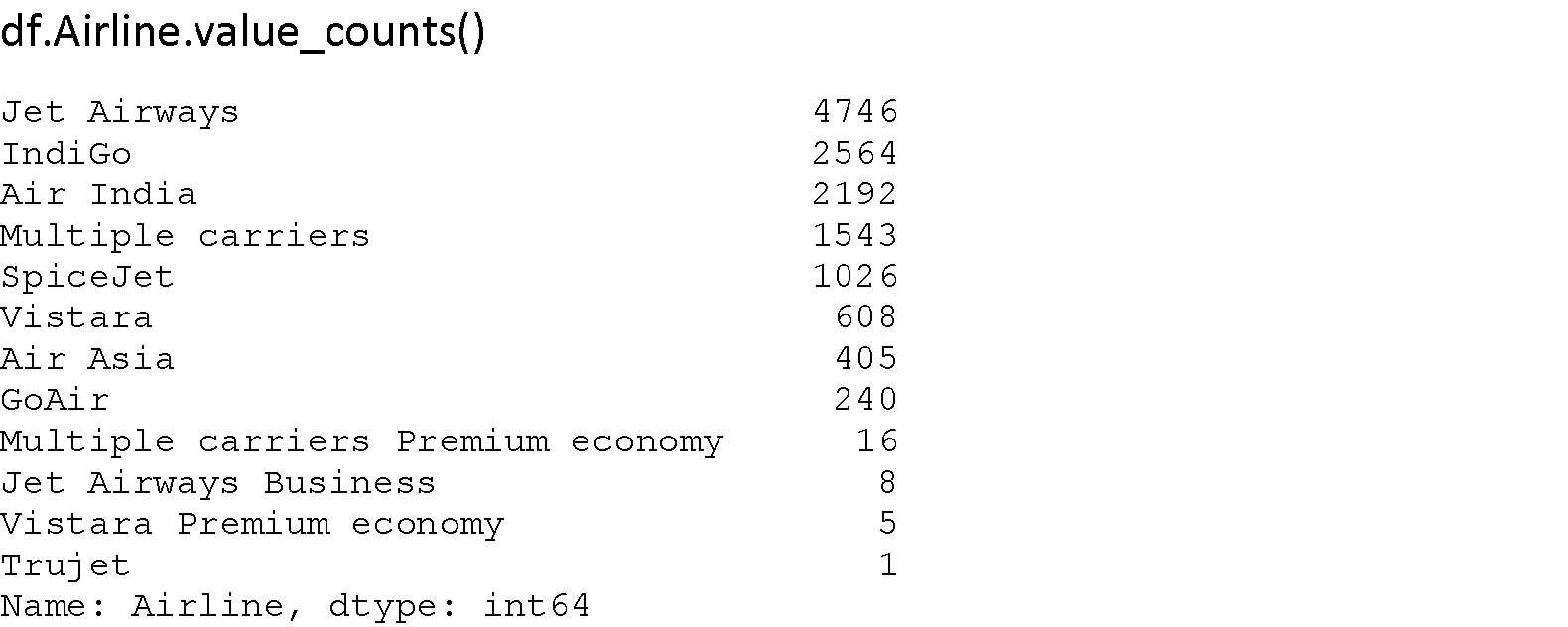
Now there are no null values in this dataset except for the price column.



**Feature Engineering:**

Replacing the Duplicate values in Airline columns:

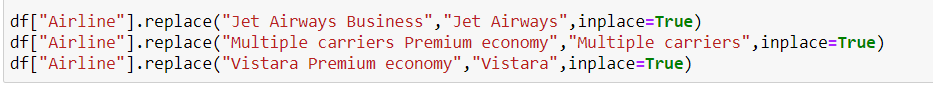
**Airline Column:**

****

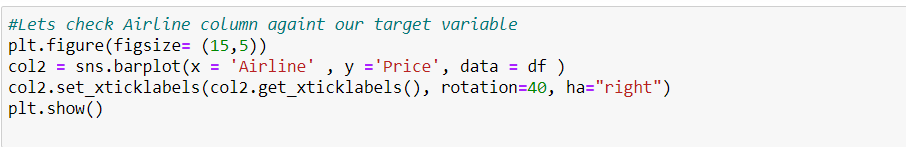
1. We can replace Jet Airway Business with Jet Airways.

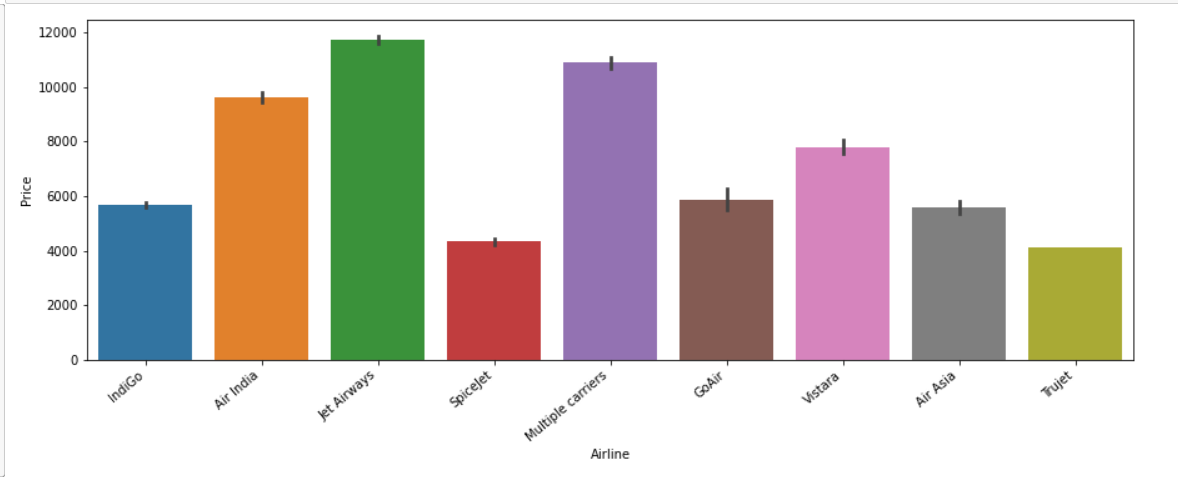
2. Vistara Premium economy with Vistara.

3. Multiple Carriers Premium economy with Multiple carriers



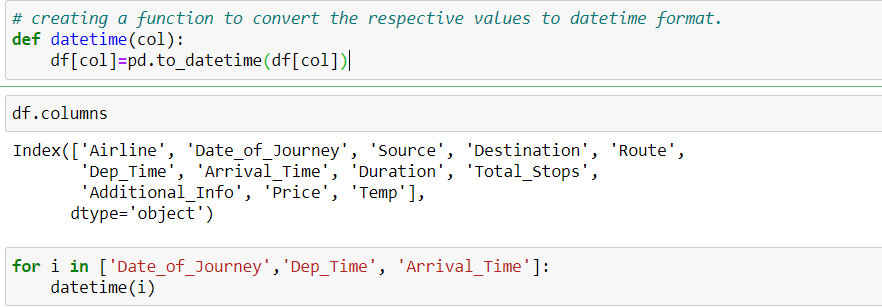
Bivariate analysis of Airline column with target variable.



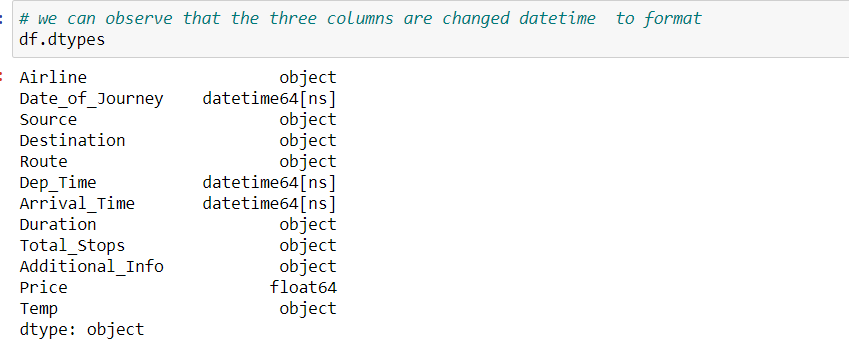
****

From the above plot we can observe that the price of jet airways is high compared to all Airlines, and low for Trujet.

**Date\_of\_Journey, Dep\_Time,Arrival\_Time are in object, so converting them into datetime format:**

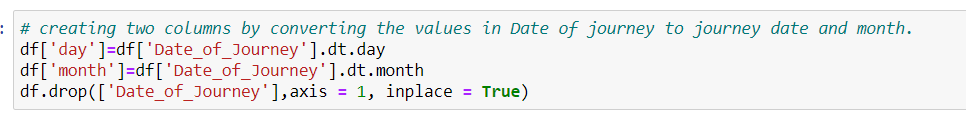
****

The respective columns are converted to datetime format, we can see them in the figure below.



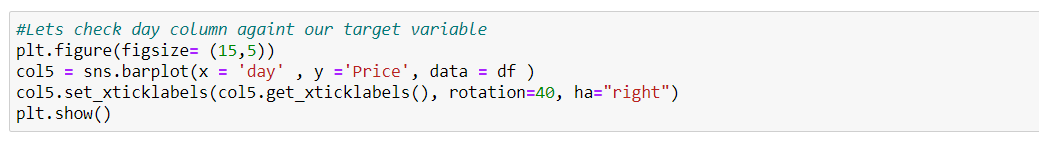
**Date of the journey:**

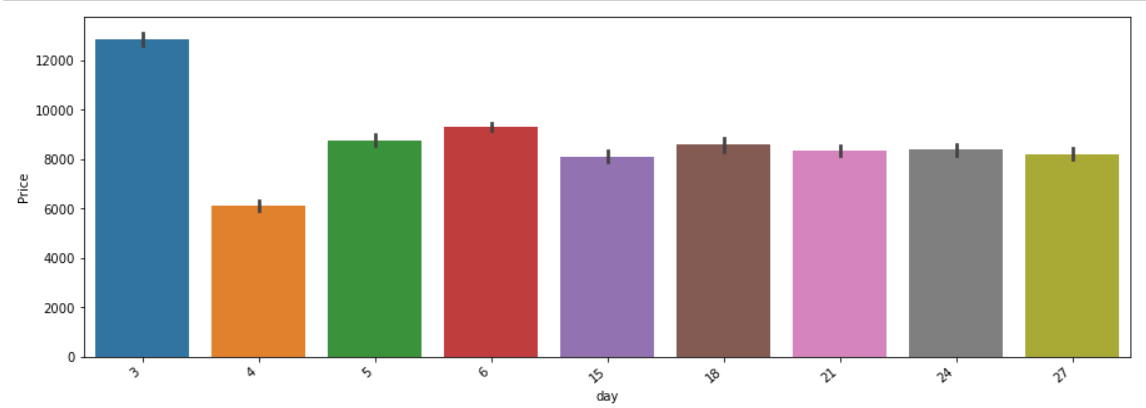
We are extracting only the day and month of the journey, As the year is the same in column so we are excluding it and dropping the Date of journey column.



We extracted day and month from the Date of journey column and dropped it .

Bivariate analysis ofDay column with target variable.





From the above plot we can observe that the 3rd day and 4th day of the month are having high and low prices respectively.

**Extracting the useful information Departure time and Arrival time column:**

We can extract below information from Departure time and Arrival time columns:

1. Year

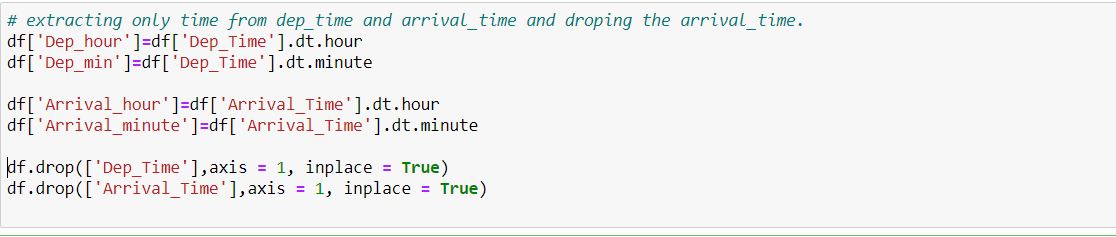
2. Month

3. Day

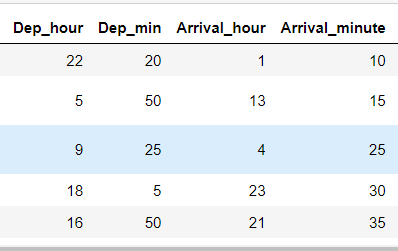
4. Time of the day

All the data provided is of year 2021, month May and day Wednesday.so no use in extracting year, month and day from this column.

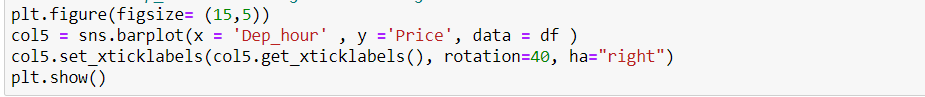
Let’s extract “hour” and “minute” from the “Departure time” and “Arrival column”.

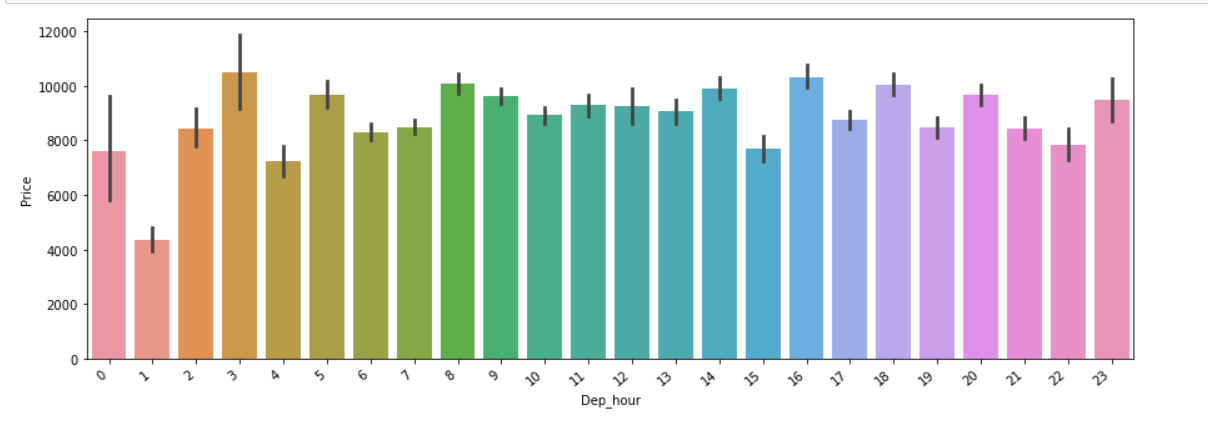


As shown in the below figure, we extract only the hours and minutes from the Departure and Arrival features and drop them.



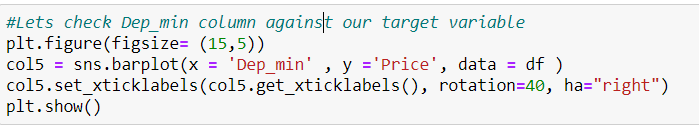
Bivariate analysis of Departure hour column with target variable.

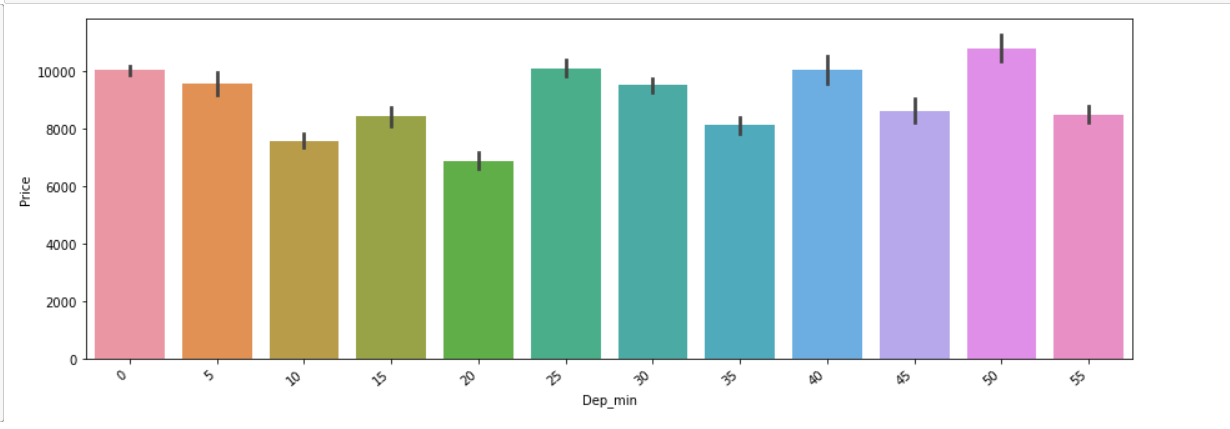




From the above plot we can see that the price is high when the dep\_hour is at 3 AM, and low at 1 AM.

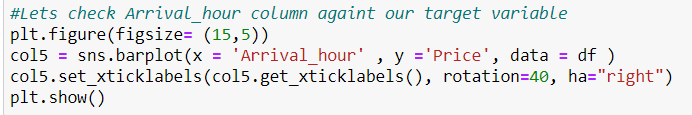
Bivariate analysis of Departure minute column with target variable.

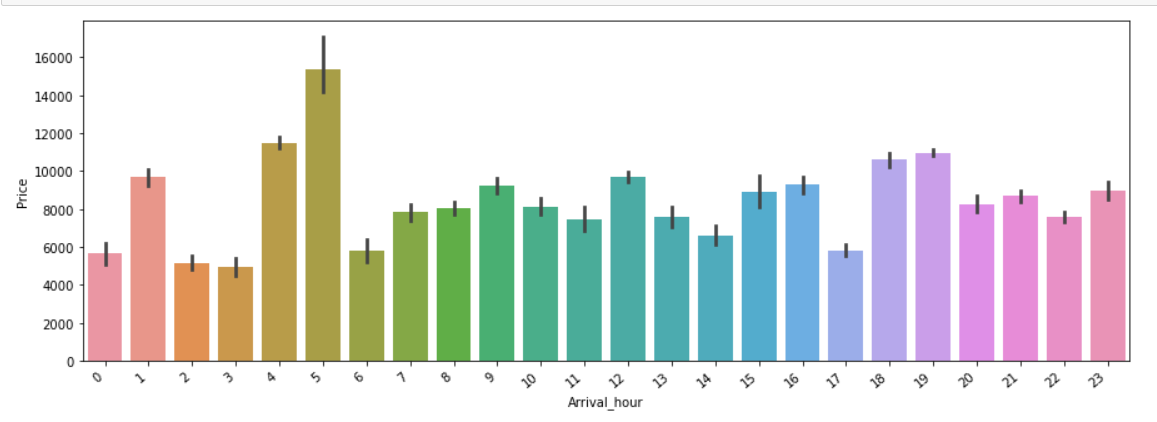




From the above plot we can see that the price is high at the 50th minute of departure time and low at 20th minute of departure time.

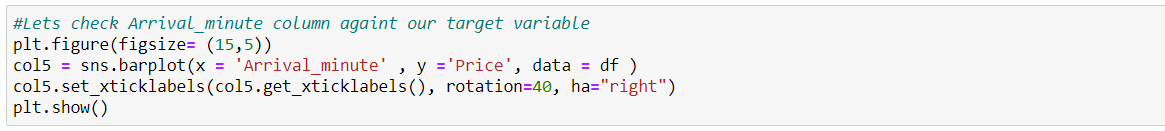
Bivariate analysis of Arrival hour column with target variable.

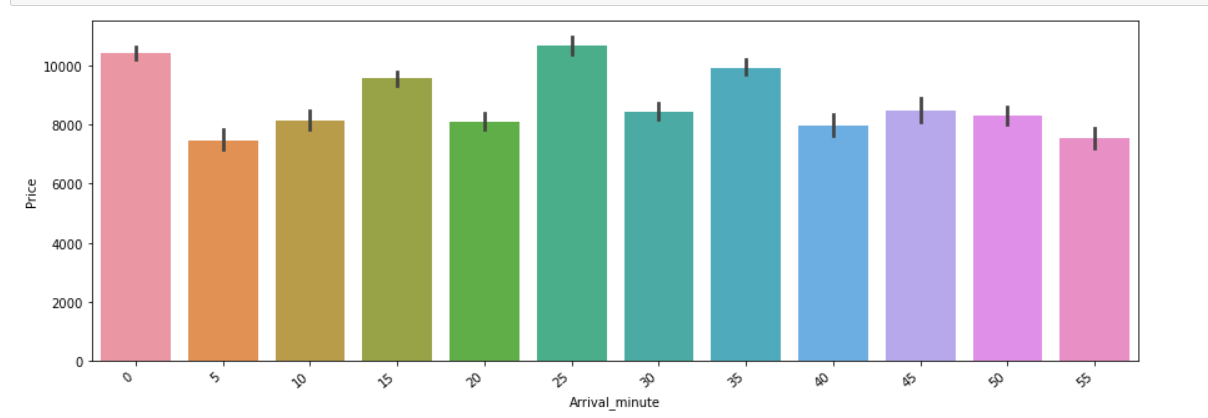




From the above plot we can see that the price is high at the 5AM of Arrival time and low at 3AM of Arrival time.

Bivariate analysis of Arrival minute column with target variable.

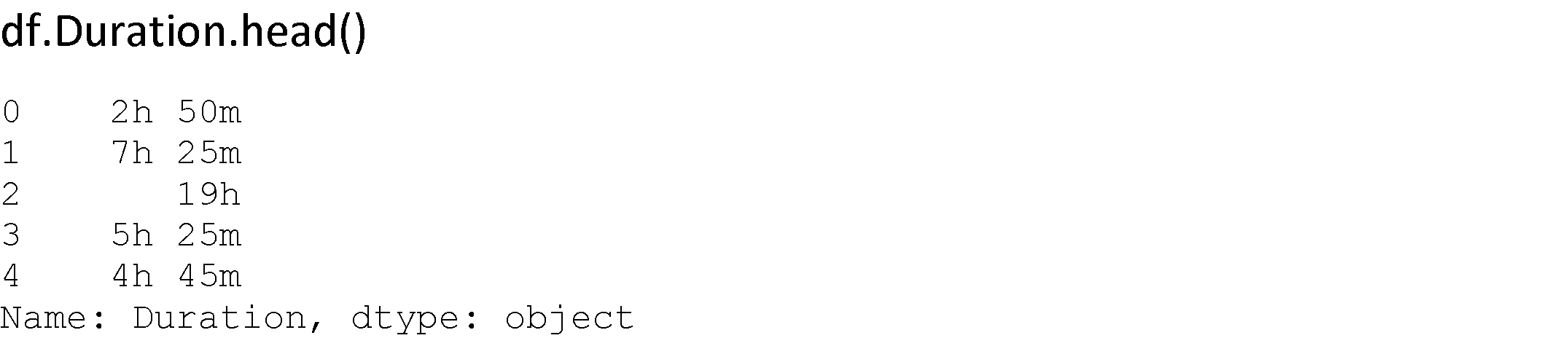




From the above plot we can see that the price is high at the 25th minute of Arrival time and low at 55th minute of Arrival time

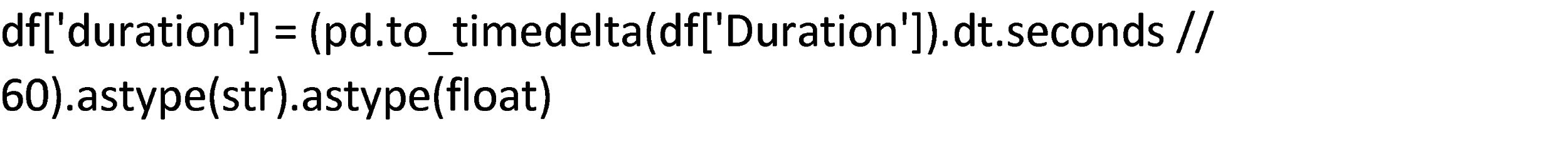
**Duration column:**

We can see that the duration column is object DataType, which was supposed to float.

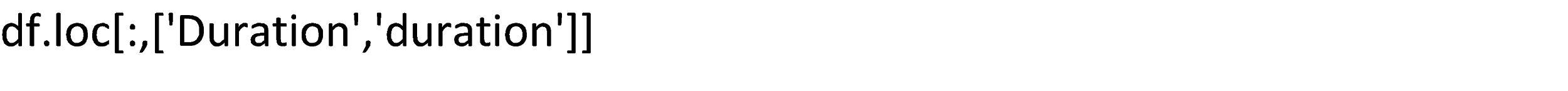


We have removed ‘h’ and ‘m’ from these columns and multiplied them with 60.

Let’s create a new column called duration and extract the data in minutes.



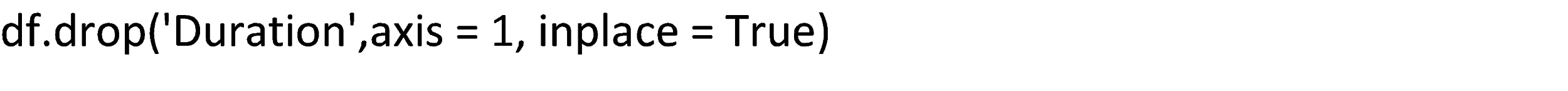
Comparing old duration column with extracted duration column is in minutes:



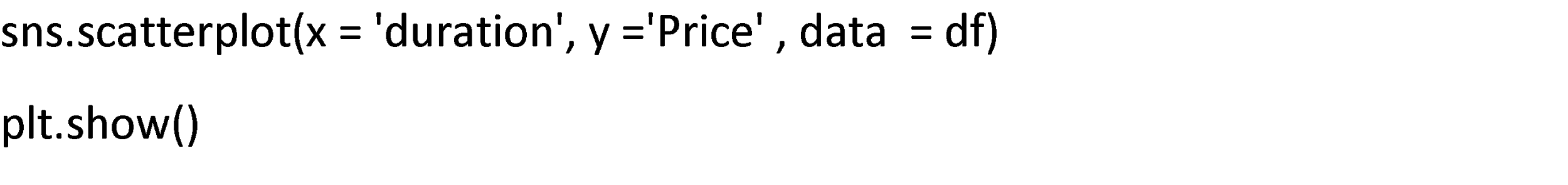


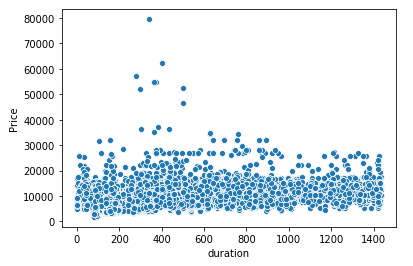
Now we extracted the useful data from the Duration column in float format.

Let’s drop old duration column, as we extracted useful information:



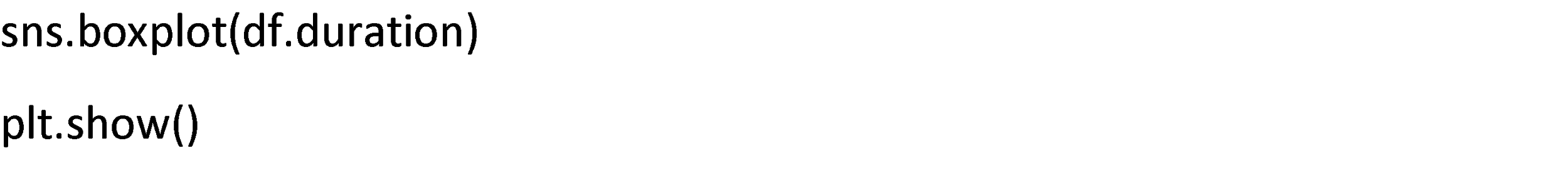
Bivariate analysis of duration column with target variable.

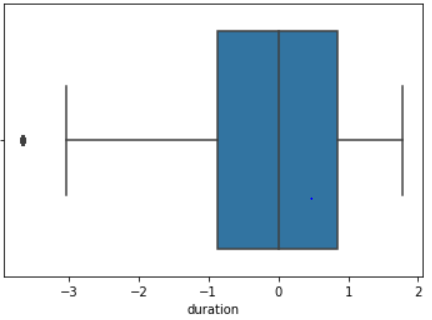




From the above plot we can see that duration and Price have moderate positive correlation with the duration column.

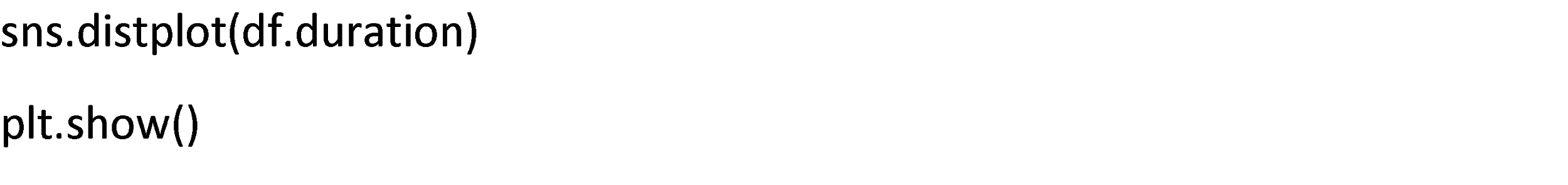
Let’s check for outliers in the duration column.

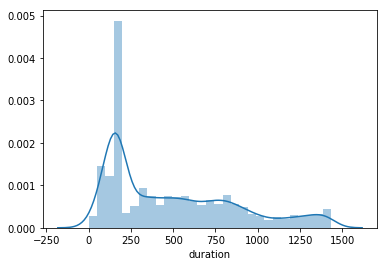




From the above plot we can see there is only one outlier in the duration column so we are neglecting the outlier.

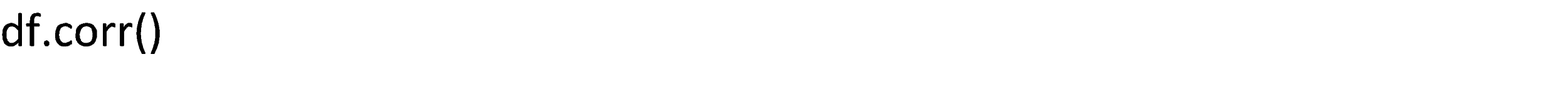
Let’s check for distribution

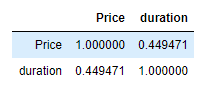




We can see that the duration column is slightly right skewed.

Let’s check correlation between these columns with the Target variable since this is the only numerical column except the Price column.

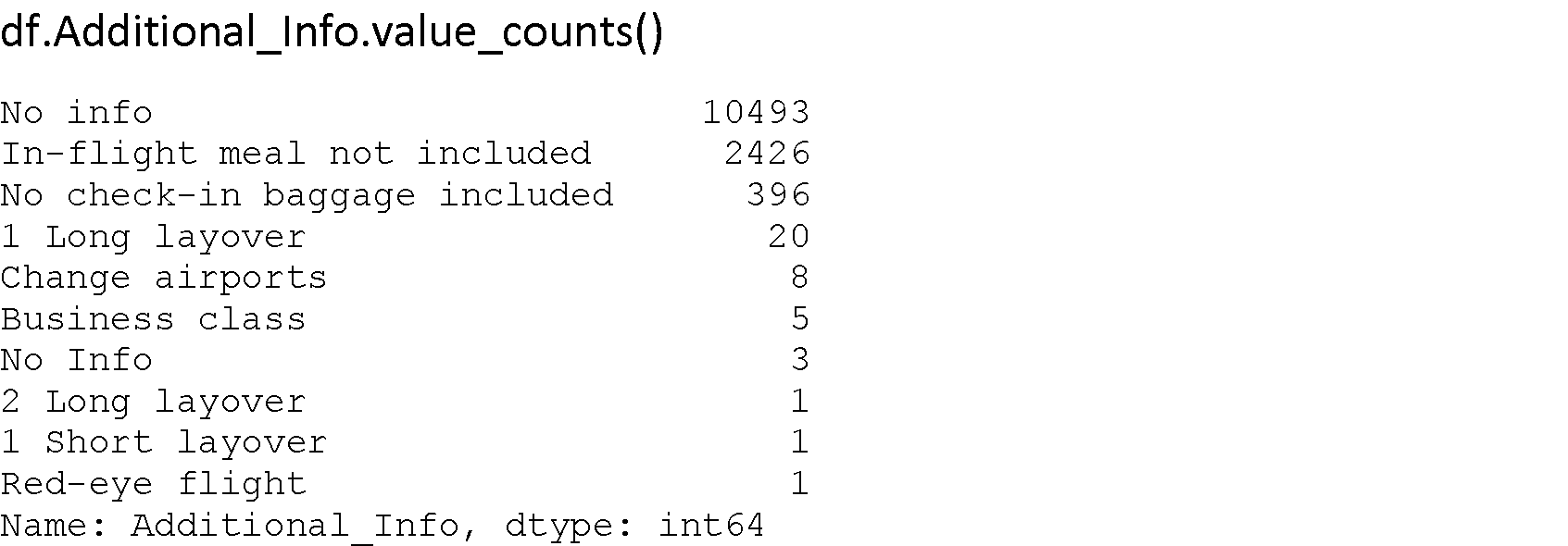




These are the only two numerical columns, correlation is 0.449.

**Replacing the Duplicate values in Additional Info column:**

**Additional Info column:**

****

Additional info column has 10493 no info values which means 77 % of the information is no info.

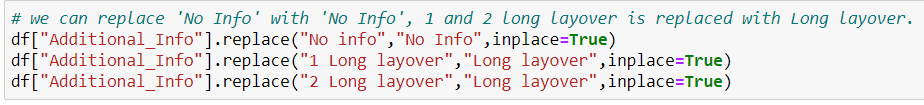
We have 2426 rows for In-flight meal not included.

We have only 20 rows for 1 long layover, 2 rows for 2 long layover, 1 row for 1 short layover.

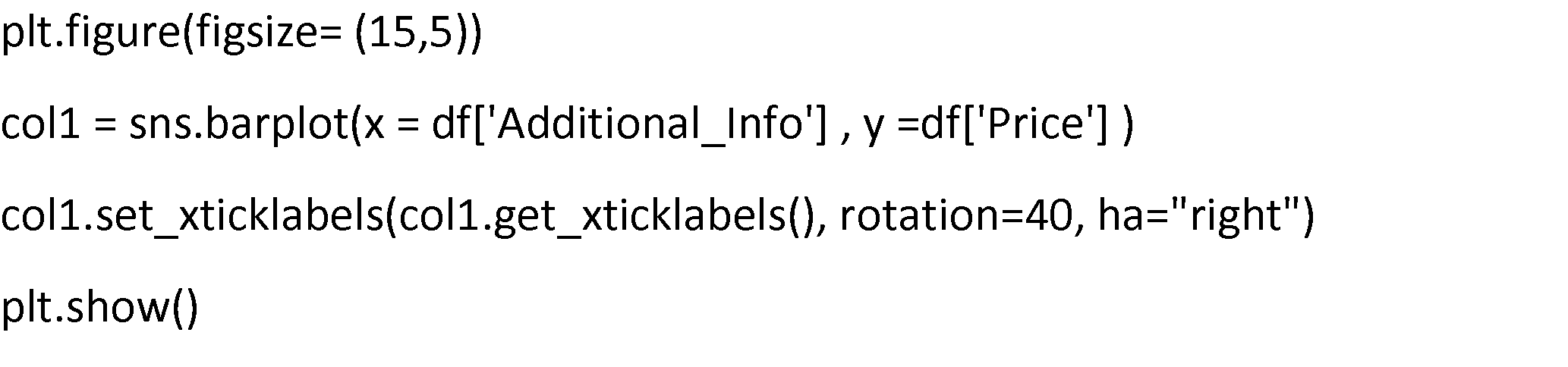
SO we can replace some values in this columns:

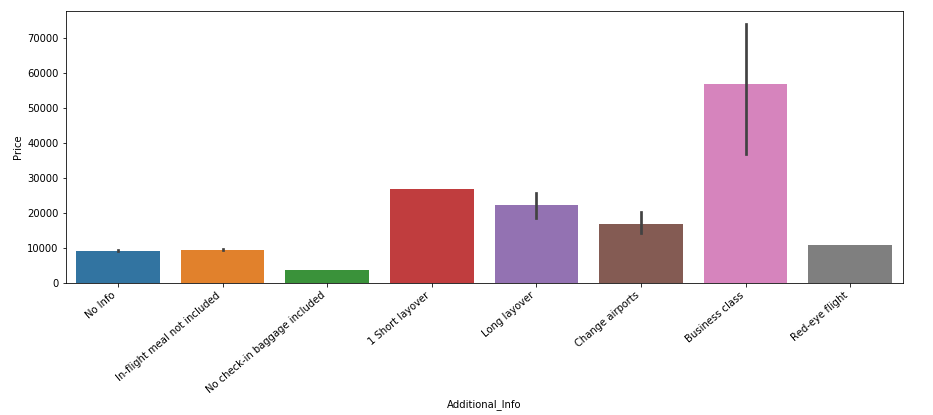
1. no info and NO Info

2. 1 and 2 long layover with long layover



Bivariate analysis of Additional info column with target variable.

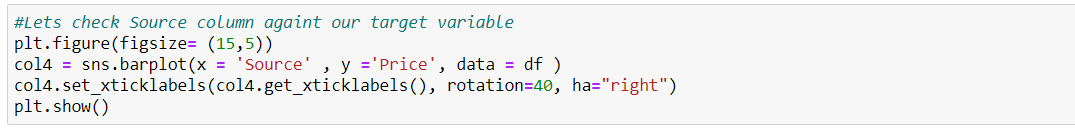


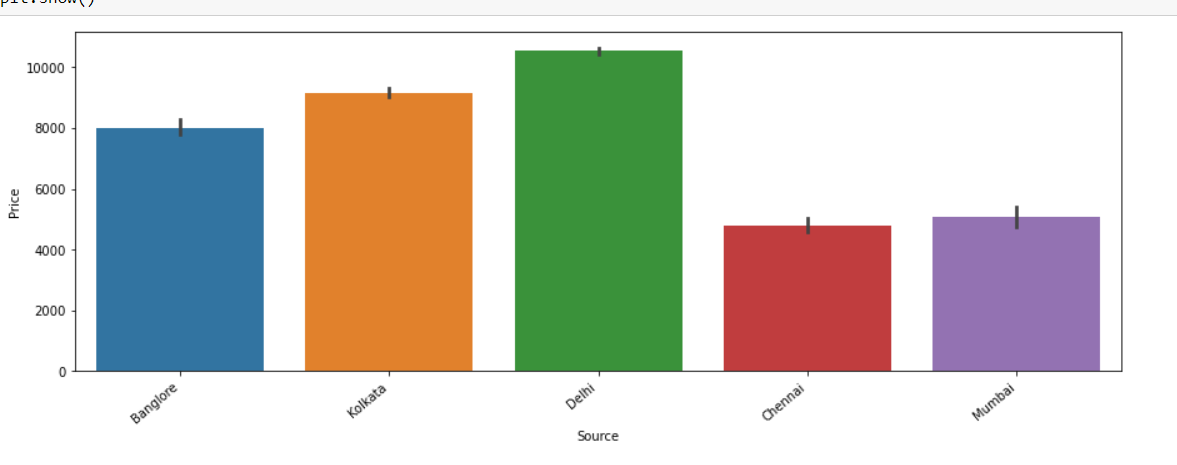


From the above plot we can see that business class has the highest price among all.

**Source column:**

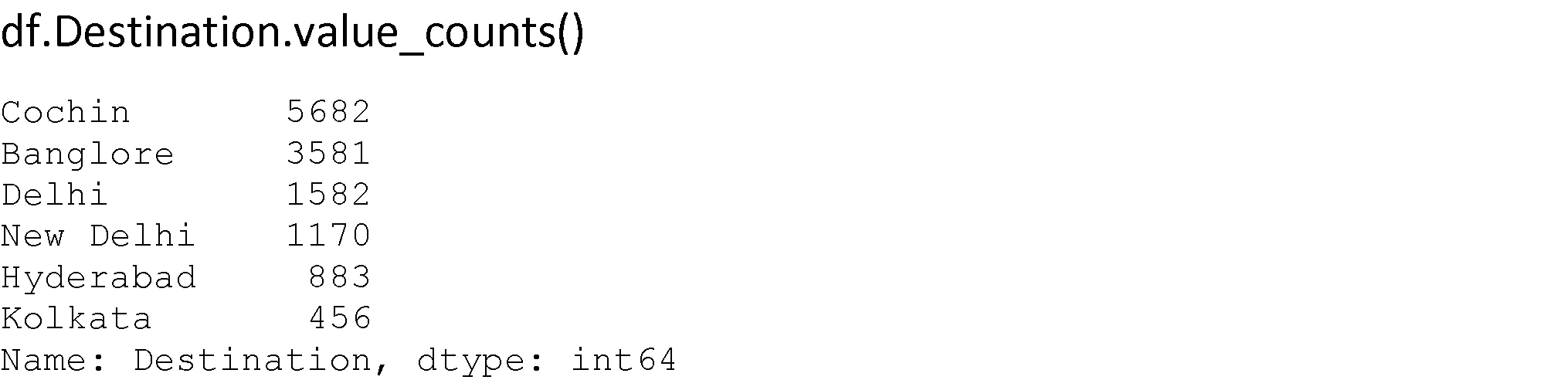
Bivariate analysis of source column with target variable.



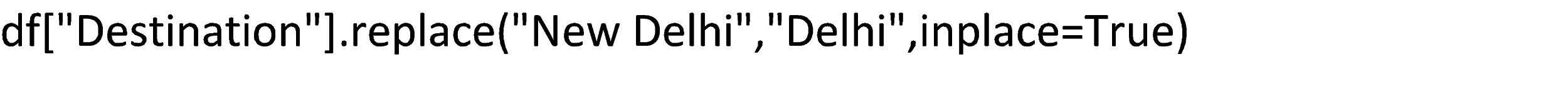


From the above plot we can see that people who travelled from Delhi have paid more flight price compared to all.

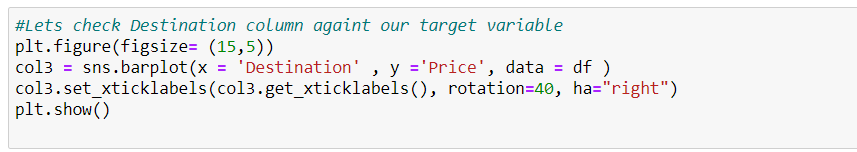
**Destination column:**

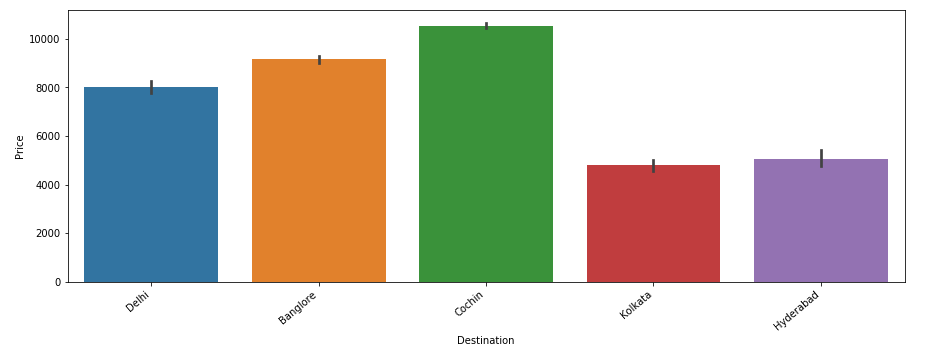
****

We can replace New Delhi with Delhi.



Bivariate analysis of destination column with target variable.

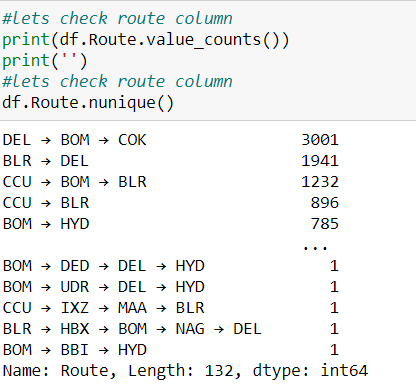




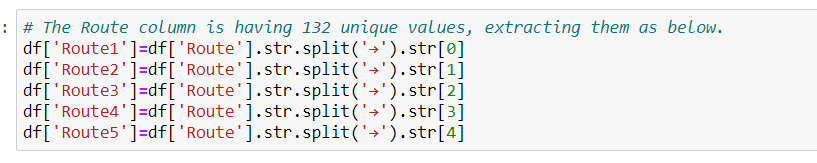
From the above plot we can see that people who travelled to Cochin have paid more flight price compared to all.

**Route column:**

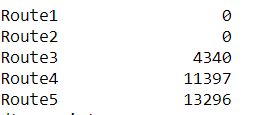
We have 132 value counts in the Route column,

****

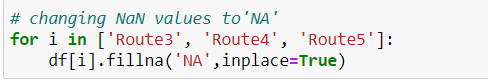
Categorizing the total routes into Route1, Route2 and Route3, Route4 and Route5 . After that drop the Route column.



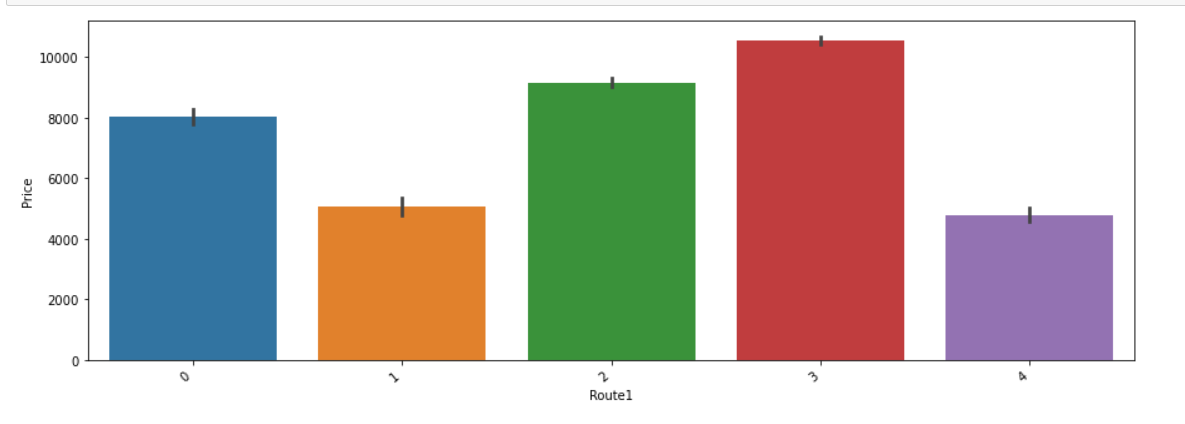
Now we can see the null values in the extracted Routes.



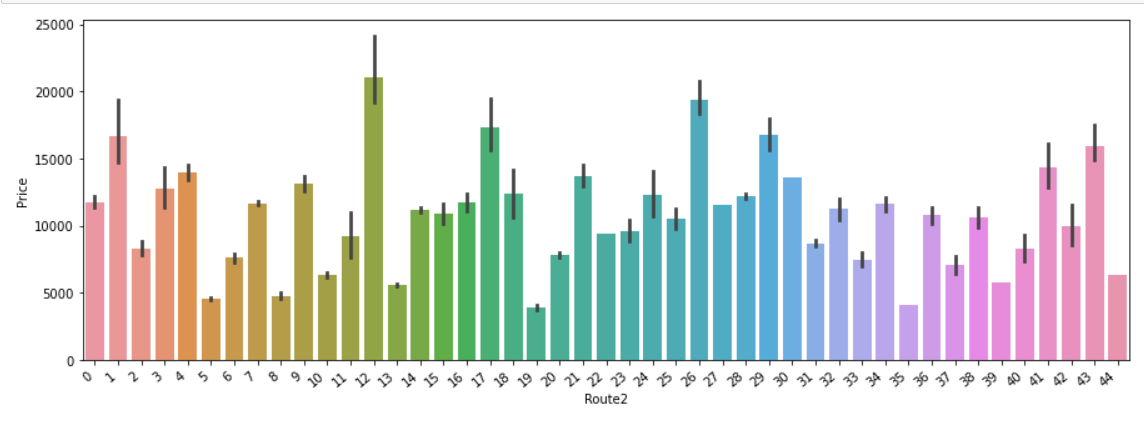
We are filling null values with NA.

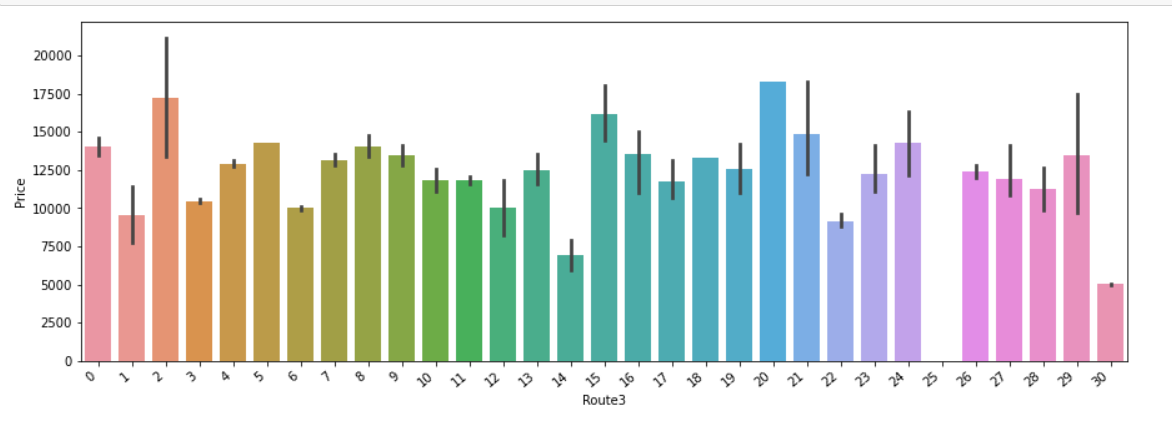


Bivariate analysis of Route columns with target variable.

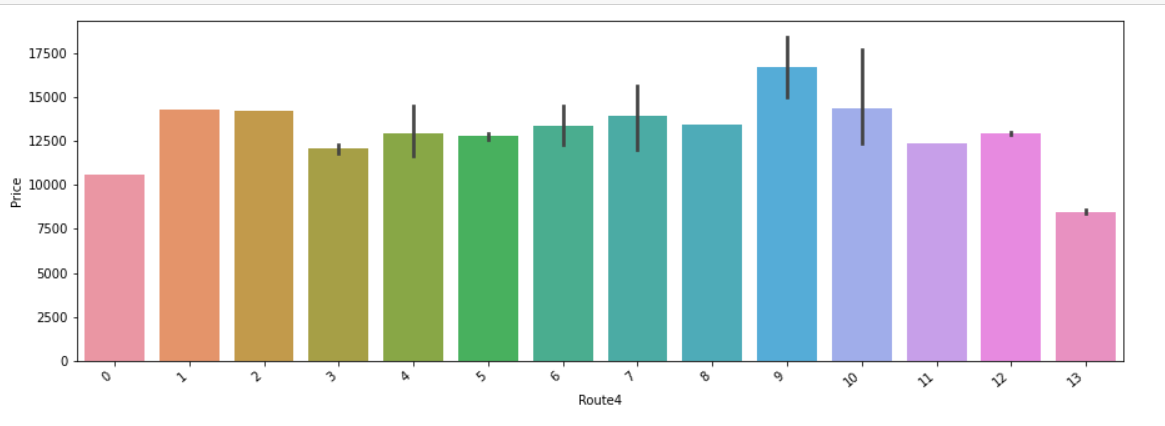


From the above plot we can observe that Route5 3rd way has high price values.

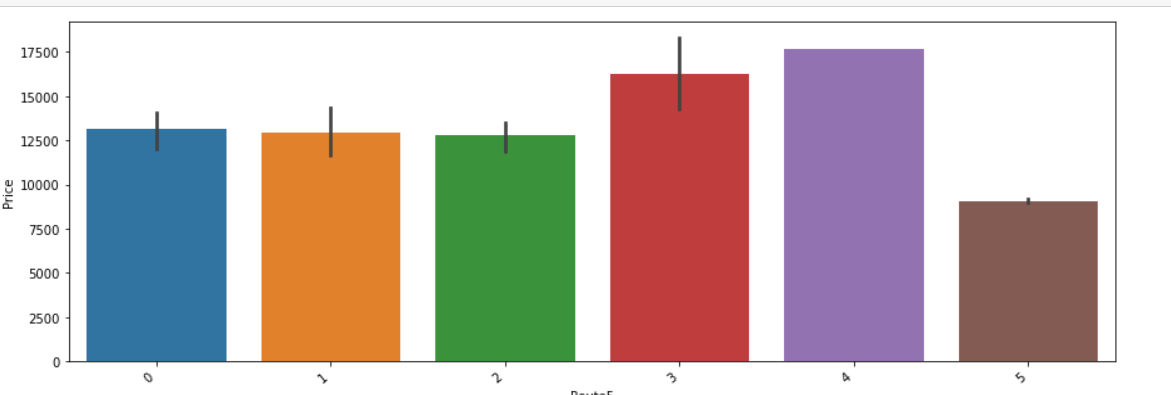
From the above plot we can observe that Route5 12th way has high price values.



From the above plot we can observe that Route5 20th way has high price values.



From the above plot we can observe that Route5 9th way has high price values.



From the above plot we can observe that Route5 4th way has high price values.

**Total Stops column:**

Encoding Total stops



Bivariate analysis of Total stops column with target variable.

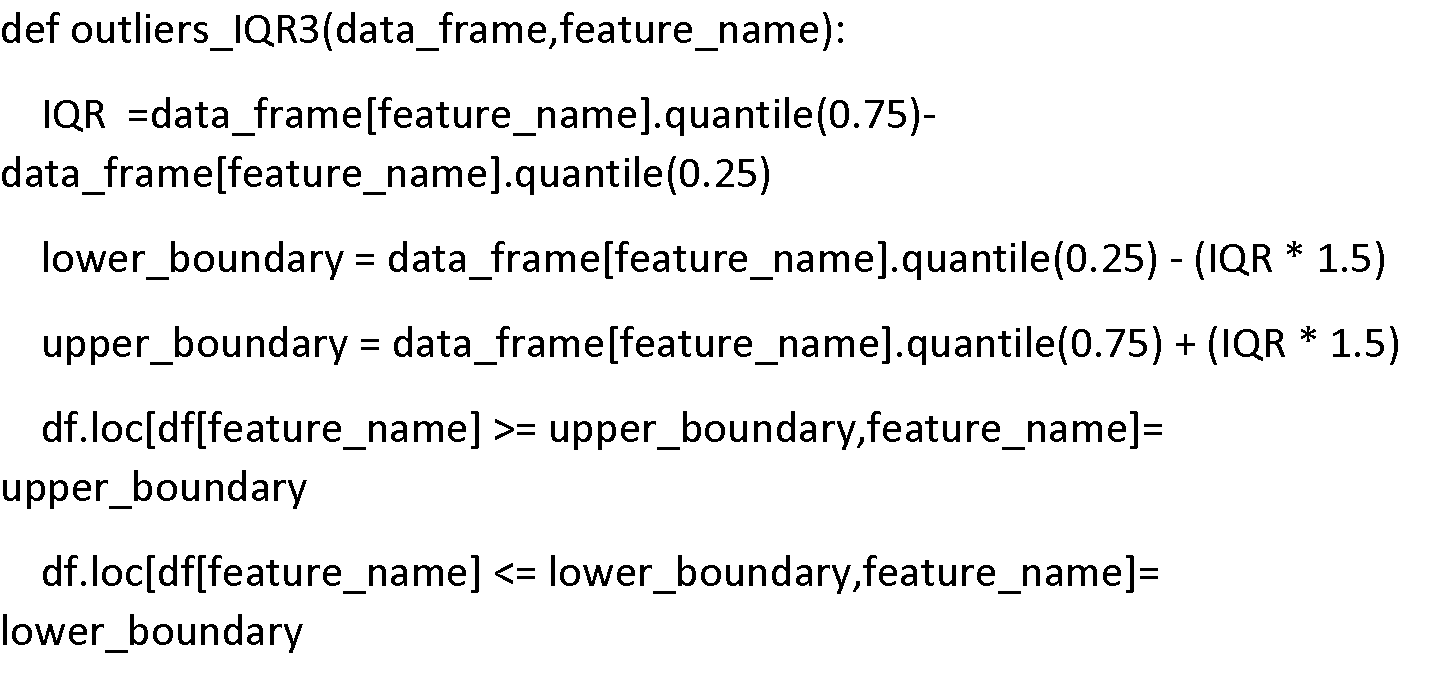
****

From the above plot we can see that if the total stops is 4, then the flight price is high.

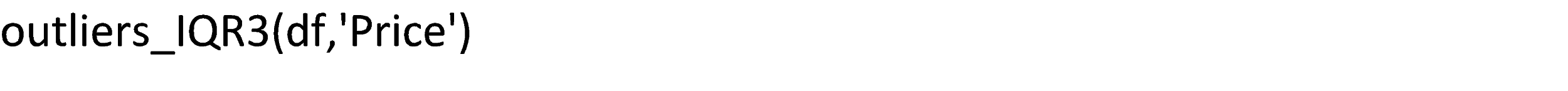
**Outlier Removal for Target column:**

Since we saw there are some outliers in Price using the visualization method, now let’s remove them.

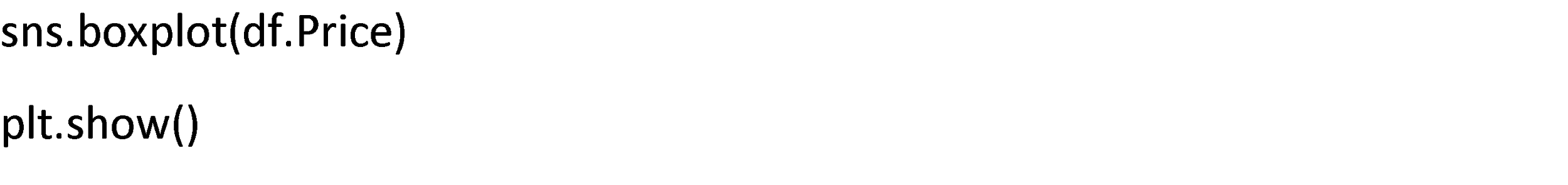
Function of outlier Removal when data is not normal distributed or Right skewed:

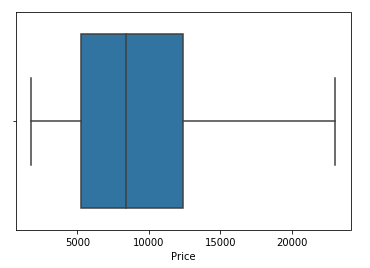


Calling the function to remove outliers:



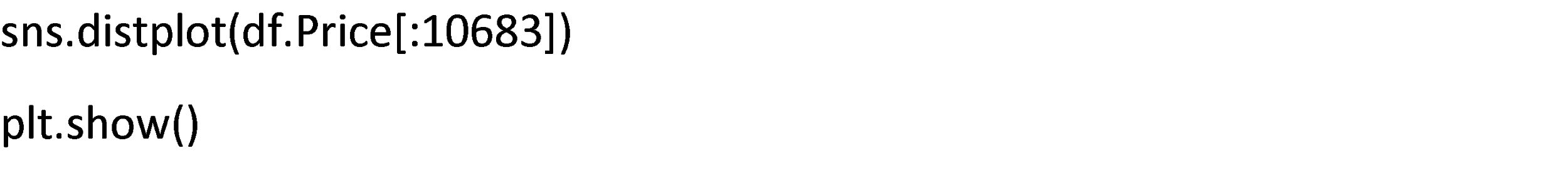
Now let’s check for outliers using box plot:

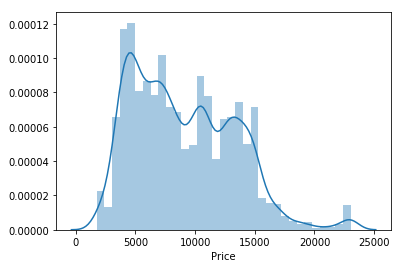




Outliers have been completely removed from our target column.

Let’s check the distribution of the target column:

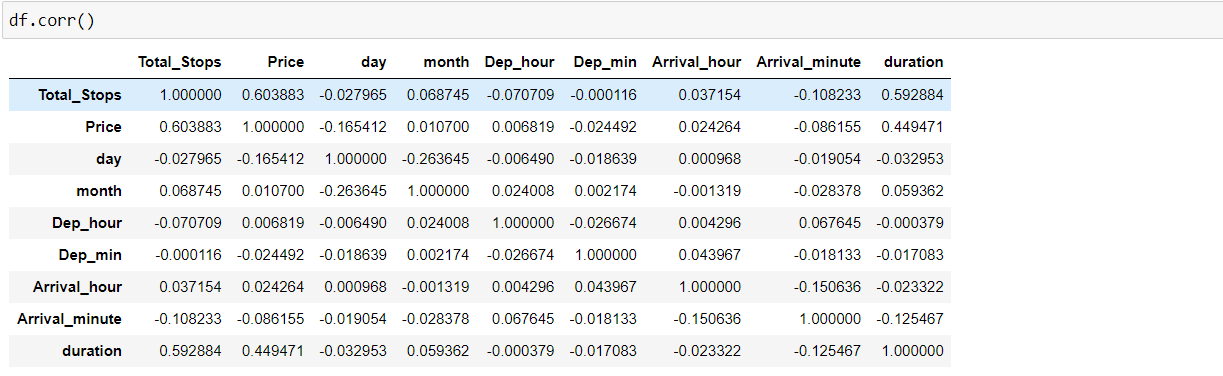




Now our target columns is almost normal distributed.

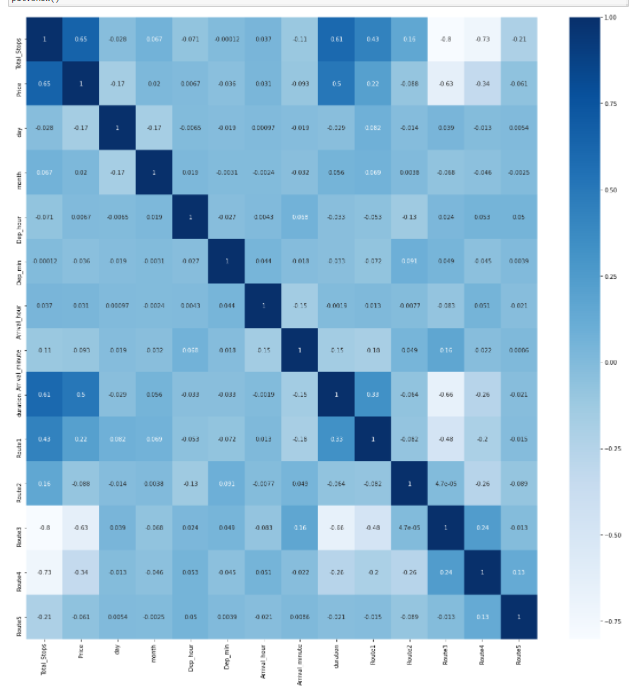
**Correlation:**

Checking correlation to find out the relationship between the features and target variable.



Visualizing the correlation using heatmap.

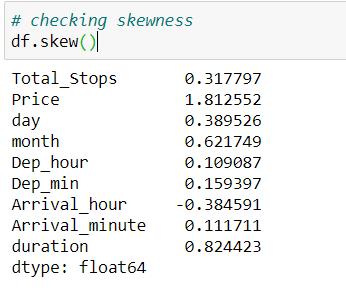




From the above heatmap we can say that price is highly positively correlated with Total stops and highly negatively correlated with Route 2 and Route3.

**skewness:**

Checking the skew of the features as show in below figure



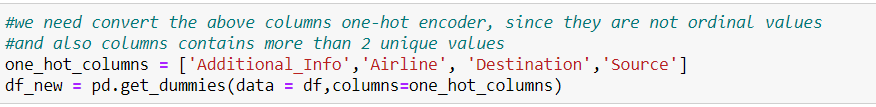
In the above figure, month and duration are having high skewness, so we performed Power transformation on those columns to reduce skewness.

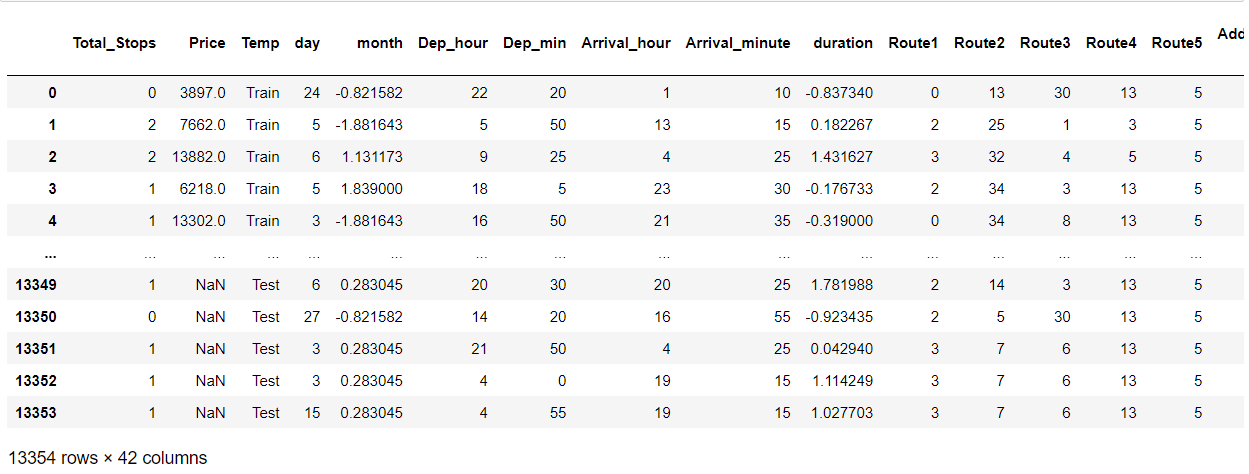


The code in the above figure is used to reduce the skewness of month and duration features by using Power Transformation.

**Encoding the Object columns:**

We need to convert all above columns using a one-hot encoder, since they are not ordinal values and also columns contain more than 2 unique values.





Now we have 42columns after one-hot encoding of object columns.

**Scaling the Input data:**

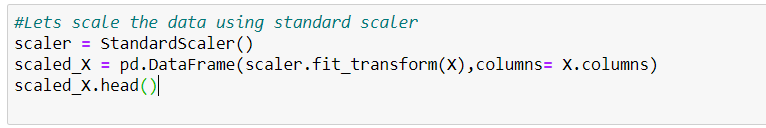
**Standard Scalar:**

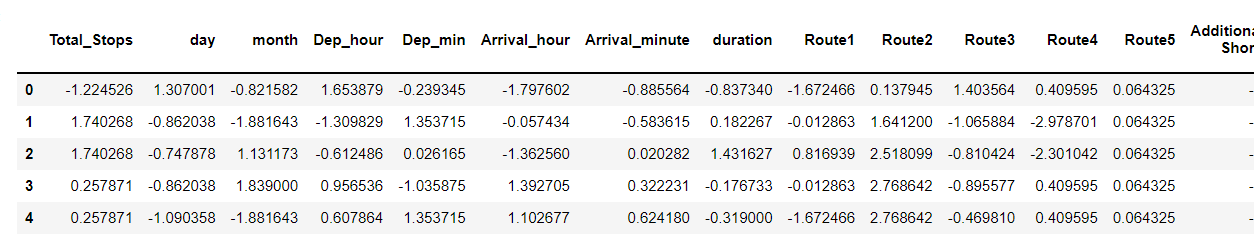
The idea behind Standard Scalar is that it will transform your data such that its distribution will have a mean value 0 and standard deviation of 1.

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform.

Standardization of a dataset is a common requirement for many machine learning estimators. They might behave badly if the individual features do not more or less look like standard normally distributed data.

Let’s scale the data using standard scalar:





Now our data is scaled using standard scalar.

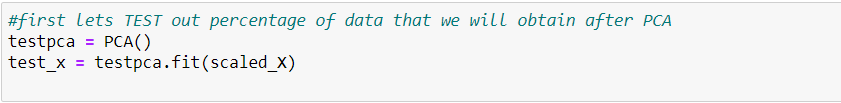
**PCA (Principal Component Analysis):**

Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation which converts a set of correlated variables to a set of uncorrelated variables. PCA is a most widely used tool in exploratory data analysis and in machine learning for predictive models.

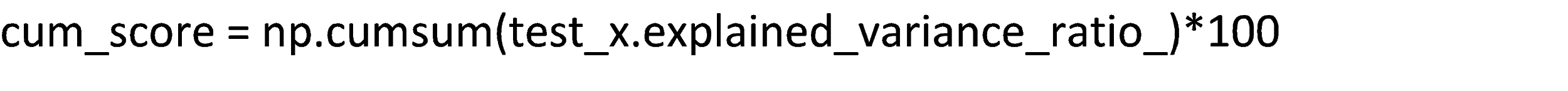
We need to use PCA because we have 44 columns in this dataset.

We should always use **scaled data** for performing the PCA.

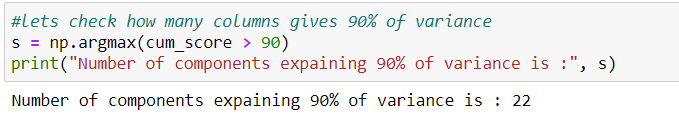
First let’s check out the percentage of data that we will obtain after PCA.



Now let’s get a cumulative score.

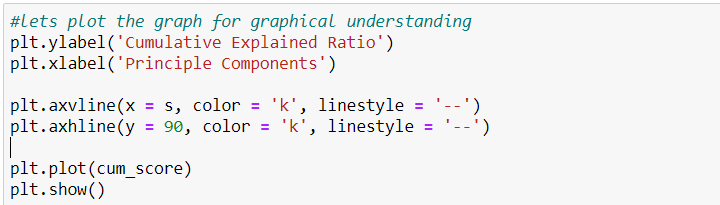


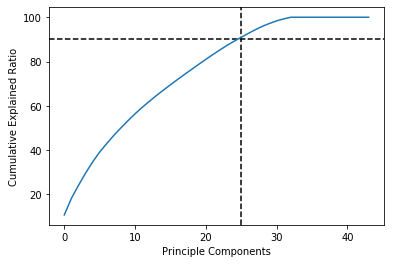
Let’s check how many columns give 90% of variance.

So let’s take 90% data which is equivalent to 22 columns.

From 40 columns, we reduced it to 22 columns using PCA with 90% of data intact.

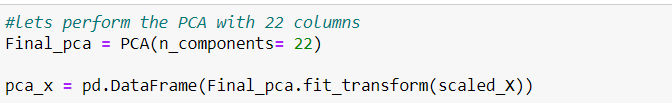
Let’s plot the graph for graphical understanding.

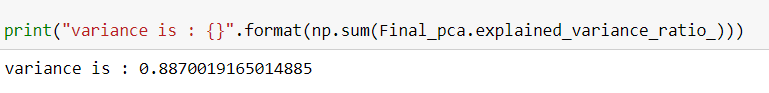


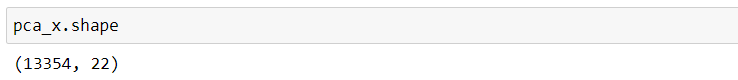


From the above plot we can see that for 22 columns 90% of data will be retrieved.

Let’s perform the PCA with 22 columns.







Now we have 25 columns and 13,354 rows with a variance ratio of 88.70 percent. Data Preprocessing is now completed.

**Pre-Processing Pipeline**

Below are the steps performed in Data Pre-Processing.

**1 .Dropping unnecessary columns:** Dropped some unnecessary columns with proper explanation.

**2. Feature Extraction:** Extracted new column from duration, Arrival date and departure date.

3. Converted Arrival columns and Departure column from float Data Type to object Data Type based on the part of the day.

**4. Duplicate Values:** Some columns had duplicate values, we combined them with proper explanations.

**5. Outliers Removal:** Removed the outliers in the Target column.

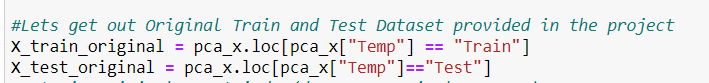
**6. Encoding:** Encoded the object columns to int DataTypes using one hot encoding.

**8. Data Scaling:** Performed data scaling using standard scalar. Scaling should be performed before PCA.

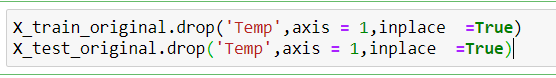
**9. PCA:** Performed PCA (principal component analysis) for dimensionality reduction.

**Let’s split our original Train and Test data which we combined for data pre-processing.**

We created a temporary column called temp, so that it will be easy to get the original train and test dataset.



Now let’s drop the temporary column from our dataset.



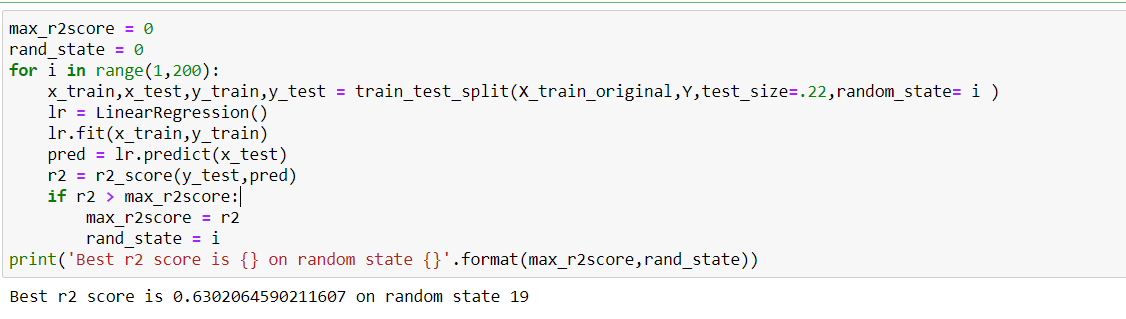
Now we got our original Train and Test data.

**Building Machine Learning Models**

Before building the model, let’s get our best random state using logistic regression.

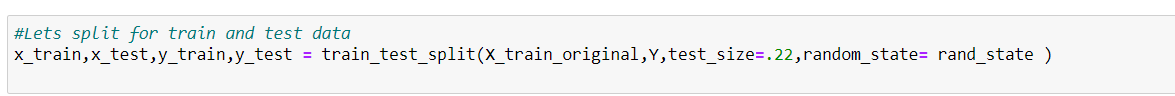
**Why Random State?**

Random state ensures that the splits that you generate are reproducible. Scikit-learn uses random permutations to generate the splits. The random state that you provide is used as a seed to the random number generator. This ensures that the random numbers are generated in the same order.



We got our best random state as: 19

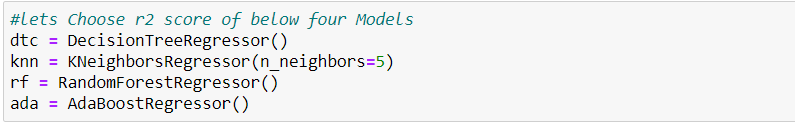
Let’s split for train and test data with the random state obtained.



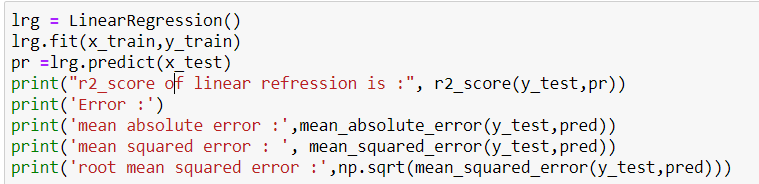
**Model selection:**

Model selection is the process of selecting one final machine learning model from among a collection of candidate machine learning models for a training dataset. Model selection is a process that can be applied both across different types of models (e.g. logistic regression, SVM, KNN, etc.)

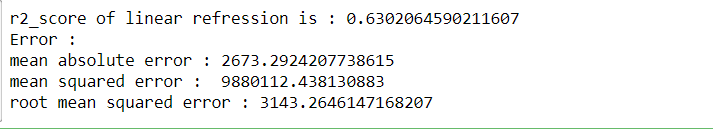
Let’s train and test our dataset among the below Models.



**Linear Regression:**

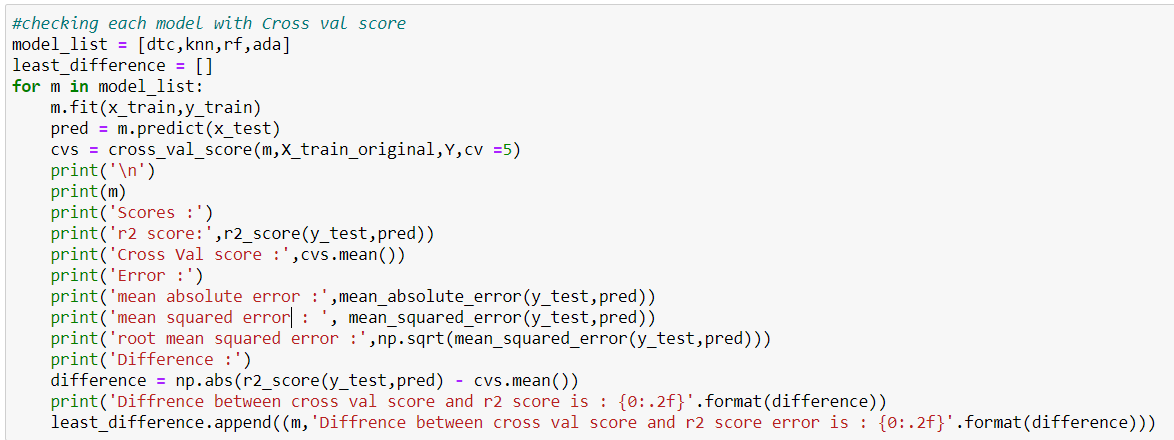
****

**Output for Linear Regression:**

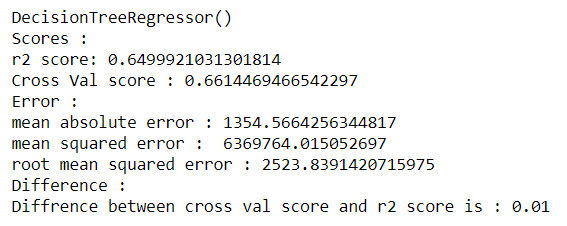
****

R2 score for linear regression is very low, so let’s check out other models.

Now let’s train and test all the other models using below code.

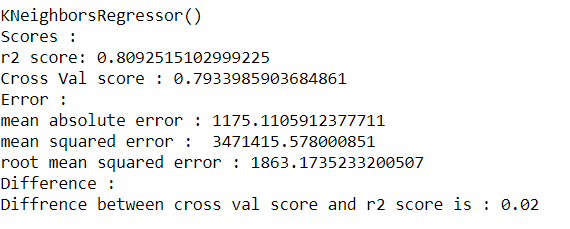


**Decision Tree Regressor Output:**

****

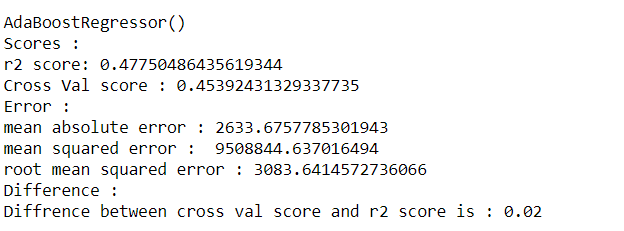
From the above output we can see that for Decision Tree R2 score is 64% and the cross validation score is also 66% which means the model is not over-fitted since the difference between cross validation score and r2 score is 0.01.

**KNeighborsRegressor Output:**

****

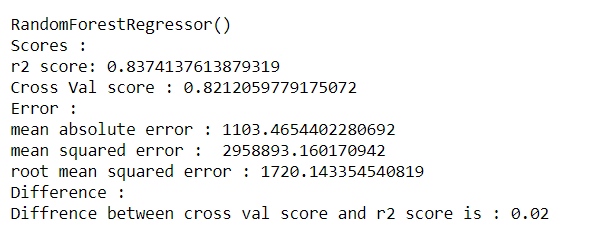
From the above output we can see that KNN R2 score is 80% and the cross validation score is 79% which means the model is not over-fitted since there is 0.02 difference between cross validation score and r2 score.

**AdaBoost Regressor Output:**

****

From the above output we can see that for Adaboost Regressor’s R2 score is 47% and the cross validation score is 45% which means the model is not over-fitted since there is 0.02 difference between cross validation score and r2 score.

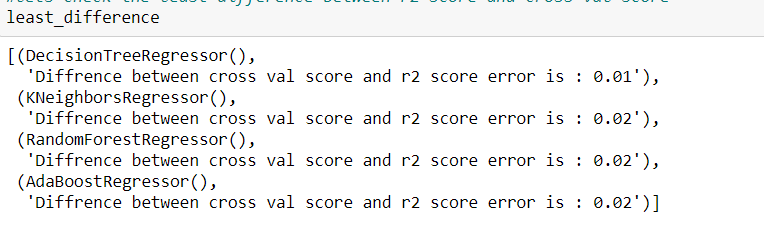
**Random Forest Regressor Output:**

****

From the above output we can see that for Random Forest Regressor R2 score is 83% and the cross validation score is 82% which means the model is not over-fitted since there is 0.02 difference between cross validation score and r2 score. R2 score is also high for this model.

**Let’s check the least difference between cross validation score and R2 score of all the models:**

In the code I created a list and appended the least difference between cross Val score and R2 score of each model.



We see that for both Decision tree and Random forest have less difference between cross validation score and r2 score. Since random forest has a 83 percent R2 score which is more than decision tree.

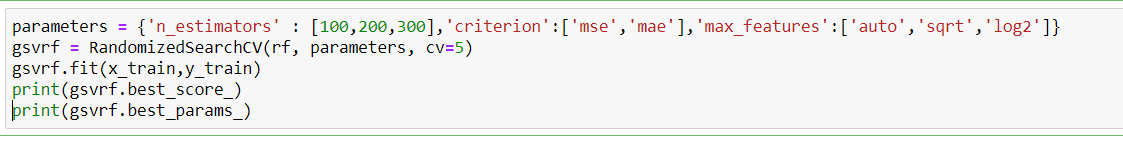
So let’s choose a random forest (83%) model for hyper parameter tuning since this model has the highest R2 score among all.

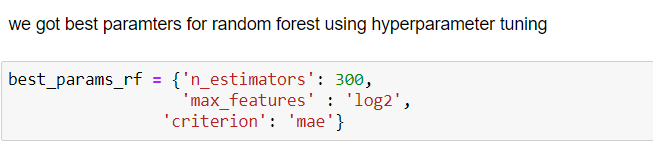
**Hyper Parameter Tuning:**

In machine learning, hyper parameter optimization or tuning is the problem of choosing a set of optimal hyper parameters for a learning algorithm. A hyper parameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.

**Hyper Parameter Tuning for Random Forest:**

Let’s create the parameter list to pass in a RandomisedSearchCV.

****

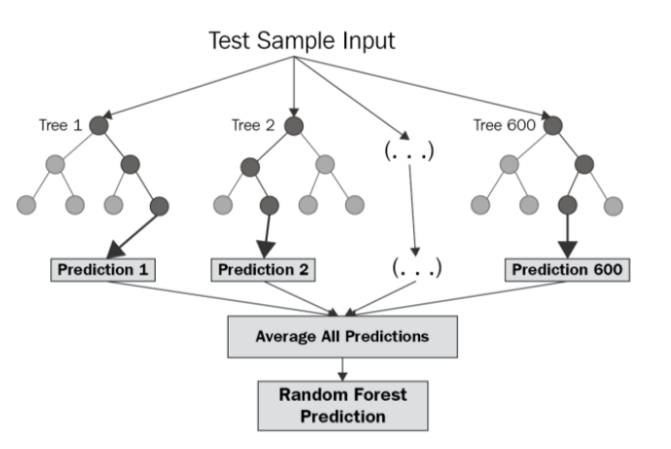
****

We got the best parameters for random forest using hyper parameter tuning.

**Creating a Final Model as Random Forest Regressor:**

**What is a Random Forest Regressor?**

Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.

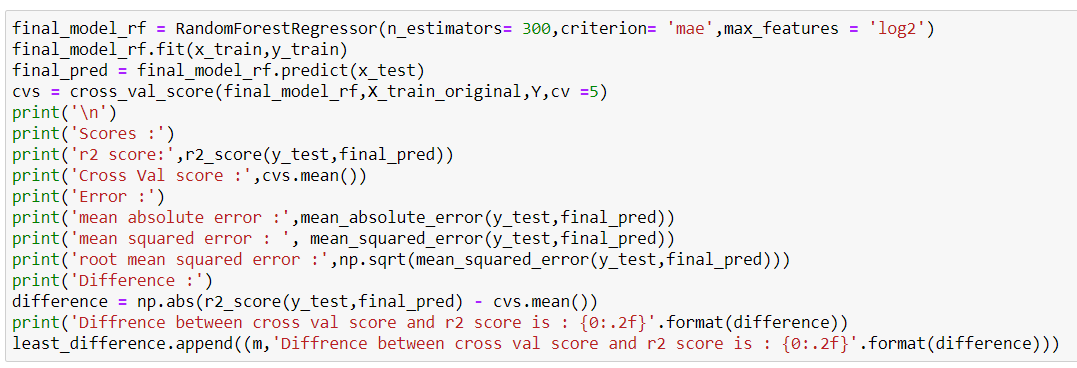


The diagram above shows the structure of a Random Forest. We can notice that the trees run in parallel with no interaction amongst them. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees. To get a better understanding of the Random Forest algorithm, let’s walk through the steps:

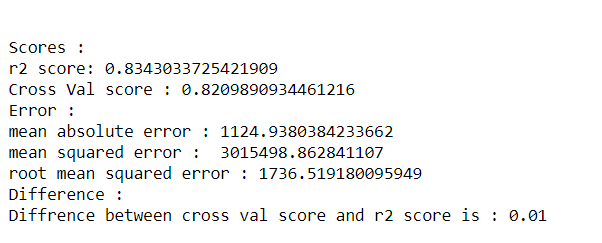
1. Pick at random *k* data points from the training set.
2. Build a decision tree associated to these *k*data points.
3. Choose the number *N*of trees you want to build and repeat steps 1 and 2.
4. For a new data point, make each one of your *N*-tree trees predict the value of *y* for the data point in question and assign the new data point to the average across all of the predicted *y*values.

A Random Forest Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationships. Disadvantages, however, include the following: there is no interpretability, overfitting may easily occur, we must choose the number of trees to include in the model.

Let’s create a random forest model with the best parameters obtained from Hyper Parameter tuning.

****

**Output of Random Forest with best parameters obtained from hyper parameter tuning:**

****

Now our r2 score is also 83% after hyper parameter tuning and the difference between cross validation score and r2 score is also 0.01 which means no over fitting.

**Metrics of our Final Model (Random Forest):**

**R2 Score:** R-squared is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model.

Best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y, disregarding the input features, would get a R2 score of 0.0.

We obtained the r2 score of 83 % which is very good!

**Root Mean Squared Error (RMSE):**

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points. RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values. Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit. Lower values of RMSE indicate better fit.

The Root Mean Squared of this model is 1736 which is quite high.

**Mean Squared Error:**

Mean squared error is the average of the squared error that is used as the loss function for least squares regression. It is the sum, over all the data points, of the square of the difference between the predicted and actual target variables, divided by the number of data points.

Mean squared error of this model is 3015498 which is quite high.

**Mean Absolute Error:**

In the context of machine learning, absolute error refers to the magnitude of difference between the prediction of an observation and the true value of that observation. MAE takes the average of absolute errors for a group of predictions and observations as a measurement of the magnitude of errors for the entire group.

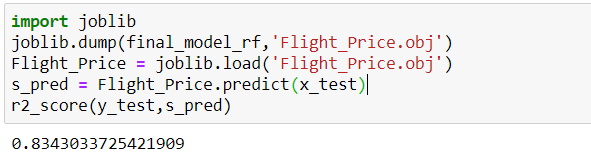
Mean Absolute error of this model is 1124.

**Saving the Final model:**

Serialization using joblib.

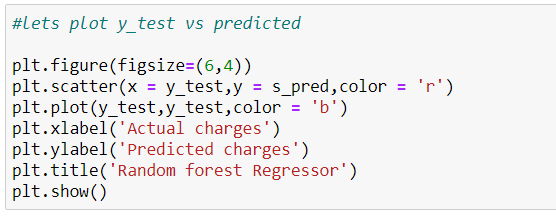
**Pickled model as a file using joblib:** Joblib is the replacement of pickle as it is more efficient on objects that carry large numpy arrays. These functions also accept file-like objects instead of filenames.

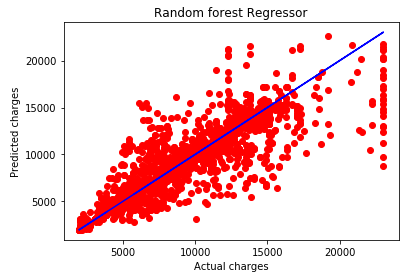
**joblib.dump** to serialize an object hierarchy.



We have an R2 score of almost 83% percent.

**Let’s plot y\_test vs predicted:**

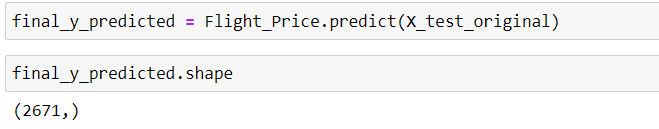
****

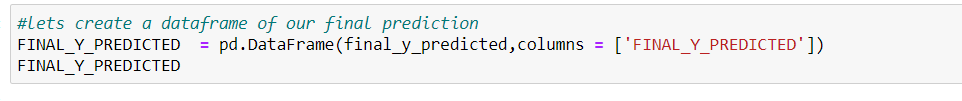
****

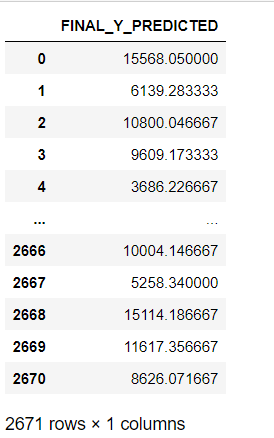
We can see that values are very close to the line.

**Prediction of our Original X\_test:**

Let’s predict the X\_test\_origanl, we don't have y\_test\_original values to compare.







**Conclusion:**

We started with importing the required libraries, loading the dataset, data exploration, checked for dimensions, missing data and learned which features are important. During this process we used seaborn and matplotlib to do the visualizations. During the data pre-processing part, we computed missing values, converted features into numeric ones, grouped values into categories and created a few new features.

After converting the categorical columns using one-hot encoding, we reduced the dimensionality of the dataset using principal component analysis with 90 percent of data intact.

Afterwards we started training 5 different machine learning models, picked one of them (random forest) and applied cross validation on it. Then we discussed how random forest works, took a look at the importance it assigns to the different features and tuned its performance through optimizing its hyper parameter values. Lastly, we looked at its R2 score, Root Mean squared error, Mean squared error and mean absolute error.

We got the 83% R2 score for RandomForest Regressor and it is not over-fitted and there is still room for improvement, like doing a more extensive feature engineering, by comparing and plotting the features against each other and identifying and removing the noisy features. The values of R-squared obtained from the algorithm give the accuracy of the model. In the future, if more data could be accessed such as the current availability of seats, the predicted results will be more accurate.