

**Ain Shams University**

**Faculty of Computer and Information Science**

**Scientific Computing department**

**Ain shams university**

**Faculty of computer and information science**

**Bioinformatics department**

**Project Title**

**Airline Ticket Price Prediction**

**By**

|  |  |  |
| --- | --- | --- |
| **Name** | **ID** | **Section** |
| **Nour Mohamed Hussein Kamaly** | **20191700701** | **5** |
| **Nourhan Abdel-Karim Khalaf Abdel-Hafez** | **20191700716** | **5** |
| **Ahmed Mohammed Samy Mohammed** | **20191700065** | **1** |
| **Mohammed Nour-Elden Abbas Ismael** | **20191700583** | **4** |
| **Abdul-Rahman Sayed Ali Mohammed** | **20191700339** | **3** |

**Under the supervision of**

**Dr. Dina Khattab**

**Scientific Computing Department,**

**Faculty of computer and Information Science**

**Ain Shams University**

We can find data here [Airline-Price](../Data/airline-price-prediction.csv)

We can find the description of data here [Data-Description](../Data/Data-Description.txt)

**Welcome To Our Web Application (Airline Ticket Price Prediction)**

Graphical user interface

Description automatically generated

**Inputs:  
(Airline – Departure time – Arrival time – Type of journey – Source – Destination –   
Flight Month – Number of stops – Week day of flight – Distance between countries)**

**Output:**

**Prediction of ticket price**

**Data Analysis phase**

**We can suggest some questions to analysis this data such that:**

Q1-What is the airline that has the most expensive ticket (business and economy)?

Table, Excel

Description automatically generatedAns:

the most expensive ticket for business class is: Vistara Airline

the most expensive ticket for economy class is: India Airline

**Table 1**

Q2-What is the airline that has the cheapest ticket (business and economy)?

Table

Description automatically generatedAns:

the most expensive ticket for business class is: Vistara Airline

the most expensive ticket for economy class is:

{Asia Airline, Go First Airline, Indigo Airline}

**Table 2**

Q3-What is the most expensive 'business type' ticket in each airline and what is its duration and date?

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| Airline | The Most Expensive price | Duration | Date |
| Vistara | 99677 | 36 | February |
| Air India | 89257 | 35 | February |

Q4-What is the cheapest 'business type' ticket in each airline and what is its duration and date?

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| Airline | Cheapest price | Duration | Date |
| Vistara | 100 | 36 | February |
| Air India | 12000 | 35 | February |
| Air India | 12000 | 36 | March |

Q5-What is the most expensive 'economy type' ticket in each airline and what is its duration and date?

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| Airline | The Most Expensive price | Duration | Date |
| Air Asia | 31917 | 19 | February |
| Go First | 31773 | 20 | February |
| Indigo | 31952 | 15 | February |
| SpiceJet | 34158 | 28 | February |
| Air India | 42349 | 45 | February |
| Vistara | 37646 | 41 | February |
| Star Air | 9682 | 2 | March |
| TruJet | 4844 | 3 | February |

Q6-What is the cheapest 'economy type' ticket in each airline and what is its duration and date?

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| Airline | Cheapest price | Duration | Date |
| Air Asia | 1105 | 19 | February |
| Air Asia | 1105 | 18 | March |
| Go First | 1105 | 20 | February |
| Go First | 1105 | 19 | March |
| Indigo | 1105 | 14 | March |
| SpiceJet | 1106 | 28 | March |
| Air India | 1526 | 42 | March |
| Vistara | 1714 | 41 | February |
| Vistara | 1714 | 36 | March |
| Star Air | 2000 | 2 | March |
| TruJet | 3124 | 3 | February |
| TruJet | 3124 | 3 | March |

Chart, pie chart

Description automatically generatedTable

Description automatically generated

**Figure 1**

**Table 3**

Q7-How much did the longest flight cost in business and economy types?

Table

Description automatically generatedAns:

the longest flight in business type cost: 52446.90

the longest flight in economy type cost: 6572.47

**Table 4**

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generatedQ8-What time of the day has the most expensive (business and economy) ticket in each airline?

**Figure 2**

Ans:

the time of the day that have the most expensive business ticket is: 12:00

the time of the day that have the most expensive economy ticket is: 20:00

Table

Description automatically generatedQ9-What is the day-average ticket price (business and economy) in each day of the week?

Ans:

**Table 5**

Text

Description automatically generated with medium confidenceQ10-What is the month-average ticket price (business and economy) in each month of the year?

Ans:

**Table 6**

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generatedQ11-What is the most departure and arrival time for planes during the day?

**Figure 3**

Ans:

the most departure time at 20:00

the most arrival time at 10:00

Q12-Who is the most frequent source which takes off planes from it?

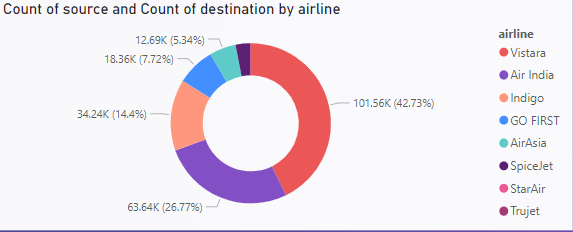
Ans:

the most frequent source which takes off planes from is: Vistara Airline

Q13-Who is the most frequent destination for planes?

Ans:

the most frequent destination which takes off planes from is: Vistara Airline



**Figure 4**

Q14-What is the most popular day and month for flight?

Ans:

the most popular day is Monday

the most popular Month is March

Chart, line chart

Description automatically generated

**Figure 5**

Q15-How much did the longest flight take?

Ans:

The longest Flight takes 50 hours

Graphical user interface, application

Description automatically generated

**Figure 6**

Q16-What is the most airline used?

Chart, funnel chart

Description automatically generatedAns:

The most airline used Vistara Airline

**Figure 7**

Q17-What is the most number of stops during the flight?

Chart, pie chart

Description automatically generatedAns:

The most number of stops during the flight is 2 hours

Q18-What is the least frequent source for airplanes? (Look Figure 4)

Ans:

The Least frequent source for airplanes is Trujet Airplane

Q19-What is the least frequent destination? (Look Figure 4)

Ans:

The Least frequent Destination for airplanes is Trujet Airplane

Q20-What is the busiest duration of the day for flights? (Look Figure 6)

Ans:

the busiest duration of the day for flights is 2 Hours

Q21-What is the Freest duration of the day for flights? (Look Figure 6)

Ans:

the Freest duration of the day for flights are: 42,46,48,45,47,50 Hours

**Report for our dataset and many variables in it**

Graphical user interface, text, application, email

Description automatically generated

Chart

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

Chart, box and whisker chart

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

A picture containing graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

Graphical user interface

Description automatically generated

Chart

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated

Table

Description automatically generated with medium confidence

A picture containing table

Description automatically generated

**Preprocessing phase**

1. Preprocessing techniques on features:

* Date :
* Format\_dates() : some dates had ‘/’ , others had ‘-’, the character needed to be unified, to be able to split with it later on, ‘-’ was chosen to replace all backslashes ‘/’. This is implemented by iterating over all the date column and replacing ‘-’ with ‘/’ using .replace() function.
* Extract\_day\_month () : extracting the month and the day of the month by iterating over the formatted date column and splitting by the dash ‘-’, adding the first item of the list to flight day feature, and adding the second item of the list to the flight month feature. Based on the month, the format of the formatted date is changed (could have been done by using pandas datetime but it swapped the month with the day so I did it manually).
* Extract\_weekday () : extracting weekday by converting the formatted date to datetime using pandas then calling function .day\_name() and filling up week day of flight feature with the result.
* Route :
* Split\_route() : the route is originally a dictionary, but it is stored as a string object in the data frame, by using the abstract syntax library’s function literal evaluation, this function returns an object of the datatype it finds in the string, so by converting it to a dictionary, I can access the source and the destination easily and put these values in new columns.
* Stop:
* Split\_num\_of\_stops() :the number of stops can be known from the first few characters, so by slicing the string up to a specific character (0 -> 7 for non-stop, 0 - > 5 for 1-stop, else it’s more than 2 stops ) we can use the integer value of number of stops to represent this feature.
* Find\_where\_is\_the\_stop() : if the number of stops is 1, an extra piece of information may be provided, which is where the stop was, this string would look like 1-stop\n\t\t\t\t\t Mumbai\n\t\t\t\t, so to get the city alone I can split the whole string by the space first, then split the second item of the returned list by the endline, and my desired city would be the first element of the returned list if the length of the list exceeds 1 (if the city information is provided).
* Price:
* Fix\_price\_format(): the price is stored in the data frame as a string object due to the presence of a comma “50,000”, so since this a string, I splitted this string by the said comma, concatenated the results of splitting, and returned the integer value.
* Time taken:
* Calculate\_time\_taken() : the time taken is stored as the number of hours and the number of minutes the flight took in a string object (example : “10h 30m”) so we can split by the space, obtain the hours and the minutes separately, and since we chose to use the hours only, to make use of the minutes, if the number of the minutes exceeded 40 (our chosen threshold ) we add 1 to the hours, there are 2 corner cases : there may not be any minutes (“7h”), and the number of minutes may be added to the hour (“1.03h m”), and both cases are handled in the code.
* Arrival and departure time:
* Categorize\_time(): there is a 100% dependency between arrival time, departure time and time taken, and this dependency can be expressed by: time taken = arrival time - departure time.

Since using the hours themselves in arrival time and departure time will cause redundancy during training, we can extract the time of the day the flight departed and arrived as some times maybe cheaper or more expensive, this is done by categorizing the time intervals to : early morning, morning, afternoon, evening.

* There are many approaches to handle categorical data, two were chosen:
* One hot encoding the categories, this ensures that the model is not biased to a larger label.
* Arranging the categories by the average of the price, giving categories of smaller averages a small number, and higher averages a high number (target encoding using price average done manually).

**This resulted in 2 datasets and training was done on both.**

1. Preprocessing techniques on the dataset:

* Outlier detection using interquartile range:
* In some problems, outliers are considered noise to the data, and they skew the predictive line, so by removing them, the model’s predictions become better, the interquartile method is a statistical analysis method that provides a range of numbers, if the value isn’t between that range then it counts as an outlier, and that observation is then removed, this is applied to our label, the price.
* Feature balance on airline feature
* Class imbalance may occur in labels (if it’s a classification problem) or may happen if a certain category appears more than the other categories in a given feature.
* In airline feature, 5 airlines have from (0% -> 7 %) of the data, this would cause the model to overlook these categories, especially when airline Vistara alone is present in 42% of the dataset, to create some sort of feature balance and decrease the number of categories, the 5 airlines with very small percentage are gathered in one new category called “Other\_airline”.
* Transforming the data to the frequency domain:

Chart

Description automatically generated with medium confidence

Figure 9

* This figure represents the average of the prices of each day in March (March not February because observations of February started from 11/2, so the first ten day’s observations are not known), and the peaks are always decreasing over some period of time.
* This lead to the thought that we may be able to extract the frequency components contributing to this signal, and there were two ways to do this.
* Transforming the dataset to the frequency domain using the Fourier transform but we faced many problems:

1. After transforming the data, the resulted data frame consisted of complex numbers, which is not a supported datatype for model training.
2. We could extract two features from the frequency component, the magnitude and the phase shift (many decisions to be took).

This led to thought of using a transform that deals only with real numbers.

* We transformed the data to the frequency domain using the discrete cosine transform, as it uses only real numbers, and still preserves the frequency components.

**This resulted in a 3rd dataset that was also used during training**

* The purpose of creating 3 datasets was that each model could pick different patterns by seeing different parts of the datasets each leading to the same labels in the end.

1. Features engineered / extracted

* Flight day / flight month: these were extracted from the date by using pandas datetime as they might have important weights contributing to the label.
* Weekday of flight: representing days of the week, extracted from date by using pandas datetime.
* Distance between the 2 countries: the distance in kilometers between source and destination, it is then normalized to values between 0 ->1 by dividing the distance by the greatest distance found.

1. Features used / discarded

* Hypothesis testing using p value was used in feature selection, our null hypothesis is that the model is learning from the feature, so if the p value exceeded 0.05 then my null hypothesis failed, and the model is not making use of the feature.
* Features used are (p value < 0.05): type, flight month, number of stops, distance between 2 countries, airline, source, destination, Saturday, Sunday, Thursday, Tuesday, departure time, and arrival time.
* Features discarded were: flight day, number of hours taken, one stop in, Friday, Monday, Wednesday.
* The above features were discarded based on hypothesis testing, but these features were discarded as they were in the wrong format (they were fixed and given a new name): date, time taken, stop, route, price.
* Ch code was dropped as there was a 100% dependency between it and the airline as the ch code is a code for the airline, so using both would cause redundancy.
* Num code was dropped as there were so many value counts each had a low frequency between the observations.

1. Sizes of training and testing sets

* As this is a time series data, choosing a random train test would lead to data leakage as we can’t let the model train on new data and test on old or shuffled data so:

The train set consists of the first 80% of the sorted dataset (sorted by date using quicksort)

The test set consists of the last 20% of the sorted dataset (newer dates)

**Modeling Phase**

**- We use 10 models for this project and execute them on 3 datasets:**

* "XGBR": XGBRegressor(),
* "PR"  : linear\_model.PoissonRegressor()
* "HGBR": HistGradientBoostingRegressor()
* "LGBMR": ltb.LGBMRegressor()
* "LR": linear\_model.LinearRegression()
* "GBR": ensemble.GradientBoostingRegressor()
* "ETR": ensemble.ExtraTreesRegressor()
* "BR": BaggingRegressor()
* "DT": DecisionTreeRegressor()
* "RF": RandomForestRegressor()

**Gradient Boosting Regressor Model:**

The main idea behind this algorithm is to build models sequentially and these subsequent models try to reduce the errors of the previous model, The objective here is to minimize the loss function by adding weak learners using gradient descent algorithm, Decision trees are used as the weak learner in gradient boosting, Specifically, regression trees are used that output real values for splits and whose output can be added together, allowing subsequent models outputs to be added and “correct” the residuals in the predictions, Trees are constructed in a greedy manner, choosing the best split points based minimize the loss function.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart

Description automatically generated

**The results of the model on the dataset which preprocessed with target encoding technique:**

**Chart, histogram

Description automatically generated**

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

**Chart, histogram

Description automatically generated**

**Extra Trees Regressor Model:**

Extra Trees is an ensemble machine learning algorithm that combines the predictions from many decisions trees, The Extra Trees algorithm works by creating a large number of decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

**Chart, histogram

Description automatically generated**

**The results of the model on the dataset which preprocessed with target encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart, histogram

Description automatically generated

**Multiple Linear Regression Model:**

Multiple regression is a technique that can be used to analyze the relationship between a single dependent variable and several independent variables. The objective of multiple regression analysis is to use the independent variables whose values are known to predict the value of the single dependent value. Each predictor value is weighed, the weights denoting their relative contribution to the overall prediction.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

**Chart

Description automatically generated**

**The results of the model on the dataset which preprocessed with target encoding technique:**

**Chart, histogram

Description automatically generated**

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart, histogram

Description automatically generated

**Random Forest Regressor Model:**

is a Supervised Learning algorithm which uses ensemble learning method for classification and regression.

The trees in random forests are run in parallel. There is no interaction between these trees while building the trees.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Diagram

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with target encoding technique:**

Diagram

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Diagram

Description automatically generated with low confidence

**Bagging Regressor Model:**

is an ensemble meta-estimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with target encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Diagram

Description automatically generated with medium confidence

**Decision Tree Regressor Model:**

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Diagram

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with target encoding technique:**

Diagram

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Diagram

Description automatically generated

**Extreme Gradient Boosting Regressor Model:**

Gradient boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modeling problems. Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. This is a type of ensemble machine learning model referred to as boosting. Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, “gradient boosting,” as the loss gradient is minimized as the model is fit, much like a neural network. Extreme Gradient Boosting is an efficient open-source implementation of the gradient boosting algorithm. It is designed to be both computationally efficient and highly effective.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with target encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart, histogram

Description automatically generated

**Poisson Regressor Model:**

Poisson regression is a form of the generalized linear model and it is used to model count data and contingency tables.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart

Description automatically generated with medium confidence

**The results of the model on the dataset which preprocessed with target encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart

Description automatically generated

**Histogram-Based Gradient Boosting Model:**

Gradient boosting is an ensemble of decision trees algorithms. A major problem of gradient boosting is that it is slow to train the model. This is particularly a problem when using the model on large datasets with tens of thousands of examples (rows).

Training the trees that are added to the ensemble can be dramatically accelerated by discretizing (binning) the continuous input variables to a few hundred unique values. Gradient boosting ensembles that implement this technique and tailor the training algorithm around input variables under this transform are referred to as histogram-based gradient boosting ensembles.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with target encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart

Description automatically generated

**Light Gradient Boosted Machine Model:**

LightGBM extends the gradient boosting algorithm by adding a type of automatic feature selection as well as focusing on boosting examples with larger gradients. This can result in a dramatic speedup of training and improved predictive performance.

**The results of the model on the dataset which preprocessed with one hot encoding technique:**

Chart, histogram

Description automatically generated

**The results of the model on the dataset which preprocessed with target encoding technique:**

**Chart, histogram

Description automatically generated**

**The results of the model on the dataset which preprocessed with one Frequency domain encoding technique:**

Chart, histogram

Description automatically generated

-We use **Artificial Neural Network** also with **(1 Hidden Layer)** and **(128 Neurons)**:

Text

Description automatically generated

Chart, histogram

Description automatically generated

-We use also **Simple Ensemble Learning Techniques (Averaging)** for each dataset:

* For One Hot Encoded Dataset:

Models:

- **Light Gradient Boosted Machine Model**

- **Histogram-Based Gradient Boosting Model**

- **Extra Trees Regressor Model**

- **Random Forest Regressor Model**

- **Bagging Regressor Model**

Results:

Text

Description automatically generated

* For Target Label Dataset:

Models:

- **Light Gradient Boosted Machine Model**

- **Histogram-Based Gradient Boosting Model**

- **Extra Trees Regressor Model**

- **Random Forest Regressor Model**

- **Bagging Regressor Model**

- **Decision tree regressor Model**

Results:

Text

Description automatically generated

* For Frequency Domain Dataset:

Models:

- **Extra Trees Regressor Model**

- **Bagging Regressor Model**

- **Decision tree regressor Model**

Results:

Text

Description automatically generated

Conclusion

Since our label is the price predicted which is a continuous data therefore this is a linear regression problem, we started with analysis phase to know more about the problem data and found out that there is a feature representing the date we treated this problem as a time series for casting problem, so we decided to use one hot encoding technique to preprocess the data, and Target label technique too, and because the visualization graph showed that the problem is a time series casting problem we used frequency domain as the third technique to preprocess the data, in the three techniques we start try to detect the related features using some statistics methods like a p-value method as we noticed that there is no correlation between the features, so we ignore some features and normalized another, after that we start implement the models and we used more than one model to pick the best the model for our problem and create a website to make our project user friendly.