Mruganshi
B20CS014

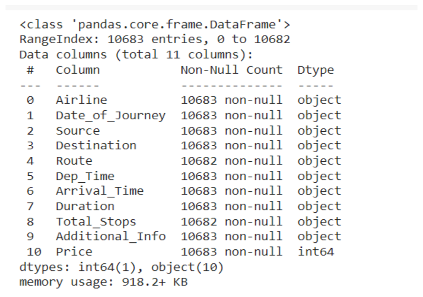
PRML Minor Project

Report

Problem Statement

We might have often heard travelers saying that flight ticket prices are so unpredictable. As data scientists, we are gonna prove that given the right data anything can be predicted. Here you will be provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities. Size of training set: 10683 records.

Data Preprocessing:



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

☐ Splitting and analysis in training and testing:

- We will split data into training and testing using the sklearn library keeping test ratio 0.7.
- After that we will analyze training and testing data by checking its number of features, name of features, number of rows, data types, number of null cells for both datasets.

☐ Handling null values

- Since there is only one row with null value we will simply drop it.

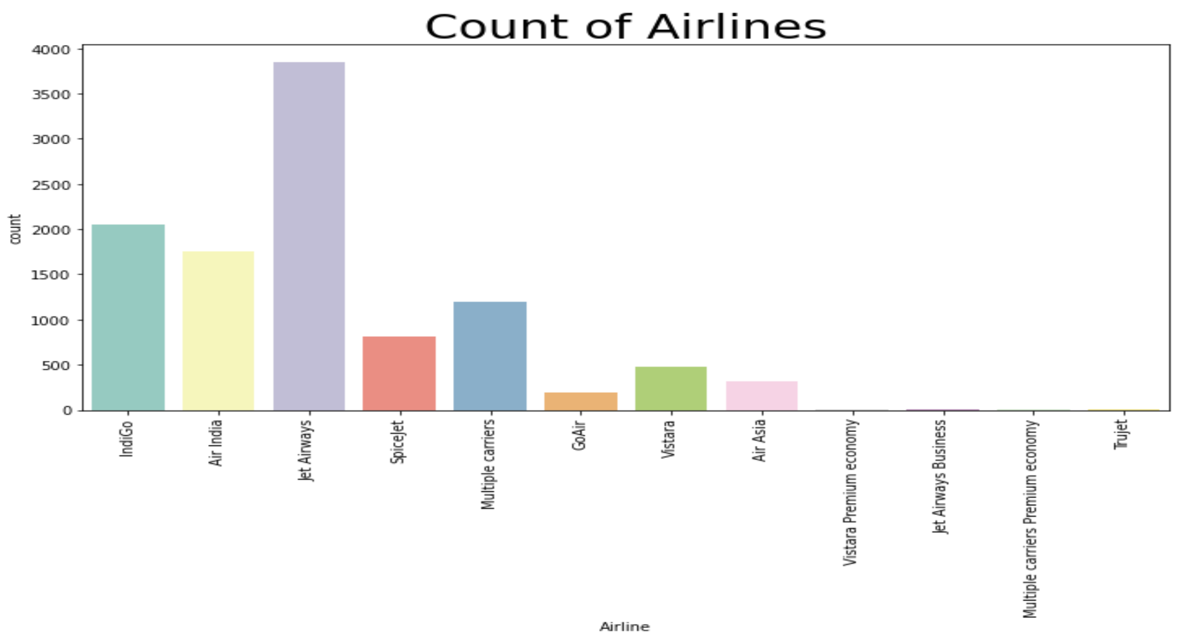
```
Original Length of Training Set :  7478
Length of Training Set after dropping NaN:  7477
```

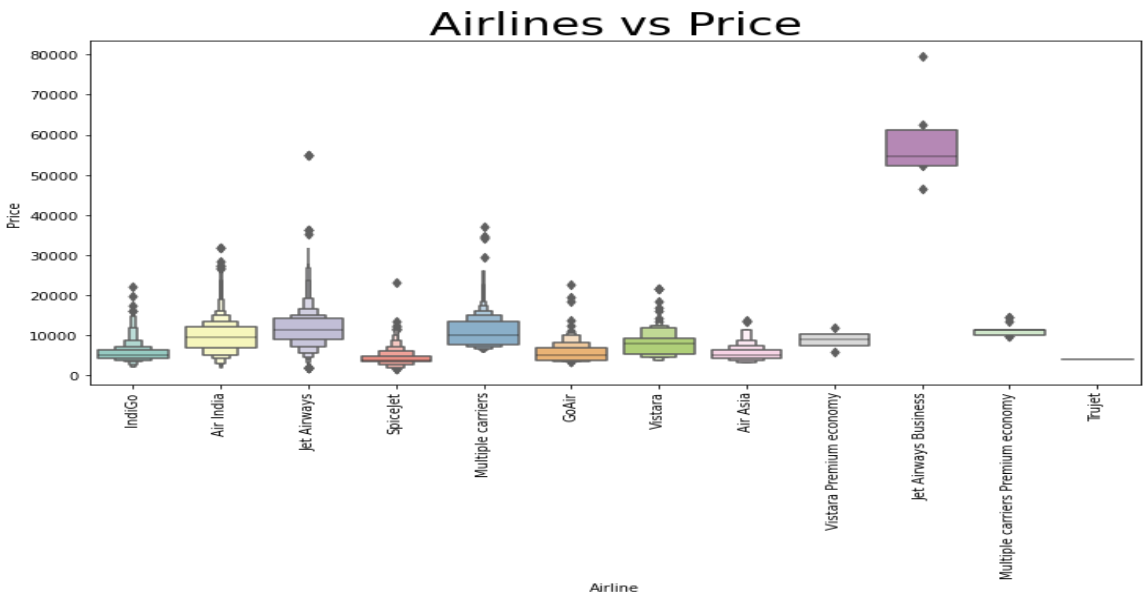
☐ Handling categorical data

- We will use label encoder to encode Airline, Source, Destination, Route, Total_Stops and Additional_Info.

☐ Visualization

In the visualization part we will count the number of occurrences according to the target column and compare column values with the target column. Below some of the plots are shown.





Exploratory Data Analysis

☐ Cleaning Journey Date

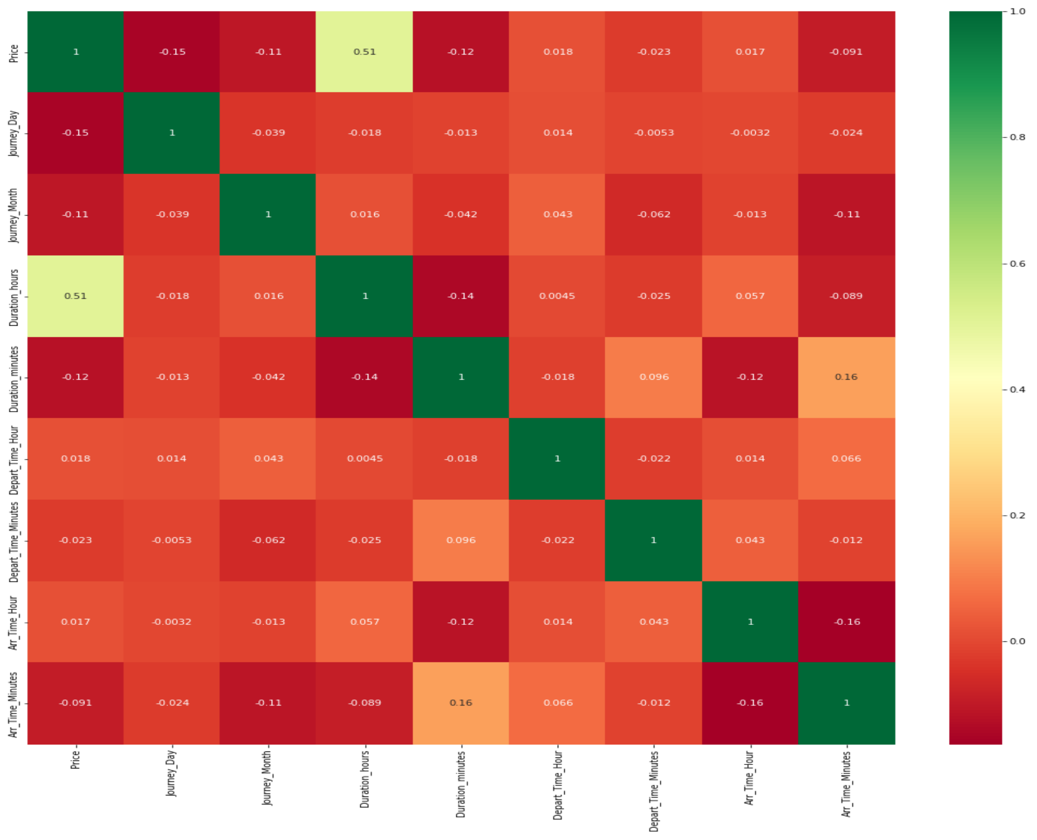
- From the description we can see that Date_of_Journey is an object data type,\ Therefore, we have to convert this data type into timestamp so as to use this column properly for prediction.
- For this we require pandas **to_datetime** to convert object data type to datetime dtype.dt.day method will extract only the day of that date and dt.month method will extract only month of that date.
- First we will convert the Date_of_Journey column into integers and after that we will drop it as it is of no use.

☐ Cleaning Departure and Arrival Times

- Similar to Journey date we will extract values for Dep_time. First we will extract hours then minutes and after that we will drop Dep_time as it will be of no use. And the same technique will be used for Arrival_time.

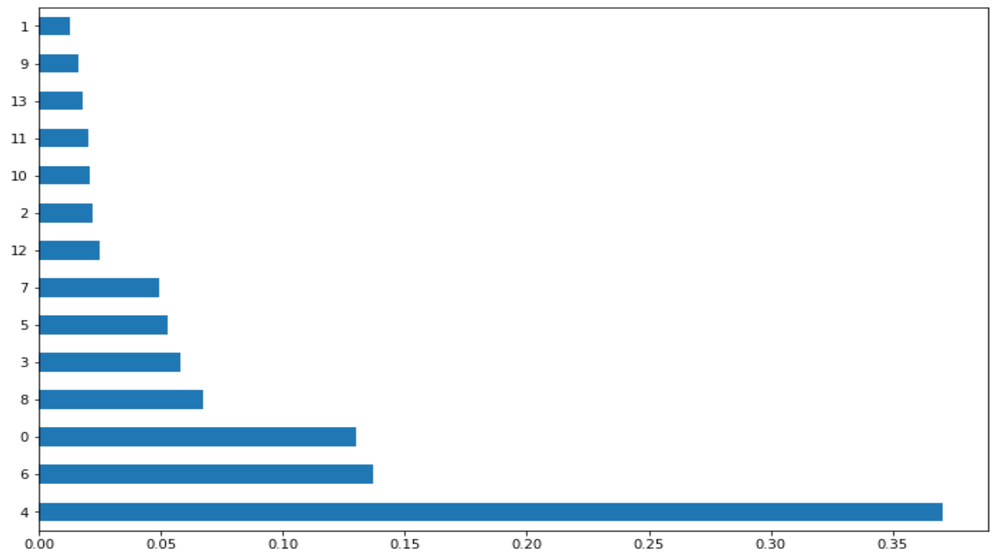
Feature Selection

☐ correlation between Independent and dependent attributes



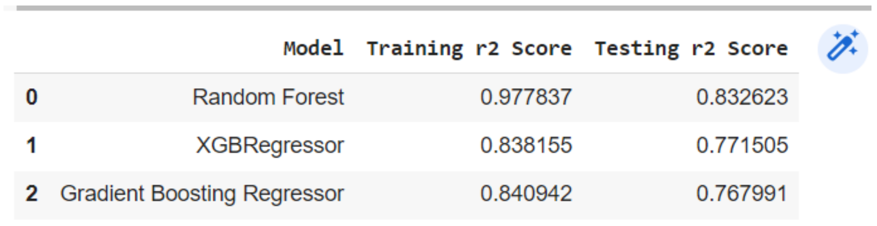
☐ Important feature using ExtraTreesRegressor and its visualization

- We will use the ExtraTreeRegressor sklearn library for feature selection and below its visualization is shown and according to that we will select features.



Fitting models

- The goal in this step is to develop a benchmark model that serves us as a baseline, upon which we will measure the performance of a better and more tuned algorithm.
- We are using different Regression Techniques and comparing them to see which algorithm is giving better performance then other and see how our model is predicting.
- So according to that we will implement **Random Forest Regressor, XGBoost Regressor, Gradient Boosting Regressor** and final accuracies for training and testing both datasets using all three datasets are shown below.

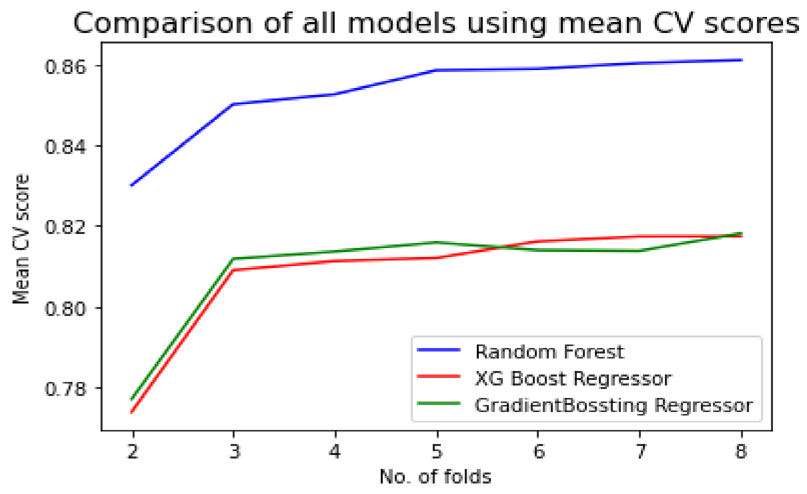


	Model	Training r2 Score	Testing r2 Score
0	Random Forest	0.977837	0.832623
1	XGBRegressor	0.838155	0.771505
2	Gradient Boosting Regressor	0.840942	0.767991

- Random Forest model gives us the best accuracy, with an R2 score of 87%, but the model is overfitting on train data.We will try to tune this model to check if we can remove overfitting.

Cross Validation

- We perform the cross validation of our model to check if the model has any overfitting issue, by checking the ability of the model to make predictions on new data, using k-folds.
- We test the five fold cross validation for Random forest and Gradient Boosting and XGBoost Regressor below its visualization is shown.



- The Random Forest Regressor provides us a cross validation score of 86%, and gradient boosting regressor and xgboost regressor gives a score of 82%. We will hypertune both the models to check if our accuracy improves.

Hypertuning the model

- By creating a grid of parameters and trying all the combinations to compare which combination gave the best results. We apply grid search on our model which gets the best accuracy.

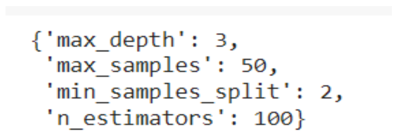
☐ Random Forest Regressor

- Hyper parameters are ,

```
{'max_depth': None,
 'max_samples': 1000,
 'min_samples_split': 2,
 'n_estimators': 50}
```

☐ XGBoost Regressor

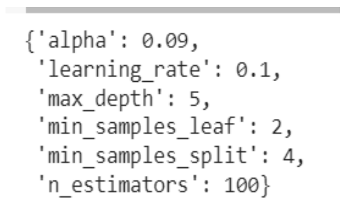
- Hyper parameters are ,



```
{'max_depth': 3,
 'max_samples': 50,
 'min_samples_split': 2,
 'n_estimators': 100}
```

☐ **Gradient Boosting Regressor**

- Hyper parameters are ,

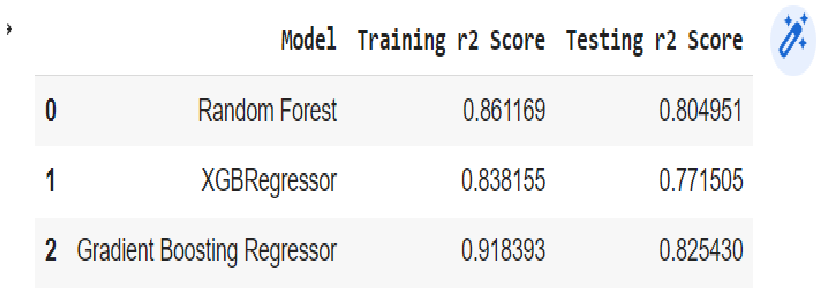


```
{'alpha': 0.09,
 'learning_rate': 0.1,
 'max_depth': 5,
 'min_samples_leaf': 2,
 'min_samples_split': 4,
 'n_estimators': 100}
```

- Testing r2 score after training with these hyper parameters is `0.8254299507930931`

Conclusion

- After training with hyperparameters final r2 scores for all models

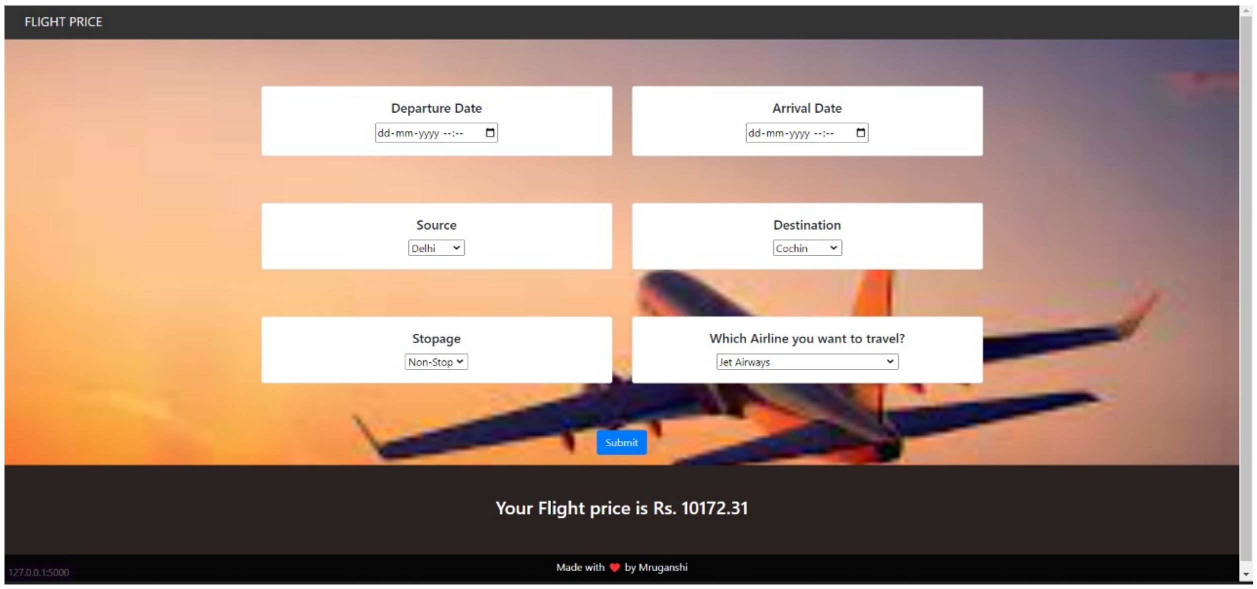


	Model	Training r2 Score	Testing r2 Score
0	Random Forest	0.861169	0.804951
1	XGBRegressor	0.838155	0.771505
2	Gradient Boosting Regressor	0.918393	0.825430

- So from above accuracies and seeing overfitting of models we can see that the best model for prediction is Gradient Boosting Regressor with the highest r2 score as 0.82540.

Deployment

- I built a flask API endpoint that was hosted on a local webserver. The API endpoint takes in a request with a list of values from a flight and returns an estimated price.
- Technologies used - python, Flask, HTML, CSS
- We can enter Departure Date, Arrival Date from the calendar that will be shown while hovering on the Date menu. Also we can select places like Delhi, Kolkata, Banglore ,chennai and In destination we can select places like Cochine, Delhi, New_Delhi, Hyderabad, Kolkata.
- In the stoppage section we can select after how many stations we want to stop like non-stop, 1,2,3 or 4 and in airline section we can select airlines like Jet Airways, Indigo, Air India, Multiple Carriers, SpiceJet, Vistara, Goair, Multiple carriers premium economy, Jet Airways Business, Vistara Premium economy and Trujet.
- After entering details, flight price prediction will be visible.
- Demo is shown below :


FLIGHT PRICE

Departure Date: dd-mm-yyyy --:--

Arrival Date: dd-mm-yyyy --:--

Source: Delhi

Destination: Cochin

Stopage: Non-Stop

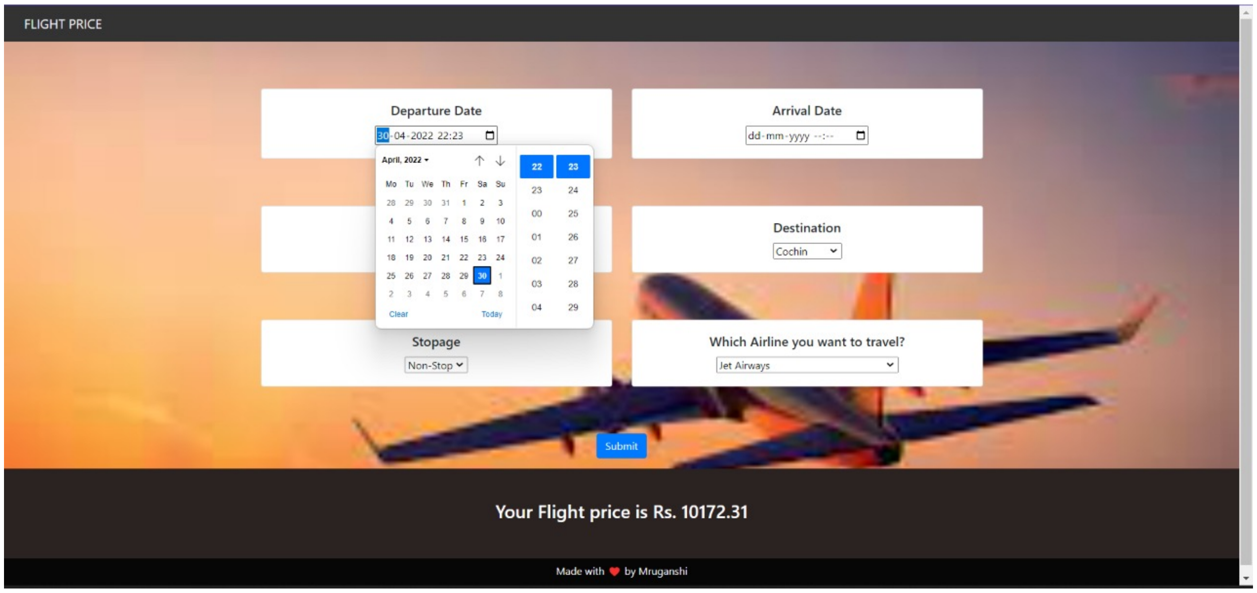
Which Airline you want to travel?: Jet Airways

Submit

Your Flight price is Rs. 10172.31

Made with ❤ by Mruganshi

127.0.0.1:5000


FLIGHT PRICE

Departure Date
30-04-2022 22:23

Arrival Date
dd-mm-yyyy --:--

Destination
Cochin

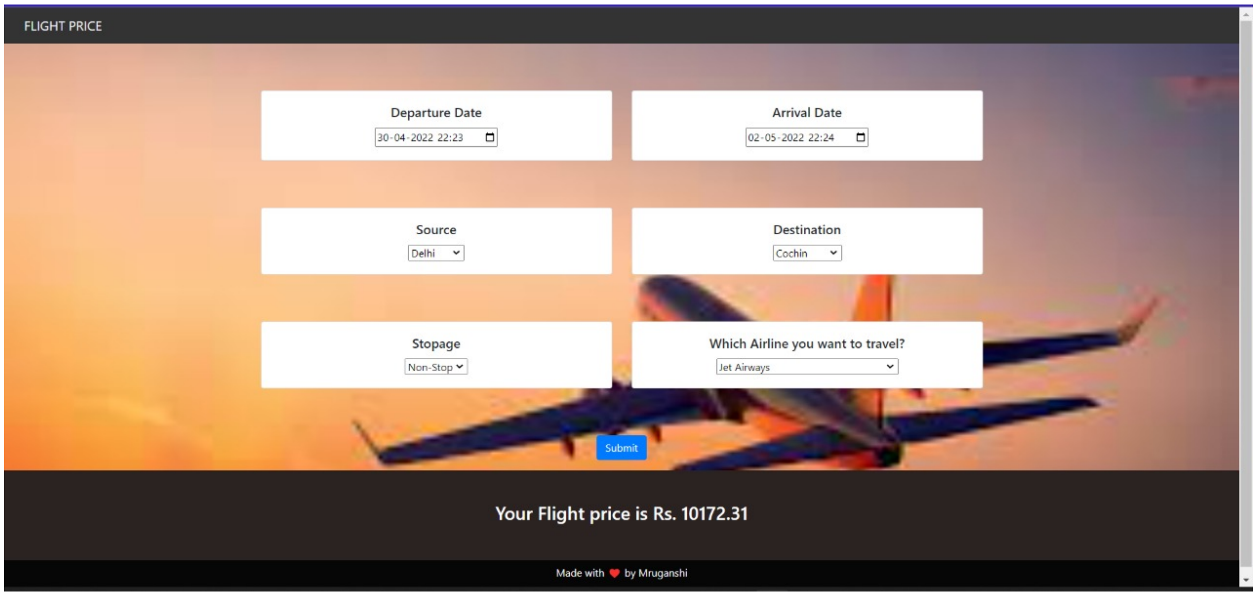
Stopage
Non-Stop

Which Airline you want to travel?
Jet Airways

Submit

Your Flight price is Rs. 10172.31

Made with ❤ by Mruganshi


FLIGHT PRICE

Departure Date
30-04-2022 22:23

Arrival Date
02-05-2022 22:24

Source
Delhi

Destination
Cochin

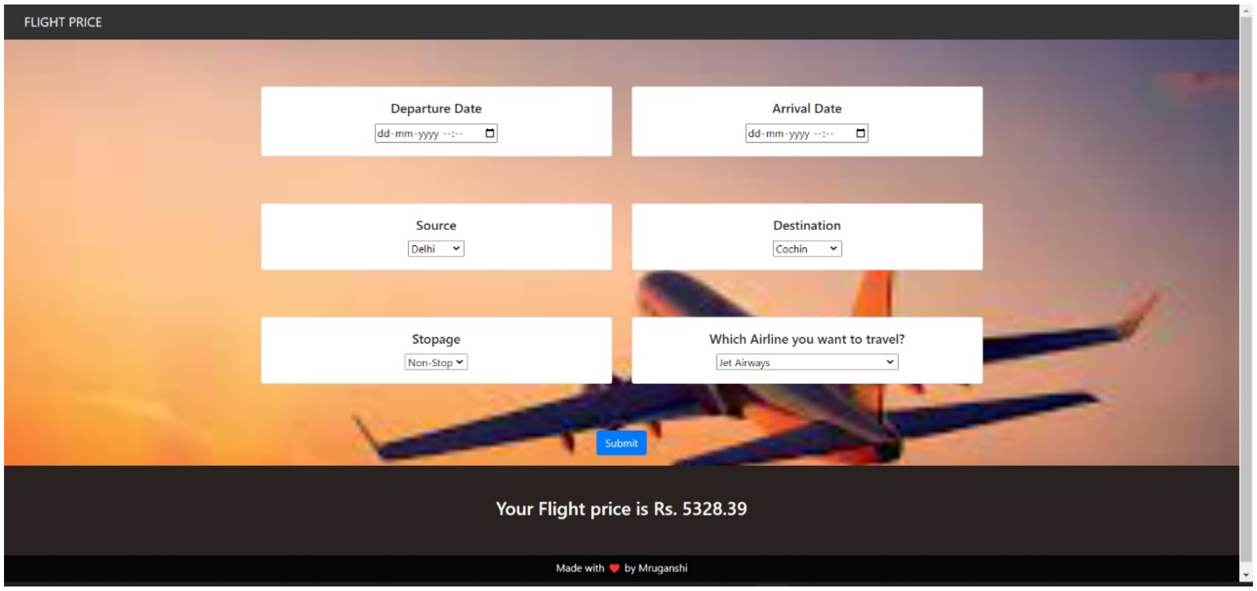
Stopage
Non-Stop

Which Airline you want to travel?
Jet Airways

Submit

Your Flight price is Rs. 10172.31

Made with ❤ by Mruganshi



Departure Date: dd-mm-yyyy --:--

Arrival Date: dd-mm-yyyy --:--

Source: Delhi

Destination: Cochin

Stopage: Non-Stop

Which Airline you want to travel?: Jet Airways

Your Flight price is Rs. 5328.39

Made with ❤ by Mruganshi