# Predict Optimal Time to Purchase Flight Ticket

Shashwat Aggarwal
2018097
shashwat18097@iiitd.ac.in

Sidhant S Sarkar
2018102
sidhant18102@iiitd.ac.in

Shikhar Sheoran
2018099
shikhar18099@iiitd.ac.in

## Abstract

*This project focuses on finding the optimal day to book a particular flight for which the flight ticket cost will be minimum before the departure date. For this purpose, we have shortlisted a set of characteristic features that are suspected to affect the price of flight tickets. The features are applied to various machine learning algorithms to predict the flight ticket cost. Performance of the various models was compared with each other. The data was collected from flight results at MakeMyTrip.com for top 7 domestic routes in India. The experiments showed that with the various model trained we are able to save an average of ₹2042 per flight. [GitHub]*

*Keywords:* ***flight ticket price; minimum cost; pricing model; machine learning; regression model***

## 1. Introduction

Flight prices vary sporadically over time for the same flight. These prices are controlled by the airlines autonomously thus making estimations of the prices difficult for the customers. Due to the high complexity of the pricing models applied by the airlines, it is very difficult for a customer to purchase an air ticket at the lowest price since the price changes dynamically. This will help save money, time and effort for anyone trying to book a flight ticket.

Several techniques that predict the flight prices, using sophisticated prediction models, have been proposed recently [1, 3]. The previous research has used machine learning models such as Linear Regression, Support Vector Machines, Multilayer Perceptrons etc., to conduct their analysis [2]. Following in their footsteps, we have used the same models, along with others, to conduct our analysis, where we aim to find the optimal day to book a particular flight by predicting the flight prices.

In the given analysis live data from *MakeMyTrip* was collected, which was then parsed and cleaned for training and validation. These steps have been given in detail in Section 2. Following this we analysed the general trends. There was a direct correlation between the flight prices and suspected features. After this, 6 regression models [SGD, MLP, Random Forest, KNN, SVM, Bagging Regression Tree] were selected. Grid search algorithm was run to find the best tuning parameters for all the models. The scores obtained were then compared by training on the top 15 features as reported by Pearson's correlation (f_regression). The best score reported using these models was found to be **0.51**

After this a completely new approach was incorporated, and the whole regression problem was modeled into a *binary classification problem*. A novel approach to generate weighted fare cost based on historical trends was calculated and reported.

Using the said calculated parameter multiple classification models were selected and trained. The best accuracy using classification models was found to be **0.89**

Finally the results were compiled and the final trends were analysed.The remaining scope of work and distribution have been discussed in the final section.

## 2. Literature Review

### Paper 1

> *Wang, Tianyi & Pouyanfar, Samira and Tian, Haiman & Tao, Yudong & Alonso, Miguel & Luis, Steven & Chen, Shu-Ching. (2019). A Framework for Airfare Price Prediction: A Machine Learning Approach.* [7]

This paper analyses Market Segment Level airfare price prediction taking into account the various economic and social factors (Other models highly local and biased). It primarily focuses on 2 public datasets published by the United States Bureau of Transportation Statistics. It then analyzes the two datasets and proposes a model framework for predicting. It follows a standard pipeline of first combining the two datasets, cleaning, then feature extraction, and finally model training. The features were ranked using Random Forest which gave priority to Distance, Seat Class, Passenger Volume for determining the price. The paper uses LR, SVM, MLP, XGBoost, RF as regression models to compare and analyse the performance. Random Forest performs the best and reports an adjusted R2 score of 0.869 and RMSE of $62.753.

### Paper 2

> *K. Tziridis & Th. Kalampokas & G.A. Papakostas. (2017). Airfare Prices Prediction Using Machine Learning Techniques* [6]

In this paper, the authors have collected flight ticket data for about 1800 flights for a particular route by scraping the web for Aegean Airlines (Greece). They have taken 8 features (departure time, arrival time, number of free luggage, days left until departure, number of intermediate stops, holiday day, overnight flight, day of week) for their research. 8 Baseline configurations were used to analyse and train a model by comparing accuracy and execution time (10-fold cross-validation was used). Combined performance for all features were calculated along with leaving one or two features out to understand the influence of the feature. overnight flight, day of week, holiday day did not have much influence in determining the flight price. Bagging Regression Tree, Random Forest Regression Tree reported the highest accuracy in all the tests.

## 3. Dataset

### 3.1. Collection

The data is scraped from *MakeMyTrip* for the top 7 domestic routes in India. The data was collected for all the routes for departure dates 2020-11-14 to 2020-12-23, with booking dates 2020-11-08 to 2020-11-18. The routes for which the data is collected are:

- Mumbai → New Delhi
- New Delhi → Mumbai
- New Delhi → Goa
- Bengaluru → New Delhi
- Mumbai → Goa
- Mumbai → Bengaluru
- New Delhi → Kolkata

The attributes that were extracted for each flight are: *airline, flight_code, departure_time, departure_city, flight_duration, arrival_time, arrival_city, flight_cost, number_of_stops, departure_date, departure_day, booking_date, booking_day*
These extracted features describe the entire flight. We have taken all the relevant features that are known to affect the price of a flight, along with some which are suspected to. We will derive features such as '*days_to_departure*' from the given features.

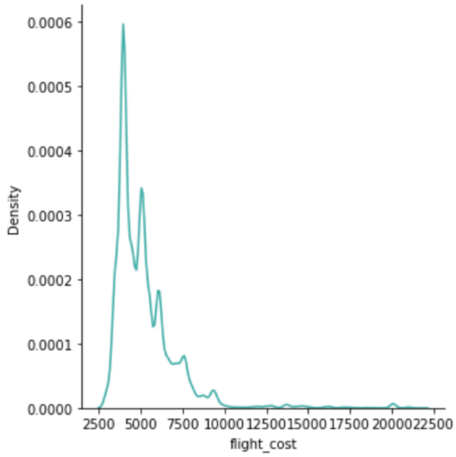

Figure 1. Flight Cost Density Plot

### 3.2. Preprocessing

All datasets for all booking dates were joined together. Airline codes were dropped. Flight cost was cleaned as it had various symbols and was of string type. Number of stops was also formatted to int. The extracted flight duration is of the format 'H hours M mins '. To make this data useful, the flight duration is converted to H*60 + M mins. This gives a wide range of flight duration and helps distinguish between times. Keeping time in hours would keep them closely related.

### 3.3. Final Features

After the preprocessing of the data, we have the final features as follows:

- *airline* : Enum (Air India, AirAsia, Go Air, IndiGo, Spicejet,Vistara)
- flight_path : One of the 7 routes we have chosen
- *departure_day* : Integer, representing the day of the week at which flight departs (0=Monday, 1=Tuesday and so on)
- *booking_day* : Integer, representing the day of the week at which flight is booked (0=Monday, 1=Tuesday and so on)
- *departure_time* : Enum (morning, afternoon, evening, night), Represents the time of day at which flight departs
- *number_of_stops* : Integer in the range [0,5], represents the number of stop between the source and the destination
- *flight_duration* : Float , represents the total duration of flight in mins. Including all layovers.
- *days_to_depart* : Integer, represents the number of days between departure date and booking date

- *flight_cost* : Float, represents the cost of the flight in ₹

## 4. Methodology

### Phase 1 : Data Collection

Described in the previous section.

### Phase 2 : Feature Engineering and Analysis

- **Days to Departure:** The exact values of booking date and departure date or not meaningful indepently. These values are used to extract the number of days left to departure from the booking date. Days to departure = Departure Date - Booking Date.
- **Departure City, Destination City:** These are merged together to formulate the routes category. Since we have shortlisted 7 routes for our model, we will one-hot encode them after combining the pairs.
- **Combining Day and Time:** The day of departure and time of departure have been clubbed. The two parameters go hand in hand in the real world and are shown by the heatmap as well. This is done to reduce extra features and represent data accurately.
- **Removing Outliers:** From Fig 1. it is clear that values of flight cost $\geq$ 11000 are belonging to the extreme range and are classified as outlier for the dataset. These datapoints are eliminated from the dataset.
- **One-hot encoding:** All the categorical data has been one hot encoded to give equal weightage to each category for each feature. [‘airline’, ‘flight_route’, ‘departure_time_day’, ‘booking_day’]

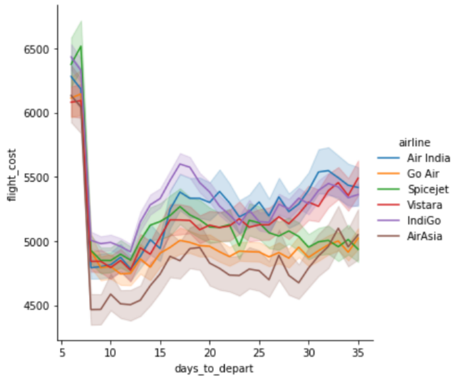

Figure 2. Comparing Flight Cost vs Days to Departure for each airline

### Phase 3 : Model Selection and Tuning

6 models were selected on the basis of previous studies discussed. These included Stochastic Gradient Descent, Multilayer Perceptron, Random Forest, K Nearest Neighbours, SVM, Bagging Regression Tree. Model hyper-parameters were tuned using the Grid Search approach with cross validation of 5 folds. [4]

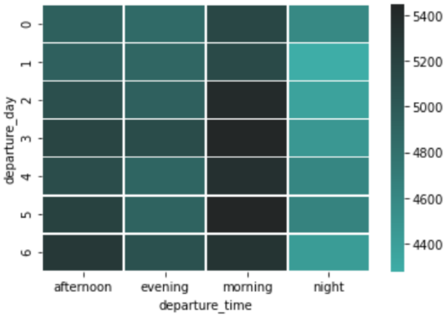

Figure 3. Heatmap of Flight Cost vs Departure Day and Departure Time

### Phase 4 : Performance Comparison and Evaluation

Using R2 score and Mean Absolute Error (MAE) as metric the models were compared. A 5-fold cross-validation procedure was applied to all the experiments and the mean performance of each model is presented in this section. Feature selection was done using the k-best method based on f_regression model. The best features obtained were then used to train models with the same tuning parameters and the results were compared to previous corresponding models.

### Problems with the Regression Model

Even after collecting large data samples., the model started plateauing in terms of learning to accurately predict the price for a discrete flight. Using a naive approach to run the model with changing the days to departure parameter from current day to 1 day left to depart only cascades the error and increases the overall error for each prediction.

A better way to approach the whole problem would be to classify a particular input with respect to the parameters passed into two classes i.e. buy or wait. Based on the general trend and historical data the model can easily learn whether or not the price in the current day might reduce in the future or not.

A simple yes no answer whether to buy the ticket at current price or not would yield better results. The next section discusses the approach followed.

## 5. A Classification Problem Approach

The given problem to find the optimal time to book a flight can be converted into a classification problem of finding if for a given input flight whether a flight ticket should be bought that day or wait until some future day to get a cheaper ticket.

As determined by previous experiments, *airline, flight_path, custom bin departure_time_day, and days_to_depart* were the most important features that impacted the flight price.

### Combining Fares by grouping:

Using these as keys the entire data-set was clubbed/grouped to form data slices. The *flight_cost* of each entry in the data slices were then aggregated in various forms:

- **Mean fare:** This is the average of the fare of all the flights in a particular group corresponding to departure day and days to departure. However, due to high standard deviation, this metric is not insightful.
- **Minimum fare:** Due to various discounts, offers and other events, minimum fare would not give a generalized insight.
- **First Quartile:** This is the middle value of the smallest value and the median value. First quartile is a good metric as it gives minimized generalization of fare.
- **Custom Fare:** It is important to give more weightage to recent trends over past trends as flight prices fluctuate sporadically. Keeping in considering the insights of first quartile metric, custom fare weights recent prices more by taking the following formula:

$$\begin{aligned}\textbf{Custom\_Fare} &= w \times (First\,Quartile\,for\,the\,currentslice) \\ &+ (1-w) \times (First\,Quartile\,of\,current - x\,days) \\ &[Here,\ \textbf{w = 0.75}\ and\ \textbf{x = 5}\ \textbf{days}\ (by\ observation)]\end{aligned} \tag{1}$$

- **First Quartile for the current slice**: represents the first quartile aggregated value of the flight costs in the given group and *days to depart d.*
- **First quartile of current - x days**: represents the first quartile aggregated value of the flight costs of the given group and *days to depart d-x.*

The above formula is an easy way to assign weight to current trends being observed in data. It calculates the weighted sum of the first quartile of current entry with the first quartile of a few days back.

Using this we are able to capture historic information of the price change and thus can use this as an important feature to train our model.

### Calculating whether to buy or wait for the this data (assigning label)

The most important task that remains is to use the custom fare to determine the classification labels.

$$\begin{aligned}\textbf{Label} = \textbf{1}\ if\ &\textbf{custom\_fare for d} < \textbf{custom\_fare for current D},\ where\ \textbf{d} < \textbf{D} \\ &[Here,\ d\,represents\,any\,slice\,with\,days\,to\,departure \\ &less\,than\,that\,of\,current\,days\,to\,departure\,slice]\end{aligned} \tag{2}$$

This tells us whether the price will dip or not for the flight selected. If the flight cost will not decrease, it will be labelled as 1 [Buy] meaning, the current time is the optimal time to purchase between current day and departure day. Label will be 0 [Wait] otherwise meaning, if the flight cost is expected to be lower in the coming day before the departure day. Hence, do not buy the flight ticket right now and wait for the optimal day.
We call this generated data-set as the *pivot data*. We then combine this data-set with the original data-set that was collected to create a modified data-set with labels attached for our classification problem.

### Running Various Models

Using the above data-set we ran our classification models. These include:

- Logistic Regression - max_iter : 100, penalty: 'l2'
- Random Forest - max_depth : 10, n_estimators : 100
- Gaussian Naive Bayes
- SVM - kernel : rbf
- MLP - alpha : 0.01, hidden_layer_sizes : (64, 32, 16)

Best model is **MLP Classifier** with **hidden layers sizes = [64, 32, 16]** with testing accuracy of **80.8%**.

| Model | Without Feature Selection | | With Feature Selection | |
|---|---|---|---|---|
| | R2 Score | MAE | R2 Score | MAE |
| Stochastic Gradient Descent | 0.3978 | 771.90 | 0.3972 | 769.69 |
| Multilayer Perceptron | 0.4355 | 732.58 | 0.4436 | 726.99 |
| **Random Forest** | **0.5145** | **659.64** | **0.4974** | **674.40** |
| K Nearest Neighbours | 0.4072 | 757.31 | 0.41834 | 736.60 |
| SVM | 0.3267 | 722.41 | 0.3257 | 728.91 |
| Bagging Regression Tree | 0.5145 | 662.08 | 0.4962 | 678.46 |

Table 1. Accuracy metrics for the models, with and without Feature Selection (F.S.)

| Model | Training Accuracy | Testing Accuracy |
|---|---|---|
| Logistic Regression | 0.71811 | 0.70621 |
| Random Forest Classifier | 0.82579 | 0.80514 |
| Gaussian Naive Bayes | 0.57524 | 0.55741 |
| **MLP Classifier** | **0.89714** | **0.80834** |
| SVM | 0.78604 | 0.76600 |

Table 2. Accuracy metrics for the models using classification problem

## 6. Results and Analysis

### For Regression

Features selected after feature selection : ***[’AirAsia’, ’Go Air’, ’IndiGo’, ’flight_duration’, ’number_of_stops’, ’Bengaluru-New Delhi’, ’Mumbai-Bengaluru’, ’Mumbai-Goa’, ’Mumbai-New Delhi’, ’New Delhi-Goa’, ’New Delhi-Mumbai’, ’dd__0’, ’morning-2’, ’morning-3’, ’morning-5’, ’days_to_depart’]***

| Model | Tuning Parameters |
|---|---|
| Stochastic Gradient Descent | ’alpha’: 0.01, ’penalty’: ’l1’ |
| Multilayer Perceptron | ’alpha’: 0.05, ’hidden_layer_sizes’: (50, 100, 50) |
| Random Forest | ’max_depth’: 13, ’n_estimators’: 350 |
| K Nearest Neighbours | ’n_neighbors’: 11 |
| SVM | ’kernel’: linear |
| Bagging Regression Tree | ‘n_estimators’: 20 |

Table 3. Hyperparameters obtained after Grid Search

**From Figure 2** As days to depart decreases, the flight cost decreases initially however with few spikes. Near the 10 days marks flight cost reaches a minimum. However, after that minimum there is a sharp rise.
**From Figure 3** Flight cost is generally high for morning flights. During the weekend (day 4-6) the cost increases as people tend to travel more during weekends on domestic routes. Night prices are relatively cheaper due to less preference of travelling at night.
The best model resulted after Grid Search turned out to be for Random Forrest. This model outperformed all other 5 models with and without feature selection. (See Table 2). The reported best mean absolute error is 659.64 with R2 score 0.5145. Thus, the model is able to predict values within a reasonable margin when considering that the average flight cost is around ₹5000.

### For Classification

The accuracy using the classification models beat the regression models by a big margin. The classification models captured the true essence of historical patterns in the flight data.

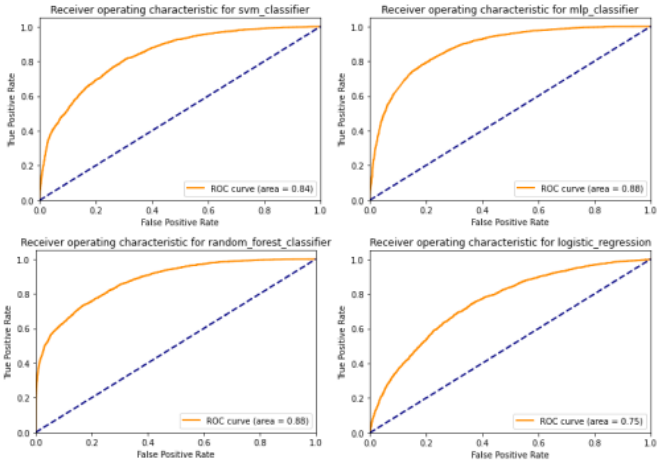

Figure 4. ROC Curve for various classification models

The best model reported an accuracy score of **89.71%** on the training set and an accuracy score of **80.83%**.

| Class | Precision | Recall | F-score | Support |
|---|---|---|---|---|
| Wait | 0.85 | 0.84 | 0.85 | 6205 |
| Buy | 0.73 | 0.74 | 0.74 | 3479 |
| accuracy | | | 0.81 | 9684 |
| macro avg | 0.79 | 0.79 | 0.79 | 9684 |
| weighted avg | 0.81 | 0.81 | 0.81 | 9684 |

Table 4. Classification Report for MLP Model

*Random Forest Classifier* was a close competitor and trained more quickly than the MLP model. The accuracy score for both are available in Table 2.

**Figure 4** Depicts the various ROC curve obtained on the testing set from the classifier models namely: SVC, MLP classifier, random forest classifier, logistic regression.

## 7. Conclusion

Using the best model (Random Forest) the mean absolute error is ₹659.64 (mean flight cost ₹5028.70) which is 13.1% error. With the classification approach, the best model (MLP Classifier) gives a testing accuracy of 80.83

If flight tickets were purchased based on the information provided by the model, it is estimated to save over **₹5 Crore** at an everage saving of ₹1800 per flight.

### Learning

All relevant data and code with full documentation is available online. [5]

Grid search is an important and useful tool to extract best features. Aggregating features helped to reduce the mean absolute error. Removing outliers is important to get a better fitting model. Regression is limited for generating high accuracy results when features are a combination of numeric and categorical values.

Analysing the dataset and correlation heatmaps gives insights into the expected importance of features. These features can further be explored with the help of feature extraction techniques. Thus, proper EDA and exploration of data is important.

Approaching a problem differently and converting the regression problem into a classification problem helped improve the overall performance and reduce the training time for the models.

### Future Work

The current model performs very well on the 7 flight path data that was collected. A major challenge and future scope of improvement can be to generalise the model to accommodate any flight path.

Currently the regression model and classification model work separately, combining them to work together would definitely impact and improve the overall pipeline, Where not only we tell a user when to buy a ticket but also successfully report how much it would cost.

### Individual Contribution

Individual contribution of each team member is shown in table 5.

| Shashwat | Sidhant | Shikhar |
|---|---|---|
| Data Preprocessing | Data Preprocessing | Data Preprocessing |
| Graph Generation | Data Collection | Data Collection |
| Data Description | Graph Generation | Feature Collation |
| Features Selection | Literature Review | Motivation |
| Abstract | Methodology | Model Testing |
| SVM | KNN,RF | MLP, BRT |
| Cost Saved Calculations | Pivot Data-Set Generation | Classifier Training |
| Conclusion | Introduction | Result/Analysis |

Table 5. Individual Contribution

## References

[1] J. A. Abdella, N. Zaki, K. Shuaib, and F. Khan. Airline ticket price and demand prediction: A survey. *Journal of King Saud University - Computer and Information Sciences*, 2019.

[2] W. Groves and M. Gini. An agent for optimizing airline ticket purchasing. volume 2, pages 1341–1342, 05 2013.

[3] P. Malighetti, S. Paleari, and R. Redondi. Pricing strategies of low-cost airlines: The ryanair case study. *Journal of Air Transport Management*, 15(4):195 – 203, 2009.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] S. Aggarwal, S. Sarkar and S. Sheoran. ”flight price prediction model”. `https://github.com/shashwataggarwal/ml-flight-predictor`.

[6] K. Tziridis, T. Kalampokas, G. A. Papakostas, and K. I. Diamantaras. Airfare prices prediction using machine learning techniques. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1036–1039, 2017.

[7] T. Wang, S. Pouyanfar, H. Tian, Y. Tao, M. Alonso, S. Luis, and S.-C. Chen. A framework for airfare price prediction: A machine learning approach. pages 200–207, 07 2019.