Flight Price Prediction

Pattern Recognition and Machine Learning Bonus Project

Challa Bhavani Sankar (B20EE014)

Introduction

This report talks about the implementation of flight price prediction using a variety of regression models. The data set includes a CSV file containing 10683 rows and 11 columns including the Price column. The prices of tickets between the months of March and June are included in the data set.
Feature selection techniques have been used to reduce noise in features. Selected models have been parameter tuned to achieve good performance of the models.

Data Pre-Processing

The following columns are included in the data set : *'Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route', 'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops', 'Additional_Info'*.

The columns *Arrival_Time, Duration*, and *Dep_Time* have been encoded in the format of hours and minutes. Each column has been separated into 'hour' and 'minute' columns
The column *Date_of_Journey* has been encoded in the format of dd/mm/yyyy. This column has been used to create a new column containing the day of the week. This column has been separated into date, month, and year columns. As all the tickets correspond to the same year, the year column has been dropped.
All other categorical columns have been label encoded.

Types of Regressors

Random Forest Regressor

Random forests or random decision forests is an ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time. For regression tasks, the mean or average prediction of the individual trees is returned.

Gradient Boost Regressor

Gradient boosting is a machine learning technique used in regression and classification tasks. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees. When a decision tree is a weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms random forest. A gradient-boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function.

XG Boost Regressor

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting that solves many data science problems in a fast and accurate way.

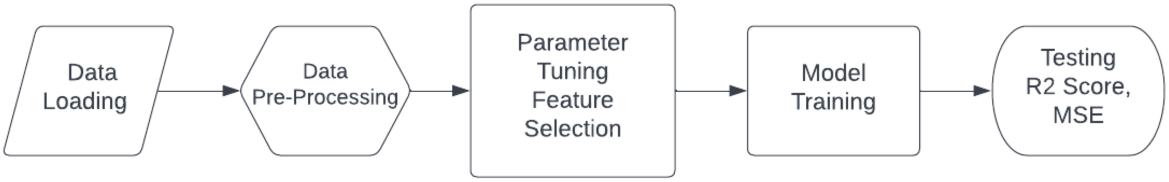
XGBoost works as Newton-Raphson in function space unlike gradient boosting which works as gradient descent in function space, a second-order Taylor approximation is used in the loss function to make the connection to the Newton-Raphson method.

Salient features of XGBoost which make it different from other gradient boosting algorithms include:

- Clever penalization of trees
- A proportional shrinking of leaf nodes
- Newton Boosting
- Extra randomization parameter
- Implementation of single, distributed systems and out-of-core computation
- Automatic Feature selection

Project Pipeline

A common pipeline was followed for all the three regression models used. And it is given as follows:



Model Building

The data has been split into 80% train data and 20% test data. The three models have been built and compared based on their runtime and accuracy. Cross validation scores have been calculated on the whole dataset to test overfitting or underfitting of models.

The below are cross validation scores:

Model	R2 Score	Mean Squared Error	Run Time
Random Forest Regressor	0.8852	2845010.373	17s
Gradient Boost Regressor	0.8267	3969469.134	12s
XG Boost Regressor	0.8839	1972492.199	3s

The Gradient Boost Regressor has least performance and medium run time. XG Boost and Random Forest have better performance compared to Gradient Boost, but XG boost has lesser runtime.

Parameter Tuning and Feature Selection

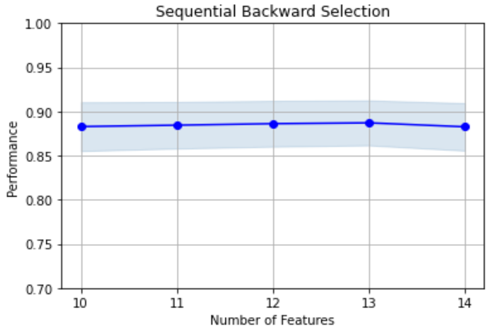
Parameter Tuning

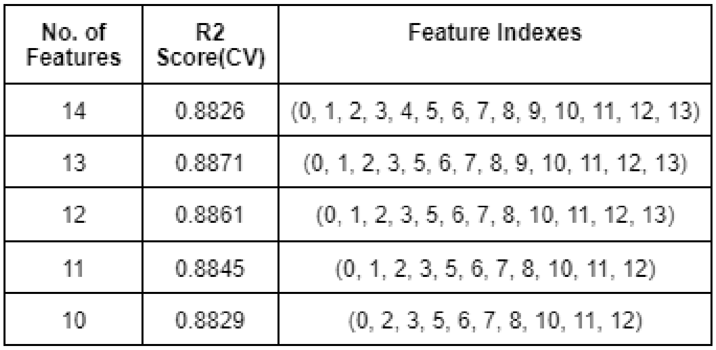
Parameter tuning has been performed on XG Boost regressor and Random Forest Regressor. Grid Search CV has been used for parameter tuning.

Feature Selection

The XG Boost regressor has been used to drop features that reduce the performance of the model. Sequential backward selection has been performed on the data set using mlxtend library. The floating parameter has been passed as False.

The sequential backward search has been performed with number of desired features as 10. The cross validation scores of no. of features ranging from 10-14 has been obtained. The best score is given when '*Duration*' column is dropped.





No. of Features	R2 Score(CV)	Feature Indexes
14	0.8826	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)
13	0.8871	(0, 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13)
12	0.8861	(0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13)
11	0.8845	(0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12)
10	0.8829	(0, 2, 3, 5, 6, 7, 8, 10, 11, 12)

Conclusion

XG Boost Regressor gives an R2 score of 0.925 after parameter tuning and feature selection where as Random Forest gives an R2 score of 0.914 after parameter tuning and feature selection XG Boost has lesser run time compared to Rando Forest. When comparing the run time, XG Boost took 1s where as Random Forest took 15s.

XGBoost is a good option for unbalanced datasets. XGBoost needs only a very low number of initial hyperparameters (depth of the tree, number of trees) when compared with the Random forest. XG Boost might have chances of overfitting as well anc can rely only when testing data is provided.

Therefore for this model ensemble learning techniques perform better than other regression models out of which XG Boost has the best performance.

References

1. https://en.wikipedia.org/wiki/XGBoost
2. https://en.wikipedia.org/wiki/Gradient_boosting
3. https://en.wikipedia.org/wiki/Random_forest
4. K. Tziridis, T. Kalampokas, G. A. Papakostas and K. I. Diamantaras, "Airfare prices prediction using machine learning techniques," 2017 25th European Signal Processing Conference (EUSIPCO), 2017, pp. 1036-1039, doi: 10.23919/EUSIPCO.2017.8081365.
5. http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/
6. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
7. https://xgboost.readthedocs.io/en/stable/get_started.html