PRML - Data Contest Report

S Vishnu Tej
CE19B084
CS5691: Pattern Recognition and Machine Learning
IIT Madras, Tamil Nadu, India
ce19b084@smail.iitm.ac.in

Royyuru Sai Prasanna Gangadhar
ME19B190
CS5691: Pattern Recognition and Machine Learning
IIT Madras, Tamil Nadu, India
me19b190@smail.iitm.ac.in

***Abstract*—Random Forest regressor is a supervised learning algorithm widely used for regression. In this paper we discussed the mathematical framework of Random Forest regressor and analyzed its application to predict the rating-scores that customers will likely assign to hotel bookings. We used numpy and pandas for data pre-processing and Random forest regressor in sklearn library to build the model. The trained model achieved a least square test error of 1.41 on more than 49000 entries.**

***Index Terms*—rating prediction, Random Forest, supervised learning, ensemble methods**

I. Introduction

Random Forest regressor is an ensemble technique which is one of the most famous machine learning algorithm, widely used for regression tasks. It has very high predictive power and builds multiple de-correlated decision trees to arrive at predictions. It is widely used in Banking sector, Stock market predictions, Health care and E-commerce.

A decision tree is built from a subset of data by partitioning. The partitioning is completed when the subset at a node has all the same values of the target variable, or when splitting no longer adds value to the predictions. Decision is a greedy algorithm. The best split from the previous node is chosen based on the criteria of Information gain or gini index[1]. This feature of the decision tree makes it highly vulnerable to over fitting. To address this problem, random forests make use of a bag of trees which are built using a subset of features of the original data set and a majority vote or average of predictions are used to predict the output to build the data set. Random forests are less likely to overfit the data.

A Random Forest regressor is a supervised learning algorithm and is one of the most adaptable and user-friendly algorithms. Multiple decision trees are trained from different subsets of training data, and random forest averages the output from multiple trees to arrive at a solution. Here each Decision tree uses a set of binary rules to determine the value of the target variable. Decision-tree biases are also eliminated by random forests by averaging out all of their predictions. It entails choosing the most significant features from the available features in training data set.

In this paper we are given the task of building prediction models for customers to predict rating-scores that they will likely assign to hotel bookings. The data set is derived from a real world scenario and contains information on payments, bookings, and customer info.

The paper begins with the mathematical framework of Random Tree regressor .In the latter sections will cover the methodologies followed like Exploratory data analysis, and how Random Forest regressor is employed to get accurate predictions

II. Parallel Ensemble Methods

Parallel Ensembles combines several models, each of which were built separately to fit data. This allows for Naive parallelisation. Example of a parallel Ensemble is bagging.

In Bagging, learners are often high capacity and generally over fit. Bagging combines several separate models learned on some subset of the data simply by averaging (or taking a majority vote).High capacity learners give very distinct models when trained on different subsets of the training data.

A. Bagging - Mathematical Framework

$$S = (x_1, y_1), ..., (x_m, y_m)$$

$$b_s \subseteq S$$

Let H(S) be the MSE of Bagged Classifier

$$H(x) = E_{bs}[h_{bs}(x)]$$

$$MSE[H] \leq E_{bs}[MSE[h_{bs}]$$

Thus, MSE of bagged model is no worse than the average MSE of individual models. it is also evident that if individual models have a very high variance the bagged model will give better predictions compared to individual models. Thus decision trees are very good candidates for bagging, because they are very easy to over fit and difficult to regularize and they are highly unstable learners. To increase the variance, each individual node of each tree is learned on only a subset of features. This helps in case there is a highly predictive feature, as all learners will use this feature and hence will be correlated. The resulting ensemble method is called the Random forest regressor/classifier.

- **Regression Random Forest:** Here the decision trees predicts what is likely to happen, given previous behaviour

Identify applicable funding agency here. If none, delete this.

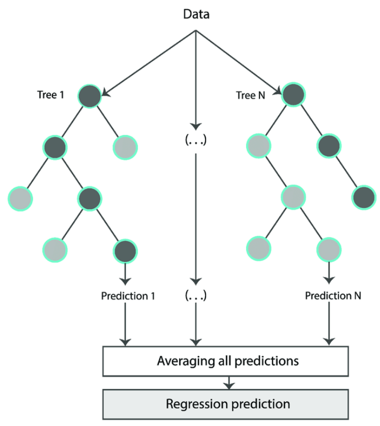

Fig. 1. Random Forest Regressor

or trends and the model selects the best solution by taking mean of the individual output from trees.

B. Regularization in Random Forests

- **Number of estimators**: The training accuracy generally increases with increase in training accuracy, and flattens out with further increase in increase in number of estimators.Number of estimators can be a hyper parameter of the model.
- **Maximum features**: Random forest classifier only uses a subset of features of the original data set to build trees.
- **bootstrap**: Sampling with replacement is generally used to build the decision trees. Bootstrap sample without replacement can also be used, but trees that are built might be poor in generalization
- **Sample size**: A fraction of samples is used to build the individual predictors. Thus,fine tuning the sample size might result in more generalized bag of predictors[2].

C. Advantages of Random Forest

- It can perform both regression and classification tasks.
- A random forest produces good predictions that can be understood easily.
- It can handle large datasets efficiently.
- The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm.

D. Disadvantages of Random Forest

- Computationally expensive than decision tree classifier
- Requires more training time than traditional decision tree algorithm.

E. Cost Function

Loss function used is in the training are the squared error and absolute error. The predictions are reported on the model that is trained using squared error as the loss functions, the only reason being that the cost function is convex. Whereas the absolute error is not convex, and hence it takes significantly longer time to train the model.

F. Evaluation of the model

In Random forest regressor model, we can use criterion like squared error, absolute error and poisson. For this project we decided to use squared error.

$$MSE = \frac{\Sigma(y_{test} - y_{pred})^2}{n} \tag{1}$$

III. THE PROBLEM

The main aim of this project is to develop a model to predict the rating of a customer based on various factors like price, agent fees, hotel category, booking status etc. For this we have decided to use a random forest regressor.

A. Exploratory Data Analysis - Creating a merged dataset

There are a total of 7 data sets namely bookings, bookings data, customer data, hotels data and payments data along with test and train data. train and the test data consists of only 2 columns: Booking id mapped with the corresponding ratings given by customers. The information of a particular booking is spread across 5 data sets. So the first objective is to create a merged data set comprising the information from other data sets like booking create timestamp, booking approved at, hotel id, seller agent id, payment type etc, booking status, customer check in time etc.

B. Exploratory Data Analysis - Handling the null values

Looking at the data sets provided we observe that in data set bookings.csv, there are nulls in booking approved time and booking check-in customer date, instead of filling the null values in each of the datasets we decided to work on the null values of these columns in the merged dataset. The data sets bookings data, customer data and payments data do not have any null values, they do not require any data preprocessing. In hotels dataset we have a negligible percent of nulls (1.85 percent) in hotel category, name-length, description length and photos quantity. So we decided to impute these nulls with mean for hotel description length and with mode of the feature for remaining 3 columns.

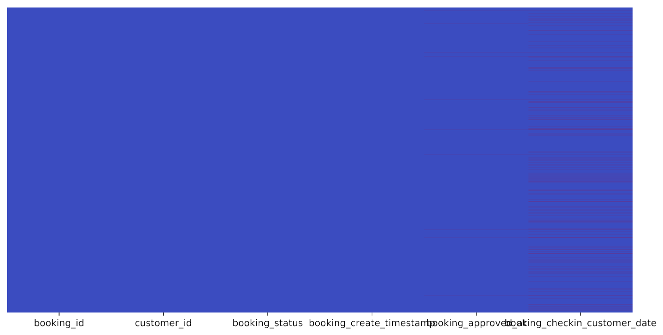

Fig. 2. Nulls in bookings.csv

Now to get all the required columns in the training dataset based on its booking id, we decided to first append customer id

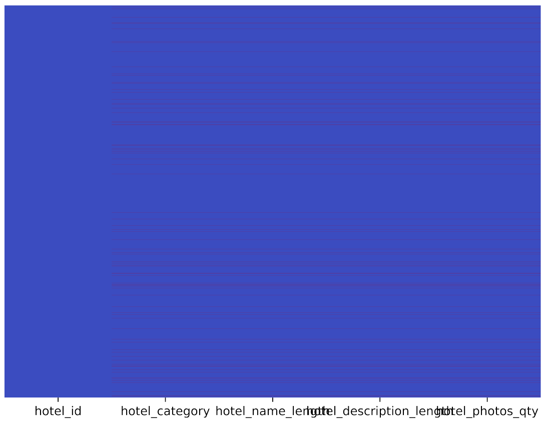

Fig. 3. Nulls in hotels data.csv

and customer unique id to the booking data which can be used as a foreign key to retrieve data like customer country from customers data set. These features(customer id and customer unique id) are also appended in payments data to retrieve data like mode of payment and number of sequential payments. We made use of the inbuilt python function "map" to map various columns across different datasets. The features customer id, customer unique id are appended by mapping booking ids of respective entries across all the data sets. The features hotel category, hotel photos quantity and hotel description length are added to bookings_data by mapping hotel_ids of respective entries. Maximum of booking sequence id is added to the training data set, by mapping booking id from the training data set to the data set of bookings_data. In the similar way seller agent id is mapped based on booking id, if multiple seller agents are present, the most frequent value has been mapped to the respective booking id. Payment type and payment value are mapped based on the booking id from the payments data. If multiple entries of the same booking id are present, payment value is replaced with sum and the payment type is replaced by the mode. In the similar way booking_create_timestamp,booking_approved_at and booking customer data all have been added to the training data by mapping respective booking ids. The country is mapped based on the customer unique id present in the customers data set and is added to the training data.

But this mapping is not perfect as there are some booking ids in the training data which are not present in the bookings_data. This resulted in the nulls in columns like price, agent fees, hotel category, description length and photos quantity. Upon careful analysis, it is observed that all the booking id's missing in the bookings data are present in the payments data. Another critical observation is that the sum of price and agent fees booking_data is exactly equal to the total payment value (present in payments data) for a particular booking id. Thus, the relation between the payment value in the payments data and price and agent fees is used to get a reasonable estimate for null values present in the bookings data whose sum of price and agent fees is closest to the required payment value. We grouped data sets based on booking_id. This helps is easy retrieval of any information from any data set based on booking id. Two functions $def closet\_val$ and $def return\_attributes\_$ were used to replace the missing values. The closest_val returns the closest values of the given feature value in the bookings_data data set based on payment value and the closest sum of price and agent fees. Later the missing values of the features price, agent fees, hotel description length, hotel name length are replaced by mean of values present in bookings_data data set, which is computed by return_attributes_. Similarly the null values in the features seller_id, hotel_photos_qty and hotel_category are replaced by mode. As seller_id on it's own doesn't add any value to the dataset we decided to use a parameter called seller rating and replacing the seller_id with the average rating of the corresponding id in the bookings_data dataset.

We decided to include 2 features diff approval and diff approval-2 referring to the time difference between booking create timestamp, booking approved at and booking create timestamp, booking checkin customer date respectively. As the null values in booking create timestamp, booking checkin customer date have not been treated in the bookings data, it is obvious we observe null values when we compute diff approval and diff approval-2. These null values have been imputed using appropriate functions impute_diffapproval and impute_diffapproval2 by filling the value of missing features with mean based on the corresponding booking status. This enables us to fill null values with reasonable time scales.

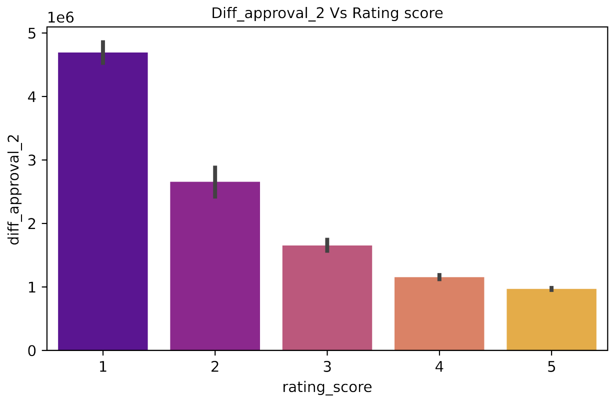

Fig. 4. Rating Score vs diff approval 2

From Fig 7, we can see the effect of various parameters on rating score. We can see that parameters like seller rating and diff_approval_2 significantly influence the final rating. This can be visualised more clearly in Fig. 4 where we can clearly see a negative correlation between diff_approval_2 and rating_score i.e if it there is a large time gap between booking_checkin_customer_date and booking_approved_at then it is more likely that the final rating is low. Similarly from Fig.

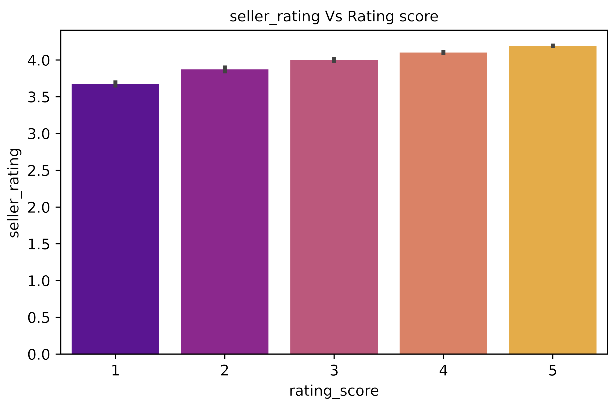

Fig. 5. Rating Score vs seller rating

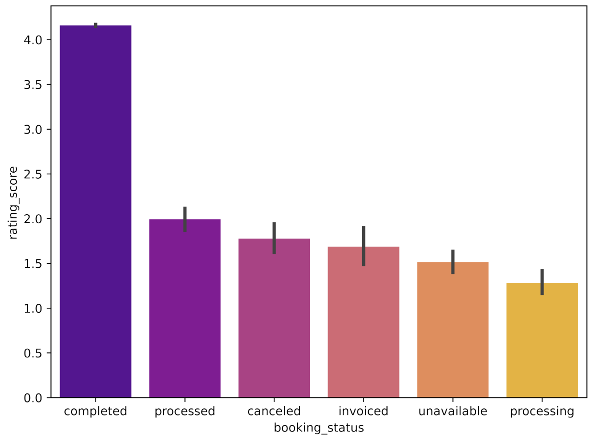

Fig. 6. Rating Score vs Booking status

5 we can see that rating_score is directly proportional to the seller rating of a given booking. Fig. 6 shows us the effect of booking status on the rating score. Bookings which are completed have higher rating compared to that of processing (or) cancelled.

C. Handling Categorical Data

Care must be taken while handling the categorical variables. The merged data set contains many categorical features like the Booking status, payment type, country and seller_agent_id. Many types of categorical encoding techniques have been explored including one-hot-encoding, frequency encoding and numerical encoding. The main problem with one-hot encoding is the curse of dimensionality. The feature seller_agent_id has more than 3000 levels, an indicator that one hot encoding cannot be used. A major draw back of numerical encoding is that the model interprets higher value numeric data as more significant which is more often not the case. The main problem with frequency encoding is lack of explainability.

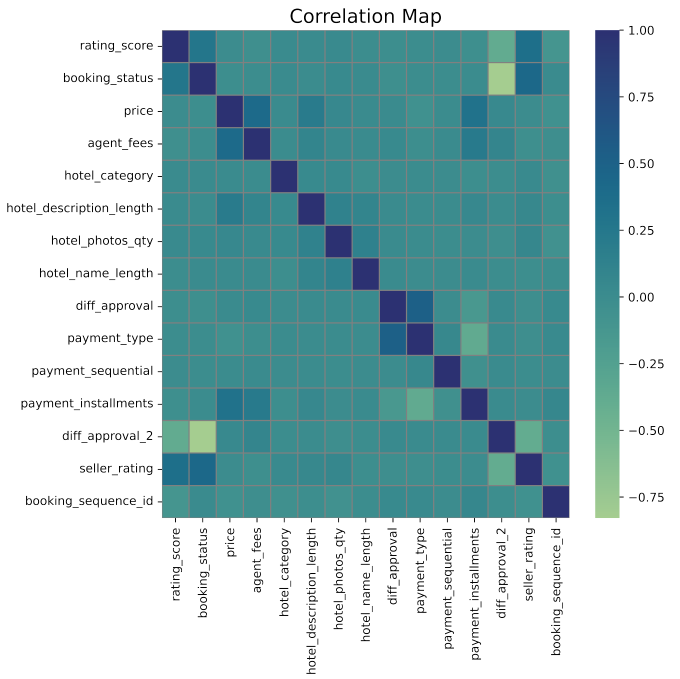

Fig. 7. Correlation Heat Map

Thus we tried a different approach to replace seller_agent_id, payment_type, booking_status. A seller agent is likely to receive positive feed back if the history of his service has received positive feedback. Similarly from figure. 6., it is evident that for the booking status "completed" is likely to receive higher rating compared to booking status "cancelled". Similarly payment_type of gift card is likely to receive higher ratings compared to payment_type "not defined". Thus we replaced these categorical variables with the average rating score in the training data, for the given respective feature value i.e "gift_card" is replaced with the mean of rating scores when the payment type is gift card. Similarly, "completed" is replaced with the mean of rating scores when the booking status is completed. The training data has been grouped by booking status, payment type and seller agent id for fast retrieval of information and replacement of categorical variables. However, the feature country is still encoded using the technique of one hot encoding.

IV. Model

A. Model selection

The prediction model used to predict rating-scores that customers will likely assign to hotel bookings is random forest regressor. The main reason for choosing random forest is that it is very unlikely to overfit the data and it gives reliable results.

Gradient boosting algorithm and k-nearest neighbours have been explored as the alternate prediction models. The training errors of various algorithms have been reported below:

- Random Forest Regressor: 1.2646

- Gradient boosting algorithm: 1.48
- K nearest neighbours: 1.57

Thus, based on the above results Random forest regressor is used as the prediction model.

B. Hyper parameter tuning using Grid search CV

There are three key hyper-parameters for random forest regressor algorithms that must be set prior to training namely: number of estimators, maximum depth, and maximum features. The optimal values for these parameters is obtained using Grid search CV which is imported from model selection module in sklearn library. After running Grid search CV, the optimal hyperparameter values obtained are :

- number of estimators: 250
- maximum depth of each tree: 9
- maximum features: 13
- Bootstrap: True

Each decision tree in the ensemble model is built from a sample taken from a training set with replacement known as the bootstrap sample. The dataset is subsequently randomized by feature bagging, which results in decorrelated trees which are highly reliable when we take an average. The individual decision tree predictions will be averaged for the regression job and the prediction is then finalised by cross-validation using the out-of-bag sample.

C. Model training

The merged data set containing multiple features from all the seven data sets is used as the training data set. The training data set is split into two parts X_train,Y_train and X_test and Y_test. The testing data set is used to choose between different models(Random forest, gradient boosting and KNN) and different techniques(undersampling and oversampling). The random forest regressor is trained using the training data with the hyper parameters obtained from Grid Search CV, using squared error as the cost function.

D. Imbalanced data

The training data is highly biased towards higher rating side, especially 5. Thus, the model which will be trained will have a natural bias towards higher value of rating, thus the model might not be very reliable. Thus, oversampling with replacement of under represented samples has been performed to reduce the bias towards high rating score. Surprisingly, validation error increased by 0.15 after oversampling of under represented samples, thus the idea of balancing the data set has been dropped.

E. Model testing

The model is then then tested on the test data samples_submission_5. Similar to the training data, a merged data set is created for the test data using the booking ids from sample_submission_5 and the trained regressor model is used to give predictions on the test data. The regressor model achieved a test score of 1.41419 using least squares as the assessment's evaluation criterion.

V. CONCLUSION

Some of the key conclusions from the analysis are:

- Rating score is highly dependent on the booking status
- Higher the difference between booking approved time and customer check in date, lower the rating score given to a particular booking
- Random forest regressor performed well compared to gradient boosting and gradient boosting performed well compared to the K-nearest neighbors, proving the reliability of ensemble methods compared to high variance KNN algorithm

Please refer the attached jupyter notebook file for detailed code and to check the result of our model test dataset. https://bit.ly/colablinkPRML

REFERENCES

[1] Bishop, Christopher M., Pattern Recognition and Machine Learning, 2006

[2] [Online sources] https://sites.google.com/site/harishguruprasad/teaching/prml-aug-2022

[3] [Online sources] https://www.javatpoint.com/machine-learning-random-forest-algorithm

[4] [Online sources] https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

[5] [Online sources] https://en.wikipedia.org/wiki/Random-forest