Cab Booking Cancellations

A model to predict the cancellation of cab due to unavailability

- **Yashwanth Kumar Yelamanchili**

Contents

Abstract

The aim of the project is to analyze the booking data that is provided by YourCabs.com and build a model to predict whether the ride would get cancelled because of the cab's unavailability.

The business faced a problem of ride cancellation by the cab drivers. Such a cancellation has an associated cost to the company, not to mention the customer's dissatisfaction and possible loss of business. A model to predict the event before it happens not only allows for a reduction of cost but also helps improve the business's customer service. Since the aim of the project is to predict if a ride would be cancelled or not, this is a classification problem. The business would be able to manage the drivers and customer relationships better by handling cancellations before any escalations.

The key objectives in the task include –

- Explore the dataset and analyze the features present
- Build supervised models on the dataset to classify the ride as cancelled or complete
- Evaluate the model's performance and validate the model on the testing dataset

Background

YourCabs.com is a company (currently possibly defunct) based out of Bangalore to bring the owners and vehicles together. The company provides a platform for the individual customers to book a cab, thus maintaining a real time demand for the supply. The service involves connecting the cab owners with the end customers, and YourCabs.com maintains the technology platform, charging a fee for each ride. As was the case during 2013, with the rise of other cab hailing platforms such as Uber and Ola (in India), the business should operate on complete efficiency. The business provides the customers with options to book point-to-point rides, long distance rides or rent a cab on an hourly basis. The business also provides an online platform, a mobile site, and the traditional way of hailing cabs. Most times, the cancellations occur at the last minute before the scheduled pick up time or is a "no show".

Kaggle.com is an online platform, quite popular amongst the data science community, to learn and compete on data projects. Companies and organizations can sponsor and create competitions for people around the world to compete against on pending business problems. YourCabs.com and Indian School of Business (ISB) came together to sponsor this contest to have participants develop predictive models to classify if a cab booking would be cancelled due to the unavailability of cabs.

Executive Summary

The project involved analyzing the ride information presented by YourCabs.com and ISB hosted via a Kaggle competition and building a classification model to predict a ride cancellation through cab unavailability (by the driver). The steps followed in building the process are as follows –

1. EDA on the data to understand the values and feature engineering to create the required variables for modelling process
2. Creating a stratified sample (balanced) dataset and a SMOTE resampled dataset. Splitting the dataset into a train and test split (70-30) for modeling purpose
3. Model building and validation through key diagnostic measures. A range of modeling techniques were used including logistic regression (with and without automated variable selection), CART, Random Forest, SVM and Neural Nets

The result of the process is summarized in the below table -

	Training Dataset			Testing Dataset		
Models	**AUC**	**MR**	**FNR**	**AUC**	**MR**	**FNR**
Logistic Regression	0.80	0.28	0.28	0.78	0.29	0.28
Decision Tree	0.75	0.12	0.46	0.70	0.16	0.45
Random Forest	0.89	0.04	0.20	0.73	0.15	0.38
Random Forest (grid search)	0.86	0.05	0.25	0.73	0.14	0.41
SVM	0.85	0.24	0.25	0.82	0.25	0.28
Neural Net	0.85	0.23	0.22	0.83	0.25	0.25

From the above results, a random forest that has been tuned for its hyper parameters performs the best on the testing data based on the misclassification rate (thought the FNR is higher than that of the SVM or Neural Network model). Also, the additional criteria of being able to find the variable importance makes this an attractive option.

The models were rebuilt using the SMOTE resampled dataset and the results (produced in a later section) are comparable.

Approach

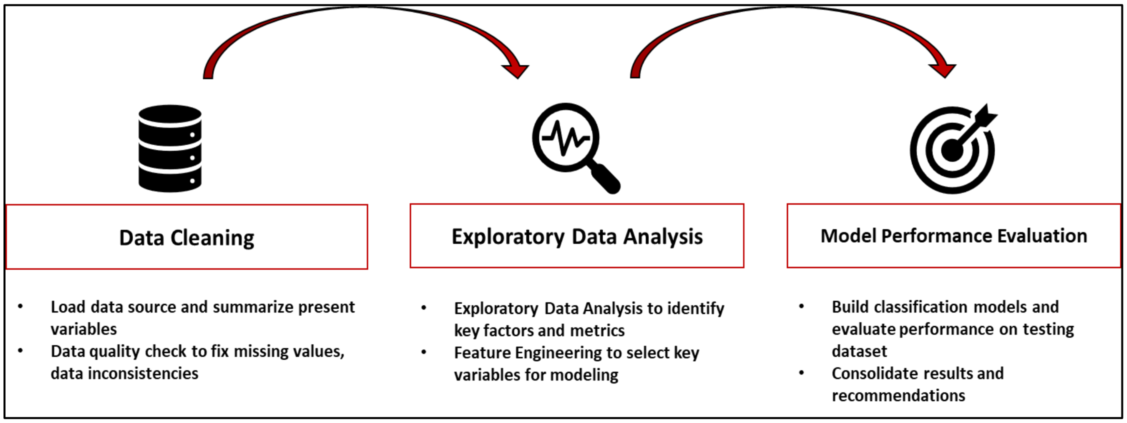
An iterative approach is taken to analyze the dataset and build the model.

Data Cleaning

A glimpse of the provided data suggests that there could be certain data discrepancies that needs to be fixed before processing. This involves summarize the dataset to check for any potential outliers, fix formatting for consistency and handle the missing values present in the data.

Exploratory Data Analysis

While a lot of features are present in the dataset, a few of the column may not be necessary for further analysis. Conducting univariates and visualizing the data helps in identifying key features for further analysis. A few columns (like identification, timestamps) are to be deleted while a few metrics are derived for further enhancements. Two of the main metrics derived are the waiting time (difference between the booking time and trip start time), and the distance between origin and destination (calculated using the latitude and longitude information). Dummy variables are created for certain categorical variables (like booking platform, ride type).



Data Cleaning

- Load data source and summarize present variables
- Data quality check to fix missing values, data inconsistencies

Exploratory Data Analysis

- Exploratory Data Analysis to identify key factors and metrics
- Feature Engineering to select key variables for modeling

Model Performance Evaluation

- Build classification models and evaluate performance on testing dataset
- Consolidate results and recommendations

Figure 1 - Analytical Approach

Modeling and Validation

A set of candidate models are identified that would be trained on the dataset. The major modelling techniques to be used are Logistic Regression, Decision Trees (that include Classification Trees, Random Forest and Boosting trees) and Neural Networks. The model is evaluated using the Weighted Mean Squared Error (as defined by the Kaggle competition). A few other criteria for model selection include AUC (Area Under the Curve), Misclassification Rate, in-sample and out-of-sample error. The result would include the final model and the classification of the testing dataset based on the model. Each model requires certain hyperparameters to be tuned (tree size, learning rate and so on) which would be achieved through cross validation. The imbalance of the data is handled using stratified sampling.

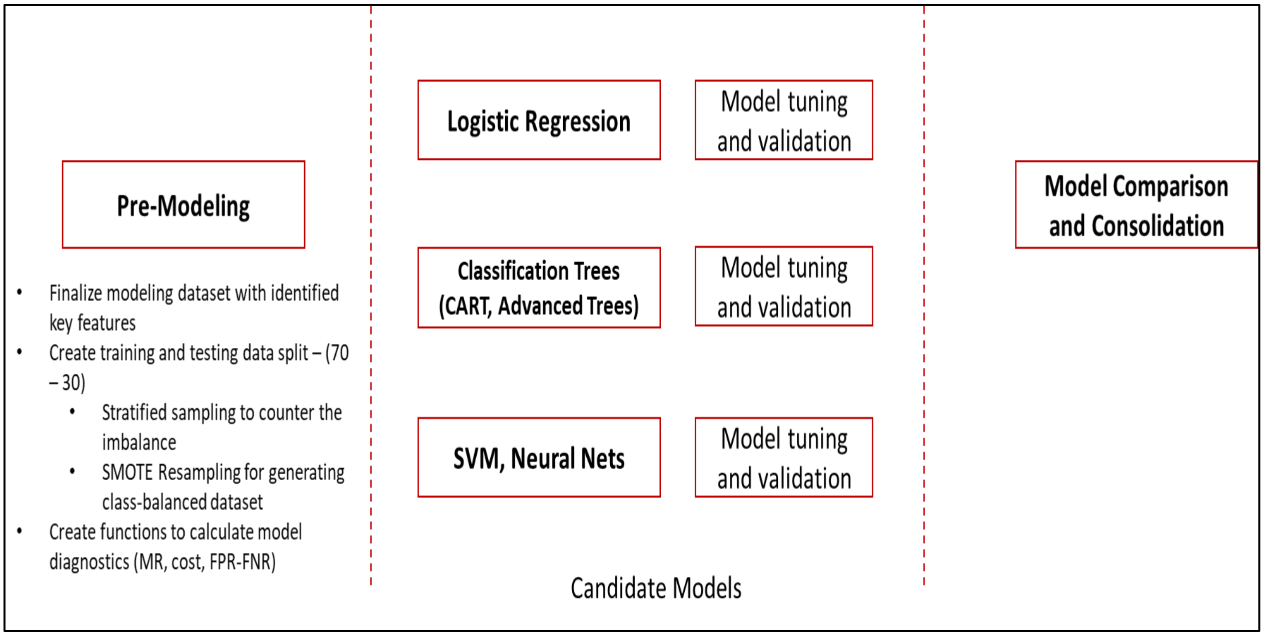

Pre-Modeling
Finalize modeling dataset with identified key features
Create training and testing data split – (70 – 30)
Stratified sampling to counter the imbalance
SMOTE Resampling for generating class-balanced dataset
Create functions to calculate model diagnostics (MR, cost, FPR-FNR)
Logistic Regression
Model tuning and validation
Classification Trees (CART, Advanced Trees)
Model tuning and validation
SVM, Neural Nets
Model tuning and validation
Candidate Models
Model Comparison and Consolidation

Figure 2 - Modeling and Validation

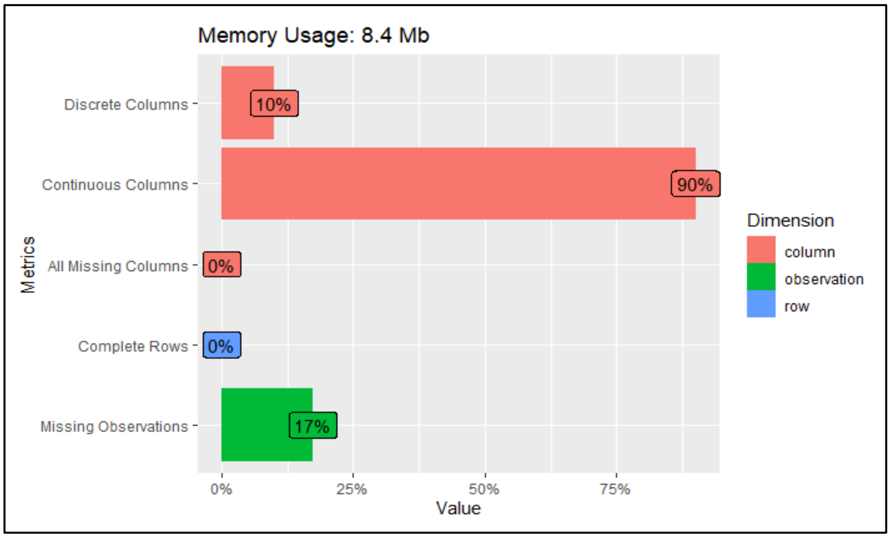
Data Exploration

We started off by exploring the dataset to understand the level of the data, the factors available that may be useful in making predictions. We also tried to determine if any new variables can be computed that can further enhance the prediction model. The dataset available is a transactional data that is, it provides the information of each booking that has been made with YourCabs.

There are two datasets that were present in the Kaggle competition A training dataset that would be used for the model building process and a testing dataset for model performance evaluation. The training dataset contains 43431 records and 20 columns, while the testing dataset contains of 10000 records and 18 columns. The training dataset contains the class and the cost of misclassification that is not present in the testing dataset. The below table summarizes the features present in the dataset.

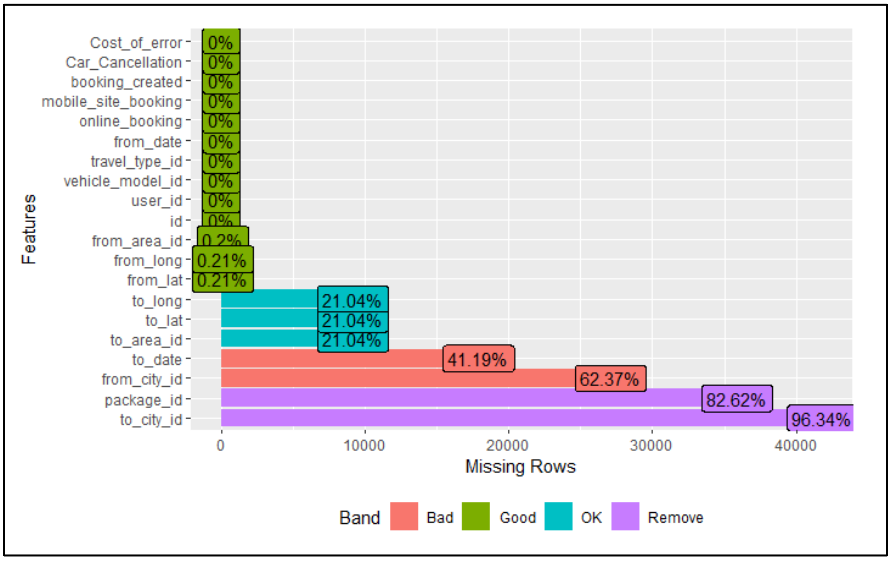
Data Field	Description	Remarks
id	booking ID	
user_id	the ID of the customer	based on mobile number
vehicle_model_id	vehicle model type	
package_id	type of package	1=4hrs & 40kms, 2=8hrs & 80kms, 3=6hrs & 60kms, 4= 10hrs & 100kms, 5=5hrs & 50kms, 6=3hrs & 30kms, 7=12hrs & 120kms Applicable only for hourly rentals
travel_type_id	type of travel	1=long distance, 2= point to point, 3= hourly rental
from_area_id	unique identifier of area	Applicable only for point to point & packages
to_area_id	unique identifier of area	Applicable only for point to point & packages
from_city_id	unique identifier of city	
to_city_id	unique identifier of city	Only for intercity
from_date	time stamp of requested trip start	
to_date	time stamp of trip end	
online_booking	if booking was done on desktop site	
mobile_site_booking	if booking was done on mobile site	
booking_created	time stamp of booking	
from_lat	latitude of from area	
from_long	longitude of from area	
to_lat	latitude of to area	
to_long	longitude of to area	
Car_Cancellation	whether the booking was cancelled	1: Booking canceled 0: Uncanceled Booking
Cost_of_error	The cost incurred if the booking is misclassified	For a cancelled booking, the misclassification cost is 1. For a cancelled booking, the cost is a function of the of the cancellation time relative to the trip start time

Data Quality Checks



Glimpse of the dataset:

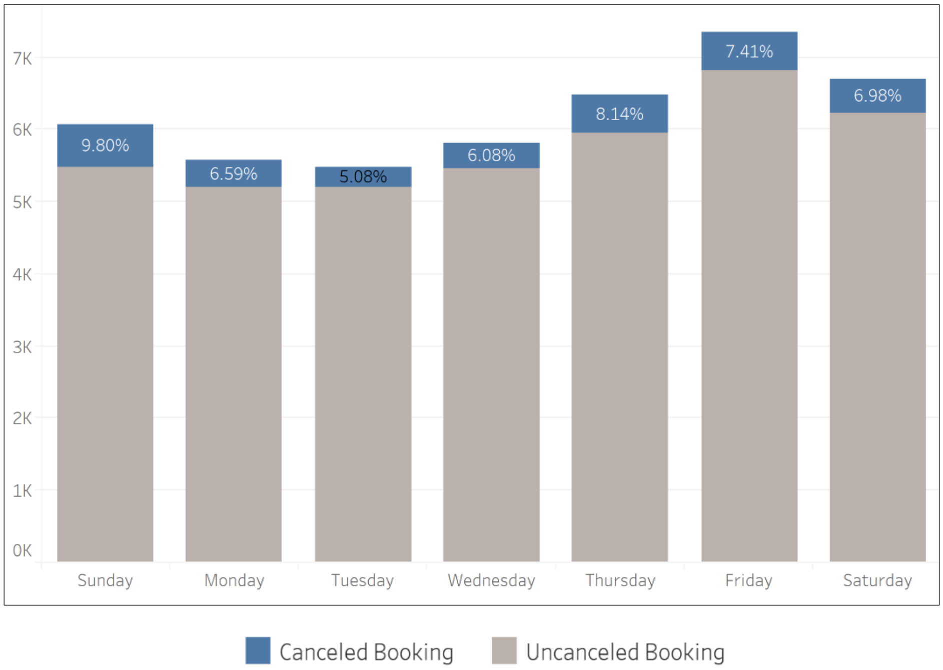
- Number of discrete variables = 2
- Number of continuous variables = 18
- There are 17% of records that have missing values



Missing value analysis:

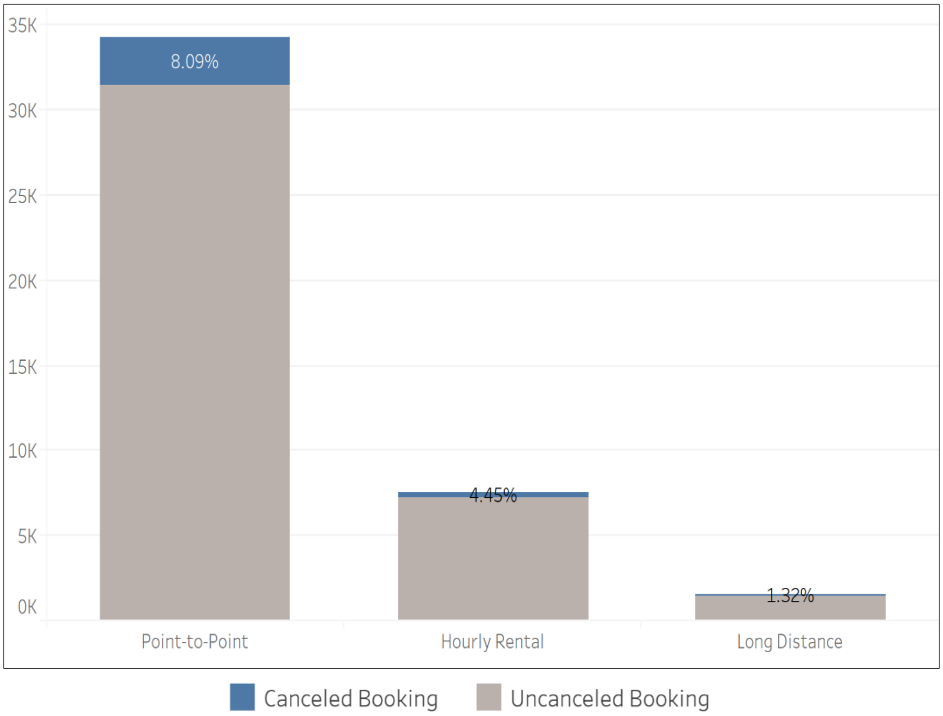
- There are 7 columns which had > 20% missing values
- Columns to_long, to_lat to_area_id are applicable only for point-to-point travel. Thus, there are missing values for other kinds of travel.
- Column package_id is applicable only for hourly rentals. Thus, there are missing values for other kinds of travel
- Rest of the 3 columns “to_date”, “from_city_id”, and “to_city_id” have been removed owing to the high percentage of missing values

Quick initial analysis of the predictors helped in identification of the variables that can likely come as significant during model building.



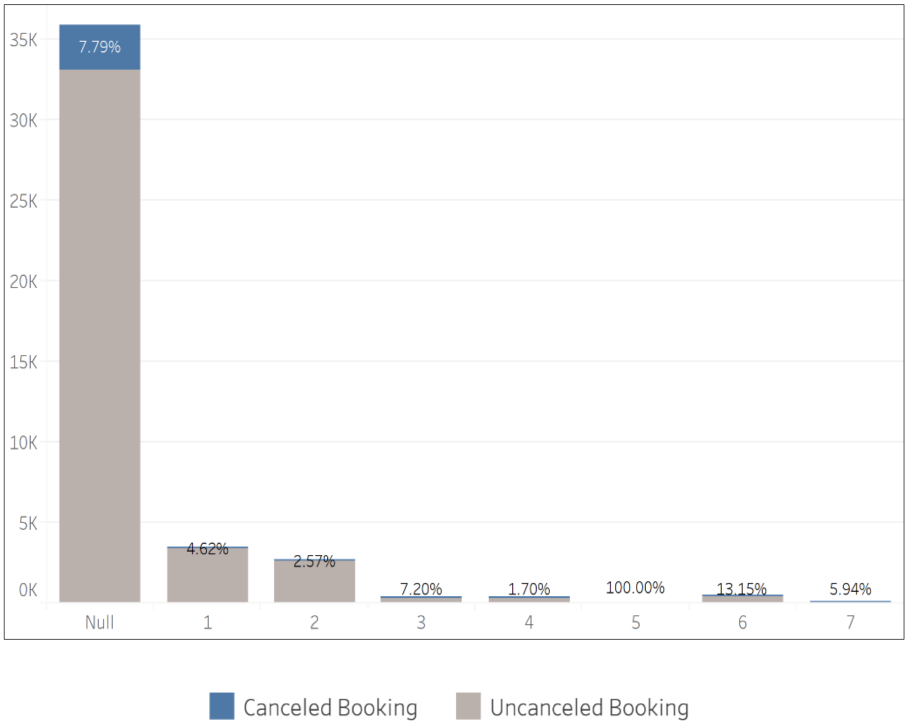
Cab Cancelations by **Day of Week:**

- Number of bookings made over the weekend are higher as compared to weekdays
- Overall Cab cancelation rate is high over the weekends, particularly on Sundays
- Wednesdays receive the least number of bookings as well as least cancelation rate



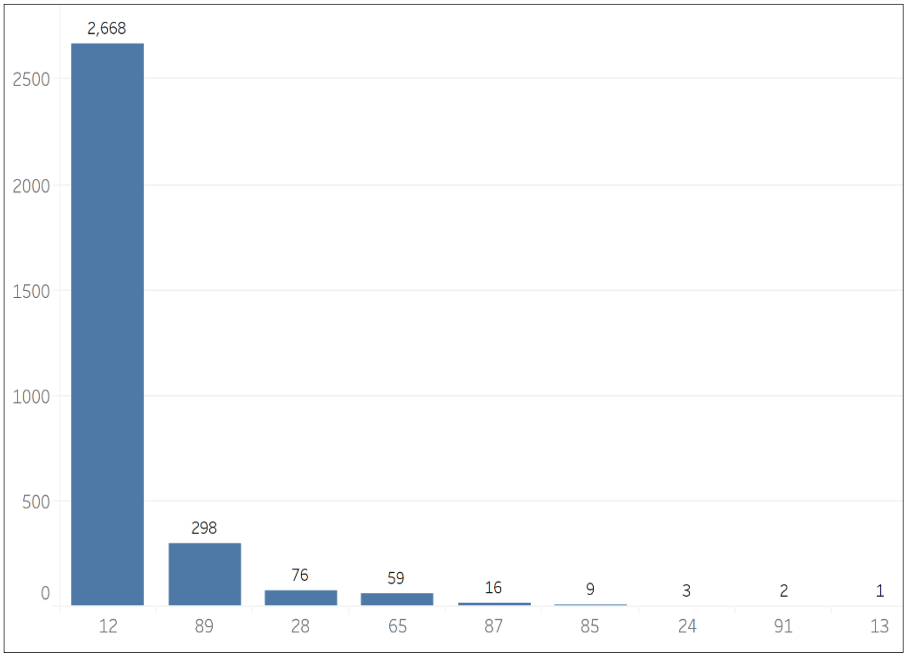
Cab Cancelations by **Travel Type:**

- Highest number of bookings are made under "point-to-point"
- "Point-to-point" travel witnesses highest cancelation rate as well
- Low cancelation rate in the "Long Distance" travel is not surprising as those would be pre-planned ones



Cab Cancelations by **Package Type:**

- Package Id is only applicable for hourly rentals
- “Null” package id corresponds to point-to-point & long-distance travels
- Highest cancelation rate observed corresponding to Null package id can be attributed to “point-to-point” travel



Cab Cancelations by **Model Id:**

- Certain cab models have highest number of cancelations compared to certain other models
- This indicates that model id can be one of the significant factors for cab cancelation predictions

Modeling

Analytical Dataset

We performed **feature engineering** to create new columns –

- **Hour** – To indicate the hour of the day the trip is scheduled to start
- **Mon, Tue……, Sun** – To indicate the day of the week
- **Trip distance** – To calculate the distance from the origin to destination using the haversine formula which determines the great-circle distance between two points on a sphere given their longitudes and latitudes
- **Wait time** – This is the difference from the time when the booking is created to the time when the trip starts
- **Mode_of_Booking** – Created a variable to indicate the mode of booking

In addition to this, we tried to impute from_city_id and to_city_id for records which have latitude and longitude information. Also package_id column has been modified to replace NA values with 0.

For forming the analytical dataset, we **dropped** the following columns -

- from_area_id - High variance column; Included **distance** as a new variable
- to_area_id - High variance column; Included distance as a new variable; Missing values
- weekday – **Created indicators for all days**
- from_date - Extracted day and hour
- booking_created – Does not influence cancellations
- from_lat - High variance column; Included distance as a new variable
- from_long - High variance column; Included distance as a new variable
- to_lat - High variance column; Included distance as a new variable; Missing values
- to_long - High variance column; Included distance as a new variable; Missing Values
- Cost_of_error – We are not estimating this variable in the analysis
- Id - High variance column; Indicator column
- user_id - High variance column; Indicator column
- time_of_day – **Created another column which indicates hour**
- is_weekend - Created indicators for all days

- from_city_id – High variance affecting model train and test split
- to_city_id - High variance affecting model train and test split

We also casted character variables into factor variables.

Data Sampling

As the dataset has class imbalance i.e. the proportion of records with cancellations are higher as compared to records without cancellations. We took a stratified sample.

Initially we tried to stratify it based on both columns ‘car_cancellation’ and ‘vehicle_type_id’ which provided a train dataset with class ratio of nearly 1:5.

When we fit a basic logistic regression model on this sample, we see a slightly higher drop in test performance as compared to the sample which is only stratified on ‘car_cancellation’. The AUC value is also quite similar in both the cases which indicates that the extra number of observations in the former sample is not really helping us explain higher variability. So we proceed with the sample stratified on just ‘car_cancellation’ for logistic regression model.

Logistic Regression

Logistic Regression is used when the dependent variable(target) is categorical. It is the go-to method for binary classification problems (problems with two class values). The coefficients are estimated using maximum likelihood estimation. It takes the form –

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

The advantages of using logistic regression is the easy interpretation of coefficients as it is a linear model. However, it might take a back seat in prediction accuracy when compared to advanced models such as random forest XGBoost etc. We initially ran the model with all the variables and checked the summary. We re-fit the model only with the significant variables as seen in the summary.

The significant variables which finally ended up in the model are

- Travel type id, Package id, Mode of booking, Trip Distance, Wait time, Tue, Wed, Fri, Hour

The performance for train and test of this model is shown below:

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	1572	615
TRUE = 1	603	1584

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	8549	3533
TRUE = 1	263	682

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.797	0.278	0.276
Test	0.779	0.291	0.278

AUC Curve

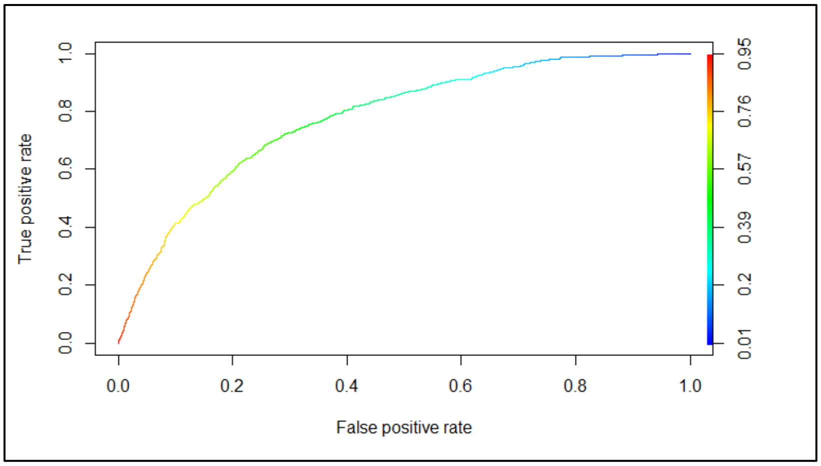


Figure 3 – ROC curve of logistic regression result on test data

The misclassification rate goes up by 1% and FNR has a negligible increase on the test dataset. We have a decent performing model with AUC value of 0.78 and an FNR of 28%.

Stepwise Selection – BIC

As a next step, we performed step wise selection with BIC criterion. This increased the AIC value of the model from 4,854 to 4,978. However, the BIC model has 3 fewer significant predictors and is a parsimonious model when compared to the initial model we built. We wanted to check the performance before we take a call on the model to go ahead with –

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	8503	3579
TRUE = 1	274	671

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Test	0.763	0.296	0.290

AUC Curve:

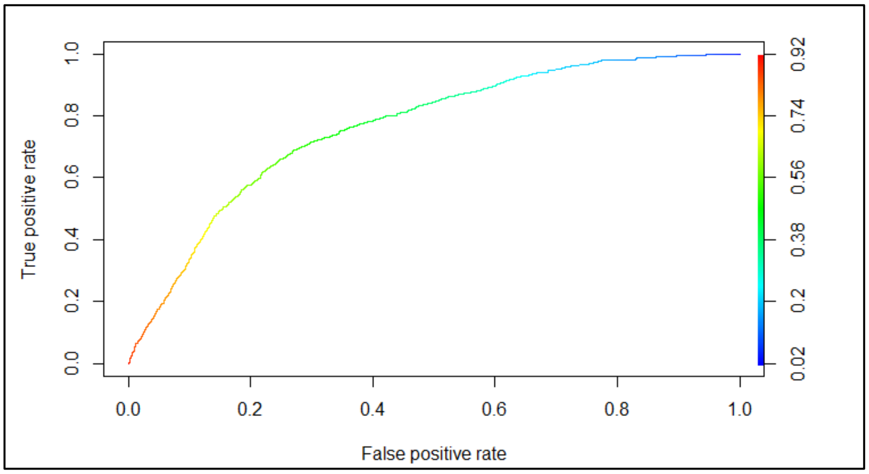


Figure 4 - ROC curve of logistic regression with BIC stepwise selection on test data

The performance does not deteriorate too much with the new model obtained from backward selection and hence we choose this model as it is simpler with 3 fewer variables as compared to the prior model.

Classification Trees

Classification trees are used to predict membership of cases or objects into classes of a categorical dependent variable from their measurements on one or more predictor variables. A Classification tree labels, records, and assigns variables to discrete classes. A Classification tree can also provide a measure of confidence that the classification is correct. A Classification tree is built through a process known as binary recursive partitioning. This is an iterative process of splitting the data into partitions, and then splitting it up further on each of the branches.

Advantages: A significant advantage of a decision tree is that it forces the consideration of all possible outcomes of a decision and traces each path to a conclusion. Also, a Decision trees model is very intuitive and easy to explain to technical teams as well as stakeholders.

Dis-advantages: A small change in the data can cause a large change in the structure of the decision tree causing instability.

Stage 1 – Classification Decision Trees:

Data

- We have implemented stratified sampling on the data on the variables ("Car_Cancellation","vehicle_model_id") and have 1:5 ratio of the training and the validation datasets. Also, the rows with NAs are omitted.
- Current dimensions of the data are: Observations: 12,177, Variables: 15
- Column names are: ["vehicle_model_id", "travel_type_id", "Car_Cancellation", "package_id_new", "mode_of_booking", "trip_dist", "wait_time", "mon", "tue", "wed", "thu", "fri", "sat", "sun", "hour"]

Model

We have implemented a classification decision tree using the "**rpart**" package in R. The rpart package takes a formula argument in which you specify the response and predictor variables, and a data argument in which you specify the data frame. We have used the 'rpart' library for model building and 'rpart.plot' for plotting.

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	9516	1010
TRUE = 1	466	1185

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	10337	423
TRUE = 1	1756	514

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.7466	0.12	0.46

Test	0.7017	0.16	0.45

The misclassification rate goes down by 4% and FNR has a negligible decrease on the test dataset. We have a low performing model with AUC value of 0.70 and an FNR of 45%.

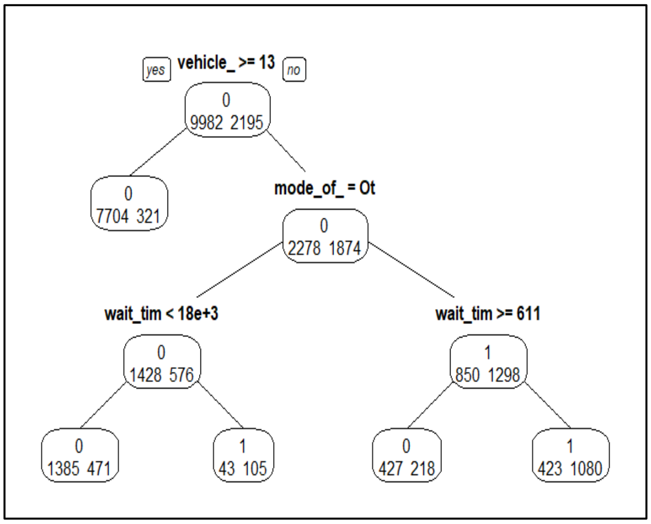


Figure 5: Decision Tree plot for base model

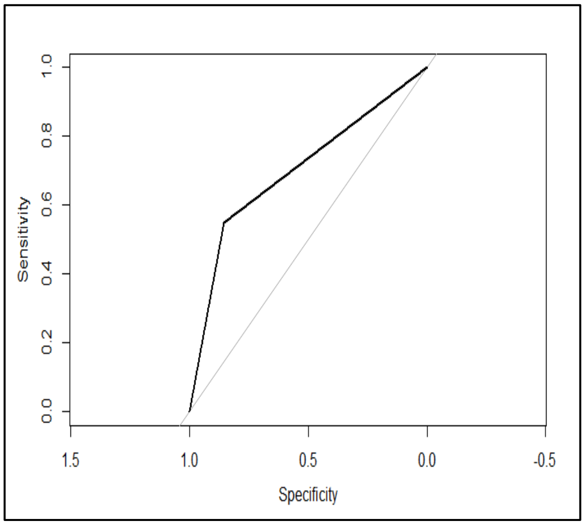


Figure 6: ROC curve of CART on test datal

Stage 2 – Pruned Decision Trees:

Since the FNR value is high, we tried to prune the tree and tried to achieve better accuracy metrics. Pruning is the process of removing leaves and branches to improve the performance of the decision tree when moving from the Training Set (where the classification is known) to real-world applications (where the classification is unknown). The tree-building algorithm makes the best split at the root node where there are the largest number of records, and considerable information. Each subsequent split has a smaller and less representative population with which to work. Towards the end, idiosyncrasies of training records at a node display patterns that are peculiar only to those records.

We have built a cp value plot the number of trees to identify at which value of size of tree we can prune. You can observe from the below graph that the cross-validation error (x-val) does not always go down when the tree becomes more complex. The analogy is when you add more variables in a regression model, its ability to predict future observations not necessarily increases. A good choice of cp for pruning is often the leftmost value for which the mean lies below the horizontal line.

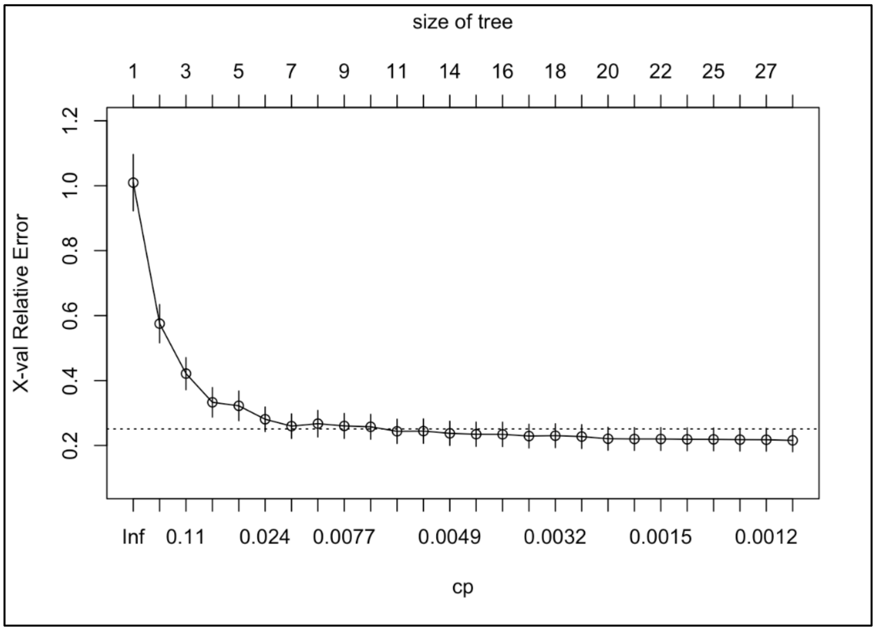


Figure 7: Cp value versus the size of the trees

Here the cp value chosen based on the above analysis is 0.008 and we have built a new model based on the same. But when compared with the original, we have realized it the same as the base model tree.

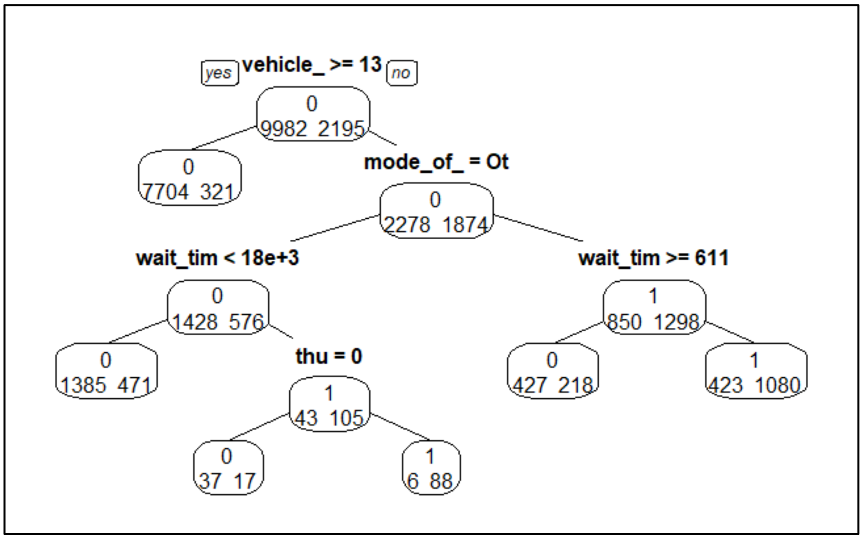


Figure 8: Pruned tree with cp=0.008

Stage 3 – Random Forest:

In addition to simple decision trees, we have also implemented Random forests for better model performance and accuracy.

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

The idea of random forests is to randomly select m out of p predictors as candidate variables for each split in each tree. Commonly, m=√P. The reason of doing this is that it can decorrelates the trees such that it reduces variance when we aggregate the trees.

By default, m=p/3 for regression tree, and m=√p for classification problem.

We have created a plot below for "OOB Error", "FPR", "FNR" and compared values across multiple trees.

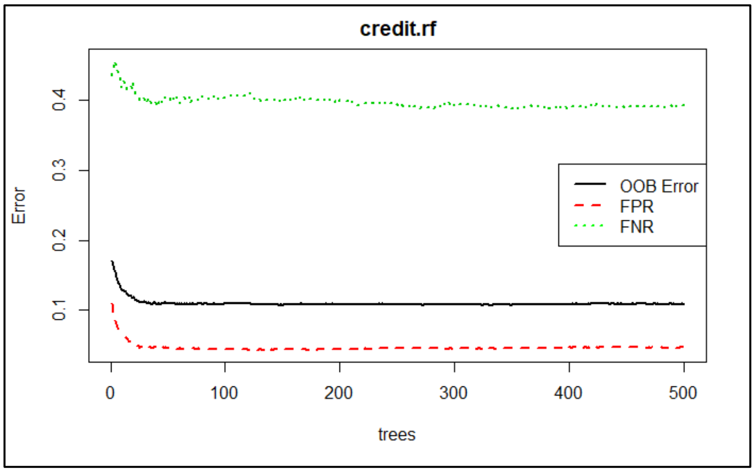


Figure 9:Comparision of OOB error, FPR and FNR across trees

As we can see, the FNR is very high, just as the confusion matrix. Below we have used type="prob" to get predicted probability and then find optimal cut-off. We have done in-sample prediction and find optimal cut-off.

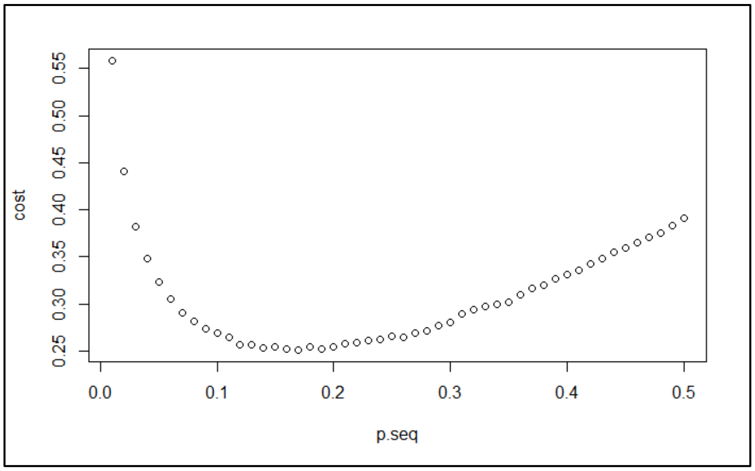


Figure 10: Finding the optimal cut-off value for the random forest model

From the above plot, we have found out that the **optimal pcut value is 0.17**

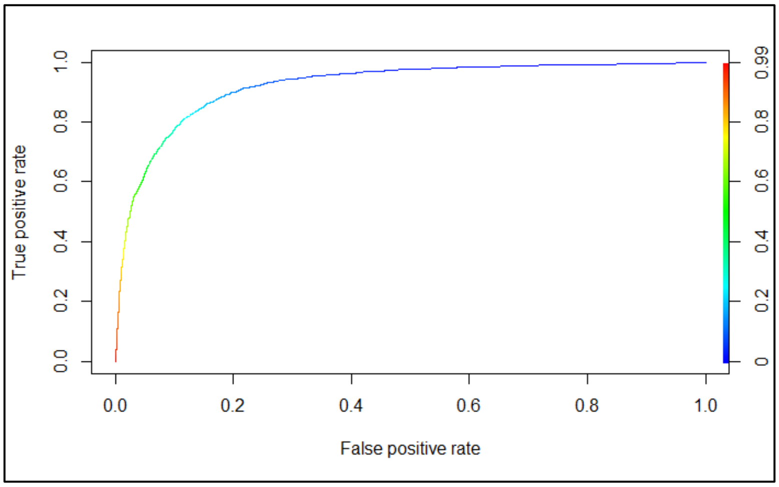


Figure 11: AUC curve for the training data using the random forest model

Below is the plot for the important features in the model:

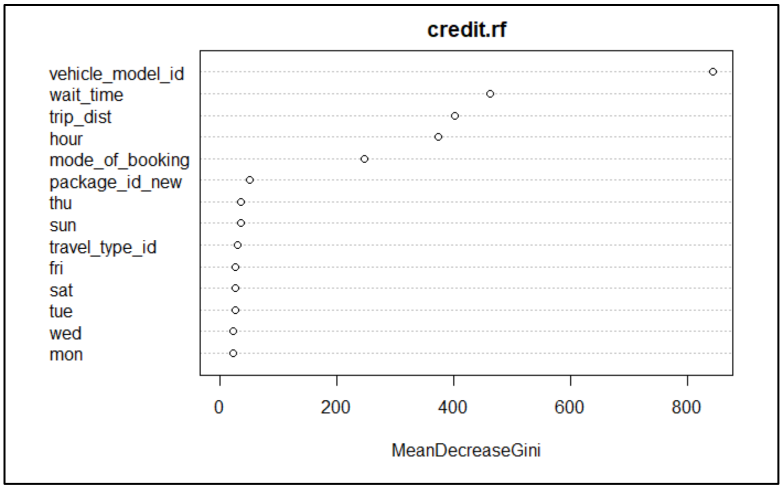


Figure 12: Important features in Random Forest model

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	9824	447
TRUE = 1	158	1748

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	10426	364
TRUE = 1	1667	573

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.89	0.04	0.20
Test	0.73	0.15	0.38

Stage 4 – Random forest using Grid Search:

Tuning algorithm is important in building modeling. In random forest model, you cannot pre-understand your result because your model is randomly processing. Tuning algorithm will help you control training process and gain better result. In this study, we will focus on two main tuning parameters in random forest model is mtry and ntree. Besides, there are many other methods but these two parameters perhaps most likely have biggest affect to model accuracy.

- mtry: Number of variable is randomly collected to be sampled at each split time.

- ntree: Number of branches will grow after each time split.

We have built a grid search model with mtry values varying from 1 to 15. Below plot describes the accuracy versus different mtry values. We can see that the graph flattens after mtry value of 6. Thus there would be no further advantage in increasing the mtry value

Thus, the ideal model would be with an accuracy of 89%.

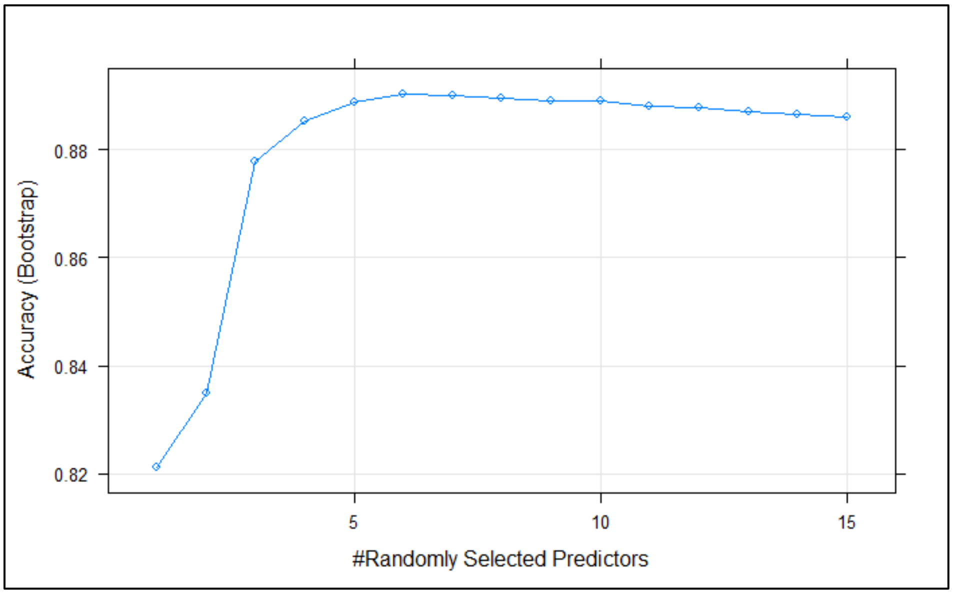


Figure 13:Accuracy versus different mtry values

Below are the model measurement metrics:

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	9819	567
TRUE = 1	163	1628

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	10620	388
TRUE = 1	1473	549

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.8627	0.05	0.25

Test	0.7325	0.14	0.41

Support Vector Machines

Support Vector Machines is a supervised classification algorithm, with roots in Machine Learning literature to handle classification (and more recently, regression problems). It has the tendency to handle non-linearity, have few parameters to tune and can handle significant data overload. SVMs are very good when there is high-dimensionality and relatively less idea about the data. There is less risk of over-fitting and works relatively well in classification problems when there is a clear margin of separation. Though it is tough to understand the final model, variable importance and inference, it produces good predictions/classifications.

SVM takes two functions, the cost (misclassification) or training samples, and the gamma parameter, which gives an estimate of what influence one sample has over another. The Cost is 1 (equal weightage) and the Gamma value is set as 1/length(dataset). The results of the SVM model are as follows -

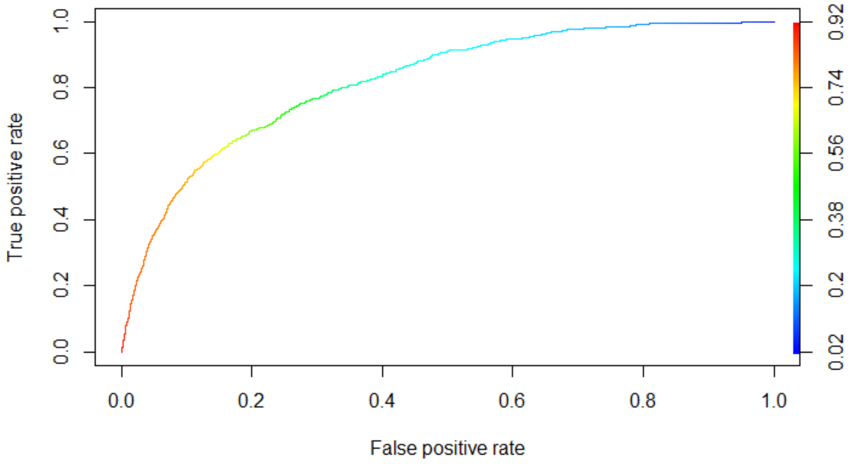


Figure 14 : ROC Curve of SVM Model on test data

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	1667	501
TRUE = 1	541	1627

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	9098	2968
TRUE = 1	272	692

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.846	0.240	0.249
Test	0.819	0.248	0.282

SVM model presents with an overall AUC of 0.81 in the testing dataset, with a Misclassification Rate of 0.25 and FNR of 0.28. While the AUC and FNR are better than the other model results, the MR is lower than the best achieved yet. As is visible from the training diagnostics, the train metrics are like the test results, showing that there is little evidence of model over-fitting.

Neural Network

Neural Nets is a process to fit extremely flexible models to the data consisting of large number of predictors. This is a black-box machine learning technique that can be used for prediction but have very low inference.

The Neural network was built using decay parameter of 0.1 and with a max iteration of 500. The result contains 1 hidden layer with 3 nodes as is below.

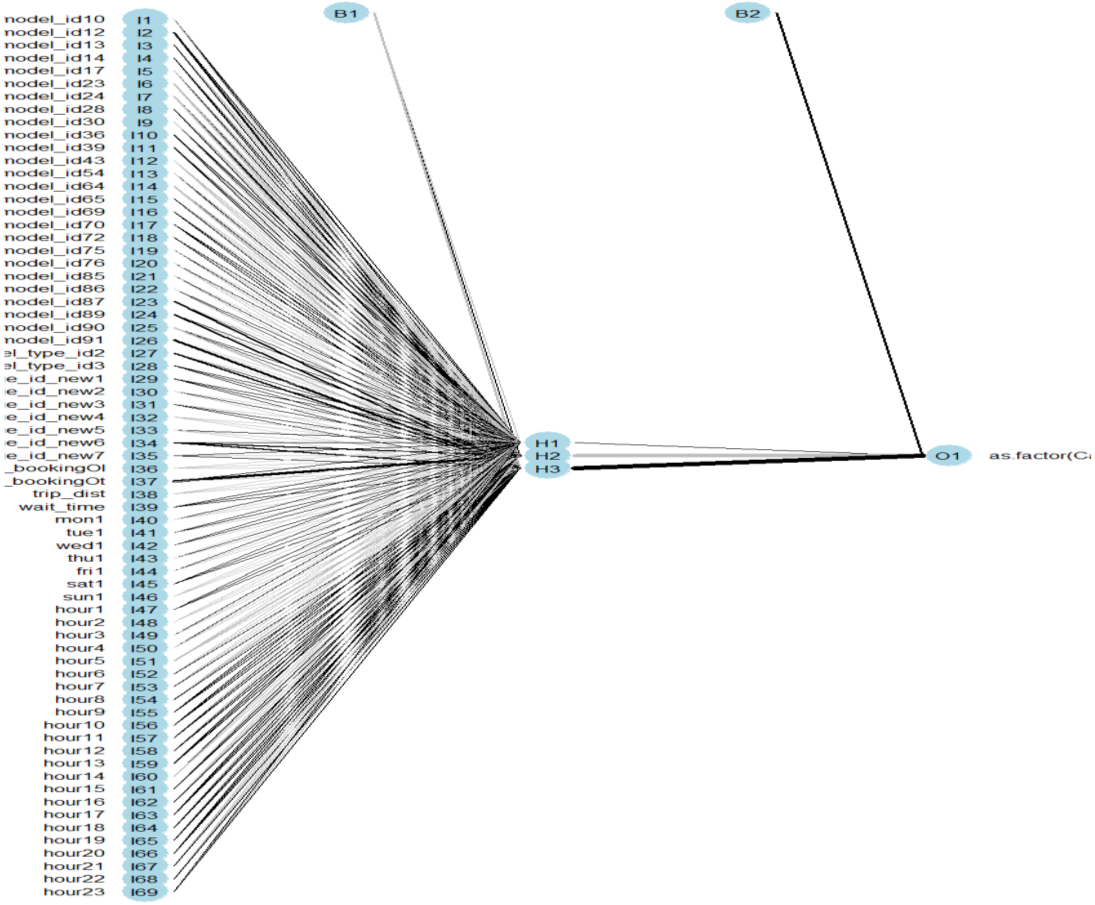


Figure 15- Neural Net Model Representation

The summary of the results from the neural network is as below.

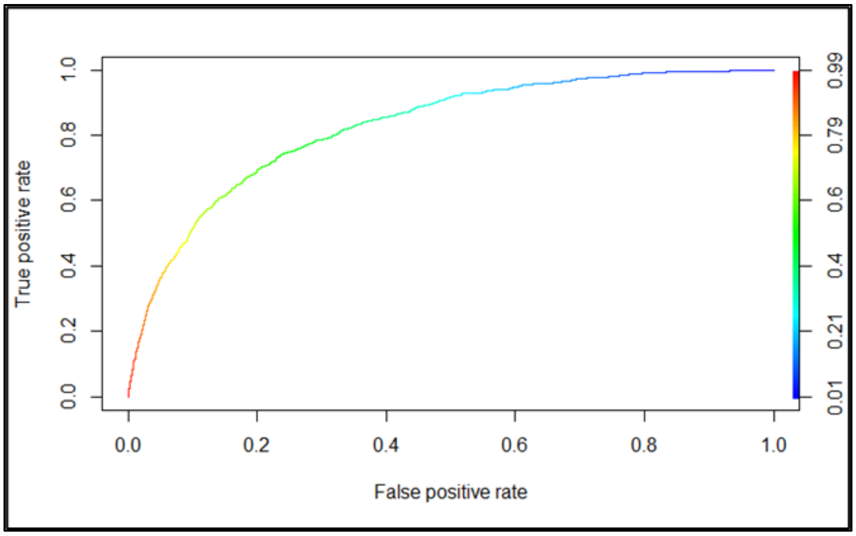


Figure 16 - ROC curve of Neural Net model on train data

Confusion Matrix:

Train:

	Predicted = 0	Predicted = 1
TRUE = 0	1654	514
TRUE = 1	473	1695

Test:

	Predicted = 0	Predicted = 1
TRUE = 0	9052	3014
TRUE = 1	243	721

Summary Table:

	AUC	Misclassification Rate	False Negative Rate
Train	0.850	0.227	0.218
Test	0.826	0.249	0.252

The results of the Neural Network are like that of the SVM output with similar predictions and mis-classification rates. The Neural Network model in fact has a better False Negative Rate.

SMOTE Resampling

Till now, the modelling approach involved taking a stratified sample from the data. The main purpose of course was to counter the imbalance in the data. Classification algorithms are generally biased towards the class that causes the imbalance (majority class). A general approach to deal with this problem, as was undertaken till now, is to under-sample the data to create a training dataset that includes equal results.

Synthetic Minority Over-Sampling Technique (SMOTE) is an approach used to resample from an imbalanced dataset to create a training dataset that is relatively balanced. This avoids the need for over-sampling, under-sampling or specification of a mis-classification error rate.

SMOTE synthesis new minority samples between the existing minority samples. In short, in a looped manner, from one of the minority class sample a k-closest neighbor is chosen and new minority instance is synthesized between the two samples.

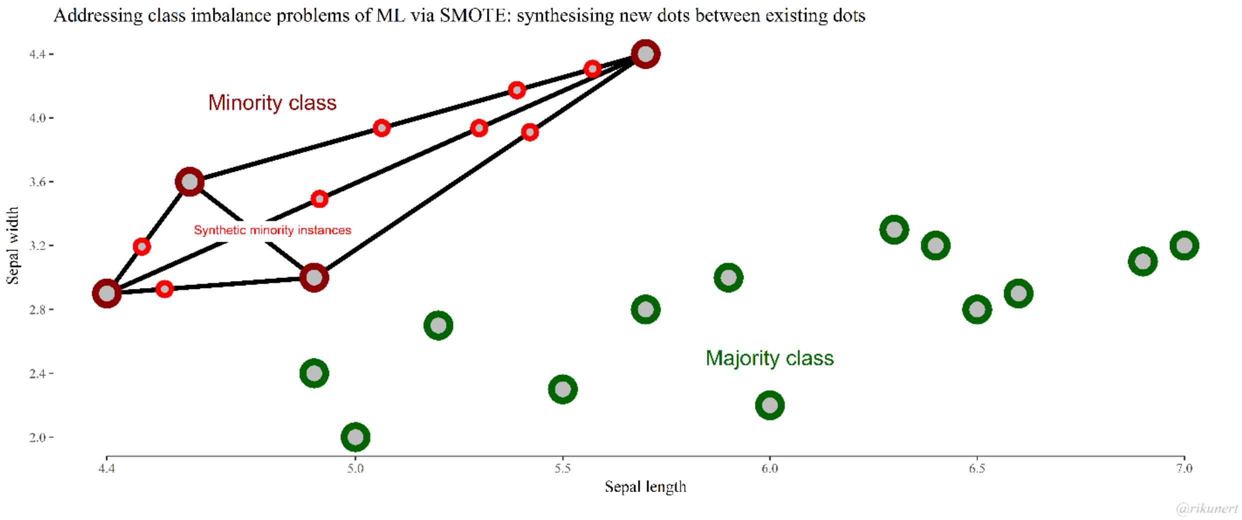


Figure 17 - Representation of SMOTE Synthesis

The original dataset consisted of ~3000 cancellations vs ~40000 non-cancellations, resulting in a huge sampling issue for model tuning. After applying the SMOTE algorithm, the analytical dataset was reduced to key 20000 samples, that were split as ~9000 cancellations and ~12000 non-cancellations. The training and testing split were carried forward using this dataset, and all the models were built again for comparison. A brief of the model result is as below.

	Training Parameters				Testing Parameters			
Models	**MR**	**FPR**	**FNR**	**AUC**	**MR**	**FPR**	**FNR**	**AUC**
Logistic Regression	0.28	0.20	0.38	0.78	0.27	0.20	0.38	0.78
Decision Tree	0.18	0.06	0.34	0.79	0.18	0.06	0.33	0.79
Pruned Decision Tree	0.17	0.08	0.28	0.81	0.17	0.08	0.29	0.81
Boosting	0.14	0.08	0.22	0.84	0.15	0.10	0.21	0.84
SVM	0.17	0.13	0.23	0.89	0.18	0.14	0.23	0.88
Neural Net	0.23	0.19	0.29	0.77	0.24	0.20	0.29	0.74

Conclusion

Through this project, multiple modelling techniques were explored to see the fit adequacy on the model dataset and the results were compared in order to reach a consensus. Few algorithms looked to perform better than the rest, for example Random Forest algorithms or SVM. As is the case, based on the project, the Random Forest model seems appropriate and adequate, but there are other options to be explored.
Two main thoughts from the project execution are as below –

1. Better feature engineering and selection could improve the model development process. The data is geo-code rich which could be further explored (for eg., calculating the actual travel distance instead of Manhattan distance). There are multiple columns that are categorical in nature (which is perhaps why the SVM algorithm performs well), which could be mined to get further use (eg. Group the waiting time, calculate Time of Day variable differently).
2. The major challenge with the data, as is in many other cases, is the significant imbalance in the classes. While one alternate approach (SMOTE) was considered, there are other options, like presenting with a misclassification cost or advanced algorithms like ADASYN.

References

- https://www.kaggle.com/c/predicting-cab-booking-cancellations/
- https://yourstory.com/2013/09/yourcabs-rajath-kedilaya-car-rental-online-booking/
- http://rikunert.com/SMOTE_explained