**Ping Tree**

**Name** – **Megharjun Mallela Srinivasa**

1. **Problem**

* Finding an optimal bid price to win the lead and to make a profit on it.
* Building a model to predict if the bid gets accepted or not and creating a rule set to increase the net revenue.

This is a binary classification problem and my analytical strategy is to compare the performance of classification algorithms namely AdaBoost, Logistic Regression, Naïve Bayes, Random Forest, Decision Trees, Neural Network, K nearest neighbors and SVM on the given dataset. These Classification algorithms will help us predict if a bid at a certain bid price, expected revenue and conversion gets accepted or rejected.

1. **Dataset**

The Soarem\_Managment\_Lead\_bid\_test dataset contains the following columns/Attributes

* Id – This Attribute consists of a unique individual key through which rows can be identified.
* AcceptedBid – This column consists of 0’s or 1’s. 0 for bid being rejected and 1 for bid being accepted.
* BidPrice – This column contains the price in dollars placed to win a bid and convert the lead into loan.
* ExpectedRevenue – This feature contains the amount of revenue we could expect to get from the lead if it turns into a loan.
* ExpectedConversion – This Attribute contains the expected conversion rate of lead to loan.

The above dataset is converted into a Lead\_Bid dataframe

The Lead\_Bid contains 53950 rows and 5 variables/attributes in total

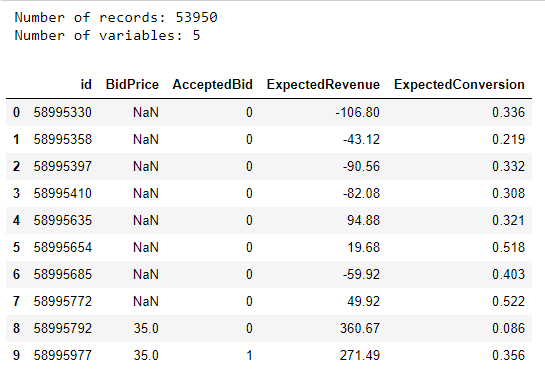


Fig-1

1. **Data Cleaning and Visualization**
   1. **Data Cleaning**

The heatmap below shows all the columns containing null/missingvalues.

Yellow blocks signify missing/null values.

A copy of the Lead\_Bid data frame is created called New\_Lead\_Bid and Data cleaning and Visualization is performed on this data frame keeping the original data frame unchanged

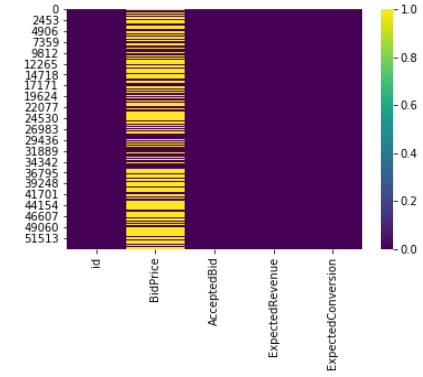


Fig-2

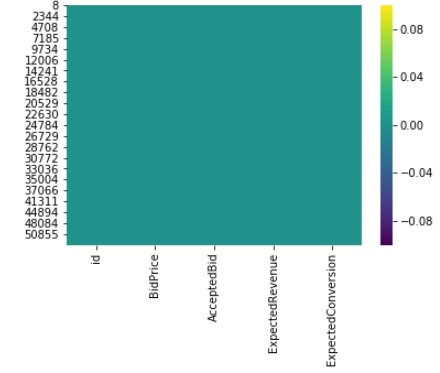
From the above heatmap we can conclude that only column BidPrice has a huge number of missing values. This is because no bids were placed in these rows. Since we want to Build/evaluate our model only with the rows containing BidPrice, we can go ahead and drop the null values. Fig-3 is a heatmap after dropping the null values

Fig-3

Since Id is unique for every row and does not play any role in predicting the dependent variable(AcceptedBid) we can go ahead and remove the column.

Dataset after removing the id column and the null values contains

Number of records: 21946

Number of variables: 4

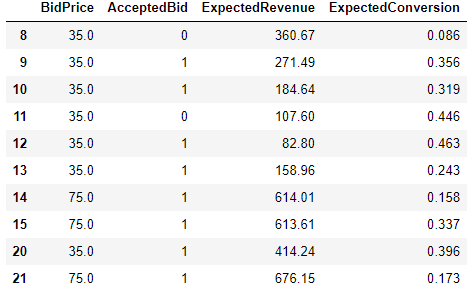


Fig-4

* 1. **Visualization and Exploratory Data Analysis**

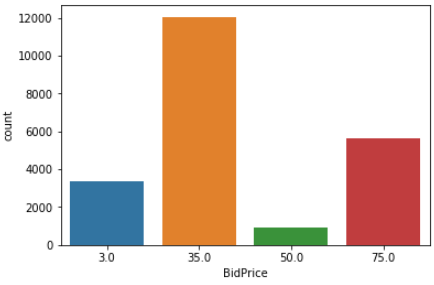
Performing Exploratory Data Analysis is the key to find insights from a dataset. From fig-5 we can see that more bids were placed at 35$(12,000 bids approximately) and very few bids were placed at 50$(1,000 rows approximately).

Fig-5

From fig-6, we can see that the bids placed at 75$ have the highest acceptance rate or the least rejection rate when compared to 35$ and 50$ bid prices. We can also see that bids placed at 3$ have the least acceptance rate. Fig-6 also shows the number of bids accepted or rejected grouped by their individual bid prices

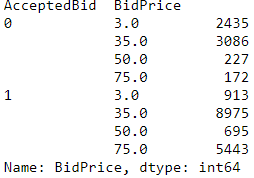
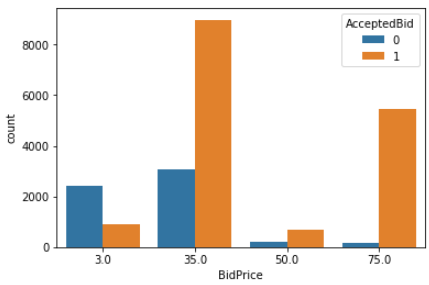


Fig-6

Fig-7 is a Correlation Matrix which expresses the extent to which two variables are linearly related. From the figure, we can see that between BidPrice and ExpectedRevenue there is a high positive correlation and between ExpectedConversion and BidPrice there is a low negative correlation. This makes sense as Higher the revenue we tend to bid at a higher price and these would generally have a low conversation rate

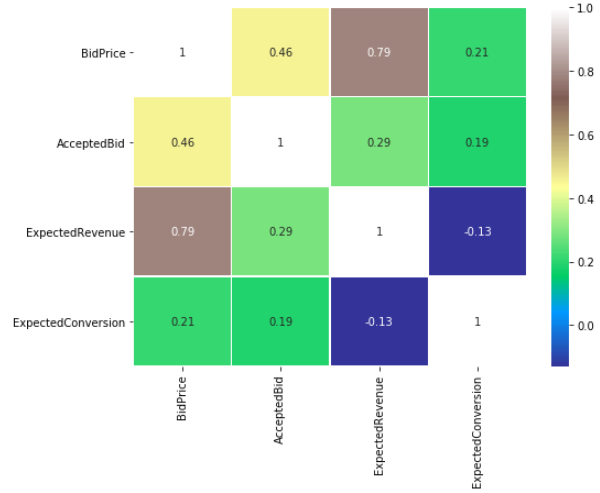


Fig-7

The describe () function computes a summary of statistics pertaining to the New\_Lead\_Bid Data Frame columns such as min, max, mean, count etc.



Fig-8

1. **Approach and Methodology**

The main goal of developing a machine learning model form our New\_Lead\_Bid dataset is to predict if a bid at a certain price, expected revenue and expected conversion gets accepted (1) or rejected (0). Therefore, we consider AcceptedBid(y) to be our dependent variable and BidPrice(X1), ExpectedRevenue(X2) and ExpectedConversion(X3) to be our independent variables. Since the dependent variable is binary we will develop a classification algorithm.

The overall work Flow is as show in Fig-9

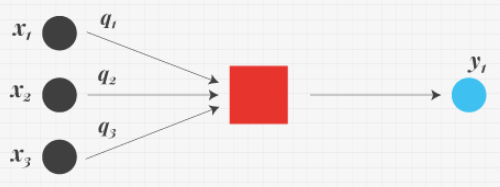




Fig-9

* 1. **Data Preprocessing**
* **Data Cleaning** – The data set was cleaned while performing visualization. Refer to fig-2, Fig-3, Fig-4
* **Splitting the data** – Now that we have a cleaned dataset(New\_Lead\_Bid) we divide the data set into X, Y variables. X containing independent variables such as BidPrice(X1), ExpectedRevenue(X2) and ExpectedConversion(X3). Y Containing AcceptedBid (target variable).

We Further divide X, Y variables into train set and test set for modeling purpose.

33% of the entire data set is allocated to test set while the remaining 67% is allocated to trainset

* **Normalization** – Data needs to be normalized or standardized before applying to machine learning algorithms. Standardization scales the data and gives information on how many standard deviations the data is placed from its mean value. Effectively, the mean of the data is 0 and the standard deviation is 1
  1. **Modelling and Predicting with Machine learning**

Python has a library called Scikit learn which is used to formulate all Machine Learning Algorithms

* **Logistic Regression** – Logistic Regression is one of the basic classification models which uses a logistic function to predict the probability of a certain class. This is one of the most popular models which is used on categorical data, especially for binary response data.

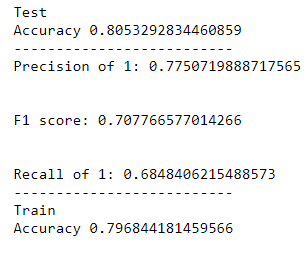
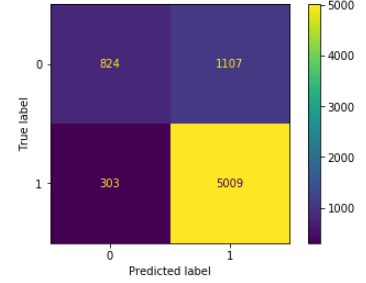


Fig-10

* **Decision Trees** - A decision tree is a map of the possible outcomes of a series of related choices. It allows an individual or organization to weigh possible actions against one another based on their costs, probabilities, and benefits.

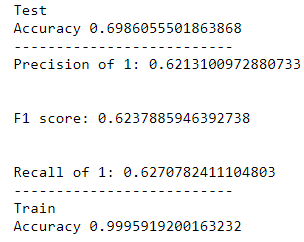
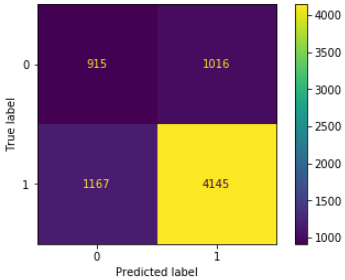


Fig-11

* **Random Forest** – Random forest is an ensemble machine learning method that models by creating multiple decision trees and classes with the majority votes becomes the models prediction.

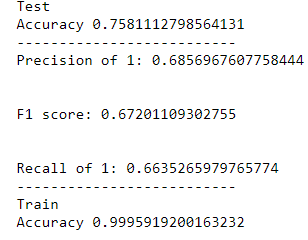
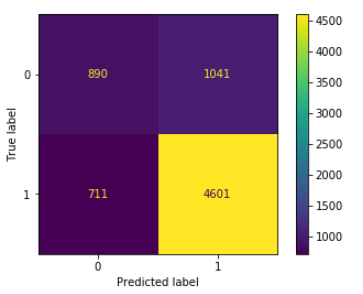


Fig-12

* **K-Nearest Neighbors (KNN)** – KNN algorithm is a non-parametric method used for classification and regression. The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to a new point and predict the label from these**.**

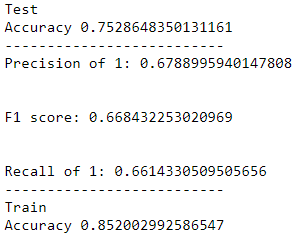
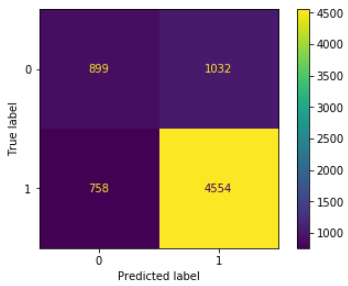


Fig-13

* **Support Vector Machine (SVM)** – Support Vector Machines are perhaps one of the most popular machine learning algorithms. They are the go-to method for high performing algorithm with a little tuning.

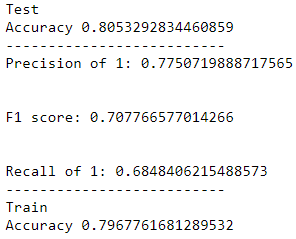
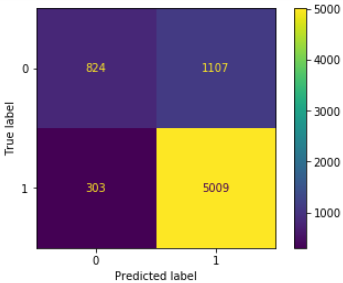


Fig-14

* **Neural Network** – Neural Network are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input.

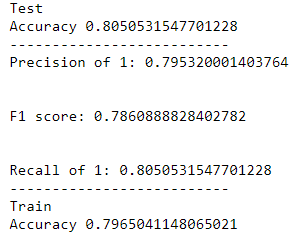
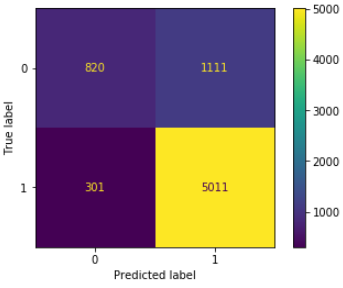


Fig-15

* **Naïve Bayes** - Naïve Bayes is yet another common classifier based on Bayes theorem and considers all features to be independent. Every feature is given equal weights while predicting the probability of each class.

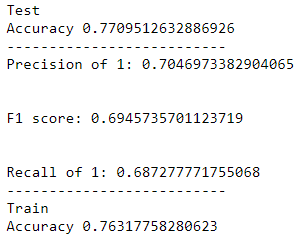
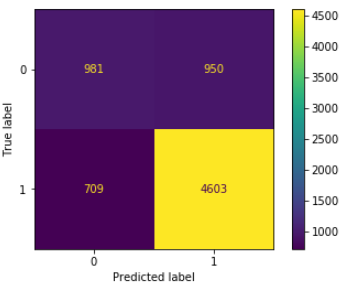


Fig-16

* **AdaBoost –** AdaBoost is a boosting algorithm which converts weak learner into a strong learner by giving more emphasis on misclassified samples. It is commonly used with decision tree as a weak learner.

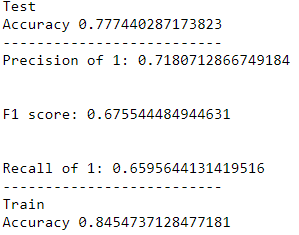
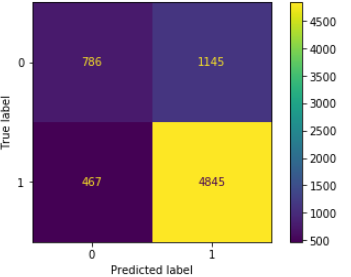


Fig-17

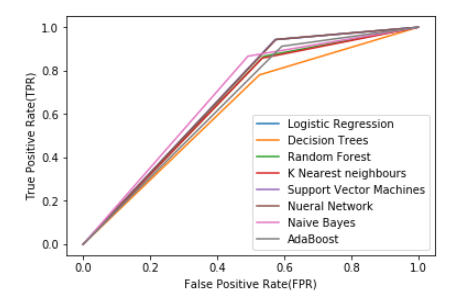
**Classification algorithms that worked**

* Neural Network, Logistic regression, Svm have High train, test accuracy
* This was expected since all the above 3 are good very good classifier algorithms
* Neural Network has an upper hand over Logistic regression and Svm due to the high Precision, F1 score and recall scores.

**Classification algorithms that did not work**

* Decision tree, Random forest was overfitting its train set with 99% accuracy these algorithms do not perform well while predicting a new data set
* K-nearest neighbors, Naïve Bayes and adaboost have average train and test set accuracy but comparatively less Precision, F1 score and recall scores.

**ROC curve and Auc**

****

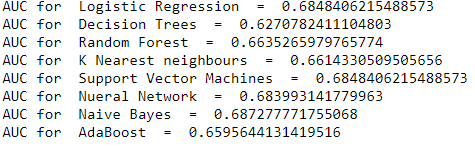


Fig-18

* 1. **Evaluation and Selecting a Model**
* **Confusion Matrix** – It is a table used for classification which shows the number of true negatives, true positive, false negative and false positive. The confusion matrix of their respective models are shows in Fid-8,9,10,11,12,13,14
* **Accuracy** – It is the ration of correct predictions to the total number of observations. We take Both the Train and test accuracy to check for over fitting and under fitting. If the training model shows 95% or greater than the model is overfitting and becomes sensitive to outliers. Underfitting occurs when model fails to learn the relationships in the training data.

Accuracy = TP+TN/(TP+FP+FN+TN)

* **Precision** – It is defined as the proportion of the number of true positives to the sum of true positives and false positives. Precision takes false positive into consideration; Thus, it is very important metric when it comes to predicting AcceptedBid because it is important that our model does not wrongly misclassify Accepted Bid(1) when the bid is rejected. Higher the precision better the model. We predict precision on only Accepted Bid(1) since Rejected Bid(0) is automatically calculated as we have only 2 classes that is 1 and 0
* **Recall** – Recall is the ration of the number of true positives to the sum of true positives and false negatives. Recall is the ability of the model to find all the data points of relevance in a dataset.
* **F1 Score-** It is a function of both Recall and Precision.
* **Macro** - In Precision, Recall and F1 Score average is initialized to macro since macro-averaging does not take class imbalance into account and the f1-score of class 1 will be just as important as the f1-score of class 2
* **Roc Curve** - AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s

**Conclusion**

Considering Test set accuracy, Precision, Recall, F1 Score and Roc Curve(Auc) **Neural Networks** is the best classifier for out dataset. Hence, I am choosing Neural Network Classifier(NN) for classification of my dataset.

1. **Rule Set**

Now that we have a classifier selected and trained to predict whether a bid is accepted or rejected, we test the classifier with a new data set formed by a set of rules.

We create a new data frame called New\_Lead\_Bid\_With\_Null which is another copy of the original data Frame Lead\_Bid

* We create a new column in our New\_Lead\_Bid\_With\_Null data frame called Revenue which is the product of ExpectedRevenue and ExpectedConversion.
* The null rows in New\_Lead\_Bid\_With\_Null data frame are not removed
* Id column which we removed in the previous dataset is not removed as we have use for it
* Below figure shows the first 5 rows in New\_Lead\_Bid\_With\_Null data frame

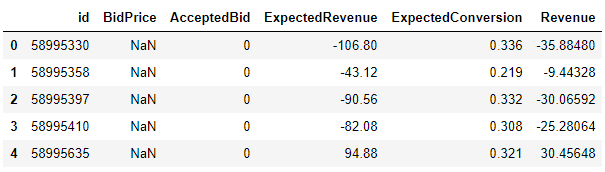


Fig-19

* We create a new column called the New\_BidPrice based on the following rules

1. If the Revenue in New\_Lead\_Bid\_With\_Null Data frame is greater than or equal 130$ we bid 75$. Since 75$ is the max we can bet, Higher the bid price more the risk therefore we bet only when we make a profit of at least 55$ or more
2. If the Revenue in the Data frame is lesser than 130$ and greater than equal to 90$ we bid 50$. The min profit we make here is 40$ and the max profit we make is 79$
3. If the Revenue in the Data frame is lesser than 90$ and greater than equal to 50$ we bid 35$. The min profit we make here is 15$ and max profit is 54$
4. If the Revenue in the Data frame is lesser than 50$ and greater than equal to 8$ we bid 30$. The min profit we make here is 5$ and the max profit we make here is 46$
5. Else we bid 0.

**Code-**

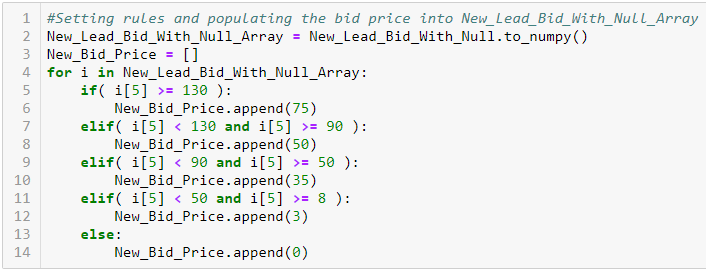


Fig-20

1. New\_BidPrice column is added to New\_Lead\_Bid\_With\_Null data frame

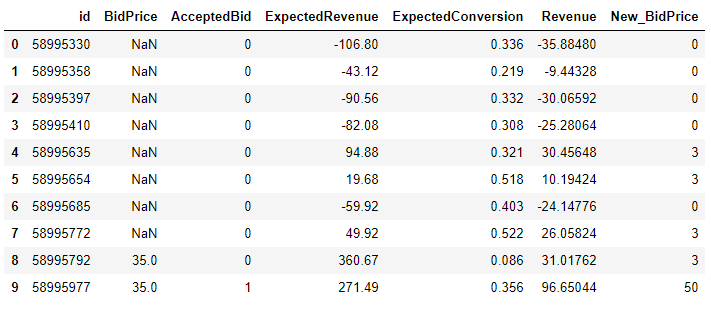


Fig-21

1. Removing Rows where New\_BidPrice is 0 or less

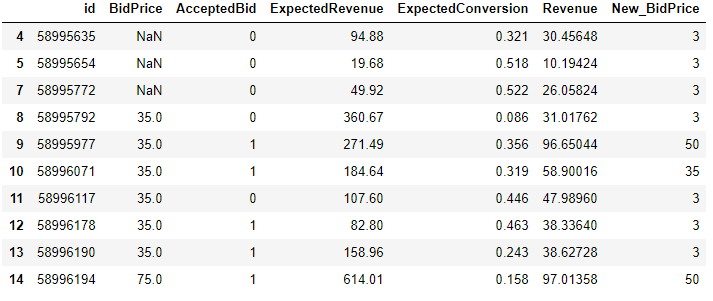


Fig-22

1. **Fitting New Data Frame into our Model and Final Net Revenue**

* A copy of New\_Lead\_Bid\_With\_Null is created called New\_Lead\_Bid
* BridPrice, AcceptedBid, Revenue is removed, id is removed from the New\_Lead\_Bid data frame

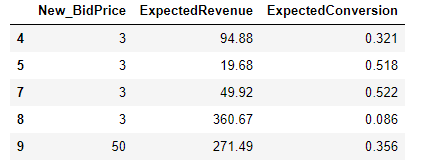


Fig-23

* scalar transformation is done to the above 3 columns
* We now consider the 3 columns in the data frame to be X1, X2, X3 and predict these values in the already trained neural network classifier
* Once we get the Acceptedbid output we add this column to our New\_Lead\_Bid\_With\_Null and remove BidPrice and AcceptedBid

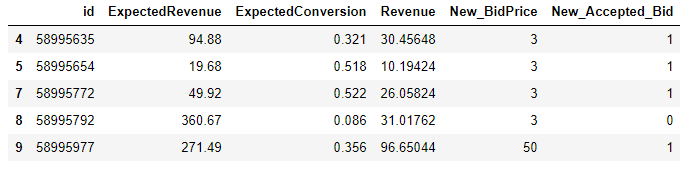


Fig-24

* Now we can calculate the New\_net\_revenue by subtracting Revenue with New\_BidPrice and we rearrange the table for max New\_Net\_revenue and consider only those bids that our model predicted to be accepted. We end up with a table like the one below
* All these above changes are made and is sorted in a separate data frame called No\_positive\_lead Data frame. The head() for the No\_positive\_lead(Final Data Frame) is shows below in fig-25

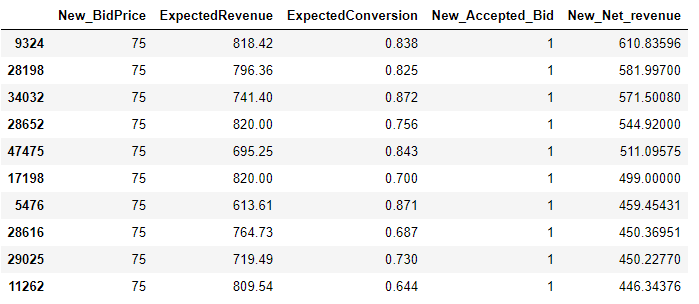


Fig-25

* Since only a specific volume of leads can be handled by the call center we get our revenue from the top 16827 rows
* By summing the New\_Net\_revenue, the revenue generated was found to be 1293282$ which is 85,628$ more than the revenue generated without using my model
* (85,628/1207654) \*100 = 7.09% increase in revenue.