Northeastern University

CS6220 - Data Mining Techniques

Online Shoppers Purchasing Intention

Team Members:

Rajath Kashyap

Mukund Wagh

Bishwarup Neogy

Abstract

Two-thirds of businesses today invest in online marketing because consumers spend so much time online and hence to make sense of the activity of users on the internet has become more important than ever. With more than 4.1 billion internet users in the world, online advertising plays a huge role in the success of any business. Although the boom of e-commerce has created a lot of potential in online marketing, most users do not complete their shopping process. It is very important for online businesses to maximize the number of people who complete the transaction to increase their revenue. There are several reasons for this shopping cart abandonment and in this project we aim to analyse and evaluate the users activity on e-commerce sites and try to predict how this affects the users shopping intent. We use features from 12,330 user sessions data collected over 1 year period and use them to build machine learning classification model. Preprocessing and feature sampling this data and using them in various machine learning models like Decision Trees, Support Vector Machines, Neural Networks etc, we will try to increase the performance of the classifier. We will get interesting insights on how user activity correlates to the revenue of the business.

Introduction

Given session data of users, analyze the user activity and try to predict if that user will buy the product thus generating revenue for a business. We plan to extract a meaningful insight of how to make this online retail shop a more successful business.

Why is it important to solve this problem?

The online market has a great economic impact for the development of a country and also for any business. The recent statistics show that the global mobile ecommerce revenue is expected to reach upto $670 billion which is just on mobile phones. Though this is a staggering statistic, it is also recorded that more than 90% of the people who visit online e-commerce sites, leave without actually making a transaction. Even when they have an intention of making a purchase , many users do not finish the transaction and just abandons their cart. It becomes a major revenue loss for online businesses if they do not manage to improve these numbers.

There are various reasons for people abandoning their purchases, like excess costs of shipping and taxes, complicated transaction process, registration process exhaustion, outdated pages, page loading time, look and feel of the site etc. If we can successfully extract insights using the users activity as to why users abandon the purchase and also try to understand the patterns of successful purchases, it will lead to a massive improvement in the businesses. Other use of knowing the purchasing intention of a user can be in giving targeted ads and offers based on their activity.

Dataset Description

Dataset is obtained from UCI machine learning repository. The dataset consists of 10 numerical and 8 categorical attributes. The 'Revenue' attribute can be used as the class label.

Administrative, Administrative Duration, Informational, Informational Duration, Product Related and Product Related Duration represent the number of different types of pages visited by the visitor in that session and total time spent in each of these page categories. The values of these features are derived from the URL information of the pages visited by the user and updated in real time when a user takes an action, e.g. moving from one page to another.

The Bounce Rate, Exit Rate and Page Value features represent the metrics measured by Google Analytics for each page in the e-commerce site. The value of Bounce Rate feature for a web page refers to the percentage of visitors who enter the site from that page and then leave ("bounce") without triggering any other requests to the analytics server during that session.

The value of "Exit Rate" feature for a specific web page is calculated as for all pageviews to the page, the percentage that were the last in the session. The "Page Value" feature represents the average value for a web page that a user visited before completing an e-commerce transaction.

The "Special Day" feature indicates the closeness of the site visiting time to a specific special day (e.g. Mother's Day, Valentine's Day) in which the sessions are more likely to be finalized with transaction. The value of this attribute is determined by considering the dynamics of e-commerce such as the duration between the order date and delivery date. For example, for Valentine's day, this value takes a nonzero value between February 2 and February 12, zero before and after this date unless it is close to another special day, and its maximum value of 1 on February 8. It can be seen as a probability score. Since February 14 is a special day, keeping in mind the shipping duration the maximum value for special day will be around the dates 8th and 10th of February. The value indicates the certainty that the purchase made on a particular day is for the upcoming special day.

The dataset also includes operating system, browser, region, traffic type, visitor type as returning or new visitor, a Boolean value indicating whether the date of the visit is a weekend, and month of the year.

Feature	**Description**
Administrative	Number of pages visited by the user dealing with account management like user profile
Administrative Duration	Total amount of time (in seconds) spent by the visitor on account management related pages
Informational	Number of pages visited by the visitor about website, communication, address information of the shopping site, seller details etc.

Informational Duration	Total amount of time (in seconds) spent by the visitor on informational pages
Product Related	Number of pages visited by visitors about product related pages like product descriptions, reviews, images etc.
Product Related Duration	Total amount of time (in seconds) spent by the visitor on product related pages
Bounce Rate	Average bounce rate value of the pages visited by the visitor. A bounce occurs whenever a user enters the page and subsequently exits without visiting another page on the website or interacting with any of the elements on the page.
Exit Rate	Average exit rate value of the pages visited by the visitor. Exit rate is the number of people who exit the website after landing on a page to the total number of views the page received.
Page Value	Average page value of the pages visited by the visitor. Page value is the average value for a page that a user visited before completing a conversion or an eCommerce transaction.
Special Day	How close the site visiting day is, to a special day.
Month	Month of the date of the visit.
Operating System	Operating system used by the visitor.
Browser	The web browser used by the visitor.
Region	Geographic region from which the session has been started by the visitor.
Traffic Type	Traffic source by which the visitor has arrived at the website. It gives a numeric value describing how the user landed on the page (e.g. banner, SMS, direct, advertisement link etc).
Visitor Type	Visitor type is the type of the user and has 3 values "New Visitor", "Returning Visitor" and "Other".
Weekend	It is a boolean value indicating whether the date of the visit is weekend.
Revenue	This is the class label indicating whether the visit has completed a transaction thus generating revenue.

Methodology

The first and foremost task to do when dealing with any data set is to check if it is clean or not. Datasets with missing or invalid values can cause unwanted biases in predictions and visualizations. Once the data has been processed, the next attempt would be to extract rudimentary information about the data set with the help of various visualization tools.

One of the most widely used techniques when dealing with datasets related to products and customers is the extraction of correlation of data. Correlation analysis explores the association between two or more variables and makes inferences about the strength of the relationship. Association rules are used by online shopping giants to suggest products to customers that can be bought with another product that they have selected. This can help boost sales and even attract customers by discounting items that are associated with each other.

During the data understanding phase we make a better sense of the data by exploring the distribution of data. We explore aspects like how the data is divided on the class variable (how balanced the data is), how certain attributes are divided among their different values (for instance how returning users, new users and other users are distributed) etc.

After getting a better sense of the individual features we try to understand how the features affect the class variable. We performed bivariate analysis on several features and discern how changes in these features might affect the prediction or the class variable. This knowledge lets us develop a better model and increase the performance of the models.

Once we obtained the cleaned and analysed data we experimented with various machine learning models to predict if there would be a sale of a product. We tried multiple models like Naive Bayes, Logistic Regression, Random Forests, SVM and Neural networks.

Data Pre-Processing:

The dataset we choose did not need much preprocessing. First we checked for null values and removed them. Most of our features were numerical so we only had to transform 4 features out of 18 to numericals. The data set had 2 string attributes which were Month and Visitor_Type. We converted the month to its corresponding number i.e. Jan-1, Feb-2 and so on. The Visitor type had 3 values and each one was converted to numbers that represent them. The two boolean attributes weekend and revenue were transformed to its binary values.

For data splitting we used normal 70-30 stratified random sampling. By using stratification, the ratio of the two classes will be maintained in the test and train sets. We also used Synthetic Minority Oversampling technique to handle the imbalance in the data.

Synthetic Minority Oversampling technique (SMOTE)

The challenge with imbalanced data is that the model will tend to produce biased results, biased towards the majority class. One more challenge is that using conventional model evaluation measures like accuracy is not reliable.

The SMOTE algorithm is used here to solve the data imbalance issue. SMOTE algorithm creates synthetic samples from the minor class rather than creating duplicate copies. For each minority class observation, SMOTE calculates the k nearest neighbors and selects two or more similar instances (using a distance measure) and multiplies an instance one attribute at a time by a random amount within the difference to the neighboring instances. This is repeated multiple times with different data points to create data.

Modeling Techniques

For the project we tried 5 different models with different configurations. We also tried dimensionality reduction to reduce the number of attributes using Recursive Feature Elimination Technique. The details of each of these is explained below.

Naive Bayes:

Naive Bayes is a simple, yet effective and commonly-used, machine learning classifier. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting. It can also be represented using a very simple Bayesian network. Naive Bayes classifiers have been especially popular for text classification, and are a traditional solution for problems such as spam detection [1].

The goal of any probabilistic classifier is, with features x_0 through x_n and classes c_0 through c_k, to determine the probability of the features occurring in each class, and to return the most likely class. Therefore, for each class, we want to be able to calculate $P(c_i \mid x_0, \ldots, x_n)$. In order to do this, we use Bayes rule. Recall that Bayes rule is the following:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

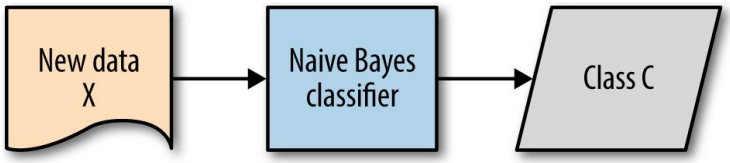
In the context of classification, you can replace A with a class, c_i, and B with our set of features, x_0 through x_n. Since P(B) serves as normalization, and we are usually unable to calculate $P(x_o, \ldots, x_n)$, we can simply ignore that term, and instead just state that $P(c_i \mid x_0, \ldots, x_n) \propto P(x_0, \ldots, x_n \mid c_i) * P(c_i)$, where $\propto$ means "is proportional to". $P(c_i)$ is simple to calculate; it is just the proportion of the data-set that falls in class i. $P(x_0, \ldots, x_n \mid c_i)$ is more difficult to compute. In order to simplify its computation, we make the assumption that x_0 through x_n are conditionally independent given c_i, which allows us to say that $PP(x_0, \ldots, x_n \mid c_i) = P(x_0 \mid c_i) *$

$P(x_1 \mid c_i) * \ldots * P(x_n \mid c_i)$. [2] This assumption is most likely not true — hence the name naive Bayes classifier, but the classifier nonetheless performs well in most situations. Therefore, our final representation of class probability is the following:

$$P(c_i|x_0,\ldots,x_n) \propto P(x_0,\ldots,x_n|c_i)P(c_i)$$
$$\propto P(c_i)\prod_{j=1}^{n} P(x_j|c_i)$$

Calculating the individual $P(x_j \mid c_i)$ terms will depend on what distribution your features follow. In the context of text classification, where features may be word counts, features may follow a multinomial distribution. In other cases, where features are continuous, they may follow a Gaussian distribution.

Note that there is very little explicit training in Naive Bayes compared to other common classification methods. The only work that must be done before prediction is finding the parameters for the features' individual probability distributions, which can typically be done quickly and deterministically. This means that Naive Bayes classifiers can perform well even with high-dimensional data points and/or a large number of data points.



Now that we have a way to estimate the probability of a given data point falling in a certain class, we need to be able to use this to produce classifications. Naive Bayes handles this in a very simple manner; simply pick the c_i that has the largest probability given the data point's features.

$$y = \underset{c_i}{argmax}\ P(c_i)\prod_{j=1}^{n} P(x_j|c_i)$$

This is referred to as the Maximum A Posteriori decision rule. This is because, referring back to our formulation of Bayes rule, we only use the P(B|A) and P(A) terms, which are the likelihood and prior terms, respectively. If we only used P(B|A), the likelihood, we would be using a Maximum Likelihood decision rule. In this project, we have used a Guassian Naive Bayes model which considers the mean and the standard deviation of the training data along with the class probabilities which are calculated using the frequency to summarize the distribution.

Logistic Regression:

Logistic Regression a classification algorithm, that is used where the response variable is categorical. The idea of Logistic Regression is to find a relationship between features and

probability of a particular outcome. If the number of classes (categories) is two, then we refer to it as Binomial Logistic Regression. Where as if there are more than two classes, we call it Multinomial Logistic Regression. Let's consider the case of Binomial Logistic Regression [3]. With binary classification, let ***'x'*** be some feature and ***'y'*** be the output which can be either a 0 or 1. The probability that the output is 1 given its input can be represented as:

$$P(y = 1 \mid x)$$

Logistic regression can be expressed as:

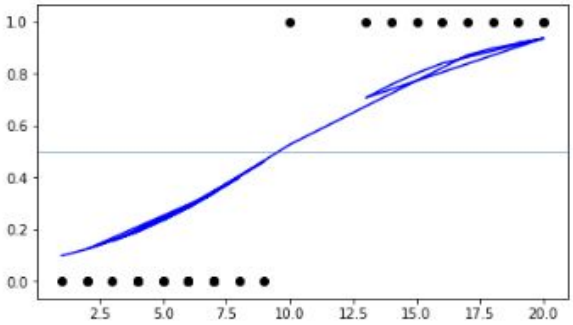
$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X.$$

where, the left hand side is called the logit or log-odds function, and p(x)/(1-p(x)) is called odds.

The odds signifies the ratio of probability of success to the probability of failure. Therefore, in Logistic Regression, linear combination of inputs are mapped to the log(odds) - the output being equal to 1. If we take the inverse of the above function, we get:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}.$$

This is known as the Sigmoid function and it gives an S-shaped curve. It always gives a value of probability ranging from 0<p<1.



Unlike linear regression model, that uses Ordinary Least Square for parameter estimation, we use *Maximum Likelihood Estimation*.

There can be infinite sets of regression coefficients. The maximum likelihood estimate is that set of regression coefficients for which the probability of getting the data we have observed is maximum. If we have binary data, the probability of each outcome is simply π if it was a success, and 1−π otherwise. Therefore we have the likelihood function:

$$\mathcal{L}(\beta; \boldsymbol{y}) = \prod_{i=1}^{N} \left(\frac{\pi_i}{1-\pi_i}\right)^{y_i} (1-\pi_i)$$

To determine the value of parameters, log likelihood function is taken, since it does not change the properties of the function. The log-likelihood is *differentiated* and using **iterative** techniques like Newton method, values of parameters that maximise the log-likelihood are determined.

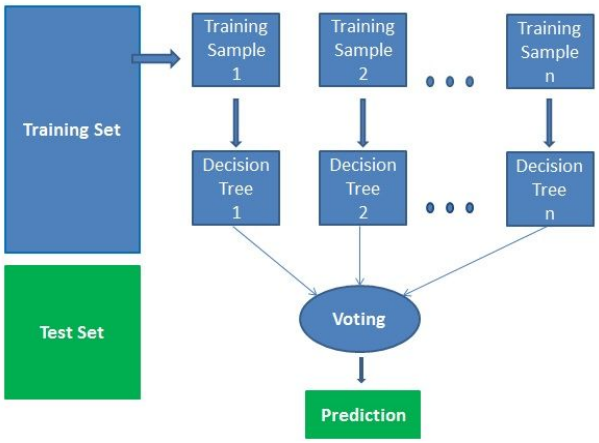
We have used a 70-30 data split to train the logistic regressor. We use Recursive Feature Elimination (RFE) which is a feature selection method that fits a model and removes the weakest feature (or features) until the specified number of features is reached. Features are ranked by the model's coef or feature importances attributes, and by recursively eliminating a small number of features per loop, RFE attempts to eliminate dependencies and collinearity that may exist in the model. We have kept the top 10 features out of the 18 features present in the original data set. The top 10 features that are selected by RFE are Informational, BounceRates, ExitRates, PageValues, SpecialDay, Month, OperatingSystems, Browser, VisitorType and Weekend.

Random Forests:

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.[4]

Random forests has a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset.

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object.



Algorithm:

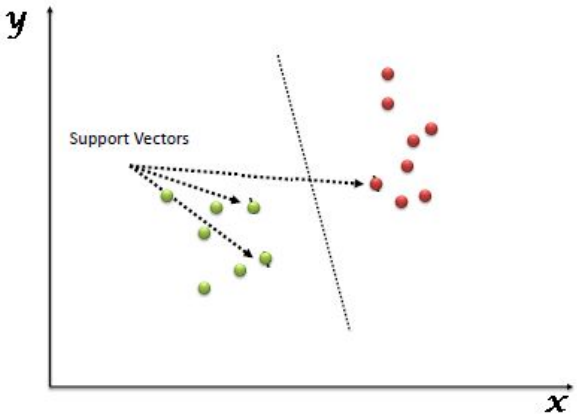
- Select random samples from a given dataset.
- Construct a decision tree for each sample and get a prediction result from each decision tree.
- Perform a vote for each predicted result.
- Select the prediction result with the most votes as the final prediction.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

We have followed the same set of specification like the logistic regression model for the Random Forest Classifier. With the 70-30 data split and RFE method to eliminate the less important feature the Random forests give a better performance. This is because random forest is an ensemble model and uses multiple decision tree outputs to give the final output. It uses a bagging method to accumulate and give the final output of the model. Since the individual decision trees can grow deep with more features which may cause a variance in the model we use the top 8 features. The top 8 features were Administrative, Administrative_Duration, ProductRelated, ProductRelated_Duration, BounceRates, ExitRates, PageValues and Month.

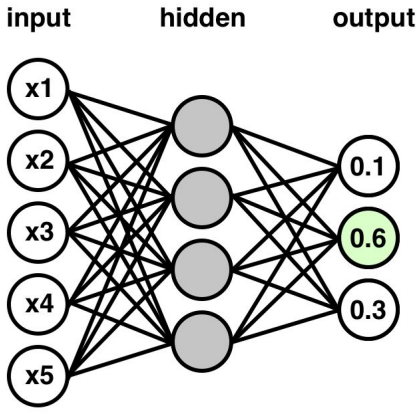
Support Vector Machines:

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well [6].



Support Vectors are simply the coordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line). In our model we have used an SVM with a Radial Basis Function kernel (RBF). This kernel can choose a non-linear decision boundary which is used to fit a non linear data. Since this performed better than a linear kernel, we can say that the data is non linear. We also used a scaling factor to the data using the gamma value. This scaling uses 1 / (n_features * X.var()) as value of gamma. We also used a penalty of 3 to the error term.

Neural networks:



Neural networks 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. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated [7].

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on. (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning, classification and regression.)

Here we have a simple neural network containing three layers. The first layer is the input layer with 17 neurons. The input to each neuron is the input feature. Each neuron then connects (i.e. acts as an input) to each of the 4 neurons in the hidden layer (in practice this means the outputs of all the neurons in the first layer are joined together and then passed as input to every neuron in

the next layer), a hidden layer just means a layer that is neither input nor output. These types of layer with all the neurons connected to each other is called a fully connected or dense layer. This hidden layer is then also fully connected to another hidden layer which is fully connected to an output layer (soft max). This final layer does the classification based on the probability values. For example, in the above diagram the class depicted by the middle node will be selected as it has the highest probability of 0.6.

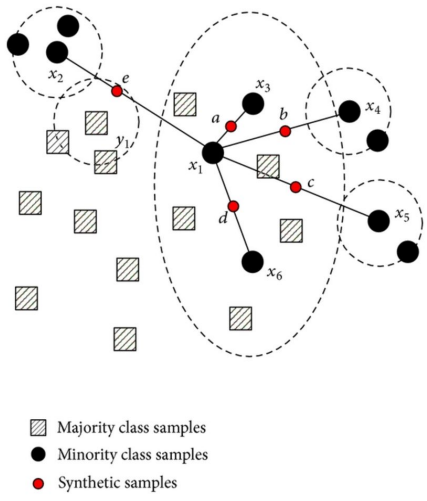
We have used RELU activation function with 60 dense nodes in our hidden layer. We use the ADAM optimization and the network is trained for 80 epochs trying to minimize a binary_crossentropy loss function.

Recursive Feature Elimination:

Recursive Feature Elimination (RFE) is a technique to reduce the dimensionality of the data. As the name suggests this algorithm recursively removes features, builds a model using the remaining attributes and calculates model accuracy. RFE is able to work out the combination of attributes that contribute to the prediction on the target variable (or class). It uses accuracy metric to rank the feature according to their importance. We have used this technique in two of our models. The Random Forest classifier and the Logistic Regression model use the RFE technique and reduces the dimensionality of the data.

Synthetic Minority Over-sampling Technique (SMOTE):

An approach to the construction of classifiers from imbalanced datasets is described. A dataset is imbalanced if the classification categories are not approximately equally represented. Often real-world datasets are predominantly composed of “normal” examples with only a small percentage of ``abnormal" or ``interesting" examples. It is also the case that the cost of misclassifying an abnormal (interesting) example as a normal example is often much higher than the cost of the reverse error.



What smote does is simple. First it finds the n-nearest neighbors in the minority class for each of the samples in the class . Then it draws a line between the neighbors and generates random points on the lines.In the given image we can observe that it finds the 5 nearest neighbors to the sample points, then draws a line to each of them. And finally it creates samples on the lines where class = minority class.

Summary of Code

We used python for coding and the pandas library to operate on the dataset (.csv). We utilized several other libraries for various operations that we needed to do throughout the duration of this project. The first thing we did after importing the data was to check if there were any corrupt or missing values. Fortunately, there were no null values and we were able to proceed with further analysis. In order to simplify our task, we converted some of the attributes (categories/booleans) into relevant numbers. Once, we completed the pre-processing step we proceeded with exploration of the data.

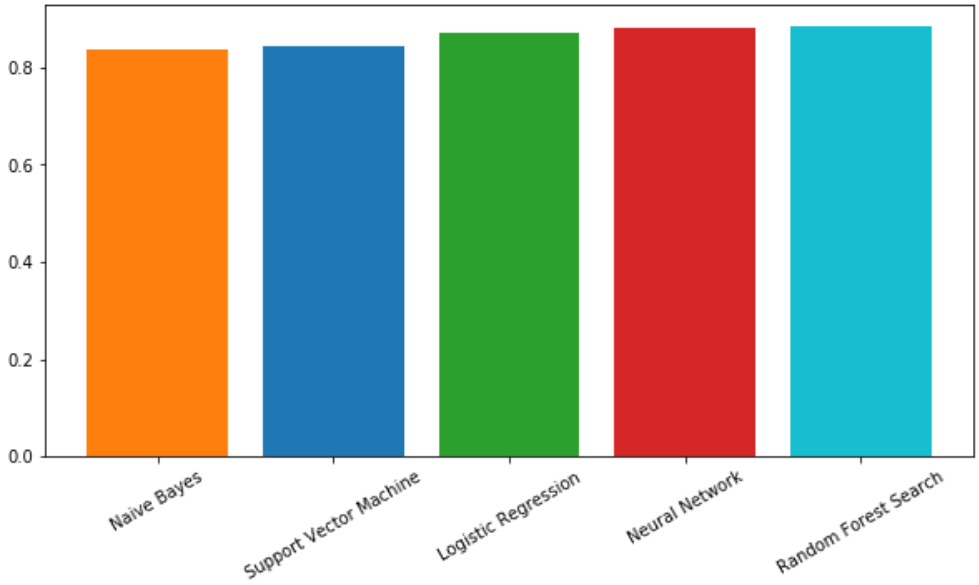
With the aid of libraries like matplotlib, plotly, etc. we created various plots to extract useful information from the dataset which were otherwise not explicitly visible. By studying these plots we were able to identify correlations between features and have included our corresponding interpretations in this report.

Next, we wanted to use the data to build a model for predicting whether an item will get sold or not. In the interest of a more comprehensive analysis, we built five models utilizing different modelling techniques: Logistic Regression, Naive Bayes, Random Forest Classifier, Support Vector Machine, Neural Network. After splitting our data for training and testing we trained and evaluated each model by calculating various metrics like accuracy, preceions, recall, f1 score, etc. After building and optimizing the models, we compared each model to find out the best performing one. To do this we plotted the ROC curves for each model and weighed them against each other based on the total covered area under the curve.

Results

Model Performance Comparison:

As stated earlier, we tried 5 different models for our prediction task. The below figure shows the accuracy comparison of the different models:

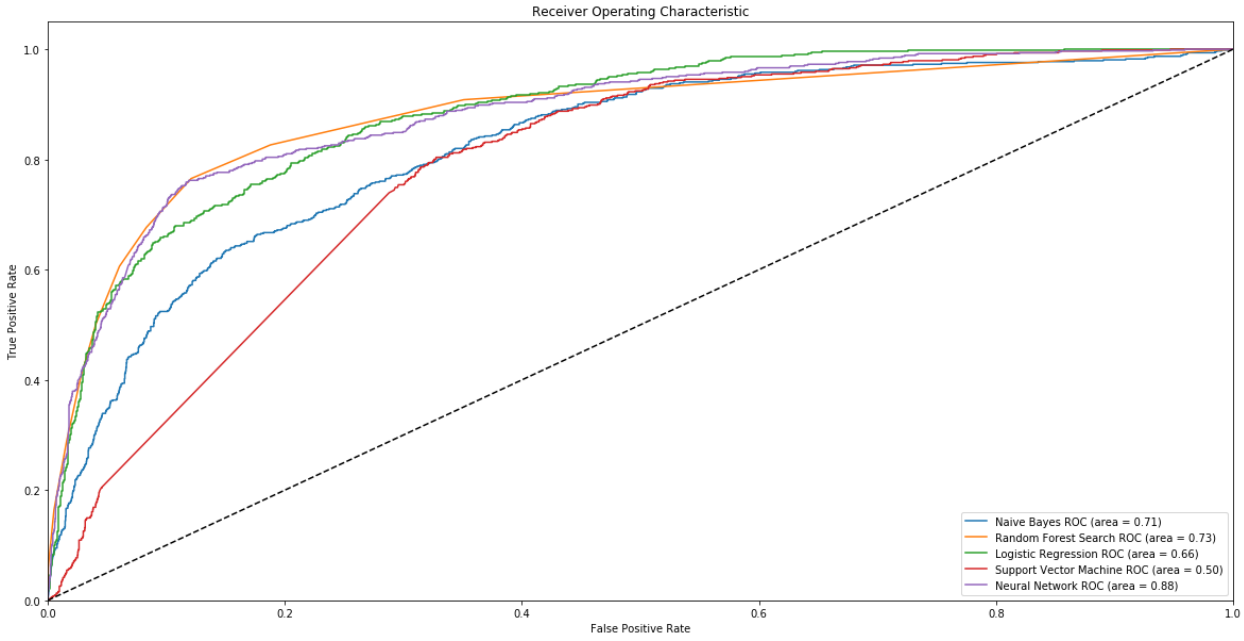


The most basic model we implemented was the Naive Bayes model. This model performed the poorest among the other models that was implemented which seems normal. The Gaussian Naive Bayes model gave us an accuracy around the low 80s. This was surprisingly followed by the Support Vector Machine. The SVM model performed worse than the Logistic Regression model. The RBF kernel SVM model was predicted to perform better but it did not do a very good job in classifying the positive classes which brought down its performance. Its accuracy was close to the Naive Bayes model in the low 80s.

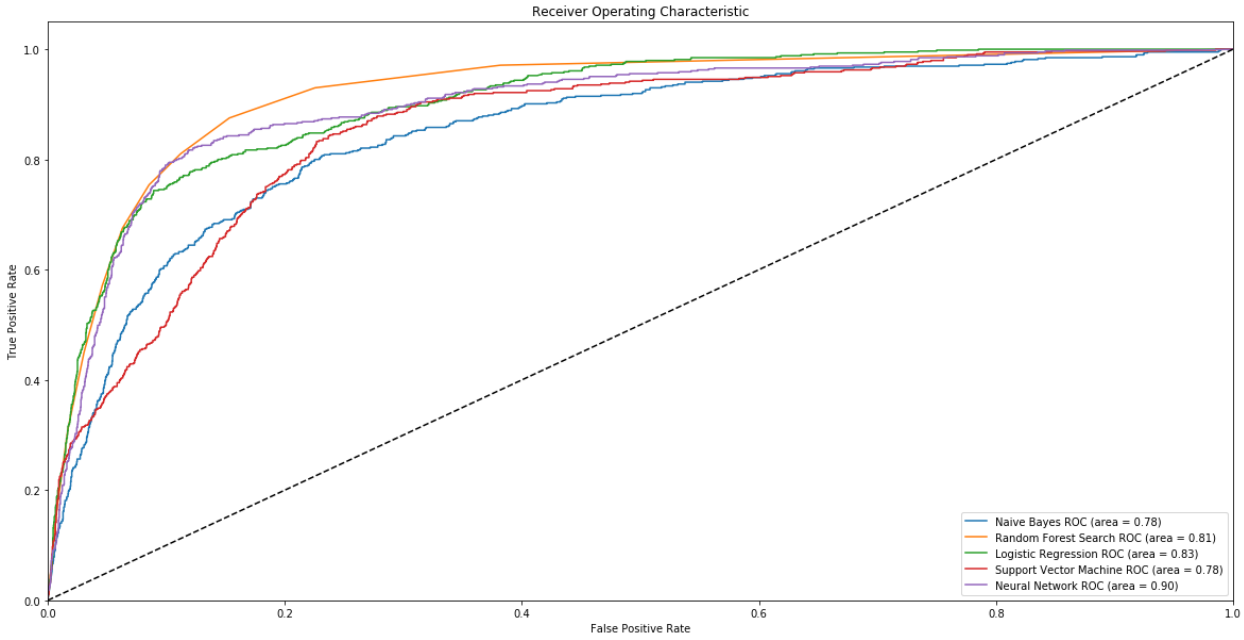
The Logistic regression did a pretty decent job fitting the data and gave us an accuracy around the mid 80 mark. This model fit the data better than an SVM model. The two best models were the neural network and the random forest. Neural network being a more complex model compared to logistic regression fit the data better. The Random Forest classifier which is an ensemble model using bagging as the ensemble method and decision tree as the individual model performed the best among the other models. The neural network and the random forest models gave very similar accuracies of around the higher 80 close to 90%.

To understand the models better, we plotted the ROC (receiver operating characteristic) curves. ROC curves are a plot of test sensitivity as the y coordinate versus its 1-specificity or false positive rate (FPR) as the x coordinate and is an effective method of evaluating the quality or performance of diagnostic tests. It tells how much model is capable of distinguishing between

classes. We calculate the area under the curve (AUC) to get a better sense of the performance. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s.



Before Optimization



After Optimization

One difference we can observe here is that the AUC of naive bayes is greater than logistic regression, even though the accuracy is the opposite. This shows that the Naive Bayes does a better job at predicting the classes right more, which means it has a better positive and negative

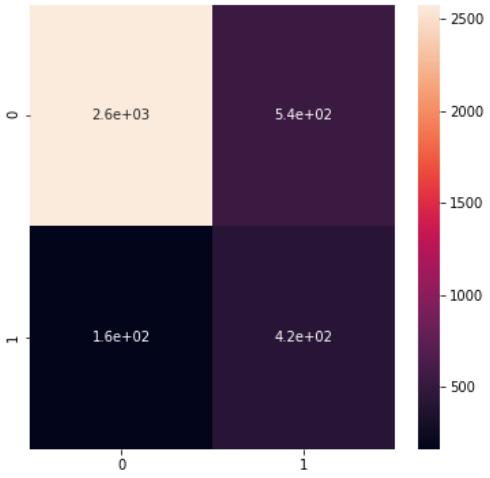
rate. The SVM performs the poorest among the models again while surprisingly the neural networks has a greater AUC compared to the random forest classifier. Even though the Neural networks have a slightly lesser accuracy it has a much better AUC values. To get a better idea of the specificity and sensitivity we can observe the confusion matrix of each classifier.

The formula for TPR and FPR is given below. So from the ROC curve and these formula we can say that more the AUC, the model has a good separability, which means it can predict TP and TN better.

$$\text{TPR /Recall / Sensitivity} = \frac{TP}{TP + FN} \qquad \text{Specificity} = \frac{TN}{TN + FP} \qquad \text{FPR = 1 - Specificity} = \frac{FP}{TN + FP}$$

The detailed results for each of the model is given below

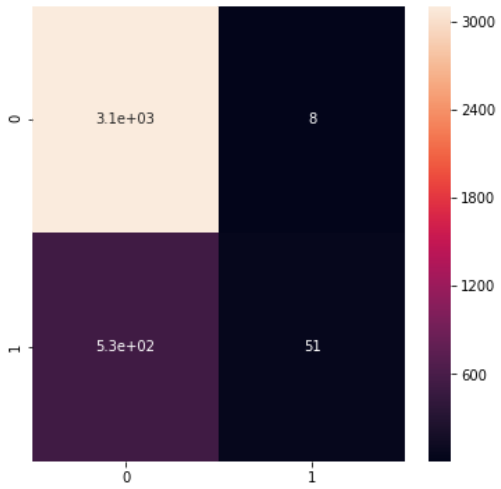
1. **Naive Bayes**



	Precision	**Recall**	**f1-score**	**support**
0	0.94	0.83	0.88	3114
1	0.44	0.72	0.55	585
accuracy			0.81	3699
macro avg	0.69	0.78	0.71	3699
weighted avg	0.86	0.81	0.83	3699

The Gaussian Naive Bayes Model we used gave expected results. Since it is a simple model it gave decent accuracy of 81%. Using the SMOTE algorithm gave the decent Recall and f1-score for the minority class as well.

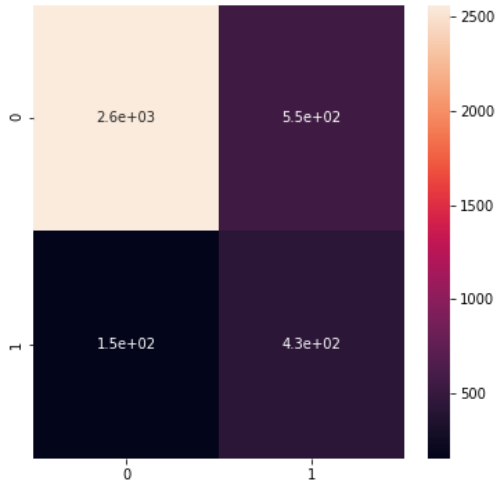
2. **Support Vector Machine (SVM):**



Without SMOTE	**Precision**	**Recall**	**f1-score**	**support**
0	0.85	1.00	0.92	3114
1	0.86	0.09	0.16	585
accuracy			0.85	3699
macro avg	0.86	0.54	0.54	3699
weighted avg	0.86	0.85	0.80	3699

This classification report and the confusion matrix shows the performance of SVM before using the SMOTE algorithm to oversample the minority class. Due the imbalance we can see the extreme recall values for the two classes. A 100% recall for the majority class and 9 % for minority class. Also the confusion matrix shows the very low values for the true positive and false positive rates, which correspond to the minority Revenue Generated class.

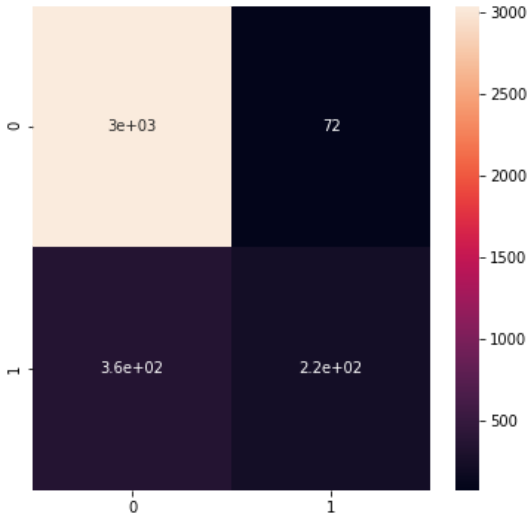
For the same RBF kernel SVM we then used the data that we oversampled using SMOTE and there was considerable improvement in the performance. The Recall values went up from 9% to 74% and for majority class it reduced to 82%. We can also observe the improved confusion matrix distribution. The overall accuracy of the model was around 81%.



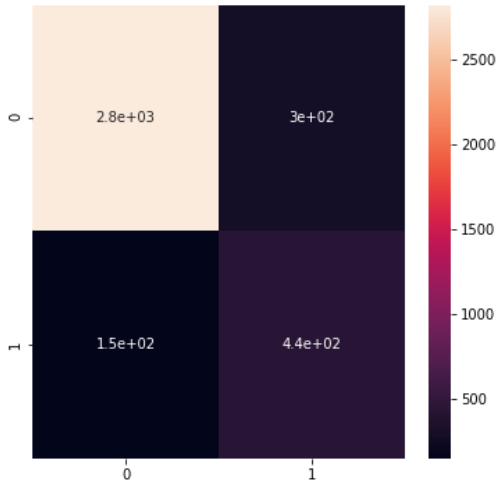
With SMOTE	Precision	Recall	f1-score	support
0	0.94	0.82	0.88	3114
1	0.44	0.74	0.55	585
accuracy			0.81	3699
macro avg	0.69	0.78	0.71	3699
weighted avg	0.86	0.81	0.83	3699

3. Logistic Regression:

For Logistic Regression we trained our model using data that was randomly sampled and obtained subpar results for Recall, especially for Class 1 (users that generated revenue by purchasing product). We then decided to apply SMOTE to over sample the imbalanced data for training, similar to what we did for the previous models. However, we decided to go a step further and applied Recursive Feature Elimination to reduce our dimension before applying SMOTE. This resulted in a substantial improvement in the Recall score and overall performance of the model.



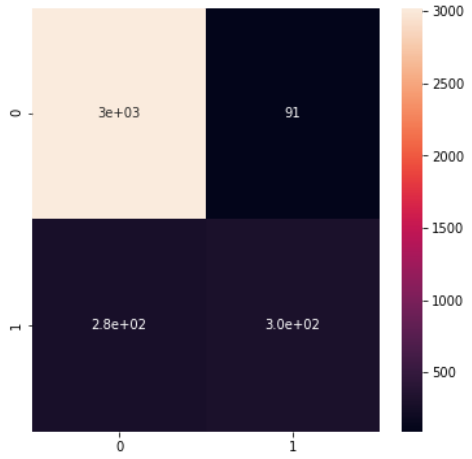
Without RFE, SMOTE	**Precision**	**Recall**	**f1-score**	**support**
0	0.89	0.98	0.93	3114
1	0.76	0.38	0.51	585
accuracy			0.88	3699
macro avg	0.82	0.68	0.72	3699
weighted avg	0.87	0.88	0.87	3699

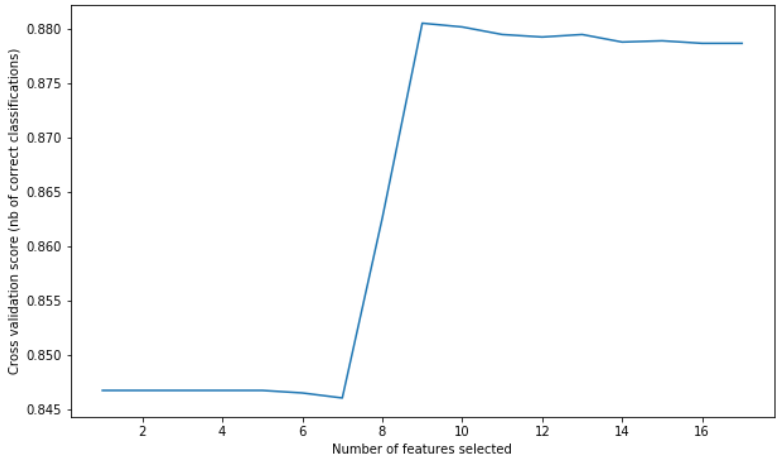


With RFE, SMOTE	Precision	Recall	f1-score	Support
0	0.95	0.91	0.93	3114
1	0.60	0.75	0.66	585
Accuracy			0.88	3699
Macro avg	0.77	0.83	0.79	3699
Weighted avg	0.89	0.88	0.89	3699

4. Random Forest Classifiers

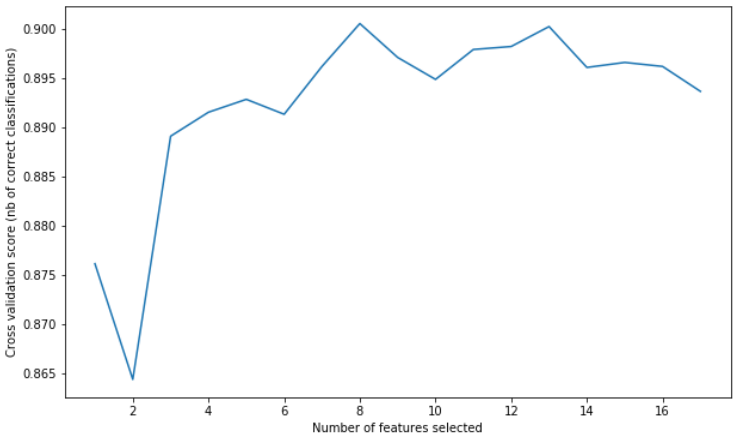
We followed a similar approach for our Random Forest Classifier based model. We used RFE to eliminate features and then executed SMOTE and obtained the following results.

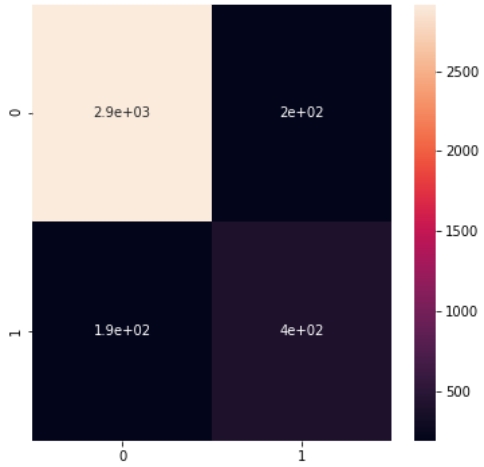




Without RFE, SMOTE	Precision	Recall	f1-score	Support
0	0.92	0.97	0.94	3114
1	0.77	0.52	0.62	585
Accuracy			0.90	3699

Macro avg	0.84	0.75	0.78	3699
Weighted avg	0.89	0.90	0.89	3699

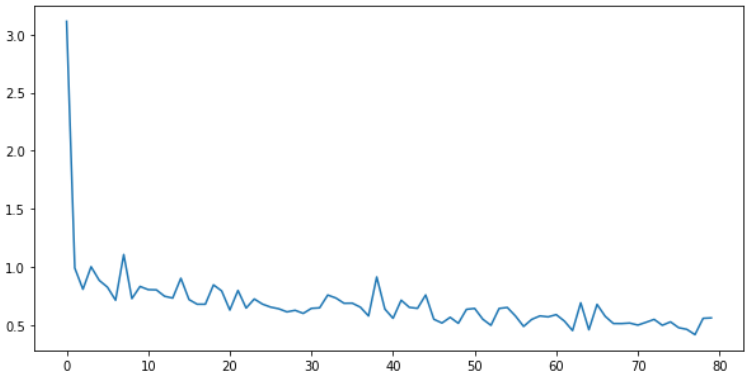


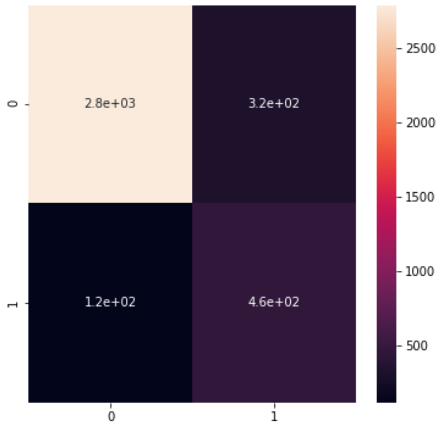


With RFE, SMOTE	**Precision**	**Recall**	**f1-score**	**Support**
0	0.94	0.94	0.94	3114
1	0.67	0.68	0.67	585
Accuracy			0.89	3699
Macro avg	0.80	0.81	0.80	3699
Weighted avg	0.90	0.90	0.90	3699

5. Neural Network

The training of neural network is an iterative process. We trained our model for 80 epochs and the below graph show the loss of function through the epochs. From the below graph we can say that the model has learnt well without overfitting or underfitting the data.

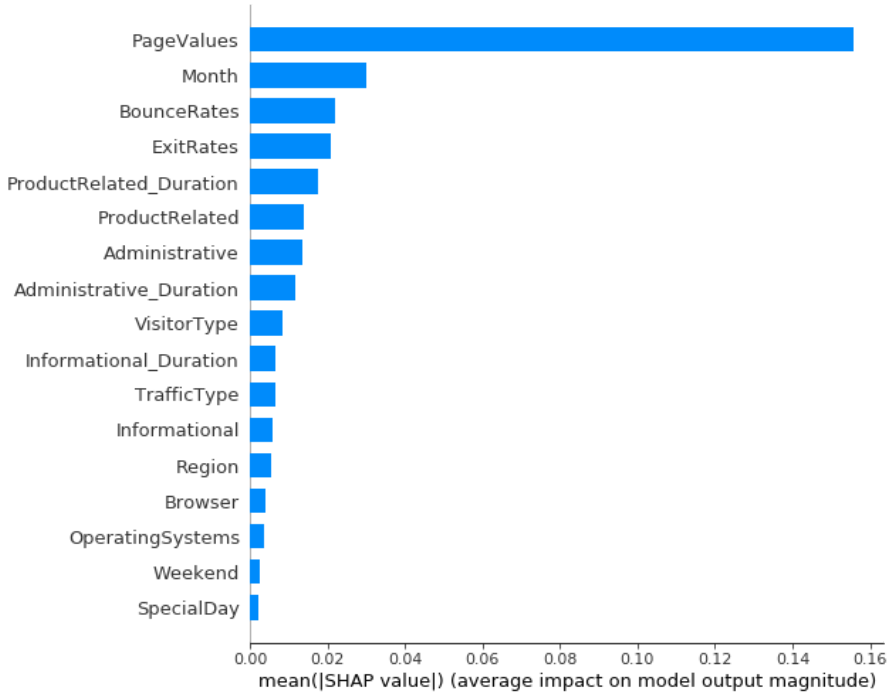


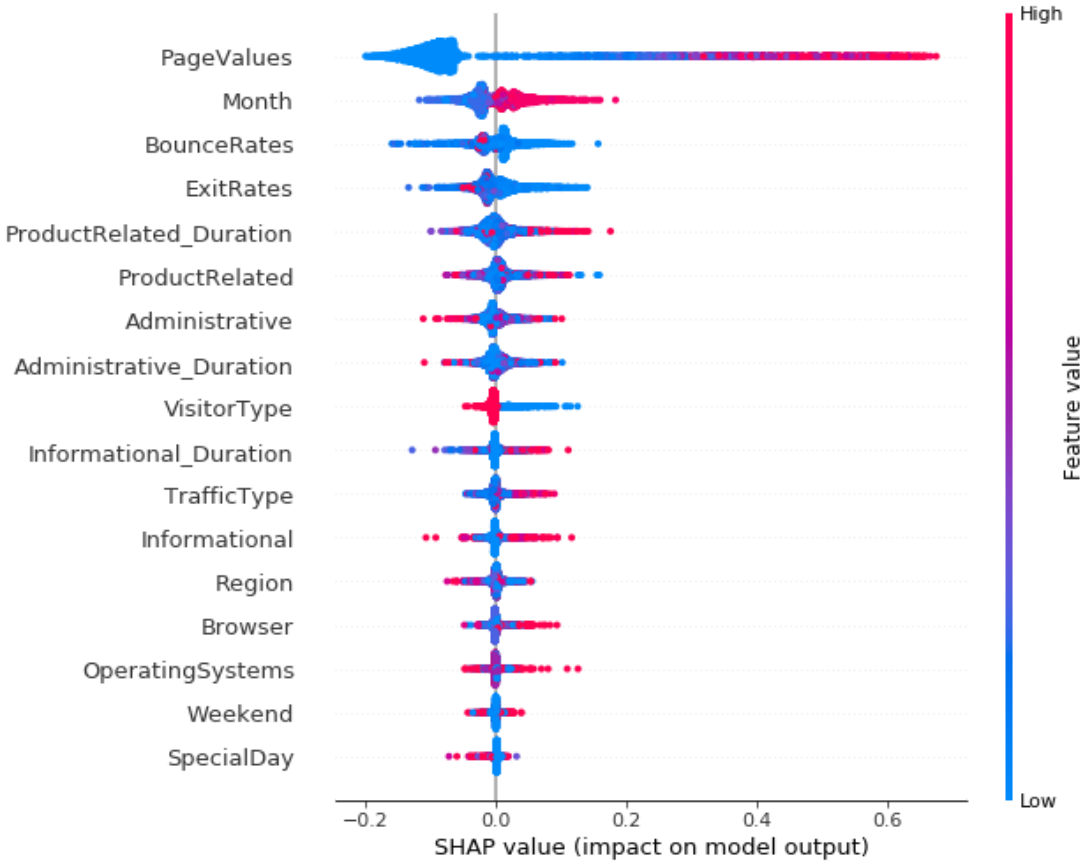


With SMOTE	Precision	Recall	f1-score	Support
0	0.96	0.90	0.93	3114
1	0.59	0.79	0.68	585
Accuracy			0.88	3699
Macro avg	0.77	0.85	0.80	3699
Weighted avg	0.90	0.88	0.89	3699

SHAP analysis:

SHAP Values (SHapley Additive exPlanations) break down a prediction to show the impact of each feature on a model. We derived these reading from the Random Forest classifier model.SHAP values interpret the impact of having a certain value for a given feature in comparison to the prediction we'd make if that feature took some baseline value. In our dataset we can see the PageValues feature affects the model the greatest and the special day feature affects it the least. The special day feature in our dataset is sparse and that may be one of the reasons for getting a low impact value for that feature. The other features lie in the mediocre range of impact and the picture gives an overall feature impact on the model output magnitude.

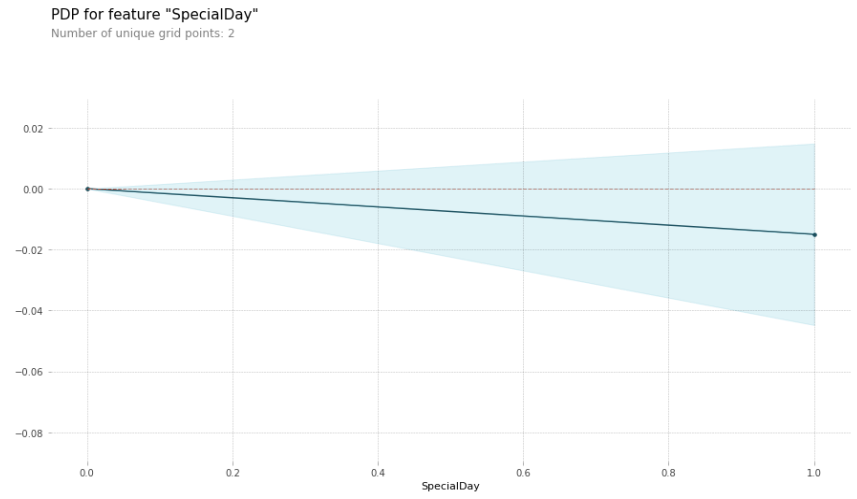

PageValues
Month
BounceRates
ExitRates
ProductRelated_Duration
ProductRelated
Administrative
Administrative_Duration
VisitorType
Informational_Duration
TrafficType
Informational
Region
Browser
OperatingSystems
Weekend
SpecialDay
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
mean(|SHAP value|) (average impact on model output magnitude)

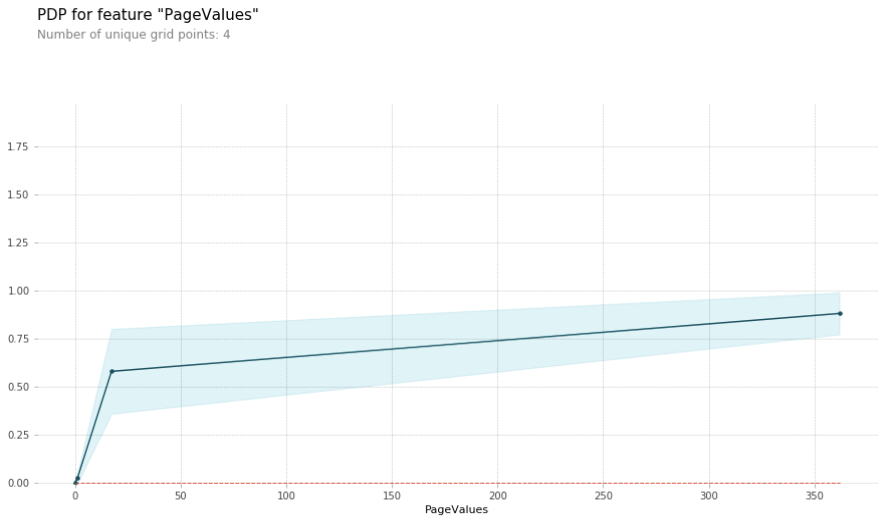

PageValues
Month
BounceRates
ExitRates
ProductRelated_Duration
ProductRelated
Administrative
Administrative_Duration
VisitorType
Informational_Duration
TrafficType
Informational
Region
Browser
OperatingSystems
Weekend
SpecialDay
−0.2
0.0
0.2
0.4
0.6
SHAP value (impact on model output)
High
Feature value
Low

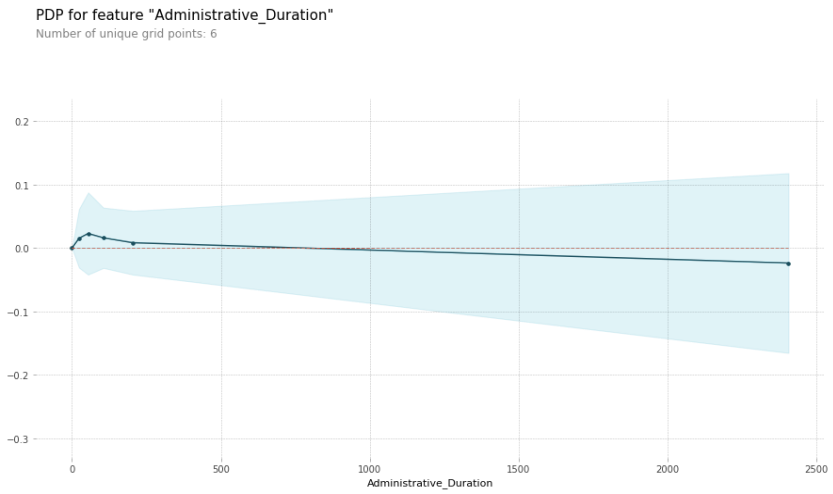
Partial Dependence Plot (PDP):

To see to what extent these features values affect the model output we used the Partial Dependence Plot (PDP). The partial dependence plot shows the marginal effect one or two features have on the predicted outcome of a machine learning model. A partial dependence plot can show whether the relationship between the target and a feature is linear, monotonic or more complex.

The PDP for page value shows that the model output almost changes exponentially with change in Page Values feature. The special day feature linearly decreases the model output and the administrative duration the prediction of the model increases with the value initially and then gradually decreases.

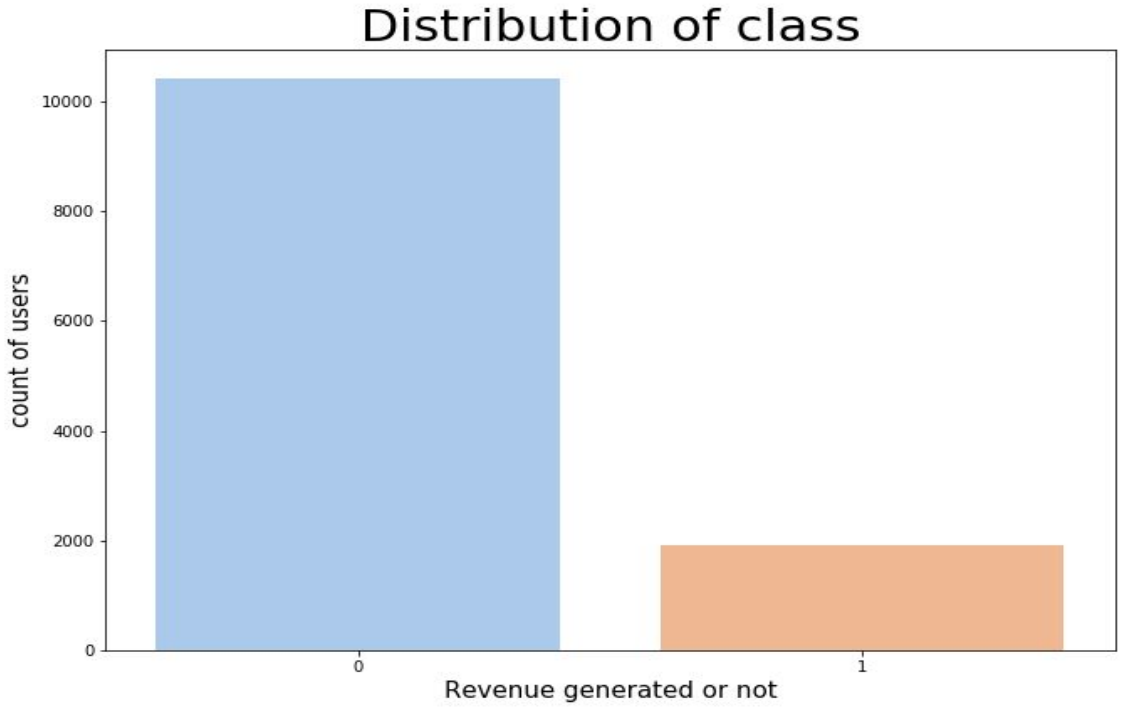


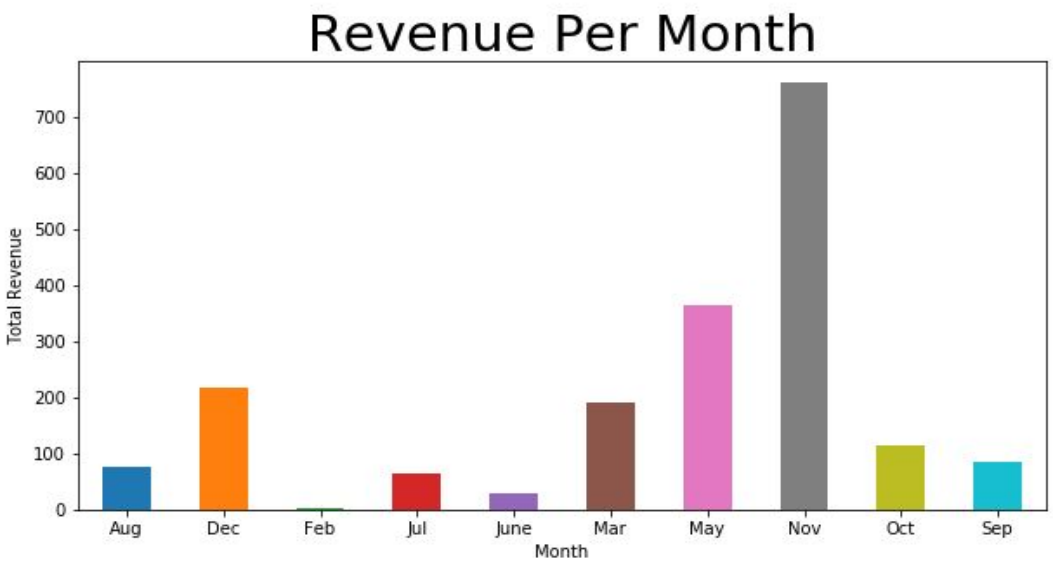


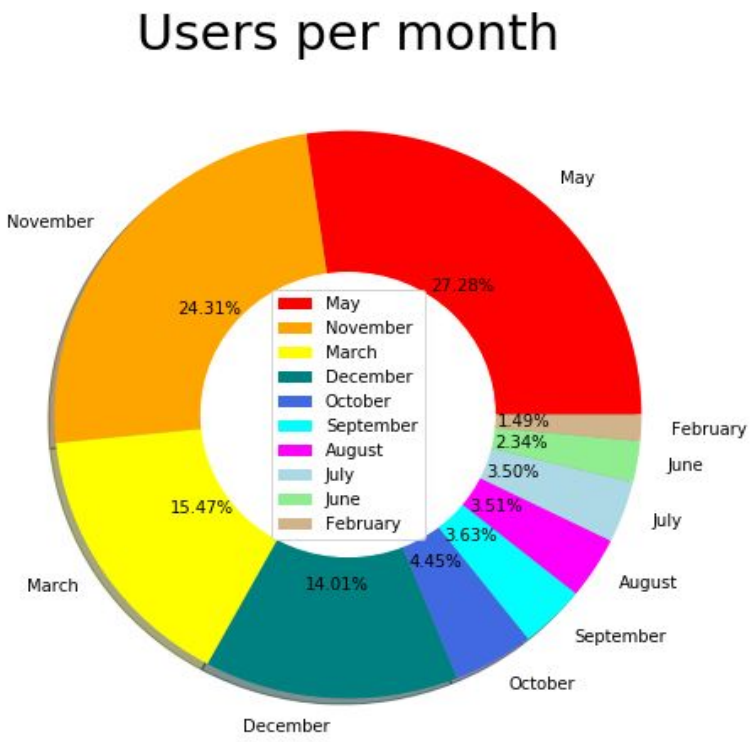


Discussion

Let's understand the distribution of the classes in our dataset. We have more than 10000 users who didn't contribute to the revenue and around 2000 users giving us revenue. So we have slightly unbalanced dataset, but we can make use of this to increase the revenue as much as possible with the available data. We need to understand how are we generating the existing revenue to know how to increase it.





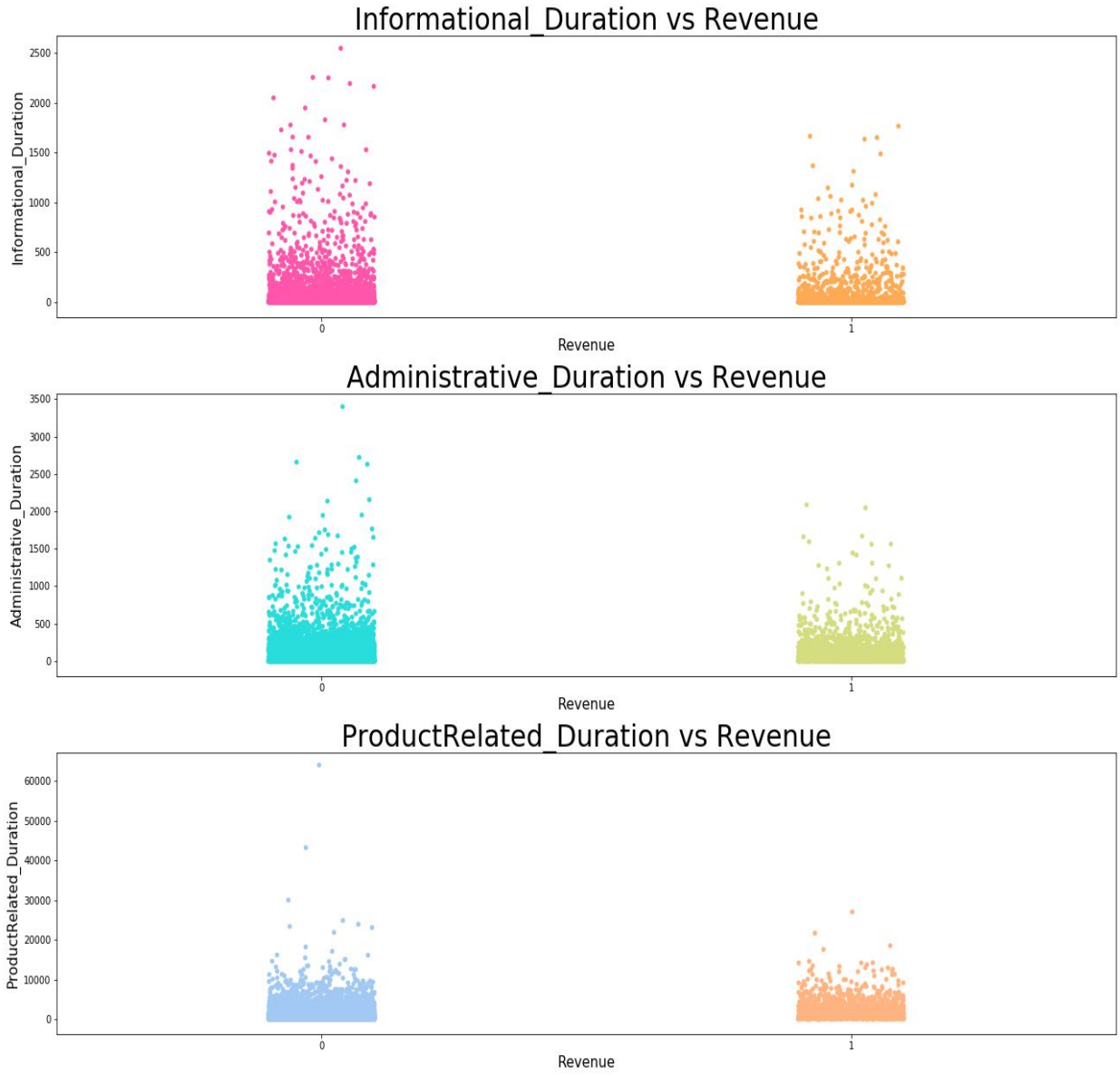


This plot shows that store gained most revenue in the month of november and least in the month of february. We can also see that the percentage of users visited the website was very high in the month of November, and a similar trend can be observed for other months. We can conclude that

when we have users on the website we gain some revenue, so it is very evident that the website addresses the needs of the users.

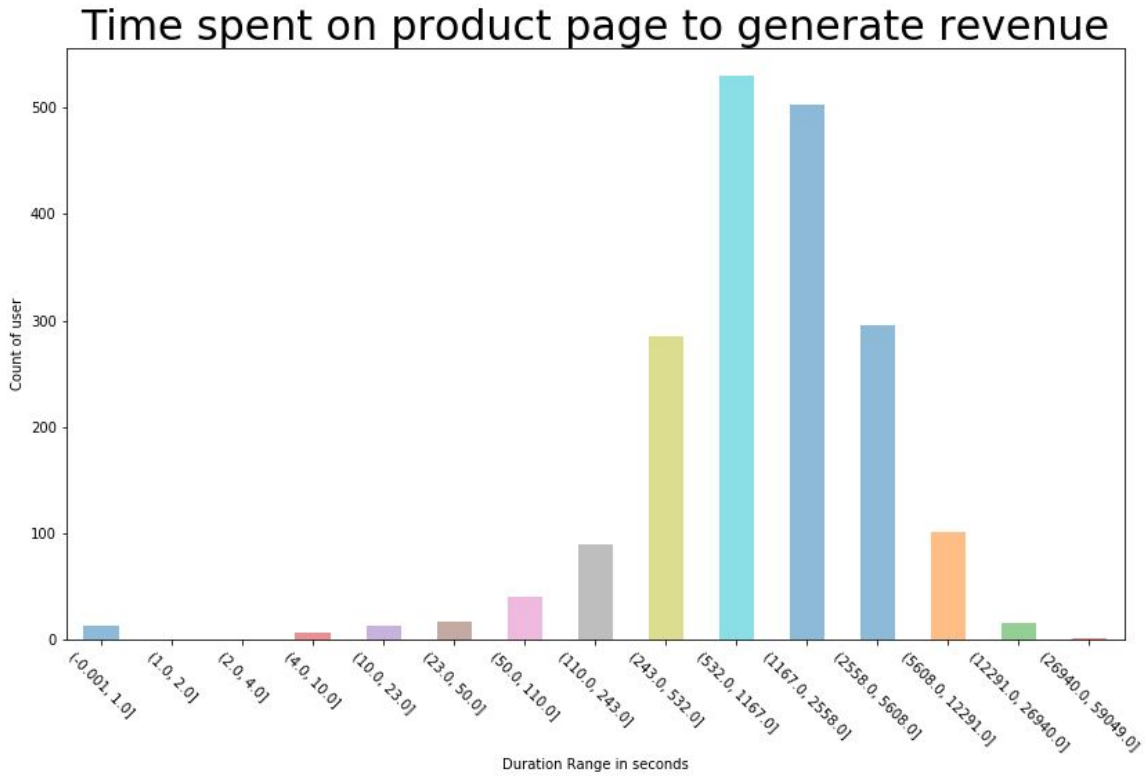
Now we need to increase the revenue of the website by turing all the users as our customers. To achieve this we need to understand how each user on the website uses it, which are the page the visit in a particular session, and which pages contribute into the revenue.

We have observed that if a user spend too much time on different pages of the website there is a high probability that we may lose that potential customer. This pattern can be studied from the bivariate analysis of the duration spent on each type of page and revenue generated or not.



In the above plot we compared how both classes over the time duration spent on the info page. This helps us understand that when the user spends more time on the page more we lose the customer, most of the transaction done by the customers take upto 200 seconds.

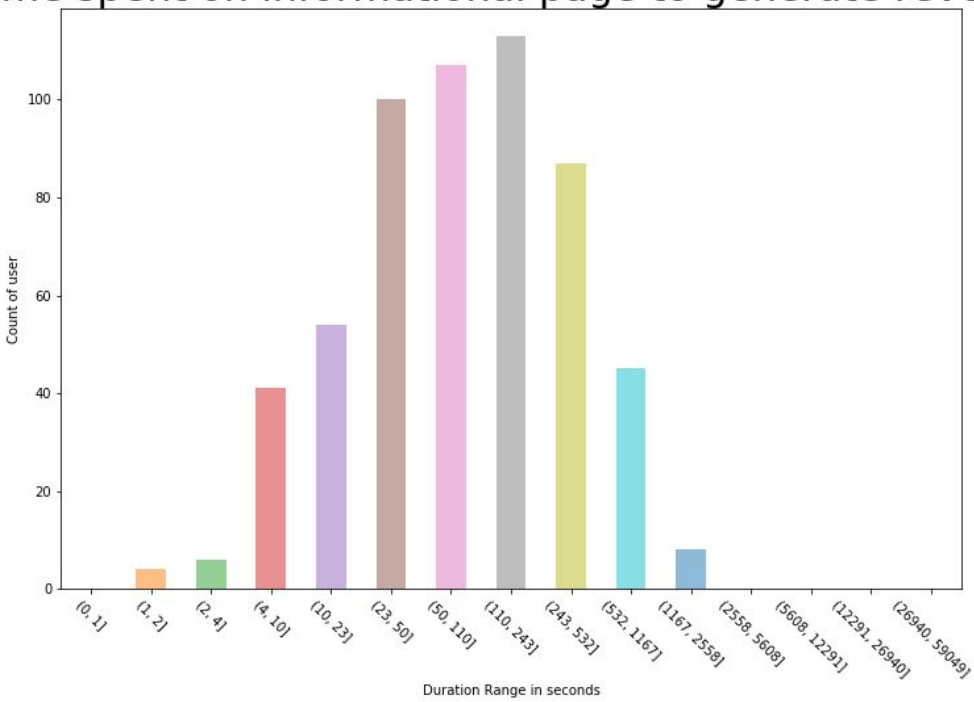
We will now understand how much time a customer spends on each product before buying it. This will give us an insight of whether we have any scope of improvement to the website itself.



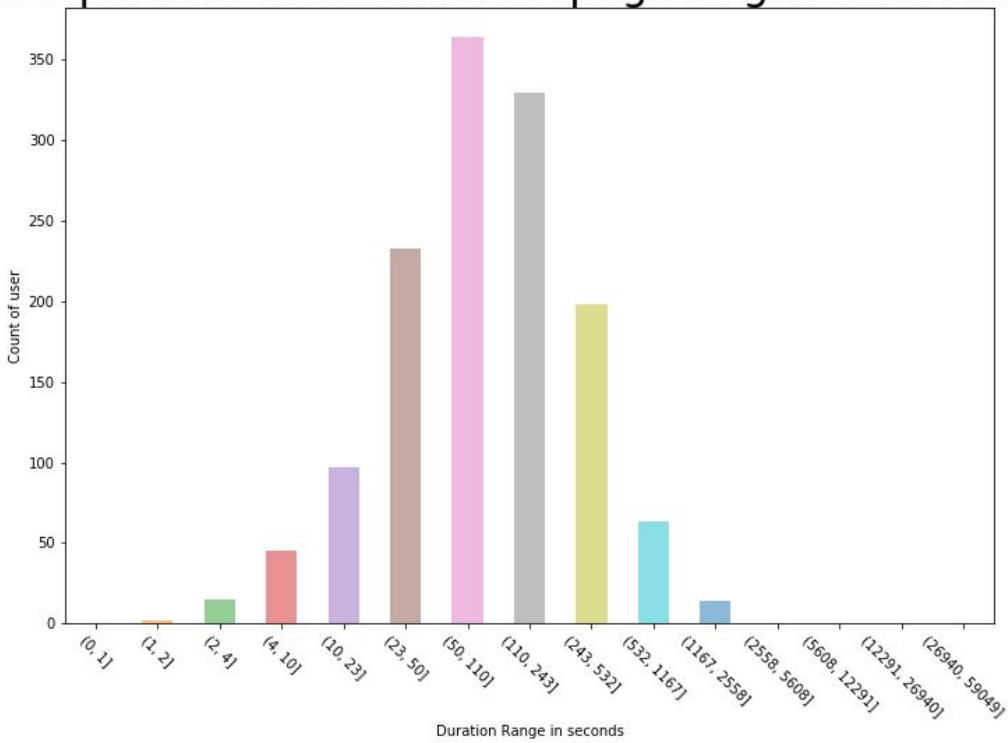
In the above plot we see that there are different bins of time spent by customer to buy a product. It suggests that typically user has to spend approx 500 to 2500 seconds in order to finalize the product and buy it. I think this is not an ideal time duration to be spent to buy the product. If have to spend too much time in finalizing the product then we might lose a potential customer. We need to improve on the overall search engine of the website and cater them with the right product when the try to find one. We have around 300 customers who have spent around 250-550 seconds to buy a product and this is an ideal time period.

I think our website design is good enough to help the customers understand the product on the product page itself. We can see from the below plot that customers have to actually visit product info page for a very less time duration. The only problem I see is that user has to visit the info page to be sure of the product they are going to buy. We have a scope for the development but can be taken on a lower priority.

Time spent on informational page to generate revenue

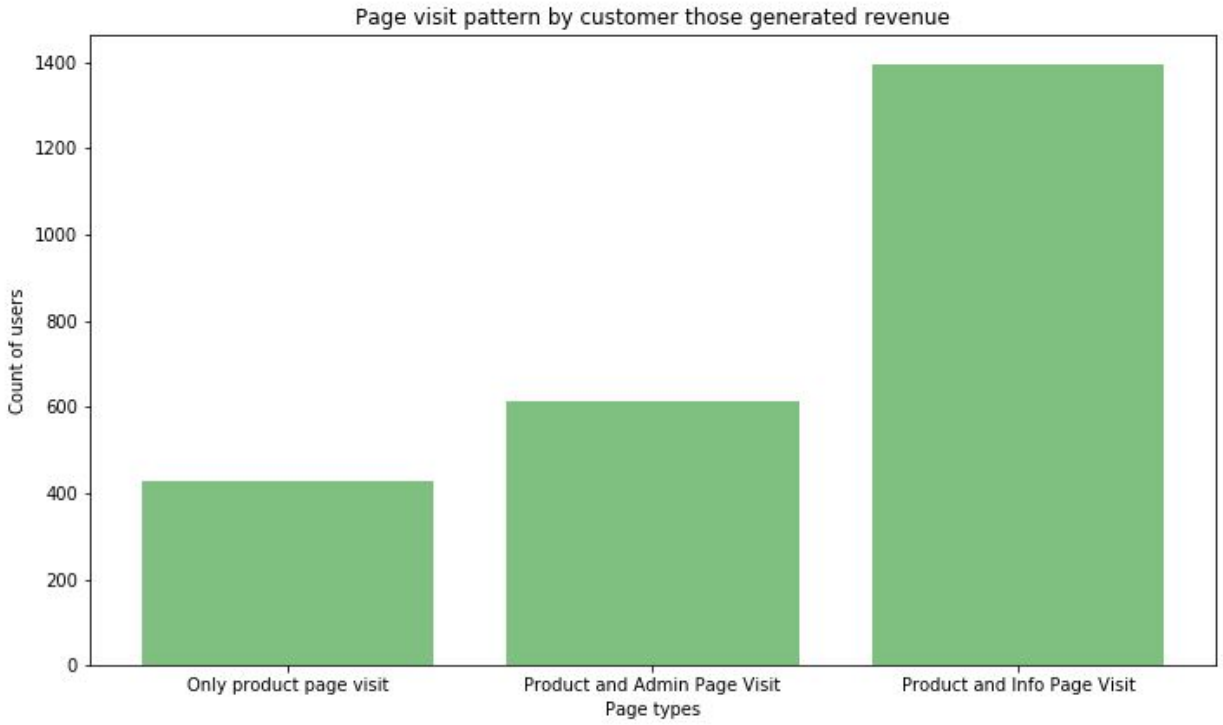


Time spent on administrative page to generate revenue



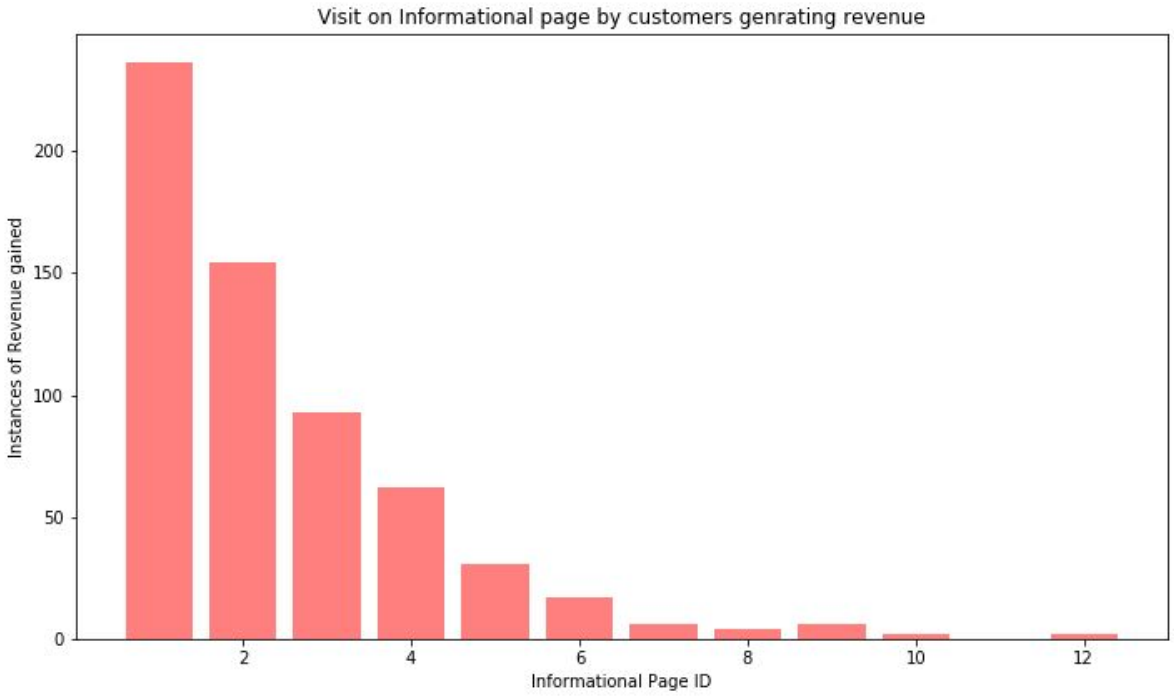
Now let's understand how much time the user spends on the administrative page. We have around 2000 unique users who have given us the revenue, and we can see that more than 70% of the users have to visit the administrative page in order to buy the product, also around 50% customers have to spend more than a minute on the administrative pages. We seriously need to redesign this, the time spent should be very less around 10-20 seconds just to confirm the payment details and shipping details.

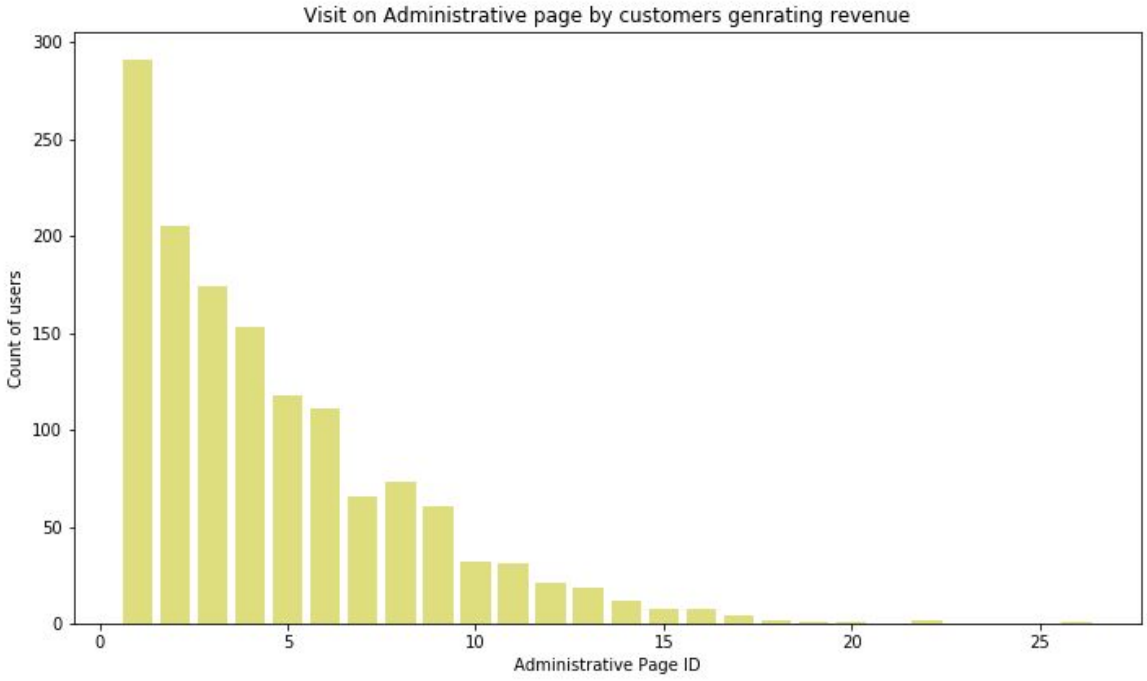
We need to understand now which pages we have to work on the most and quickly based on the data we have, we need to prioritize the tasks based on that. On analysis of how many users which aspect of the website the most will give us a more hold on prioritizing tasks.



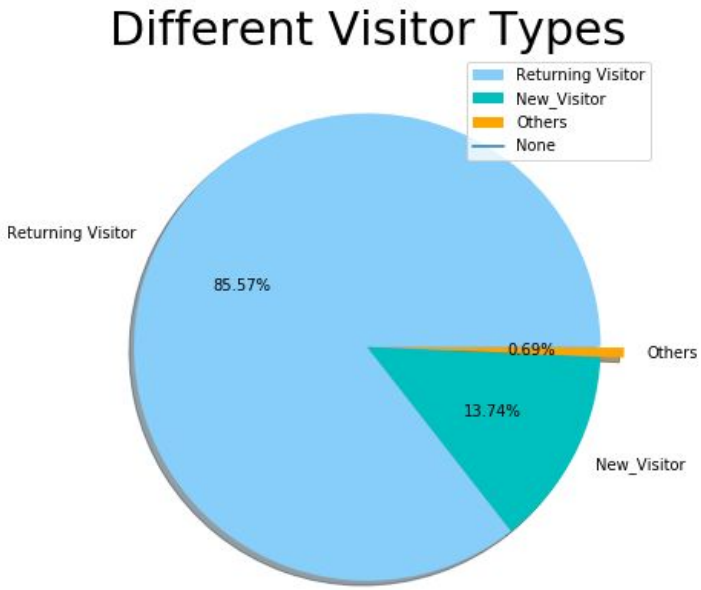
The above graph provides the analysis on how users use different pages in order to buy a product. We have around 400 users who only visited the product page and bought the item, 600 of them bought the product after visiting the admin page and 1400 bought on visiting the info page. So our top priority is to fix the informational pages, and reduce the time spent by the user on those pages, so that we retain the potential customers.

We have done some more analysis on visiting patterns of users that generated revenue for each page.

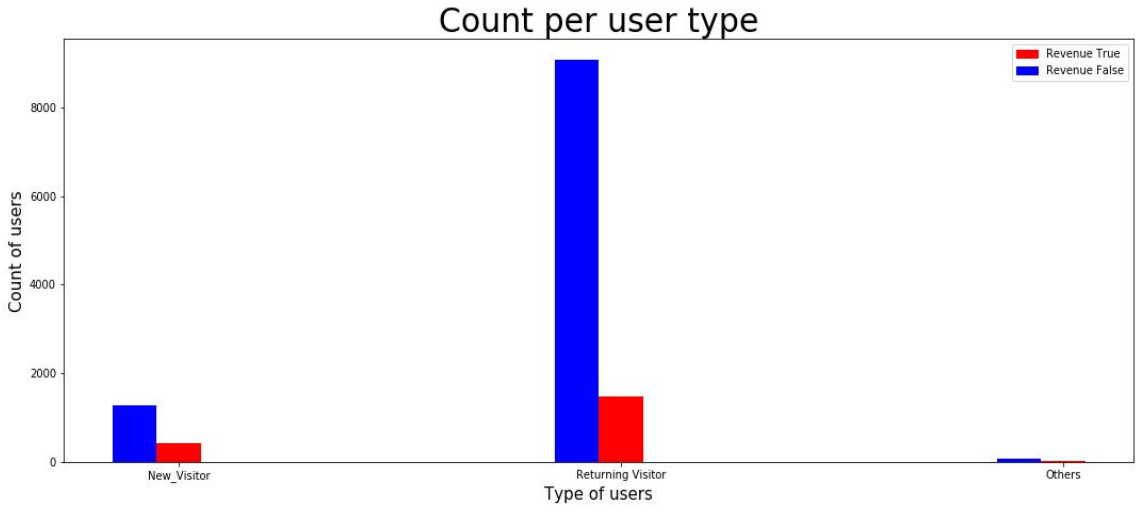




Let's now try to understand how may types of users land on our website and how the use it. We have 3 categories of users, the new users, returning users and others. When don't have confirmation of type of user we mark it as other.

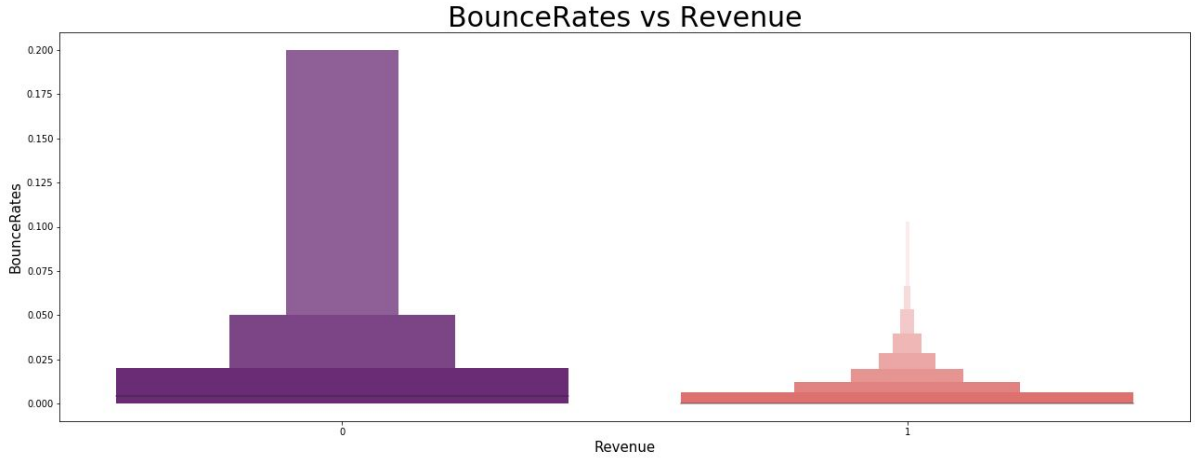


The below graph explains the distribution of the user in each type of user. We can see that the returning users give us the most of the revenue. This means we are not able to attract that many items to new visitors on our website. We need to work on that, also the number of new users are very less. We need to advertise the website itself to reach to the broader audience and that will help us increase sales.

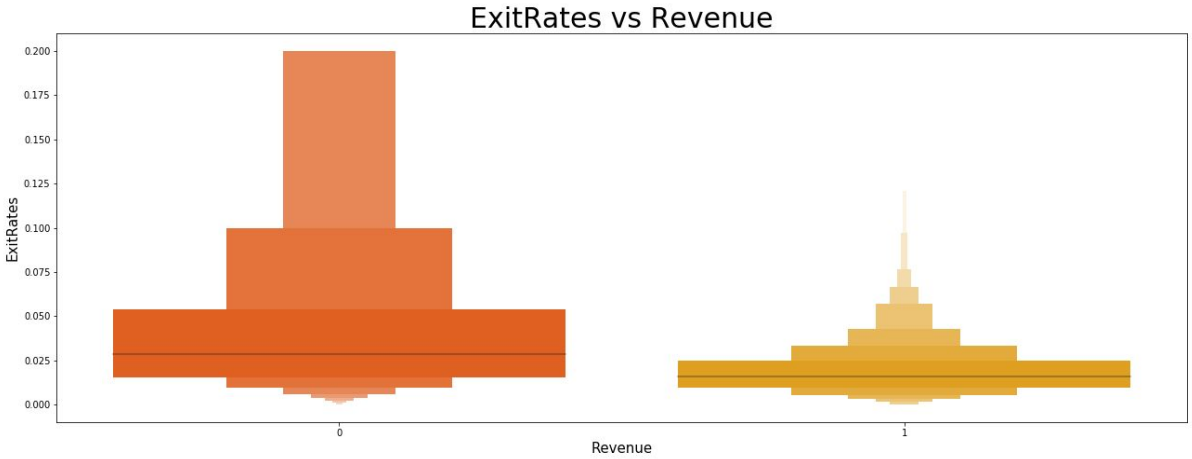


We also observed that most of the purchases are on the weekdays, we should come up with schemes and offers that will also attract customers on the weekends.

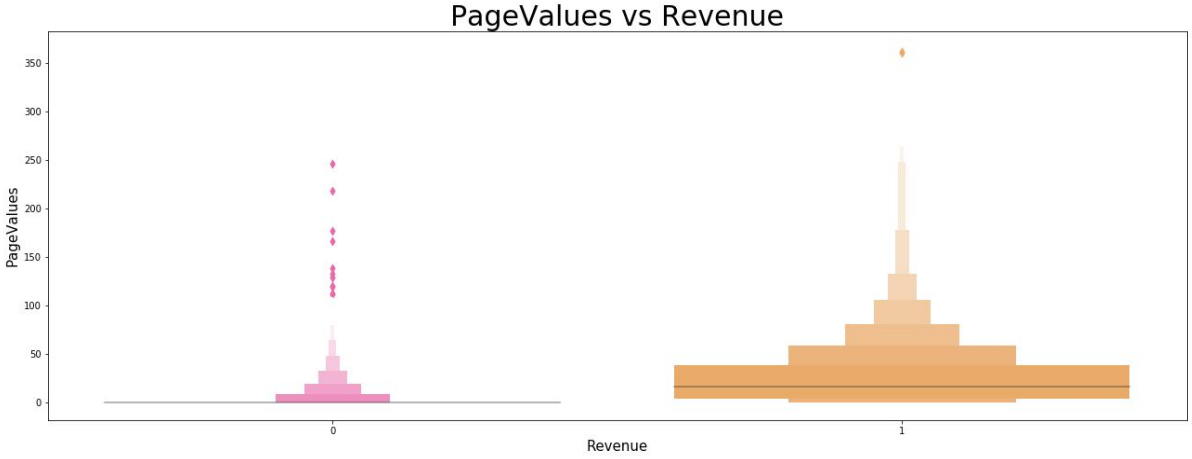
We have google analytics data for our website and that gives us a more detailed understanding of usage on each page. “A *bounce* is a single-page session on your site. In Analytics, a bounce is calculated specifically as a session that triggers only a single request to the Analytics server, such as when a user opens a single page on your site and then exits without triggering any other requests to the Analytics server during that session.” This is the definition of the Bonus rate on the google analytics help guide. We can see from the plot that we have high bounce rate for the users who didn't generate the revenue.



Exit Rate is the percentage of the users exit the page and close the session. We have the analysis of each pages exit rate and their count.



Understanding the page value for each transaction is also important let's understand what these values mean. "*Page Value* is the average value for a page that a user visited before landing on the goal page or completing an Ecommerce transaction (or both). This value is intended to give you an idea of which page in your site contributed more to your site's revenue. If the page wasn't involved in an ecommerce transaction for your website in any way, then the *Page Value* for that page will be $0 since the page was never visited in a session where a transaction occurred."



Future work

The purchasing prediction task proves to be an important starting point to online advertising and recommendation systems. Once we know that a user has bought a particular item or a set of items, we can use this information to give recommendations for buying further products. For instance if a user buys a new mobile phone, he will be interested to buy a protective case and a screen protector for the phone. So we could build a recommendation system which can do this job and using this we can send him targeted ads and offers for those products. This will help boost the online sales of that establishment.

This is also a very good starting point when a user does not buy a particular product but from the analysis we can come to know that he is interested in a particular product. These can be known from the time he has spent on the page and various other factors as explained in the previous sections. With this information we can again sed that user targeted offers for that product and other similar products, drawing him to close the purchase. These methods can be very effective in increasing sales and thus increasing the sales in the online platform. Apart from analysis, we can also look at other available techniques of tackling imbalanced data like ADASYN, Undersampling, etc.

Conclusion

In order to get a bird's eye view, let's take a look at some of the important findings of this analysis. One of the things that stood out the most was the inverse relationship between the time spent on a product page and the likelihood of that product being purchased. If a user spends more time on a product page, the probability of generating revenue from that product actually goes down. Such counterintuitive revelations were also accompanied by more expected behavior of users purchasing more during weekdays. This analysis also highlighted some of the areas of the online shopping website that can be improved, for eg: low number of new users indicate a need for better advertising. Although almost all models performed equally well when measured against their accuracies, Neural Networks and Random Forest Classifier proved to me more reliable when compared using ROC curves.

Overall, this exercise showcases the potential of data when paired with powerful data mining techniques. Through collection of relevant data and its exploitation via ingenious methods can put a spotlight on some of the most elusive yet crucial intentions of customers. This information can then be used to improve the user experience of a website which will ultimately lead to a rise in revenue for the company.

References

[1] Understanding online purchase intentions: contributions from technology and trust perspectives by Hans van der Heijden https://orsociety.tandfonline.com/doi/abs/10.1057/palgrave.ejis.3000445#.XUixW-hKhPY

[2] Online Purchasing Intention: Factors and Effects by Houda ZARRAD1, Mohsen Debabi https://pdfs.semanticscholar.org/4351/64ecb03da6892d8ca2d50983bc7cdaf25583.pdf

[3] Naive Bayes classifier, https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[4] Introduction to Naive Bayes Classification, https://towardsdatascience.com/introduction-to-naive-bayes-classification-4cffabb1ae54

[5] Logistic Regression. Simplified, https://medium.com/data-science-group-iitr/logistic-regression-simplified-9b4efe801389

[6] Understanding Random Forest. (2019). Medium. Retrieved 3 August 2019, from https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[7] Classification and Regression by Random Forest, Vol. 2/3, December 2002, Andy Liaw and Matthew Wiener.

[8] Understanding Support Vector Machines Algorithm, https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[9] Neural networks for classification: a survey - IEEE Journals & Magazine. (2019). Ieeexplore.ieee.org. Retrieved 3 August 2019, from https://ieeexplore.ieee.org/document/897072/citations#citations

[10] An online purchase intentions model: The role of intention to search: The Sixth Triennial AMS/ACRA Retailing Conference, 2000☆1 by SoyeonShim, Mary AnnEastlick, Sherry LLotz, PatriciaWarrington

[11] Google analytics help guide https://support.google.com/analytics/?hl=en#topic=3544906