PROJECT REPORT

Online Shoppers Purchasing Intention Classification

Table of Contents:

1. PROBLEM DESCRIPTION

This is a classification problem with the target variable “Revenue” which has 2 categories, True or False. The goal is to predict whether a session will produce Revenue with the help of features containing several types of session information.

2. DATA SOURCE

UCI Repository - Online Shoppers Purchase Intention

3. DATA DESCRIPTION

The dataset has information about customers to an online store and their browsing behavior and whether they made a purchase or not. The dataset was collected from an online store over 9 months between October 2012 and June 2013. It has 10 numerical and 8 categorical variables. The variable 'Revenue' can be used as the target variable.

The dataset consists of 12,330 instances and 18 attributes. The detailed description of each feature is mentioned below:

Feature Name	Feature Description
Administrative	This is the number of administrative-type pages visited by the user on the website.
Administrative Duration	Duration of time (in seconds) that the user spent on the administrative pages of the website.
Informational	This is the number of informational pages visited by the user on the website.
Informational Duration	Duration of time (in seconds) that the user spent on the website's informational pages.
Product-Related	This is the number of product-related pages visited by the user on the website.
Product-Related Duration	Duration of time (in seconds) that the user spent on the website's product-related pages.
Bounce Rate	Bounce Rate for a webpage is the percentage of users who entered the website from that page and bounced/left the website without triggering any other requests during that session.
Exit Rate	The value of "Exit Rate" feature for a web page is calculated as for all pageviews to the page, the percentage that were the last in the session.
Page Value	Page Value of a web page represents the average value for the web page that a user visited before completing an e-commerce transaction.
Special Day	The value of Special Day feature indicates the closeness of the site visiting time to a specific special day (e.g., Father’s Day, Valentine's Day) in which the sessions are more likely to be finalized with transaction.
Operating System	Categorical Feature showing the User’s Operating System.

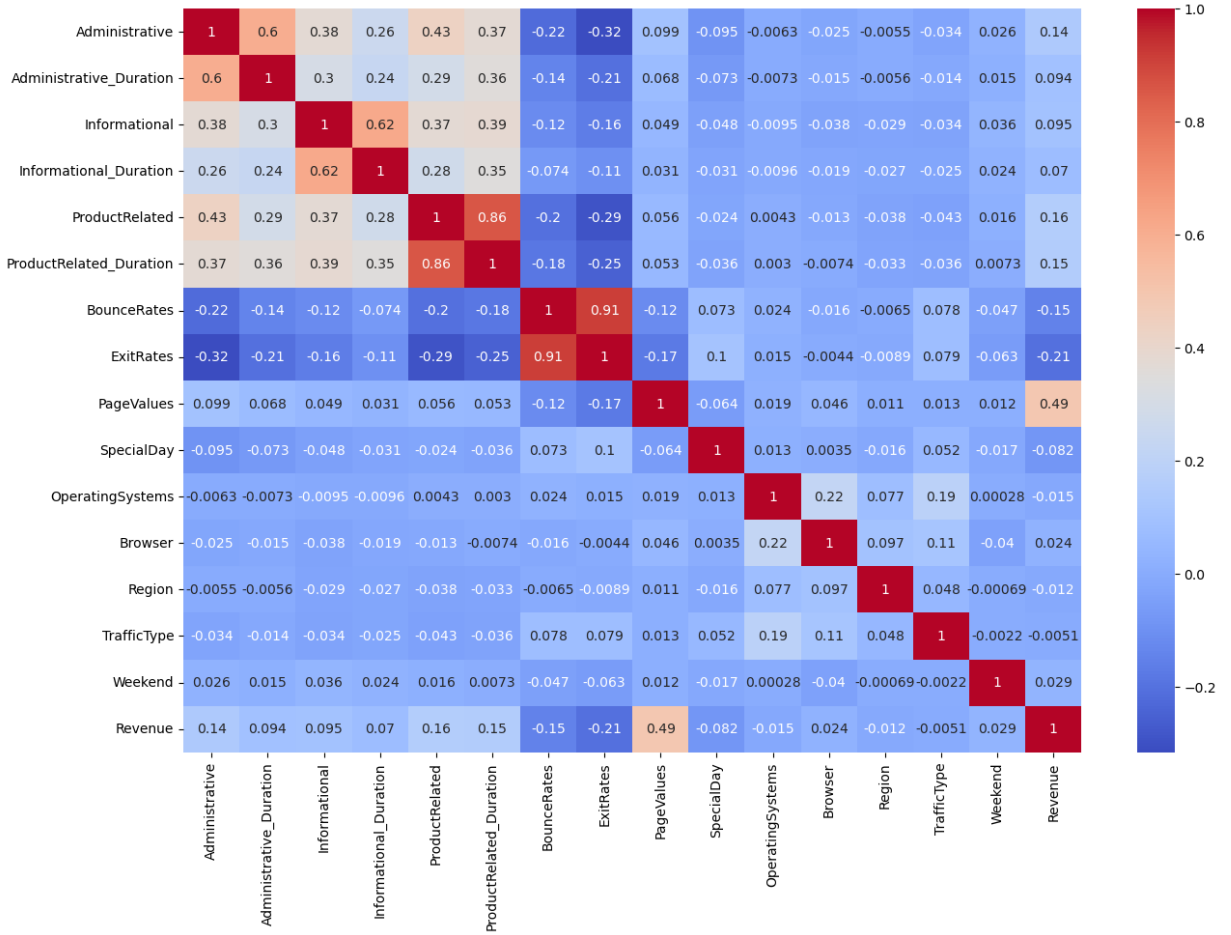
Browser	Browser information for the user.
Region	Categorical feature containing user's region.
Traffic Type	This is the type of traffic source for the user (e.g., search engine, social media, etc.).
Visitor Type	Shows if the user is returning or a new visitor
Weekend	Boolean value indicating whether it is the weekend.
Month	Month of the year.

Information on the datasetr was obtained from [1]

4. DATA EXPLORATION:

4.1. Correlation Analysis:

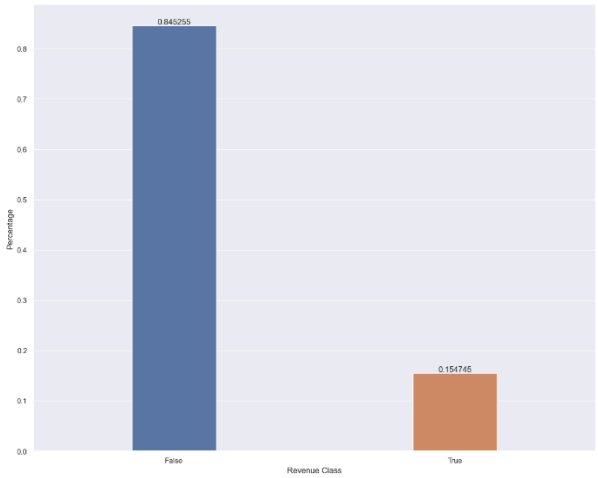
To capture the relationship between all the independent variables, we built a correlation heatmap. It can be seen from the map that variables like Product Related/ProductRelated_Duration (0.86), Bounce Rates/Exit Rates (0.91), Administrative/Administrative Duration (0.6), and Informational/Informational Duration (0.62) have high correlation between them. These features were removed during feature selection.



4.2. Class Imbalance:

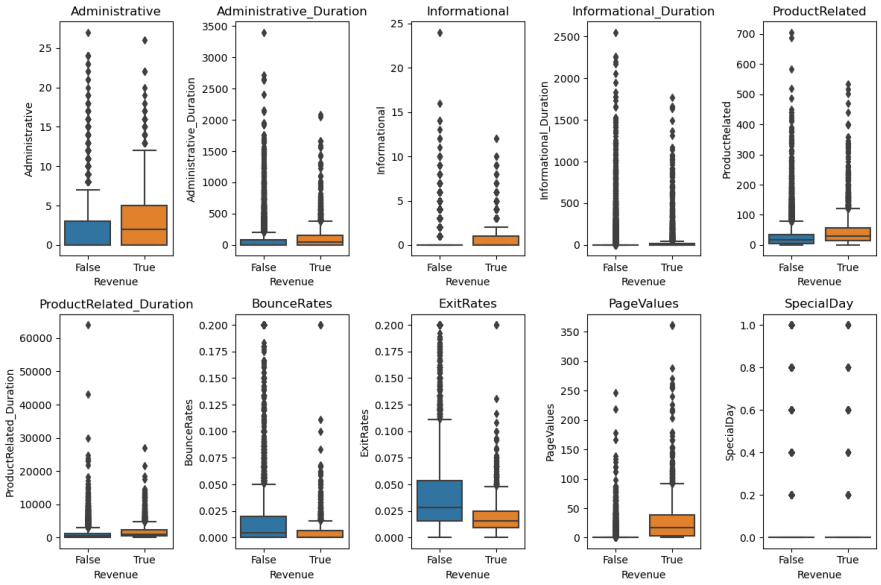
A bar plot was plotted to check the imbalance of the classes in the dataset. It was found that the percentage of each class in the dataset is:

Class TRUE: 0.155 %
Class FALSE: 0.845 %



4.3. Boxplots:

The following boxplots show the values of each feature plotted against the respective revenue variable. Looking at the domain knowledge and the way the data was collected [1], we do not remove the outlier values as there might be occurrences of those in the test dataset.



4.4. Exploratory Data Analysis:

By carrying out exploratory data analysis of each feature, we obtained the mean, minimum, maximum, and other descriptive metrics of all numerical features, and the value counts for categorical features.

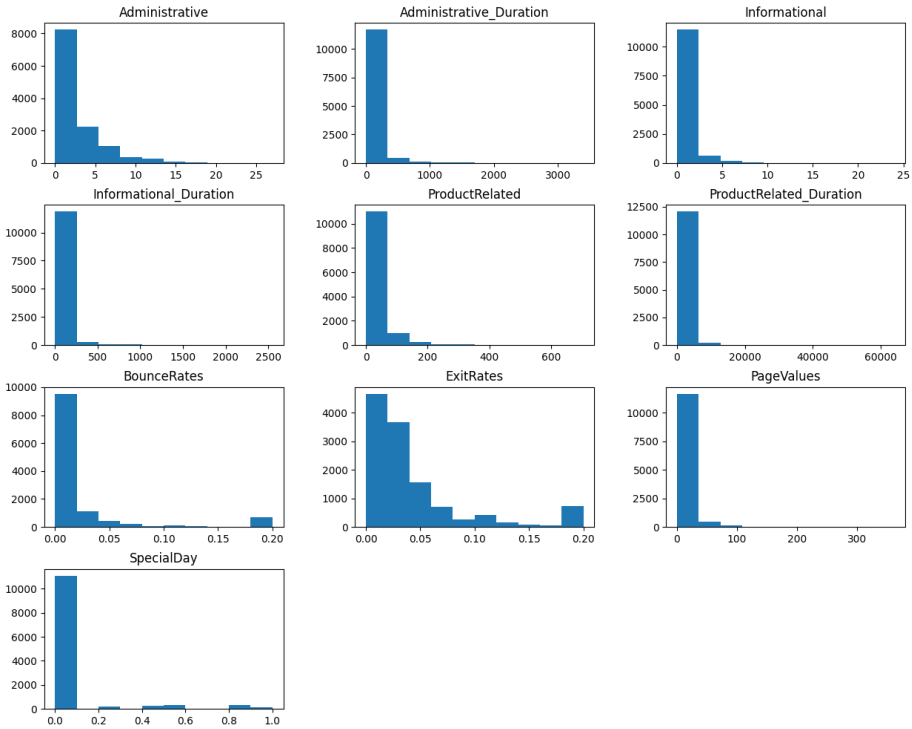
	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	ExitRates	PageValues	SpecialDay
count	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000
mean	2.315166	80.818611	0.503569	34.472398	31.731468	1194.746220	0.043073	5.889258	0.061427
std	3.321784	176.779107	1.270156	140.749294	44.475503	1913.669288	0.048597	18.568437	0.198917
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	7.000000	184.137500	0.014286	0.000000	0.000000
50%	1.000000	7.500000	0.000000	0.000000	18.000000	598.936905	0.025156	0.000000	0.000000
75%	4.000000	93.256250	0.000000	0.000000	38.000000	1464.157214	0.050000	0.000000	0.000000
max	27.000000	3398.750000	24.000000	2549.375000	705.000000	63973.522230	0.200000	361.763742	1.000000

The figure below shows the no. of categories in each of the categorical variables in the dataset:

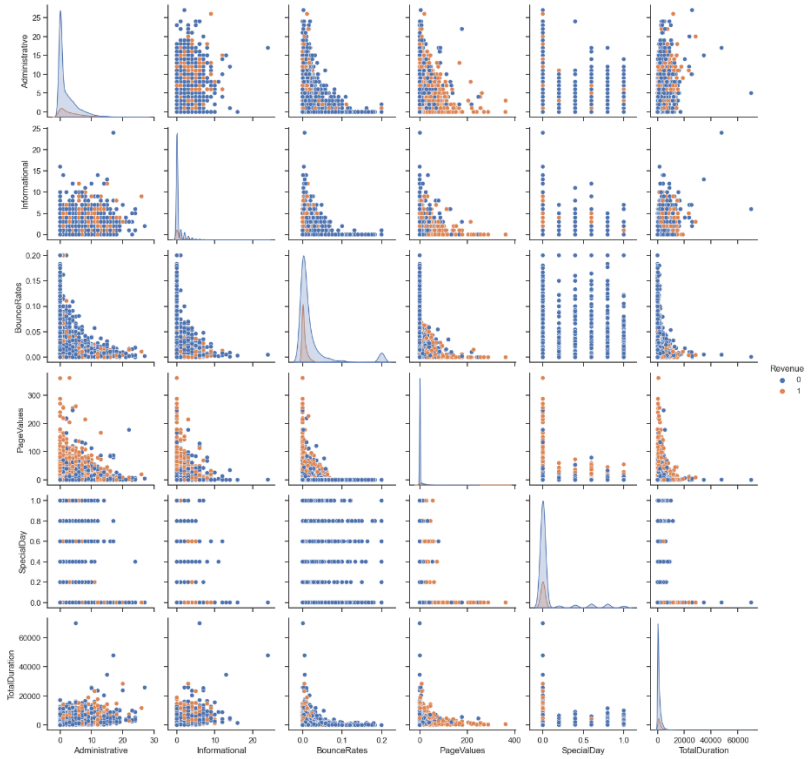
```
 Month              10
OperatingSystems     8
Browser             13
Region               9
TrafficType         20
VisitorType          3
Weekend              2
Revenue              2
```

The categorical features Month, and Visitor Type were one-hot encoded and the weekend and revenue features were label encoded. This was done so that the model doesn't assign importance to features with a higher numerical values after encoding.

To understand the distribution of the numerical variables, a histogram of all the numerical variables was plotted. It was seen that there is skewness in every variable.

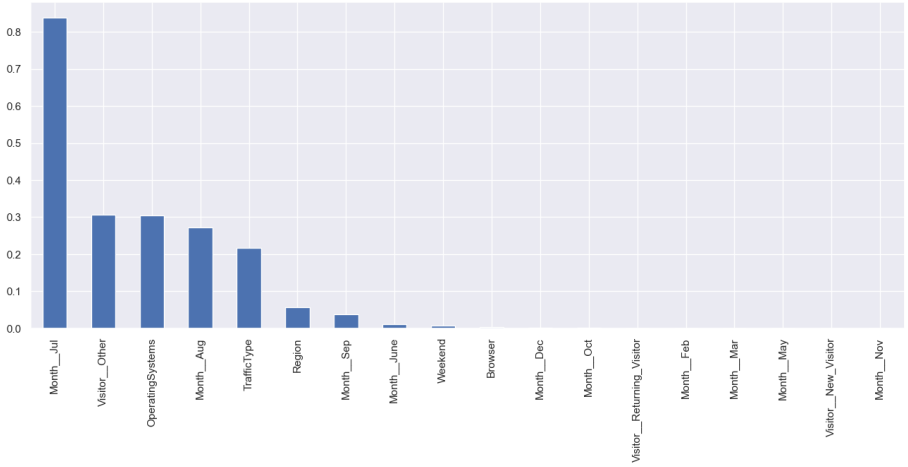


Following the EDA, we plot a pairplot of the numerical features to check the linear separability of the classes. The plot shows the scatter plot of the respective features with the separator as our target variable. We can observe the two classes are majorly concentrated together and for the most part they overlap. We can infer that the two classes are not linearly separable.

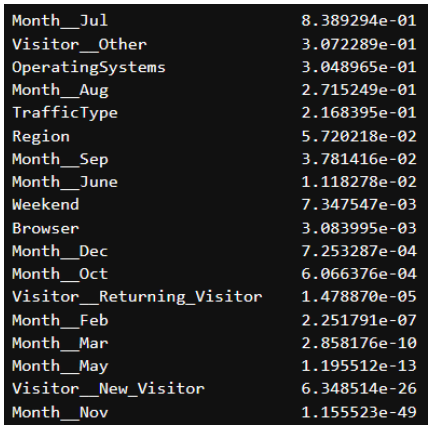


4.5. Feature Selection

We use the Pearson's R for assessing the relation between numerical features and use the chi squared test to assess the relation between categorical variables. To employ the chi-square test, the categorical features Month, Weekend, and Visitor Type were one-hot encoded and the test was applied on the updated categorical features. The p-values obtained for each feature is indicated in the plot, where we observe the features, Month_Jul, Visitor_Other, OperatingSystems, Month_Aug, Traffic_Type, and Region have a p-value of > 0.05, so we drop those features as they are not significant with our Target variable Revenue'.



The p-values obtained from the chi-squared test are:



Looking at the domain knowledge, we can see that the features Administrative Duration, Informational Duration, and Product related Duration combine to give the total time spent by the customer on the website. So, we combine the three features into a single one, namely TotalDuration.

With these feature selection techniques, we reduce the number of features for models from 28 to 18. This would enable the model to learn faster, and its complexity also decreases in the process.

5. MODEL IMPLEMENTATION

Following the feature selection and checking for linear separability, we decided to implement Naïve Bayes, Logistic Regression, Soft Margin SVM, and Neural Networks.

The data was split into train, validation, and test data in the proportion:

Dataset	Split Ratio
Training Set	80%
Test Set	20%

The training set is used for the learning process, which is to fit the parameters of the classifier. The validation dataset is used so that the model parameters can be tuned to improve the model performance and provide an unbiased evaluation of the model fit on the training dataset. For example, selecting the number of hidden layers and nodes in a neural network, the learning rate in classical machine learning models. The test set is used to get an unbiased performance evaluation of the fully fitted model.

5.1. Logistic Regression

Since our model is a classification problem, with the end goal being to predict whether a website visit is converted to a sale or not, it is appropriate to assume that Logistic regression is a good algorithm to start as it is the most common classification model. Using Logistic regression, we have predicted the “Revenue” on a test dataset and calculated the Accuracy, precision, recall and f1

score on the test dataset based on the confusion matrix that has been generated by the results.

The gradient descent algorithm has been used in the development of this model. The minimum of a differentiable function can be found via an iterative optimization method known as gradient descent. We test several variables during this process and adjust them until we identify the ones that reduce the output.

The learning rate controls how often we update the parameters. We might "overshoot" the optimal value if the learning rate is too high. The same holds true if it is too tiny; getting to the ideal levels will take too long. The sigmoid function has been used to map an extensive spectrum of input values into a narrow range. Mathematically, sigmoid function is given as:

$$y = g(z) = \frac{1}{1 + e^{-z}} = \frac{e^{z}}{1 + e^{z}}$$

A function called sigmoid has been created in the code to calculate the sigmoid function.

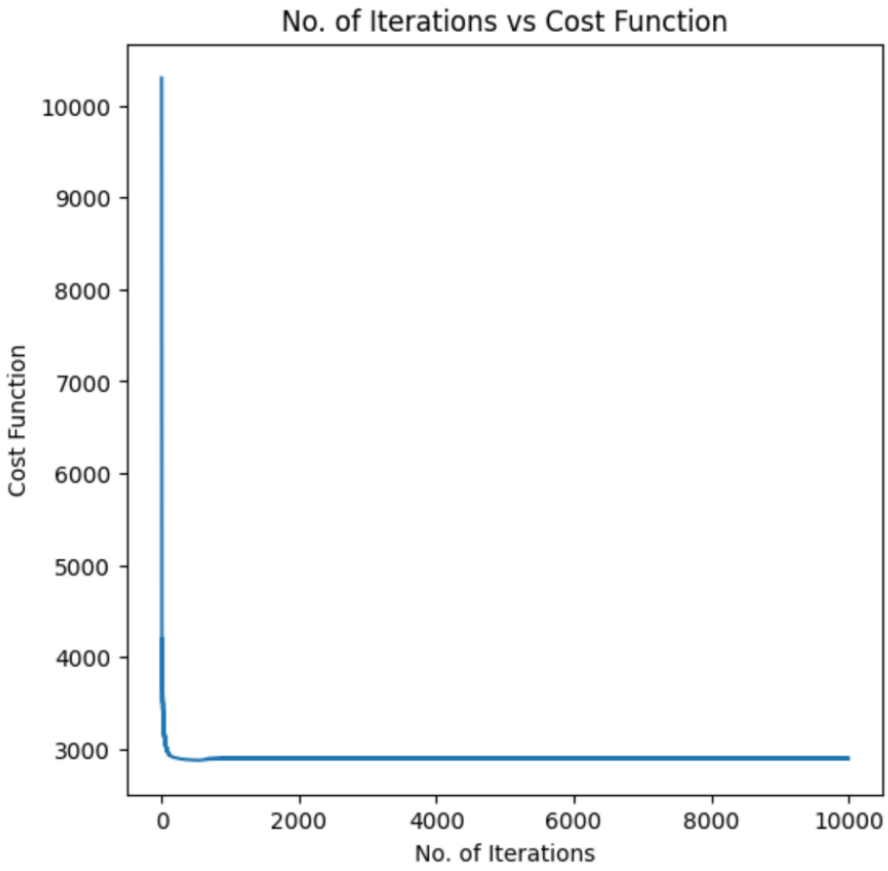
The cost function controls how well the model performs. It is used to evaluate how well the model's predictions correspond to the observed outcomes. For "m" observations, the cost can be calculated quantitatively as follows:

$$J(\theta) = \frac{1}{m}[\sum_{i=1}^{m} -y^{(i)} log(h_\theta(x^{(i)})) + (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))]$$

Parameters:

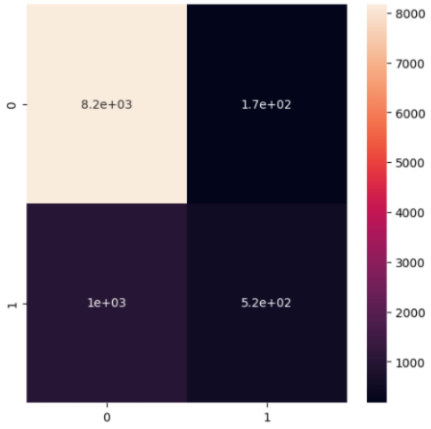
Number of steps in descent: 100000

Learning Rate: 0.0001



After training the model on the training dataset, the following results were observed:

Confusion matrix:



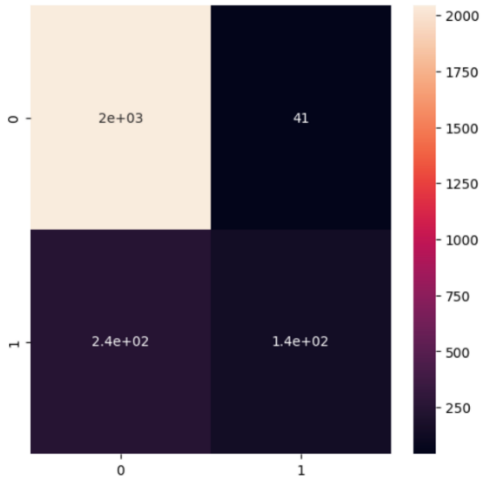
Interpretation of the above plot:

	Predicted 0	**Predicted 1**
0	**8165**	**173**
1	**1005**	**521**

	precision	recall	f1-score	support
0	0.89	0.98	0.93	8338
1	0.75	0.34	0.47	1526
accuracy			0.88	9864
macro avg	0.82	0.66	0.70	9864
weighted avg	0.87	0.88	0.86	9864

Results on testing data:

Confusion Matrix:



Interpretation of the above plot:

	Predicted 0	**Predicted 1**
0	**2043**	**41**
1	**240**	**142**

	precision	recall	f1-score	support
0	0.89	0.98	0.94	2084
1	0.78	0.37	0.50	382
accuracy			0.89	2466
macro avg	0.84	0.68	0.72	2466
weighted avg	0.88	0.89	0.87	2466

Conclusion:

Based on the provided metrics for both the training and test data, the model has performed well in terms of precision and recall for class 0 (customers for whom revenue is false).

For class 0, the precision and recall values are high, which indicates that the model is correctly identifying a high percentage of instances where the customer did not buy, and the instances that it classified as not buying are indeed not buying. This high precision and recall value for class 0 suggests that the model is performing well in identifying customers who are unlikely to buy, which can be beneficial for the business to focus on other strategies to attract them to buy.

However, for class 1 (customers for whom revenue is true), the precision and recall values are low, indicating that the model is not performing well in identifying customers who are likely to buy. This lower precision and recall value for class 1 suggests that the model may be missing some potential customers who are likely to buy, which can negatively impact business revenue.

Overall, the accuracy score is not a good metric to evaluate the model's performance in this case, as there is an imbalance in the class distribution. The model is performing well for class 0 but not for class 1, so the F1 score is a better metric that considers both precision and recall.

In conclusion, the model may require further tuning or improvement to improve its performance in identifying potential customers who are likely to buy (class 1).

5.2. Naïve Bayes:

Naïve Bayes is a probabilistic classification algorithm that calculates the probabilities of an instance belonging to all existing classes using Bayes Theorem and predicts the class with the highest probability. Gaussian Naïve Bayes is the extension of Naïve Bayes.

Gaussian Naïve Bayes works on two strong assumptions that all features of the dataset are independent, and the continuous features come from a Gaussian distribution. This assumption makes Gaussian Naïve Bayes one of the simplest models to implement as all we need is to calculate the mean and standard deviation for the training data.

The following is the Bayes Theorem calculation in its simplest version, where the marginal probability of the occurrence P(B) is referred to as the prior and the probability that we are interested in computing P(A|B) is called the posterior probability. P(B|A) is the likelihood that we calculate for each instance with respect to the independent variables.

Mathematically, conditional probability of A given B can be denoted as

$$P(A|B) = (P(B|A) * P(A)) / P(B)$$

For each class, we calculate the likelihoods (P(B|A)) and multiply them with the prior probability of the class (P(A)). Since the total probability (P(B)) will be the same for the entire dataset, it can be ignored during the calculation.

For a categorical variable, the likelihood of an instance for a class will be the count (occurrence) of that category in the column for that class divided by the count (occurrence) of that category in the entire dataset.

For a numerical variable, the likelihood of an instance for a class comes from the probability distribution function (mean, standard deviation) of that variable in that class.

Since all the variables are assumed to be independent, the combined likelihoods of features B1, B2,….,BN can be represented as the product of individual likelihoods:

P(B|A) = P(B1|A) * P(B2|A) *….*P(BN|A)

We implemented Gaussian Naïve Bayes on the training data to obtain the following results:

```
Precision of the model is : 0.8333333333333334
Recall of the model is : 0.0037425149700598802
F1_score of the model is : 0.007451564828614009
```

Results for the testing data were as follows:

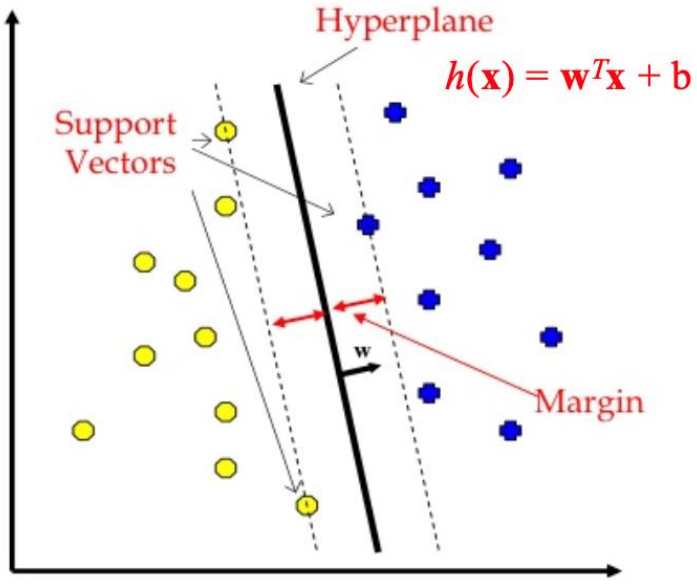
```
Precision of the model is : 0.5897435897435898
Recall of the model is : 0.04020979020979021
F1_score of the model is : 0.07528641571194762
```

5.3. SVM (Support Vector Machines)

SVM is a model for classification based on maximum margin. The goal is to find an optimal hyperplane that maximizes the separation margin between the classes.

Objective function for SVM is given as:

$$\begin{aligned} \underset{\mathbf{w},\, b}{\text{minimize}} \quad & \|\mathbf{w}\|_2^2 \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 \quad \forall i \in \{1, \ldots, n\} \end{aligned}$$



For a Soft-Margin SVM, the optimization function becomes:

$$\begin{aligned} \underset{\mathbf{w},\, b,\, \zeta}{\text{minimize}} \quad & \|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\zeta_i \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0 \quad \forall i \in \{1, \ldots, n\} \end{aligned},$$

where ζi is the error associated with each misclassification or each point which is within or on the wrong side of the margin, and 'C' is the penalty associated with each misclassification error.

The Dual problem for this optimization problem is given by:

$$\textbf{Objective Function: } \max_{\boldsymbol{\alpha}} \; L_{dual} = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$$

$$\textbf{Linear Constraints: } 0 \leq \alpha_i \leq C, \;\; \forall i \in \mathbf{D} \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0$$

The KKT conditions, which check for the optimal solutions between primal and dual models are:

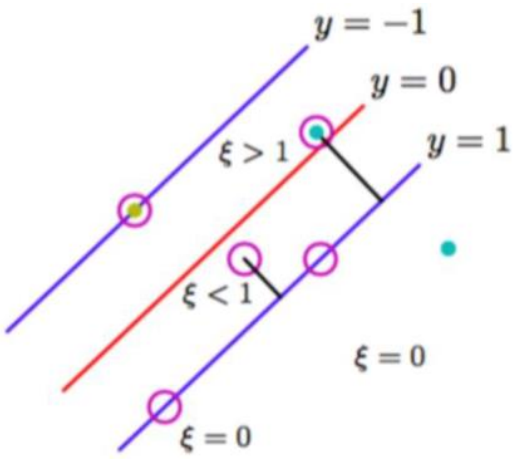
$$\alpha_i \left(y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right) = 0 \text{ with } \alpha_i \geq 0$$

$$\beta_i (\xi_i - 0) = 0 \text{ with } \beta_i \geq 0$$

For the data point with $\alpha_i > 0$

(1) $\xi_i > 0$, which implies that $C - \alpha_i = 0$, that is, $\alpha_i = C$, or

(2) $C - \alpha_i > 0$, that is $\alpha_i < C$. => $\xi_i = 0$



Owing to the computational complexity of the SVM model, we initially took a sample of 500 records from the pre-processed data to run through the model. The processing time in seconds that the model took was obtained using the process_time() python function.

CPU processing time when the SVM model was run: 54.140625 seconds

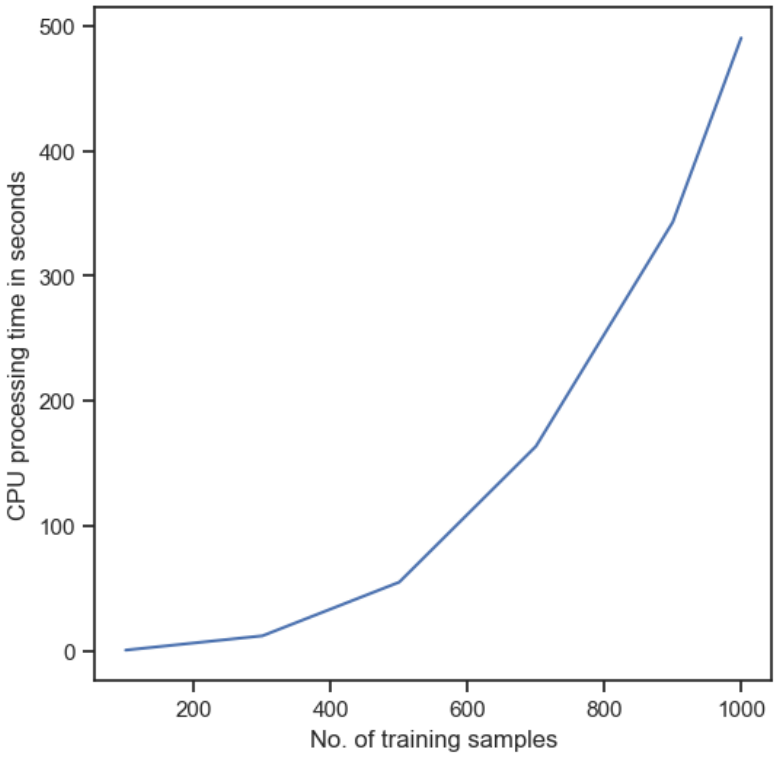
Time taken to train the SVM model: 59.5065 seconds

The metrics obtained from the 500 records sample were:

[[409 16]
[32 43]]

The f1 score of the model was obtained as 64.18 %
The precision of the model was obtained as 72.88 %
The recall of the model was obtained as 57.3%

To assess the complexity of the model, we plotted the CPU processing with the no. of samples used to train the model in an iterative manner. From the plot below, we can see that the time complexity of our SVM model in the big-O notation is $O(n^2)$.



Since, training a model with 500 samples takes ~ 1 min for the SVM model, the model training more no. of samples was not implemented with the existing graphics of the computer on Jupyter Lab. However, Compared the Naïve Bayes and Logistic Regression, the Soft Margin SVM model performed better.

5.4. Neural Networks

For our dataset, we have used a simple fully connected neural network (dense neural network), with 1 input layer, 1 hidden layer, and an output layer where the sigmoid function is applied to give a probability between 0 and 1 for the respective classes. In this neural network, the layers are interconnected in a feedforward way where each neuron is connected to every other neuron in the subsequent layer. In the hidden layers, activations such as relu, tanh, etc can be applied. But for our model we mainly work with relu in the hidden layers and a sigmoid in the output layer.

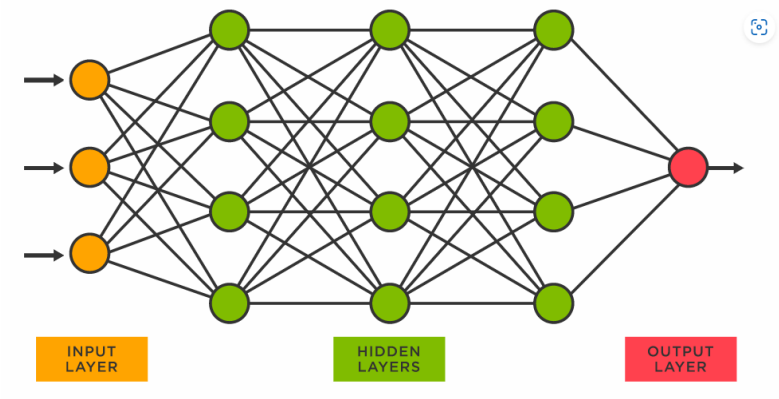
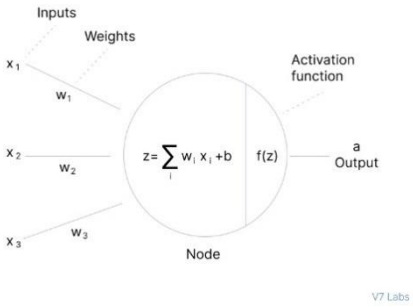


Fig: A dense neural network - Example

A neural network functions in the following way where for each node, a weighted sum is calculated as an input and that is then fed into the activation function, the result of which would be the input for the subsequent layers.



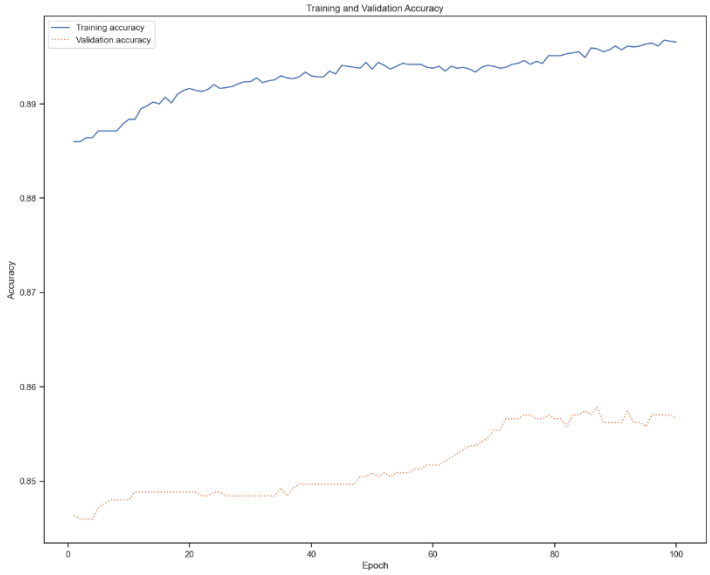
The model summary for our model is:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Hidden_layer_1 (Dense)      (None, 8)                 216

 Output_layer (Dense)        (None, 1)                 9

=================================================================
Total params: 225
Trainable params: 225
Non-trainable params: 0
_________________________________________________________________
```

After training the model with the above parameters, we did a cutoff analysis using precision and accuracy. The candidate cutoff values were chosen to be [0.1, 0.2, 0.5, 0.8, 0.95]. The results obtained were:

	cutoff	precision	accuracy
0	0.10	0.903014	0.725625
1	0.20	0.711009	0.862249
2	0.50	0.329620	0.884678
3	0.80	0.024902	0.868292
4	0.95	0.002621	0.851188

Analyzing the results, we decided to keep the cutoff value as 0.2 for testing the model on the test set.



Comparing the plots for the training and validation accuracy, we can see that as the no. of epochs increase the validation accuracy is consistently low as compared to the training dataset. In addition, the validation is also consistently increasing that can imply that the model can generalize well when tuned with the proper hyperparameters.

6. CONCLUSION

The goal of our model was to determine the likelihood that a customer would purchase a product from a website, we had to make a trade off between accuracy of the model and the precision and recall. Since classifying whether a customer would make a purchase can affect the resources put into the marketing for a website. On the other hand, if the classification is incorrect there is a risk of losing value and money.

With the given dataset and its high imbalance, feature selection and its related tasks were important as they directly affect our machine learning model. With stratification and feature selection the models performed better when compared to training the models with feature selection. From the above metrics we can say that Neural Networks provide the best accuracy, but they also risk overfitting on the training set. SVM performed better than Naïve Bayes and Logistic Regression, but the computational complexity of the model prevents us from training the model on the complete training data.

7. REFERENCES

[1]. Sakar, C. Okan, et al. "Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks." *Neural Computing and Applications* 31 (2019): 6893-6908.