# ONLINE SHOPPERS PURCHASING INTENTION CLASSIFICATION


Northeastern University
College *of* Engineering

## IE 7300 STATISTICAL LEARNING FOR ENGINEERS

**FINAL PROJECT REPORT**

**TEAM MEMBERS:**

AMAN SUNIL KUMAR
VISHNU RAM DINESAN
SHAIKH ZEESHAN ALI
AKSHAY DWIVEDI

**DATE: APRIL 17TH, 2023**

**SPRING 2023**

**MECHANICAL AND INDUSTRIAL ENGINEERING DEPARTMENT**

# Abstract:

In this project, we demonstrate our study, where we predict online shoppers' buying intentions based on the time they spent on the website. To support our study, we take into consideration, factors like Month/Week/Day, Average Time Spent on Each Product, Region, etc., and the impact these factors have on the user's activity. To deep dive into the analysis, we start with the Exploratory Data Analysis where we perform Feature Engineering, Feature Selection, and created visualizations in order to understand the different factors influencing consumer behavior. We preprocessed the data and by means of visualization, we were able to figure out that there was an imbalance in the dataset, for which we used SMOTE for oversampling, giving us a balanced output. Following the data-wrangling process, we implemented and assessed the algorithms and compared the models to determine the best model.

# Introduction:

The goal of this project is to predict whether a customer is likely to purchase an online shopping session. This type of prediction could be valuable for an online retailer, as it could help them optimize their website design, marketing strategies, and customer engagement efforts. The problem is classified by two determining factors, True (if the shopper ends up shopping) and False (if they do not).

The online shoppers purchasing intention classification dataset contains the information of each session the shopper visits online, based on which a prediction is made whether a person is shopped/ not shopped. We use 3 models to classify the observations. First, we propose a baseline model and then implement Logistic Regression, Naïve Bayes, and Neural Networks to classify the results.

# Data Description:

The online shopper's purchasing intention classification dataset includes 18 features & 12,330 records about various individuals who made online purchases over the course of a year, with 17 feature variables and 1 goal variable, "Revenue," included. This project's objective is to forecast variable Revenue utilizing their pertinent attributes. Around 84.5% of the 12,330 sessions are negative, while 15.5% are positive. 10 numerical and 7 category factors were used to predict the class.

The description of each variable is shown in the table below:

| ***Predictor*** | ***Type*** | ***Description*** |
|---|---|---|
| Administrative | Numerical | Visits to administrative pages throughout the session |
| Administrative Duration | Numerical | Time elapsed on administrative pages |
| Informational | Numerical | The number of informational pages accessed throughout the session |
| Informational Duration | Numerical | Total amount of time spent on information pages |
| Product Related | Numerical | Pages connected to products that were viewed throughout the session in number |
| Product Related Duration | Numerical | Time spent overall on product-related pages |
| Bounce Rate | Numerical | The proportion of visitors who arrive on that page of the website and leave without performing any further tasks. |
| Exit Rate | Numerical | The proportion of website pageviews that conclude on that particular page. |
| Page Value | Numerical | The value of the target page's averaged value and/or the conclusion of an eCommerce transaction |
| Special Day | Numerical | The closeness of the browsing date to special days or holidays is indicated by this value. |
| Month | Categorical | Contains the session's month in string form. |

| | | |
|---|---|---|
| Operating Systems | Categorical | An integer value that represents the user's operating system at the time the page was viewed. |
| Browser | Categorical | A value that represents the user's browser when they viewed the page. |
| Region | Categorical | An integer indicating the area in which the user is situated. |
| Traffic Type | Categorical | A numeric value that indicates the user's traffic classification. |
| Visitor Type | Categorical | A string indicating a visitor's status as New Visitor, Returning Visitor. |
| Weekend | Categorical | Whether the session is on a weekend is indicated by a Boolean. |

# Methods:

1. **Logistic Regression:**

   Logistic regression is a parametric classification model that calculates the likelihood that a record will fall into the success category. We can categorize the output variable by converting these probabilities to either 0 or 1 based on a threshold. A linear function of the predictors can be used to model the outcome variable.

2. **Naïve Bayes:**

   As a classification model, the Gaussian Naïve Bayes algorithm is used to entail and manageably simplify the computation of probability for each hypothesis. When the target values are given, the attribute values are assumed as conditionally independent and are computed as P(d1|h) * P(d2|H) and continue, instead of computing every single attribute value individually, P(d1, d2, d3|h). The file of a fitted naive Bayes model contains a list of probabilities.

3. **Neural Networks:**

   Neural networks, just like a human brain, solve problems using the complex computational technique. There are three layers in a Neural Network algorithm; an input layer, an output layer, and a hidden layer. Just like neurons in a human brain, these 3 layers are connected via nodes and form a network. Neural Networks is widely practiced in the real world and form a base for Deep Learning.

# Exploratory Data Analysis (EDA):

1. **Understanding the data:**

   To get a broad perspective of the data and the datatypes, we first load the initial five records:

```
df.head()
```

| | Administrative | Administrative_Duration | Informational | Informational_Duration | ProductRelated | ProductRelated_Duration | BounceRates | ExitRates | PageValues |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0.0 | 0 | 0.0 | 1 | 0.000000 | 0.20 | 0.20 | 0.0 |
| 1 | 0 | 0.0 | 0 | 0.0 | 2 | 64.000000 | 0.00 | 0.10 | 0.0 |
| 2 | 0 | 0.0 | 0 | 0.0 | 1 | 0.000000 | 0.20 | 0.20 | 0.0 |
| 3 | 0 | 0.0 | 0 | 0.0 | 2 | 2.666667 | 0.05 | 0.14 | 0.0 |
| 4 | 0 | 0.0 | 0 | 0.0 | 10 | 627.500000 | 0.02 | 0.05 | 0.0 |

```
df.head()
```

| | ProductRelated_Duration | BounceRates | ExitRates | PageValues | SpecialDay | Month | OperatingSystems | Browser | Region | TrafficType | VisitorType | Weekend | Revenue |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0.000000 | 0.20 | 0.20 | 0.0 | 0.0 | Feb | 1 | 1 | 1 | 1 | Returning_Visitor | False | False |
| | 64.000000 | 0.00 | 0.10 | 0.0 | 0.0 | Feb | 2 | 2 | 1 | 2 | Returning_Visitor | False | False |
| | 0.000000 | 0.20 | 0.20 | 0.0 | 0.0 | Feb | 4 | 1 | 9 | 3 | Returning_Visitor | False | False |
| | 2.666667 | 0.05 | 0.14 | 0.0 | 0.0 | Feb | 3 | 2 | 2 | 4 | Returning_Visitor | False | False |
| | 627.500000 | 0.02 | 0.05 | 0.0 | 0.0 | Feb | 3 | 3 | 1 | 4 | Returning_Visitor | True | False |

Now that we have an idea of the features we are going to work with, we assess if the records have any null values and receive the following output:

```
df.isnull().sum()
```

```
Administrative             0
Administrative_Duration    0
Informational              0
Informational_Duration     0
ProductRelated             0
ProductRelated_Duration    0
BounceRates                0
ExitRates                  0
PageValues                 0
SpecialDay                 0
Month                      0
OperatingSystems           0
Browser                    0
Region                     0
TrafficType                0
VisitorType                0
Weekend                    0
Revenue                    0
dtype: int64
```

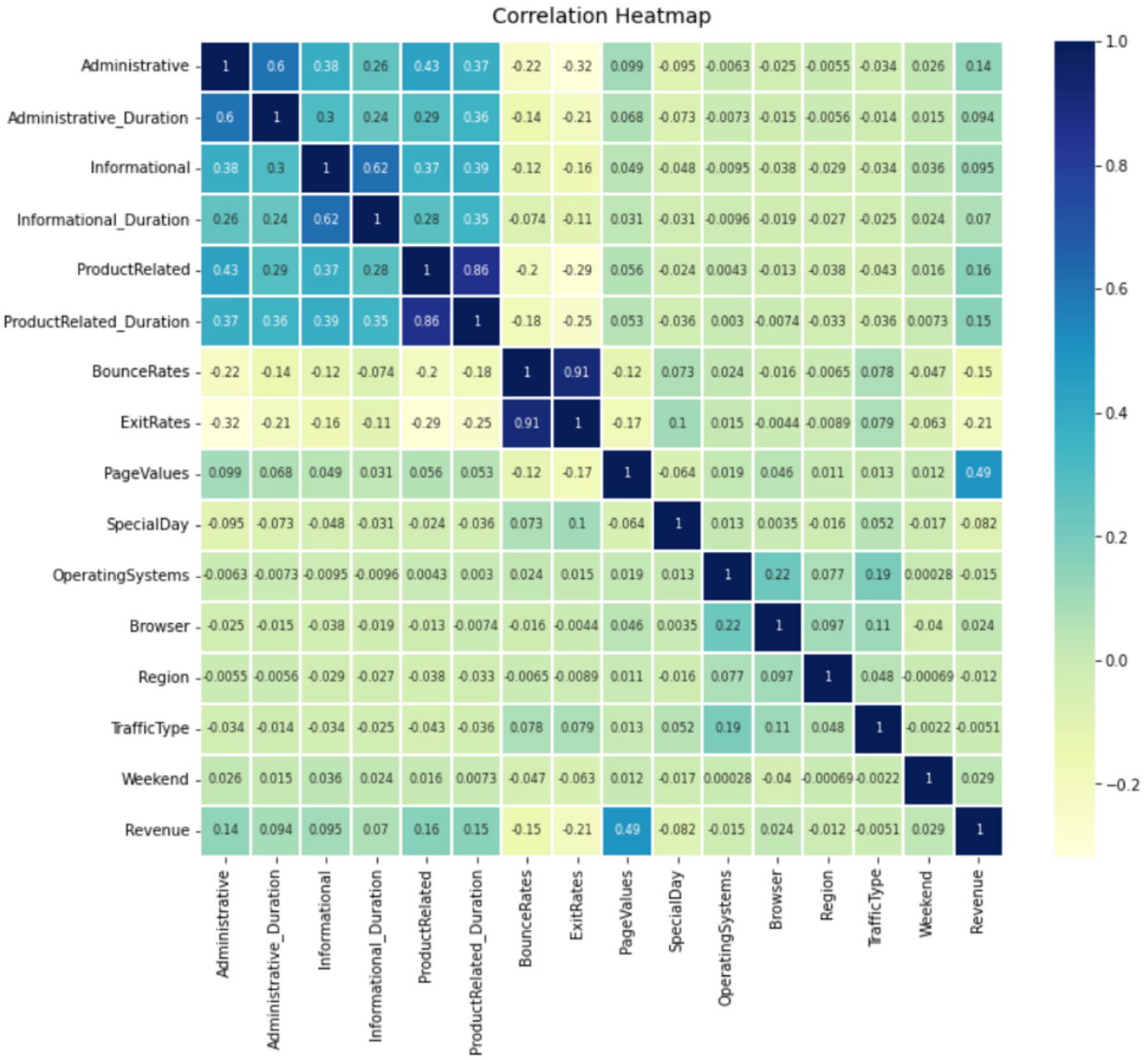
We also wanted to understand the datatypes, because if we need to convert any features which may not be able to be implemented in the modeling, we can make them either numerical or categorical variables.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Administrative           12330 non-null  int64  
 1   Administrative_Duration  12330 non-null  float64
 2   Informational            12330 non-null  int64  
 3   Informational_Duration   12330 non-null  float64
 4   ProductRelated           12330 non-null  int64  
 5   ProductRelated_Duration  12330 non-null  float64
 6   BounceRates              12330 non-null  float64
 7   ExitRates                12330 non-null  float64
 8   PageValues               12330 non-null  float64
 9   SpecialDay               12330 non-null  float64
 10  Month                    12330 non-null  object 
 11  OperatingSystems         12330 non-null  int64  
 12  Browser                  12330 non-null  int64  
 13  Region                   12330 non-null  int64  
 14  TrafficType              12330 non-null  int64  
 15  VisitorType              12330 non-null  object 
 16  Weekend                  12330 non-null  bool   
 17  Revenue                  12330 non-null  bool   
dtypes: bool(2), float64(7), int64(7), object(2)
memory usage: 1.5+ MB
```

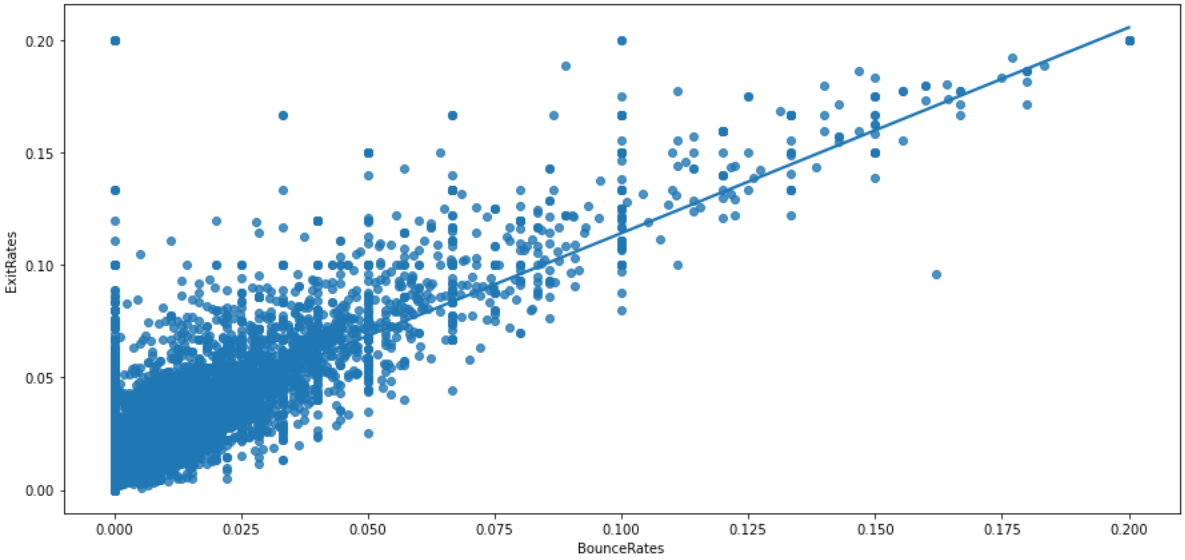
## 2. Feature Engineering and Feature Selection:

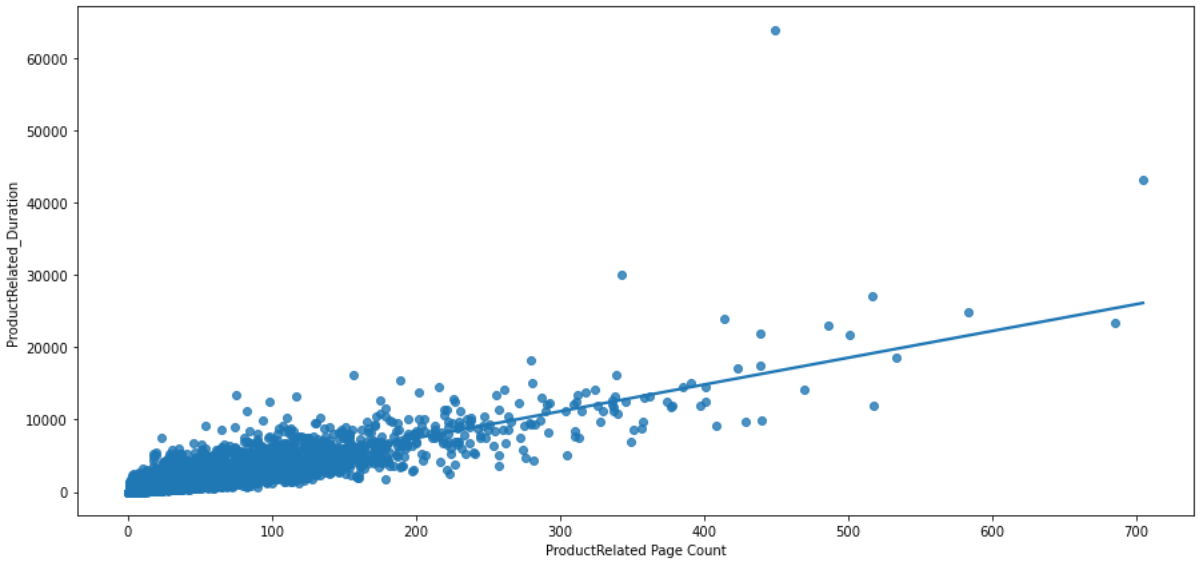
A machine learning project report's Feature Engineering section is a crucial piece since it gives a thorough explanation of the procedure used to develop, alter, and choose the features that will be utilized to train the model. In our project, we start our Feature Engineering process by taking a look at the correlation among each feature through the Correlation Matrix as given below.



From the above Correlation Matrix, we can observe the correlation between the Bounce Rate and Exit Rate is very strong, so Bounce Rate will be the first feature that we will drop. For 6 features (Administrative, Administrative_Duration, Informational, Informational_, ProductRelated, ProductRelated_Duration) as the correlations are relatively higher, we find the average and combine them to form 3 features to reduce the correlations, eventually reducing the number of features.

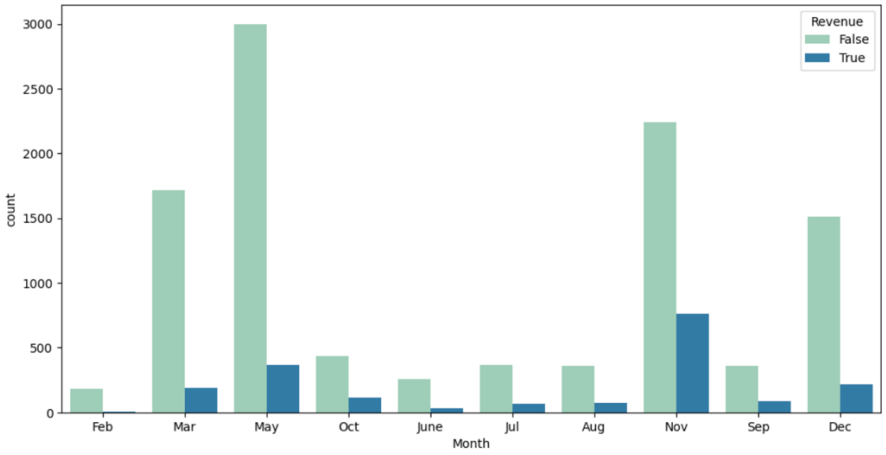
To support our argument of having to drop the highly correlated features, we compared them by visualizing the scatter plot as follows:



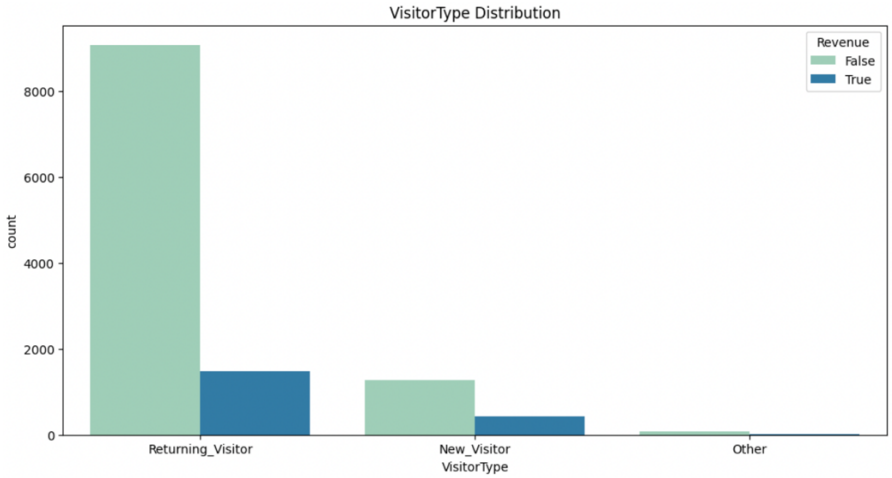



As we can see from the scatterplot above, both plots have a positive association but the strongest relationship occurs in the first scatterplot while the second scatterplot has a weak relationship. The strongest relationship occurs when the slope is 1. Thus, if one variable rises by one, the other variable similarly rises by the same number. The angle of this line is 45 degrees. Hence, we drop the feature Bounce Rate.

Other useful insights include:

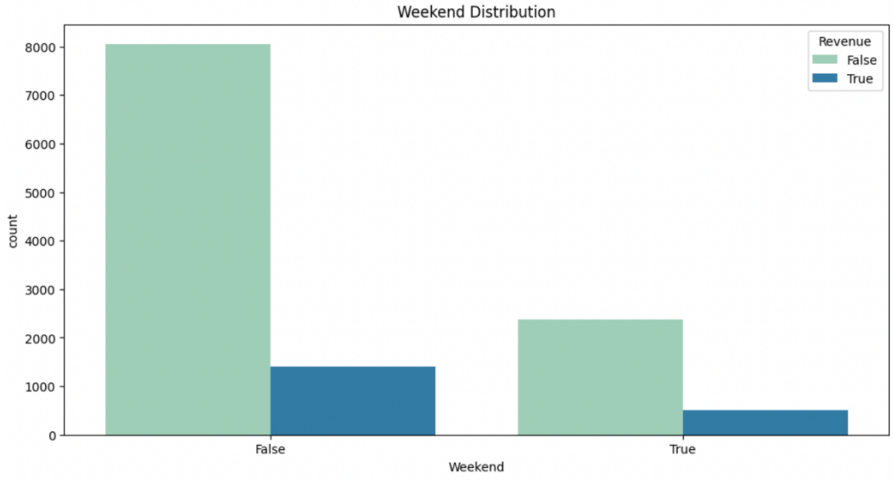



This pair plot shows us the monthly information on whether the consumer purchases the product or not; True being a completed transaction that generates revenue, False being just a visit that wasn't converted into a sale, adding any revenue. From the plot we can observe that the highest revenue generating month is November while the lowest is January and the month with the highest user viewing but not purchasing is May while the lowest the again January. We can see a trend showing us that consumers are less active in the month of January.




- The pair plot above shows us the 3 different types of consumers:
  (i) Returning Visitor
  (ii) New Visitor
  (iii) Other.

  We can clearly see that the returning visitor has generated more revenue compared to the other 2 types of consumers but also has the highest amount of users who did not purchase after viewing.




- The plot above gives us an insight into the activity on Weekends and Weekdays and we can notice that there is a huge difference in activity generated on Weekdays and Weekends but the revenue generated is relatively closer.

Moving on to the next steps in data wrangling, we encoded the three categorical variables "Month," "Visitor Type," and "Weekend" using the pd.get_dummies() function. Categorical variables are no longer included in the dataset.

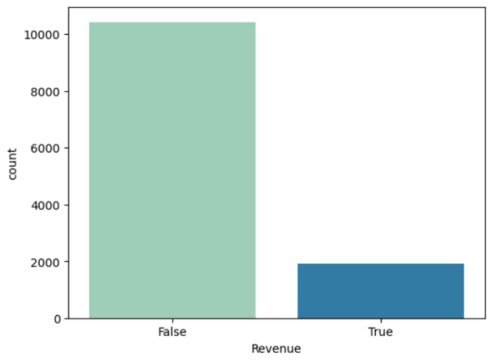
'Administrative_Duration' and 'Administrative', are combined with 'Informational Duration' and Information, 'ProductRelated_duration' and 'ProductRelated', and the original features were removed from the table to eliminate redundancy.

```
[ ] df['avg_administrative_page_dur'] = df['Administrative_Duration'] / (df['Administrative'] + 0.00001)
    df['avg_info_page_dur'] = df['Informational_Duration'] / (df['Informational'] + 0.00001)
    df['avg_product_page_dur'] = df['ProductRelated_Duration'] / (df['ProductRelated'] + 0.00001)

[ ] new_cols = ['avg_administrative_page_dur','avg_info_page_dur','avg_product_page_dur', 'ExitRates', 'PageValues', 'SpecialDay', 'Month',
          'OperatingSystems', 'Browser', 'Region', 'TrafficType', 'VisitorType',
          'Weekend', 'Revenue']
    df2 = df[new_cols]

    df2.head(10)
```

| | avg_administrative_page_dur | avg_info_page_dur | avg_product_page_dur | ExitRates | PageValues | SpecialDay | Month | OperatingSystems | Browser | Region | TrafficType | VisitorType | Weekend | Revenue |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0 | 0.0 | 0.000000 | 0.200000 | 0.0 | 0.0 | Feb | 1 | 1 | 1 | 1 | Returning_Visitor | False | False |
| 1 | 0.0 | 0.0 | 31.999840 | 0.100000 | 0.0 | 0.0 | Feb | 2 | 2 | 1 | 2 | Returning_Visitor | False | False |
| 2 | 0.0 | 0.0 | 0.000000 | 0.200000 | 0.0 | 0.0 | Feb | 4 | 1 | 9 | 3 | Returning_Visitor | False | False |
| 3 | 0.0 | 0.0 | 1.333327 | 0.140000 | 0.0 | 0.0 | Feb | 3 | 2 | 2 | 4 | Returning_Visitor | False | False |
| 4 | 0.0 | 0.0 | 62.749937 | 0.050000 | 0.0 | 0.0 | Feb | 3 | 3 | 1 | 4 | Returning_Visitor | True | False |
| 5 | 0.0 | 0.0 | 8.116662 | 0.024561 | 0.0 | 0.0 | Feb | 2 | 2 | 1 | 3 | Returning_Visitor | False | False |
| 6 | 0.0 | 0.0 | 0.000000 | 0.200000 | 0.0 | 0.4 | Feb | 2 | 4 | 3 | 3 | Returning_Visitor | False | False |
| 7 | 0.0 | 0.0 | 0.000000 | 0.200000 | 0.0 | 0.0 | Feb | 1 | 2 | 1 | 5 | Returning_Visitor | True | False |
| 8 | 0.0 | 0.0 | 18.499908 | 0.100000 | 0.0 | 0.8 | Feb | 2 | 2 | 2 | 3 | Returning_Visitor | False | False |
| 9 | 0.0 | 0.0 | 245.999180 | 0.022222 | 0.0 | 0.4 | Feb | 2 | 4 | 1 | 2 | Returning_Visitor | False | False |

OperatingSystem, Browser, Region, and TrafficType columns were eliminated as they had little to no impact on our goal.

### 3. Balancing Dataset:




Revenue, the target class of our dataset, has two classes: a majority class that reports purchasing intention as False and a minority class that reports shopping intention as True. Only 15% of the observations are attributed to the minority class. This may result in subpar results from our model. We choose to employ the SMOTE method to address this problem.

To make the target variable imbalanced and improve model training, the Synthetic Minority Oversampling technique generates synthetic data for the minority class (No extra information is provided to us by the technique). Before producing dummy data for the minority class, we can under sample the majority class.

For our project, we choose to omit this step and instead oversample members of the minority class. SMOTE was used in 13147 training samples, of which 7284 were deemed false and 5827 were deemed true. As a result, 45% of the training data are classified as false which is a lot more than the 15% we started with.

```
[29] counter = Counter(y_train)
     print(counter)

     Counter({0: 7304, 1: 1327})

[30] smo = SMOTE(random_state = 2, sampling_strategy=0.8)
     X_train_smo, y_train_smo = smo.fit_resample(X_train, y_train)

     counter = Counter(y_train_smo)
     print(counter)

     Counter({0: 7304, 1: 5843})
```

# Results:

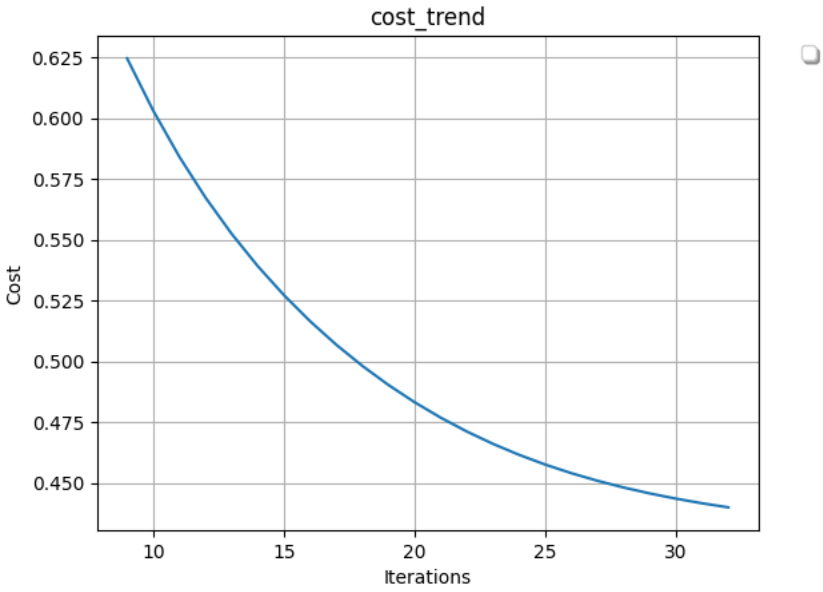
We selected three models to utilize for this classification task and compared the outcomes. Implemented models include Logistic Regression, Naive Bayes, and Neural Networks. 1327 observations in our first sample have a class of 1, while 7304 have a class of 0. As a result, a random model should be more accurate than 85%, hence 85% is what we can say about our baseline accuracy. However, we should be aware that in this work it is more crucial to accurately categorize the target class, which is why we will utilize the precision, recall, and F1-score measures.

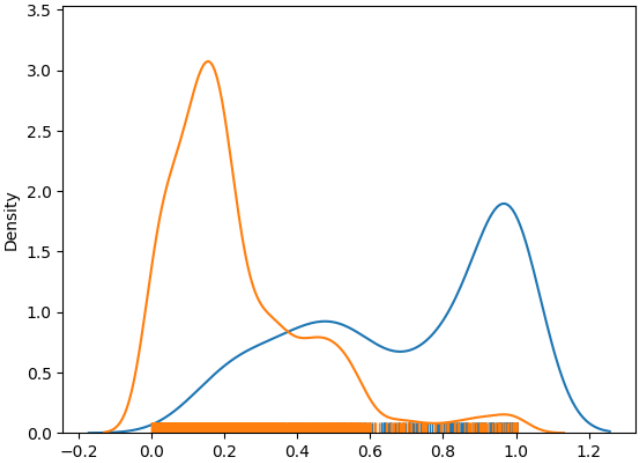
We divided the dataset into sets for training, developing, and testing. Based on the validation results, we will train the models and attempt to optimize the hyper-parameters. Finally, we shall present the test set's outcomes.

## Logistic Regression:

In order to classify the observations using the binary target variable, we first apply logistic regression. We may then classify the observations using various thresholds after the sigmoid function provides us with a probability ranging from 0 to 1. To maximize the F1-score for the validation set, we aim to maximize the learning rate, epsilon, regularization term, and threshold for this model.

With learning rate=0.00005, epsilon=0.0001, regularization term=0.001 and threshold=0.6, we first fit our model. The baseline logistic regression model will be based on this. For the development set, this model's accuracy is 0.89, precision is 0.67 and its F1 score is 0.62. We can observe for this model that the cost has not decreased to its absolute minimum; this may be because our model has a low epsilon.




We experiment with a variety of learning rates, tolerances, regularization terms, and thresholds. The table below displays the outcomes.

| Sl. No. | Learning Rate | Epsilon | Lambda | Threshold | F-1 Score | Precision | Recall | Accuracy |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.00005 | 0.0001 | 0.0001 | 0.6 | 0.62 | 0.67 | 0.59 | 0.89 |
| 2 | 0.0001 | 0.0001 | 0.0001 | 0.5 | 0.59 | 0.54 | 0.66 | 0.86 |
| 3 | 0.01 | 0.0001 | 0.01 | 0.5 | 0.51 | 0.45 | 0.6 | 0.82 |
| 4 | 0.1 | 0.0001 | 0.0001 | 0.5 | 0.57 | 0.63 | 0.52 | 0.88 |
| 5 | 0.0005 | 0.0001 | 0.1 | 0.5 | 0.62 | 0.59 | 0.65 | 0.87 |

After experimenting with a variety of learning rates as mentioned above, we can conclude that the baseline logistic regression model gives us the best Accuracy (0.89), Precision of 0.67 and F-1 Score of 0.62, making it the best Logistic Regression model.

**Naïve Bayes:**

We used Naïve Bayes Classifier to classify our dataset after modeling with Logistic Regression. According to the dataset, the Gaussian Naive Bayes model categorizes a new data point. It is the most basic kind of model where calculations are made based on likelihoods and probabilities. A predictor's probability for a particular class is its likelihood. The prior can be calculated by dividing the number of records in the training dataset by the percentage of observations that have a class label. We trained the priors for the training set and generated predictions for both the training and test sets to see if the model is overfitting or not. Following are the scores we obtained:

| Training: | Score |
|---|---|
| F-1 Score | 0.5714 |
| Precision | 0.6376 |
| Recall | 0.5176 |

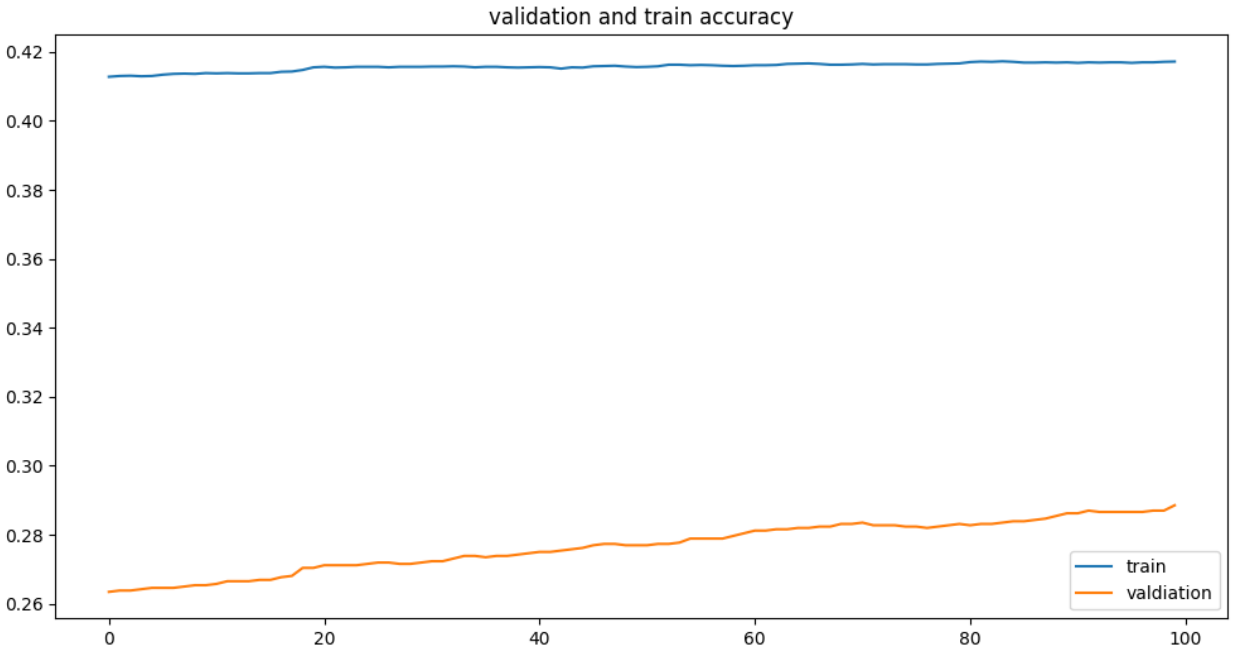
| Development: | Score |
|---|---|
| F-1 Score | 0.5605 |
| Precision | 0.6238 |
| Recall | 0.5089 |

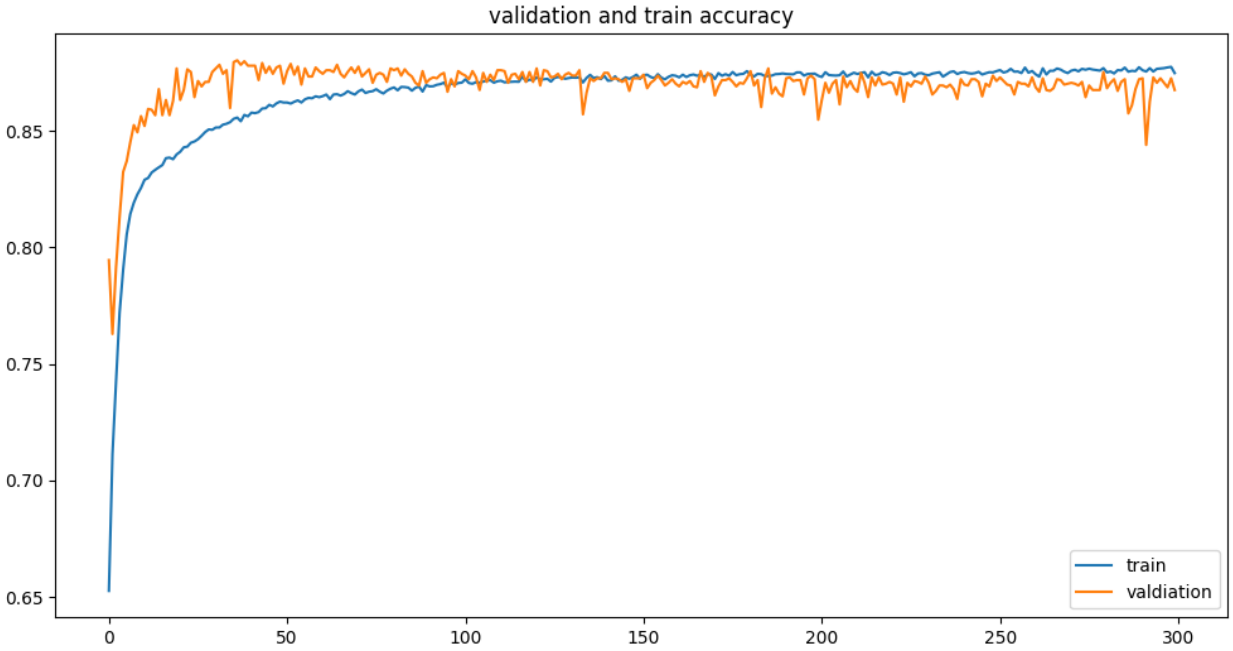
| Testing: | Score |
|---|---|
| F-1 Score | 0.5604 |
| Precision | 0.6291 |
| Recall | 0.5053 |

The accuracy obtained for the test data is 0.8657.

**Neural Network:**

We will go over various neural network models in this section and demonstrate why to apply the bias-variance trade-off to our models. From the plot below, we can say that the data is overfitted. This is because we can see the accuracy of the training data is greater than the validation data. In order to rectify this, we need to perform Bias-Variance Tradeoff by tuning the hyperparameters.




We used different parameters with our models to find the best results and obtained the following are the scores:

| Sl. No. | Epochs | Hidden Layers | Batch Size | Learning Rate | Threshold | Accuracy | F-1 Score | Precision | Recall |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 200 | 2 | 128 | 0.0001 | 0.6 | 0.88 | 0.67 | 0.59 | 0.74 |
| 2 | 200 | 3 | 128 | 0.0005 | 0.6 | 0.88 | 0.67 | 0.6 | 0.76 |
| 3 | 200 | 3 | 128 | 0.008 | 0.6 | 0.86 | 0.63 | 0.55 | 0.74 |
| 4 | 200 | 3 | 64 | 0.006 | 0.6 | 0.86 | 0.61 | 0.53 | 0.72 |
| 5 | 200 | 4 | 128 | 0.0005 | 0.75 | 0.87 | 0.66 | 0.57 | 0.77 |

The model with 200 epochs, 3 hidden layers, and a batch size of 128 was determined to be the best model. To strike a balance between precision and recall, we set the threshold at 0.6. The graphs show the train and validation accuracy and loss.

**Test Dataset Result:**

The results for the performed models are as follows:

| Sl. No. | Models | Accuracy | F1-score | Precision | Recall |
|---|---|---|---|---|---|
| 1 | Logistic Regression | 0.89 | 0.62 | 0.67 | 0.59 |
| 2 | Naïve Bayes | 0.86 | 0.56 | 0.62 | 0.5 |
| 3 | Neural Networks | 0.67 | 0.67 | 0.6 | 0.76 |

We can observe that the most balanced precision and recall are provided by logistic regression and neural networks, while the best F1-score is provided by neural network models. The outcomes are different for naive bayes, though as the F1-score has a value of 0.56, which is lower than the other two.

# Discussion:

The project's discussion of three models for this classification problem concludes. As there are numerous features for classifying consumer intentions, this project's feature engineering component is crucial. We then had to deal with the issue of an unbalanced dataset. The majority of visitors did eventually leave the session without making any purchases. To create synthetic data and balance the training dataset, we employed the SMOTE approach. To prevent extraneous information from leaking into the test sets, we must be careful to just balance the training data and maintain the validation and test data unbalanced. We then used the training set to train the models, and based on the validation results, we selected the top models. The fully connected neural network and logistic regression were the most effective models for this job.

We can raise the threshold to get higher precision if we desire more accuracy in forecasting which clients would really shop. In order for our model to function well for unknown data, it is also critical that we consider the bias-variance trade-off, as mentioned above.