Customer's e-Purchasing Intention Analytics

Group 59

Snehal Gopal Dhone
Krishna Teja Samudrala

dhone.s@northeastern.edu
samudrala.k@northeastern.edu

Percentage of Effort Contributed by Student 1: 50%

Percentage of Effort Contributed by Student 2: 50%

Signature of Student 1: Snehal Dhone

Signature of Student 2: Krishna Samudrala

Submission Date: 04/21/2023

Table of Contents

1. Problem Setting

The business problem of online shoppers' intention for a project in data mining is to understand and predict the behavior of online shoppers to improve the overall shopping experience and increase sales for the online retailer.

- Background of the domain: Online shopping has become increasingly popular in recent years due to the convenience it offers and the wide variety of products available to consumers. However, as the number of online shoppers increases, so does the amount of data generated by their interactions with the retailer's website. This data can come from various sources such as website logs, social media, and customer reviews.
- Applications: Data mining can be used to improve website design and navigation, personalize product recommendations, and predict customer behavior. For example, data mining can be used to analyze customer browsing and purchase history to make personalized product recommendations, or to identify patterns in customer behavior such as popular products or the best time for a promotion.
- Challenges: One of the main challenges of using data mining in online shopping is the large amount of data that is generated. It can be difficult to process and make sense of this data, as it may be unstructured and distributed across multiple platforms. Another challenge is the need to protect customer privacy. Online retailers must ensure that any data they collect is used ethically and that customers' personal information is kept secure.

The business problem is to use data mining techniques to better understand online shoppers' intentions and preferences, to improve the shopping experience and increase sales for the online retailer, while considering the large amount of data generated and the need to protect customer privacy.

2. Problem Definition

The problem of online shoppers' purchasing intention refers to the challenge of predicting whether a visitor to an e-commerce website will make a purchase or not. With the growth of e-commerce, many businesses are heavily dependent on their online sales channels. Therefore, accurately predicting online shoppers' purchasing intention is critical for businesses to optimize their marketing strategies, improve customer engagement, and increase revenue. This problem is complicated by the fact that the vast majority of visitors to e-commerce websites do not make a purchase. This results in an imbalanced dataset, where the number of non-purchasing visitors heavily outweighs the number of purchasing visitors. This imbalance can lead to biased models and inaccurate predictions, making it challenging to develop effective marketing strategies to improve online sales.

3. Data Source

UCI Machine Learning Repository, Center for Machine Learning and Intelligent Systems (https://archive.ics.uci.edu/ml/datasets/Online+Shoppers+Purchasing+Intention+Dataset#)

4. Data Description

Size of the dataset is 12,330 x 18 i.e., it consists of 18 features, with 12,330 rows of information (also called as a session). Out of the 18 features, 10 are numerical and 8 are categorical. The dataset was formed so that each session would belong to a different user in a 1-year period to avoid any tendency to a specific campaign, special day, user profile, or period.

Feature	**Description**
Administrative	This represents how many pages of this kind (administrative) the user accessed
Administrative Duration	This represents the time spent on pages in this category
Informational	This is the number of pages of this type (informational) that the user visited
Informational Duration	This represents the time spent on pages in this category
Product Related	This is the number of product related pages that the user visited
Product Related Duration	This is the amount of time spent in this category of pages
Bounce Rates	The percentage of visitors who enter the website through that page and exit without triggering any additional tasks
Month	Contains the month the pageview occurred, in string form
Exit Rates	The percentage of visitors who arrive at the website via that page and leave without completing any subsequent tasks
Page Values	A page's average value is calculated by averaging it with the value of the target page and/or the successful completion of an online purchase

Special Day	This value shows the proximity of the browsing date to significant days or holidays (for example, Mother's Day or Valentine's Day) in the calendar year
Operating Systems	It represents the operating system that the user was on when viewing the page
Browser	An integer value representing the browser that the user was using to view the page
Region	It represents the region the user is located in.
Traffic Type	It represents what type of traffic the user is categorized into
Visitor Type	A string representing whether a visitor is New Visitor, Returning Visitor, or Other
Weekend	A boolean representing whether the session is on a weekend
Revenue	It represents whether the user completed the purchase

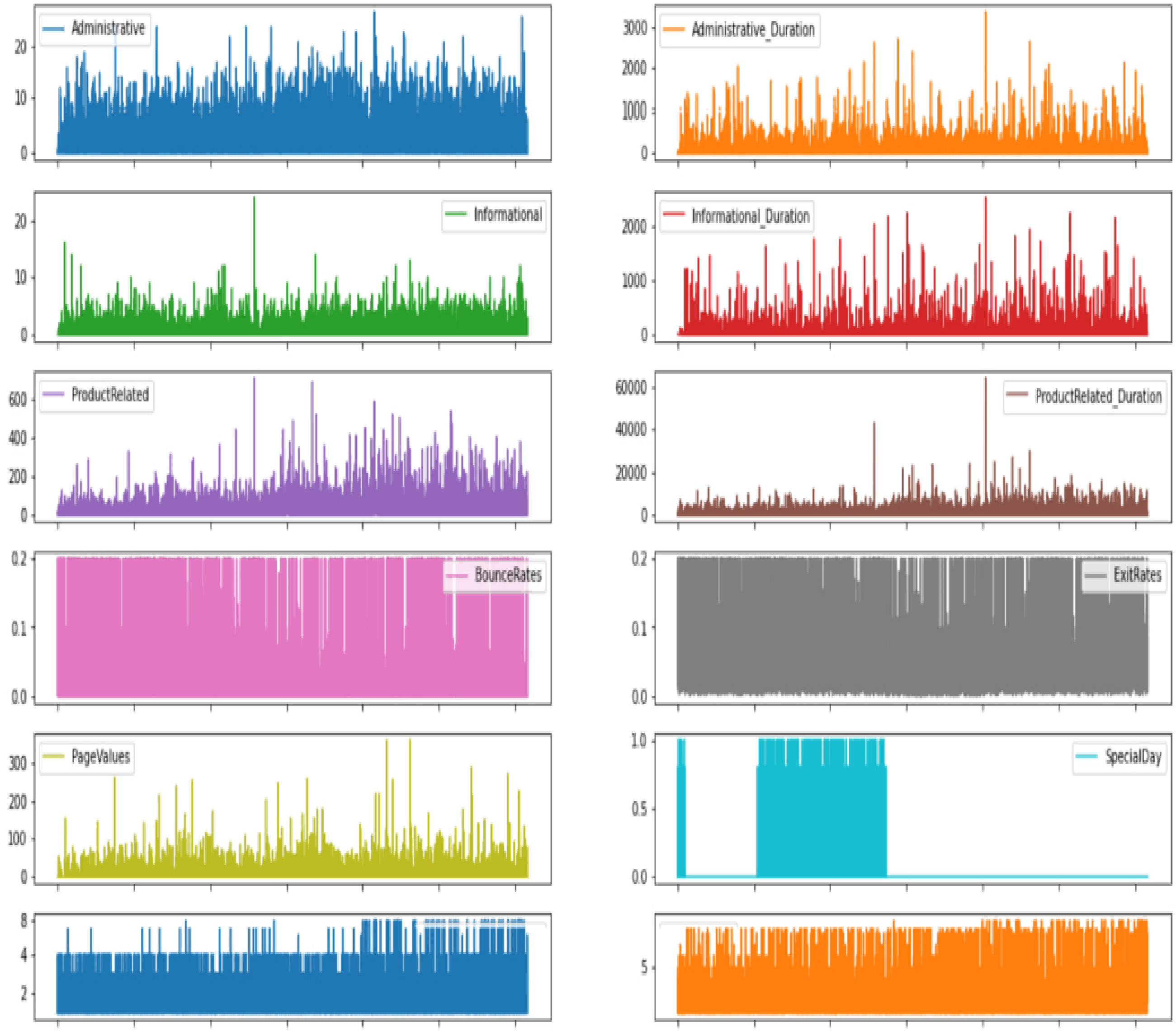
5. Data Collection

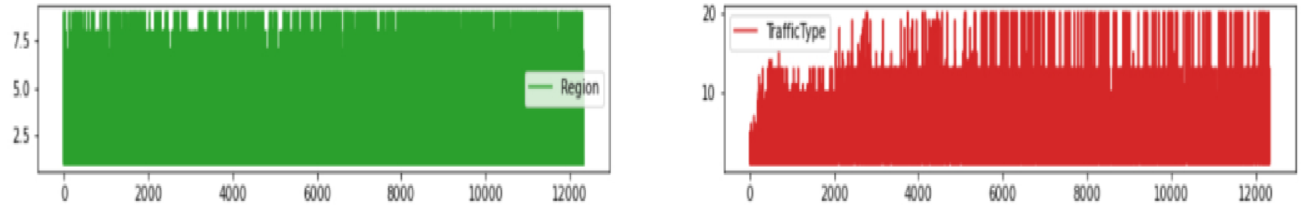
Source of the data is the UCI Machine Learning Repository. The data was collected on 25th Jan, 2023. The data was collected through website logs of an online store. The sample size is 12,330 instances.

6. Data Visualization

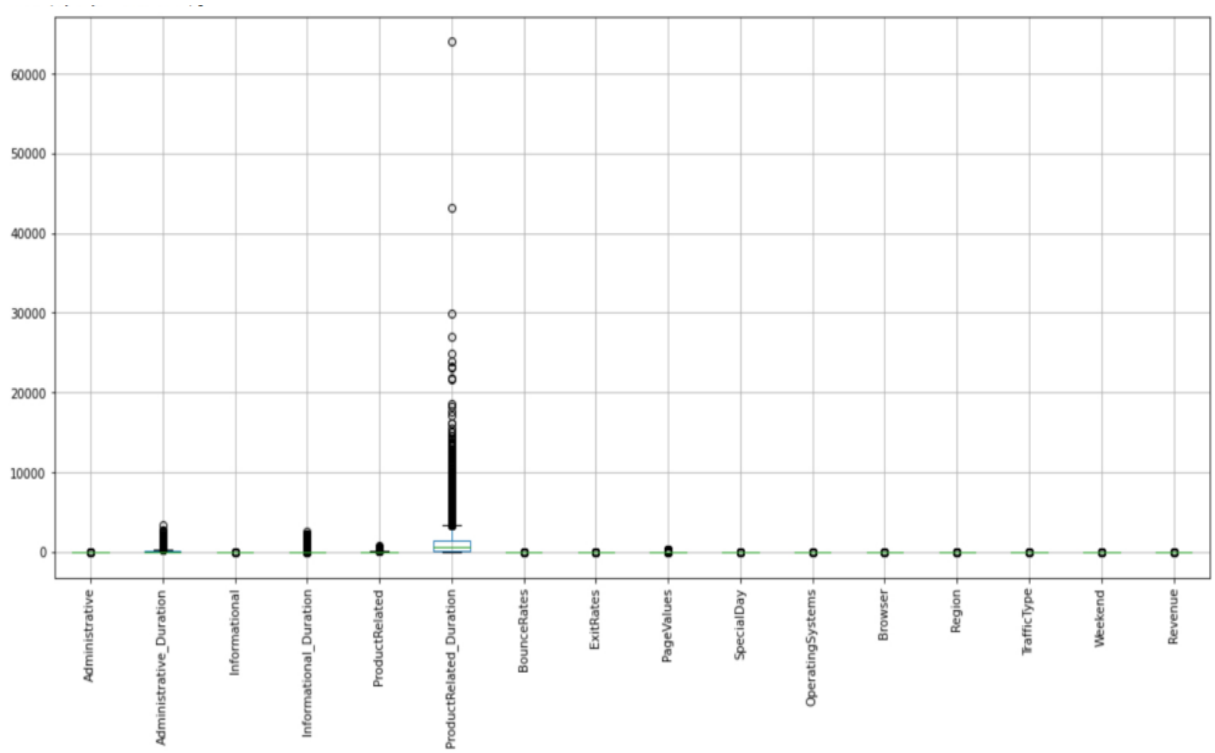
We presented visualizations of the data that provide insights into the relationships between the variables in the dataset as follows:

a. Plot of each variable to show the distribution of the data:


Administrative
20
10
0
Administrative_Duration
3000
2000
1000
0
Informational
20
10
0
Informational_Duration
2000
1000
0
ProductRelated
600
400
200
0
ProductRelated_Duration
60000
40000
20000
0
BounceRates
0.2
0.1
0.0
ExitRates
0.2
0.1
0.0
PageValues
300
200
100
0
SpecialDay
1.0
0.5
0.0
8
4
2
5

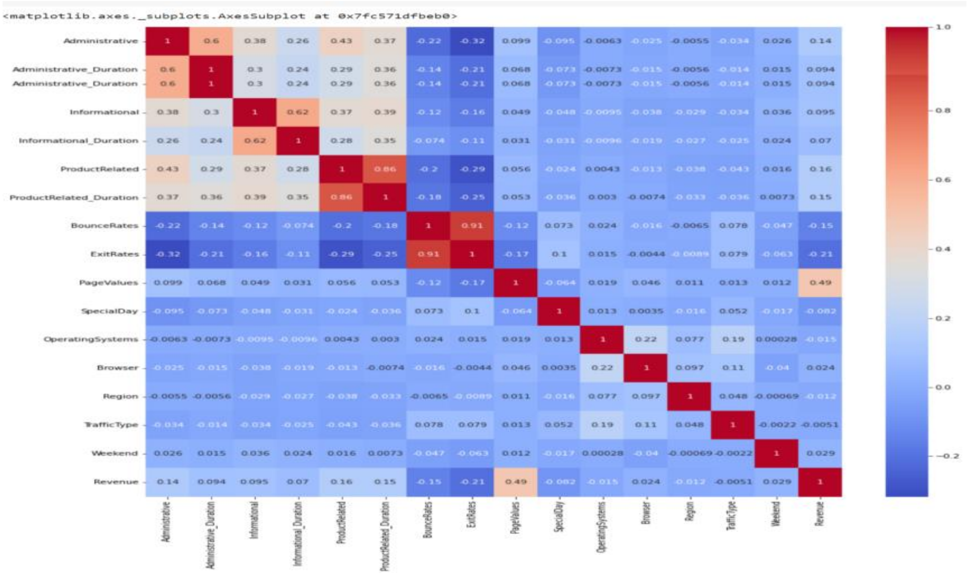

Region
7.5
5.0
2.5
TrafficType
20
10
0
2000
4000
6000
8000
10000
12000

b. Boxplot of each variable to find any outliers in the dataset:



From the above boxplot, it can be concluded that the range of ProductRelated_Duration variable varies to a greater extent. Hence, it needs to be dealt with in the dataset to avoid errors while training the model.

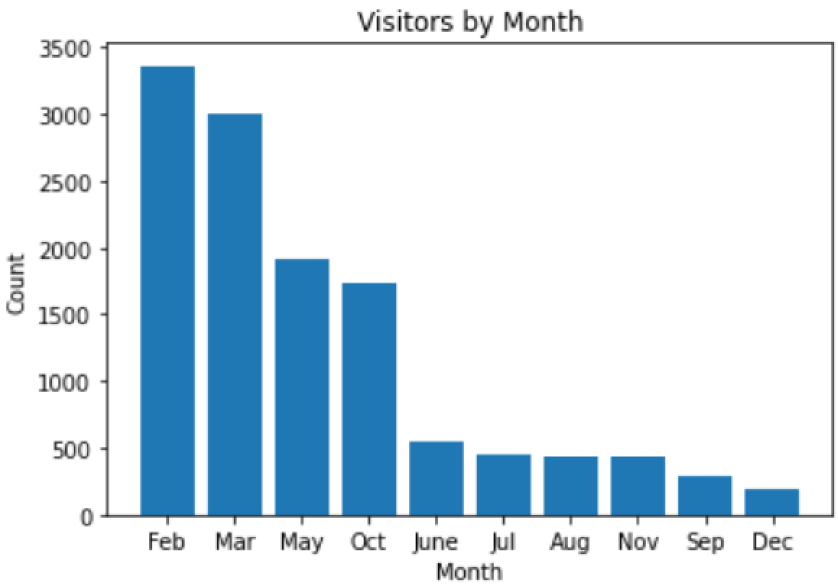
c. Heatmap or correlation matrix to show the correlations between pairs of variables: This plot shows how each feature in the dataset is correlated with every other feature. It can help identify which features are most important for predicting the target variable.



```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc571dfbeb0>
```

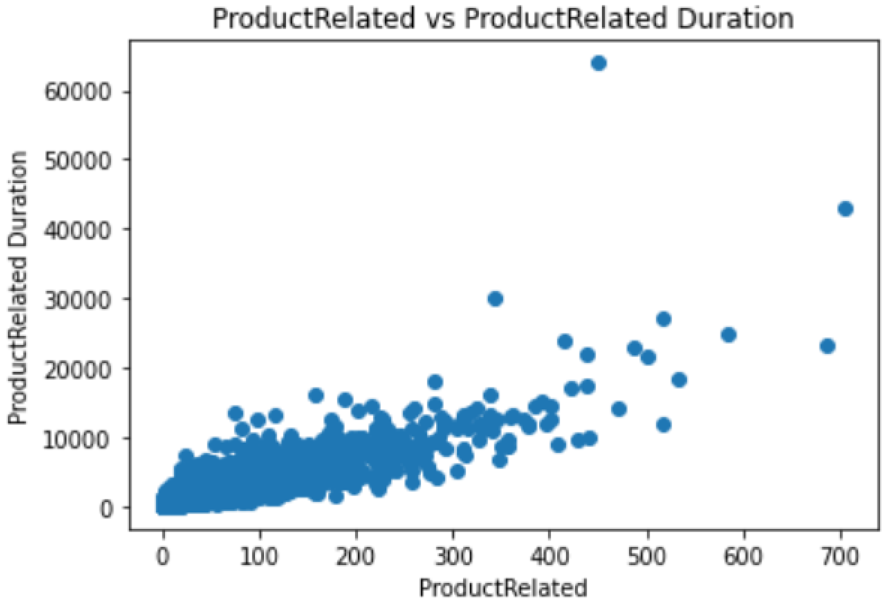
	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	OperatingSystems	Browser	Region	TrafficType	Weekend	Revenue
Administrative	1	0.6	0.38	0.26	0.43	0.37	-0.22	-0.32	0.099	-0.095	-0.0063	-0.025	-0.0055	-0.034	0.026	0.14
Administrative_Duration	0.6	1	0.3	0.24	0.29	0.36	-0.14	-0.21	0.068	-0.073	-0.0073	-0.015	-0.0056	-0.014	0.015	0.094
Administrative_Duration	0.6	1	0.3	0.24	0.29	0.36	-0.14	-0.21	0.068	-0.073	-0.0073	-0.015	-0.0056	-0.014	0.015	0.094
Informational	0.38	0.3	1	0.62	0.37	0.39	-0.12	-0.16	0.049	-0.048	-0.0095	-0.038	-0.029	-0.034	0.036	0.095
Informational_Duration	0.26	0.24	0.62	1	0.28	0.35	-0.074	-0.11	0.031	-0.031	-0.0096	-0.019	-0.027	-0.025	0.024	0.07
ProductRelated	0.43	0.29	0.37	0.28	1	0.86	-0.2	-0.29	0.056	-0.024	0.0043	-0.013	-0.038	-0.043	0.016	0.16
ProductRelated_Duration	0.37	0.36	0.39	0.35	0.86	1	-0.18	-0.25	0.053	-0.036	0.003	-0.0074	-0.033	-0.036	0.0073	0.15
BounceRates	-0.22	-0.14	-0.12	-0.074	-0.2	-0.18	1	0.91	-0.12	0.073	0.024	-0.016	-0.0065	0.078	-0.047	-0.15
ExitRates	-0.32	-0.21	-0.16	-0.11	-0.29	-0.25	0.91	1	-0.17	0.1	0.015	-0.0044	-0.0089	0.079	-0.063	-0.21
PageValues	0.099	0.068	0.049	0.031	0.056	0.053	-0.12	-0.17	1	-0.064	0.019	0.046	0.011	0.013	0.012	0.49
SpecialDay	-0.095	-0.073	-0.048	-0.031	-0.024	-0.036	0.073	0.1	-0.064	1	0.013	0.0035	-0.016	0.052	-0.017	-0.082
OperatingSystems	-0.0063	-0.0073	-0.0095	-0.0096	0.0043	0.003	0.024	0.015	0.019	0.013	1	0.22	0.077	0.19	0.00028	-0.015
Browser	-0.025	-0.015	-0.038	-0.019	-0.013	-0.0074	-0.016	-0.0044	0.046	0.0035	0.22	1	0.097	0.11	-0.04	0.024
Region	-0.0055	-0.0056	-0.029	-0.027	-0.038	-0.033	-0.0065	-0.0089	0.011	-0.016	0.077	0.097	1	0.048	-0.00069	-0.012
TrafficType	-0.034	-0.014	-0.034	-0.025	-0.043	-0.036	0.078	0.079	0.013	0.052	0.19	0.11	0.048	1	-0.0022	-0.0051
Weekend	0.026	0.015	0.036	0.024	0.016	0.0073	-0.047	-0.063	0.012	-0.017	0.00028	-0.04	-0.00069	-0.0022	1	0.029
Revenue	0.14	0.094	0.095	0.07	0.16	0.15	-0.15	-0.21	0.49	-0.082	-0.015	0.024	-0.012	-0.0051	0.029	1

d. Bar plot of visitors by month: This plot shows how the number of visitors fluctuates throughout the year. It can also highlight which months generate the most revenue.



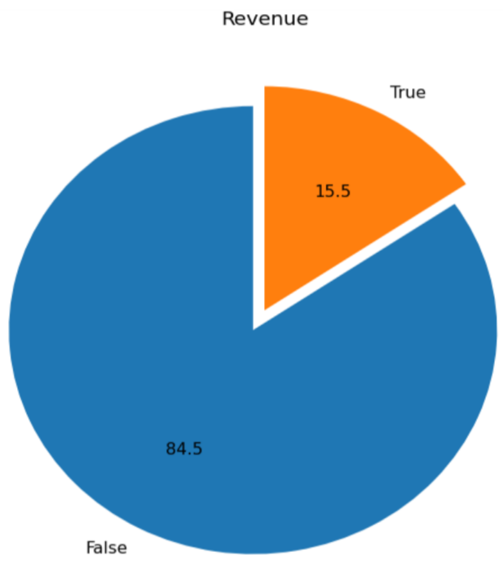
From the above bar plot, it can be concluded that most of the online shoppers are active during the month of February, followed by March, May and so on.

e. Scatterplot of a pair of variables to show the relationships between them: This plot shows if there is a correlation between the amount of time spent on any category of pages (ProductRelated Duration) and the number of product related pages that the user visited (ProductRelated).



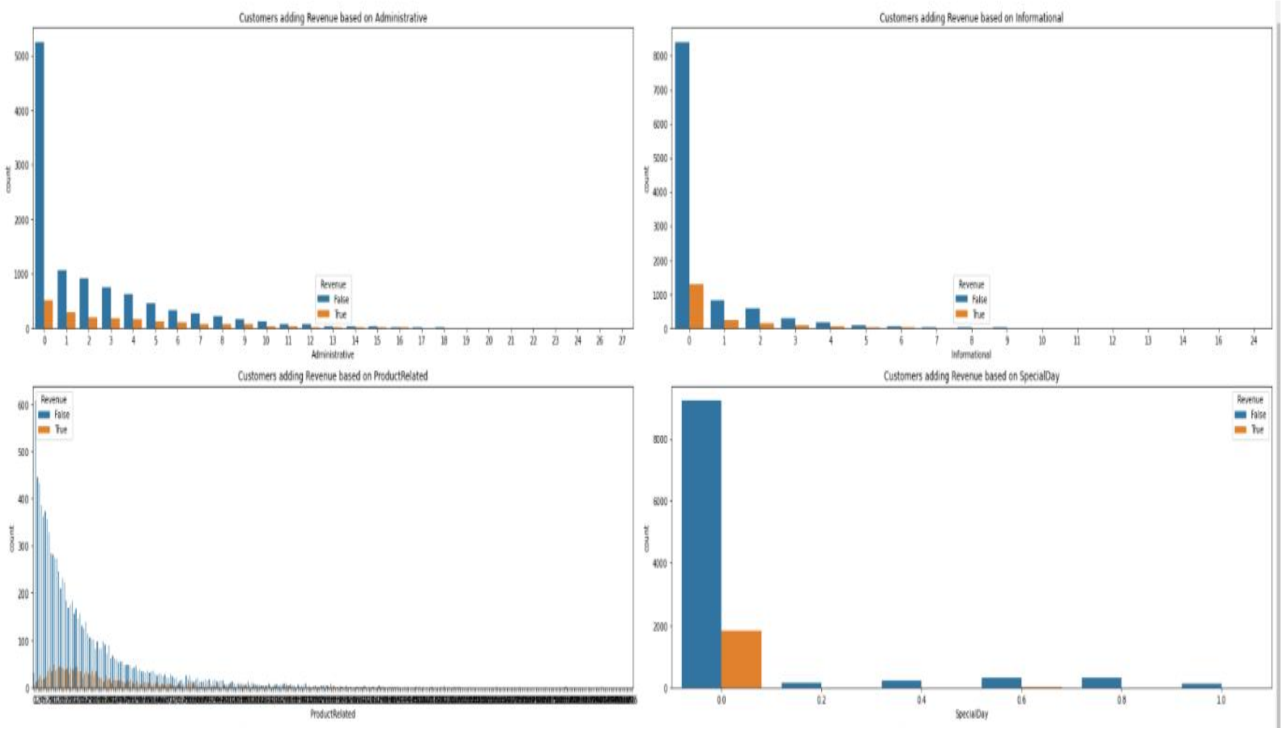
From the above scatter plot, it can be concluded that they are highly correlated.

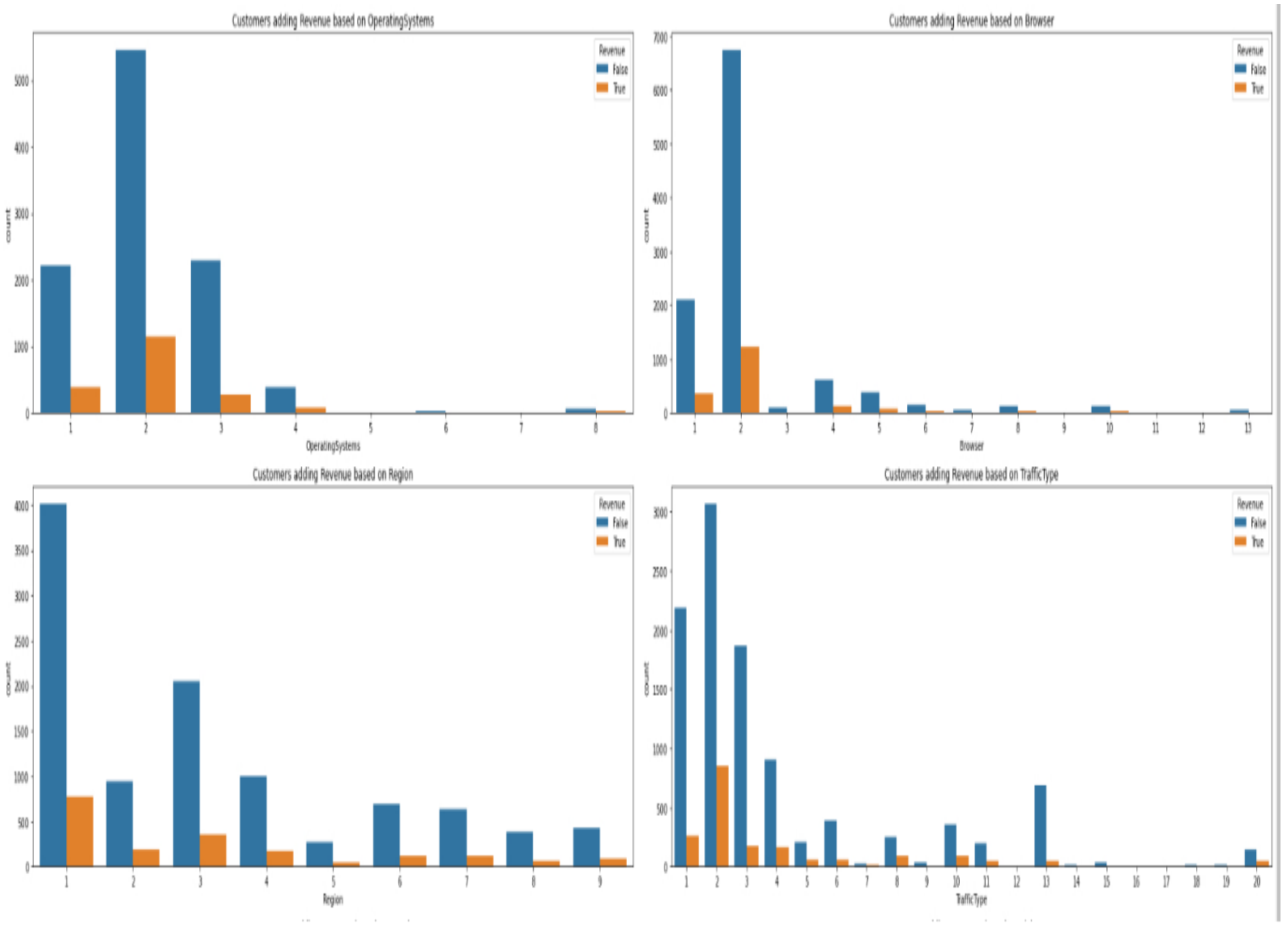
f. Pie chart of the target variable: This plot shows the distribution of the target variable i.e. Revenue generated.



From the above pie chart, it can be concluded that this variable highly imbalanced, which needs to be taken care of while training the model to avoid biasness.

g. Additional visualizations: Customers adding revenue based on Administrative, Informational, ProductRelated, Special Day, Operating Systems, Browser, Region, and Traffic Type.

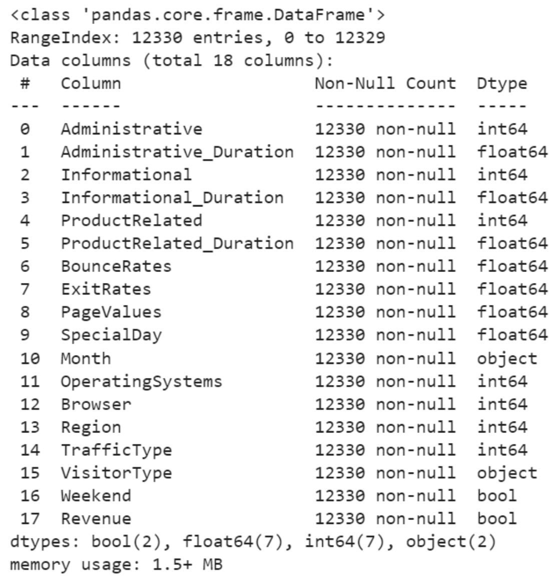




7. <u>Data Exploration</u>

We explored the data in more depth to provide insights into its properties as follows:

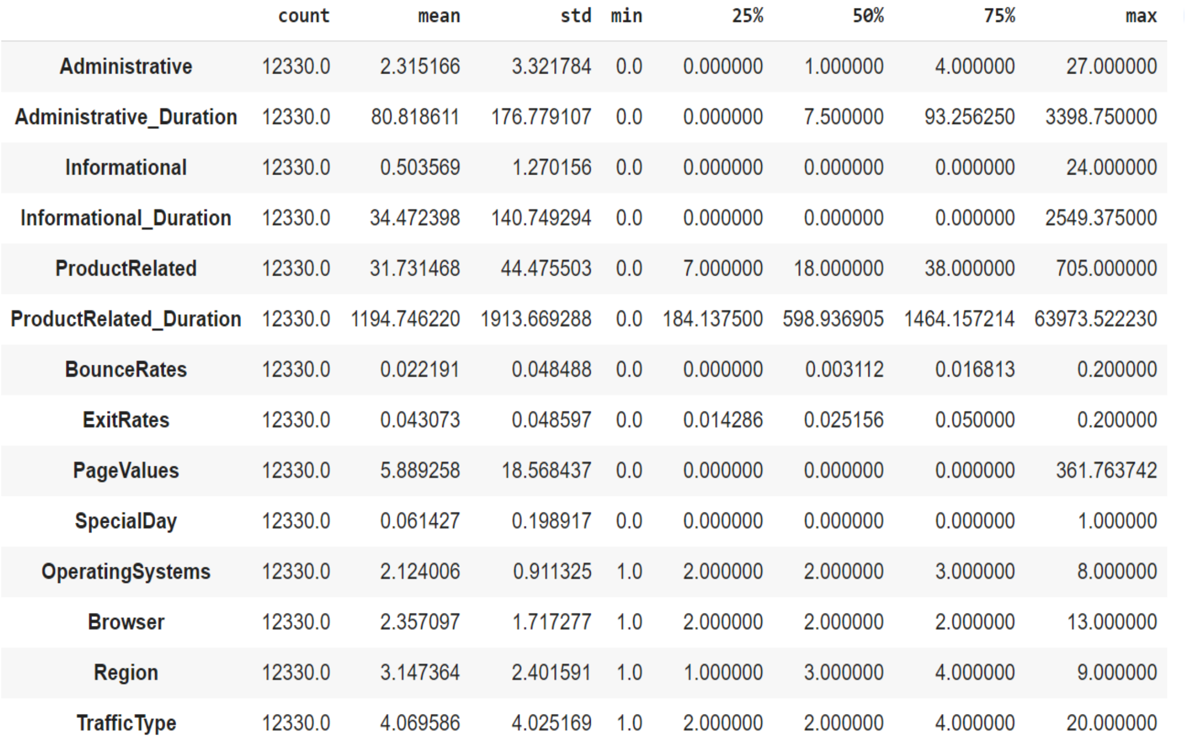
a. Description of variables:



```
<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
```

The above description includes variable names, data types, and if any missing values are present.

b. A summary of the statistics of the variables:



	count	mean	std	min	25%	50%	75%	max
Administrative	12330.0	2.315166	3.321784	0.0	0.000000	1.000000	4.000000	27.000000
Administrative_Duration	12330.0	80.818611	176.779107	0.0	0.000000	7.500000	93.256250	3398.750000
Informational	12330.0	0.503569	1.270156	0.0	0.000000	0.000000	0.000000	24.000000
Informational_Duration	12330.0	34.472398	140.749294	0.0	0.000000	0.000000	0.000000	2549.375000
ProductRelated	12330.0	31.731468	44.475503	0.0	7.000000	18.000000	38.000000	705.000000
ProductRelated_Duration	12330.0	1194.746220	1913.669288	0.0	184.137500	598.936905	1464.157214	63973.522230
BounceRates	12330.0	0.022191	0.048488	0.0	0.000000	0.003112	0.016813	0.200000
ExitRates	12330.0	0.043073	0.048597	0.0	0.014286	0.025156	0.050000	0.200000
PageValues	12330.0	5.889258	18.568437	0.0	0.000000	0.000000	0.000000	361.763742
SpecialDay	12330.0	0.061427	0.198917	0.0	0.000000	0.000000	0.000000	1.000000
OperatingSystems	12330.0	2.124006	0.911325	1.0	2.000000	2.000000	3.000000	8.000000
Browser	12330.0	2.357097	1.717277	1.0	2.000000	2.000000	2.000000	13.000000
Region	12330.0	3.147364	2.401591	1.0	1.000000	3.000000	4.000000	9.000000
TrafficType	12330.0	4.069586	4.025169	1.0	2.000000	2.000000	4.000000	20.000000

The above statistical summary includes count, mean, standard deviation, minimum value, maximum value, and 1st, 2nd and 3rd quartiles. The outliers, correlation and more has been explored under data visualization part.

8. Data Processing

a. **Preprocessing:** In this step, the dataset is preprocessed by dropping unnecessary columns, handling missing values, and converting categorical variables to numerical. In this case, we drop the columns Administrative, Informational, ProductRelated, Month, OperatingSystems, Browser, Region, TrafficType, and VisitorType as they are not needed for analysis. Finally, the Weekend and Revenue columns are converted from categorical variables to numerical variables.

b. **Splitting the dataset:** Here, the dataset is splitted into training and testing datasets using the train_test_split() function from the sklearn.model_selection library. The drop() function is used to drop the Revenue column from the features dataset X and assign it to the target variable y. A test size of 0.2 is used, which means that 20% of the data will be used for testing, and the rest will be used for training.

c. **Scaling the features:** In this step, the features in the dataset are scaled using the StandardScaler() function from the sklearn.preprocessing library. Scaling the features ensures that all the features have the same scale, which can improve the performance of some machine learning algorithms.

d. **Encode categorical variables:** Machine learning models cannot work with categorical data in their raw form, so it is needed to encode them into numerical values. There are different methods for encoding categorical variables, and the choice of method depends on the nature of the data and the machine learning algorithm being used. In this dataset, the categorical variables are: "Month", "OperatingSystems", "Browser", "Region", "TrafficType", "VisitorType", and "Weekend". One-hot encoding is used to convert these variables into numerical values. LabelEncoder assigns a unique integer to each category in a categorical variable. This can create an ordering of categories that might not make sense in certain situations. For example, in the Month column, January is encoded as 5, while December is encoded as 2. This can be misleading, as there is no inherent ordering of months. Hence, in this case One-hot encoding is more appropriate to use.

e. **Split the data into training, validation and testing sets:** To evaluate the performance of our machine learning model, it is needed to split the dataset into training, validation and testing sets. The training set is used to train the model, validation set is used evaluate the performance of the model during training, and the testing set is used to evaluate its performance on unseen data. The ‘train_test_split’ function from scikit-learn to split the data into training, validation and testing sets. The function takes four parameters: X, y, test_size, and random_state. The X parameter contains the final dataset with all features encoded and concatenated, y contains the target variable, ‘test_size’ specifies the proportion of the data to use for testing (in this case 15%, validation 15%), and random_state is used to ensure that the split is reproducible.

f. **Balance the dataset:** SMOTE (Synthetic Minority Over-sampling Technique) is a widely used method for handling imbalanced datasets. In the context of online shoppers' purchasing intention, imbalanced data is often encountered since the majority of visitors to e-commerce websites do not make a purchase. This can lead to biased models and inaccurate predictions. To address this issue, we implemented the SMOTE algorithm to balance the dataset. SMOTE works by creating synthetic samples of the minority class by interpolating between existing minority class instances. This technique helps to increase the number of minority class samples and reduce the imbalance in the dataset, which in turn helps to improve the performance of our machine learning models. By using SMOTE, we were able to balance the dataset, which helped to improve the performance of our models and reduce the bias in our predictions.

9. Exploration of Data Mining models

To explore the candidate data mining models, first the problem and the goal is defined. Based on the provided dataset of online shopper's purchasing intention, the goal is to predict whether the shopper will make a purchase (Revenue = True) or not (Revenue = False). Next, we need to consider the characteristics of the dataset. It contains both numerical and categorical variables, and the target variable is binary. Therefore, we can consider using the following candidate models:

a. **Logistic Regression:** This is a popular classification algorithm that can be used for binary classification problems like this one. It is simple to understand and implement, and it works well when there is a linear relationship between the input variables and the output.

Advantages:

- It is simple and easy to implement.
- It can handle both binary and multiclass classification problems.
- It outputs probabilities that can be interpreted as the likelihood of a sample belonging to a particular class.
- It can be used as a baseline model to compare with other more complex models.

Disadvantages:

- It assumes a linear relationship between the independent variables and the log-odds of the dependent variable.
- It can be sensitive to outliers and multicollinearity.
- It does not perform well when the data is not linearly separable.

b. **Decision Tree:** A decision tree is a versatile and easy-to-understand algorithm that can be used for both classification and regression problems. It is particularly useful when there are non-linear relationships between the input variables and the output. Decision trees can also handle categorical variables without the need for encoding.

Advantages:

- It is easy to understand and interpret, as the decision rules are visible in the form of a tree.

- It can handle both categorical and numerical data, and can be used for both classification and regression problems.
- It can identify complex relationships between the features.
- It can handle missing values and outliers.

Disadvantages:

- It is prone to overfitting, especially when the tree is deep.
- It can be sensitive to small changes in the data, which can lead to different trees being generated.
- It can create biased trees if there is an imbalance in the class distribution.

c. **Random Forest**: Random Forest is an ensemble algorithm that combines multiple decision trees to improve the accuracy and robustness of the model. It is particularly useful when dealing with high-dimensional data, and it can handle both categorical and numerical variables.

Advantages:

- It can handle both categorical and numerical data, and can be used for both classification and regression problems.
- It can identify complex relationships between the features.
- It can handle missing values and outliers.
- It reduces the risk of overfitting by aggregating multiple decision trees.

Disadvantages:

- It can be computationally expensive, especially for large datasets and deep trees.
- It can be difficult to interpret the results, as the decision rules are spread across multiple trees.
- It can create biased trees if there is an imbalance in the class distribution.

d. **Naive Bayes**: Naive Bayes is a probabilistic algorithm that makes assumptions about the independence of features. It is known for its simplicity, fast training, and high accuracy in text classification tasks. Naive Bayes can handle a large number of features and is suitable for datasets with a high dimensionality. However, it may not perform well on datasets with highly correlated features.

Advantages:

- It is a simple and easy-to-understand classification algorithm that can be easily implemented.
- It performs well with a high-dimensional dataset and requires a small amount of training data to estimate the necessary parameters.
- It is computationally efficient, making it a good choice for real-time predictions and large datasets.
- It handles both categorical and continuous data well, and it is less prone to overfitting than other models.

Disadvantages:

- It makes a strong assumption of independence among features, which is not always true in real-world scenarios.
- It can result in poor accuracy if the independence assumption is not met.
- It has a bias towards categorical inputs, which can lead to poor performance with continuous data.
- It does not take into account the interaction between features, which can be important for some applications.

e. **XGBoost**: XGBoost (Extreme Gradient Boosting) is a popular open-source gradient boosting library used for both regression and classification tasks. It is a tree-based model that sequentially adds new trees to improve the prediction accuracy of the model. XGBoost has gained immense popularity in recent years and has been used in many real-world applications due to its high accuracy and fast computation speed.

Advantages:

- It is highly accurate due to its ability to use a combination of weak models to form a strong model.
- It can handle missing data by automatically learning the missing values from the data.
- It is highly optimized for speed and can handle large datasets with millions of rows and columns.
- It provides built-in regularization to prevent overfitting, which is a common problem in machine learning.

- It provides a feature importance score that helps in identifying the most important features in the dataset.

Disadvantages:

- It can easily overfit the data if the hyperparameters are not set correctly.
- It is a complex model and requires a good understanding of the underlying algorithm to use it effectively.
- It requires high computational resources and may not be suitable for low-end machines.

To select the best final model or models, cross-validation and performance metrics are used, such as accuracy, precision, recall, and F1 score. We will also consider the interpretability of the model, the computational complexity, and the ease of implementation. Based on the characteristics of the dataset and the candidate models, we will start with logistic regression and decision tree models and then try more complex models such as random forest, Naïve Bayes and Neural Network. After training and evaluating the models using cross-validation and performance metrics, we will select the best final model based on the overall performance and the trade-offs between interpretability and complexity. We will also consider using a combination of models to improve the performance and interpretability.

10. Model performance evaluation and Interpretation

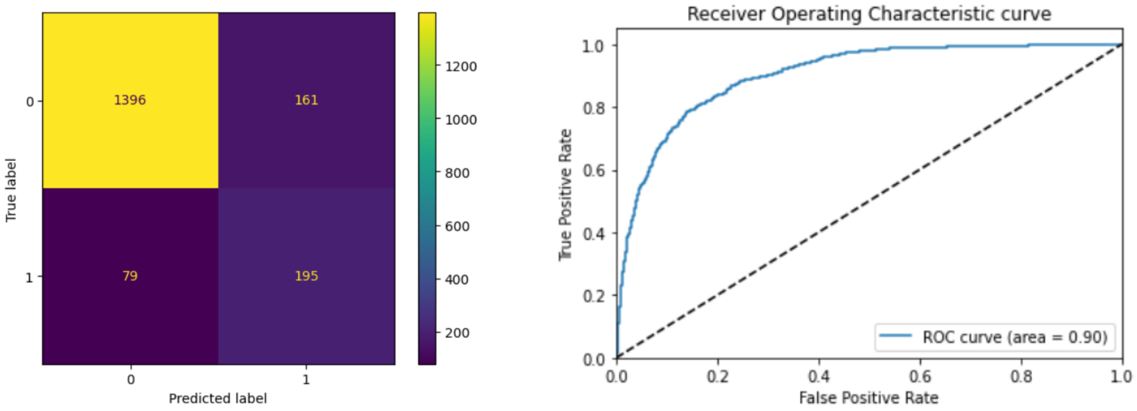
It is a crucial step in the machine learning workflow. It involves evaluating the performance of trained model on a test set and interpreting the results to make decisions or take actions. Based on the evaluation metrics of Accuracy, F1 Score, MCC, Sensitivity, Specificity, and ROC AUC Score, we selected the best model for our classification problem as ‘Random Forest’.

ML Model	Logistic Regression	Decision Tree	Random Forest	Naïve Bayes	XG Boost
Accuracy	0.869	0.869	0.901	0.746	0.887
F1 Score	0.619	0.56	0.667	0.455	0.607
MCC	0.548	0.483	0.608	0.35	0.544
Sensitivity	0.722	0.573	0.69	0.708	0.718
Specificity	0.891	0.923	0.927	0.753	0.917
ROC AUC Score	0.902	0.743	0.923	0.821	0.921

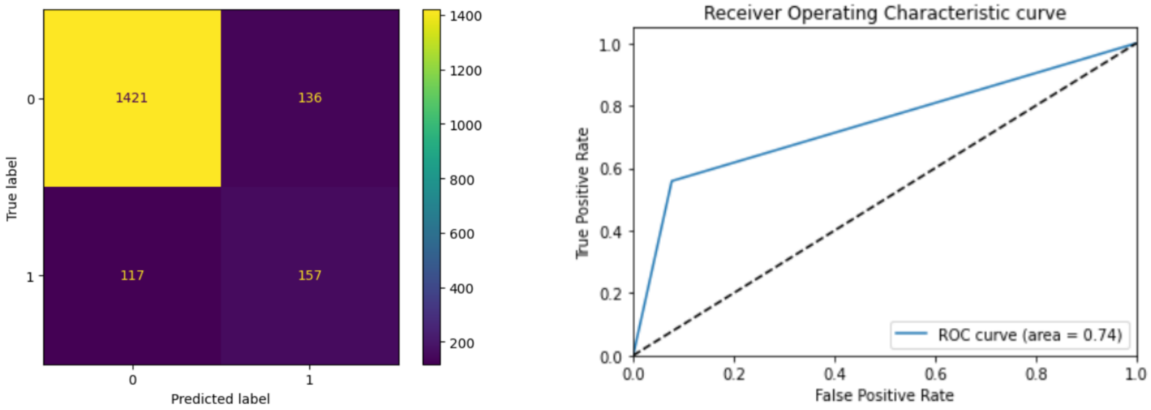
- Accuracy measures the proportion of correct predictions made by the model. However, it can be misleading for our imbalanced dataset, as even a model that always predicts the majority class will have a high accuracy. Thus, it is not the most reliable metric for evaluating highly imbalanced classification models. Hence, we are not solely relying on this metric to select the best-performing model.
- F1 Score takes into account both precision and recall, providing a more balanced measure of the model's performance on both the minority and majority classes.
- Matthews Correlation Coefficient (MCC) is a correlation coefficient between the observed and predicted binary classifications. MCC ranges between -1 and +1, where +1 represents a perfect prediction, 0 represents a random prediction, and -1 represents an inverse prediction.
- Sensitivity measures the proportion of true positives that are correctly identified by the model. In the context of imbalanced classification, it is important to have a high sensitivity to capture as many instances of the minority class as possible.
- Specificity measures the proportion of true negatives that are correctly identified by the model. It is important to have a high specificity to correctly identify instances of the majority class.
- ROC AUC Score measures the model's ability to distinguish between the minority and majority classes. It is a useful metric for evaluating the overall performance of the model across different thresholds.

When evaluating the performance of the models, we considered a combination of these metrics, rather than relying on a single metric. A good model i.e., the Random Forest model has high F1 score, MCC, sensitivity, specificity, and ROC AUC score, indicating that it performs well on both the minority and majority classes, while also being able to distinguish between them.

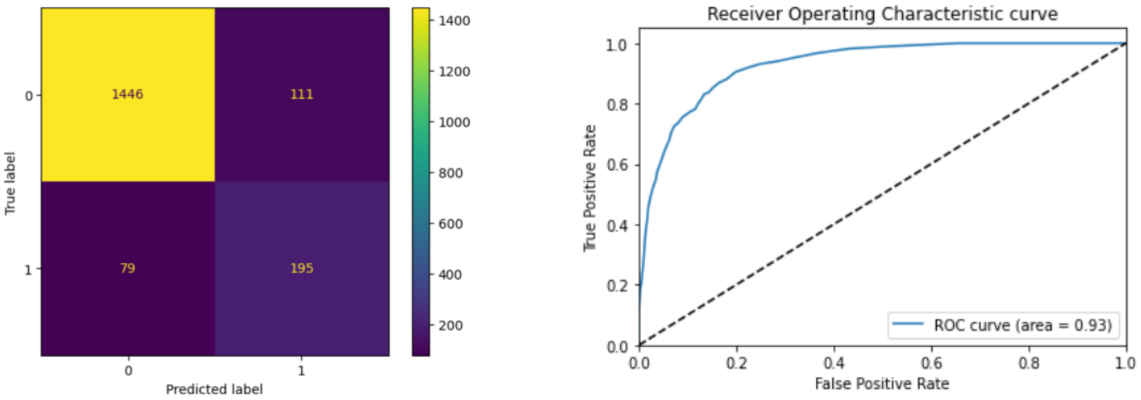
The ROC curve and the confusion matrix are complementary evaluation metrics, providing different types of information about the performance of a classification model. The confusion matrix gives a more detailed breakdown of the types of errors made by the model, while the ROC curve provides a visual representation of the trade-off between sensitivity and specificity for different threshold values. The following figures shows confusion matrix and ROC curve for each of the model:



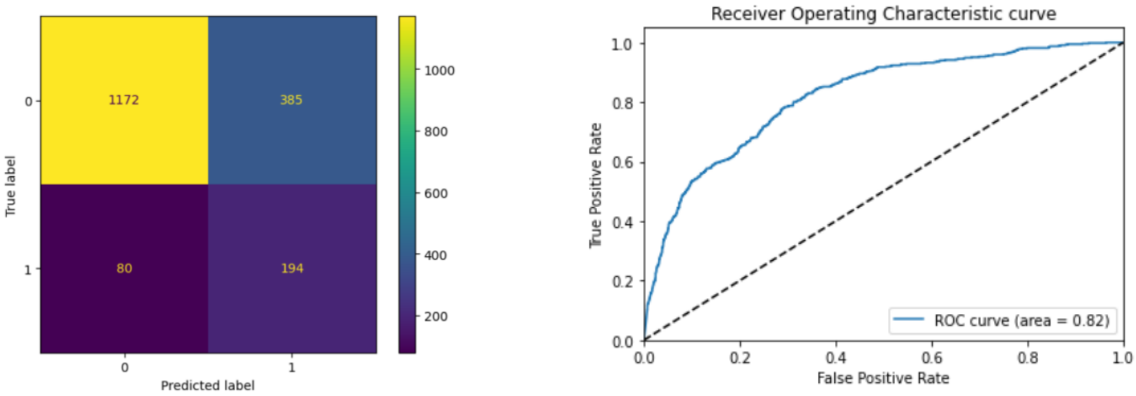
a. Figure shows confusion matrix and ROC curve for Logistic Regression model



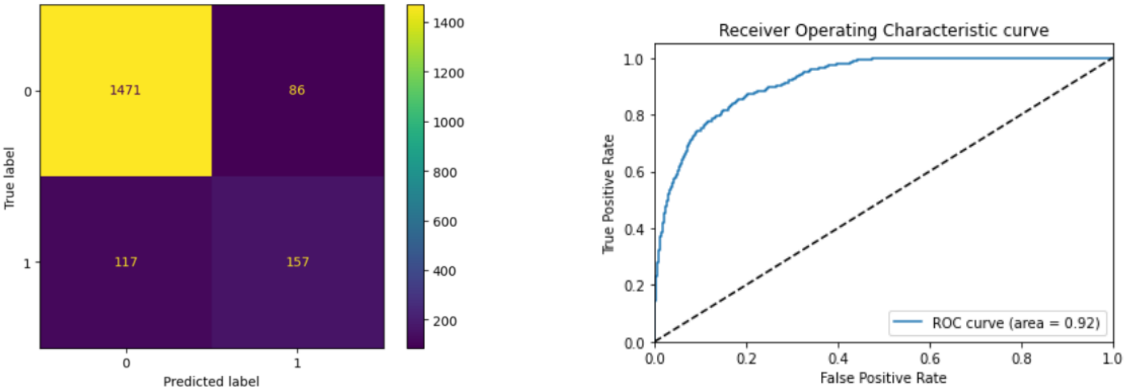
b. Figure shows confusion matrix and ROC curve for Decision Tree model



c. Figure shows confusion matrix and ROC curve for Random Forest model



d. Figure shows confusion matrix and ROC curve for Naïve Bayes model



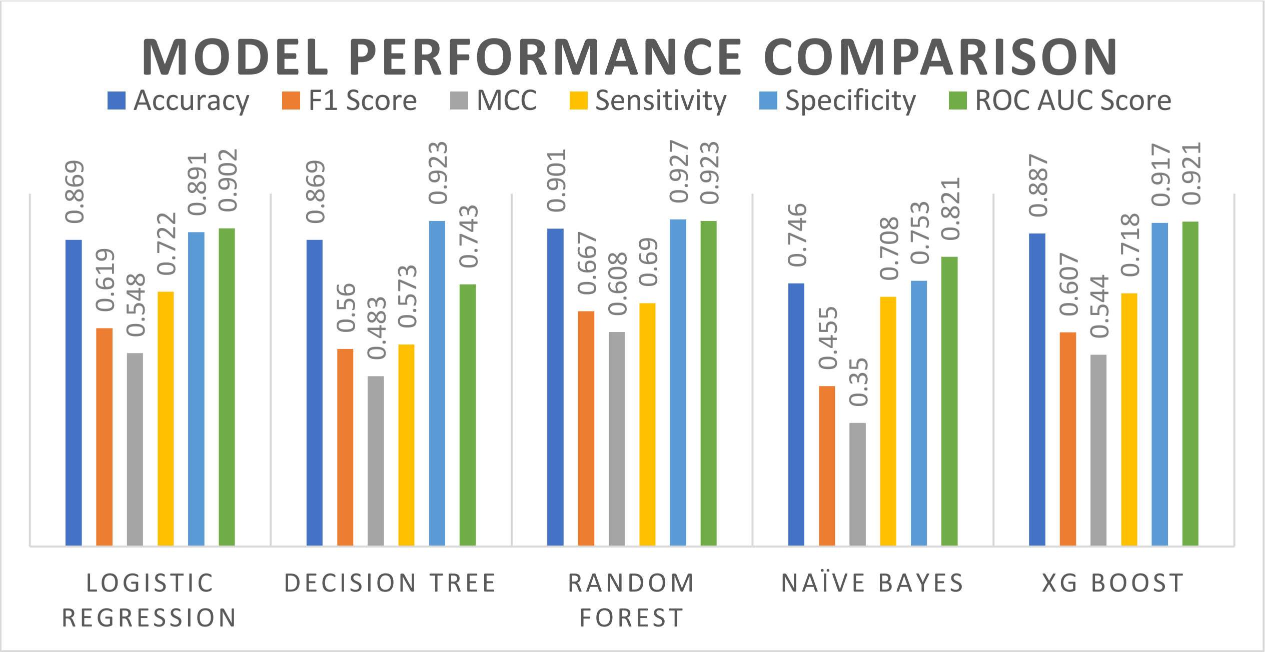
e. Figure shows confusion matrix and ROC curve for XGBoost model

11. <u>Conclusion</u>

In this project, we aimed to build a classification model to predict the revenue range of a company using various machine learning algorithms. We started with exploratory data analysis, where we visualized the data and checked for missing values and outliers. Then we preprocessed the data by encoding categorical variables, scaling numerical variables, and splitting the data into training and testing sets.

We trained five different models: logistic regression, decision tree, random forest, naive Bayes, and XGBoost. We evaluated the models using various metrics such as accuracy, F1 score, MCC, sensitivity, specificity, and ROC AUC score. We also plotted the confusion matrices and ROC curves for each model to visualize their performance.

Based on the evaluation metrics, we found that Random Forest outperformed all other models in terms of accuracy, F1 score, MCC, and ROC AUC score. However, we also noticed that the Random Forest model was overfitting the training data. To overcome this, we used early stopping and reduced the complexity of the model by tuning hyperparameters.



In conclusion, Random Forest is a powerful algorithm for classification tasks, especially when dealing with imbalanced datasets. It has several advantages, such as high accuracy, fast computation, and built-in regularization. However, it also has some disadvantages, such as being prone to overfitting and requiring careful tuning of hyperparameters. Overall, we were able to build a robust classification model to predict the revenue range of a company, and our analysis showed the importance of selecting appropriate evaluation metrics and tuning hyperparameters to improve model performance.