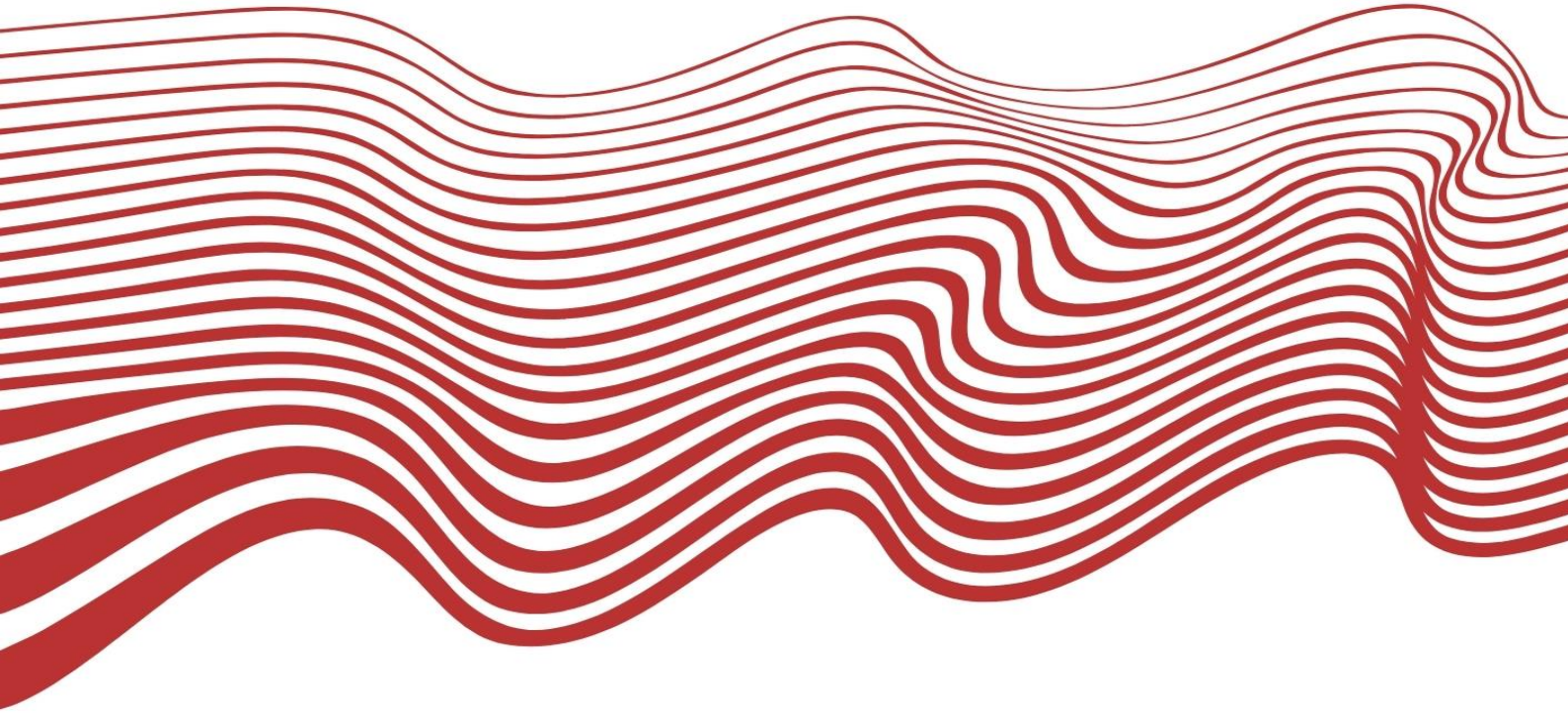
LEVERAGING ML FOR A DATA DRIVEN APPROACH IN E-COMMERCE



DARIO DI NUZZO

STUDENT ID: 23226607

CMP7228 Machine Learning

Contents

List of Figures

List of tables

Domain Description

Introduction to E-commerce Analytics

E-commerce analytics, a critical field in the digital economy, delves into data analysis and statistical methods to decode online shopping behaviours (Akter and Wamba, 2016). It encompasses various aspects, from tracking user activities to analysing purchasing patterns. This analytical approach is fundamental in e-commerce, guiding strategic decisions and operational planning (Micol Policarpo et al., 2021).

Importance of Data Analysis in Understanding Customer Behaviour Online

Data analysis in e-commerce, as noted by Kavitha et al. (2020), is crucial for understanding online customer behaviours. In today's digital landscape, where user data is abundant, such analysis provides valuable insights into customer trends and preferences. It plays a key role in enabling businesses to tailor experiences, boost satisfaction, and fine-tune marketing strategies. Furthermore, data-driven approaches in e-commerce, highlighted by Miguel Alves Gomes and Meisen (2023), are essential for effective customer segmentation, personalised marketing, and enhancing the overall shopping experience.

Current Trends and Challenges in the E-commerce Sector ok

In the continuing evolving e-commerce sector, the interplay of advanced technologies and consumer expectations is reshaping the industry. According to Liu et al. (2024) AI and Machine Learning are revolutionising the sector, enhancing data processing and customer service automation. This shift is not just about data analysis but about foreseeing and meeting customer needs with unprecedented precision. The trend towards Personalisation and Customisation, as Bilal et al. (2024) highlights, has become a new norm in the digital marketplace. Businesses are leveraging analytics to craft tailored experiences, significantly boosting customer engagement. However, as Chevalier (2024) and Van Bekkum and Zuiderveen Borgesius (2023) stressed, the surge in data-centric strategies brings the challenge of User Privacy and Data Security into sharp focus. Protecting consumer data against breaches and maintaining privacy is not just a technical issue but a crucial aspect for building trust and credibility in the digital marketplace.

Problem Definition

Objective 1: Predicting User Engagement (ProductRelated_Duration)

The primary goal under this objective is to predict 'ProductRelated_Duration', which represents the amount of time users spend on product-related pages of an e-commerce website. This metric is a critical indicator of user engagement and interest in the site's offerings. By accurately forecasting this variable, businesses can better understand the factors that drive customer interest and interaction with products, leading to more informed decisions about content placement and website design. Predictive models like Linear Regression and Random Forest Regression will be employed to identify the key determinants of user engagement, thereby providing actionable insights for enhancing user experience and possibly influencing purchase behaviour.

Objective 2: Revenue Prediction Analysis (Revenue Prediction)

This objective concentrates on employing machine learning techniques to predict the likelihood of a session resulting in a transaction, labelled as 'Revenue'. This binary classification task is pivotal for understanding and predicting which user sessions are likely to contribute to revenue generation. Various models such as Logistic Regression, K-Nearest Neighbour, Support Vector Machine and Ensemble Methods such as Random Forest, AdaBoost, and XGBoost will be utilised. These models will help unravel the complex patterns of user behaviour that lead to purchases, enabling businesses to optimise their marketing strategies and tailor user experiences to maximise conversion rates.

Objective 3: User Engagement Segmentation (Clustering User Patterns)

The third objective aims to segment users based on their engagement patterns on the e-commerce platform, using unsupervised learning algorithm K-Means. This segmentation exercise involves identifying distinct groups of users according to variables such as 'Informational Duration', 'Bounce Rates', and 'Exit Rates'. The segmentation is expected to reveal varying levels of engagement, providing insights into different customer profiles. By understanding these segments, businesses can tailor their marketing and content strategies more effectively to cater to the diverse needs and preferences of different user groups.

Data Set Description

The dataset from the UCI Machine Learning Repository, titled "Online Shoppers Purchasing Intention Dataset," is a rich compilation of data aimed at understanding customer behaviour on an e-commerce website (UCI, 2018). Comprising 12,330 sessions, the dataset's primary objective is to discern whether a session ends in a transaction (revenue generation). It includes 10 numerical and 8 categorical features, capturing diverse aspects of a user's interaction with the website. These aspects range from the type of pages visited to the duration spent on them, along with various other metrics such as bounce and exit rates (Trivedi et al., 2022). The dataset also incorporates temporal and behavioural data, including the time of the visit, type of visitor, and whether the visit occurred over a weekend (Mootha, Sridhar and Devi, 2021). Such comprehensive data enables a nuanced analysis of factors influencing online shopping decisions. Each feature in this dataset provides a unique insight into user behaviour, crucial for understanding and predicting online purchasing intentions (Sakar et al., 2018). The blend of numerical and categorical data offers a comprehensive view, facilitating a deeper analysis of factors that drive online shopping behaviours.

Number	Feature Name	Feature Type	Description
1	Administrative	Numerical	Number of visits to administrative pages.
2	Administrative_Duration	Numerical	Total time spent on administrative pages.
3	Informational	Numerical	Number of visits to informational pages.
4	Informational_Duration	Numerical	Total time spent on informational pages.
5	ProductRelated	Numerical	Number of visits to product-related pages.
6	ProductRelated_Duration	Numerical	Total time spent on product-related pages.
7	BounceRates	Numerical	Bounce rate of the website's pages.
8	ExitRates	Numerical	Exit rate from the website's pages.
9	PageValues	Numerical	Average value of a web page that a user visited.
10	SpecialDay	Numerical	Proximity of the visit time to a special day.
11	Month	Categorical	Month of the year during the visit.
12	OperatingSystems	Categorical	Type of operating system used by the visitor.
13	Browser	Categorical	Type of browser used by the visitor.
14	Region	Categorical	Geographical region from which the session was initiated.
15	TrafficType	Categorical	Type of traffic that led the visitor to the website.
16	VisitorType	Categorical	Categorisation of the visitor as "Returning" or "New".
17	Weekend	Categorical	Boolean indicator of whether the visit occurred on a weekend.
18	Revenue	Categorical	Boolean indicating whether the visit resulted in revenue.

Table 1: Feature Description

Data Set Exploration

Exploratory Data Analysis (EDA) is a critical step in data analysis, involving the use of statistical summaries and visualisations to understand initial patterns and characteristics in raw data (Komorowski et al., 2016). Using Python's Pandas DataFrame, the dataset was examined to understand its structure and data types (Pandas, 2020). This process included checking for and addressing data quality issues, such as removing 125 duplicate rows to ensure dataset integrity. No missing or null values were found, confirming the dataset's readiness for advanced analytical modelling (appendix 1).

Histograms

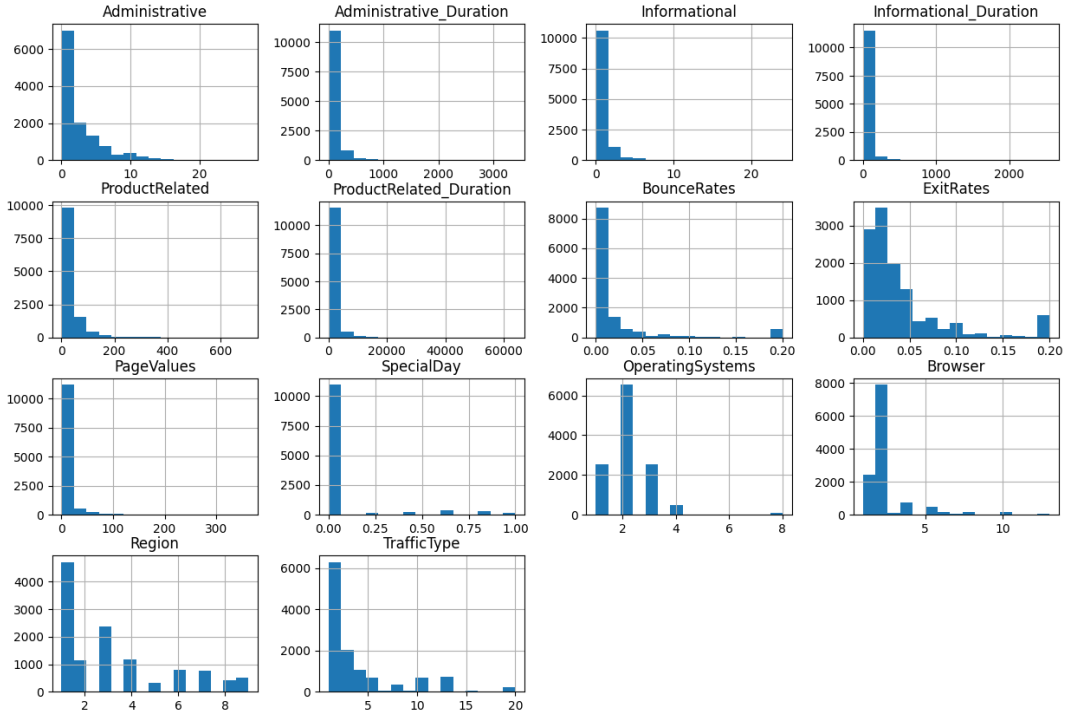


Figure 1: Histograms Plot

The histograms reveal insights into user behaviour of the e-commerce website (Figure 1). Users typically engage in few 'Administrative' activities and spend less time on 'Administrative_Duration'. Similarly, 'Informational' pages and 'Informational_Duration' see minimal engagement, with users spending little time on informational content. 'ProductRelated' pages have more varied visit counts, but very high visits are rare, and 'ProductRelated_Duration' also varies, with longer durations being uncommon. 'BounceRates' are generally low, suggesting users often browse multiple pages. 'ExitRates' are more evenly distributed. 'PageValues' and 'SpecialDay' appear to have a negligible direct effect on user sessions. Users predominantly use one or two types of 'OperatingSystems' and 'Browsers', indicating limited technical platform diversity. The 'Region' and 'TrafficType' histograms indicate certain regions and traffic types are more prevalent, potentially highlighting the website's popularity and user acquisition methods.

Box Plots

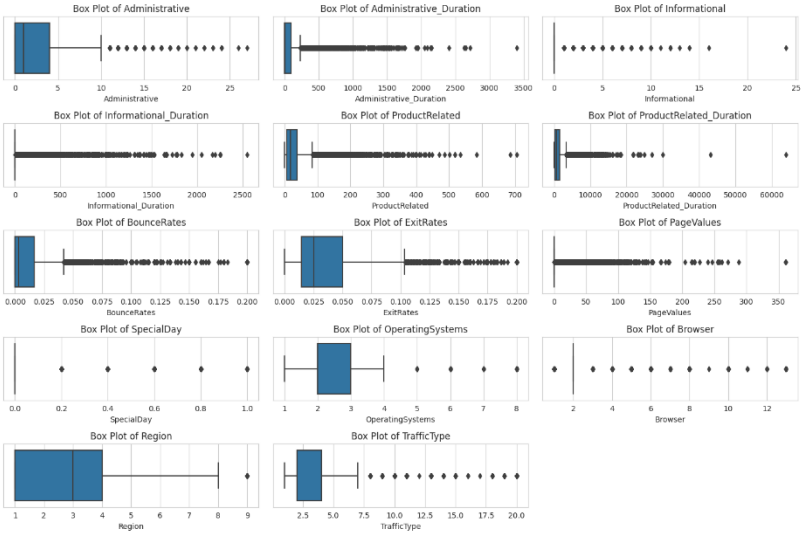


Figure 2: Box Plots

The box plots (Figure 2) reveal user engagement trends: 'Administrative' and 'Informational' activities have low engagement with notable exceptions. 'ProductRelated' activities show varied interactions, while 'BounceRates' and 'ExitRates' expose some abrupt departures. 'PageValues' impact revenue occasionally, 'SpecialDay' has little effect on behaviour, and user preferences for 'OperatingSystems' and 'Browsers' differ widely. 'Region' and 'TrafficType' data suggest varied effectiveness in user reach.

Scatter Plots

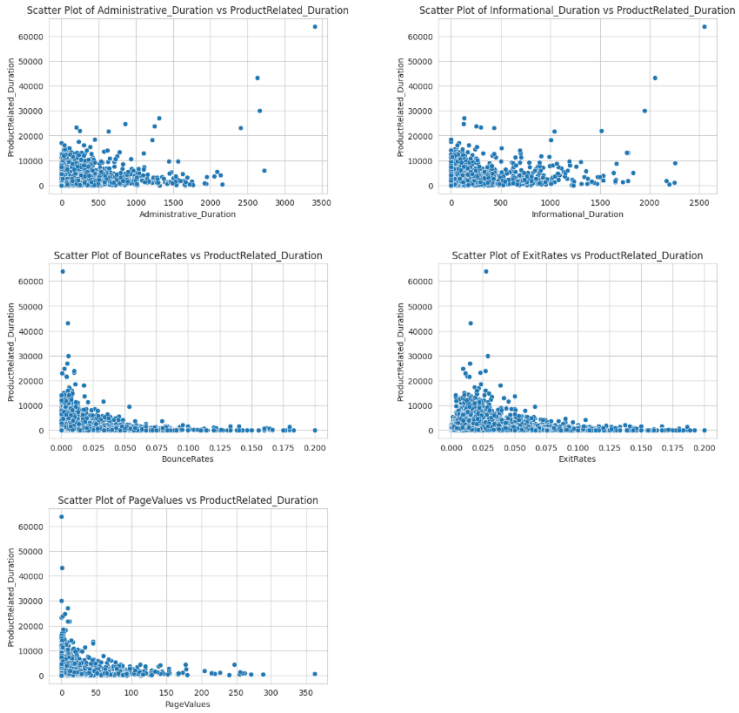


Figure 3: Scatter Plots

The scatter plots suggest different relationships between variables, which could be further examined through regression analysis (figure 3). A positive correlation between 'Administrative_Duration' and

'Informational_Duration' with 'ProductRelated_Duration' suggests these factors may predict increased time spent on product-related pages. Conversely, a negative correlation with 'BounceRates' and 'ExitRates' could predict reduced 'ProductRelated_Duration'. 'PageValues' show a complex relationship with time spent on product pages, requiring a nuanced regression approach.

Bar Charts (Revenue)

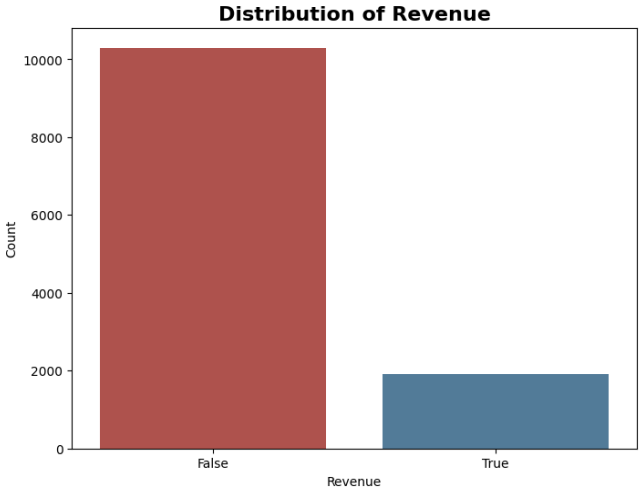


Figure 4: Distribution of Revenue

The bar chart illustrates the distribution of sessions that resulted in revenue against those that did not on an e-commerce platform (figure 4). It is clear from the chart that the number of sessions without revenue (labelled as 'False') significantly surpasses those that generated revenue (labelled as 'True'). Specifically, there were 10,297 sessions that did not result in revenue, while only 1,908 sessions were revenue-generating.

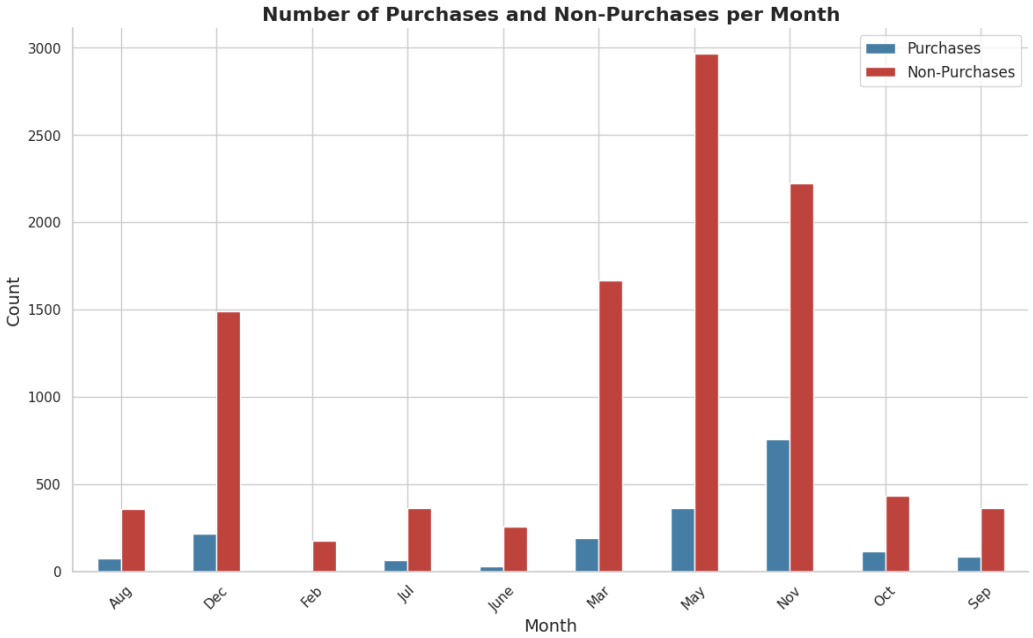


Figure 5: Bar Chart, Distribution of Purchases and Non-Purchases

The bar chart compares the number of purchases and non-purchases made each month of the e-commerce platform (figure 5). Notably, spikes in non-purchases occur during May and November, possibly coinciding with major sales events. While purchases also increase in these months, they do not match the growth in non-purchases, suggesting that promotions and seasonal factors boost traffic but not necessarily sales. This consistent disparity highlights opportunities to enhance conversion rates. Analysing these seasonal trends and customer behaviours could inform targeted marketing strategies aimed at converting more visitors into buyers.

Correlation Analysis

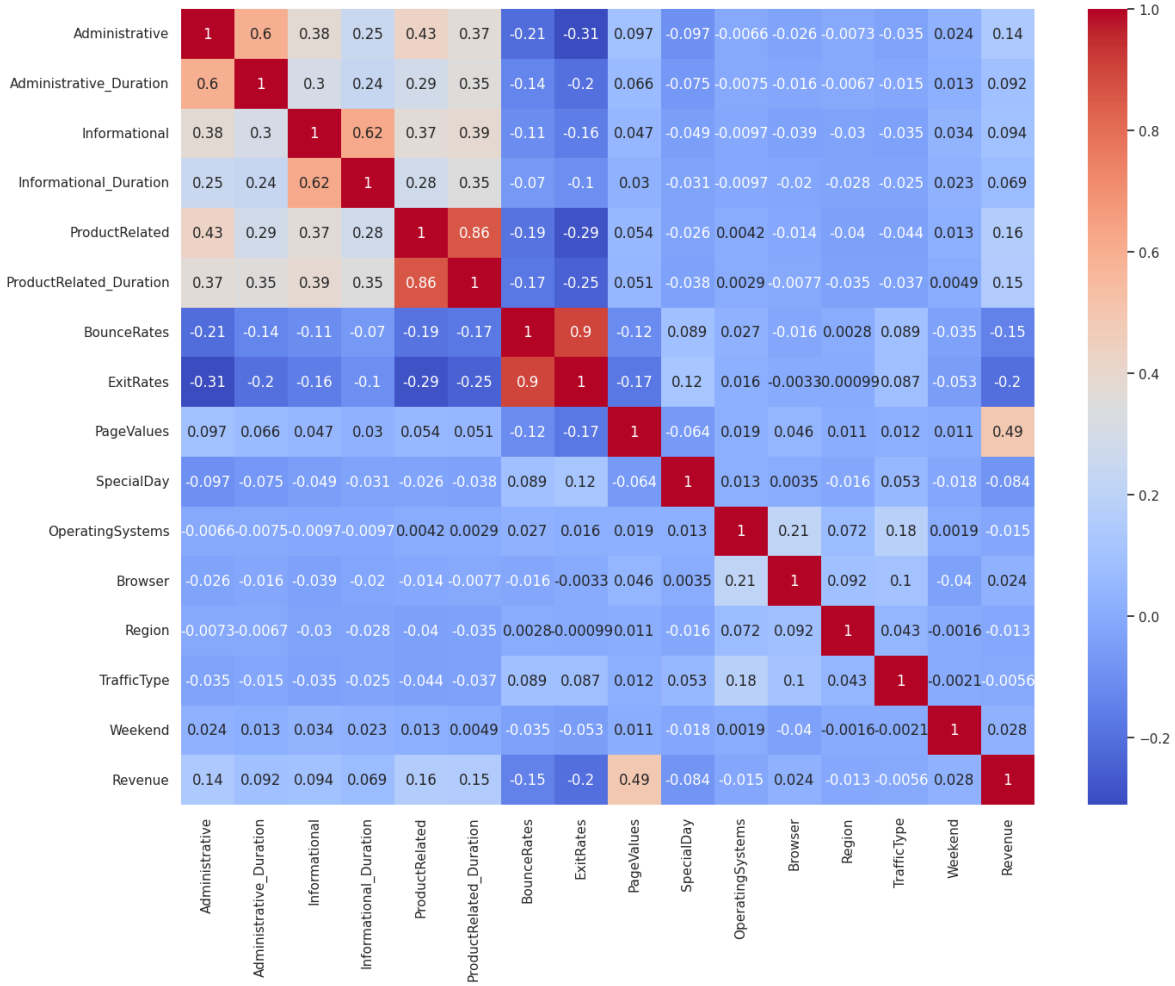


Figure 6: Correlation Matrix, Heatmap

The heatmap is useful for quickly identifying relationships between variables, which can be crucial for tasks such as feature selection in machine learning (Necula, 2023). In this heatmap, the colour intensity reflects the strength of the correlation, with red indicating a strong positive correlation and blue indicating a strong negative correlation (figure 6). White or lighter colours suggest little to no correlation between the variables. There's a moderate correlation between the time spent on administrative tasks and the number of such actions. Similarly, more time on informational pages correlates with increased informational activities. A strong correlation exists between the number of product pages visited and time spent on them. Bounce rates and exit rates are linked, suggesting that high rates in one predict high rates in the other. Page value moderately correlates with revenue generation, indicating some pages significantly drive sales. However, special days have minimal impact on user behaviour compared to other factors.

Scatterplot Matrix (Multivariate Analysis)

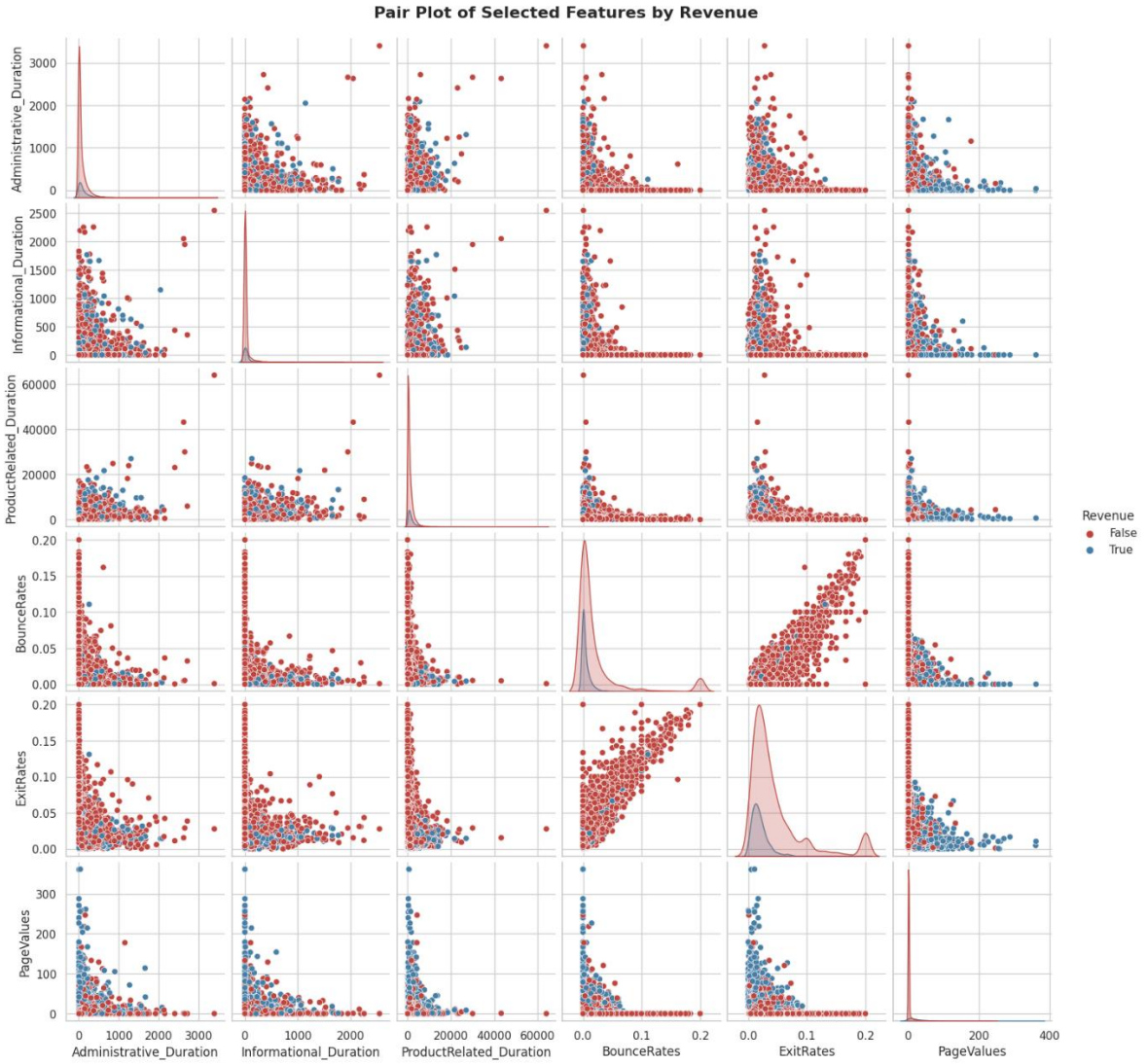


Figure 7: Pair Plot, Multivariate Analysis

The pair plot, also known as scatterplot matrix, presents a visual comparison of various website metrics against each other, differentiated by whether a user session resulted in a purchase (figure 7). For some variables, such as 'ProductRelated_Duration', there is a visible trend where sessions with purchases (blue points) tend to have higher values. 'BounceRates' and 'ExitRates' seem to be lower for sessions that resulted in purchases, which aligns with the expectation that users who buy something tend to explore more pages and do not leave the site immediately. 'PageValues' shows a strong distinction between the two groups, with purchase sessions having higher values, suggesting that sessions where users visit pages with higher page values are more likely to result in purchases.

Feature Engineering

In the dataset analysis, outliers were intentionally retained as they provide valuable insights into purchasing decisions, revealing unique and potentially influential customer behaviours that enrich the understanding of the data. In enhancing the predictive models, two new features were engineered from the existing data. 'Avg_Duration_Per_Visit' calculates the average time spent per page type (administrative, informational, product-related) per visit, offering a normalised view of user engagement. This metric adjusts for variances in session length (appendix 2).

Similarly, 'Bounce_Exit_Rate_Avg' combines bounce rate and exit rate, both indicators of session quality and user satisfaction. Averaging these rates provides an overall picture of site exit behaviour, useful for pinpointing potential issues in content or user experience. These new features enrich the dataset by providing deeper insights into user behaviour and are crucial for the predictive models, offering a more nuanced understanding of user interactions on the website.

Experiment and Evaluation

Regression Analysis

In the regression task, the goal was to forecast the 'ProductRelated_Duration' using other available data points in the dataset. This measure is vital as it reflects the time users are actively engaging with product-related pages, which is a significant indicator of interest and potential purchasing intent.

```
MSE (Linear Regression): 1282530.0165796638
R2 (Linear Regression): 0.7201080508007809
```

Figure 8: MSE and R2, Linear Regression

```
MSE (Random Forest): 609383.4951122614
R2 (Random Forest): 0.8670116628446105
```

Figure 9: MSE and R2, Random Forest Regressor

Two predictive modelling techniques were employed for this task, Linear Regression and Random Forest Regressor. Linear Regression (figure 8), known for its simplicity and interpretability, indicated that approximately 0.72% (appendix 3) of the variability in 'ProductRelated_Duration' could be comprehensively explained by the independent variables included in the model. This level of explanation is substantial, considering the complexity of online user behaviour. However, the Random Forest Regressor (figure 9), a more sophisticated model that leverages multiple decision trees to improve predictive power and control over-fitting, outperformed the Linear Regression model. It achieved an R2 score of 0.86% (appendix 4), signifying a very strong fit and suggesting that it could capture the nuances in the data that the simpler Linear Regression model might have missed.

	Feature	Importance
4	ProductRelated	0.773804
14	Avg_Duration_Per_Visit	0.182657
1	Administrative_Duration	0.024162
3	Informational_Duration	0.004840
2	Informational	0.003233
12	TrafficType	0.002168
0	Administrative	0.001674
6	ExitRates	0.001066
13	Weekend	0.000988
21	Month_May	0.000986

Figure 10: Feature Importance List, R.F. Regressor

The regression models offered insights into which features significantly predicted user engagement (figure 10). 'ProductRelated', 'Avg_Duration_Per_Visit', and 'Administrative_Duration' emerged as top contributors. These findings underscore the direct impact of the number of product pages visited and the engagement depth on certain pages. However, the models also revealed the trade-off between predictive power and model interpretability, which is a crucial consideration for practical applications in business contexts.

Classification Experiment

The dataset was initially divided into features (X) and the target variable (y), focusing on 'Revenue' as the primary variable for prediction. These features were further categorised into numerical and categorical types, each undergoing specialised preprocessing. Numerical features underwent standardisation to ensure a mean of zero and a standard deviation of one. In contrast, categorical features were one-hot encoded, converting them into a format suitable for model training (figure 11). The dataset was then split into training and testing sets, maintaining a 70/30 ratio to ensure distinct data points for training and performance evaluation (appendix 5).

```
(['Administrative',
  'Administrative_Duration',
  'Informational',
  'Informational_Duration',
  'ProductRelated',
  'ProductRelated_Duration',
  'BounceRates',
  'ExitRates',
  'PageValues',
  'SpecialDay',
  'OperatingSystems',
  'Browser',
  'Region',
  'TrafficType',
  'Avg_Duration_Per_Visit',
  'Bounce_Exit_Rate_Avg'],
 ['Month', 'VisitorType', 'Weekend'])
```

Figure 11: List of Features

The initial phase of the analysis involved logistic regression as a preliminary model (figure 12). This model attained an 88% accuracy rate in cross-validation, signifying a high level of consistency across different subsets of the dataset. Notably, it demonstrated a 90% precision in identifying sessions that did not result in revenue, categorised as the 'False' class. However, its performance was less impressive in predicting revenue-generating sessions ('True' class), with a lower precision of 75% and a mere 40% recall. This discrepancy indicated that while the model was effective in identifying non-revenue sessions, it was less definitive in confirming revenue-generating sessions.

```
Cross-Validation Metrics on Training Set:
CV Accuracy: 0.88 ± 0.01

Confusion Matrix on Test Set:
[[3044   72]
 [ 330  216]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.90      0.98      0.94      3116
        True       0.75      0.40      0.52       546

    accuracy                           0.89      3662
   macro avg       0.83      0.69      0.73      3662
weighted avg       0.88      0.89      0.88      3662
```

Figure 12: Logistic Regression Preliminary Model

To address the class imbalance often present in datasets, the Synthetic Minority Over-sampling Technique (SMOTE) was applied (appendix 6). SMOTE synthesises samples from the minority class to balance the dataset (Blagus and Lusa, 2013). After integrating SMOTE, the logistic regression model showed improved detection of the minority class, increasing the recall for revenue sessions to 75% but with a predictable decrease in precision due to more false positives (figure 13). The model's performance, post-SMOTE integration, was graphically represented using Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) metrics (figure 14).

```
Cross-Validation Metrics on Training Set (with SMOTE):
CV Accuracy: 0.85 ± 0.01

Confusion Matrix on Test Set (with SMOTE):
[[2726  390]
 [ 135  411]]

Classification Report on Test Set (with SMOTE):
              precision    recall  f1-score   support

       False       0.95      0.87      0.91      3116
        True       0.51      0.75      0.61       546

    accuracy                           0.86      3662
   macro avg       0.73      0.81      0.76      3662
weighted avg       0.89      0.86      0.87      3662
```

Figure 13: Logistic Regression, (SMOTE)

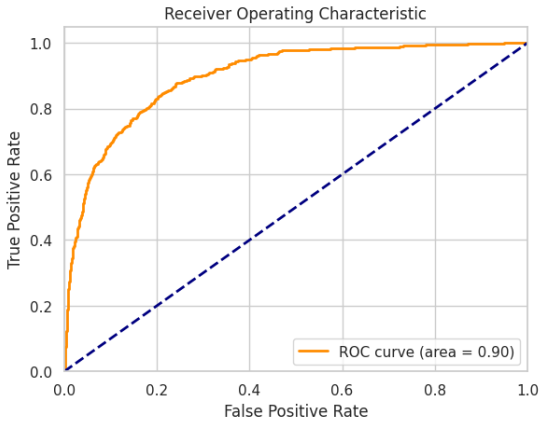


Figure 14: ROC, Logistic Regression (SMOTE)

Subsequently, the Random Forest model was introduced (Appendix 7). Known for its ability to capture complex patterns through decision tree ensembles, it was further enhanced with SMOTE. The model achieved a notable 90% accuracy, and its feature importance analysis provided critical insights into revenue generation, identifying 'PageValues', 'ExitRates', and 'ProductRelated_Duration' as significant factors.

```
Cross-Validation Metrics on Training Set with SMOTE (Random For
CV Accuracy: 0.89 ± 0.01

Confusion Matrix on Test Set with SMOTE (Random Forest):
[[2890  226]
 [ 156  390]]

Classification Report on Test Set with SMOTE (Random Forest):
              precision    recall  f1-score   support

       False       0.95      0.93      0.94      3116
        True       0.63      0.71      0.67       546

    accuracy                           0.90      3662
   macro avg       0.79      0.82      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```

Figure 15: Random Forest Report

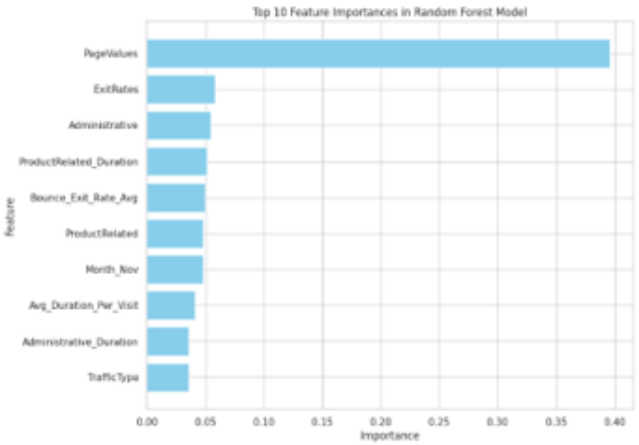
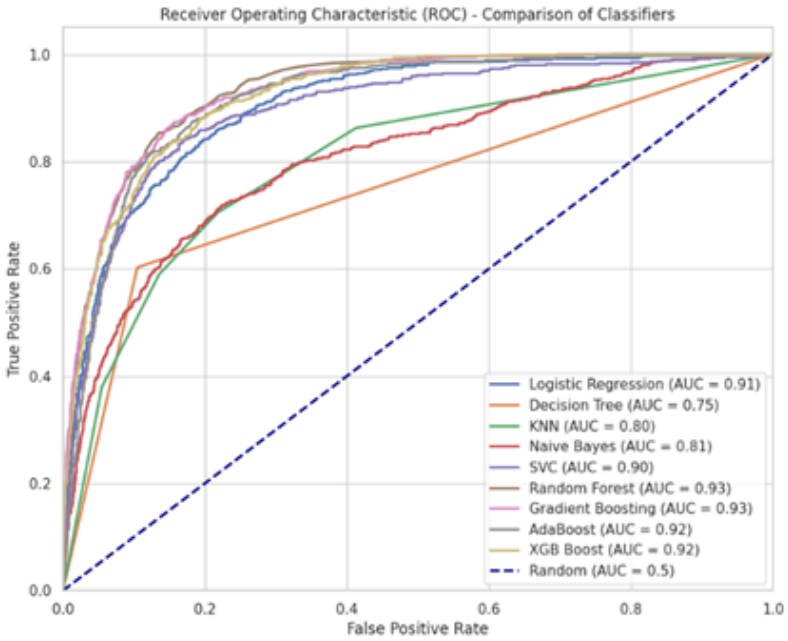


Figure 16: Feature Importance Plot, R.F. Classifier

The experimentation also employed the K-Nearest Neighbours (KNN) model, which bases predictions on proximity principles (Guo et al., 2003). Key parameters such as the number of neighbours and leaf size were optimised for enhanced generalisation (appendix 8). Support Vector Machines (SVM) were also utilised, recognised for their effectiveness in high-dimensional spaces (Tang et al., 2008). Parameters such as the penalty parameter 'C', gamma values, and kernel types were tuned for optimal decision boundaries (appendix 9). Decision tree and Ensemble methods (appendix 10), including Gradient Boosting, AdaBoost, and XGBoost, were implemented to amalgamate weaker learners into robust predictive models (Scikit Learn, 2012). Hyperparameters for these methods were fine-tuned using GridSearchCV, ensuring optimal performance (Gunasegaran and Cheah, 2017; Scikit Learn, 2019). The models' effectiveness was measured using Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) scores, which highlighted sensitivity and specificity trade-offs at varying thresholds (Wu and Flach, 2005). These metrics, along with confusion matrices, were instrumental in comparing the models' performance before and after GridSearchCV tuning. A higher

AUC score signified superior model performance, with the ROC curve visually representing the



balance between sensitivity and specificity across different thresholds (figure 17).

Figure 17: ROC and AUC Plot (all models)

In the final stages of the experiment, advanced ensemble techniques as Stacking and Voting classifiers were introduced. These methods aimed to capitalise on the collective strengths of various models for improved predictive accuracy. Stacking involved combining the predictions of multiple models and applying Logistic Regression as Meta Learner to derive final predictions (figure 18). In contrast, the Hard Voting method combined individual model predictions for a final decision. The effectiveness of these ensemble methods was evident in their performance metrics, often on surpassing or being more balanced than individual models (figure 19).

```
Confusion Matrix for Stacked Model:
[[2930  186]
 [ 184  362]]

Classification Report for Stacked Model:
              precision    recall  f1-score   support

       False       0.94      0.94      0.94      3116
        True       0.66      0.66      0.66       546

    accuracy                           0.90      3662
   macro avg       0.80      0.80      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```

Figure 18: Staked Model Report

```
Confusion Matrix for Voting Model:
[[2897  219]
 [ 155  391]]

Classification Report for Voting Model:
              precision    recall  f1-score   support

       False       0.95      0.93      0.94      3116
        True       0.64      0.72      0.68       546

    accuracy                           0.90      3662
   macro avg       0.80      0.82      0.81      3662
weighted avg       0.90      0.90      0.90      3662
```

Figure19: Voting Classifier Report

This experimentation employed a diverse and sophisticated blend of machine learning techniques to predict revenue generation in e-commerce. The integration of various models, especially advanced ensemble methods in the later stages, showcased the benefits of combining multiple analytical

approaches. This strategy enhanced predictive accuracy and provided deeper insights into e-commerce data analysis. Furthermore, the use of these varied techniques highlights the complexity of e-commerce predictions and the value of a multifaceted, data-driven approach in contemporary analytic practices.

Clustering Experiment

The clustering experiment aimed to segment website users based on engagement metrics such as 'Informational_Duration', 'BounceRates', and 'ExitRates'. This segmentation was achieved using unsupervised learning methods, with K-Means clustering being the algorithm of choice due to its recognised efficiency (Tabianan, Velu and Ravi, 2022). Clustering is a form of unsupervised learning that groups data points so that those within each group are similar to each other than to those in other groups (Na, Xumin and Yong, 2010). This task is particularly relevant for understanding different user behaviours and can guide personalised marketing strategies. Before applying K-Means, dimensionality reduction was performed using Principal Component Analysis also named PCA (appendix 11). PCA simplifies the complexity of high-dimensional data while preserving trends and patterns, which is crucial for effective clustering (Ding and He, 2004).

Elbow Method and Silhouette Score

The optimal number of clusters was determined using the Elbow method (appendix 11). This method evaluates the inertia decrease rate against the number of clusters, plotting explained variance as a function of the number of clusters (Nainggolan et al., 2019). The 'elbow' point on this curve, represents the ideal number of clusters for segmentation (figure 20).

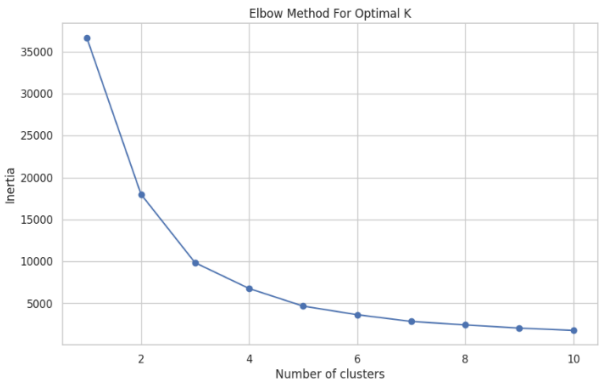


Figure 20: Elbow Metod

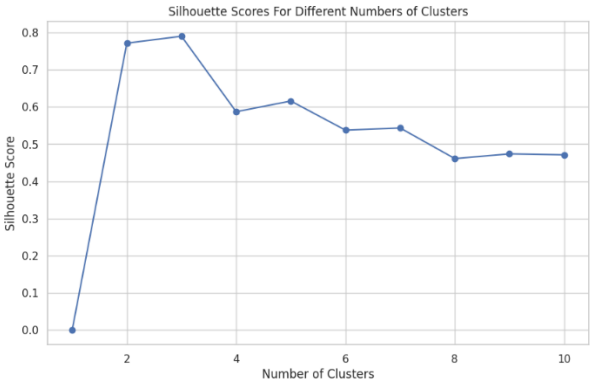


Figure 21: Silhouette Scores

Subsequently, silhouette scores were calculated for different cluster sizes to assess the distinctiveness of the clusters (Shutaywi and Kachouie, 2021). A high silhouette score (appendix 12), approximately 0.79 in this case, signifies well-separated and clearly defined clusters (figure 21). Three clusters were identified as the optimal segmentation for user interaction patterns, suggesting that users could be broadly categorised into three distinct types based on their website interactions. This segmentation is invaluable for customising user experiences and refining marketing efforts. Finally, to aid in visualisation and understanding of these clusters, 2D and 3D plots were created. These visuals will provide in the analysis chapter a clear picture of how data points are distributed across the clusters, offering concrete insights into the various user segments.

Analysis and Results chapter

Regression Analysis

The journey began with regression analysis, where the primary goal was to predict 'ProductRelated_Duration', a key metric indicating the time users spend on product-related pages. Two models were at the forefront Linear Regression and Random Forest Regressor. Linear Regression (figure 22) provided a basic understanding, explaining approximately 0.72% of the variability in 'ProductRelated_Duration'. However, the Random Forest Regressor (figure 23) took the lead, explaining up to 0.86% of the variability, a notable improvement that highlighted its ability to capture the complexities in user behaviour more effectively.

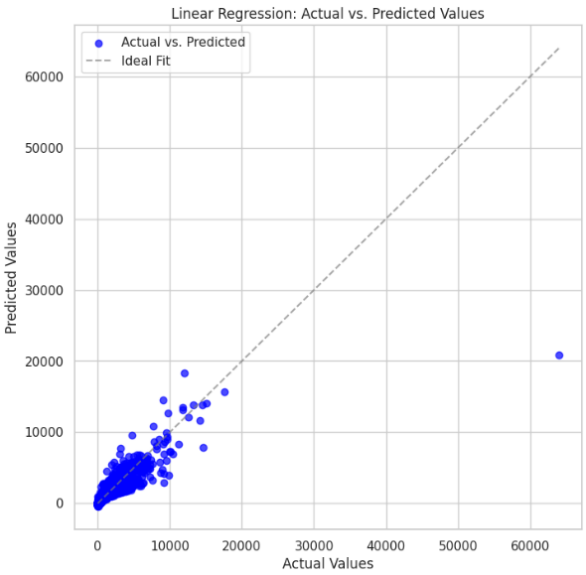


Figure 22: Linear Regression, Plot

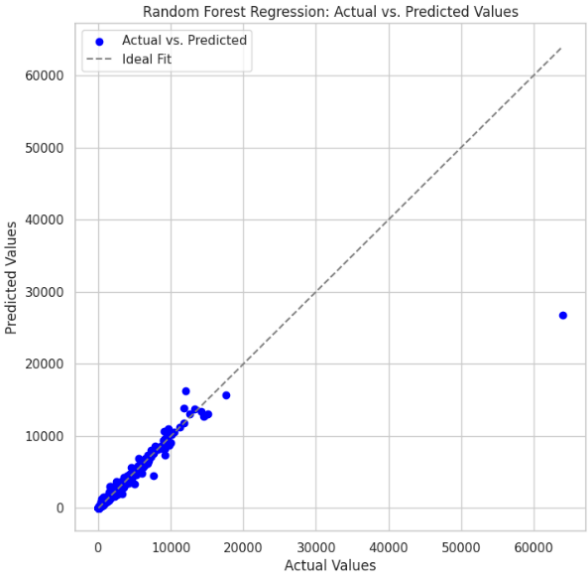


Figure 23: Random Forest Regressor, Plot

Feature Importance Analysis

A critical aspect of the regression analysis was the evaluation of feature importance, which identified 'ProductRelated', 'Avg_Duration_Per_Visit', and 'Administrative_Duration' as key contributors to predicting the time spent on product-related pages. The 'ProductRelated' variable's prominence underscores the direct relationship between the number of product pages visited and the time allocated to them. The 'Avg_Duration_Per_Visit' offers a nuanced view of engagement by averaging the duration across different types of pages, hence providing a balanced metric of user interest. 'Administrative_Duration' being a significant predictor suggests that administrative interactions, possibly reflecting user inquiries or account management activities, are also indicative of engagement levels.

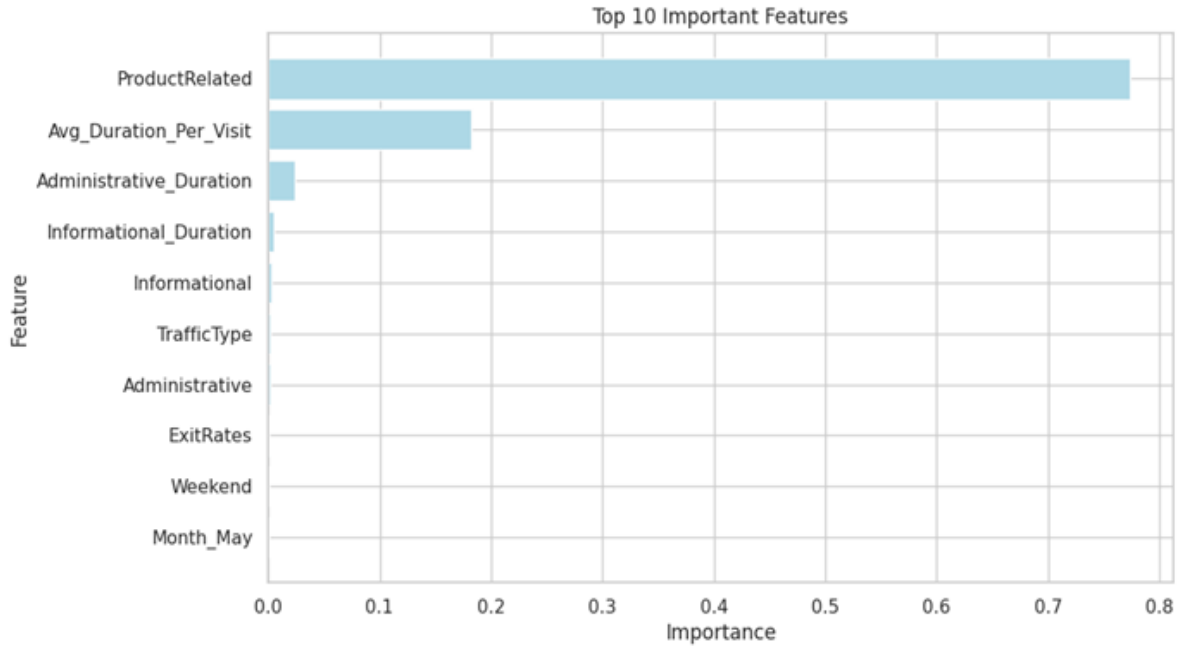


Figure 24: Feature Importance Plot, R.F. Regressor

The critical implication of this analysis is the affirmation that certain user behaviours and interactions on the website are strong indicators of their engagement. However, it's also essential to recognise that while the Random Forest model exhibits a better fit, it comes at the cost of interpretability, which is a trade-off that must be considered when applying the model in a business context. The model's complexity can make it challenging to fully understand the underlying reasons for its predictions, which is crucial when making data-driven decisions to enhance user experience and ultimately drive sales.

Classification Analysis

In predicting 'Revenue' for e-commerce, various machine learning models underwent scrutiny for their predictive accuracy before and after fine-tuning. Initially, Logistic Regression showed promise with an 85% accuracy post-SMOTE implementation, but its recall for revenue sessions was moderate, signalling potential for refinement.

Pre-Tuning Insights

Decision Tree, KNN, and Naive Bayes yielded mixed results, with Naive Bayes standing out for recall despite lower overall accuracy. SVC demonstrated its capacity with 87% accuracy and solid recall. The ensemble methods, particularly Random Forest, Gradient Boosting, and AdaBoost, outperformed single estimators, balancing precision and recall effectively. XGB Boost maintained high precision, showing resilience in managing imbalanced data. The Decision Tree model demonstrated a comparable accuracy with notable precision for the false class and reasonable recall for the true class. KNN and Naive Bayes provided contrasting results, with Naive Bayes showing a striking recall for the true class, despite a lower overall accuracy. SVC, with its 87% accuracy, offered promising recall rates, indicating its potential effectiveness in classifying sessions. Random Forest, Gradient Boosting, and AdaBoost outperformed single estimators in terms of accuracy and balance between precision and recall for both classes. XGB Boost, with its high precision for the false class and solid recall for the true class, showcased its robustness in handling imbalanced data.

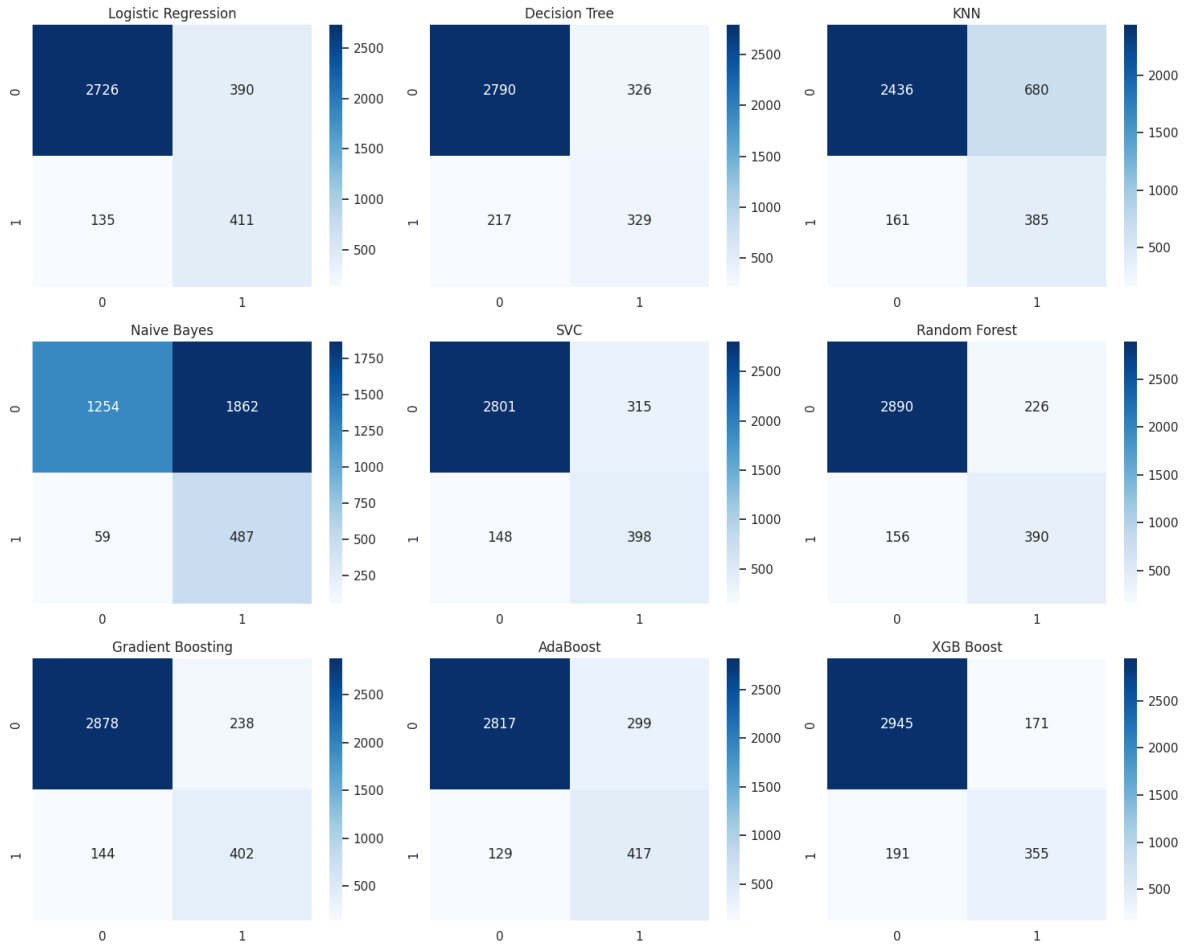


Figure 25: Matrices Before Tuning

After Tuning

Post-tuning with GridSearchCV (appendix 13), the Decision Tree and KNN models displayed notable improvements in precision and recall. SVC retained high precision and increased its recall. Random Forest stood out with high accuracy and enhanced recall for the revenue class, indicating strong predictive performance. Gradient Boosting and AdaBoost demonstrated gains, particularly in achieving a balanced recall for the true class. XGB Boost maintained high accuracy and exhibited an improved recall for the revenue class, affirming its effectiveness in e-commerce predictions. Notably, the Voting Classifier managed to surpass the staked model in recall with lower precision in recognising the first class.

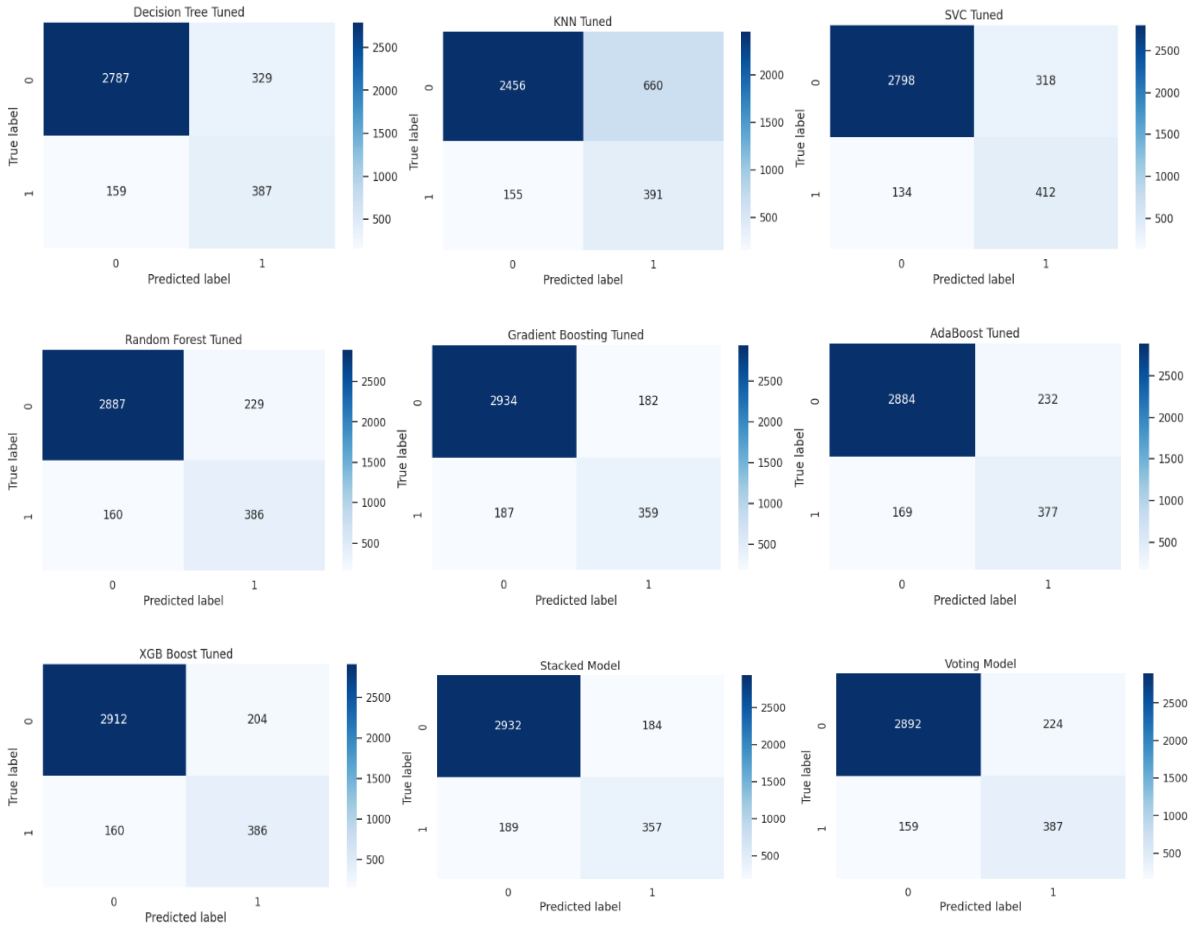


Figure 26: Matrices After Tuning

The confusion matrices provided a visual representation of each model's performance, elucidating the nuances of their predictive capabilities. These matrices depicted the true positives, false positives, true negatives, and false negatives, providing a comprehensive view of each model's strengths and weaknesses. The analysis demonstrated the efficacy of XGBoost in predicting e-commerce revenue, while emphasising the importance of model diversity and optimisation for strategic decision-making. Finally, this analysis highlights the transformational impact of a data-driven approach in e-commerce revenue prediction.

Clustering Analysis

Understanding user behaviour is fundamental in e-commerce, where each interaction holds potential value (Islam et al., 2023). The segmentation of users into distinct groups based on their site engagement patterns can inform targeted business strategies.

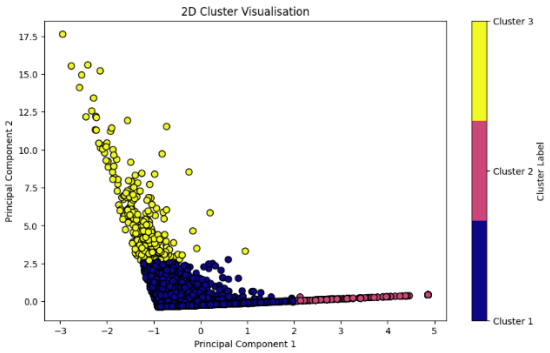


Figure 27: 2D Plot Clustering

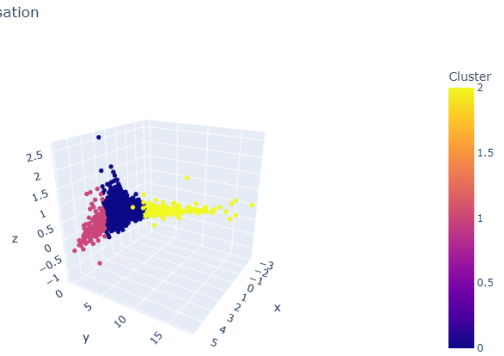


Figure 28: 3D Plot Clustering

In e-commerce user segmentation, "The Quick Browsers" are characterised by brief visits and high exit metrics, potentially indicating early buying stages or ineffective content capture. To harness this opportunity, businesses can enhance initial content to deepen engagement and mitigate bounce rates.

Cluster 1 (Blue)

The "Engaged Researchers" allocate moderate time to consuming information, actively evaluating their purchasing decisions. For e-commerce platforms, the challenge is to maintain these users' interest and guide them towards transactions, possibly through enhanced content and targeted communication.

Cluster 2 (Red)

The "Committed Explorers," who invest substantial time on the website, are likely in the decisive phase of their purchase journey. They represent a prime conversion opportunity if businesses can provide detailed content tailored to their research-intensive behaviour.

Cluster 3 (Yellow)

This group is characterised by their brief and efficient interactions with the e-commerce platform. "Quick Browsers" tend to have shorter session durations and higher bounce rates. For " Quick Browsers," the focus should be on captivating content; for "The Engaged Researchers," on depth and accessibility of information; and for "The Committed Explorers," on detailed, personalised content. This strategic differentiation in content and user experience can effectively navigate users towards conversion, leveraging the specific needs and behaviours of each segment in the customer journey.

Conclusion

In the Analysis and Results chapter, Linear Regression and regression Random Forest Regressor have proven effective in decoding user engagement on e-commerce platforms, focusing on 'ProductRelated_Duration' as target variable. Feature importance analysis revealed significant predictors such as 'ProductRelated' visits and 'Avg_Duration_Per_Visit', emphasising the impact of quality content on user engagement. Classification models, after tuning, showed marked improvements in accuracy and recall, with ensemble methods where the XGBoost Classifier excelled in precision and balance. The clustering analysis distinguished three user segments; "The Quick Browsers", "The Engaged Researchers", and "The Committed Explorers" each requiring tailored content strategies to maximise conversion. In conclusion, this multifaceted approach illustrates the value of nuanced analytics in e-commerce, highlighting the interplay between machine learning models and strategic business applications.

Limitation and Future Research

This report on e-commerce analytics reveals significant insights but also highlights key limitations that guide future research. A notable gap is the dataset's focus on session metrics, lacking in-depth user demographics, which could offer a richer understanding of customer behaviours. Questions about the models' generalisability arise, as their effectiveness across various e-commerce platforms remains untested. Additionally, the complexity and unpredictability of online user behaviour are not fully captured, indicating a possible oversimplification in current models. This calls for more sophisticated approaches that better reflect the dynamic nature of online interactions. The rapid advancement of AI and machine learning technologies also suggests that current methods may soon become outdated, urging a focus on newer, more powerful analytical tools. Moreover, integrating these models with real-time data is an unexplored yet promising area, offering opportunities for agile and responsive decision-making.

Strategic Implication for e-commerce

The report's findings have significant strategic implications for e-commerce. Key among them is the need for personalised content tailored to distinct user segments, enhancing engagement and conversion rates. This personalisation not only improves user experience but also fosters customer loyalty. Insights from predictive models are critical for data-driven decision-making, shifting from intuition-based to data-informed strategies. This shift involves applying data insights across various business aspects, such as optimising website design and refining marketing campaigns, to better meet customer needs and anticipate future trends. Additionally, prioritising high-impact variables is essential in the data-rich e-commerce environment. Focusing on variables with significant predictive power streamlines resources and efforts, enhancing operational efficiency and user experience. This report illustrates the transformative potential of advanced analytics in e-commerce. By effectively integrating machine learning models with strategic business practices, e-commerce entities can gain nuanced insights into customer behaviour. The future of e-commerce will significantly rely on the ability to leverage these data-driven insights for informed decision-making, ensuring both enhanced user experiences and business growth. However, navigating the challenges of model complexity and ethical data usage will be crucial in harnessing the full potential of e-commerce analytics.

References

Afifi, A., May, S., Donatello, R.A. and Clark, V. (2019). *Practical Multivariate analysis.* 6th ed. Virginia: Chapman and Hall/CRC, pp.37–56.

Akter, S. and Wamba, S.F. (2016). Big Data Analytics in E-commerce: a Systematic Review and Agenda for Future Research. *Electronic Markets*, [online] 26(2), pp.173–194. https://doi.org/10.1007/s12525-016-0219-0.

Bilal, M., Zhang, Y., Cai, S., Akram, U. and Halibas, A. (2024). Artificial Intelligence Is the Magic Wand Making customer-centric a reality! An Investigation into the Relationship between Consumer Purchase Intention and Consumer Engagement through Affective Attachment. *Journal of Retailing and Consumer Services*, [online] 77, p.103674. https://doi.org/10.1016/j.jretconser.2023.103674.

Blagus, R. and Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced Data. *BMC Bioinformatics*, 14(1). https://doi.org/10.1186/1471-2105-14-106.

Chevalier, S. (2024). *Personalization in e-commerce - Statistics & Facts*. [online] Statista. Available at: https://www.statista.com/topics/11400/personalization-in-e-commerce/#topicOverview [Accessed 14 Jan. 2024].

Ding, C. and He, X. (2004). K-means Clustering via Principal Component Analysis. In: *Proceedings of the twenty-first International Conference on Machine Learning*. p.29. https://doi.org/doi/10.1145/1015330.1015408.

Gunasegaran, T. and Cheah, Y.-N. (2017). Evolutionary Cross Validation | IEEE Conference Publication | IEEE Xplore. In: *ieeexplore.ieee.org*. [online] Amman, Jordan: IEEE. https://doi.org/10.1109/ICITECH.2017.8079960.

Guo, G., Wang, H., Bell, D., Bi, Y. and Greer, K. (2003). KNN Model-Based Approach in Classification. *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, 2888, pp.986–996. https://doi.org/10.1007/978-3-540-39964-3_62.

Islam, S., Naeem, J., Emon, A.S., Baten, A., Al Mamun, A., Waliullah, G.M., Rahman, Md.Saifur. and Mridha, M.F. (2023). Prediction of Buying Intention: Factors Affecting Online Shopping | IEEE Conference Publication | IEEE Xplore. 2023 International Conference on Next-Generation Computing, IoT and Machine Learning (NCIM). https://doi.org/10.1109/NCIM59001.2023.10212766.

Kavitha, Duncan T, S., Ravikumar, P. and E, V. (2020). Online Shopping Customer Behaviour Analysis Using Centrality Measures | IEEE Conference Publication | IEEE Xplore. *ieeexplore.ieee.org*. https://doi.org/10.1109/ICAIT47043.2019.8987252.

Komorowski, M., Marshall, D.C., Salciccioli, J.D. and Crutain, Y. (2016). Exploratory Data Analysis. *Secondary Analysis of Electronic Health Records*, pp.185–203. https://doi.org/10.1007/978-3-319-43742-2_15.

Liu, H., Zhao, J., Zhou, L., Yang, J. and Liang, K. (2024). Intelligent performance evaluation of e-commerce express services using machine learning: A case study with quantitative analysis. *Expert Systems with Applications*, 240, p.122511. https://doi.org/10.1016/j.eswa.2023.122511.

Micol Policarpo, L., da Silveira, D.E., da Rosa Righi, R., Antunes Stoffel, R., da Costa, C.A., Victória Barbosa, J.L., Scorsatto, R. and Arcot, T. (2021). Machine Learning through the Lens of e-commerce initiatives: an up-to-date Systematic Literature Review. *Computer Science Review*, 41, p.100414. https://doi.org/10.1016/j.cosrev.2021.100414.

Miguel Alves Gomes and Meisen, T. (2023). A Review on Customer Segmentation Methods for Personalized Customer Targeting in e-commerce Use Cases. *Information Systems and e-Business Management*, 21, pp.527–570. https://doi.org/10.1007/s10257-023-00640-4.

Mootha, S., Sridhar, S. and Devi, K. (2021). *A Stacking Ensemble of Multi Layer Perceptrons to Predict Online Shoppers' Purchasing Intention | IEEE Conference Publication | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/abstract/document/9315447/authors#full-text-header [Accessed 13 Jan. 2024].

Na, S., Xumin, L. and Yong, G. (2010). Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm | IEEE Conference Publication | IEEE Xplore. *ieeexplore.ieee.org*. https://doi.org/10.1109/IITSI.2010.74.

Nainggolan, R., Perangin-angin, R., Simarmata, E. and Tarigan, A.F. (2019). Improved the Performance of the K-Means Cluster Using the Sum of Squared Error (SSE) Optimized by Using the Elbow Method. *Journal of Physics: Conference Series*, 1361, p.012015. https://doi.org/10.1088/1757-899X/336/1/012017.

Necula, S.-C. (2023). Exploring the Impact of Time Spent Reading Product Information on E-Commerce Websites: a Machine Learning Approach to Analyze Consumer Behavior. *Behavioral Sciences*, 13(6), p.439. https://doi.org/10.3390/bs13060439.

Pandas (2020). *pandas.DataFrame — pandas 0.25.3 documentation*. Pydata.org. Available at: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html [Accessed 13 Jan. 2024].

Sakar, C.O., Polat, S.O., Katircioglu, M. and Kastro, Y. (2018). Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. *Neural Computing and Applications*, 31(10), pp.6893–6908. https://doi.org/10.1007/s00521-018-3523-0.

Scikit Learn (2012). *1.11. Ensemble Methods — scikit-learn 0.22.1 Documentation*. Scikit-learn.org. Available at: https://scikit-learn.org/stable/modules/ensemble.html [Accessed 14 Jan. 2024].

Scikit Learn (2019). *sklearn.model_selection.GridSearchCV — scikit-learn 0.22 Documentation*. Scikit-learn.org. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html [Accessed 14 Jan. 2024].

Shutaywi, M. and Kachouie, N.N. (2021). Silhouette Analysis for Performance Evaluation in Machine Learning with Applications to Clustering. *Entropy*, 23(6), p.759. https://doi.org/10.3390/e23060759.

Smith, P.F., Ganesh, S. and Liu, P. (2013). A Comparison of Random Forest Regression and Multiple Linear Regression for Prediction in Neuroscience. *Journal of Neuroscience Methods*, 220(1), pp.85–91. https://doi.org/10.1016/j.jneumeth.2013.08.024.

Tabianan, K., Velu, S. and Ravi, V. (2022). K-Means Clustering Approach for Intelligent Customer Segmentation Using Customer Purchase Behavior Data. *Sustainability*, 14(12), p.7243. https://doi.org/10.3390/su14127243.

Tang, Y., Zhang, Y.-Q., Chawla, N. and Krasser, S. (2008). *SVMs Modeling for Highly Imbalanced Classification | IEEE Journals & Magazine | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/document/4695979 [Accessed 14 Jan. 2024].

Trivedi, S.K., Patra, P., Srivastava, P.R., Zhang, J.Z. and Zheng, L.J. (2022). What prompts consumers to purchase online? A machine learning approach. *Electronic Commerce Research*. https://doi.org/10.1007/s10660-022-09624-x.

UCI (2018). *UCI Machine Learning Repository*. archive.ics.uci.edu. Available at: https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset [Accessed 13 Jan. 2024].

Van Bekkum, M. and Zuiderveen Borgesius, F. (2023). Using sensitive data to prevent discrimination by artificial intelligence: Does the GDPR need a new exception? *Computer Law & Security Review*, 48, p.105770. https://doi.org/10.1016/j.clsr.2022.105770.

Wu, S. and Flach, P. (2005). A Scored AUC Metric for Classifier Evaluation and Selection. *in Second Workshop on ROC Analysis in ML*.

Appendices

UCI Repository Online Shoppers Behaviour Intention Data Set:

https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset

Google Colaboratory Link:

https://colab.research.google.com/drive/1kI6Z8Urk8GqV_sEhaprSsoPdu57ijtXD?usp=sharing

Appendix 1

```
# Checking for missing or null values in the dataset
missing_values = data.isnull().sum()
missing_values[missing_values > 0]
```

```
Series([], dtype: int64)
```

```
# Checking for NaN values across the entire dataset
nan_values_total = data.isna().sum().sum()
nan_values_total
```

```
0
```

```
# Checking for duplicate rows in the dataset
duplicate_rows = data.duplicated().sum()
duplicate_rows
```

```
125
```

```
# Dropping duplicate rows
data = data.drop_duplicates()
duplicate_rows = data.duplicated().sum()
duplicate_rows
```

```
0
```

Appendix 2

```
# Creating a new variable: 'Avg_Duration_Per_Visit'
# This represents the average time a user spends on each page they visit
data['Avg_Duration_Per_Visit'] = (data['Administrative_Duration'] + data['Informational_Duration'] + data['ProductRelated_Duration']) / (data['Administrative'] + data['Informational'] + data['ProductRelated'])
data['Avg_Duration_Per_Visit'].fillna(0, inplace=True)  # Handling division by zero

# Creating a new feature by combining BounceRates and ExitRates:'Bounce_Exit_Rate_Avg'
# Using a simple average for this combination
data['Bounce_Exit_Rate_Avg'] = (data['BounceRates'] + data['ExitRates']) / 2

# Display the first few rows of the dataset to show the new feature
data[['BounceRates', 'ExitRates', 'Bounce_Exit_Rate_Avg']].head()

# Displaying the first few rows with the new variable
data.head()
```

ration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue	Avg_Duration_Per_Visit	Bounce_Exit_Rate_Avg
0.0	1	0.000000	0.20	0.20	0.0	0.0	Feb	1	1	1	1	Returning_Visitor	False	False	0.000000	0.200
0.0	2	64.000000	0.00	0.10	0.0	0.0	Feb	2	2	1	2	Returning_Visitor	False	False	32.000000	0.050
0.0	1	0.000000	0.20	0.20	0.0	0.0	Feb	4	1	9	3	Returning_Visitor	False	False	0.000000	0.200
0.0	2	2.666667	0.05	0.14	0.0	0.0	Feb	3	2	2	4	Returning_Visitor	False	False	1.333333	0.095
0.0	10	627.500000	0.02	0.05	0.0	0.0	Feb	3	3	1	4	Returning_Visitor	True	False	62.750000	0.035

Appendix 3

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split

# Preparing the data with the new variable included
X = data.drop(['ProductRelated_Duration', 'Revenue'], axis=1)  # Exclude target and non-feature column
y = data['ProductRelated_Duration']

# Handling categorical variables via one-hot encoding
X = pd.get_dummies(X, drop_first=True)

# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training the Linear Regression model
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# Predicting on the test set
y_pred = lin_reg.predict(X_test)

# Evaluating the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("MSE (Linear Regression):", mse)
print("R2 (Linear Regression):", r2)
```

```
MSE (Linear Regression): 1282530.0165796638
R2 (Linear Regression): 0.7201080508007809
```

Appendix 4

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import cross_val_score
import numpy as np

# Initialising the Random Forest Regressor
rf_reg = RandomForestRegressor(random_state=42)

# Training the model using cross-validation
rf_cv_scores = cross_val_score(rf_reg, X_train, y_train, cv=10, scoring='r2')

# Average R2 score from cross-validation
average_r2_rf = np.mean(rf_cv_scores)

print("Average R2 Score (Random Forest):", average_r2_rf)
```

```
Average R2 Score (Random Forest): 0.9740173838000953
```

```
# Training the Random Forest model on the entire training set
rf_reg.fit(X_train, y_train)

# Predicting on the test set
y_pred_rf = rf_reg.predict(X_test)

# Evaluating the model
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

print("MSE (Random Forest):", mse_rf)
print("R2 (Random Forest):", r2_rf)
```

```
MSE (Random Forest): 609383.4951122614
R2 (Random Forest): 0.8670116628446105
```


Appendix 5

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import classification_report
import numpy as np


# Separating the target variable and features
X = data.drop('Revenue', axis=1)
y = data['Revenue']

# Identifying numerical and categorical columns
numerical_cols = X.select_dtypes(include=['int64', 'float64']).columns
categorical_cols = X.select_dtypes(include=['object', 'bool']).columns

# Creating a column transformer for preprocessing
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_cols),
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_cols)
    ])

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Appendix 6

```
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline as ImbPipeline  # Import from imblearn to correctly integrate SMOTE

# Creating the preprocessing and training pipeline with SMOTE
pipeline_smote = ImbPipeline([
    ('preprocessor', preprocessor),
    ('smote', SMOTE(random_state=42)),
    ('classifier', LogisticRegression(random_state=42, max_iter=1000))
])

# Cross-validation on the training set (after SMOTE)
cv_scores_smote = cross_val_score(pipeline_smote, X_train, y_train, cv=5, scoring='accuracy')

# Training the logistic regression model on the training set with SMOTE
pipeline_smote.fit(X_train, y_train)

# Predicting on the test set
y_pred_smote = pipeline_smote.predict(X_test)

# Confusion Matrix and Classification Report on the test set
conf_matrix_smote = confusion_matrix(y_test, y_pred_smote)
class_report_smote = classification_report(y_test, y_pred_smote)

# Printing the results
print("Cross-Validation Metrics on Training Set (with SMOTE):")
print(f"CV Accuracy: {np.mean(cv_scores_smote):.2f} ± {np.std(cv_scores_smote):.2f}")
print("\nConfusion Matrix on Test Set (with SMOTE):")
print(conf_matrix_smote)
print("\nClassification Report on Test Set (with SMOTE):")
print(class_report_smote)
```


Appendix 7

```
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline as ImbPipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix, classification_report

# Creating the preprocessing and training pipeline with SMOTE and Random Forest
pipeline_rf_smote = ImbPipeline([
    ('preprocessor', preprocessor),
    ('smote', SMOTE(random_state=42)),
    ('classifier', RandomForestClassifier(random_state=42))
])

# Cross-validation on the training set with SMOTE
cv_scores_rf_smote = cross_val_score(pipeline_rf_smote, X_train, y_train, cv=5, scoring='accuracy')

# Training the Random Forest model on the training set with SMOTE
pipeline_rf_smote.fit(X_train, y_train)

# Predicting on the test set
y_pred_rf_smote = pipeline_rf_smote.predict(X_test)

# Confusion Matrix and Classification Report on the test set with SMOTE
conf_matrix_rf_smote = confusion_matrix(y_test, y_pred_rf_smote)
class_report_rf_smote = classification_report(y_test, y_pred_rf_smote)

# Printing the results
print("Cross-Validation Metrics on Training Set with SMOTE (Random Forest):")
print(f"CV Accuracy: {np.mean(cv_scores_rf_smote):.2f} ± {np.std(cv_scores_rf_smote):.2f}")
print("\nConfusion Matrix on Test Set with SMOTE (Random Forest):")
print(conf_matrix_rf_smote)
print("\nClassification Report on Test Set with SMOTE (Random Forest):")
print(class_report_rf_smote)
```

```
Cross-Validation Metrics on Training Set with SMOTE (Random Forest):
CV Accuracy: 0.89 ± 0.01

Confusion Matrix on Test Set with SMOTE (Random Forest):
[[2890  226]
 [ 156  390]]
```

Appendix 8

```
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline as ImbPipeline

# Hyperparameter grid for KNN
param_grid_knn = {
    'classifier__n_neighbors': [5, 10, 15],
    'classifier__weights': ['uniform', 'distance'],
    'classifier__algorithm': ['auto', 'ball_tree', 'kd_tree'],
    'classifier__leaf_size': [20, 30, 40]  # Added leaf_size as an example additional parameter
}

# Create the pipeline with a KNN classifier
pipeline_knn = ImbPipeline([
    ('preprocessor', preprocessor),
    ('smote', SMOTE(random_state=42)),
    ('classifier', KNeighborsClassifier())
])

# GridSearchCV for KNN
grid_search_knn = GridSearchCV(pipeline_knn, param_grid=param_grid_knn, cv=3, scoring='accuracy', n_jobs=-1)

# Fit GridSearchCV
grid_search_knn.fit(X_train, y_train)

# Best hyperparameters and score for KNN
best_params_knn = grid_search_knn.best_params_
best_score_knn = grid_search_knn.best_score_

print("Best hyperparameters for KNN:")
print(best_params_knn)
print(f"Best cross-validation score: {best_score_knn:.2f}")

# Predict on the test set using the best model
y_pred_knn = grid_search_knn.predict(X_test)

# Confusion Matrix and Classification Report on the test set
conf_matrix_knn = confusion_matrix(y_test, y_pred_knn)
class_report_knn = classification_report(y_test, y_pred_knn)

print("\nClassification Report for Best KNN Model:")
```


Appendix 9

```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

# Hyperparameter grid for SVC
param_grid_svc = {
    'classifier__C': [0.1, 1, 10],
    'classifier__gamma': ['scale', 'auto'],
    'classifier__kernel': ['linear', 'rbf']
}

# Create the pipeline with an SVC classifier
pipeline_svc = ImbPipeline([
    ('preprocessor', preprocessor),
    ('smote', SMOTE(random_state=42)),
    ('classifier', SVC(random_state=42, probability=True))  # probability=True to enable predict_proba
])

# GridSearchCV for SVC
grid_search_svc = GridSearchCV(pipeline_svc, param_grid=param_grid_svc, cv=5, scoring='accuracy', n_jobs=-1)

# Fit GridSearchCV
grid_search_svc.fit(X_train, y_train)

# Best hyperparameters and score for SVC
best_params_svc = grid_search_svc.best_params_
best_score_svc = grid_search_svc.best_score_

print("Best hyperparameters for SVC:")
print(best_params_svc)
print(f"Best cross-validation score: {best_score_svc:.2f}")

# Predict on the test set using the best model
y_pred_svc = grid_search_svc.predict(X_test)

# Confusion Matrix and Classification Report on the test set
conf_matrix_svc = confusion_matrix(y_test, y_pred_svc)
class_report_svc = classification_report(y_test, y_pred_svc)

print("\nClassification Report for Best SVC Model:")
print(class_report_svc)

# Plotting the confusion matrix
plt.figure(figsize=(7, 4))
```

Appendix 10

```
# List of classifiers to apply
classifiers = {
    "Logistic Regression": LogisticRegression(random_state=42, max_iter=1000),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "KNN": KNeighborsClassifier(),
    "Naive Bayes": GaussianNB(),
    "SVC": SVC(probability=True, random_state=42),  # probability=True for ROC/AUC
    "Random Forest": RandomForestClassifier(random_state=42),
    "Gradient Boosting": GradientBoostingClassifier(random_state=42),
    "AdaBoost": AdaBoostClassifier(random_state=42),
    "XGB Boost": XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_state=42)
}

# Applying each classifier
for name, classifier in classifiers.items():
    # Creating the preprocessing and training pipeline with SMOTE and the current classifier
    pipeline = ImbPipeline([
        ('preprocessor', preprocessor),
        ('smote', SMOTE(random_state=42)),
        ('classifier', classifier)
    ])

    # Cross-validation on the training set with SMOTE
    cv_scores = cross_val_score(pipeline, X_train, y_train, cv=10, scoring='accuracy')

    # Training the model on the training set with SMOTE
    pipeline.fit(X_train, y_train)

    # Predicting on the test set
    y_pred = pipeline.predict(X_test)

    # Confusion Matrix and Classification Report on the test set
    conf_matrix = confusion_matrix(y_test, y_pred)
    class_report = classification_report(y_test, y_pred)

    # Printing the results
    print(f"{name} - Cross-Validation Metrics on Training Set with SMOTE:")
    print(f"CV Accuracy: {np.mean(cv_scores):.2f} ± {np.std(cv_scores):.2f}")
    print("\nConfusion Matrix on Test Set:")
    print(conf_matrix)
    print("\nClassification Report on Test Set:")
    print(class_report)
    print("-------------------------------------------------------\n")
```


```
----------------------------------------------------

Decision Tree - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.85 ± 0.01

Confusion Matrix on Test Set:
[[2790  326]
 [ 217  329]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.93      0.90      0.91      3116
        True       0.50      0.60      0.55       546

    accuracy                           0.85      3662
   macro avg       0.72      0.75      0.73      3662
weighted avg       0.86      0.85      0.86      3662

----------------------------------------------------

Naive Bayes - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.47 ± 0.03

Confusion Matrix on Test Set:
[[1254 1862]
 [  59  487]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.96      0.40      0.57      3116
        True       0.21      0.89      0.34       546

    accuracy                           0.48      3662
   macro avg       0.58      0.65      0.45      3662
weighted avg       0.84      0.48      0.53      3662

----------------------------------------------------

KNN - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.77 ± 0.01

Confusion Matrix on Test Set:
[[2436  680]
 [ 161  385]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.94      0.78      0.85      3116
        True       0.36      0.71      0.48       546

    accuracy                           0.77      3662
   macro avg       0.65      0.74      0.67      3662
weighted avg       0.85      0.77      0.80      3662

----------------------------------------------------

SVC - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.87 ± 0.01

Confusion Matrix on Test Set:
[[2801  315]
 [ 148  398]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.95      0.90      0.92      3116
        True       0.56      0.73      0.63       546

    accuracy                           0.87      3662
   macro avg       0.75      0.81      0.78      3662
weighted avg       0.89      0.87      0.88      3662

----------------------------------------------------

Gradient Boosting - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.89 ± 0.01

Confusion Matrix on Test Set:
[[2878  238]
 [ 144  402]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.95      0.92      0.94      3116
        True       0.63      0.74      0.68       546

    accuracy                           0.90      3662
   macro avg       0.79      0.83      0.81      3662
weighted avg       0.90      0.90      0.90      3662

----------------------------------------------------

AdaBoost - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.88 ± 0.01

Confusion Matrix on Test Set:
[[2817  299]
 [ 129  417]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.96      0.90      0.93      3116
        True       0.58      0.76      0.66       546

    accuracy                           0.88      3662
   macro avg       0.77      0.83      0.80      3662
weighted avg       0.90      0.88      0.89      3662

----------------------------------------------------

XGB Boost - Cross-Validation Metrics on Training Set with SMOTE:
CV Accuracy: 0.89 ± 0.01

Confusion Matrix on Test Set:
[[2945  171]
 [ 191  355]]

Classification Report on Test Set:
              precision    recall  f1-score   support

       False       0.94      0.95      0.94      3116
        True       0.67      0.65      0.66       546

    accuracy                           0.90      3662
   macro avg       0.81      0.80      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```

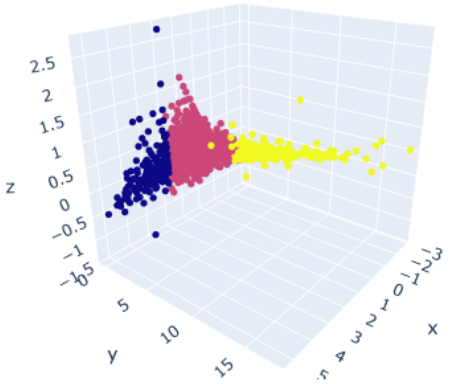
Appendix 11

```python
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import matplotlib.colors  # Importing the necessary module for custom colormap
import numpy as np

# Selecting the relevant features
features = ['Informational_Duration', 'BounceRates', 'ExitRates']
X = data[features]

# Standardising the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Applying PCA
pca = PCA()
X_pca = pca.fit_transform(X_scaled)

# Elbow method for determining the optimal number of clusters
inertia = []
K = range(1, 11)
for k in K:
    kmeans = KMeans(n_clusters=k, init='k-means++', n_init=10, random_state=42)
    kmeans.fit(X_pca)
    inertia.append(kmeans.inertia_)

# Plotting the Elbow Method
plt.figure(figsize=(10, 6))
plt.plot(K, inertia, 'bo-')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.title('Elbow Method For Optimal K')
plt.show()
```


Appendix 12

```
# Calculating silhouette score for 3 clusters
silhouette_score_3_clusters = silhouette_score(X_pca, kmeans.labels_)

print('The average silhouette score is :',silhouette_score_3_clusters)
```

3D Cluster Visualisation




```
The average silhouette score is : 0.7900786853834033
```

Appendix 13

```
Best hyperparameters for Decision Tree:
{'classifier__criterion': 'gini', 'classifier__max_depth': 10, 'classifier__min_samples_leaf': 1, 'classifier__min_samples_split': 10}
Best cross-validation score: 0.87

Classification Report for Best Decision Tree Model:
              precision    recall  f1-score   support

       False       0.95      0.89      0.92      3116
        True       0.54      0.71      0.61       546

    accuracy                           0.87      3662
   macro avg       0.74      0.80      0.77      3662
weighted avg       0.89      0.87      0.87      3662


Best hyperparameters for KNN:
{'classifier__algorithm': 'auto', 'classifier__leaf_size': 20, 'classifier__n_neighbors': 10, 'classifier__weights': 'uniform'}
Best cross-validation score: 0.77

Classification Report for Best KNN Model:
              precision    recall  f1-score   support

       False       0.94      0.79      0.86      3116
        True       0.37      0.72      0.49       546

    accuracy                           0.78      3662
   macro avg       0.66      0.75      0.67      3662
weighted avg       0.86      0.78      0.80      3662
```

```
Best hyperparameters for SVC:
{'classifier__C': 0.1, 'classifier__gamma': 'scale', 'classifier__kernel': 'linear'}
Best cross-validation score: 0.88

Classification Report for Best SVC Model:
              precision    recall  f1-score   support

       False       0.95      0.90      0.93      3116
        True       0.56      0.75      0.65       546

    accuracy                           0.88      3662
   macro avg       0.76      0.83      0.79      3662
weighted avg       0.90      0.88      0.88      3662
```

```
Best hyperparameters for GradientBoostingClassifier:
{'classifier__learning_rate': 0.1, 'classifier__max_depth': 10, 'classifier__min_samples_leaf': 1, 'classifier__min_samples_split': 4, 'classifier__n_estimators': 200, 'classifier__subsample': 0.8}
Best cross-validation score: 0.90

Classification Report for Best GradientBoostingClassifier Model:
              precision    recall  f1-score   support

       False       0.94      0.94      0.94      3116
        True       0.66      0.66      0.66       546

    accuracy                           0.90      3662
   macro avg       0.80      0.80      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```

```
Best hyperparameters for AdaBoostClassifier:
{'classifier__learning_rate': 1, 'classifier__n_estimators': 200}
Best cross-validation score: 0.89

Classification Report for Best AdaBoostClassifier Model:
              precision    recall  f1-score   support

       False       0.94      0.93      0.93      3116
        True       0.62      0.69      0.65       546

    accuracy                           0.89      3662
   macro avg       0.78      0.81      0.79      3662
weighted avg       0.90      0.89      0.89      3662
```

```
Best hyperparameters for XGBClassifier:
{'classifier__colsample_bytree': 0.8, 'classifier__learning_rate': 0.1, 'classifier__max_depth': 3, 'classifier__min_child_weight': 2, 'classifier__n_estimators': 200, 'classifier__subsample': 1}
Best cross-validation score: 0.90

Classification Report for Best XGBClassifier Model:
              precision    recall  f1-score   support

       False       0.95      0.93      0.94      3116
        True       0.65      0.71      0.68       546

    accuracy                           0.90      3662
   macro avg       0.80      0.82      0.81      3662
weighted avg       0.90      0.90      0.90      3662
```

```
Confusion Matrix for Stacked Model:
[[2932  184]
 [ 189  357]]

Classification Report for Stacked Model:
              precision    recall  f1-score   support

       False       0.94      0.94      0.94      3116
        True       0.66      0.65      0.66       546

    accuracy                           0.90      3662
   macro avg       0.80      0.80      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```

```
Confusion Matrix for Voting Model:
[[2892  224]
 [ 159  387]]

Classification Report for Voting Model:
              precision    recall  f1-score   support

       False       0.95      0.93      0.94      3116
        True       0.63      0.71      0.67       546

    accuracy                           0.90      3662
   macro avg       0.79      0.82      0.80      3662
weighted avg       0.90      0.90      0.90      3662
```