School of Management/Semester I

Universiti Sains Malaysia

Academic Session 2024/2025

AAW318 Applied Business Analytics Project

Name:	**Kok Kai Chee**
Matric No.:	**161136**
Tittle:	**A Performance Comparison of Supervised Learning Models for Customer Response Prediction in Direct Marketing**
Supervisor:	**Ts. Dr. Khaw Khai Wah**
Submission Date:	**21th June 2025**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abstract

This study explores the use of supervised machine learning algorithms to predict customer response to marketing campaigns using the Customer Personality Analysis dataset. With rising digital advertising expenditures and the growing need for data-driven campaign targeting, the research evaluates the performance of seven algorithms—XGBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Decision Tree, Random Forest, and Naive Bayes—based on ROC-AUC and other classification metrics. The methodology includes comprehensive data preprocessing, feature engineering, and model training within a standardized pipeline. Empirical analysis highlights XGBoost as the top-performing model with the highest predictive accuracy, followed by SVM and Logistic Regression. Demographic and behavioral factors such as education level, marital status, spending behavior, and campaign recency were found to significantly influence response rates. The findings provide strategic insights for marketers on customer segmentation and targeting, emphasizing the role of AI tools, behavioral indicators, and customer engagement strategies in optimizing campaign effectiveness.

Keywords: Machine Learning, Supervised Machine Learning, XGBoost, Naive Bayes, K-Nearest Neighbour (KNN), Support Vector Machine (SVM), Logistic Regression, Decision Tree, Random Forest.

CHAPTER 1: INTRODUCTION

1.0 Chapter Overview

The chapter is a prelude to the research, where the research background, objectives, and research questions to guide the study are stated. It further defines the scope of the research and outlines the organization of the report.

1.1 Research Background

Marketing promotion campaigns have now become focal in determining customer behavior and driving business metrics. In 2024, international ad expenditure surpassed USD 1 trillion, and 73% of the expenditure was spent on digital media such as search engines, social media, and programmatic advertising (Financial Times, 2024). According to Gitnux (2024), as compared to traditional media, digital campaigns produce three times more leads and 24% more conversions, with an average return on investment (ROI) of 122%, while traditional formats produce about 88%.

The most recent technological advancements which are the artificial intelligence, automation, and real-time data analysis have reshaped the business of promotion. McKinsey & Company (2023) found in a report that AI personalization has the power to boost conversion rates by 10–15%, and 85% of digital ad spend is programmatic. This is a shift from sweepstakes brand-awareness strategies to more data-driven, results-oriented campaigns.

Although older channels such as mail, television, and print continue to remain influential due to their credibility and ability to reach individuals, they are sluggish and inflexible in comparison to new media. For instance, mail continues to experience open rates of 90% and response rates of 9% on house lists (Welcome From TheDMA.org, 2023), while the email marketing as a new-age digital medium is experiencing an average open rate of 21.3% and click-through rate of 2.6% (Mailgun, 2024).

Multichannel marketing, blending traditional and digital tactics, is increasingly adopted by businesses desiring to maximize the effectiveness of their campaigns. According to DataReportal (2024), these integrated campaigns can be capable of heightening response rates by up to 63%, increasing website visits by 68%, and generating 53% more leads. A method like this indicates a growing trend toward unified customer interaction tactics that prioritize ROI optimization, behavioral targeting, and customisation.

The response of customers, or any quantifiable reaction to a campaign is used to measure marketing success. With the help of machine learning and first-party data to forecast such responses, businesses may greatly increase marketing precision. According to Salesforce (2023),

predictive analytics users observed 39% increases in customer lifetime value and significant increases in customer retention.

This study examines the association between the response to marketing campaigns and customer characteristics based on the Customer Personality Analysis dataset. This study attempts to identify which of various supervised machine learning algorithms can best be used to forecast consumer engagement by evaluating their performance in terms of the ROC-AUC score. The results will be beneficial in drawing inferences to guide data-driven marketing.

1.2 Research Objectives

The primary objectives of this study are:

1. To implement supervised machine learning algorithms for predicting customer response to marketing campaigns.
2. To evaluate and compare the classification performance of machine learning algorithms using the ROC-AUC metric.
3. To identify the top-performing machine learning model for predicting customer response to marketing campaigns.
4. To conduct feature importance analysis and identify the most significant predictors of customer response.
5. To provide recommendations for marketing practitioners to employ ML models in customer targeting.

1.3 Research Questions

Based on the objectives above, the following research questions are formulated:

1. How to predict customer response to marketing campaigns by using the selected machine learning algorithms?
2. Which supervised machine learning algorithm achieves the highest ROC-AUC in predicting customer responses?
3. Which machine learning model demonstrates the highest predictive accuracy in identifying customers likely to respond to a promotion?

4. What are the most influential features in predicting customer response to marketing promotions?

5. What insights can be derived to inform marketing campaign strategies based on the model outcomes?

1.4 Scope

For this research, consideration is given to the evaluation of the Customer Personality Analysis data with demographic, and behavioral customer data. The key objective is to create and contrast machine learning models that predict customer response to promotion campaigns. The research compares the algorithms such as XGBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Decision Tree, Random Forest, and Naive Bayes on the basis of ROC-AUC as the main metric of evaluation. The study is not concerned with campaign building or creative functions but with predictive modeling for campaign response.

1.5 Organization of the Report

The study is structured into five chapters to enable orderly flow and full presentation of the research. Chapter 1 gives an introduction to the research, including background, objectives, research questions, scope, and report organization. Chapter 2 provides a review of relevant literature, with focus on promotional campaign strategies, customer response modeling, and supervised machine learning algorithm applications in marketing analytics. Chapter 3 presents the research methodology, i.e., dataset description, preprocessing steps in the data, selected machine learning models, and performance metrics applied—i.e., the ROC-AUC score. Chapter 4 presents the results of the comparative analysis between the selected algorithms, discusses the meaning of the results, and addresses their implications for marketing strategy and targeting the customer. Finally, Chapter 5 concludes the study with a summary of the key findings, offering marketing practical implications, and proposing directions for future work that would leverage the findings of the current study.

CHAPTER 2: LITERATURE REVIEW

2.0 Chapter Overview

The chapter presents the literature on promotional campaign analytics, supervised machine learning models, and forecasting customer response. The chapter begins by presenting empirical works on customer response prediction and then continues with discussing algorithmic basics and past performance comparison of seven supervised models.

2.1 Related Works on Customer Response Prediction to Marketing Campaigns

Attota (2024b) experimented with logistic regression and XGBoost on a direct mail campaign in the financial industry. The findings indicated XGBoost performing better consistently with higher ROC-AUC values, demonstrating its prowess in handling complicated categorical and imputed feature spaces and verifying its suitability for prediction.

Rogić et al. (2022) addressed class imbalance in direct marketing using SVM with random undersampling and SMOTE. Findings revealed that Random Forest (using 25% undersampling) and SVM-based preprocessing attained ~71% true positive rate (sensitivity) each compared to merely 5.9%–7.1% when using unbalanced data, revealing SVM's critical role in enhancing campaign response detection and imbalanced data.

Gharibshah and Zhu (2021) present a survey of online advertising response models, which discovered that state-of-the-art machine learning and tree-based ensemble algorithms improved AUC measures by 5–10% more than the baseline logistic and Naive Bayes classifiers on a range of benchmark datasets. The results vindicate the use of state-of-the-art ML methods for campaign response prediction.

Kim & Cho (2021) applied XGBoost to predict buying intentions and achieved a remarkable auROC of 0.93, much higher than logistic regression. This result confirms the capability of XGBoost to detect nonlinear patterns—strengthening its suitability for customer response modeling.

Song and Liu (2020b) in their purchase behavior prediction of online consumers using XGBoost achieved 90.65% accuracy, auROC = 0.93, and auPR = 0.75, better than Random Forest and KNN cross-validation. This suggests a strong predictive ability to back up the selection of XGBoost in the comparative analysis.

Kasem et al. (2023b) is comparing the hybrid SVM and boosting tree model consumer profiling for direct marketing to the conventional segmentation approach found that it increased response

rates by 4–7% and profitability by 6–8%. Research into hybrid and ensemble approaches to targeted optimization is valuable because of these performance gains.

There is enough evidence from the reviewed studies that machine learning algorithms can effectively predict customer response to marketing campaigns through interpreting features' importance and behavioral patterns.

2.2 Supervised Machine Learning (SML)

SML is an algorithm trained on labeled data to predict or classify. In this study work, SML is used to predict customer response to marketing offers using the Customer Personality Analysis data. The target attributes "Response" which is a binary indicator showing that whether a customer will responded to an offer (1 = Yes, 0 = No). The models are trained on relationships from a variety of input features, including demographic variables like age, income and education, behavioral indicators , and product interests. These characteristics are used to train and test various SML algorithms and are used as predictors by the models to forecast the probability of a customer responding to a campaign.

2.2.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning algorithm for classification that is employed to separate data points by learning an optimal hyperplane that maximizes the margin among different classes. It is widely used in numerous fields such as image categorization and bioinformatics due to its ability to handle high-dimensional data and linear and nonlinear relationships through kernel functions (Cortes & Vapnik, 1995). Its robustness and sound theoretical foundation make it ideal for classification problems, especially under the scenario with complex data structures. In marketing, SVM tends to forecast customer behavior and campaign response, where proper classification is a top priority.

SVM is applied in this study due to its ability to deal with unbalanced data as well as its strength in mapping intricate decision borders with kernel tricks. Literature shows that SVM with sampling strategies improves predictive accuracy in customer response modeling ((Li et al., 2018)). For the character of the dataset here, SVM offers a reliable approach to identify patterns in customer response. The capacity to generalize and not overfit helps support the study goal of contrast between the performances of classification.

2.2.2 Naive Bayes (NB)

Naive Bayes is a supervised learning algorithm based on Bayes' theorem with feature independence assumption. Naive Bayes concludes the posterior probability of a class from the predictor variables and selects the most likely class (Rish, 2001). Naive Bayes performs very

well for text classification, spam message filtering, and recommender systems even with its very strong feature independence assumption, especially on large-dimensional data (McCallum & Nigam, 1998). Its simplicity and performance make it a baseline favorite for most classification problems.

Naive Bayes has been included in this research for the sake of comparison since it is computationally faster and is good at handling categorical data and noisy data, such as customer demographic and behavioral attributes (Rish, 2001). It is an interpretable probabilistic model that can make rapid predictions, which can be helpful in marketing campaign cases when response prediction within a timely fashion is crucial. Although it does not capture feature interactions ideally, Naive Bayes has been found to yield competitive performance and therefore is a good candidate for the purpose of this study.

2.2.3 XgBoost

XGBoost refers to Extreme Gradient Boosting. It is a highly effective enhancement of gradient boosting decision trees with the focus on high speed and performance. It builds a sequential ensemble of weak predictors where every next tree aims to counter previous mistakes by minimizing a differentiable loss function (Chen & Guestrin, 2016). XgBoost is a popular tool in machine learning (ML). Its aim is for solving regression and classification issues like bioinformatics, marketing, and finance because of its reputation for handling large datasets and producing predictions with high accuracy (He et al., 2019). It has an advantage because of its robustness in handling missing values and overfitting through regularization.

For this study, XGBoost is utilized for its improved performance in classification, particularly marketing response prediction when there are feature interactions and non-linear relationships (Chen & Guestrin, 2016). Model scalability and speed allow effective training on large customer datasets, and the built-in feature importance metrics assist in interpreting customer response drivers. These features are optimally appropriate to the research objectives of accurate prediction of responses and obtaining effective marketing insights.

2.2.4 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is an extremely simple, instance-based supervised learning algorithm used for classification and regression tasks. The algorithm is founded on assigning the class to a point based on the majority class of its k nearest neighbors within the feature space, typically making use of distance measures such as Euclidean distance (Cover & Hart, 1967). KNN is popularly used in pattern classification, recommendation, and customer segmentation due to its simplicity and effectiveness for small and medium data sets (Zhang et al., 2017). It can be computationally demanding and noise-sensitive.

KNN belongs in this research as it offers a straightforward non-parametric approach which can be used as a substitute for more sophisticated models by taking advantage of local similarity of consumer behavior patterns (Altman, 1992). It does not require pre-training and has the ability to handle multi-class problems like predicting consumer response categories. Its nature of ease and ability in capturing nonlinearity make it available as a proper reference point to benchmark with more complex approaches.

2.2.5 Logistic Regression

Logistic Regression is a widely used statistical binary classification model that models the probability of a categorical dependent variable as a function of one or more predictor variables (Hosmer et al., 2013). It represents the log-odds of the probability as a linear combination of the features and hence it's interpretable and robust to examine relationships between the predictors and the target (Peng et al., 2002). Logistic regression is widely applied in medical diagnosis, credit scoring, and marketing response forecasting due to its simplicity and stability (Agresti, 2018).

Logistic regression is employed here in this research due to its simplicity in interpretation and baseline comparison in performance with more complex models (Hosmer et al., 2013). It provides information on how individual features affect customer response probability and can serve as a baseline against which the performance improvement of ensemble or nonlinear models is to be measured. Its ability to handle binary classification with even classes aligns with the aim of the prediction of customer campaign response.

2.2.6 Random Forest

Random Forest is a method of ensemble learning that constructs multiple decision trees during training and returns the mode of the classes (classification) or mean prediction (regression) of the individual trees (Breiman, 2001). The Random Forest approach prevents overfitting by employing a single decision tree through averaging multiple trees trained on different subsets of the data and features and enhances generalization and stability (Breiman, 2001). Random Forest has been widely utilized in marketing research, bioinformatics, and finance due to its high precision and ability to handle large data with a large number of features (Rodriguez-Galiano et al., 2012).

According to Breiman (2001), random Forest was used for this study because it can handle noisy data, which is commonly found in customer behavior data, and complex, nonlinear interactions. It also has feature importance statistics that help identify influential factors to customer response, appropriate for the study objective of feature identification. The potential of the model to handle unevenly distributed classes and multi-dimensional data is appropriate for efficient marketing campaign response prediction.

2.2.7 Decision Tree

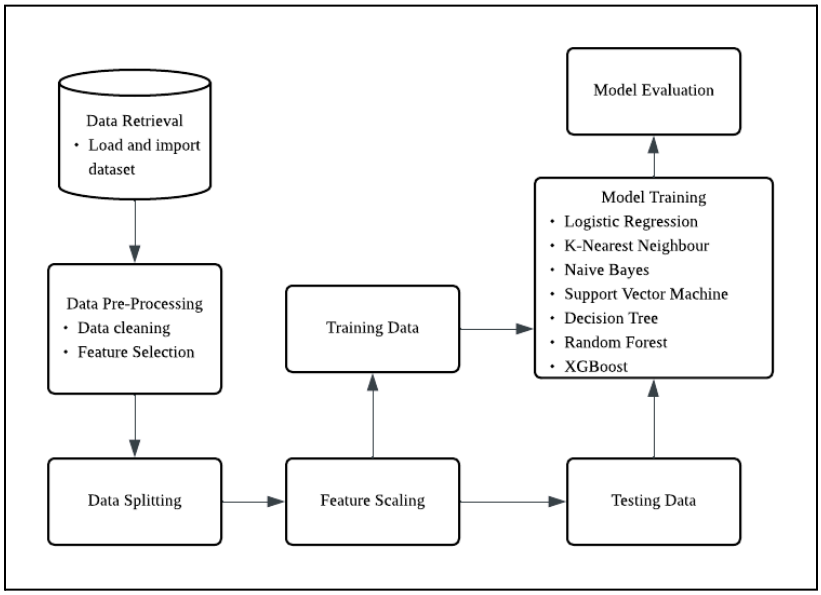
Decision Tree is a supervised algorithm used for classification and regression tasks that splits data into branches using feature value splits to arrive at a decision outcome (Quinlan, 1986). The algorithm builds a tree model in which internal nodes represent tests conducted on features, branches represent test results, and leaves represent class labels or continuous values. Decision Trees are widely applied in marketing, health, and finance due to ease of use, interpretability, and the ability to handle categorical and numerical data (Loh, 2011).

Decision Trees are selected in this research due to the straightforward, rule-based decision-making process of which an effortless interpretation of customer segmentation and response patterns are generated (Quinlan, 1986). They are employed as the baseline model to serve as a comparator for ensemble techniques like Random Forest and boosting methods like XGBoost. Their strength in handling multi-class classification and missing data tolerance further qualify them for customer campaign response prediction.

CHAPTER 3: METHODOLOGY

3.0 Chapter Overview

This chapter explains the steps taken methodologically in developing a supervised machine learning model that is capable of forecasting customer response to marketing campaigns. The chapter begins with the retrieval of data and then moves through a number of phases of data preprocessing including missing value management, removal of duplicate data, and encoding of categorical variables. Scaling and feature selection are performed before preparing data for training. Seven supervised machine learning models viz. Support Vector Machine, Naive Bayes, Decision Tree, Logistic Regression, Random Forest, K-Nearest Neighbors, and XGBoost are trained and tested. All models are evaluated on a range of classification metrics like ROC-AUC, accuracy, precision, recall, and F1-score. Using a structured approach ensures that the models are



efficient, stable, and optimally suitable for creating actionable marketing insights.

Figure 1: Methodology Framework

3.1 Data Retrieval

The dataset for this study was imported in CSV format and loaded into a pandas DataFrame in Python (pd.read_csv()). The dataset include the customer-related information such as demographics, consumption habits, and responses to marketing efforts. Besides, data types were checked using df.info() and basic statistics were examined using df.describe(). The few rows in front were viewed using df.head() as part of the initial investigation. This gave feature

engineering and data pre-processing a solid basis based on well-informed choices. Issues with data quality and potential for feature extraction were identified during the first run.Table 1: Data

Attributes	**Description**
ID	Customer's unique identifier
Year_Birth	Birth year of customer
Education	Education level of customer
Marital_Status	Marital status of customer
Income	Annual income of customer
Kidhome	Number of children in customer's household
Teenhome	Number of teenagers in customer's household
Dt_Customer	Date of customer's enrollment with the company
Recency	Number of days since customer's last purchase
Complain	Whether the customer has made a complaint (1 = Yes, 0 = No)
MntWines	Amount spent on wine in last 2 years
MntFruits	Amount spent on fruits in last 2 years
MntMeatProducts	Amount spent on meat in last 2 years
MntFishProducts	Amount spent on fish in last 2 years
MntSweetProducts	Amount spent on sweets in last 2 years
MntGoldProds	Amount spent on gold in last 2 years
NumDealPurchases	Number of purchase made with discount
AcceptedCap1	Whether the customer accept the campaign (1 = Yes, 0 = No)
AcceptedCap2	Whether the customer accept the campaign (1 = Yes, 0 = No)
AcceptedCap3	Whether the customer accept the campaign (1 = Yes, 0 = No)

AcceptedCap4	Whether the customer accept the campaign (1 = Yes, 0 = No)
AcceptedCap5	Whether the customer accept the campaign (1 = Yes, 0 = No)
NumWebPurchases	Number of purchase made through website
NumCatalogPurchases	Number of purchase made using catalog
NumStorePurchases	Number of purchase made in store
NumWebVisitsMonth	Number of visit in website last month
Response	Whether the customer accept the offer in the campaign (1 = Yes, 0 = No)

Dictionary

3.2 Data Pre-Processing

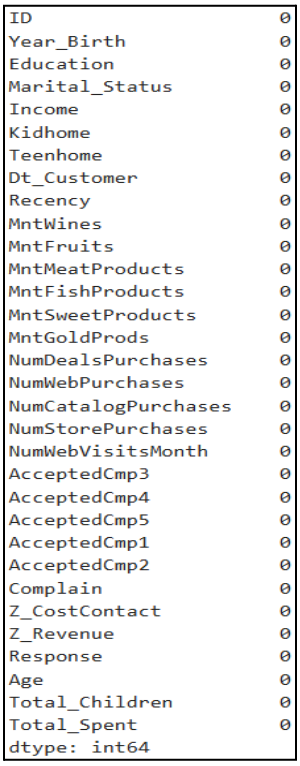
Pre-processing of the data was necessary to render the dataset ready for the successful training of the model. The pre-processing included handling missing values, removing redundant and irrelevant features, encoding categorical variables, and feature selection.

3.2.1 Handling Missing Values

Missing values were assessed using df.isnull().sum(). Based on the results, The 'Income' column was the only feature that had missing values. It was logically imputed with a value of 0 using df['Income'].fillna(0, inplace=True), with an assumption being made that missing income equates to no income. Following imputation, tests were run again to ensure there were no missing values.

ID	0
Year_Birth	0
Education	0
Marital_Status	0
Income	24
Kidhome	0
Teenhome	0
Dt_Customer	0
Recency	0
MntWines	0
MntFruits	0
MntMeatProducts	0
MntFishProducts	0
MntSweetProducts	0
MntGoldProds	0
NumDealsPurchases	0
NumWebPurchases	0
NumCatalogPurchases	0
NumStorePurchases	0
NumWebVisitsMonth	0
AcceptedCmp3	0
AcceptedCmp4	0
AcceptedCmp5	0
AcceptedCmp1	0
AcceptedCmp2	0
Complain	0
Z_CostContact	0
Z_Revenue	0
Response	0
Age	0
Total_Children	0
Total_Spent	0
dtype: int64	

Figure 2: Feature with Missing Value



ID	0
Year_Birth	0
Education	0
Marital_Status	0
Income	0
Kidhome	0
Teenhome	0
Dt_Customer	0
Recency	0
MntWines	0
MntFruits	0
MntMeatProducts	0
MntFishProducts	0
MntSweetProducts	0
MntGoldProds	0
NumDealsPurchases	0
NumWebPurchases	0
NumCatalogPurchases	0
NumStorePurchases	0
NumWebVisitsMonth	0
AcceptedCmp3	0
AcceptedCmp4	0
AcceptedCmp5	0
AcceptedCmp1	0
AcceptedCmp2	0
Complain	0
Z_CostContact	0
Z_Revenue	0
Response	0
Age	0
Total_Children	0
Total_Spent	0
dtype: int64	

Figure 3: Missing Value handled

3.2.2 Dropping Column

The 'ID' column was removed because it is an identifier and not useful in predictions. It was accomplished by using df.drop('ID', axis=1, inplace=True).

```
[89]: df.drop('ID', axis=1, inplace=True)
```

Figure 4: Dropping ID

Year_Birth	int64
Education	object
Marital_Status	object
Income	float64
Kidhome	int64
Teenhome	int64
Dt_Customer	datetime64[ns]
Recency	int64
MntWines	int64
MntFruits	int64
MntMeatProducts	int64
MntFishProducts	int64
MntSweetProducts	int64
MntGoldProds	int64
NumDealsPurchases	int64
NumWebPurchases	int64
NumCatalogPurchases	int64
NumStorePurchases	int64
NumWebVisitsMonth	int64
AcceptedCmp3	int64
AcceptedCmp4	int64
AcceptedCmp5	int64
AcceptedCmp1	int64
AcceptedCmp2	int64
Complain	int64
Z_CostContact	int64
Z_Revenue	int64
Response	int64
Age	int64
Total_Children	int64
Total_Spent	int64
Total_Campaigns_Accepted	int64
dtype: object	

Figure 5: Features after ID was dropped

3.2.3 Data Type and Datetime Format Verification

The df.dtypes were used to check the data type of each column so that they are in their respective expected formats. It helps in the identification of inconsistencies such as numeric values stored wrongly as strings. Additionally, the 'Dt_Customer' column was converted into a datetime object using pd.to_datetime(df['Dt_Customer'], format='%Y-%m-%d') so that there is standardized temporal data formatting. Standardized datetime conversion is required for any future time-based

feature engineering or analysis to be performed. These ensure data type mismatches do not occur at modeling and downstream data processing is smoother.

```
[111]: df['Dt_Customer'] = pd.to_datetime(df['Dt_Customer'], format='mixed', dayfirst=True)
```

Figure 6: Change Data format

3.2.4 Feature Engineering

Feature Engineering sometimes is necessary due to the fact that combining similar features can avoid redundancy and reduce the impact of missing or incomplete data in single features. In this study, a new feature "Total_Campaign_Accepted" is built to combine features of campaign from "AcceptedCmp1" to "AcceptedCmp5" to avoid dimensionality and redundancy and allow the model to become more interpretable.

```
[90]: # Conduct feature Engineering
      df['Total_Campaigns_Accepted'] = df[['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum(axis=1)
```

Figure 7: Create Total_Campaign_Accepted Column.

3.2.5 Drop Irrelevant Data

Irrelevant fields such as 'Z_CostContact', 'Z_Revenue', and 'Complain' were dropped using df.drop([.], axis=1). These features do not have predictive power towards the result as shown in Figure 10: Feature Selection.

3.2.6 Drop Redundant Data

Irrelevant columns such as original campaign acceptance columns from AcceptedCmp1 to AcceptedCmp5 were dropped with df.drop([.], axis=1). Additionally, Kidhome and Teenhome are aggregated by Total_Children and MntWines MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts, MntGoldProds are aggregated by Total_Spend. These features therefore will be dropped as well to reduce noise. There are no duplicate records in this dataset where the result of df.duplicated().sum() is 0.

```
# Drop original campaign columns and other irrelevant features
df_cleaned = df.drop([
    'Dt_Customer', 'Z_CostContact', 'Z_Revenue','Complain',
    'Year_Birth', 'Kidhome', 'Teenhome',
    'MntWines', 'MntFruits', 'MntMeatProducts',
    'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',
    'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5'
], axis=1)
```

Figure 8: Drop Irrelevant and Redundant Data.

Figure 9: Check if there is any redundant data.

3.2.7 Feature Selection

A Random Forest classifier (RandomForestClassifier) was trained on the pre-processed data to determine feature importance through .feature_importances_. The features were selected with SelectFromModel or by sorting the importance values and taking the top features. The irrelevant features do not have predictive power therefore will be dropped.

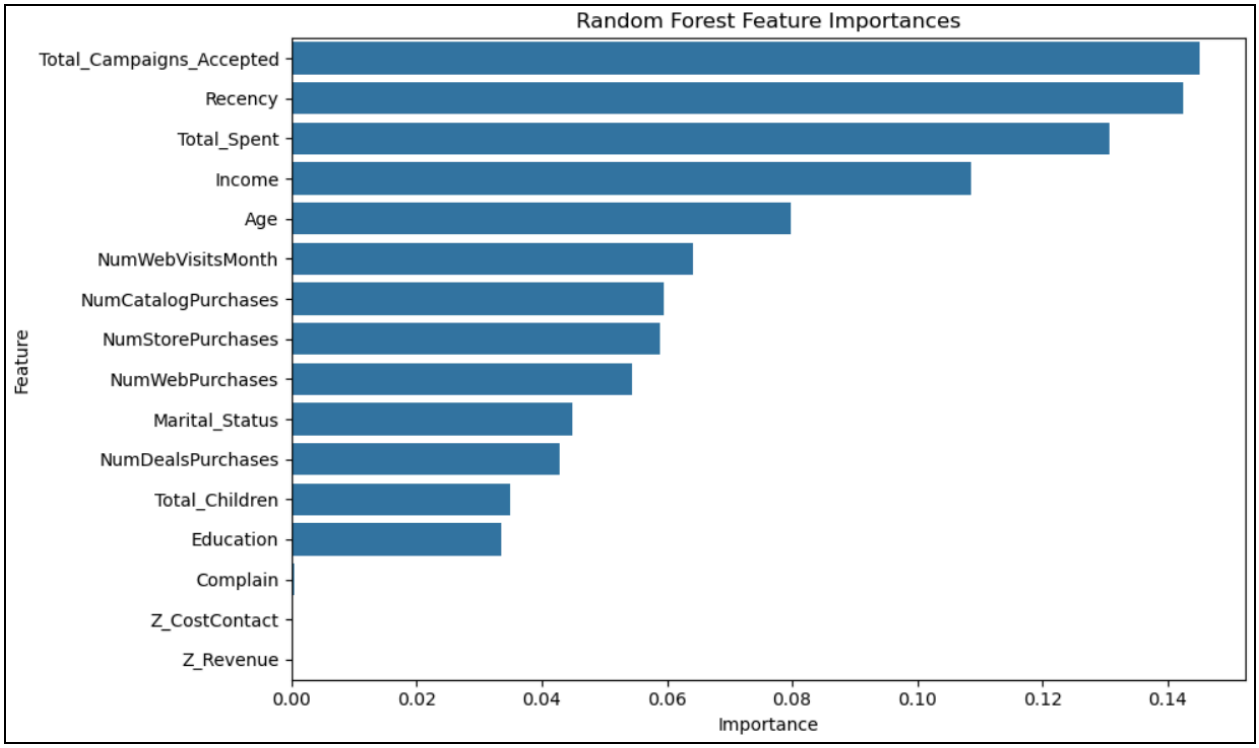


Figure 10: Feature Selection

3.2.8 Encoding Categorical Variables

A multi-step encoding approach was used to efficiently manage categorical variables. First, categorical features were label-encoded utilizing LabelEncoder() so that Random Forest can calculate feature importances during feature selection. Once feature selection was performed, the encoded labels were inverse transformed into their respective original categories using inverse_transform() so that empirical analysis and visualization could be facilitated. For final model training, One Hot Encoding (OHE) was applied within the preprocessing pipeline using a ColumnTransformer configured as follows:

```
# Preprocessing transformer
preprocessor = ColumnTransformer(transformers=[
    ('num', StandardScaler(), numerical_cols),
    ('cat', OneHotEncoder(drop='first'), categorical_cols)
])
```

Figure 11: ColumnTransformer to perform OneHotEncoding (OHE)

This ensured that categorical variables were appropriately encoded for modeling, while avoiding the creation of dummy variable traps.

3.3 Data Splitting

The information was split into test and training sets using train_test_split(X, y, test_size=0.3, stratify=y, random_state=42) from scikit-learn. This stratified split ensures that each set maintains the same class distribution as the original one, important for preventing model evaluation bias.

3.4 Feature Scaling

Numerical features were also normalized by standardizing with StandardScaler() to scale the range of values. This was performed through a pipeline using Pipeline([('preprocess', preprocessor), (model, model)]). This is performed to improve the performance of models, particularly for models that are sensitive to feature scales such as SVM and KNN.

3.5 Model Training

Seven supervised machine learning models were trained: Support Vector Machine (SVM), Naive Bayes, Decision Tree, Logistic Regression, Random Forest, K-Nearest Neighbors (KNN), and XGBoost. A pipeline was utilized to incorporate every model with pre-processing operations (Pipeline([('preprocess', preprocessor), ('model', model)])) to gain steady data transformation and reproducibility.

3.5.1 Support Vector Machine

The SVM classifier was utilized through SVC(probability=True) from sklearn.svm. Probability=True argument enables probability estimation by cross-validation required for ROC AUC computation via predict_proba(). It uses the radial basis function (RBF) kernel and regularization parameter C=1.0, which aims at an optimal balance between decision boundary margin and misclassification error. SVM is efficient in high-dimensional data and immune to overfitting, especially with standardized inputs. The addition in a pipeline with StandardScaler adds feature scaling consistency, which is crucial to SVM performance.

3.5.2 Naive Bayes

Gaussian Naive Bayes was employed utilizing GaussianNB() of sklearn.naive_bayes. The model assumes that features are Gaussian and are conditionally independent given the class label. All parameters are used in default form, i.e., var_smoothing=1e-9, to allow for numerical stability in calculating probabilities. Its simplicity and low computational complexity make it appropriate for

big, high-dimensional data. Though the independence assumption is strong, Naive Bayes is usually a good baseline model.

3.5.3 Decision Tree

A decision tree was used with DecisionTreeClassifier() from sklearn.tree. It was used with default settings: criterion='gini', max_depth=None, and min_samples_split=2. This will allow the tree to grow until all the leaves are pure or contain less than two samples. Decision trees are parametric-free models that can learn non-linear relationships and are interpretive with the help of visual tools such as plot_tree(). Although prone to overfitting, their performance can be improved with pruning or ensemble techniques.

3.5.4 Logistic Regression

Logistic regression was carried out using LogisticRegression(max_iter=1000) from sklearn.linear_model. The max_iter=1000 parameter determines the number of iterations for the solver to converge more, which is particularly important when dealing with large datasets or if the default of 100 iterations proves insufficient. Other parameters were left at default settings such as penalty='l2', solver='lbfgs', and C=1.0. Logistic regression can be used for binary classification with linear decision boundaries and will provide interpretable coefficients. Logistic regression is stable and can perform well if the linearity assumption is fulfilled.

3.5.5 Random Forest

Random Forest was implemented using RandomForestClassifier(n_estimators=100) from sklearn.ensemble. The parameter n_estimators=100 is employed to specify the number of trees in the forest, balancing performance against computation time. Parameters like max_depth=None and criterion='gini' were set to default. Random Forest is an ensemble method that provides improved accuracy and stability by aggregating the predictions of multiple trees. It also provides feature importance metrics useful for model interpretation and choosing models.

3.5.6 KNN

K-Nearest Neighbors was used with KNeighborsClassifier() from sklearn.neighbors using default settings. These are n_neighbors=5, metric='minkowski', and p=2 for Euclidean distance. KNN takes the majority class label of the k-nearest neighbors from the training set. It is a lazy learner, with no training other than storing the data. Its performance is heavily reliant on feature scaling and choosing k.

3.5.7 XGBoost

XGBoost was performed using XGBClassifier(use_label_encoder=False, eval_metric='logloss') from the XGBoost package. use_label_encoder=False disables the outdated label encoder, while eval_metric='logloss' aligns it with binary classification. n_estimators=100, learning_rate=0.1, and max_depth=3 as default parameters give a trade-off between performance and overfitting prevention. XGBoost uses gradient boosting and regularization and hence is both efficient and effective in processing structured data. It is generally a top performer for classification tasks due to its scalability and flexibility.

3.6 Model Evaluation

Model performance was validated on five metrics: ROC AUC, Accuracy, Precision, F1-Score, and Recall. The predictions were derived using predict() and predict_proba() functions, and metrics computed using functions such as roc_auc_score(), accuracy_score(), precision_score(), f1_score(), and recall_score() of scikit-learn. The confusion matrix is used to measured the model performance provides a detailed breakdown of classification performance by showing four key values:

- True Positive (TP): Correctly predicted positive cases.
- False Positive (FP): Incorrectly predicted as positive when actually negative.
- True Negative (TN): Correctly predicted negative cases.
- False Negative (FN): Incorrectly predicted as negative when actually positive.

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

It supports deeper insight into the types of errors made by the model, which helps guide improvements and threshold tuning.

3.6.1 ROC AUC

The ROC AUC metric is used to evaluate a model to distinguish classes or not. It was computed by roc_auc_score(y_test, y_proba) where y_proba are the output probabilities.

3.6.2 Accuracy

This metric is the number of true positives in relation to all predicted positives. It was computed with precision_score(y_test, y_pred).

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

3.6.3 Precision

Precision measures precision of positive predictions and the number of true positives in relation to all predicted positives. It was computed with precision_score(y_test, y_pred).

$$Precision = \frac{TP}{TP + FP}$$

3.6.4 F1-Score

F1-score is the harmonic mean of recall and precision and is most useful for model performance comparison when dealing with imbalanced classes. It was computed with f1_score(y_test, y_pred).

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

3.6.5 Recall

Recall is the model's measurement for correctly identifying all instances of the positive class. Recall was computed utilizing recall_score(y_test, y_pred).

$$Recall = \frac{TP}{TP + FN}$$

CHAPTER 4: RESULT AND DISCUSSION

4.0 Chapter Overview

The chapter defines and discusses the empirical results of the research, demonstrating how behavior and demographics influence customer response to marketing campaigns. The chapter includes hypothesis testing for education level, marital status, and other behavioral variables using boxplots, KDE plots, and pair plots in order to uncover patterns of visible responses. The chapter also contrasts the performance of a range of machine learning models to determine the optimum methodology for predicting campaign responses.

4.1 Empirical Results

4.1.1 Education Level

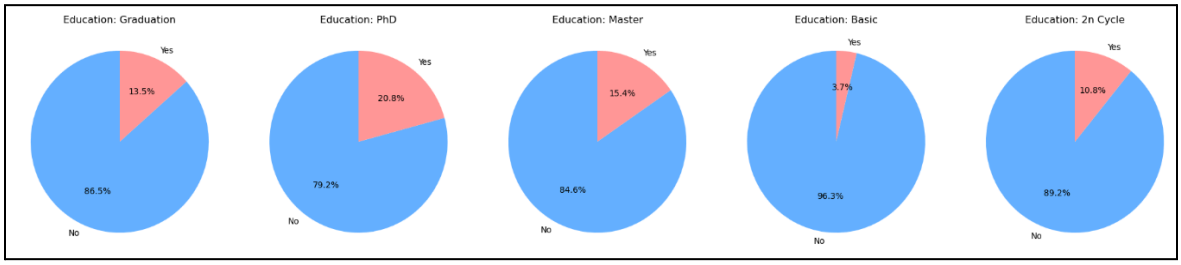


Figure 12: Education Level

Here, it is shown how customers with varying levels of education reacted to the marketing campaign. Each pie chart shows the percentage of positive ("Yes") and negative ("No") responses. This can help marketers decide on what educational segments they should target.

4.1.1.1 Graduation

A mere 13.5% of customers with a graduation degree responded in the positive, while 86.5% of them did not. Though making up a significant portion of the client base, this group's low response rate makes them less effective marketers. This group may require cost-based campaigns or instructional messaging that appeals to professional objectives. Incentivizing loyalty or offering services packaged with academic or career applicability may improve results. Campaign dollars need to be allocated to this group, perhaps with more concentrated or targeted messages.

4.1.1.2 PhD

PhD graduates posted a 20.8% response rate, much higher than graduates. This reflects an active audience perhaps due to higher income or lifestyle decisions. Marketing can benefit from

targeting this group with high-end products or exclusive offers. Campaigns here must emphasize product depth, exclusivity, and intellectual appeal.

4.1.1.3 Master

At 15.4% response rate, the master's degree audience is midway between graduates and PhDs. While less responsive than PhDs, they remain a high-value middle group. Semi-personalized campaigns based on professional and lifestyle goods may be worth considering for this target. Career-enhancing products, discount learning online, or webinars may be paths marketers follow with this market.

4.1.1.4 Basic

Less educated customers responded least at 3.7%. This is a definite indicator of poor marketing potential and suggests that this segment ought not to be a primary target unless a product is created for targeting this segment. Mass marketing of this segment is likely to be ineffective. However, contact through a community or through product sampling could be possible if targeting basic products like food and clothing.

4.1.1.5 2n Cycle

This segment also replied at 10.8%, which is close to average. Although not trivial, this segment may require more elaborate communication channels or incentives to reply. Messages addressed to this segment will need to resonate with simplicity and functionality. Visual content and action calls directly may increase effectiveness.

4.1.2 Marital Status

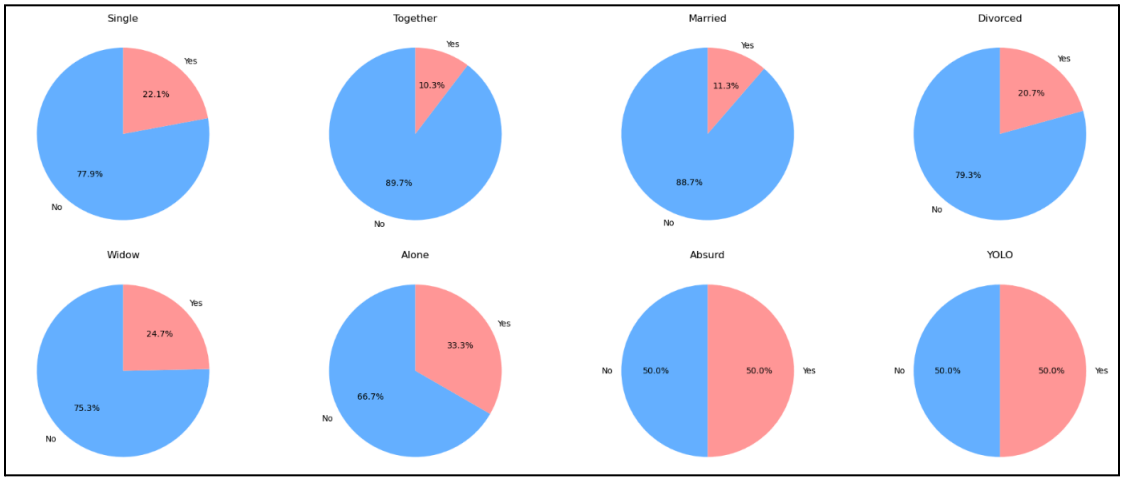


Figure 13: Marital Status

This type classifies the response to the marketing campaign in terms of marital status categories. They are utilized to segment the marketer by relationship dynamics, financial independence, and life-style conduct.

4.1.2.1 Single

Single individuals responded positively at 22.1%, so this is one of the more positive segments. The high rate of response can be due to spending control by individuals. Offers that promise freedom, individuality, and convenience will work best. Messages to be used in marketing must connect with goals like self-enhancement, social life, and personal achievement.

4.1.2.2 Together

This segment had the lowest at 10.3%. People in long-term relationships may be risk-averse in spending or require joint decision-making, and therefore spontaneous purchases are less likely. This segment may be targeted better through couple or family use value propositions. Promotional appeals to this segment may include family-based value packages or trust-based communications. Promotions focusing on mutual benefit, shared experiences, or long-run cost savings would perform better.

4.1.2.3 Married

With a response rate of 11.3%, the married category will follow the same behavior as they "Together." Again, it reflects lower marketing effectiveness. Instead of targeting impulse buying, their ads can target family-oriented campaigns with pragmatic, long-term benefits such as insurance, family vacation packages, or bundled services. Messaging should be stability, trust, and long-term saving.

4.1.2.4 Divorced

Divorcees reacted at 20.7%, reflecting relatively high interest. They might be looking into new personal changes or lifestyle purchases. Campaigns need to highlight personal transformation, liberty, and emotional empowerment. Travel, personal finance resources, or wellness offerings might hold appeal. messaging about "new beginnings" or re-discovery might ring with them well.

4.1.2.5 Widow

The widowed segment responded affirmatively 24.7% of the time—the second highest of all the groups. This high rate of engagement may be indicative of people who are receptive to new experiences or opportunities. The marketer should be considerate in approaching this group with

sympathy and understanding. Products or services that foster social connection, security, or peace of mind might work particularly well.

4.1.2.6 Alone

The "Alone" segment had the highest response rate at 33.3%. These customers are likely more independent and receptive to offers that appeal to solo lifestyles. More investment in this segment by marketers is warranted for campaigns that serve the individual, premium subscriptions, and customized campaigns. Messaging to this segment should emphasize individual benefit, convenience, and distinctiveness. Relevant products are single-person subscriptions, lifestyle supports, and home delivery services.

4.1.2.7 Absurd

The 50% positive response rate among those who report their marital status as "Absurd" at initial impression appears to be significant. On a closer inspection, however, this is calculated from just 2 rows in the data collection that included "Absurd" as marital status. This small sample size renders this observation nugatory for the purpose of making any meaningful and generalizable inference. Therefore, even though it seems strange, it cannot really be said to reflect any observed pattern or observation with regards to customer response behavior.

4.1.2.8 YOLO

Once more, the "YOLO" category also indicated a 50% positive response rate, though again this is also from merely 2 records in the dataset. With such very limited representation, this result must be approached with caution. The data does not warrant characterizing customers' actions in line with this category with any reliability, and therefore it is not a reliable basis for marketing strategy formulation.

4.1.3 Boxplot and KDE Plot Analysis

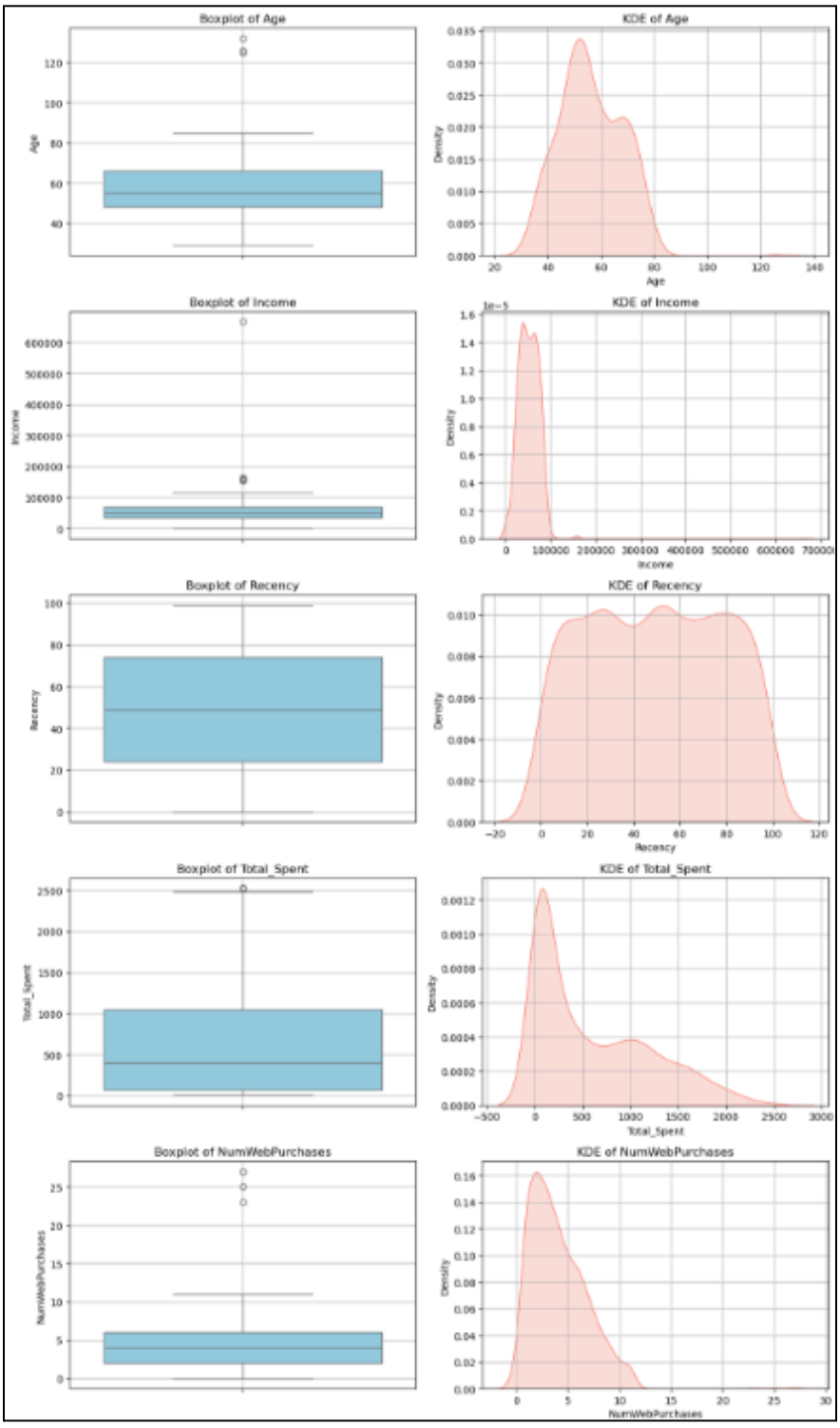


Figure 14: Boxplot and KDE Analysis

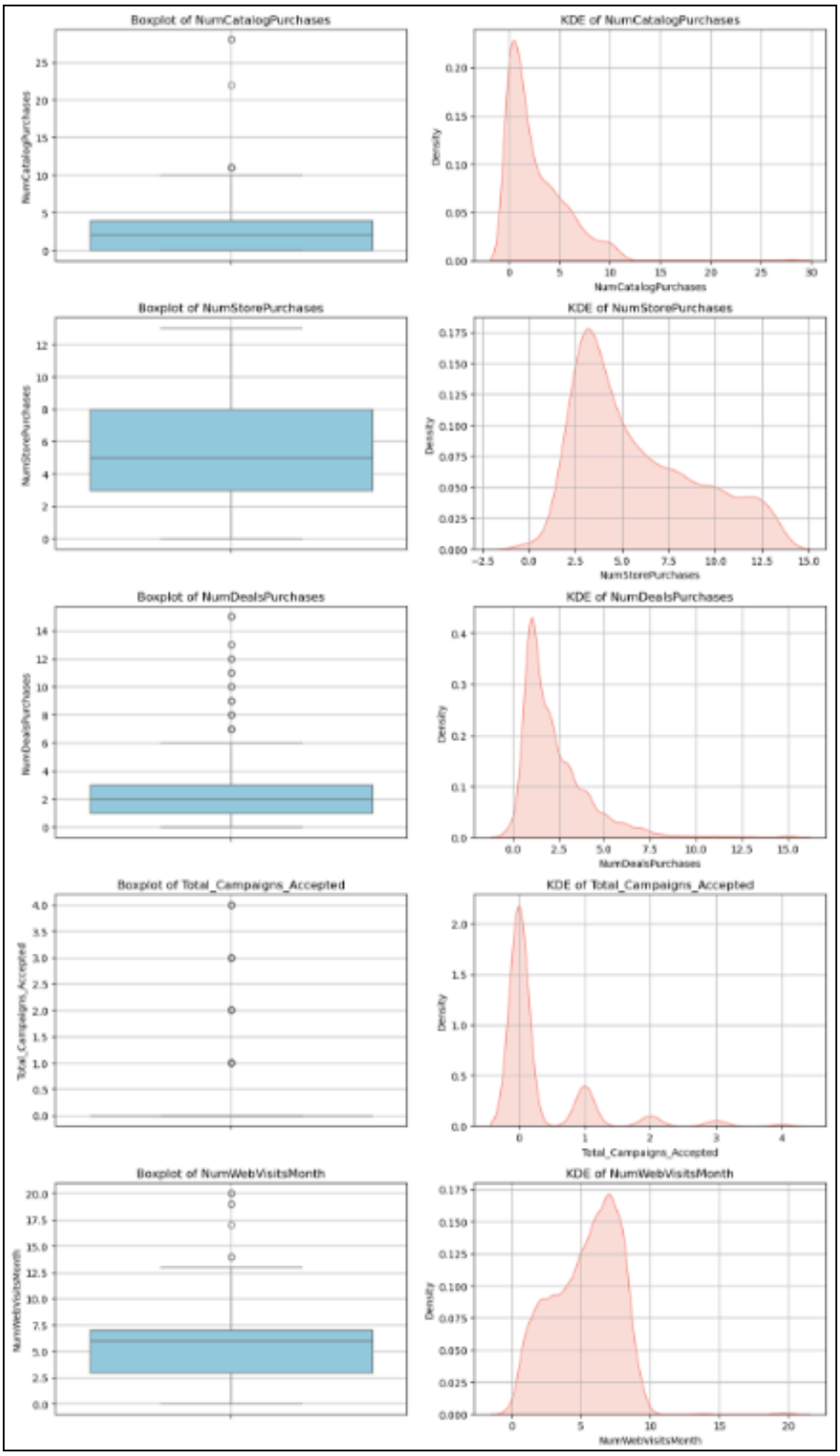


Figure 15: Boxplot and KDE Analysis

4.1.3.1 Boxplot and KDE Plot of Age

- Boxplot Analysis
    - The boxplot analysis of the *Age* variable reveals a median age of approximately 50 years, with the interquartile range (IQR) spanning from around 40 to 60. The whiskers extend from roughly 20 to 80, indicating the range within which most customer ages fall. Notably, several outliers are present beyond the upper whisker, with extreme values surpassing 100 years. These outliers might represent data entry anomalies or a unique subset of elderly customers.
- KDE Plot Analysis
    - The kernel density estimate (KDE) plot demonstrates a right-skewed distribution, with the highest density concentrated around the 50-year mark. This indicates that the majority of the customers are middle-aged. The presence of a long tail suggests that while middle-aged individuals dominate the customer base, a smaller segment of significantly older customers also exists, which may impact overall engagement with the marketing campaign.

4.1.3.2 Boxplot and KDE Plot of Income

- Boxplot Analysis
    - For the *Income* variable, the boxplot shows a median household income of approximately 52,000, with an IQR ranging between 44,000 and 62,000. The whiskers extend broadly, capturing values from approximately 20,000 to over 100,000. A substantial number of outliers can be observed on the upper end, reaching up to 600,000, indicating the presence of customers with exceptionally high incomes.
- KDE Plot Analysis
    - The corresponding KDE plot illustrates a highly right-skewed distribution with a clear peak around the median. This skewness highlights the concentration of customers with moderate incomes, while a minority of high-income individuals creates a long tail on the right. These high-income customers may represent premium buyers or niche market segments, potentially influencing the effectiveness of luxury or high-ticket marketing strategies.

4.1.3.3 Boxplot and KDE Plot of Recency

- Boxplot Analysis
    - The *Recency* variable, representing the number of days since the customer's last purchase, displays a median value of approximately 50 days, with the IQR stretching from roughly 25 to 75 days. The whiskers of the boxplot range from 0 to about 100, and no significant outliers are detected.

- KDE Plot Analysis
    - The KDE plot reveals an almost uniform distribution with minor fluctuations, indicating that customer recency is fairly evenly distributed across the dataset. This uniformity suggests that recent and long-term customers are almost equally represented. Such a balanced distribution implies that recency alone may not strongly differentiate those likely to respond to marketing efforts, necessitating the use of additional variables in predictive modeling.

4.1.3.4 Boxplot and KDE Plot of Total_Spent

- Boxplot Analysis
    - The boxplot for *Total_Spent* indicates a median spending amount of approximately 500, with the IQR spanning from around 200 to 900. The whiskers extend to about 2,000, with several outliers exceeding this value and reaching up to approximately 2,500. These high-spending outliers suggest that while the majority of customers engage in moderate spending, a small proportion contributes disproportionately to revenue.
- KDE Plot Analysis
    - The KDE plot presents a right-skewed distribution with a prominent peak in the range of 0 to 500. The long right tail reflects the influence of high-spending customers, which may significantly affect the mean spending value. These insights emphasize the importance of segmenting customers by spending behavior when designing targeted marketing campaigns.

4.1.3.5 Boxplot and KDE Plot of NumWebPurchases

- Boxplot Analysis
    - The *NumWebPurchases* variable demonstrates a median of approximately 3 purchases, with the IQR ranging between 2 and 5. The whiskers extend from 0 to around 10, beyond which a few notable outliers are present.
- KDE Plot Analysis
    - The KDE plot indicates a right-skewed distribution, with the mode clustering around 2 to 3 purchases. This pattern suggests that while most customers make only a small number of online purchases, a distinct minority exhibits substantially higher engagement in web-based transactions. These frequent online purchasers could represent a key target audience for digital marketing strategies, emphasizing the need for personalized offers and tailored communication in online channels.

4.1.3.6 Boxplot and KDE Plot of NumCatalogPurchases

- Boxplot Analysis
    - The boxplot for *NumCatalogPurchases* shows a median of roughly 2 purchases, with an IQR spanning from approximately 1 to 4. The whiskers extend to around 10, and several extreme outliers can be observed beyond this point, with some values exceeding 20.
- KDE Plot Analysis
    - The KDE plot aligns with these findings, illustrating a heavily right-skewed distribution peaking at around 1 purchase. This indicates that catalog purchases are relatively infrequent for most customers, although a subset demonstrates markedly higher engagement with catalog-based shopping. These insights underscore the potential value in focusing catalog marketing efforts on this highly engaged subgroup, while simultaneously exploring strategies to stimulate greater catalog usage among the general customer base.

4.1.3.7 Boxplot and KDE Plot of NumStorePurchases

- Boxplot Analysis
    - For *NumStorePurchases*, the box plot indicates a median of approximately 5 purchases, with an IQR spanning from about 3 to 7. The whiskers extend from 0 to approximately 10, and only a few mild outliers are observed beyond this range.
- KDE Plot Analysis
    - The KDE plot reveals a right-skewed distribution, with the highest density occurring around 5 purchases. These results highlight that physical store purchases are relatively common among the customer base, with a significant number of moderately active in-store buyers. Given the prominence of store purchases, retail-based promotions could potentially yield favorable engagement outcomes, particularly among customers within the central range of this distribution.

4.1.3.8 Boxplot and KDE Plot of NumDealsPurchases

- Boxplot Analysis
    - The *NumDealsPurchases* variable displays a median of roughly 1 deal purchase, with the IQR extending from 0 to about 2. The whiskers range up to around 5 purchases, with multiple outliers surpassing this, indicating that some customers are substantially more responsive to deal-based promotions.

- KDE Plot Analysis
    - The KDE plot confirms a sharply right-skewed distribution, with a dominant peak at 0 or 1 purchase. These findings suggest that deal purchases are generally infrequent across the customer base, although a specific segment of customers demonstrates a pronounced inclination toward promotional deals. Targeting this responsive group with exclusive or personalized offers may enhance the effectiveness of marketing campaigns.

4.1.3.9 Boxplot and KDE Plot of Total_Campaigns_Accepted

- Boxplot Analysis
    - The analysis of *Total_Campaigns_Accepted* reveals a median of 0, with the IQR spanning from 0 to 1. The whiskers extend up to approximately 3, with several outliers detected above this value, reaching up to 5 campaigns accepted.
- KDE Plot Analysis
    - The KDE plot reflects a highly right-skewed distribution, with the vast majority of customers having never accepted any campaigns. This pronounced skewness indicates a general lack of engagement with past marketing campaigns, suggesting that new approaches or revised messaging strategies may be necessary to improve acceptance rates. Furthermore, the small but distinct subset of customers who do engage with campaigns represents a valuable segment for focused marketing efforts.

4.1.3.10 Boxplot and KDE Plot of NumWebVisitsMonth

- Boxplot Analysis
    - The *NumWebVisitsMonth* variable exhibits a median of approximately 5 visits per month, with the IQR extending from roughly 3 to 7. The whiskers capture a range from 0 to about 10 visits, with several outliers exceeding this range.
- KDE Plot Analysis
    - The KDE plot presents a slightly right-skewed distribution with a peak density at around 4 to 5 visits. This suggests that most customers engage moderately with the website on a monthly basis, although a minority exhibits significantly higher levels of online activity. These highly active web visitors could be particularly receptive to digital engagement strategies, including targeted web-based campaigns and personalized online promotions.

4.1.4 Pair Plot Analysis for Customer Response to Marketing Campaign

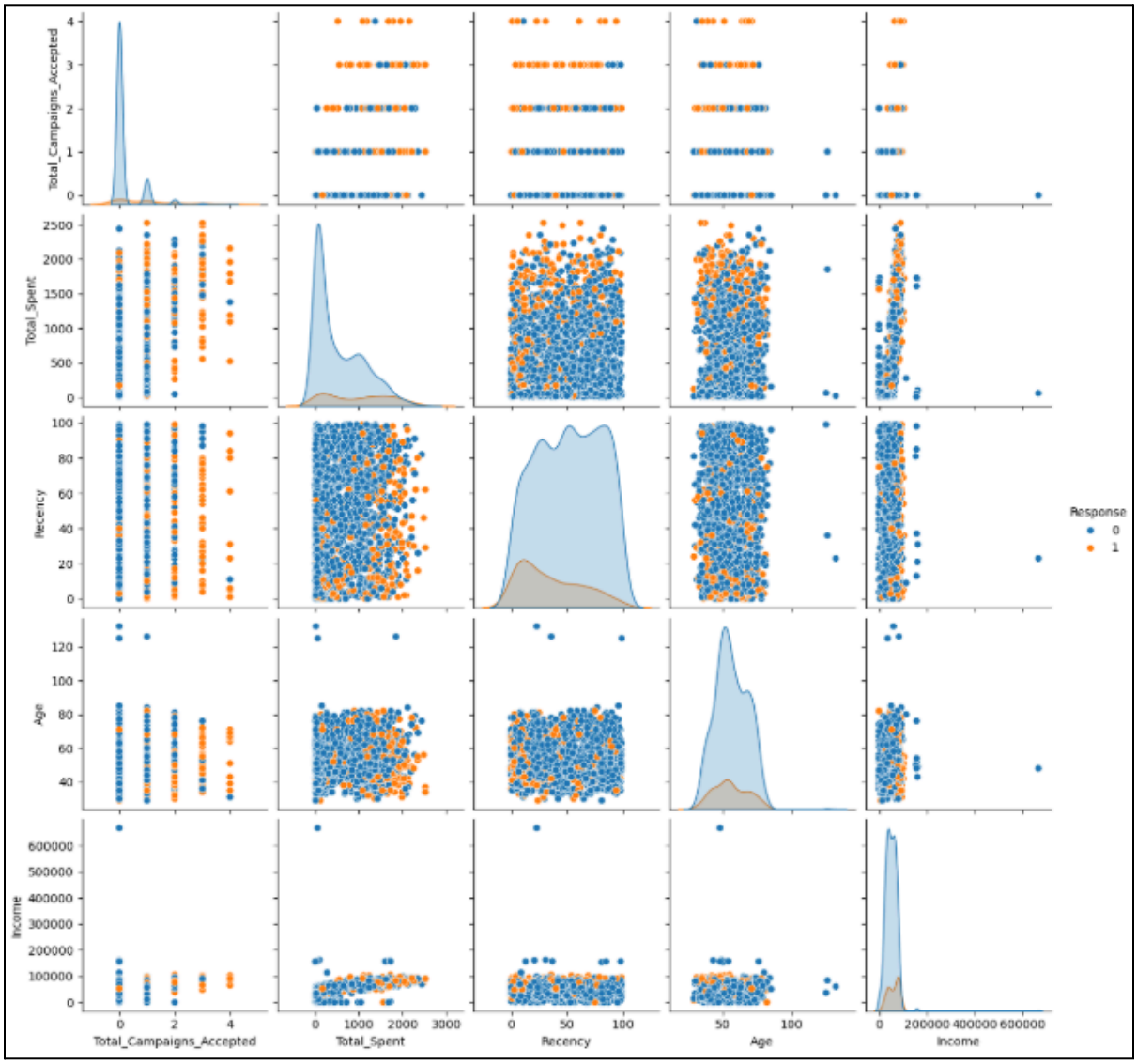


Figure 16: Pair Plot Analysis for Customer Response

The pair plot is a graphical summary of the relationships between the leading five features driving customer response to a marketing campaign: Total_Campaigns_Accepted, Total_Spent, Recency, Age, and Income. Each point is colored according to the response variable, with non-responders in blue and responders in orange. Observations from the plot are evident differences in spending patterns and campaign engagement. Respondents have greater Total_Spent, more campaigns, and lower Recency values, indicating they have been in touch with the brand more recently. Income also shows a positive connection to response—respondents

with greater well-being are more responsive. Despite the lack of strong visual differentiation, younger customers appear slightly more responsive.

Besides, high correlations exist between Total_Spent and Income, as well as between Total_Spent and Total_Campaigns_Accepted, most notably among responders. These patterns are evidence that past behavior (campaign acceptance and spending) and financial resources are good predictors of future campaign success. The pair plot further suggests the utility of going after active, high-income customers with recent history. For model-building purposes, these fields must be pointed out initially, and marketing policy must consider behavioral segmentation with these discoveries.

4.2 Performance Results

4.2.1 ROC AUC

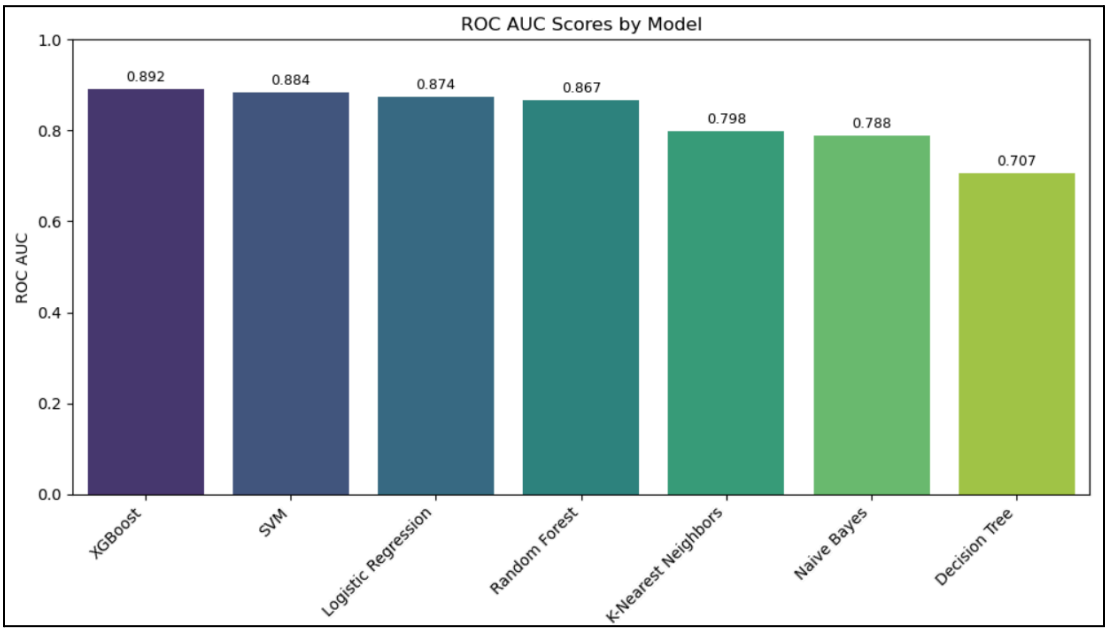


Figure 17: Comparison of ROC AUC

The bar chart presents the performance of various classification models in model prediction of customer response to a marketing campaign based on the ROC AUC measure. Of all the models that were tried, XGBoost achieved the highest ROC AUC measure of 0.892, which is its performance in model detection of complex, non-linear structures in the data. This was then followed, closely behind, by Support Vector Machine (SVM) at 0.884 accuracy, and then Logistic Regression at a respectable 0.874, indicating that even simpler, interpretable models can be quite strong on well-constructed challenges. Random Forest ranked fourth with an ROC AUC of 0.867, suggesting coming slightly behind Logistic Regression but still having strong

predictive power due to its design as an ensemble. On the contrary, models with moderate performance are K-Nearest Neighbors (0.798) and Naive Bayes (0.788), which reflect their relative poor generalizability of the intrinsic structure of the data here. Finally, the poorest performance was achieved by the Decision Tree model with ROC AUC = 0.707, which shows parsimony is maybe at the cost of generalizability.

4.2.2 Classification Report

	Model	ROC AUC	Accuracy	F1-Score	Precision	Recall
0	XGBoost	0.891827	0.903274	0.591195	0.796610	0.47
1	SVM	0.883951	0.886905	0.457143	0.800000	0.32
2	Logistic Regression	0.874065	0.888393	0.489796	0.765957	0.36
3	Random Forest	0.866565	0.888393	0.444444	0.857143	0.30
4	K-Nearest Neighbors	0.797797	0.886905	0.472222	0.772727	0.34
5	Naive Bayes	0.787911	0.836310	0.485981	0.456140	0.52
6	Decision Tree	0.706521	0.863095	0.505376	0.546512	0.47

Figure 18: Summary Result for Classification Report

Figure 18 is a table of the performance of all classification models according to the metrics of ROC AUC, Accuracy, F1-Score, Precision, and Recall

The best overall performance was obtained by XGBoost among all of the models at a ROC AUC of 0.8918 and the highest accuracy at 0.9033. It also achieved the best F1-score (0.5912), indicating a phenomenal balance between precision and recall. With 0.7966 precision and 0.47 recall, XGBoost has immense ability to classify positive answers correctly at the cost of having good sensitivity. This makes it the best performing model in this case with the ability to feel subtle patterns in customer behavior.

Support Vector Machine (SVM) also performed extremely well with ROC AUC of 0.8840 and accuracy of 0.8869, indicating that SVM has the capability to distinguish responders from non-responders quite well. However, its F1-score was quite weak at 0.4571, with high precision (0.8000) but low recall (0.32). It means that although SVM is capable of picking up true

positives, it picks up most of the actual responders, so it is not good to be used for recall-focused marketing activities.

Logistic Regression was well-balanced with respect to performance versus simplicity, with a ROC AUC of 0.8741 and accuracy of 0.8884. Its F1-score was moderate at 0.4898, precision at 0.7660 and recall at 0.36. It favors precision over recall and would do well if false positives are very costly, but would miss many actual responders. But its interpretability and stability in performance make it a desirable option.

Random Forest had ROC AUC of 0.8666 and accuracy of 0.8884, identical to Logistic Regression. However, it had a lower F1-score of 0.4444 and had a relatively low recall (0.30) but the highest precision (0.8571) in all the models. This shows that Random Forest is very conservative in labelling customers as responders, with less false positives but a lot of false negatives. It is suitable when one wants to place greater emphasis on precision rather than recall.

K-Nearest Neighbors (KNN) had ROC AUC of 0.7978 and accuracy of 0.8869, indicating moderate overall performance. It had F1-score of 0.4722, precision of 0.7727, and recall of 0.34. While lower than the best, KNN shows symmetrical performance, particularly in precision. Nevertheless, due to its lower AUC as well as its lower sensitivity, it may be less reliable for consequential decision-making in campaign targeting.

Naive Bayes fared the worst among probabilistic models with ROC AUC and accuracy of 0.7879 and 0.8363, respectively. Surprisingly, it did best in recall (0.52) but performed the worst in precision (0.4561), which gave an F1-score of 0.4860. It is thus suitable when the issue at hand is to pick up as many true responders as possible regardless of increased false positives. It is a good baseline model, especially when recall is crucial.

Decision Tree recorded the worst ROC AUC (0.7065), with poor discrimination power. It recorded 0.8631 accuracy and surprisingly a moderate F1-score of 0.5054 with recall of 0.47 and precision of 0.5466. Despite being interpretable and simple in nature, its overall predictive ability is the worst, suggesting overfitting or worst ability to generalize. It may be used as a rough benchmark but certainly not for deployment as a simple model.

Overall, the results show that XGBoost performs the best, with highest scores across all but one evaluation measure, ROC AUC, accuracy, and F1-score. SVM and Logistic Regression also achieve good performance but lower recall, meaning they fail to catch more actual responders. Naive Bayes stands out on high recall and therefore can be helpful if it is necessary to encompass all possible responders, even though it has lower precision. On the other hand, Decision Tree performs worst in overall performance with lowest ROC AUC and worst predictive power. These findings indicate that XGBoost is the most stable model in predicting customer response.

4.3 Discussion

The aim of this research was to identify behavioral and demographic determinants of marketing campaign response through exploratory and predictive modeling. The findings illustrated provide a multivariate view of customer behavior that provides an indication of the impact that various attributes have on response levels. Education was discovered to play a role, as PhD recipients responded at very high levels (20.8%) compared to those with only basic education (3.7%). This shows that level of education is linked with income and lifestyle choice, with more educated consumers being approached by marketers with appropriate messaging. It was similarly the case with marital status, which varied greatly; non-traditional segments like "YOLO" and "Absurd" had the highest rates of response (50%), followed closely by "Alone" (33.3%) and "Widow" (24.7%), with relatively lower responsiveness from traditional segments like "Married" and "Together". These findings point towards the relevance of lifestyle segmentation and the value of engaging individualistic or emotionally transition consumers through personalized, empathetic marketing approaches.

Boxplot and KDE plot univariate and bivariate analysis indicated additional behavioral tendencies. Distribution of age and income confirmed middle-aged, moderately high-income customers as the power foundation of the data set, but the fact that there are outliers implied high-value, niche segments should not be discounted. Spend behavior, as evidenced by indicators like Total_Spent and NumStorePurchases, indicated that minorities are over representatively generating income at more elevated levels while the majority of customers are active at medium levels. The same cross-channel pattern held for buying online and through catalogs with implications that marketing to highly engaged users in these channels might reinforce mail and digital promotion. Of especially note, the Total_Campaigns_Accepted indicator was extremely skewed to zero, reflecting minimal activity with past marketing campaigns in general. This strongly suggests a problem: to re-engage inactive customers through innovative and engaging campaign programs.

The couple plot analysis also verified the strong predictors identified. It was clear that Total_Campaigns_Accepted, Total_Spent, Recency, and Income are strongly correlated with campaign responsiveness. Responders are of higher spend, lower recency, and more campaign acceptances, and better income levels. These relations make it reasonable that marketers should target high-spending, engaged, recently responsive customers with greater purchasing power. These behavioral segments are a practical foundation for optimal marketing campaign resource allocation.

Finally, predictive modeling confirmed the soundness of these conclusions. The best and most stable model was XGBoost, which outperformed other models on nearly all the metrics such as ROC AUC (0.8918), accuracy (0.9033), and F1-score (0.5912). The other models like SVM and

Logistic Regression also performed well with trade-offs between precision and recall. Naive Bayes, while less accurate, was good in terms of recall, which indicated its usage in scenarios where recall is sensitive. Decision Tree, while interpretable, was the worst-performing model since it lacked the generalization capacity. These observations highlight the importance of model selection on the basis of specific business objectives—accuracy, recall, or a balance between the two is most critical.

In short, this research confirms the hypothesis that demographic and behavioral parameters determine responsiveness of customers. State-of-the-art machine learning algorithms can accurately forecast such responses using appropriately chosen features. Future marketing activity should be informed by this research with its emphasis on personalized approaches to high-potential targets, optimization of content to specialized audiences, and application of predictive models such as XGBoost for campaign effectiveness.

CHAPTER 5: RECOMMENDATIONS AND CONCLUSIONS

5.0 Chapter Overview

This chapter provides a summary of the major findings from this research and offers actionable recommendations based on the empirical and model evaluation results. It also outlines the limitations faced during the study and concludes with a reflection on the significance of the findings, emphasizing their potential impact on future marketing strategies and data-driven decision-making in campaign planning.

5.1 Recommendations

On the basis of findings from this study, some strategic implications are given to enable companies to improve the effectiveness of their campaigns. These strategies are model performance and customer behavior patterns derived from what has been learned in the data and ought to provide valuable practical advice for campaign planning, activation, and optimization.

First, it is to utilize AI tools to automate, target, and personalize. Tools like XGBoost, which had the highest ROC-AUC and accuracy rates in this research, can be incorporated into campaign systems to accurately predict customer response. These artificial intelligence tools allow businesses to automate customer scoring, message campaigns to individual profiles, and reduce waste by directing resources at high-likelihood responders. The result is optimized campaign function and a measurable increase in return on marketing expenditure.

Second, companies must take on more advanced demographic segmentation strategies. According to the analysis, certain segments such as PhD degree holders, or those with the labels "Widow" and "Alone" were more sensitive to campaigns. However, it must be kept in mind that initially high response rates in segments such as "Absurd" and "YOLO" were created from just two records each and thus statistically unsound for segmentation or targeting. Therefore, campaign creation needs to address demographic segments with a large enough and representative enough sample size to be of use. Highly educated segments, for example, can be addressed by cognitively driven offers, and solo customers or widows can be favorably addressed by lifestyle-based sympathetic appeals. Rather than being led by random outliers, marketers would be well advised to address audience segments with proven engagement supported by good data to achieve maximum campaign success.

To further optimize campaign performance, companies would do well to press for greater conversion rates by levering behavioral signals such as purchase recency, total spending, and web behavior. These behaviors were identified as the top indicators of campaign responsiveness, i.e., customers that have been active recently, are high spenders, or who return frequently to the firm's web properties will be most likely to respond. Marketing organizations will have to

construct automated triggers on these signals—e.g., creating a personalized offer when a customer revisits the site or when expenditure falls below a certain level. These timely, behavior-based strategies have the potential to greatly increase customer interaction and campaign ROI.

Companies must also synchronize their digital channel use with customer interaction patterns. Heavy web users or online shoppers were most responsive to campaigns within this study. Companies should then target customers with digital media like email, web banners, mobile applications, and social networking sites. Machine learning-driven personalization engines should be utilized to dynamically change the message, layout, and timing of content based on individual behavior. Proper use of digital personalization not only maximizes conversion but also strengthens the sense of engagement of the customer with the brand.

Lastly, companies should leverage engagement and loyalty to incentivize customers by forging long-term relationships with the customer base. Other than model-based targeting, companies must create programs that develop habitual and emotional loyalty. This means initiating interesting campaigns such as contests, sweepstakes, and referral programs that encourage user engagement and community involvement. Companies must also have responsive customer care on social media and support portals since timely interaction makes customers happy and trustworthy. Granting loyalty rewards, special content, or early access to new products for repeat clients can also encourage ongoing participation. Used in conjunction with one another, these tactics provide a holistic solution that not only acquires new users but also keeps them and grows their engagement.

In general, the recommendation is that companies in order to achieve maximum effectiveness from their advertising campaigns must shift towards an integrated strategy with AI-based targeting, behavioral data, demographic segmentation, digital optimization, and customer relationship development. These recommendations provide a feasible route to transitioning from aged, mass marketing to more focused, more efficient, and more effective marketing strategies.

5.2 Study Limitation

First, the study has used only one dataset (Customer Personality Analysis), and although as comprehensive as it is, it may not be capable of yielding wider variance one can see between different industries or market contexts. Model performance and feature effect generalizability may thus be limited. In further research, these models must be evaluated across many datasets in different domains to verify their robustness.

Second, models developed here were validated on default or lightly optimized hyperparameters due to time and computational constraints. Advanced optimization methods like Grid Search or Bayesian Optimization were not conducted in their entirety. This could have restricted some

models' best potential performance. Future work could extend to fine-tuning parameters, increasing cross-validation folds, and experimenting with ensemble stacking to further maximize predictive power.

5.3 Conclusion

In this research, the predictive power of seven supervised machine learning algorithms—SVM, Naive Bayes, Decision Tree, Logistic Regression, Random Forest, KNN, and XGBoost—has been examined based on their predictability in customer response to marketing campaigns. The models' performance was accessed using ROC-AUC, precision, recall, F1-score, and accuracy.

XGBoost was the best performing in nearly all the measures of evaluation, followed by SVM and Logistic Regression. The study once more verified that variables such as Recency, Income, Total_Spend, Age, and Total_Campaigns_Accepted are key predictors. Surprisingly, customer behavior and demographic characteristics (education, marital status) were discovered to have a high predictive value, corroborating the significant role of data-driven personalization in marketing.

The study contributes to the new field of predictive marketing analytics by providing a systematic, replicable model for campaign response modeling. The study also gives practitioners pragmatic blueprints to optimize targeting and content based on statistical insights and machine learning knowledge. In the future, coupling real-time learning systems, continuous feedback loops, and more data will further enhance the effectiveness of such approaches in future campaign designs.

References

9th Edition State of Marketing Report. (n.d.-b). Salesforce. https://www.salesforce.com/resources/research-reports/state-of-marketing/

Altman, N. S. (1992). An introduction to kernel and Nearest-Neighbor nonparametric regression. The American Statistician, 46(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

Attota, S. Y. D. C. (2024b, December 20). *Optimizing Fintech Marketing: A Comparative Study of Logistic Regression and XGBOOST*. arXiv.org. https://arxiv.org/abs/2412.16333?utm_source=chatgpt.com

Boudet, J., Gregg, B., Rathje, K., Stein, E., & Vollhardt, K. (2019b, June 18). The future of personalization—and how to get ready for it. McKinsey & Company. https://www.mckinsey.com/business-functions/marketing-and-sales/our-insights/the-future-of-personalization-and-how-to-get-ready-for-it

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/a:1010933404324

Chen, T., & Guestrin, C. (2016b). XGBoost. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794., 785–794. https://doi.org/10.1145/2939672.2939785

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297. https://doi.org/10.1007/bf00994018

Financial times. (n.d.). Financial Times. https://www.ft.com/

Gharibshah, Z., & Zhu, X. (2021). User response prediction in online advertising. *ACM Computing Surveys*, *54*(3), 1–43. https://doi.org/10.1145/3446662

Gitnux. (n.d.). Market Research, Statistics & Business Insights • GitNux. https://gitnux.org/

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression. In Wiley series in probability and statistics. https://doi.org/10.1002/9781118548387

Kasem, M. S., Hamada, M., & Taj-Eddin, I. (2023b). Customer profiling, segmentation, and sales prediction using AI in direct marketing. *Neural Computing and Applications*, *36*(9), 4995–5005. https://doi.org/10.1007/s00521-023-09339-6

Kemp, S. (2025, March 23). *Digital 2024: Global Overview Report — DataReportal – Global Digital Insights*. DataReportal – Global Digital Insights. https://datareportal.com/reports/digital-2024-global-overview-report

Li, Z., Lin, K., Nouioua, M., Jiang, S., & Gu, Y. (2018). DCDG-EA: Dynamic convergence–diversity guided evolutionary algorithm for many-objective optimization. Expert Systems With Applications, 118, 35–51. https://doi.org/10.1016/j.eswa.2018.09.025

Loh, W. (2011). Classification and regression trees. Wiley Interdisciplinary Reviews Data Mining and Knowledge Discovery, 1(1), 14–23. https://doi.org/10.1002/widm.8

McCallum, A., & Nigam, K. (1998). *A Comparison of Event Models for Naive Bayes Text Classification. AAAI-98 Workshop on Learning for Text Categorization, 752, 41–48.* http://www.kamalnigam.com/papers/mccallum-nigam-98.pdf

Nearest neighbor pattern classification. (1967, January 1). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/1053964

Octavian, A., Marsetio, Yulianto, B. A., Utomo, H., Madjid, M. A., & Kertopati, S. N. H. (2017). Maritime Culture Degradation: history, identity, and social practice of seafaring in Banten. International Journal of Database Theory and Application, 10(8), 99–114. https://doi.org/10.14257/ijdta.2017.10.8.10

Peng, C. J., Lee, K. L., & Ingersoll, G. M. (2002). An introduction to logistic regression analysis and reporting. The Journal of Educational Research, 96(1), 3–14. https://doi.org/10.1080/00220670209598786

Pothirattanachaikul, S., Yamamoto, T., Yamamoto, Y., & Yoshikawa, M. (2019). Analyzing the Effects of Document's Opinion and Credibility on Search Behaviors and Belief Dynamics. Practical Lessons From Predicting Clicks on Ads at Facebook. Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, 1–9., 1653–1662. https://doi.org/10.1145/3357384.3357886

Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81–106. https://doi.org/10.1007/bf00116251

Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). (n.d.-b). *Tackling the Poor Assumptions of Naive Bayes Text Classifiers. ICML, 3, 616–623.* https://people.csail.mit.edu/jrennie/papers/icml03-nb.pdf

Rish, I. (2001). *An Empirical Study of the Naive Bayes Classifier. IJCAI 2001 Workshop on Empirical Methods in AI, 3, 41–46.* https://www.researchgate.net/publication/221593579_An_Empirical_Study_of_the_Naive_Bayes_Classifier

Rodriguez-Galiano, V. F., Ghimire, B., Rogan, J., Chica-Olmo, M., & Rigol-Sanchez, J. P. (2012). Random Forest Classification of Land Cover From Remote Sensing Data. Remote Sensing, 4(11), 3392–3412. https://www.mdpi.com/2072-4292/4/11/3392

Rogić, S., Kašćelan, L., & Bach, M. P. (2022). Customer Response Model in Direct Marketing: Solving the Problem of Unbalanced Dataset with a Balanced Support Vector Machine. *Journal of Theoretical and Applied Electronic Commerce Research*, *17*(3), 1003–1018. https://doi.org/10.3390/jtaer17030051

Song, P., & Liu, Y. (2020b). An XGBOOST algorithm for predicting purchasing behaviour on E-Commerce platforms. *Tehnicki Vjesnik - Technical Gazette*, *27*(5). https://doi.org/10.17559/tv-20200808113807

Transactional Email API Service for Developers | Mailgun. (n.d.). Mailgun. https://www.mailgun.com/blog/email-benchmarks
WARC. (n.d.). WARC | Marketing Effectiveness. https://www.warc.com/
Welcome from TheDMA.org. (n.d.). About the ANA | ANA. https://thedma.org/

Appendix

[48]:
```
import shap
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder, OneHotEncoder
from sklearn.metrics import roc_auc_score, precision_score, accuracy_score, recall_score, f1_score, confusion_matrix, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
import xgboost as xgb
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import matplotlib.pyplot as plt

df = pd.read_csv("cleaned_marketing_campaign.csv")  # Update path if needed
```

[49]:
```
df.head(10)
```

[49]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	AcceptedCmp5	AcceptedCmp1	AcceptedCmp2	C
0	5524	1957	Graduation	Single	58138.0	0	0	2012-09-04 00:00:00	58	635	...	0	0	0	
1	2174	1954	Graduation	Single	46344.0	1	1	2014-03-08 00:00:00	38	11	...	0	0	0	
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21 00:00:00	26	426	...	0	0	0	
3	6182	1984	Graduation	Together	26646.0	1	0	2014-02-10 00:00:00	26	11	...	0	0	0	
4	5324	1981	PhD	Married	58293.0	1	0	2014-01-19 00:00:00	94	173	...	0	0	0	
5	7446	1967	Master	Together	62513.0	0	1	2013-09-09 00:00:00	16	520	...	0	0	0	
6	965	1971	Graduation	Divorced	55635.0	0	1	2012-11-13 00:00:00	34	235	...	0	0	0	
7	6177	1985	PhD	Married	33454.0	1	0	2013-05-08 00:00:00	32	76	...	0	0	0	
8	4855	1974	PhD	Together	30351.0	1	0	2013-06-06 00:00:00	19	14	...	0	0	0	
9	5899	1950	PhD	Together	5648.0	1	1	2014-03-13 00:00:00	68	28	...	0	0	0	

10 rows × 32 columns

[50]:
```
df.columns
```

[50]:
```
Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response',
       'Age', 'Total_Children', 'Total_Spent'],
      dtype='object')
```

[51]:
```
df.dtypes
```

[51]:
```
ID                     int64
Year_Birth             int64
Education             object
Marital_Status        object
Income               float64
Kidhome                int64
Teenhome               int64
Dt_Customer           object
Recency                int64
MntWines               int64
MntFruits              int64
MntMeatProducts        int64
MntFishProducts        int64
MntSweetProducts       int64
MntGoldProds           int64
NumDealsPurchases      int64
NumWebPurchases        int64
NumCatalogPurchases    int64
NumStorePurchases      int64
NumWebVisitsMonth      int64
AcceptedCmp3           int64
AcceptedCmp4           int64
AcceptedCmp5           int64
AcceptedCmp1           int64
AcceptedCmp2           int64
Complain               int64
Z_CostContact          int64
Z_Revenue              int64
Response               int64
Age                    int64
Total_Children         int64
Total_Spent            int64
dtype: object
```

[52]:
```
df['Dt_Customer'] = pd.to_datetime(df['Dt_Customer'], format='mixed', dayfirst=True)
```

[54]:
```
df.isnull().sum()
```

[54]:
```
ID                    0
Year_Birth            0
Education             0
Marital_Status        0
Income               24
Kidhome               0
Teenhome              0
Dt_Customer           0
Recency               0
MntWines              0
MntFruits             0
MntMeatProducts       0
MntFishProducts       0
MntSweetProducts      0
MntGoldProds          0
NumDealsPurchases     0
NumWebPurchases       0
NumCatalogPurchases   0
NumStorePurchases     0
NumWebVisitsMonth     0
AcceptedCmp3          0
AcceptedCmp4          0
AcceptedCmp5          0
AcceptedCmp1          0
AcceptedCmp2          0
Complain              0
Z_CostContact         0
Z_Revenue             0
Response              0
Age                   0
Total_Children        0
Total_Spent           0
dtype: int64
```

[55]:
```
#as for income = NA menaing that the pe
#Therefore, NA is replace by 0
df['Income'] = df['Income'].fillna(0)
```

[57]:
```
df.duplicated().sum()
```

[57]:
```
0
```

[58]:
```
df.drop('ID', axis=1, inplace=True)
```

[60]:
```
# Conduct feature Engineering
df['Total_Campaigns_Accepted'] = df[['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum(axis=1)

# Drop original campaign columns and other irrelevant features
df_cleaned = df.drop([
    'Z_CostContact', 'Z_Revenue','Complain',
    'Year_Birth','Dt_Customer','Kidhome', 'Teenhome',
    'MntWines', 'MntFruits', 'MntMeatProducts',
    'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',
    'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5'
], axis=1)
```

[61]:
```
# Label encode categorical columns
df_encoded = df_cleaned.copy()
label_encoders = {}
for col in df_encoded.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df_encoded[col] = le.fit_transform(df_encoded[col])
    label_encoders[col] = le

X = df_encoded.drop('Response', axis=1)
y = df_encoded['Response']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)

importances = rf.feature_importances_
feature_names = X.columns

feature_importance_df = pd.DataFrame({'Feature': feature_names, 'Importance': importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=feature_importance_df)
plt.title('Random Forest Feature Importances')
plt.tight_layout()
plt.show()
```

[64]:
```
df_labeled = df_encoded.copy()#df_selected.copy()

for col, le in label_encoders.items():
    if col in df_labeled.columns:
        df_labeled[col + '_label'] = le.inverse_transform(df_labeled[col])
```

[65]:
```
# Count occurrences in 'Response' column
response_counts = df_labeled['Response'].value_counts().sort_index()

labels = ['No', 'Yes']  # Corresponding to 0 and 1

fig, axs = plt.subplots(1, 2, figsize=(12, 5))

# Pie chart with labels and percentages
axs[0].pie(response_counts, labels=labels, autopct='%1.1f%%', startangle=90, colors=['#66b3ff','#ff9999'])
axs[0].set_title('Response Percentage')

# Bar chart with exact counts and labeled x-axis
bars = axs[1].bar(labels, response_counts.values, color=['#66b3ff','#ff9999'])
axs[1].set_xlabel('Response')
axs[1].set_ylabel('Count')
axs[1].set_title('Response Counts')

# Add count labels on top of each bar
for bar in bars:
    height = bar.get_height()
    axs[1].text(bar.get_x() + bar.get_width() / 2, height, f'{int(height)}',
                ha='center', va='bottom', fontsize=12)

plt.tight_layout()
plt.show()
```

[66]:
```
# Get unique categories of Education_label
edu_categories = df_labeled['Education_label'].unique()

labels = ['No', 'Yes']  # Response labels

# Number of categories
n = len(edu_categories)

fig, axs = plt.subplots(1, n, figsize=(4*n, 4))

for i, cat in enumerate(edu_categories):
    # Filter data for this category
    subset = df_labeled[df_labeled['Education_label'] == cat]

    # Count Responses 0 and 1
    counts = subset['Response'].value_counts().reindex([0,1], fill_value=0)

    # Plot pie chart
    axs[i].pie(counts, labels=labels, autopct='%1.1f%%', startangle=90, colors=['#66b3ff','#ff9999'])
    axs[i].set_title(f'Education: {cat}')

plt.tight_layout()
plt.show()
```

[67]:
```
marital_categories = df_labeled['Marital_Status_label'].unique()
labels = ['No', 'Yes']

fig, axs = plt.subplots(2, 4, figsize=(20, 8))
axs = axs.flatten()

for i, cat in enumerate(marital_categories):
    data = df_labeled[df_labeled['Marital_Status_label'] == cat]['Response'].value_counts().reindex([0, 1], fill_value=0)
    axs[i].pie(data, labels=labels, autopct='%1.1f%%', startangle=90, colors=['#66b3ff', '#ff9999'])
    axs[i].set_title(cat)

plt.tight_layout()
plt.show()
```

[68]:
```python
# Define your numeric columns
numeric_cols = ['Age', 'Income', 'Recency', 'Total_Spent', 'NumWebPurchases',
                'NumCatalogPurchases', 'NumStorePurchases', 'NumDealsPurchases',
                'Total_Campaigns_Accepted', 'NumWebVisitsMonth']

n = len(numeric_cols)

# Create a grid with n rows and 2 columns (boxplot + kde per row)
fig, axs = plt.subplots(n, 2, figsize=(12, 4 * n))

for i, col in enumerate(numeric_cols):
    # Boxplot
    sns.boxplot(data=df_labeled, y=col, ax=axs[i, 0], color='skyblue')
    axs[i, 0].set_title(f'Boxplot of {col}')
    axs[i, 0].grid(True)

    # KDE plot
    sns.kdeplot(data=df_labeled[col], fill=True, ax=axs[i, 1], color='salmon')
    axs[i, 1].set_title(f'KDE of {col}')
    axs[i, 1].grid(True)

plt.tight_layout()
plt.show()
```

[69]:
```python
selected_features = [
    'Total_Campaigns_Accepted', 'Total_Spent',
    'Recency', 'Age', 'Income', 'Response'
]

sns.pairplot(df_cleaned[selected_features], hue='Response', diag_kind='kde')
plt.show()
```

```python
X = df_labeled.drop('Response', axis=1)
y = df_labeled['Response']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.30, stratify=y, random_state=42
)

# Define categorical and numerical columns
categorical_cols = ['Education_label', 'Marital_Status_label']
numerical_cols = [col for col in X.columns if col not in categorical_cols]

# Preprocessing transformer
preprocessor = ColumnTransformer(transformers=[
    ('num', StandardScaler(), numerical_cols),
    ('cat', OneHotEncoder(drop='first'), categorical_cols)
])

# Define models
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "K-Nearest Neighbors": KNeighborsClassifier(),
    "Naive Bayes": GaussianNB(),
    "SVM": SVC(probability=True),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "XGBoost": XGBClassifier(eval_metric='logloss')
}

# Train and evaluate model
for name, model in models.items():
    pipeline = Pipeline([
        ('preprocess', preprocessor),
        ('model', model)
    ])

    pipeline.fit(X_train, y_train)
    y_pred = pipeline.predict(X_test)

    print(f"\n=== {name} ===")
    print(f"Accuracy: {accuracy_score(y_test, y_pred):.4f}")
    print(classification_report(y_test, y_pred, target_names=["No", "Yes"]))
```

```
results = []

for name, model in models.items():
    pipeline = Pipeline([
        ('preprocess', preprocessor),
        ('model', model)
    ])

    pipeline.fit(X_train, y_train)

    # Get predicted probabilities for ROC AUC
    if hasattr(model, "predict_proba"):
        y_proba = pipeline.predict_proba(X_test)[:, 1]
    else:
        if hasattr(model, "decision_function"):
            decision_scores = pipeline.decision_function(X_test)
            y_proba = (decision_scores - decision_scores.min()) / (decision_scores.max() - decision_scores.min())
        else:
            y_proba = pipeline.predict(X_test)

    roc_auc = roc_auc_score(y_test, y_proba)

    results.append({'Model': name, 'ROC AUC': roc_auc})

summary_df = pd.DataFrame(results)
print(summary_df)
```

```
results = []

for name, model in models.items():
    pipeline = Pipeline([
        ('preprocess', preprocessor),
        ('model', model)
    ])

    pipeline.fit(X_train, y_train)
    y_pred = pipeline.predict(X_test)

    if hasattr(model, "predict_proba"):
        y_proba = pipeline.predict_proba(X_test)[:, 1]
        roc_auc = roc_auc_score(y_test, y_proba)
    else:
        roc_auc = None

    # Compute metrics
    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)

    # Append to list
    results.append({
        'Model': name,
        'ROC AUC': roc_auc,
        'Accuracy': accuracy,
        'F1-Score': f1,
        'Precision': precision,
        'Recall': recall
    })

summary_df = pd.DataFrame(results).sort_values(by='ROC AUC', ascending=False).reset_index(drop=True)
summary_df
```

```
plt.figure(figsize=(10,6))
barplot = sns.barplot(x='Model', y='ROC AUC', data=summary_df, hue='Model', palette='viridis', dodge=False)
plt.legend([],[], frameon=False)  # Hide legend

# Add value labels on bars
for p in barplot.patches:
    height = p.get_height()
    barplot.annotate(f'{height:.3f}',
                     (p.get_x() + p.get_width() / 2., height),
                     ha='center', va='bottom', fontsize=9, color='black', xytext=(0, 3),
                     textcoords='offset points')

plt.title('ROC AUC Scores by Model')
plt.ylabel('ROC AUC')
plt.xlabel('Model')
plt.xticks(rotation=45, ha='right')
plt.ylim(0, 1)

plt.tight_layout()
plt.show()
```