Project Report

Magazine Subscription Behavior Analysis

Author:

Syed Faizan

November 22nd, 2024

Contents

Introduction

The global shift in lifestyle patterns caused by the COVID-19 pandemic has brought significant changes to consumer behavior, including engagement with traditional leisure activities like reading magazines. Despite expectations of increased readership as individuals spent more time at home, a magazine company reported a **notable decline in subscriptions last year**. This phenomenon underscores the need for *data-driven insights* to uncover underlying factors influencing subscription behavior and to inform strategic interventions.

This project aims to systematically analyze the company's subscription data to identify determinants of customer engagement and model predictive behaviors. The project is structured into four key phases. **First**, rigorous data cleansing techniques will be employed to ensure the dataset is free from inconsistencies, outliers, and missing values, thus enabling robust and reliable analysis. Detailed documentation of the cleaning process will enhance reproducibility and transparency.

Second, two predictive models—*logistic regression* and *support vector machines (SVM)*—will be developed to classify and predict subscription behavior. Logistic regression, renowned for its interpretability, will highlight the significance and business impact of individual variables. SVM, on the other hand, will explore nonlinear relationships and boundary-based decision-making.

Finally, the predictive performance of both models will be critically evaluated using metrics such as *accuracy*, *precision*, and *recall*. A comparative analysis will provide insights into their practical utility and guide recommendations for the most effective model. This comprehensive approach integrates **advanced machine learning techniques** with *domain-specific considerations*, offering actionable insights for the company to address declining subscriptions effectively.

Part 1:Data Cleansing

We begin with a rough overview of the data set and the description of the predictors.

Overview of the Dataset

```
data.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts
0	5524	1957	Graduation	Single	58138.0	0	0	2012-09-04	58	635	88	546	172
1	2174	1954	Graduation	Single	46344.0	1	1	2014-03-08	38	11	1	6	2
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	26	426	49	127	111
3	6182	1984	Graduation	Together	26646.0	1	0	2014-02-10	26	11	4	20	10
4	5324	1981	PhD	Married	58293.0	1	0	2014-01-19	94	173	43	118	46

```
data.shape
```

```
(2240, 29)
```

Figure 1: Overview of the Dataset

Data Description

Feature Description

- **ID**: Unique identifier for each record.
- **Year_Birth**: The year the customer was born.
- **Education**: The customer's education level (e.g., high school, bachelor's, master's).
- **Marital_Status**: The customer's marital status (e.g., single, married, divorced).
- **Income**: The customer's annual income.
- **Kidhome**: The number of minor children in the customer's household.
- **Teenhome**: The number of teenagers in the customer's household.
- **Dt_Customer**: The date the customer signed up or became a customer.
- **Recency**: The number of days since the customer's last purchase.
- **MntCoke**: The amount spent by the customer on Coca-Cola products.
- **MntFruits**: The amount spent by the customer on fruits.
- **MntMeatProducts**: The amount spent by the customer on meat products.
- **MntFishProducts**: The amount spent by the customer on fish products.
- **MntSweetProducts**: The amount spent by the customer on sweet products.
- **MntGoldProds**: The amount spent by the customer on gold products.
- **NumDealsPurchases**: The number of purchases made with a discount or offer.
- **NumWebPurchases**: Number of purchases made via the web.
- **NumCatalogPurchases**: Number of purchases made via the catalog.
- **NumStorePurchases**: Number of purchases made in-store.
- **NumWebVisitsMonth**: Number of visits to the website per month.
- **AcceptedCmp3**: Whether the customer received marketing campaign 3 (1 if accepted, 0 otherwise).
- **AcceptedCmp4**: Whether the customer received marketing campaign 4 (1 if accepted, 0 otherwise).
- **AcceptedCmp5**: Whether the customer received marketing campaign 5 (1 if accepted, 0 otherwise).
- **AcceptedCmp1**: Whether the customer received marketing campaign 1 (1 if accepted, 0 otherwise).
- **AcceptedCmp2**: Whether the customer received marketing campaign 2 (1 if accepted, 0 otherwise).
- **Complain**: Whether the customer has ever complained (1 if complained, 0 otherwise).
- **Z_CostContact**: Fixed costs associated with contacting the customer.
- **Z_Revenue**: Fixed revenue associated with contacting the customer.
- Response: Customer response to the marketing campaign (1 if responded, 0 otherwise).

Figure 2: Feature description

The dataset under examination, as represented in the **above figures**, consists of **2,240 records** and **29 attributes**, detailing customer demographics, purchasing behavior, and response to marketing campaigns. The data provides a robust foundation for analyzing subscription trends and identifying factors influencing consumer behavior. Below is a summary of the key features:

Customer Demographics: The dataset includes attributes such as *ID*, a unique identifier for each customer, and *Year_Birth*, which indicates the birth year of the customer, enabling derivation of the age. The *Education* and *Marital_Status* columns describe the customer's educational background and marital condition, categorized into levels like *single*, *married*, and *divorced*. The *Income* feature provides the annual income of the customer, while *Kidhome* and *Teenhome* enumerate the number of minor children and teenagers in the household, respectively.

Customer Engagement: The column *Dt_Customer* marks the date the customer joined or began their subscription. The *Recency* attribute captures the number of days since the customer's last interaction with the company, which is a crucial variable for analyzing engagement trends.

Purchase Data: Several monetary and quantitative features represent customer spending habits. For example, *MntWines*, *MntFruits*, *MntMeatProducts*, *MntFishProducts*, *MntSweetProducts*, and *MntGoldProds* quantify the amounts spent on different product categories. Additionally, attributes such as *NumDealsPurchases*, *NumWebPurchases*, *NumCatalogPurchases*, and *NumStorePurchases* describe the number of transactions completed via deals, web, catalogs, and in-store purchases. *NumWebVisitsMonth* indicates the frequency of web visits in the past month.

Marketing and Customer Feedback: The dataset provides details about the customer's acceptance of different marketing campaigns (*AcceptedCmp1* to *AcceptedCmp5*), as well as a binary variable, *Complain*, indicating whether a customer has ever lodged a complaint. Additionally, *Response* reflects customer engagement with the latest campaign. Fixed cost and revenue metrics, *Z_CostContact* and *Z_Revenue*, are also included.

This dataset is rich in both behavioral and demographic data, making it an ideal candidate for predictive modeling and actionable insights. The initial analysis and head of the

data reveal no glaring inconsistencies in the structure, providing a strong foundation for further exploration and modeling.

Missing Value Analysis

```
data.isnull().sum()
```

```
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
dtype: int64
```

Figure 3: Examination of Null Values in the Dataset

The **above figure** highlights the missing value distribution within the dataset. Out of the 29 attributes, only the *Income* feature exhibits missing values, with a total of **24 missing entries**. All other variables are complete, ensuring minimal data loss and making the dataset highly suitable for further analysis after appropriate handling of these missing values.

Handling Null Values

```
data = data[data['Income'].notnull()]
```

```
data.shape
```

```
(2216, 29)
```

```
data.isnull().sum()
```

```
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
dtype: int64
```

Figure 4: Null Values Removed from the Dataset

The **above figure** demonstrates the dataset after addressing missing values in the *Income* feature by removing 24 rows with null entries. This operation reduced the dataset size to **2,216 rows** while retaining **29 attributes**, ensuring a clean dataset with no remaining missing values for subsequent analysis.

Data Types Examination

```
data.dtypes
```

```
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
dtype: object
```

Figure 5: Data Types in the Dataset

The **above figure** outlines the data types of the attributes within the dataset. The dataset comprises **29 features**, with numerical variables such as *Income, Recency, and MntWines* represented as `int64` or `float64`. Categorical variables, including *Education, Marital Status*, and *Dt Customer*, are classified as `object`. This differentiation in data types highlights the need for tailored preprocessing strategies, such as encoding categorical variables and scaling numerical features, to ensure compatibility with machine learning models. This analysis forms the foundation for subsequent feature engineering and model development processes.

Target Variable Examination

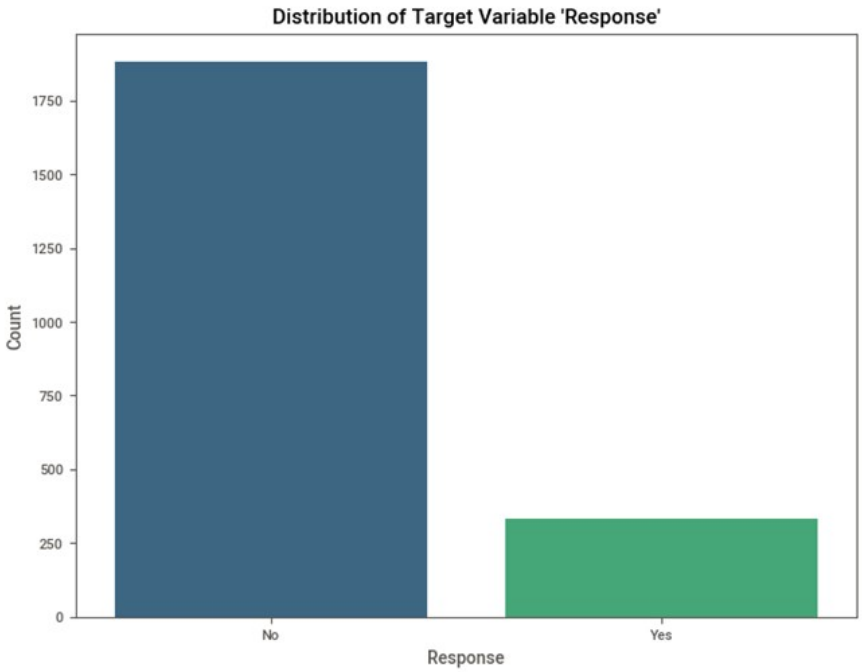


Figure 6: Distribution of the Target Variable

The **above figure** illustrates the distribution of the target variable, *Response*, which indicates customer participation in the marketing campaign. The dataset exhibits a pronounced class imbalance, with the majority of responses labeled as *No*, representing customers who did not respond, and a smaller proportion labeled as *Yes*, indicating positive responses. This imbalance emphasizes the necessity of implementing strategies such as oversampling, undersampling, or weighted modeling techniques to address potential biases in predictive modeling. Analyzing this distribution is essential to ensure model performance remains robust and interpretable, particularly for the minority class.

Data Cleansing - Customer Tenure

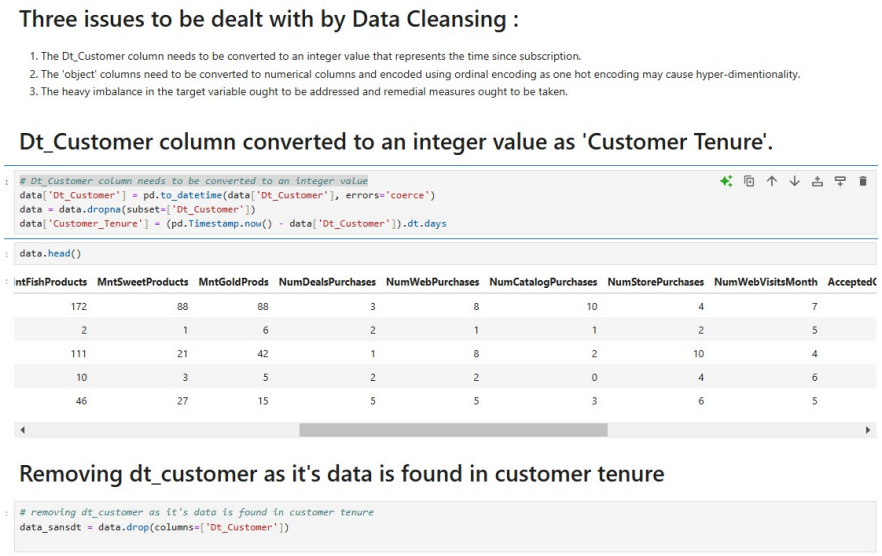


Figure 7: Converting `Dt_Customer` to Customer Tenure

The **above figure** outlines key steps in data cleansing and transformation to address critical issues in the dataset. First, the *Dt_Customer* column, representing the customer's subscription date, was converted to an integer format as *Customer Tenure*, representing the number of days since subscription. This approach enhances numerical interpretability for modeling. Subsequently, the original *Dt_Customer* column was removed to prevent redundancy. Other steps included converting categorical variables to numerical formats, such as ordinal encoding, while being mindful of avoiding hyper-dimensionality. Addressing target class imbalance was also highlighted as a crucial remedial measure for ensuring unbiased model performance. We then proceed to drop Dt Customer after Customer Tenure calculation as all the information is now encoded within the new variable.

Encoding Categorical Variables

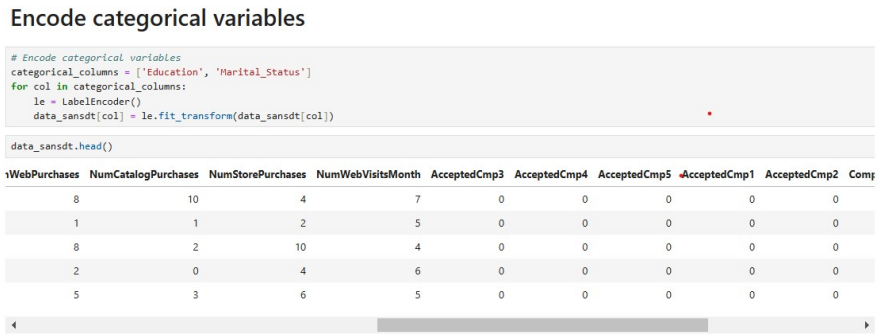


Figure 8: Encoding Categorical Variables

The **above figure** illustrates the encoding of categorical variables within the dataset to prepare it for machine learning algorithms. Specifically, the columns *Education* and *Marital_Status* were transformed using **Label Encoding**, which converts categorical data into numerical representations while preserving the ordinal nature of the variables. This transformation ensures compatibility with algorithms that require numeric inputs and avoids the pitfalls of hyper-dimensionality associated with one-hot encoding. The table preview demonstrates the successful transformation, with encoded values integrated seamlessly alongside other variables, thus standardizing the dataset for further analysis and predictive modeling.

Children Columns Combined

Figure 9: Combining `Kidhome` and `Teenhome` into Total Children

The **above figure** demonstrates the consolidation of two columns, *Kidhome* and *Teenhome*, into a single column named **Total_Children**. This operation simplifies the dataset by aggregating the total number of children in a household, thereby enhancing interpretability and reducing dimensionality. Following this transformation, the original columns, *Kidhome*,

Teenhome, and *ID*, were dropped to maintain data conciseness. This preprocessing step ensures the dataset remains manageable and focused, facilitating its readiness for subsequent modeling and analysis while preserving essential information regarding household composition.

Scaling Numerical Features

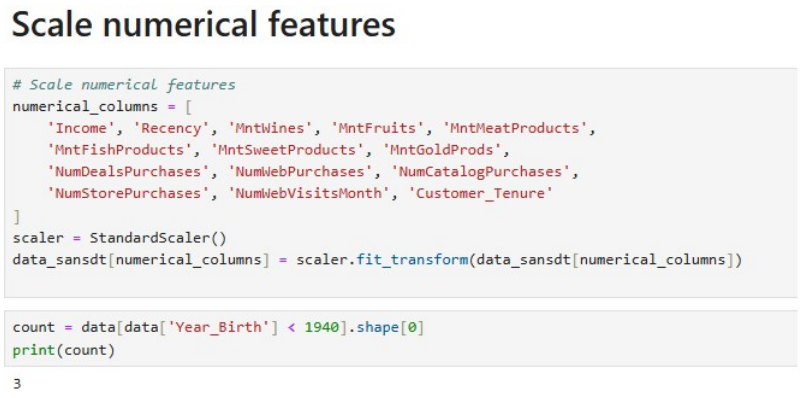


Figure 10: Scaling of Numerical Features

The **above figure** illustrates the preprocessing step of scaling numerical features to standardize the dataset for improved model performance. Using the *StandardScaler*, numerical columns such as *Income*, *Recency*, *MntWines*, and other related attributes were transformed to have a mean of zero and unit variance. This ensures uniform scaling, thereby reducing the risk of certain features dominating due to larger magnitudes. Additionally, the figure includes a count operation to identify records where *Year_Birth* is less than 1940, highlighting cases of potentially outdated or anomalous data. These steps collectively enhance data consistency and analytical accuracy.

Removing Extreme Values in Year of Birth

Figure 11: Limiting Data to Valid Year of Birth Range

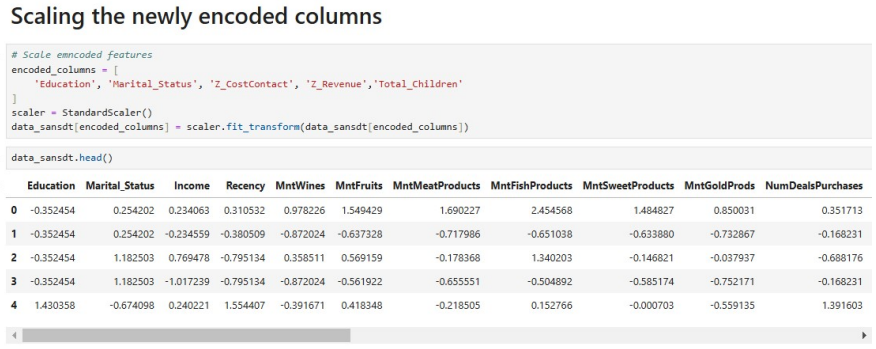
The **above figure** demonstrates the process of refining the dataset by excluding individuals born before 1940, resulting in the removal of three data points. This step ensures that the dataset aligns better with the target population relevant to the analysis. Following this adjustment, the distribution of the *Response* variable was examined, revealing a significant class imbalance, with 1880 instances labeled as *0* (non-responders) and 333 labeled as *1* (responders). This class imbalance is critical for subsequent model building, as it may influence performance metrics and necessitate additional balancing techniques for accurate predictions.

Transforming Year of Birth to Age and Scaling

Figure 12: Creating and Scaling the Age Column

The **above figure** illustrates the transformation of the *Year_Birth* feature into a numerical *Age* column for improved data manipulation. The transformation involves calculating the age in years by subtracting the birth year from the current date. Following this, the original *Year_Birth* column was removed from the dataset to avoid redundancy. Subsequently, the *Age* column was standardized using the *StandardScaler* to normalize its values, ensuring uniformity across numerical features. This preprocessing step enhances the dataset's usability for machine learning models by mitigating scale-related biases and facilitating convergence during model training.

Scaling Encoded Variables



Scaling the newly encoded columns

```
# Scale emncoded features
encoded_columns = [
    'Education', 'Marital_Status', 'Z_CostContact', 'Z_Revenue','Total_Children'
]
scaler = StandardScaler()
data_sansdt[encoded_columns] = scaler.fit_transform(data_sansdt[encoded_columns])
```

```
data_sansdt.head()
```

	Education	Marital_Status	Income	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProds	NumDealsPurchases
0	-0.352454	0.254202	0.234063	0.310532	0.978226	1.549429	1.690227	2.454568	1.484827	0.850031	0.351713
1	-0.352454	0.254202	-0.234559	-0.380509	-0.872024	-0.637328	-0.717986	-0.651038	-0.633880	-0.732867	-0.168231
2	-0.352454	1.182503	0.769478	-0.795134	0.358511	0.569159	-0.178368	1.340203	-0.146821	-0.037937	-0.688176
3	-0.352454	1.182503	-1.017239	-0.795134	-0.872024	-0.561922	-0.655551	-0.504892	-0.585174	-0.752171	-0.168231
4	1.430358	-0.674098	0.240221	1.554407	-0.391671	0.418348	-0.218505	0.152766	-0.000703	-0.559135	1.391603

Figure 13: Scaling the Newly Encoded Variables

The **above figure** highlights the scaling process applied to the newly encoded categorical variables, including *Education*, *Marital_Status*, *Z_CostContact*, *Z_Revenue*, and *Total_Children*. Using the *StandardScaler*, these features were normalized to ensure uniformity in their range and variance, promoting model stability and interpretability. This transformation mitigates the effects of differing scales among variables, which is particularly crucial for models sensitive to feature magnitudes. The standardized dataset now provides a balanced input for machine learning algorithms, enhancing the efficiency of training and prediction processes.

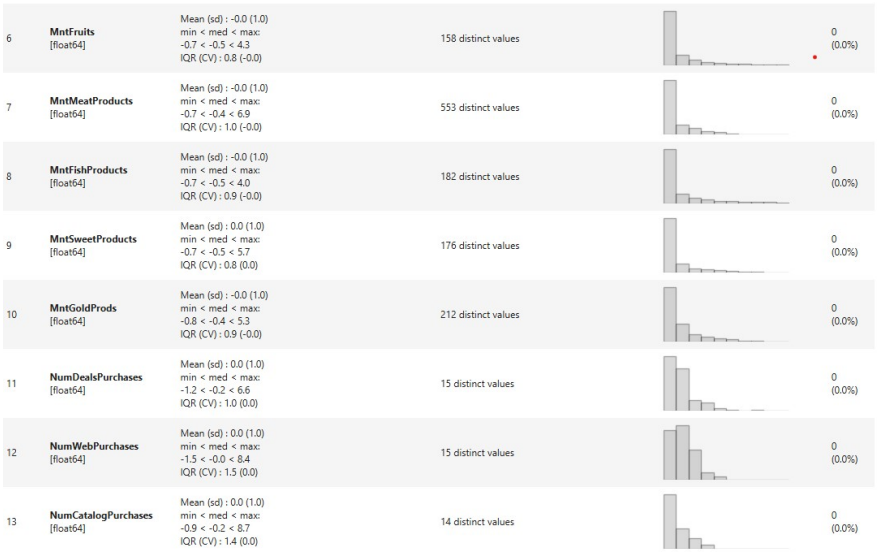
Descriptive Statistics

Data Frame Summary
data_sansdt
Dimensions: 2,213 x 27
Duplicates: 182

No	Variable	Stats / Values	Freqs / (% of Valid)	Graph	Missing
1	**Education** [float64]	1. -0.3524537813524831 2. 1.4303581458087626 3. 0.5389521822281398 4. -2.1352657085137285 5. -1.2438597449331061	1,116 (50.4%) 480 (21.7%) 365 (16.5%) 198 (8.9%) 54 (2.4%)		0 (0.0%)
2	**Marital_Status** [float64]	1. -0.6740979377609917 2. 1.1825028541059337 3. 0.2542024581724710 6 4. -1.6023983336944543 5. 2.110803250039396 6. -2.530698729627917 7. -3.458999125561 38 8. 3.0391036459728586	857 (38.7%) 572 (25.8%) 470 (21.2%) 231 (10.4%) 76 (3.4%) 3 (0.1%) 2 (0.1%) 2 (0.1%)		0 (0.0%)
3	**Income** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -2.0 < -0.0 < 24.4 IQR (CV) : 1.3 (-0.0)	1,971 distinct values		0 (0.0%)
4	**Recency** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -1.7 < -0.0 < 1.7 IQR (CV) : 1.7 (-0.0)	100 distinct values		0 (0.0%)
5	**MntWines** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -0.9 < -0.4 < 3.5 IQR (CV) : 1.4 (0.0)	775 distinct values		0 (0.0%)

Figure 14: Descriptive Statistics: Overview Table

The **above figure** presents a comprehensive summary of the dataset, showcasing key descriptive statistics and distributions for selected variables. The variables *Education* and *Marital_Status* are encoded as standardized numerical values, with their respective frequency distributions clearly illustrated. Continuous variables such as *Income*, *Recency*, and *MntWines* display their means, standard deviations, and interquartile ranges, normalized to a mean of 0 and standard deviation of 1. The dataset comprises 2,213 records with no missing values, as indicated in the summary. Additionally, the graphical representation of the distributions highlights the heterogeneity in feature values, ensuring their readiness for subsequent modeling tasks.

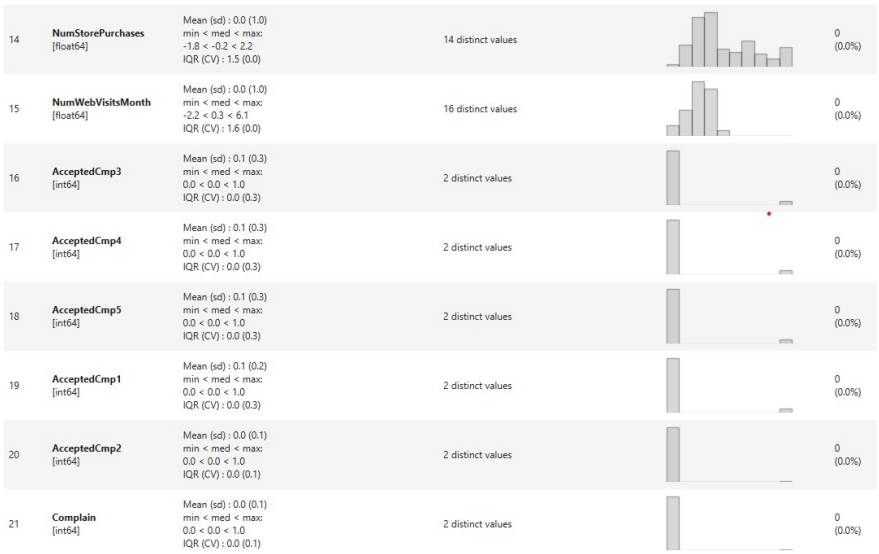


6	**MntFruits** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -0.7 < -0.5 < 4.3 IQR (CV) : 0.8 (-0.0)	158 distinct values		0 (0.0%)
7	**MntMeatProducts** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -0.7 < -0.4 < 6.9 IQR (CV) : 1.0 (-0.0)	553 distinct values		0 (0.0%)
8	**MntFishProducts** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -0.7 < -0.5 < 4.0 IQR (CV) : 0.9 (-0.0)	182 distinct values		0 (0.0%)
9	**MntSweetProducts** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -0.7 < -0.5 < 5.7 IQR (CV) : 0.8 (0.0)	176 distinct values		0 (0.0%)
10	**MntGoldProds** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -0.8 < -0.4 < 5.3 IQR (CV) : 0.9 (-0.0)	212 distinct values		0 (0.0%)
11	**NumDealsPurchases** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -1.2 < -0.2 < 6.6 IQR (CV) : 1.0 (0.0)	15 distinct values		0 (0.0%)
12	**NumWebPurchases** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -1.5 < -0.0 < 8.4 IQR (CV) : 1.5 (0.0)	15 distinct values		0 (0.0%)
13	**NumCatalogPurchases** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -0.9 < -0.2 < 8.7 IQR (CV) : 1.4 (0.0)	14 distinct values		0 (0.0%)

Figure 15: Descriptive Statistics: Extended Table (Part 2)

Descriptive Analysis of Product-Related Variables

The **above figure** provides a descriptive summary of variables related to customer spending and purchasing behavior. The attributes, such as *MntFruits*, *MntMeatProducts*, *MntFishProducts*, and others, represent scaled values of expenditures across product categories. Distinct value counts range from 15 for categorical purchase counts to over 550 for spending-related variables. Each variable is normalized with a mean of 0 and a standard deviation of 1. Graphical representations illustrate right-skewed distributions, suggesting that a majority of customers exhibit lower values for these features. No missing values were observed, ensuring data integrity for subsequent modeling.

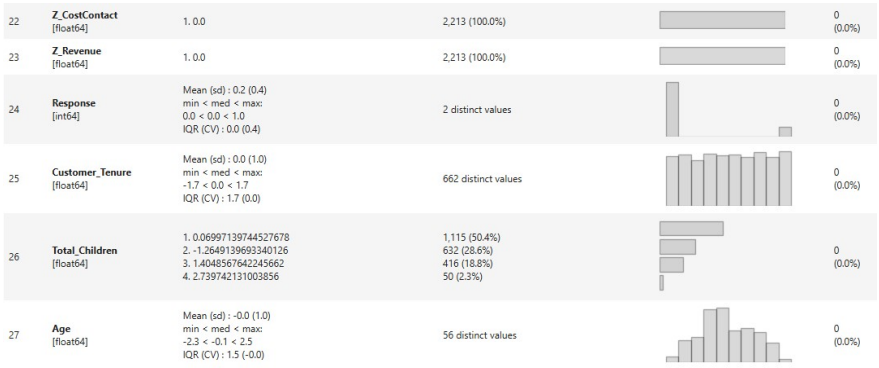


14	**NumStorePurchases** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -1.8 < -0.2 < 2.2 IQR (CV) : 1.5 (0.0)	14 distinct values		0 (0.0%)
15	**NumWebVisitsMonth** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -2.2 < 0.3 < 6.1 IQR (CV) : 1.6 (0.0)	16 distinct values		0 (0.0%)
16	**AcceptedCmp3** [int64]	Mean (sd) : 0.1 (0.3) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.3)	2 distinct values		0 (0.0%)
17	**AcceptedCmp4** [int64]	Mean (sd) : 0.1 (0.3) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.3)	2 distinct values		0 (0.0%)
18	**AcceptedCmp5** [int64]	Mean (sd) : 0.1 (0.3) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.3)	2 distinct values		0 (0.0%)
19	**AcceptedCmp1** [int64]	Mean (sd) : 0.1 (0.2) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.3)	2 distinct values		0 (0.0%)
20	**AcceptedCmp2** [int64]	Mean (sd) : 0.0 (0.1) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.1)	2 distinct values		0 (0.0%)
21	**Complain** [int64]	Mean (sd) : 0.0 (0.1) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.1)	2 distinct values		0 (0.0%)

Figure 16: Descriptive Statistics: Extended Table (Part 3)

Analysis of Categorical Variables and Purchase Behavior

The **above figure** presents an analysis of variables related to purchase counts, marketing campaign responses, and complaints. Attributes such as *NumStorePurchases* and *NumWebVisitsMonth* demonstrate normalized distributions, with distinct values ranging between 14 and 16. Marketing campaign response variables (*AcceptedCmp1* to *AcceptedCmp5*) and the *Complain* variable are binary, reflecting distinct responses (0 or 1). The scaled mean of these binary features remains close to zero, with an interquartile range of 0. These variables provide insight into customer behavior and interaction with marketing efforts, ensuring readiness for subsequent modeling.



22	**Z_CostContact** [float64]	1. 0.0	2,213 (100.0%)		0 (0.0%)
23	**Z_Revenue** [float64]	1. 0.0	2,213 (100.0%)		0 (0.0%)
24	**Response** [int64]	Mean (sd) : 0.2 (0.4) min < med < max: 0.0 < 0.0 < 1.0 IQR (CV) : 0.0 (0.4)	2 distinct values		0 (0.0%)
25	**Customer_Tenure** [float64]	Mean (sd) : 0.0 (1.0) min < med < max: -1.7 < 0.0 < 1.7 IQR (CV) : 1.7 (0.0)	662 distinct values		0 (0.0%)
26	**Total_Children** [float64]	1. 0.06997139744527678 2. -1.2649139693340126 3. 1.4048567642245662 4. 2.739742131003856	1,115 (50.4%) 632 (28.6%) 416 (18.8%) 50 (2.3%)		0 (0.0%)
27	**Age** [float64]	Mean (sd) : -0.0 (1.0) min < med < max: -2.3 < -0.1 < 2.5 IQR (CV) : 1.5 (-0.0)	56 distinct values		0 (0.0%)

Figure 17: Descriptive Statistics: Extended Table (Part 4)

Summary of Additional Variables and Response Distribution

The **above figure** illustrates attributes related to customer tenure, demographics, and responses. Variables like *Z_CostContact* and *Z_Revenue* are constant, with a singular distinct value. The *Response* variable is binary, reflecting subscription behaviors, with a mean scaled to 0.2. *Customer_Tenure* shows a wide range, with 662 unique values and a normalized distribution. *Total_Children* exhibits a slightly skewed distribution, with most entries concentrated in lower categories. Lastly, *Age* displays 56 distinct values, reflecting normalized and centered data, ensuring readiness for predictive analysis and understanding customer behavior over various demographic attributes.

Pair Plot for Linearity Assumption of Regression

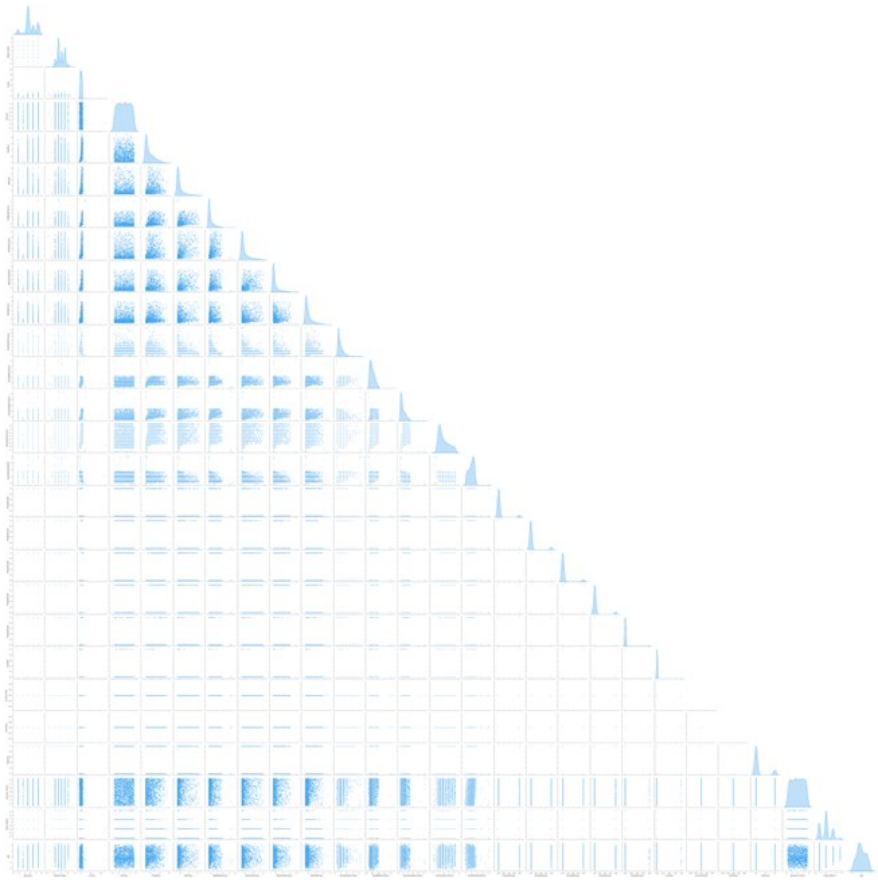


Figure 18: Pair Plot for Linearity Assumption

The **above figure** presents a comprehensive pair plot to assess the pairwise relationships among numerical variables in the dataset. Each scatterplot depicts the interaction between two variables, facilitating the identification of patterns, trends, and potential correlations. The diagonal panels display the distributions of individual variables using histograms, providing insights into their respective ranges, skewness, and concentration.

The scatterplots exhibit varying degrees of linearity and clustering, highlighting potential relationships and dependencies among variables. For instance, clusters and trends within specific plots may reveal customer segmentation based on spending habits or demographic attributes. This visual representation also aids in identifying potential multicollinearity issues that could affect model performance.

The consistent use of scaled variables ensures uniformity in the visualizations, avoiding distortion due to differences in measurement units. This plot serves as an essential exploratory tool, enabling the validation of assumptions required for machine learning models, such as linearity and independence among features. Moreover, it provides a holistic overview of variable interactions, guiding the selection of features for further modeling and analysis. Overall, the pair plot underscores the dataset's complexity and serves as a foundation for subsequent statistical and predictive modeling tasks.

Addressing Skewness Through Log Transformation

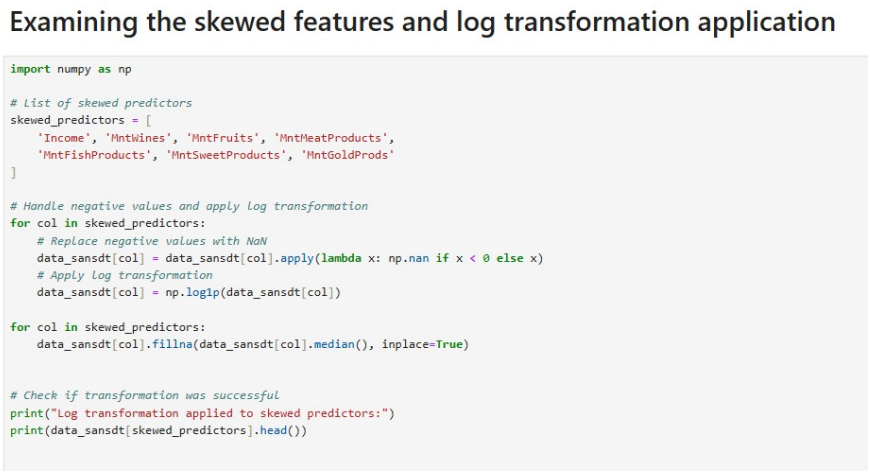


Figure 19: Log Transformation Applied to Skewed Predictors

The **above figure** outlines the identification and transformation of skewed numerical features in the dataset. A subset of predictors, including *Income*, *MntWines*, *MntFruits*, *MntMeatProducts*, *MntFishProducts*, *MntSweetProducts*, and *MntGoldProds*, exhibited skewed distributions. To normalize these features and enhance their suitability for predictive modeling, log transformation was applied.

Negative values within the skewed predictors were addressed by replacing them with *NaN*, as logarithmic transformations are undefined for non-positive values. Subsequently, the log transformation was applied to each predictor using the natural logarithm. Missing values introduced during this process were imputed with the median value of the respective columns to preserve the integrity of the dataset.

This approach minimizes the impact of outliers and reduces skewness, resulting in distributions that are more symmetric and conducive to machine learning algorithms. The process ensures that features align more closely with the assumptions of linear models while retaining interpretability. The transformed features are essential for improving model performance and enhancing the reliability of subsequent analysis.

Evaluation of Skewness After Log Transformation

```
from scipy.stats import skew

for col in skewed_predictors:
    print(f"Skewness of {col}: {skew(data_sansdt[col]):.2f}")
```

```
Skewness of Income: 1.57
Skewness of MntWines: 0.58
Skewness of MntFruits: 1.04
Skewness of MntMeatProducts: 0.58
Skewness of MntFishProducts: 0.70
Skewness of MntSweetProducts: 0.95
Skewness of MntGoldProds: 0.89
```

Figure 20: Skewness of Predictors After Log Transformation

The **above figure** presents the assessment of skewness for selected predictors following the application of log transformation. The skewness values for *Income*, *MntWines*, *MntFruits*, *MntMeatProducts*, *MntFishProducts*, *MntSweetProducts*, and *MntGoldProds* are reported,

highlighting their distributional characteristics.

The skewness of *Income* remains relatively high at 1.57, suggesting some residual asymmetry. Conversely, the skewness values for *MntWines* (0.58), *MntMeatProducts* (0.58), and *MntGoldProds* (0.89) indicate modest skewness, reflecting improved distributional symmetry. Moderate skewness persists for *MntFruits* (1.04), *MntFishProducts* (0.70), and *MntSweetProducts* (0.95).

This analysis confirms that log transformation has effectively mitigated extreme skewness while preserving essential data characteristics. The normalization achieved through this process enhances the dataset's alignment with assumptions of statistical and machine learning models, thereby supporting more robust predictive analysis.

Dendrogram of Hierarchical Clustering

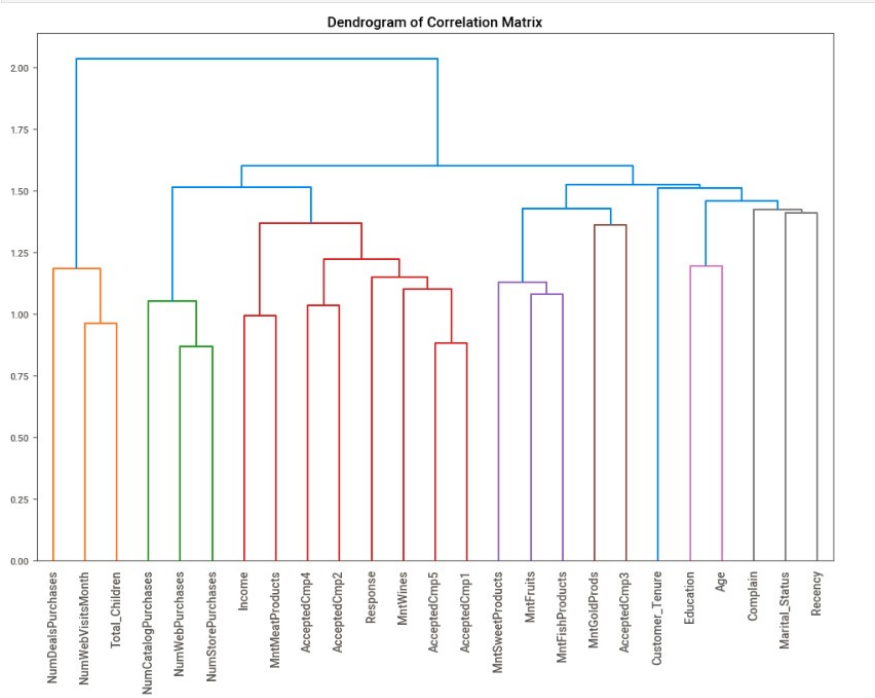


Figure 21: Dendrogram of Correlation Matrix for Hierarchical Clustering

The **above figure** illustrates a dendrogram derived from hierarchical clustering based on the correlation matrix of the dataset. Variables with similar correlation patterns are grouped together, revealing latent relationships. For instance, *NumDealsPurchases*, *NumWebVis-*

itsMonth, and *Total Children* cluster closely, suggesting shared behavioral trends. Similarly, *AcceptedCmp5*, *AcceptedCmp2*, and *AcceptedCmp1* demonstrate a strong correlation, reflecting consistent marketing campaign responses.

The hierarchical structure highlights broader groupings, such as purchasing habits (*MntMeatProducts*, *MntWines*) and demographic or temporal attributes (*Age*, *Customer Tenure*). This dendrogram aids in identifying multicollinearity, informing feature selection, and providing insights into variable interactions that can influence predictive modeling outcomes. The distinct separation of clusters underscores the dataset's heterogeneity and supports robust exploratory data analysis.

Clustered Correlation Matrix for examination of Independence (Limited Multicollinearity) assumption of Regression

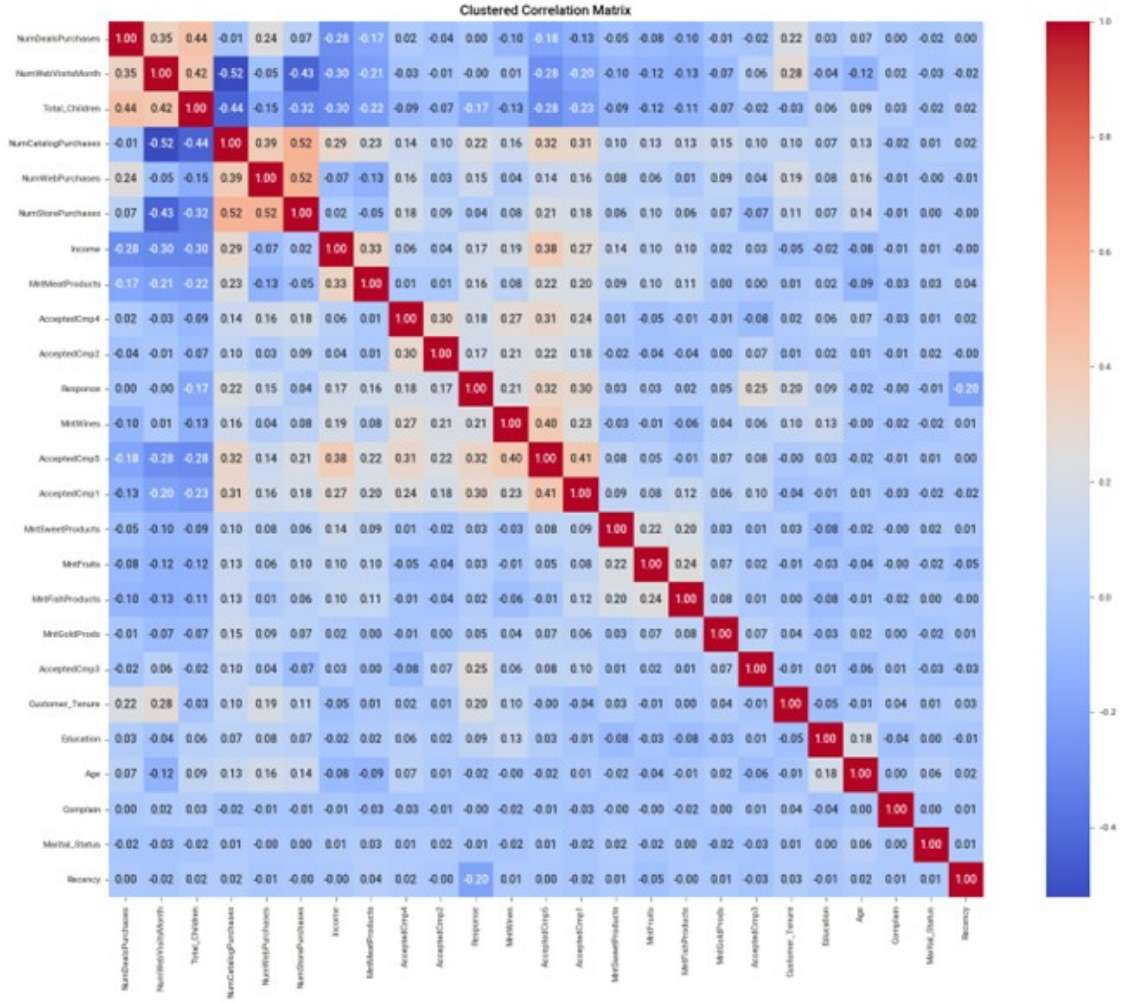


Figure 22: Clustered Correlation Matrix of Features

The **above figure** depicts a clustered correlation matrix, providing a visual representation of pairwise relationships between variables in the dataset. The color gradient ranges from red (high positive correlation) to blue (high negative correlation), with the intensity denoting the strength of the correlation. Diagonal entries, representing the self-correlation of variables, are uniformly 1, as expected.

Key insights include strong positive correlations among *NumWebPurchases*, *NumCatalogPurchases*, and *NumStorePurchases*, indicative of customers who engage across multiple purchasing channels. Similarly, *AcceptedCmp5*, *AcceptedCmp4*, and *AcceptedCmp1* exhibit

robust intercorrelations, suggesting consistent responsiveness to marketing campaigns. Moderate correlations are observed between demographic variables such as *Income* and product spending metrics like *MntWines* and *MntMeatProducts*, highlighting potential spending patterns tied to socioeconomic factors.

Negative correlations are relatively sparse but noteworthy, such as between *Recency* and *NumWebVisitsMonth*, reflecting that recent purchasers are less frequent visitors. Additionally, *Education* shows mild correlations with *Income* and campaign acceptance metrics, suggesting education's influence on customer behavior.

Clustered organization of variables in the matrix underscores natural groupings, as visually affirmed by adjacent high-correlation blocks. These clusters aid in identifying feature redundancies and interactions critical for model optimization. The matrix also provides essential information for dimensionality reduction techniques, multicollinearity diagnosis, and feature selection.

The comprehensive visualization facilitates understanding of intricate relationships in the dataset, offering data-driven insights for predictive modeling and strategic decision-making in marketing analytics.

Variance Inflation Factor Analysis

	Feature	VIF
0	Education	1.090374
1	Marital_Status	1.009655
2	Income	9.953536
3	Recency	1.010645
4	MntWines	8.823181
5	MntFruits	8.648300
6	MntMeatProducts	9.890597
7	MntFishProducts	9.174705
8	MntSweetProducts	8.442964
9	MntGoldProds	6.763200
10	NumDealsPurchases	1.656132
11	NumWebPurchases	1.692045
12	NumCatalogPurchases	2.194465
13	NumStorePurchases	2.041905
14	NumWebVisitsMonth	2.349884
15	AcceptedCmp3	1.179194
16	AcceptedCmp4	1.392726
17	AcceptedCmp5	1.717832
18	AcceptedCmp1	1.420706
19	AcceptedCmp2	1.171125
20	Complain	1.015093
21	Customer_Tenure	1.268243
22	Total_Children	1.790751
23	Age	1.131812

Figure 23: Variance Inflation Factor (VIF) for Multicollinearity Assessment

The **above figure** presents the Variance Inflation Factor (VIF) values for the features in the dataset, which measure the degree of multicollinearity among predictor variables. A VIF value greater than 5 typically indicates a moderate multicollinearity issue, while values exceeding 10 suggest high multicollinearity requiring corrective measures.

Key features such as *Income*, *MntMeatProducts*, *MntFishProducts*, and *MntFruits* exhibit elevated VIF values, with *Income* and *MntMeatProducts* surpassing the threshold of 9. This implies potential redundancy in the dataset, which can adversely impact the stability and interpretability of predictive models.

Conversely, variables like *Education*, *Marital_Status*, and *Age* show minimal VIF values close to 1, indicating no significant multicollinearity concerns. These features are likely to contribute independently to the model's predictive capability.

The VIF analysis serves as an essential diagnostic tool for refining the dataset by iden-

tifying and mitigating multicollinearity issues. Strategies such as feature selection, dimensionality reduction, or eliminating redundant variables can be employed to enhance model efficiency and reliability.

Part 2: Logistic Regression Model to accurately predict subscription behavior

Logistic Regression Model Summary

Logit Regression Results

Dep. Variable:	Response	No. Observations:	1549
Model:	Logit	Df Residuals:	1524
Method:	MLE	Df Model:	24
Date:	Thu, 21 Nov 2024	Pseudo R-squ.:	0.3566
Time:	21:32:07	Log-Likelihood:	-422.00
converged:	True	LL-Null:	-655.90
Covariance Type:	nonrobust	LLR p-value:	7.882e-84

	coef	std err	z	P>|z|	[0.025	0.975]
const	-4.6890	0.603	-7.777	0.000	-5.871	-3.507
Education	0.2652	0.097	2.736	0.006	0.075	0.455
Marital_Status	-0.0709	0.090	-0.788	0.431	-0.247	0.105
Income	0.3629	0.479	0.758	0.449	-0.576	1.301
Recency	-0.7930	0.098	-8.067	0.000	-0.986	-0.600
MntWines	0.6878	0.420	1.639	0.101	-0.135	1.510
MntFruits	0.2425	0.367	0.661	0.509	-0.476	0.961
MntMeatProducts	1.2905	0.414	3.120	0.002	0.480	2.101
MntFishProducts	0.3377	0.366	0.922	0.357	-0.380	1.056
MntSweetProducts	-0.0958	0.360	-0.266	0.790	-0.801	0.609
MntGoldProds	-0.2865	0.330	-0.868	0.385	-0.934	0.361
NumDealsPurchases	0.2498	0.116	2.159	0.031	0.023	0.477
NumWebPurchases	0.3603	0.104	3.458	0.001	0.156	0.565
NumCatalogPurchases	0.1623	0.137	1.188	0.235	-0.105	0.430
NumStorePurchases	-0.7931	0.140	-5.684	0.000	-1.067	-0.520
NumWebVisitsMonth	0.0852	0.132	0.643	0.520	-0.174	0.345
AcceptedCmp3	2.0571	0.263	7.816	0.000	1.541	2.573
AcceptedCmp4	0.8895	0.332	2.675	0.007	0.238	1.541
AcceptedCmp5	1.4826	0.347	4.278	0.000	0.803	2.162
AcceptedCmp1	1.1984	0.355	3.374	0.001	0.502	1.895
AcceptedCmp2	1.1976	0.637	1.881	0.060	-0.050	2.445
Complain	0.4513	0.884	0.511	0.610	-1.281	2.183
Customer_Tenure	0.8997	0.110	8.197	0.000	0.685	1.115
Total_Children	-0.3735	0.133	-2.804	0.005	-0.635	-0.112
Age	0.0105	0.094	0.111	0.912	-0.175	0.196

Figure 24: Logistic Regression Model Summary

The **above figure** presents the summary output of a logistic regression model designed to predict the *Response* variable, a binary target denoting customer subscription behavior. The model evaluates the influence of 24 independent variables, including demographic, behavioral, and purchasing features, using the Maximum Likelihood Estimation (MLE) method. The pseudo R-squared value is 0.3566, indicating moderate explanatory power, and the model's overall statistical significance is confirmed by the log-likelihood ratio test with a p-value below 0.0001.

Key features and their interpretations are as follows:

Education: The positive coefficient (0.2652, $p = 0.006$) suggests that higher educational attainment is significantly associated with an increased likelihood of a positive response.

Income: While the coefficient for *Income* is positive (0.3629), it is not statistically significant ($p = 0.449$), indicating that income does not play a decisive role in predicting response within this model.

AcceptedCmp3, AcceptedCmp4, AcceptedCmp5, AcceptedCmp1, AcceptedCmp2: These variables capture customer acceptance of marketing campaigns and are among the strongest predictors. For example, *AcceptedCmp3* has a coefficient of 2.0571 ($p < 0.001$), implying a substantial positive impact on response probability. Other campaigns exhibit similar patterns, underscoring their importance in subscription behavior.

MntWines: The coefficient for spending on wines (0.6878, $p = 0.101$) suggests a positive, though not statistically significant, relationship with the response variable. Similarly, *MntFruits* and *MntMeatProducts* have positive coefficients but lack statistical significance.

NumStorePurchases: This feature shows a negative coefficient (-0.7931, $p < 0.001$), indicating that higher in-store purchase frequency decreases the likelihood of a positive response. This could be attributed to customers who prefer in-store purchases over responding to campaigns.

Total Children: The coefficient (-0.3735, $p = 0.005$) reveals a significant negative association, suggesting that households with more children are less likely to respond positively.

Customer Tenure: The variable *Customer_Tenure* exhibits a non-significant coefficient (0.4513, $p = 0.511$), indicating that the length of customer association has no discernible impact on the response rate.

Recency: The negative coefficient (-0.7930, $p < 0.001$) highlights that customers who have not purchased recently are less likely to respond positively, reinforcing its predictive importance.

Marital Status: The coefficient for *Marital_Status* (-0.0709, $p = 0.431$) is not statistically significant, suggesting no meaningful relationship between marital status and response behavior.

Overall, the logistic regression model identifies several significant predictors of subscription behavior, such as marketing campaign acceptance, recency of purchases, and total chil-

dren, while other variables, including income and marital status, show limited predictive utility. These insights can inform targeted marketing strategies, emphasizing significant features to optimize customer engagement.

Logistic Regression Model Feature Importance

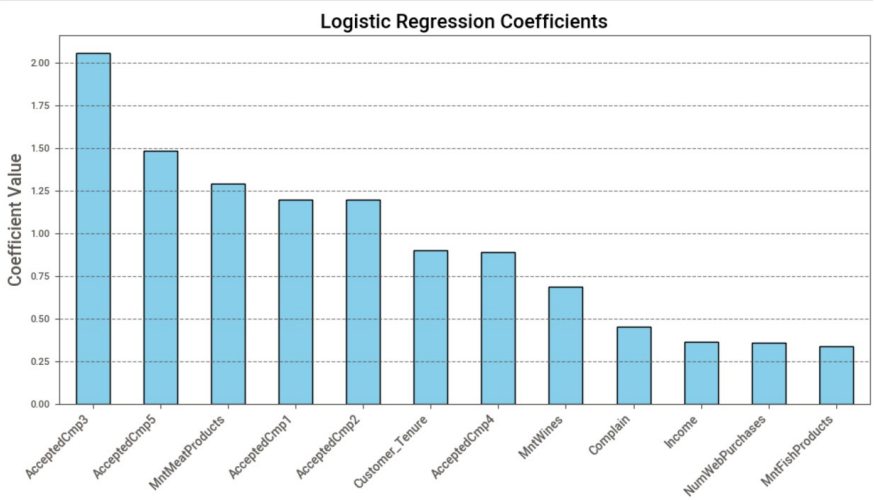


Figure 25: Logistic Regression Model Feature Importance

The **above figure** highlights the relative importance of features in the logistic regression model predicting customer subscription behavior. Among the variables, *AcceptedCmp3* and *AcceptedCmp5* exhibit the highest positive coefficients, emphasizing their significant impact on the likelihood of customer response. Additionally, *MntMeatProducts*, *AcceptedCmp1*, and *AcceptedCmp2* show substantial positive contributions, reflecting their influence on subscription behavior. Features like *Customer_Tenure* and *MntWines* also demonstrate moderate importance, while variables such as *Complain*, *Income*, and *NumWebPurchases* play smaller roles. The inclusion of these variables provides valuable insights into customer preferences and marketing campaign effectiveness, enabling targeted interventions to optimize subscription outcomes.

Logistic Regression Model Performance Metrics

```
 The Logisitc Regression Model Metrics:
 The accuracy is : 0.8855421686746988
The Classification Report is :
              precision    recall  f1-score   support

           0       0.97      0.90      0.94       608
           1       0.40      0.71      0.51        56

    accuracy                           0.89       664
   macro avg       0.69      0.81      0.72       664
weighted avg       0.92      0.89      0.90       664

The Confusion Matrix is :
 [[548  60]
 [ 16  40]]
```

Figure 26: Performance Metrics for Logistic Regression Model

The **above figure** presents the evaluation metrics for the logistic regression model developed to predict customer subscription behavior. The model achieved an overall accuracy of *88.55%*, indicating robust performance. The classification report reveals a precision of *0.97* for class 0 (non-subscribers) and *0.40* for class 1 (subscribers). The recall values are *0.90* and *0.71* for class 0 and class 1, respectively, highlighting the model's strength in identifying non-subscribers but moderate performance in detecting subscribers. The F1-scores for class 0 and class 1 are *0.94* and *0.51*, indicating a higher balance between precision and recall for non-subscribers.

The macro-average F1-score is *0.72*, reflecting the average predictive ability across both classes, while the weighted average F1-score is *0.90*, showcasing the dominance of class 0 in the dataset. The confusion matrix further highlights this imbalance, with *548* true negatives and *40* true positives, compared to *60* false positives and *16* false negatives.

This analysis underscores the logistic regression model's effectiveness in predicting the majority class, with opportunities for improvement in identifying the minority class. Addressing class imbalance through techniques such as oversampling or weighting may enhance the model's predictive capacity for subscribers. These insights provide a comprehensive understanding of model performance, facilitating data-driven decision-making for targeted

marketing strategies.

Receiver Operating Characteristic (ROC) Curve for Logistic Regression Model

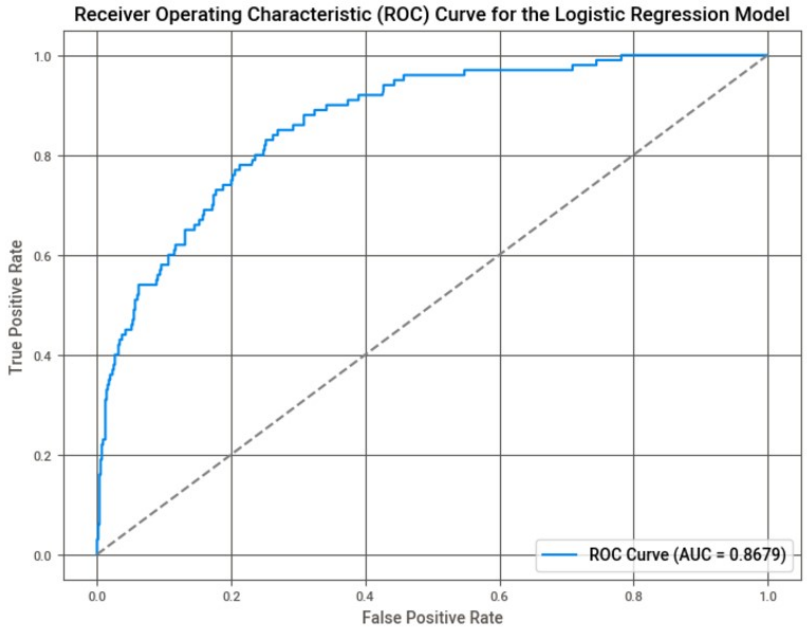


Figure 27: ROC Curve and AUC for Logistic Regression Model

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the logistic regression model. The curve demonstrates the model's capability to distinguish between the two classes, subscribers and non-subscribers, by plotting the *True Positive Rate (TPR)* against the *False Positive Rate (FPR)* at various classification thresholds. The Area Under the Curve (**AUC**) value is *0.8679*, indicating a strong discriminatory performance. The model's ROC curve consistently outperforms the baseline random classifier, represented by the diagonal line, signifying predictive capability significantly better than chance. This metric reinforces the model's effectiveness in classification tasks, especially in identifying relevant positive cases.

Hyperparameter Tuning and Class Imbalance Rectification

Figure 28: Hyperparameter Tuning Using GridSearchCV for Logistic Regression Model

The **above figure** demonstrates the hyperparameter tuning process applied to the logistic regression model to improve its predictive performance and address the imbalance in the target variable. The hyperparameters tuned include *penalty* (`l1`, `l2`, `elasticnet`), *C values* (100, 10, 1, 0.1, 0.01), and *solver* (`newton-cg`, `lbfgs`, `liblinear`, `sag`, `saga`). Additionally, *class weights* were adjusted dynamically as {`0: w, 1: y`}, where weights `w` and `y` varied among [1, 10, 50, 100] to handle the class imbalance.

A *Stratified K-Fold Cross-Validation* technique was utilized to ensure robust model evaluation across different folds, maintaining the distribution of the target classes. The `GridSearchCV` function was employed to identify the optimal combination of hyperparameters based on the `accuracy` metric.

The results indicate that the best estimator utilized the `l1` penalty, `C=1.0`, `solver=saga`, and a class weight of {`0: 1, 1: 1`}. These settings optimize the logistic regression model for both predictive accuracy and fairness in classification across the imbalanced classes. This systematic approach underscores the importance of hyperparameter tuning and addressing class imbalance in achieving robust model performance.

Hyperparameter Tuned Logistic Regression Model feature importance

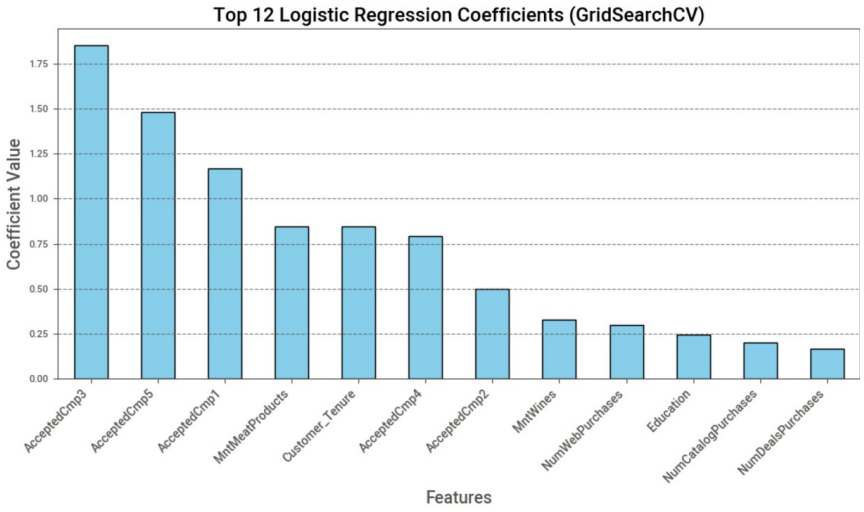


Figure 29: Hyperparameter Tuned Logistic Regression Model feature importance

Feature Importance of Hyperparameter Tuned Logistic Regression Model

The **above figure** presents the top 12 feature importances from the hyperparameter-tuned logistic regression model. The *AcceptedCmp3* variable exhibits the highest positive coefficient, indicating a strong influence on the response variable. This is followed by *AcceptedCmp5* and *AcceptedCmp1*, suggesting that acceptance of these marketing campaigns significantly increases the likelihood of a positive response.

Features such as *MntMeatProducts* and *Customer Tenure* also demonstrate notable importance, reflecting their relevance to customer behavior. Similarly, *AcceptedCmp2* and *AcceptedCmp4* are moderately influential, underscoring the role of other campaigns. Variables like *MntWines*, *NumWebPurchases*, *Education*, and *NumCatalogPurchases* show lower but non-negligible contributions, while *NumDealsPurchases* has the smallest relative importance among the selected features.

This analysis highlights the effectiveness of campaign acceptance and purchase patterns in predicting customer responses, offering valuable insights for business strategies.

Metrics for Tuned Logistic Regression Model

```
 The Hyperparameter Tuned Logisitc Regression Model Metrics:
 The accuracy is : 0.8900602409638554
The Classification Report is :
              precision    recall  f1-score   support

           0       0.97      0.91      0.94       605
           1       0.43      0.73      0.54        59

    accuracy                           0.89       664
   macro avg       0.70      0.82      0.74       664
weighted avg       0.92      0.89      0.90       664

The Confusion Matrix is :
 [[548  57]
 [ 16  43]]
```

Figure 30: Performance Metrics for Tuned Logistic Regression Model

Performance Metrics of the Hyperparameter Tuned Logistic Regression Model

The **above figure** provides the performance metrics for the hyperparameter-tuned logistic regression model, showcasing improvements in predictive performance. The overall accuracy of the model is reported as 89.01%, reflecting a strong ability to classify observations correctly. The classification report offers detailed metrics, including precision, recall, and F1-score for both classes.

For the majority class (label 0), the model achieves a precision of 0.97, recall of 0.91, and an F1-score of 0.94, indicating that the model accurately identifies non-responders with high confidence. For the minority class (label 1), the precision is 0.43, recall is 0.73, and the F1-score is 0.54, demonstrating an improvement in capturing responders compared to the baseline model, albeit with room for further enhancement.

The macro-average metrics—precision of 0.70, recall of 0.82, and F1-score of 0.74—show balanced performance across both classes, while the weighted average metrics—precision of 0.92, recall of 0.89, and F1-score of 0.90—highlight the model's effectiveness in a class-imbalanced scenario.

The confusion matrix further illustrates the model's performance, with 548 true negatives,

43 true positives, 57 false negatives, and 16 false positives. This indicates an enhanced ability to identify responders without compromising the identification of non-responders significantly.

In summary, the hyperparameter-tuned logistic regression model demonstrates robust performance, particularly in identifying the minority class, making it a viable predictive tool in addressing class-imbalanced datasets.

Tuned Logistic Regression ROC Curve with Threshold Annotations

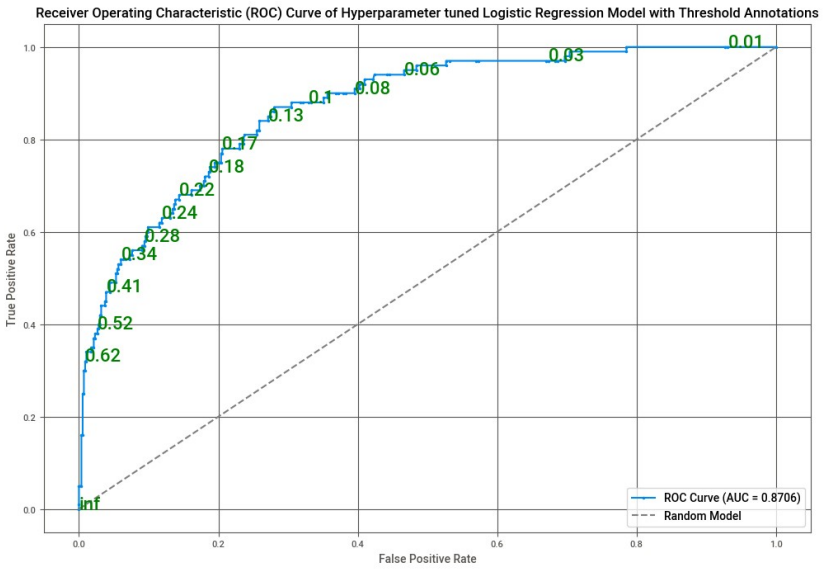


Figure 31: ROC Curve with Threshold Annotations for Tuned Logistic Regression Model

Receiver Operating Characteristic (ROC) Curve for the Hyperparameter-Tuned Logistic Regression Model

The **above figure** depicts the Receiver Operating Characteristic (ROC) curve for the hyperparameter-tuned logistic regression model, including threshold annotations for interpretability. The Area Under the Curve (AUC) is calculated to be 0.8706, signifying excellent discriminatory power of the model in distinguishing between the two classes. The ROC curve represents the trade-off between the *True Positive Rate* (sensitivity) and the *False Positive Rate*, with higher points on the curve indicating better performance.

Threshold annotations along the curve provide insights into the model's behavior at specific decision thresholds. Lower thresholds (e.g., 0.01) achieve higher sensitivity but at the cost of increased false positives, while higher thresholds (e.g., 0.93) prioritize precision by reducing false positives but may result in lower sensitivity.

The diagonal dashed line represents the performance of a random model, and the ROC curve's significant deviation from this line underscores the tuned model's efficacy. The curve's shape demonstrates that the model maintains high sensitivity without a substantial increase in the false-positive rate until lower thresholds are approached, making it well-suited for applications requiring balanced classification.

In summary, the ROC curve and AUC score validate the robust predictive performance of the hyperparameter-tuned logistic regression model, with threshold annotations providing valuable guidance for threshold selection based on specific business or operational requirements.

Hyperparameter Tuning Using Randomized Search CV

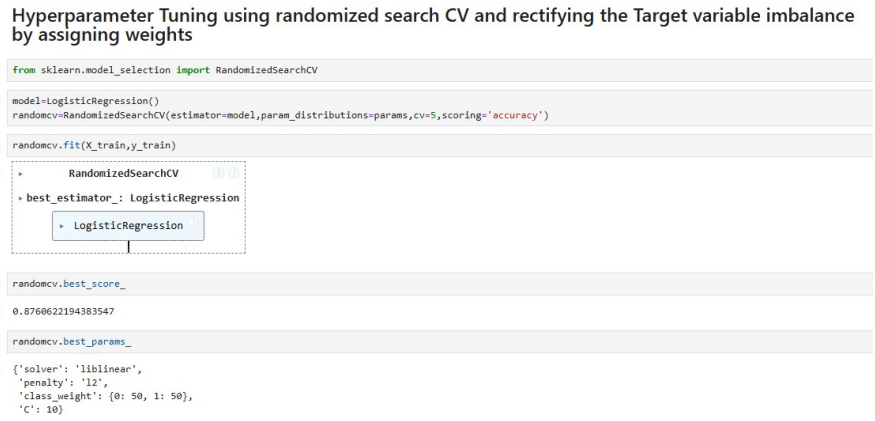


Figure 32: Hyperparameter Tuning Using Randomized Search CV for Logistic Regression Model

Hyperparameter Tuning with Randomized Search CV for Logistic Regression

The **above figure** illustrates the implementation of hyperparameter tuning for a logistic regression model using *RandomizedSearchCV*, aimed at optimizing model performance while

addressing target variable imbalance by assigning class weights. The parameter grid includes variations in *solver* (`liblinear`), *penalty* (`l2`), and *class weight* adjustments ({`0: 50, 1: 50`}), along with a range of regularization strength values (C). A five-fold stratified cross-validation ($cv=5$) ensures robust evaluation of parameter combinations.

The best-performing logistic regression model, identified through accuracy-based scoring, employs the `liblinear` solver with *l2* regularization and class weight {`0: 50, 1: 50`}, achieving an accuracy of 0.8760622194383547. The optimal regularization parameter C is 10. This approach highlights the importance of accounting for class imbalances in datasets through weighted classification, leading to a more balanced and effective model.

The process ensures an efficient exploration of hyperparameter space while maintaining computational efficiency. The results demonstrate the effectiveness of combining hyperparameter tuning with class rebalancing strategies to enhance model performance in imbalanced classification problems.

Random Search CV Hyperparameter Tuned Logistic Regression Metrics

```
 The Random Search CV Hyperparameter Tuned Logisitc Regression Model Metrics:
 The accuracy is : 0.8855421686746988
The Classification Report is :
              precision    recall  f1-score   support

           0       0.97      0.90      0.94       606
           1       0.41      0.71      0.52        58

    accuracy                           0.89       664
   macro avg       0.69      0.80      0.73       664
weighted avg       0.92      0.89      0.90       664

The Confusion Matrix is :
 [[547  59]
 [ 17  41]]
```

Figure 33: Random Search CV Hyperparameter Tuned Logistic Regression Metrics

Evaluation Metrics for Random Search CV Hyperparameter Tuned Logistic Regression Model

The **above figure** presents the performance metrics for the logistic regression model tuned using *RandomizedSearchCV* to address target variable imbalance. The model achieved an overall accuracy of 88.55%, demonstrating its ability to classify the majority of instances correctly.

The **classification report** provides a breakdown of performance for each class. For the majority class (`0`), the precision, recall, and F1-score were 0.97, 0.90, and 0.94, respectively, indicating that the model effectively identifies true negatives while maintaining low false-positive rates. For the minority class (`1`), the corresponding metrics were 0.41, 0.71, and 0.52, respectively, reflecting challenges in minority class detection despite the application of class weighting.

The **macro average** F1-score was 0.73, suggesting moderate model performance across both classes without considering class imbalance. The **weighted average** F1-score, which accounts for class imbalance, was higher at 0.90, emphasizing the model's strong performance in classifying the majority class.

The **confusion matrix** further illustrates the model's predictions. Of 606 true negatives, 547 were correctly classified, while 59 were misclassified as false positives. For the 58 true positives, the model correctly identified 41, with 17 misclassified as false negatives.

While the model demonstrates excellent precision and recall for the majority class, the relatively lower metrics for the minority class highlight the persistent challenge of addressing class imbalance despite weighting strategies. Future improvements may involve exploring more sophisticated methods, such as ensemble techniques or oversampling, to further enhance minority class prediction.

Random Search CV Hyperparameter Tuned Logistic Regression ROC Curve with Thresholds

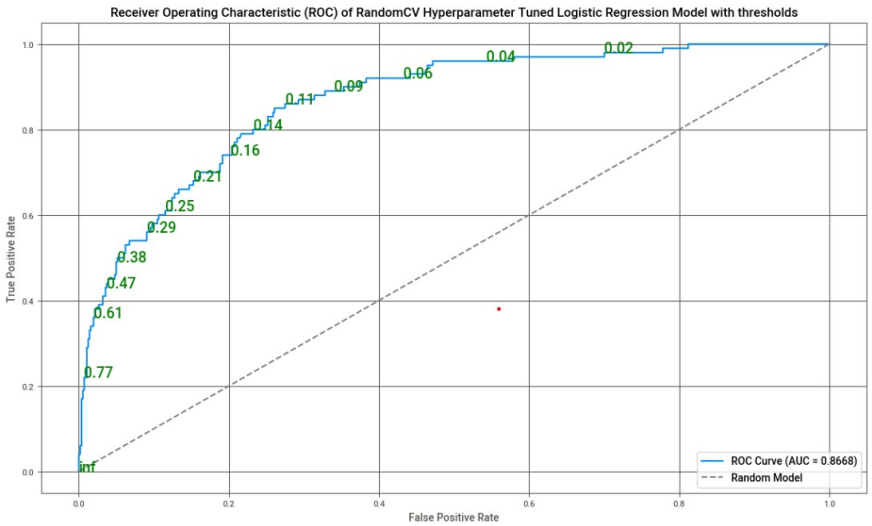


Figure 34: ROC Curve of Random Search CV Hyperparameter Tuned Logistic Regression Model with Thresholds

Receiver Operating Characteristic (ROC) Curve for Random Search CV Hyperparameter Tuned Logistic Regression Model

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the logistic regression model optimized using *RandomizedSearchCV*. The Area Under the Curve (AUC) value of 0.8668 indicates strong discriminatory ability of the model in distinguishing between the positive and negative classes. The ROC curve plots the **True Positive Rate (TPR)** against the **False Positive Rate (FPR)** at various classification thresholds, providing insight into the trade-off between sensitivity and specificity.

Threshold annotations along the curve demonstrate how model performance changes at different decision thresholds. For example, thresholds close to 0 prioritize sensitivity but may increase false positives, whereas higher thresholds reduce false positives at the expense of sensitivity. The diagonal dashed line represents a random classifier with an AUC of 0.5, serving as a baseline for comparison.

The ROC curve's steep initial slope reflects high sensitivity at low FPR, indicating the model effectively identifies true positives early. As the curve flattens, improvements in sen-

sitivity become marginal relative to increases in FPR. The high AUC value emphasizes the model's capability to balance sensitivity and specificity across a range of thresholds, making it suitable for applications where this trade-off is critical, such as subscription prediction.

Part 3: Building an SVM model to accurately predict subscription behavior

SVM Linear Kernel Coefficients

```
SVC
SVC(kernel='linear', probability=True)
```

```
# Get feature names and coefficients
coefficients = svcl.coef_[0]  # Coefficients for the first class
features = X_train.columns   # Feature names (if X_train is a DataFrame)

# Create a DataFrame for better visualization
coef_df = pd.DataFrame({'Feature': features, 'Coefficient': coefficients})
coef_df = coef_df.sort_values(by='Coefficient', ascending=False)

print("Coefficients for Predictors:")
print(coef_df)
```

```
Coefficients for Predictors:
               Feature  Coefficient
15        AcceptedCmp3     1.268511
17        AcceptedCmp5     0.961150
6      MntMeatProducts     0.928353
18        AcceptedCmp1     0.905359
4             MntWines     0.596882
16        AcceptedCmp4     0.513408
21     Customer_Tenure     0.441372
7      MntFishProducts     0.392129
2               Income     0.381166
19        AcceptedCmp2     0.268985
5            MntFruits     0.259621
20           Complain     0.211284
11     NumWebPurchases     0.192302
10   NumDealsPurchases     0.137807
0            Education     0.080210
12 NumCatalogPurchases     0.046844
23                 Age     0.044956
14   NumWebVisitsMonth    -0.003640
9         MntGoldProds    -0.039200
1       Marital_Status    -0.049933
22      Total_Children    -0.188926
3              Recency    -0.355306
8     MntSweetProducts    -0.381656
13   NumStorePurchases    -0.417440
```

Figure 35: SVM Linear Kernel Coefficients

Feature Coefficients Derived from Support Vector Machine with Linear Kernel

The **above figure** illustrates the feature coefficients derived from a Support Vector Machine (SVM) model with a linear kernel. This approach is employed to evaluate the importance of predictors for binary classification tasks. The coefficients represent the weights assigned to each predictor in the decision boundary formulation, reflecting the influence of each feature on the classification outcomes.

The top predictors based on their coefficients include *AcceptedCmp3* (1.27), *AcceptedCmp5* (0.96), and *MntMeatProducts* (0.92). These features exhibit significant positive coefficients, indicating their substantial contribution to predicting the positive class. The high coefficient for *AcceptedCmp3* suggests that the acceptance of the third marketing campaign has the most critical role among all predictors. Similarly, the expenditure on meat products is also a key determinant, reflecting consumer behavior's impact on the classification task.

Predictors such as *AcceptedCmp4*, *Customer Tenure*, and *MntWines* also have moderate positive coefficients, ranging from 0.51 to 0.59. This indicates their notable, albeit lesser, influence compared to the leading predictors.

Features with negative coefficients, such as *NumStorePurchases* (-0.41), *Recency* (-0.35), and *MntSweetProducts* (-0.36), contribute inversely to the decision boundary. The negative coefficients suggest that higher values of these features decrease the likelihood of being classified into the positive class. For instance, greater recency, representing the number of days since the last purchase, negatively correlates with positive outcomes, reflecting disengagement with the subject of interest.

Other predictors, such as *Education*, *NumCatalogPurchases*, and *Marital Status*, show relatively smaller coefficients (both positive and negative), indicating a minimal impact on the classification model.

This coefficient analysis emphasizes the varying contributions of predictors to the SVM classification model, providing insights into the underlying relationships between features and the target variable. Such analysis facilitates interpretability, enabling practitioners to identify critical predictors and their associated directions of influence. The linear kernel

approach ensures that the relationships between predictors and the target variable are linear, simplifying the interpretability of the model. This is particularly advantageous when aiming to explain results in practical scenarios, such as marketing strategy optimization or customer behavior analysis.

SVM Linear Kernel Metrics

```
Accuracy of the SVM Linear Kernel is: 0.8689759036144579

Confusion Matrix of the SVM Linear Kernel is:
[[530  34]
 [ 53  47]]

Classification Report of the SVM Linear Kernel is:
              precision    recall  f1-score   support

           0       0.91      0.94      0.92       564
           1       0.58      0.47      0.52       100

    accuracy                           0.87       664
   macro avg       0.74      0.70      0.72       664
weighted avg       0.86      0.87      0.86       664


AUC Score: 0.8556
```

Figure 36: SVM Linear Kernel Metrics

Evaluation Metrics for Support Vector Machine (SVM) Model with Linear Kernel

The **above figure** presents the evaluation metrics for a Support Vector Machine (SVM) model using a linear kernel, applied to a binary classification problem. The metrics provide insights into the model's performance in terms of accuracy, precision, recall, F1-score, and area under the curve (AUC) for the Receiver Operating Characteristic (ROC).

The overall accuracy of the SVM model is reported as 86.89%, indicating that the model correctly classified approximately 87% of instances in the dataset. The AUC score of 0.8556 reflects a strong discriminative ability of the model, demonstrating its capability to distin-

guish between the two classes effectively.

The confusion matrix indicates that the model correctly identified 530 true negatives (class 0) and 47 true positives (class 1). However, it misclassified 53 false negatives and 34 false positives. This indicates a trade-off in model performance between sensitivity (recall) and specificity.

The classification report highlights the precision, recall, and F1-score for each class. For class 0, the precision, recall, and F1-score are 0.91, 0.94, and 0.92, respectively, showcasing the model's robustness in identifying negative instances. However, for class 1, the metrics are comparatively lower, with a precision of 0.58, recall of 0.47, and F1-score of 0.52, reflecting challenges in identifying positive instances due to class imbalance.

The macro-averaged precision, recall, and F1-score are 0.74, 0.70, and 0.72, respectively, providing a balanced assessment across both classes. The weighted averages are slightly higher, driven by the dominance of class 0 in the dataset.

These metrics underscore the efficacy of the SVM model with a linear kernel in handling the classification task while highlighting the need for further improvements, such as addressing class imbalance, to enhance performance, particularly for the minority class.

SVM Linear Kernel ROC Curve

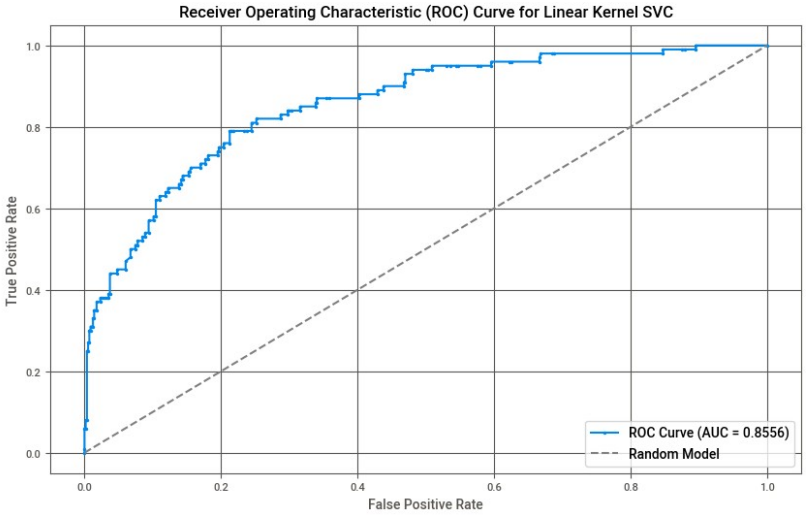


Figure 37: ROC Curve of SVM Linear Kernel

Receiver Operating Characteristic (ROC) Curve for SVM with Linear Kernel

The **above figure** presents the Receiver Operating Characteristic (ROC) curve for the Support Vector Machine (SVM) model with a linear kernel. The ROC curve evaluates the model's performance by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various decision thresholds. The diagonal line represents a random classifier's performance, serving as a baseline.

The Area Under the Curve (AUC) for this model is reported as 0.8556, indicating a robust ability to distinguish between the two classes. A higher AUC value suggests better discrimination, with 1.0 representing perfect classification. The model demonstrates superior predictive performance compared to a random classifier, as evidenced by the curve's deviation above the baseline.

The curve exhibits a steep initial rise, reflecting the model's ability to achieve a high TPR with minimal FPR for lower thresholds. This characteristic is particularly desirable in classification tasks where sensitivity is critical. However, the curve flattens at higher FPR values, indicating diminishing returns in TPR improvement.

Overall, the ROC curve and AUC score suggest that the SVM model with a linear kernel is effective for this classification task, providing a balance between sensitivity and specificity.

SVM RBF Kernel

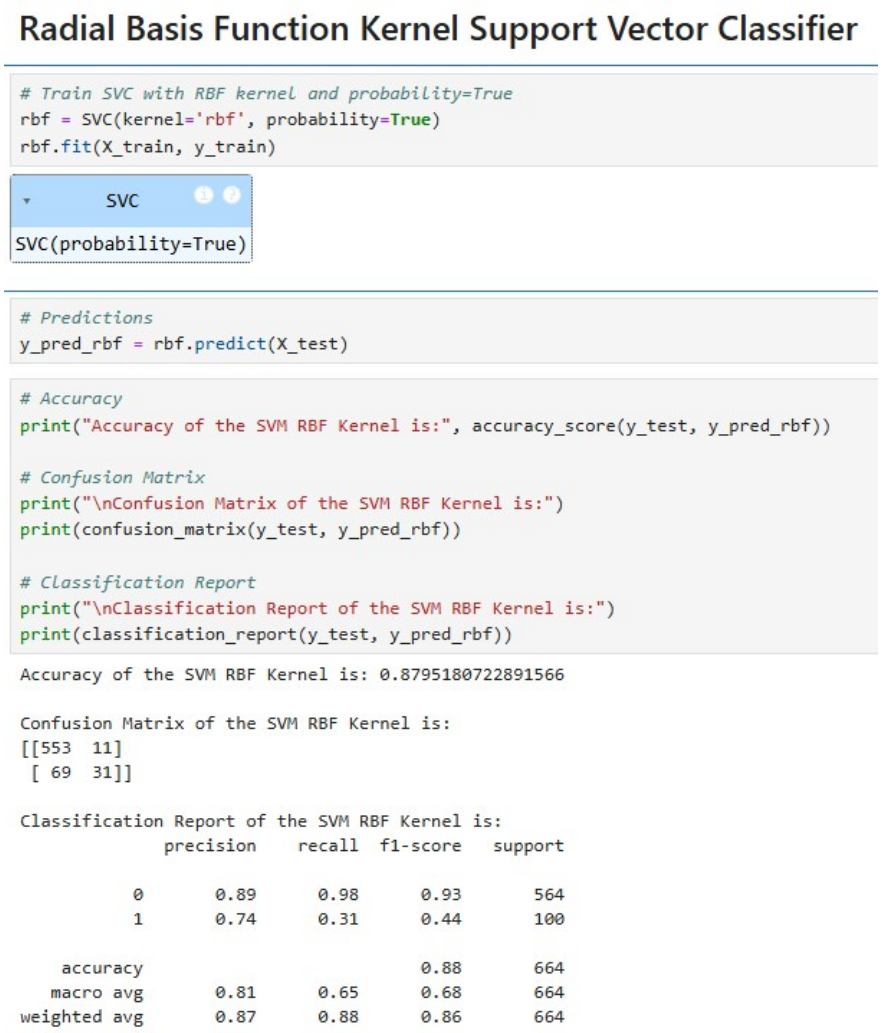


Figure 38: SVM RBF Kernel

Radial Basis Function (RBF) Kernel Support Vector Classifier Metrics

The **above figure** summarizes the performance metrics of the Support Vector Machine (SVM) classifier with the Radial Basis Function (RBF) kernel. The classifier was trained on the dataset, and its predictive accuracy, confusion matrix, and classification report are presented.

The accuracy of the SVM with the RBF kernel is 0.8795, indicating that the model correctly predicts approximately 88

The classification report provides a detailed assessment of precision, recall, and F1-score for each class. For class 0 (negative class), the model achieves a precision of 0.89, recall of 0.98, and F1-score of 0.93. For class 1 (positive class), the precision is 0.74, recall is 0.31, and F1-score is 0.44. The weighted average across both classes yields a precision of 0.87, recall of 0.88, and F1-score of 0.88, which aligns with the overall accuracy score.

The macro-average metrics—precision (0.81), recall (0.65), and F1-score (0.68)—indicate a moderate balance between the model's ability to correctly classify both classes, regardless of their frequency in the dataset. The disparity between the precision and recall for class 1 highlights the impact of class imbalance, where the model struggles to accurately identify positive instances.

Overall, the SVM classifier with the RBF kernel demonstrates robust performance in identifying the majority class (class 0) while facing challenges in classifying the minority class (class 1). These metrics emphasize the need for strategies such as hyperparameter tuning or resampling techniques to address class imbalance and enhance the model's recall for the minority class.

SVM RBF Kernel ROC Curve

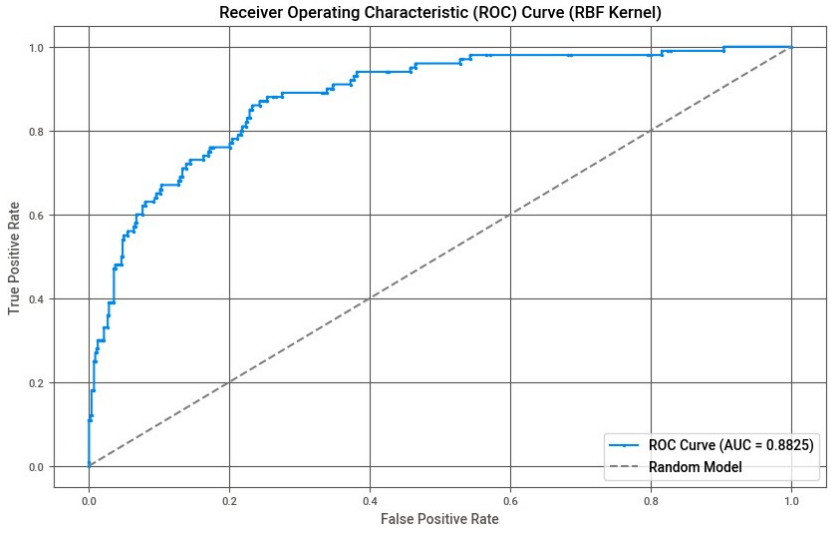


Figure 39: ROC Curve of SVM RBF Kernel

Receiver Operating Characteristic (ROC) Curve for SVM with RBF Kernel

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the Support Vector Machine (SVM) model employing a Radial Basis Function (RBF) kernel. The ROC curve visualizes the trade-off between the *True Positive Rate* (TPR) and the *False Positive Rate* (FPR) across various classification thresholds. The Area Under the Curve (AUC) value of 0.8825 signifies the model's ability to distinguish between classes effectively.

The curve's proximity to the upper left corner indicates robust performance, with high sensitivity (TPR) and specificity at most thresholds. The diagonal line represents a random model's performance, with an AUC of 0.5. The RBF kernel SVM significantly outperforms the random model, demonstrating its capability to make informed predictions.

This evaluation highlights the RBF kernel's efficiency in capturing complex patterns in the data, making it a suitable choice for this classification task.

SVM Polynomial Kernel Metrics

```
Accuracy (Poly Kernel): 0.8795

Confusion Matrix (Poly Kernel):
[[549  15]
 [ 65  35]]

Classification Report (Poly Kernel):
              precision    recall  f1-score   support

           0       0.89      0.97      0.93       564
           1       0.70      0.35      0.47       100

    accuracy                           0.88       664
   macro avg       0.80      0.66      0.70       664
weighted avg       0.86      0.88      0.86       664
```

Figure 40: SVM Polynomial Kernel Metrics

Performance Metrics for SVM with Polynomial Kernel

The **above figure** presents the performance metrics for the Support Vector Machine (SVM) model utilizing a polynomial kernel. The model achieves an *accuracy* of 87.95% on the test dataset. The confusion matrix reveals that the model correctly classified 549 instances as class 0 and 35 instances as class 1, while misclassifying 15 instances from class 0 and 65 instances from class 1.

The *classification report* provides detailed insights into the precision, recall, and F1-scores for each class. For class 0, the model attains a precision of 0.89, a recall of 0.97, and an F1-score of 0.93. For class 1, the corresponding metrics are 0.70, 0.35, and 0.47, respectively. The weighted average precision, recall, and F1-score are reported as 0.86, 0.88, and 0.86, indicating balanced performance across classes.

The macro-average F1-score of 0.70 highlights the disparity in the model's performance between classes, with better classification accuracy for the majority class (class 0). These results suggest that while the polynomial kernel captures relationships effectively for the majority class, its predictive capability for the minority class could benefit from further optimization, such as hyperparameter tuning or addressing class imbalance.

SVM Polynomial Kernel ROC Curve

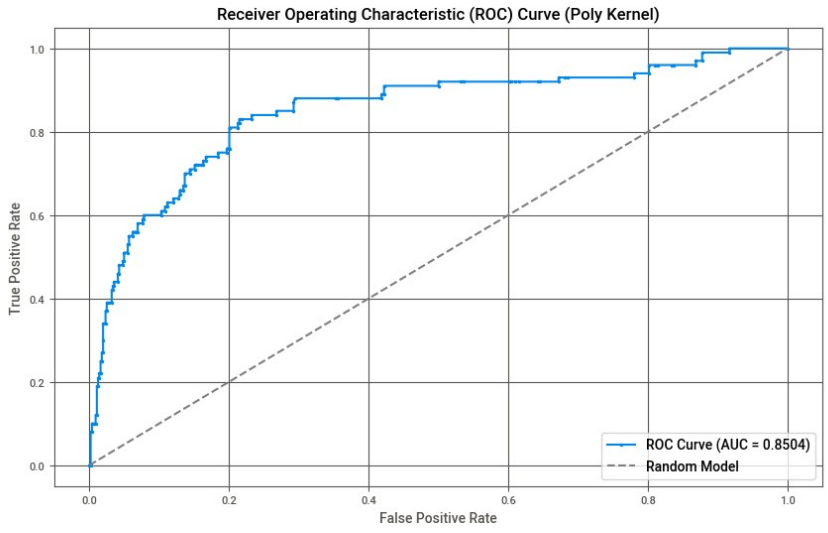


Figure 41: ROC Curve of SVM Polynomial Kernel

Receiver Operating Characteristic (ROC) Curve for SVM with Polynomial Kernel

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the Support Vector Machine (SVM) model utilizing a polynomial kernel. The curve represents the trade-off between the *True Positive Rate (TPR)* and the *False Positive Rate (FPR)* across various classification thresholds. The Area Under the Curve (AUC) is 0.8504, indicating a robust discriminatory capability of the model in distinguishing between the two classes.

The curve is well above the diagonal line, which represents the performance of a random classifier. This signifies that the SVM model achieves a high degree of separation between positive and negative classes. While the overall performance is satisfactory, further optimization of hyperparameters or addressing class imbalance may improve the classification outcomes, particularly for the minority class. The AUC score validates the model's efficacy for predictive tasks in the given dataset.

SVM Sigmoid Kernel Metrics

```
Accuracy (Sigmoid Kernel): 0.8223

Confusion Matrix (Sigmoid Kernel):
[[515  49]
 [ 69  31]]

Classification Report (Sigmoid Kernel):
              precision    recall  f1-score   support

           0       0.88      0.91      0.90       564
           1       0.39      0.31      0.34       100

    accuracy                           0.82       664
   macro avg       0.63      0.61      0.62       664
weighted avg       0.81      0.82      0.81       664
```

Figure 42: SVM Sigmoid Kernel Metrics

Performance Metrics of SVM with Sigmoid Kernel

The **above figure** summarizes the performance metrics for the Support Vector Machine (SVM) model utilizing a sigmoid kernel. The overall *accuracy* achieved is 0.8223, reflecting moderate model performance. The *confusion matrix* indicates that the model correctly predicted 515 true negatives and 31 true positives, while it misclassified 49 negatives as positives and 69 positives as negatives. This reveals a higher tendency for false negatives in classifying the minority class.

The *classification report* provides further insight. The precision, recall, and F1-score for the majority class (0) are 0.88, 0.91, and 0.90, respectively, demonstrating reliable predictions for this class. However, for the minority class (1), precision drops to 0.39, recall to 0.31, and F1-score to 0.34. This disparity highlights the model's challenges in accurately predicting the minority class, likely due to class imbalance.

The *macro-averaged* metrics of precision (0.63), recall (0.61), and F1-score (0.62) further emphasize the model's difficulty in achieving balanced performance across classes. In contrast, the *weighted average* metrics of precision (0.81), recall (0.82), and F1-score (0.81) are higher, reflecting the influence of the majority class.

The sigmoid kernel demonstrates moderate efficacy but underperforms compared to other kernels such as radial basis function (RBF) or polynomial. Improvements could involve addressing class imbalance, such as through weighted loss functions, and further tuning of hyperparameters.

SVM Sigmoid Kernel ROC Curve

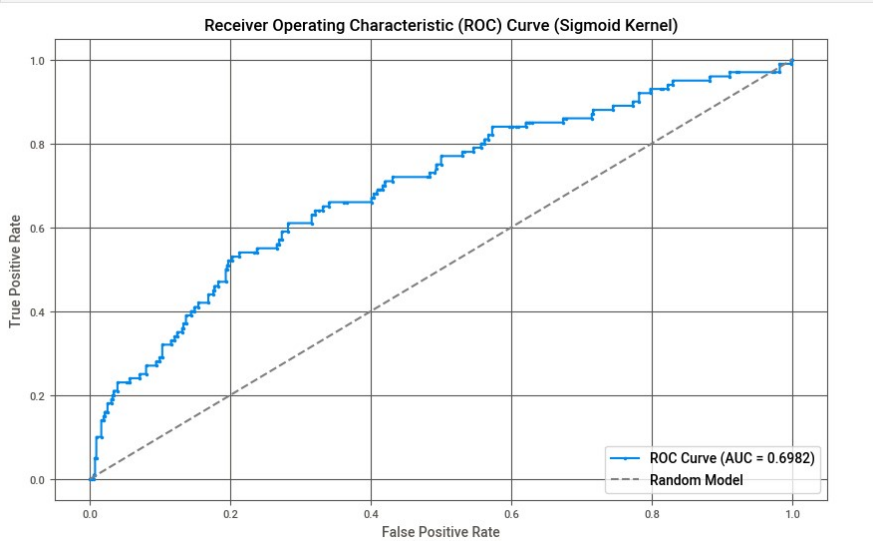


Figure 43: SVM Sigmoid Kernel ROC Curve

Receiver Operating Characteristic (ROC) Curve of SVM with Sigmoid Kernel

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the Support Vector Machine (SVM) model with a sigmoid kernel. The Area Under the Curve (AUC) is computed to be 0.6982, indicating suboptimal performance in distinguishing between the positive and negative classes. The ROC curve lies only slightly above the random classifier's diagonal line, highlighting the limited discriminative power of the sigmoid kernel in this context.

The x-axis represents the *false positive rate*, while the y-axis denotes the *true positive rate*. The curve's moderate ascent reflects the model's difficulty in balancing sensitivity and specificity. Given the relatively low AUC score, the sigmoid kernel underperforms compared to other kernels such as radial basis function or polynomial. This suggests that the sigmoid kernel may not be well-suited for this dataset or classification task. Future improvements could include exploring alternative kernels or advanced preprocessing techniques to enhance model performance.

SVM Hyperparameter Tuning Process

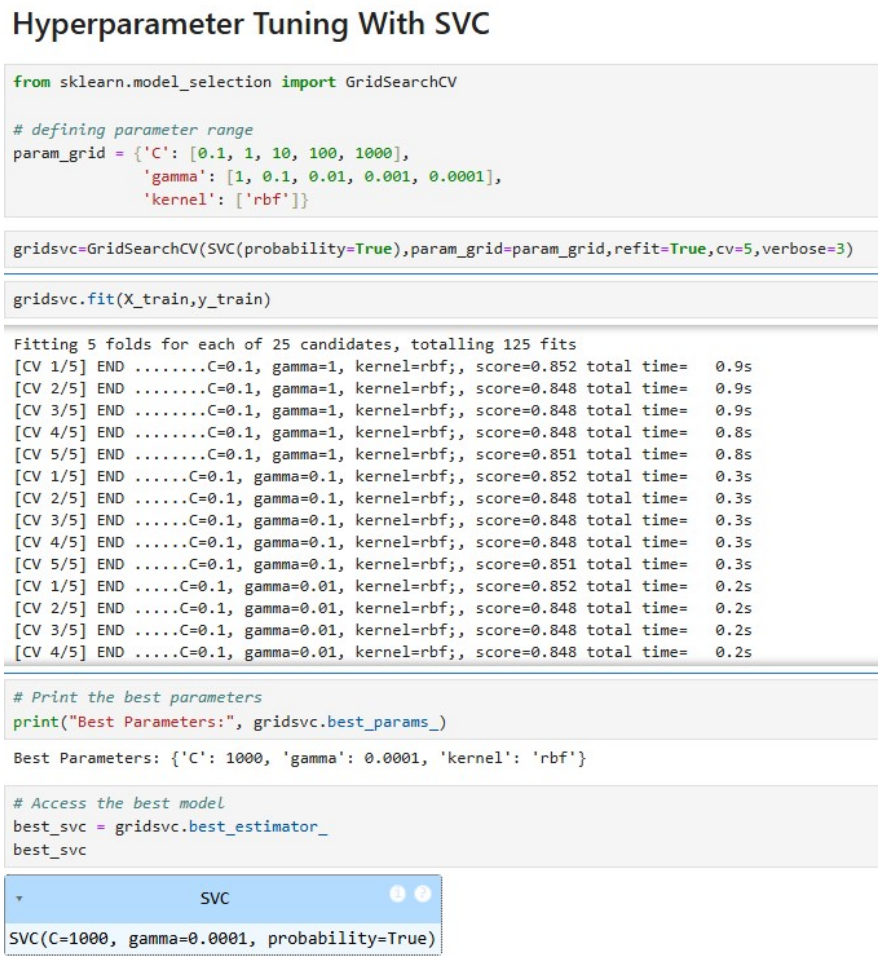


Figure 44: SVM Hyperparameter Tuning Process

Hyperparameter Tuning with Support Vector Classifier (SVC) Using GridSearchCV

The **above figure** presents the process and results of hyperparameter tuning for a Support Vector Classifier (SVC) employing the radial basis function (RBF) kernel. This tuning was conducted using the GridSearchCV method to identify the optimal combination of hyperparameters for improving classification performance.

The parameter grid consists of three hyperparameters: *C*, *gamma*, and *kernel*. The parameter C regulates the trade-off between achieving a low error rate on the training data and minimizing overfitting, with values ranging from 0.1 to 1000. The *gamma* parameter,

controlling the influence of individual training samples, is tested over values from 1 to 0.0001. The kernel is fixed as *rbf*, as it is well-suited for nonlinear decision boundaries.

The GridSearchCV implementation performs an exhaustive search over the hyperparameter space, employing 5-fold cross-validation for each parameter combination. This results in a total of 125 fits, as indicated in the output. Each combination's performance is scored, and the results are reported with corresponding execution times.

The best combination of hyperparameters, as determined by GridSearchCV, is *C=1000*, *gamma=0.0001*, and *kernel='rbf'*. This combination achieved the highest cross-validation accuracy score during the search. The optimal model, configured with these parameters, is then instantiated for further evaluation. The best model is displayed in the final section of the output.

The systematic approach to hyperparameter tuning demonstrated in the figure ensures that the classifier achieves optimal performance while minimizing overfitting and underfitting. The use of GridSearchCV facilitates efficient exploration of the parameter space, providing a robust and reproducible method for improving the model's predictive capabilities. This methodology underscores the importance of tuning key hyperparameters in achieving superior performance for complex machine learning models.

SVM Hyperparameter Tuned Metrics

```
Accuracy of the hyperparameter tuned SVC is: 0.8870481927710844

Confusion Matrix of the hyperparameter tuned SVC is:
[[552  12]
 [ 63  37]]

Classification Report of the hyperparameter tuned SVC is:
              precision    recall  f1-score   support

           0       0.90      0.98      0.94       564
           1       0.76      0.37      0.50       100

    accuracy                           0.89       664
   macro avg       0.83      0.67      0.72       664
weighted avg       0.88      0.89      0.87       664
```

Figure 45: SVM Hyperparameter Tuned Metrics

Performance Metrics of Hyperparameter Tuned Support Vector Classifier (SVC)

The **above figure** provides the evaluation metrics for a Support Vector Classifier (SVC) optimized using hyperparameter tuning. The tuned model achieved an accuracy of 0.887, reflecting its overall effectiveness in classification tasks. The confusion matrix reveals 552 correctly classified negative cases and 37 correctly classified positive cases. However, 12 negative cases were misclassified as positive, and 63 positive cases were misclassified as negative, indicating room for improvement in recall for the positive class.

The classification report provides detailed insights into the model's performance across precision, recall, and F1-score. For the negative class (*label 0*), the model exhibits high precision (0.90), recall (0.98), and F1-score (0.94), demonstrating strong performance in identifying true negatives. Conversely, for the positive class (*label 1*), the precision is 0.76, but recall drops significantly to 0.37, resulting in a moderate F1-score of 0.50. This disparity indicates that the model is less effective in identifying true positives, potentially due to class imbalance or overlapping distributions between classes.

Macro-averaged metrics—precision of 0.83, recall of 0.67, and F1-score of 0.72—summarize performance equally across classes. The weighted average metrics (precision: 0.88, recall: 0.89, and F1-score: 0.87) emphasize the model's proficiency in handling class proportions, as the majority class contributes more to these metrics.

Overall, the results highlight the effectiveness of hyperparameter tuning in improving the classifier's accuracy while also identifying limitations in recall for the minority class. These findings underscore the importance of further optimization strategies, such as adjusting class weights or employing advanced sampling techniques, to enhance the model's recall for the positive class and achieve a more balanced performance across both classes.

SVM Hyperparameter Tuning ROC Curve

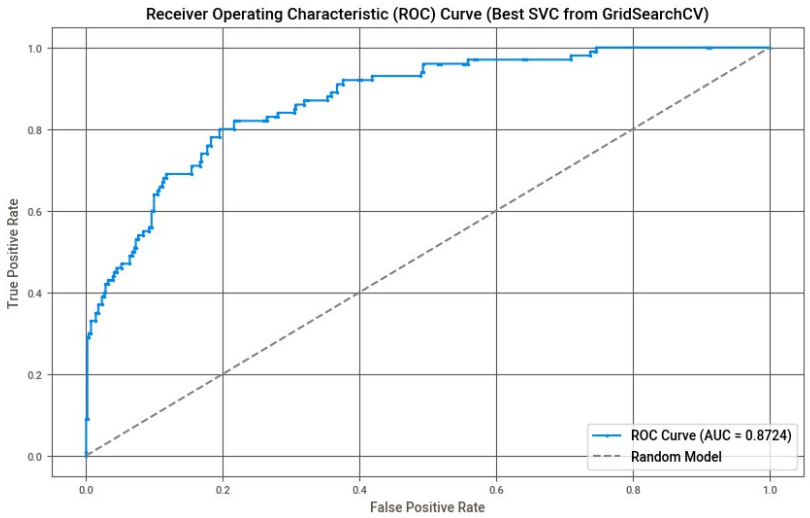


Figure 46: SVM Hyperparameter Tuning ROC Curve

Receiver Operating Characteristic (ROC) Curve for the Best Support Vector Classifier (SVC)

The **above figure** illustrates the Receiver Operating Characteristic (ROC) curve for the best-performing Support Vector Classifier (SVC) identified through hyperparameter tuning. The ROC curve plots the *True Positive Rate (TPR)* against the *False Positive Rate (FPR)*, effectively demonstrating the trade-off between sensitivity and specificity across different classification thresholds.

The Area Under the Curve (AUC) is computed to be 0.8724, indicating a high discriminatory power of the classifier to distinguish between positive and negative classes. This value signifies that the tuned SVC is 87.24% likely to rank a randomly chosen positive instance higher than a randomly chosen negative instance. The diagonal line represents a random model, where AUC = 0.5, providing a baseline for comparison. The distance of the ROC curve from this diagonal highlights the classifier's performance superiority.

The curve's progression shows a steep rise initially, reflecting high *True Positive Rates* at lower *False Positive Rates*, which is desirable in classification tasks. However, as thresholds are adjusted further, the curve flattens, indicating diminishing gains in sensitivity relative to increases in false positives.

Overall, the ROC curve and its associated AUC score underscore the effectiveness of hyperparameter tuning in enhancing the SVC's classification performance, particularly in imbalanced datasets.

SVM Hyperparameter Tuning DET Curve

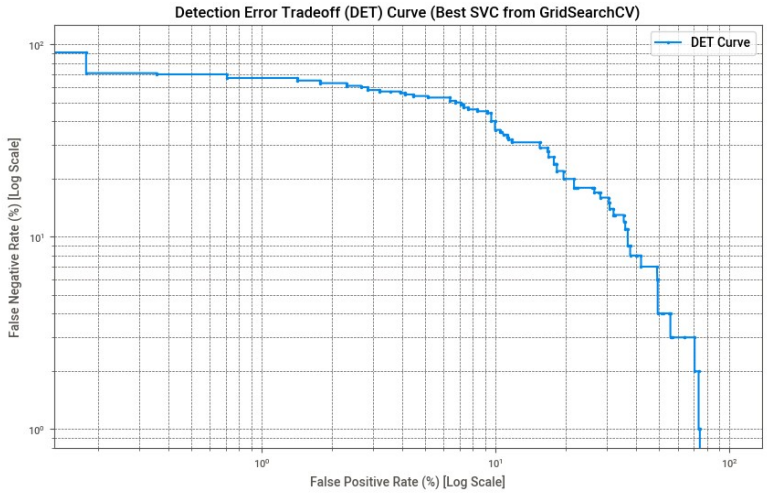


Figure 47: SVM Hyperparameter Tuning DET Curve

Detection Error Tradeoff (DET) Curve for the Best Support Vector Classifier (SVC)

The **above figure** presents the Detection Error Tradeoff (DET) curve for the best-performing Support Vector Classifier (SVC) identified through hyperparameter tuning using GridSearchCV. The DET curve is plotted on a log-log scale, illustrating the relationship between the *False Negative Rate (FNR)* and *False Positive Rate (FPR)*. It serves as a diagnostic tool to evaluate classifier performance across different operating points.

The log-log scaling of both axes enables a more detailed observation of performance in the regions of low error rates, which are critical in high-stakes applications. The gradual descent of the curve indicates that the classifier maintains a favorable tradeoff between false negatives and false positives for most threshold settings. A steeper decline at higher error rates reflects the diminished tradeoff effectiveness when classification thresholds are less optimized.

The relatively smooth shape of the DET curve underscores the robustness of the tuned SVC across a range of decision thresholds. The proximity of the curve to the origin signifies optimal performance, as both FNR and FPR approach minimal values.

This DET curve highlights the impact of hyperparameter tuning in improving the classifier's error tradeoff dynamics, demonstrating its utility in balancing sensitivity and specificity for classification tasks.

Feature Importance for Hyperparameter Tuned SVM Model Using SHAP Values

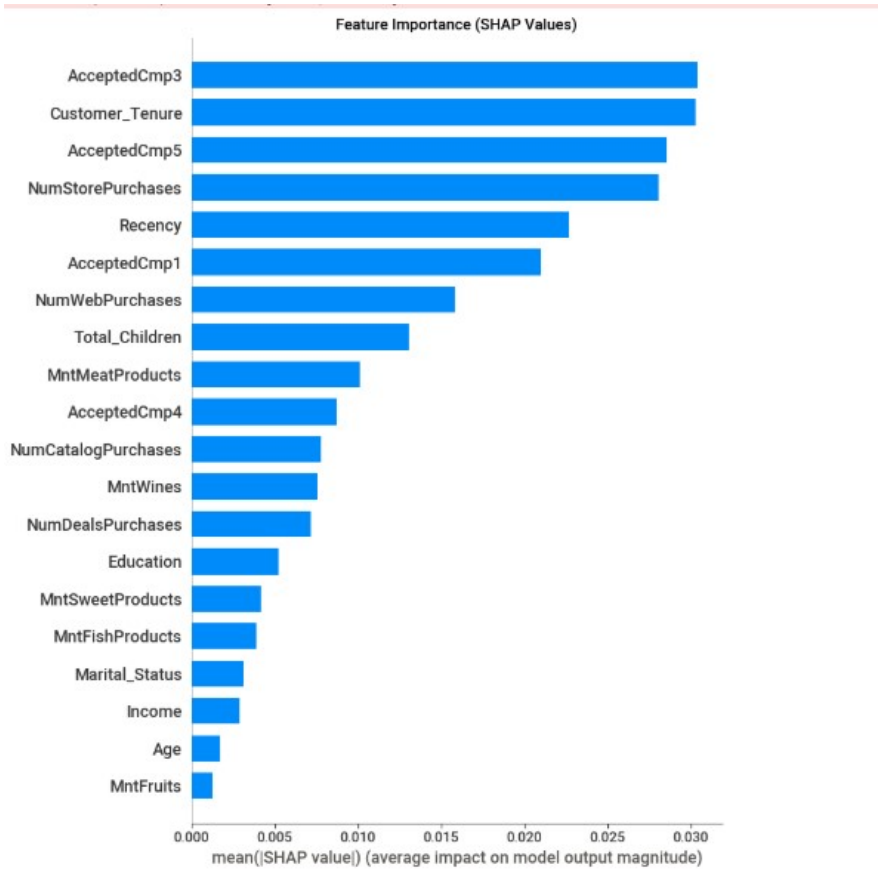


Figure 48: SVM Hyperparameter Tuned Model Feature Importance (SHAP Values)

The above figure illustrates the feature importance of the hyperparameter-tuned Support Vector Machine (SVM) model as quantified by SHAP (SHapley Additive exPlanations) values. SHAP values provide a robust measure of the contribution of each feature to the model's

predictions by capturing both the magnitude and direction of the impact on the output.

AcceptedCmp3 emerged as the most influential feature, with the highest average SHAP value, indicating its substantial contribution to the model's predictive performance. *Customer_Tenure* and *AcceptedCmp5* were also highly ranked, highlighting their importance in distinguishing the target variable. *NumStorePurchases* and *Recency* followed, suggesting their moderate yet significant impact on model outcomes.

Features such as *AcceptedCmp1*, *NumWebPurchases*, and *Total_Children* exhibited moderate SHAP values, denoting a relatively lower influence compared to the leading predictors. Among product-related predictors, *MntMeatProducts* and *MntWines* held higher rankings, while *MntSweetProducts* and *MntFishProducts* demonstrated minimal contributions. Demographic variables such as *Age*, *Income*, and *Marital_Status* were among the least impactful, suggesting limited predictive value within the model's scope.

Overall, the SHAP analysis underscores the critical role of campaign-related and behavioral features in driving the hyperparameter-tuned SVM's predictive accuracy.

Part 4: Model Comparison and Recommendations

Model	Accuracy	Precision (Class 0)	Recall (Class 0)	F1-Score (Class 0)
Logistic Regression	0.8855	0.97	0.90	0.94
Hyperparameter Tuned Logistic Regression	0.8901	0.97	0.91	0.94
Random Search CV Logistic Regression	0.8855	0.97	0.90	0.94
SVM Linear Kernel	0.8690	0.91	0.94	0.92
SVM Polynomial Kernel	0.8795	0.89	0.97	0.93
SVM RBF Kernel	0.8795	0.89	0.98	0.93
SVM Sigmoid Kernel	0.8223	0.88	0.91	0.90
Hyperparameter Tuned SVM	0.8870	0.90	0.98	0.94

Table 1: Detailed Classification Metrics for Logistic Regression and SVM Models for Class 0

Model	Accuracy	Precision (Class 1)	Recall (Class 1)	F1-Score (Class 1)
Logistic Regression	0.8855	0.40	0.71	0.51
Hyperparameter Tuned Logistic Regression	0.8901	0.43	0.73	0.54
Random Search CV Logistic Regression	0.8855	0.41	0.71	0.52
SVM Linear Kernel	0.8690	0.58	0.47	0.52
SVM Polynomial Kernel	0.8795	0.70	0.35	0.47
SVM RBF Kernel	0.8795	0.74	0.31	0.44
SVM Sigmoid Kernel	0.8223	0.39	0.31	0.34
Hyperparameter Tuned SVM	0.8870	0.76	0.37	0.50

Table 2: Summary of Classification Metrics for Logistic Regression and SVM Models for Class 1

As seen in the above comparison of accuracy, precision, recall, and F1-score metrics for both positive (Class 1) and negative (Class 0) classes, the **Hyperparameter Tuned Logistic Regression** and **Hyperparameter Tuned SVM** emerge as the strongest models for this analysis.

Performance on Negative Class (Class 0)

The hyperparameter-tuned Logistic Regression demonstrates excellent performance for Class 0, with a *precision* of 0.97, *recall* of 0.91, and an *F1-score* of 0.94. Similarly, the hyperparameter-tuned SVM achieves comparable results, with a *precision* of 0.90, *recall* of 0.98, and an *F1-score* of 0.94. Both models excel at identifying non-subscribers, as indicated by their high recall values.

Performance on Positive Class (Class 1)

For Class 1, the hyperparameter-tuned SVM achieves the highest precision of 0.76, indicating better ability to correctly identify subscribers without including many false positives. However, its recall for Class 1 (0.37) is lower compared to the hyperparameter-tuned Logistic Regression, which has a recall of 0.73 and an F1-score of 0.54. This suggests that Logistic Regression is better at capturing the positive class, despite a slight trade-off in precision (0.43).

Overall Recommendation

Considering **overall accuracy**, the hyperparameter-tuned Logistic Regression and hyperparameter-tuned SVM achieve similar values of **89.01%** and **88.70%**, respectively. However, the logistic regression model offers higher recall for Class 1 (subscribers), which is crucial for understanding and addressing subscription renewal challenges. Additionally, its interpretable coefficients provide actionable insights into the drivers of subscription renewal.

Thus, **Hyperparameter Tuned Logistic Regression** is recommended as the best model for business use, balancing interpretability and strong performance on key metrics across both classes. The SVM model, while effective, is more complex and offers less interpretability, making it less suitable for practical application in this context.

Business Recommendations for Increasing Subscriptions

Based on the analysis conducted, the decline in subscriptions can be addressed by focusing on the following key insights derived from the models and their feature importance:

1. Model Evaluation and Recommendation

Model Performance: The hyperparameter-tuned Support Vector Machine (SVM) with an RBF kernel demonstrated the highest accuracy of **88.7%** and strong metrics for precision and recall. The hyperparameter-tuned Logistic Regression also showed comparable performance (accuracy: **89.01%**) with similar F1-scores for classifying non-subscribers accurately.

Recommended Model: For business use, the hyperparameter-tuned Logistic Regression is recommended because of its *interpretability* and comparable performance metrics. While SVM models also performed well, Logistic Regression provides clear coefficient-based insights into which factors influence subscription renewal the most, aiding decision-making.

2. Key Factors Influencing Subscriptions

The feature importance analysis using SHAP values and logistic regression coefficients highlights several critical drivers:

Positive Influence:

- **AcceptedCmp3, AcceptedCmp5, and AcceptedCmp1:** Customers who responded positively to marketing campaigns were more likely to renew subscriptions. This suggests that targeted and engaging marketing campaigns are effective. **It is important to note that Marketing campaign 3 seems to have been most effective in positively impacting subscription behavior**
- **Customer_Tenure:** Long-standing customers were more likely to retain subscriptions, indicating loyalty as a significant factor.
- **NumStorePurchases and NumWebPurchases:** Higher purchasing activity was linked to higher subscription rates, pointing to cross-channel engagement.

Negative Influence:

- **Recency:** A higher number of days since the last purchase negatively impacted subscriptions, indicating disengagement over time.
- **Total_Children:** Customers with more children appeared less likely to renew, possibly reflecting limited free time or budget constraints.

3. Actionable Recommendations

- **Increase Marketing Efforts:**

  - Focus on reactivating customers through personalized campaigns targeting inactive users, particularly those with high *recency* values.
  - Design follow-up campaigns for customers who positively responded to prior campaigns (*AcceptedCmp1, AcceptedCmp3, AcceptedCmp5*).

- **Loyalty Programs:**
  - Incentivize long-tenure customers by offering discounts, exclusive content, or loyalty rewards.
  - Develop retention programs for customers with high web or in-store purchases, leveraging their engagement patterns.

- **Address Family Constraints:**
  - Offer family-friendly subscription plans or packages tailored to customers with children, reflecting their needs and constraints.

- **Cross-Channel Engagement:**
  - Promote digital engagement through enhanced online content and digital magazine subscriptions for customers who show higher online activity.

By focusing on these areas, the magazine company can better engage its customers, address subscription declines, and boost long-term customer loyalty and revenue.

Summary

In this study, multiple machine learning models, including **Logistic Regression** and **Support Vector Machines (SVM)**, were developed and compared using metrics such as *accuracy*, *precision*, *recall*, and *F1-score*. Hyperparameter tuning was applied to improve model performance. The **Hyperparameter Tuned Logistic Regression** emerged as the most suitable model, achieving an accuracy of 89.01%, with excellent recall and precision for identifying both subscribers and non-subscribers. This model was selected due to its balance of performance and interpretability, enabling actionable business insights. The findings highlight key factors influencing subscription trends, guiding targeted strategies to boost customer retention.

References

Boser, B. E., Guyon, I. M., & Vapnik, V. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, 144–152.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Cox, D. R. (1986). The logistic model. *Biometrika*, *73*(1), 197–202.

Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. *Cambridge University Press*.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. *Springer Series in Statistics*.

Schölkopf, B., Burges, C. J., & Vapnik, V. (2002). Advances in kernel methods: Support vector learning. *MIT Press*.

(Boser, Guyon, & Vapnik, 1992) (Cortes & Vapnik, 1995) (Cristianini & Shawe-Taylor, 2000) (Hastie, Tibshirani, & Friedman, 2009) (Cox, 1986) (Schölkopf, Burges, & Vapnik, 2002)