Logo

Description automatically generated

**College of Professional Studies**

**Northeastern University San Jose**

**MPS Analytics**

**Course: ALY6020 – Predictive Analytics**

**Module 3 Project: Understanding Magazine Subscription Behavior**

**Submitted on:**

Oct 13th, 2023

**Submitted to:**  **Submitted by:**

Professor: BEHZAD AHMADI BHAGYASHRI KADAM

**INTRODUCTION**

**Given Problem Statement:**

Last year, a magazine company noticed a drop in their subscriptions, which was puzzling since people were home more often and likely reading. To figure out what went wrong, we used their data in a four-step approach. First, we cleaned up the data to make sure it was good to use. Then, we built a model using logistic regression to predict who might subscribe and looked at which factors mattered most. We did the same with another method called SVM. Lastly, we compared the two models to see which one did better and gave our recommendation on which one to use.

**Understanding the Dataset**

The given dataset comprises a total of **2240 records and 29 columns/attributes** each representing a unique magazine subscriber and detailing their respective attributes such as Year of Birth, Education, Marital Status, and so forth. Among these attributes, there are demographic details like income, number of children at home, and recent purchase behaviors like the amount spent on wines or meats. The dataset also captures the response to various marketing campaigns (AcceptedCmp1, AcceptedCmp2, etc.) and the overall response (Response) which seems to be the target variable, indicating whether a subscriber responded positively to the magazine's offerings.

Below are the datatypes of the attributes:

A screenshot of a computer

Description automatically generated

**PART 1 : DATA CLEANING**

* **Checking the number of missing values for each Attribute in the dataset**

A screenshot of a computer

Description automatically generated

From the above output, we can observe that there are 24 missing values for Income column. Before , imputing the missing values, lets visualize the distribution of Income attribute :

A graph of a tall tower

Description automatically generated

From the above distribution , we can observe the income distribution is right-skewed , hence median would be a better choice to impute the missing values.

**Imputing the missing values with Income’s Median :**

A screenshot of a computer

Description automatically generated

After imputing the missing values with median , we can see that there are no missing values in the dataset.

**Feature Engineering :**

Further , lets convert the **Dt\_Customer** attribute to datetime format which further allows for easy date-related calculations & operations.

I have further created a new column - ‘**Days\_Since\_Customer’** which calculates the number of days since the earliest subscription date in the dataset for each customer. provides a continuous numerical representation of each customer

A computer code with text

Description automatically generated

Further , I have dropped the Dt\_Customer Column as it is not needed and checked the datatypes of all columns of our dataset:

A screenshot of a computer

Description automatically generated

**Checking for Duplicate Records :**

A screenshot of a computer

Description automatically generated

There are no duplicate records in our dataset.

**DESCRIPTIVE CHARACTERISTICS OF THE DATASET**

**A screenshot of a computer screen

Description automatically generated**

The above displays the statistical information for several numerical columns related to magazine subscribers' and below are some key insights:

1. Demographics:

* The average subscriber is born around the year 1968, suggesting that the majority of the subscribers are middle-aged.
* A significant portion of subscribers have children, with an average of 0.44 kids and 0.51 teenagers at home. This could indicate that the magazine might be popular among families.

2. Income:

- The average income of subscribers is approximately $52,238, but there's a wide range, with some subscribers earning as low as $1,730 and some earning as high as $666,666. This suggests that the magazine appeals to a diverse economic audience.

3. Spending Habits:

* Subscribers, on average, spend the most on wines ($303.94) and meat products ($166.95). This could indicate a preference or interest in these products among the subscribers.
* The spending on other products like fruits, fish, sweets, and gold products is relatively lower.

4. Engagement with Campaigns:

* The acceptance rates for campaigns vary. For instance, about 7.28% of subscribers accepted the third campaign (`AcceptedCmp3`), which might indicate the effectiveness of different campaigns.
* The overall response rate to the last campaign was 14.91%, suggesting that a good portion of the subscribers are engaged and responsive to the magazine's campaigns.

5. Customer Loyalty:

* The engineered feature `Days\_Since\_Customer` shows that, on average, subscribers have been customers for about 345 days. This suggests a relatively recent engagement for most subscribers with the magazine, but there's a wide range, with some being customers for as long as 699 days.

6. Feedback:

* A very small percentage of subscribers (0.94%) have lodged complaints, indicating general satisfaction among the majority of subscribers.

In summary, the magazine has a diverse subscriber base in terms of age and income. While they spend significantly on wines and meat products, there's an opportunity to engage them more with other product categories. The varying acceptance rates for different campaigns suggest that understanding the preferences of subscribers can lead to more effective campaigns in the future.

**PART 2 : LOGISTIC REGRESSION MODEL**

Before building the model , we need to prepare the data and perform one-hot encoding of the categorical variables like ‘**Education’ & ‘Martial\_Status’** so that they are converted to a format that can be used by our model.

Next , we split the data into **training & test sets (30% - test and 70% train)**

A screenshot of a computer program

Description automatically generated

The above output shows that X\_train has 1568 records and 36 features , X\_test has 672 records and 36 features , y\_train has 1568 records of target variable and y\_test has 672 records of target.

Next , lets initialize and train the model using the training data and predict on the testing data to predict the subscription behavior.

Lastly , the performance of this model is evaluated using metrics like accuracy , precision , recall etc. from the classification report generated.

**A screenshot of a computer

Description automatically generated**

From the above results, we understand the below:

* Our model correctly predicted the subscription behavior for approximately 87.2% of the customers on the test set.
* Also, the precision of our model for correctly predicting subscription (Class1) is 63% which means that the model predicts a customer will subscribe, its correct 63% of the time.
* Further, the recall for predicting a subscription is 0.23. This means the model identifies 23% of all actual subscribers.
* The F1-Score for predicting a subscription is 0.34, which is relatively low. This indicates that our model might be struggling to balance precision and recall for the positive class.
* The support is the number of actual occurrences of the class in the test set. There are 577 customers who didn't subscribe and 95 who did.
* **Summary**: The model performs well in predicting non-subscribers (class 0) with a high precision and recall. However, it struggles to predict subscribers (class 1) accurately, as indicated by the lower precision, recall, and F1-score for class 1.

The imbalance in the dataset (more non-subscribers than subscribers) might be affecting the model's performance.

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

To understand the significance of the variables & their business impact, I have analyzed the **coefficients values** of all the variables which represent the log odds for a unit change in the corresponding feature , holding all other features constant. Below are some key insights :

* **AcceptedCmp3, AcceptedCmp1, AcceptedCmp5**: These features have positive coefficients, indicating that customers who accepted these campaigns are more likely to subscribe. This suggests that these campaigns were effective in driving subscriptions.
* **NumWebVisitsMonth:** A positive coefficient suggests that as the number of web visits in a month increases, the likelihood of a subscription also increases. This could indicate that more engaged customers (those visiting the website frequently) are more likely to subscribe.
* **Marital\_Status\_Single:** Being single increases the likelihood of subscribing compared to the reference category (which is the category dropped during one-hot encoding). This could suggest that single individuals are more receptive to the magazine's content or offers.
* **Education\_PhD:** Customers with a PhD are more likely to subscribe compared to the reference education category. This might indicate that the magazine's content appeals to highly educated individuals.
* **Teenhome**: A negative coefficient suggests that having teenagers at home decreases the likelihood of subscribing. This could be due to different reading preferences or time constraints of households with teenagers.
* **NumStorePurchases**: A negative coefficient indicates that as the number of in-store purchases increases, the likelihood of a subscription decreases. This might suggest that customers who frequently shop in-store are less engaged with the magazine or its online content.

**PART 3 : Support Vector Machine (SVM) Model**

The below code for SVM model initializes, trains and evaluates an SVM model with a linear kernel. After training, it predicts the subscription behavior on the test data and finally evaluates the model using accuracy & other classification metrics.

**A screenshot of a computer

Description automatically generated**

From the above results, we understand that the below:

* Our model has an accuracy of approx. 86% which means it has correctly predicted the subscription behavior for approximately 86% of the customers on the test set.
* Also, the precision of our model for correctly predicting subscription (Class1) is 50% which means only 50% of the predicted subscribers were actually subscribers.
* Further, the recall for predicting a subscription is 0.23. This means the model identifies 17% of all actual subscribers.
* The F1-Score for predicting a subscription is 0.25, which is relatively low. This indicates that our model might be struggling to balance precision and recall for the positive class.
* The support is the number of actual occurrences of the class in the test set. There are 577 customers who didn't subscribe and 95 who did.
* **Summary**: The SVM model performs well in identifying non-subscribers (class 0) with high precision, recall, and F1-score.

However, the model struggles to identify subscribers (class 1) accurately, as evidenced by the low recall and F1-score for this class.

The imbalance in the dataset (more non-subscribers than subscribers) might be affecting the model's performance, especially for the minority class (subscribers).

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer code

Description automatically generated**

Below are some key insights from above coefficients values:

* **AcceptedCmp5, AcceptedCmp1, and NumWebPurchases** have the highest positive coefficients. This suggests that these features have a strong positive influence on the target variable, meaning that an increase in their values is likely to increase the probability of subscriptions.
* On the other hand, **NumStorePurchases, Teenhome, and Recency** have the most negative coefficients. This indicates that an increase in their values is likely to decrease the probability of a positive response/subscriptions.

**PART 4: COMPARISON**

Lets compare the results , observations & metrics of both the Logistic Regression & SVM Models and summarize the conclusion :

**1. Overall Accuracy:**

* Logistic Regression: 87.2%
* SVM: 85.9%

**2. Precision (for class 1: Subscribed):**

* Logistic Regression: 0.63
* SVM: 0.50

**3. Recall (for class 1: Subscribed):**

* Logistic Regression: 0.23
* SVM: 0.17

**4. Variables Deemed Significant:**

* Both the models provided coefficients for each feature and features with higher absolute coefficients and low p-values (typically < 0.05) are considered significant.
* For Logistic model – significant variables were AcceptedCmp variables , NumWebVisitsMonth and Marital\_Status\_Single
* For SVM Model , significant variables were AcceptedCmp5, AcceptedCmp1, and NumWebPurchases

**Summary:**

- Accuracy: Both models have comparable accuracy, with the Logistic Regression model slightly outperforming the SVM.

- Precision: The Logistic Regression model has a higher precision for predicting subscriptions, meaning it has a lower false positive rate. This suggests that when the Logistic Regression model predicts a subscription, it's more likely to be correct compared to the SVM.

- Recall: Both models have relatively low recall for predicting subscriptions, but the Logistic Regression model is slightly better. This means both models miss a significant number of actual subscribers, but the Logistic Regression model misses slightly fewer.

- Interpretability: One advantage of the Logistic Regression model is its interpretability. The coefficients provide direct insights into the relationship between each feature and the target variable, and p-values can be used to determine the significance of each feature. SVM, on the other hand, is more of a black-box model, and while it provides coefficients, interpreting them in the context of the problem can be more challenging.

- **Recommendation:** Based on the three metrics (accuracy, precision, and recall), the logistic regression model performs slightly better than the SVM model. While the difference in accuracy is not substantial, the higher precision and recall values for the logistic regression model make it more reliable, especially if we are particularly interested in predicting class 1.

Furthermore, logistic regression provides probabilities for its predictions, which can be useful for understanding the confidence level of predictions or for setting different thresholds for classification. SVM, on the other hand, is more of a black-box model, and while it can be powerful, its results can sometimes be harder to interpret.

Considering the above points and the metrics, I would recommend the Logistic Regression model for this dataset. However, it's essential to note that model selection should also consider other factors like business context, interpretability, and potential model deployment scenarios.

**References**

1. https://www.analyticsvidhya.com/blog/2021/10/building-an-end-to-end-logistic-regression-model/
2. https://medium.com/javarevisited/evaluating-the-logistic-regression-ae2decf42d61
3. <https://northeastern.instructure.com/courses/160443/pages/lesson-3-4-implementing-logistics-regression?module_item_id=9500055>

**Appendix**

|  |
| --- |
| import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  import seaborn as sns  #Loading data#  df = pd.read\_excel('/Users/bhagyashrikadam/Documents/NEU\_ASSIGNMENTS/ALY6020/Module3/marketing\_campaign.xlsx')  df.shape  print(df.dtypes)  df.head(10)  **## PART1 ##**  **## Data Cleaning ##**  **# Checking the number of missing values ##**  **missing\_values = df.isnull().sum()**  **print(missing\_values)** |
| # Visualizing the distribution of the 'Income' column  plt.figure(figsize=(10, 6))  sns.histplot(df['Income'], kde=True, bins=30)  plt.title('Distribution of Income')  plt.xlabel('Income')  plt.ylabel('Frequency')  plt.show()  # Imputing missing values in 'Income' with its median  median\_income = df['Income'].median()  df['Income'].fillna(median\_income, inplace=True)  # Check again for missing values  missing\_values\_updated = df.isnull().sum()  print(missing\_values\_updated)  # Convert 'Dt\_Customer' column to datetime format  df['Dt\_Customer'] = pd.to\_datetime(df['Dt\_Customer'])  # Calculate the number of days since the earliest date in the dataset  earliest\_date = df['Dt\_Customer'].min()  df['Days\_Since\_Customer'] = (df['Dt\_Customer'] - earliest\_date).dt.days  # Drop the original 'Dt\_Customer' column  df = df.drop(['Dt\_Customer'], axis=1)  # Checking data types of each column  data\_types = df.dtypes  data\_types  # Checking for duplicate rows  duplicate\_rows = df.duplicated().sum()  duplicate\_rows  # Descriptive Statistics #  df.describe()  #PART2 : Building the Logistic regression Model #  from sklearn.model\_selection import train\_test\_split  from sklearn.preprocessing import OneHotEncoder  # 1. Data Preparation  # One-hot encoding for categorical variables  df\_encoded = pd.get\_dummies(df, columns=['Education', 'Marital\_Status'], drop\_first=True)  # Splitting data into training and testing sets ('Response' is the target variable)  X = df\_encoded.drop(['ID', 'Response'], axis=1)  y = df\_encoded['Response']  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)  # Display the shape of training and testing sets  X\_train.shape, X\_test.shape, y\_train.shape, y\_test.shape  from sklearn.linear\_model import LogisticRegression  from sklearn.metrics import accuracy\_score, classification\_report  # 2. Model Building  # Initialize the logistic regression model  logreg = LogisticRegression(solver="liblinear", max\_iter=1000, random\_state=42)  # Fit the model to the training data  logreg.fit(X\_train, y\_train)  # Predict on the test set  y\_pred = logreg.predict(X\_test)  # 3. Model Evaluation  # Calculate accuracy  accuracy = accuracy\_score(y\_test, y\_pred)  classification\_rep = classification\_report(y\_test, y\_pred)  print("Accuracy:", accuracy)  print("\nClassification Report:\n", classification\_rep)  # 4. Analyze Coefficients  # Display coefficients  coefficients = pd.DataFrame({  'Feature': X.columns,  'Coefficient': logreg.coef\_[0]  }).sort\_values(by='Coefficient', ascending=False)  print("\nCoefficients:\n", coefficients)  # Part3 : SVM Model #  from sklearn.svm import SVC  from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix  # Initialize the SVM model  svm\_model = SVC(kernel='linear', random\_state=42)  # Fit the model to the training data  svm\_model.fit(X\_train, y\_train)  # Predict on the test set  y\_pred\_svm = svm\_model.predict(X\_test)  # Evaluate the model's performance  accuracy\_svm = accuracy\_score(y\_test, y\_pred\_svm)  classification\_rep\_svm = classification\_report(y\_test, y\_pred\_svm)  print("Accuracy:", accuracy\_svm)  print("\nClassification Report:\n", classification\_rep\_svm)  # Extracting the coefficients  coefficients\_svm = svm\_model.coef\_[0]  # Creating a DataFrame to display the coefficients alongside feature names  svm\_coefficients\_df = pd.DataFrame({  'Feature': X.columns,  'Coefficient': coefficients\_svm  }).sort\_values(by='Coefficient', ascending=False)  print(svm\_coefficients\_df) |