Understanding Magazine Subscription Behaviour

Rajashekar Korutla

Applied Machine Intelligence, Roux Institute at Northeastern University

ALY 6020: Predictive Analytics

CONTENTS

1 Problem Statement/Business Requirements

2 Overview of data

3. Data pre-processing including data exploration analysis

4. Details of your modeling strategy

5. Estimation of model’s performance

6. Insights and conclusions

Problem Statement/Business Requirements

A magazine company is trying to understand last year's decline in subscriptions. With people spending more time at home, they thought people would be spending more time reading. Using this dataset, help the company understand what is or isn't working.

Overview Of Data

From the marketing dataset, it can clearly be seen that there are 2240 rows and 29 columns in the dataset.

So, in order to get the list of columns, I used the df.columns line of code and stored the list of columns in the variable named col_list. Here is the line of code to get the columns from the dataset:

```
col_list = list(df.columns)
```

After getting the list of columns, I have used the following command to get the datatypes of the columns that are present in the dataset.

```
df.info()
```

Data Pre-Processing Including Data Exploration Analysis

Then it happened to realize that there are certain columns with the dataset as object, which are supposed to be changed to numeric for the model to consider them for further process.

For which, I have used the get_dummies function from the pandas library to change their datatype from object to integer.

The get_dummies function would create a dummy column for every category of data present in that specific column, thereby having only Boolean values in the columns.

Here is the line of code, that represents the function:

```
df = pd.get_dummies(df , columns = ['Education','Marital_Status'],drop_first = True)
```

when the drop_first is set to True, all the first previous columns are being dropped , thereby giving limited number of columns for the model processing.

Then I found that there exists certain null values in the column of ‘Income’ , these null values have been dropped by the following line of code:

```
df = df.dropna()
```

After the above processing is done,this is the information regarding the dataset:

```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ID                      2240 non-null   int64
 1   Year_Birth              2240 non-null   int64
 2   Income                  2216 non-null   float64
 3   Kidhome                 2240 non-null   int64
 4   Teenhome                2240 non-null   int64
 5   Dt_Customer             2240 non-null   object
 6   Recency                 2240 non-null   int64
 7   MntWines                2240 non-null   int64
 8   MntFruits               2240 non-null   int64
 9   MntMeatProducts         2240 non-null   int64
 10  MntFishProducts         2240 non-null   int64
 11  MntSweetProducts        2240 non-null   int64
 12  MntGoldProds            2240 non-null   int64
 13  NumDealsPurchases       2240 non-null   int64
 14  NumWebPurchases         2240 non-null   int64
 15  NumCatalogPurchases     2240 non-null   int64
 16  NumStorePurchases       2240 non-null   int64
 17  NumWebVisitsMonth       2240 non-null   int64
 18  AcceptedCmp3            2240 non-null   int64
 19  AcceptedCmp4            2240 non-null   int64
 20  AcceptedCmp5            2240 non-null   int64
 21  AcceptedCmp1            2240 non-null   int64
 22  AcceptedCmp2            2240 non-null   int64
 23  Complain                2240 non-null   int64
 24  Z_CostContact           2240 non-null   int64
 25  Z_Revenue               2240 non-null   int64
 26  Response                2240 non-null   int64
 27  Education_Basic         2240 non-null   uint8
 28  Education_Graduation    2240 non-null   uint8
 29  Education_Master        2240 non-null   uint8
 30  Education_PhD           2240 non-null   uint8
 31  Marital_Status_Alone    2240 non-null   uint8
 32  Marital_Status_Divorced 2240 non-null   uint8
 33  Marital_Status_Married  2240 non-null   uint8
 34  Marital_Status_Single   2240 non-null   uint8
 35  Marital_Status_Together 2240 non-null   uint8
 36  Marital_Status_Widow    2240 non-null   uint8
 37  Marital_Status_YOLO     2240 non-null   uint8
dtypes: float64(1), int64(25), object(1), uint8(11)
```

The date is also later separated into few other columns named day, month and year, so that it automatically changes the type to the integer.

It can be done by the following line of code:

```
df['Dt_Customer'] = pd.to_datetime(df['Dt_Customer'])
df['Year_Customer'] = df['Dt_Customer'].apply(lambda x: x.year)
df['Month_Customer'] = df['Dt_Customer'].apply(lambda x: x.month)
df['Day_Customer'] = df['Dt_Customer'].apply(lambda x: x.day)
df = df.drop('Dt_Customer', axis=1)
```

Details Of Modeling Strategy

The target variable ‘Response’ has been separated as ‘y’. All the other features have been put together into 'x’, to form the feature matrix. The data has been split for training data and test data, as 70% for training and the remaining 30% for test data.

This is implemented by the following lines of code:

```
x = df.drop(labels = ['Response'],axis = 1)
y = df['Response']
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3 , shuffle = 30)
```

Implementing Logistic Regression

After the data has been split into training data and test data, I have imported the logistic regression algorithm from the scikit learn library, with the following line of code:

```
from sklearn.linear_model import LogisticRegression
```

then the training data has been fit in the model, and predicted the test data with the following lines of code:

```
lr = LogisticRegression()
logreg = lr.fit(x_train,y_train)
y_pred = logreg.predict(x_test)
```

Implementing Support Vector Machine Algorithm

From scikit learn, support vector machine for classification can be imported with the following line of code:

```
from sklearn import svm
from sklearn.svm import LinearSVC
```

After fitting the model with the training data, I used this line of code to know the accuracy of how the model has been fit on the training data :

```
score = lsvc.score(x_train, y_train)
```

then I found it to be – `accuracy_Score: 0.8478401031592521`

Estimation Of Model's Performance

The metrics used for evaluation are classification report, confusion matrix. The classification report shows a representation of the main classification metrics such as precision, recall, f1-score. This gives a deeper intuition of the classifier behavior over global accuracy. The metrics are defined in terms of true and false positives, and true and false negatives. a true positive is when the actual class is positive as is the estimated class. A false positive is when the actual class is negative, but the estimated class is positive.

Confusion matrix is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values.

Classification report for logistic regression algorithm is the following:

	precision	recall	f1-score	support
0	0.99	0.87	0.92	645
1	0.13	0.65	0.22	20
accuracy			0.86	665
macro avg	0.56	0.76	0.57	665
weighted avg	0.96	0.86	0.90	665

Similarly, classification report for the support vector machine algorithm is:

	precision	recall	f1-score	support
0	0.85	1.00	0.92	568
1	0.00	0.00	0.00	97
accuracy			0.85	665
macro avg	0.43	0.50	0.46	665
weighted avg	0.73	0.85	0.79	665

Insights And Conclusions

When the same dataset is tested with both different algorithms of logistic regression and support vector machines, I found out that every model's performance is different on the dataset.

The accuracy is 1% lesser when tried support vector machines, leaving the accuracy to be 85% with that algorithm of support vector machines.

The accuracy is found to be 86% when the dataset is tested with logistic regression algorithm.

I found both the models are performing good on the dataset , without any situation of underfitting or overfitting.