

Market Campaign Prediction

Name:	Aman Dhillon
Registration No./Roll No.:	20032
Institute/University Name:	IISER Bhopal
Program/Stream:	Economic Sciences
Problem Release date:	Aug. 17, 2023
Date of Submission:	Nov. 19, 2023

1 Introduction

The research aim at prediction of success of the market campaign based on customer's response for a company. This is the marketing data of a company with data on customer profiles, product preferences, channel performance etc. The class labels are marked as '0' for 'failure' and '1' for 'success' of the campaign. Using the features shown in Figure 1, we will be estimating the failure or success of the campaign based on the given dataset. The dataset provided has class imbalance problem i.e. success has 301 data points out of given 2016 data points. Last variable is the class labels for the datapoints.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2016 entries, 0 to 2015
Data columns (total 26 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Year_Birth            2016 non-null  int64
1   Education             2016 non-null  object
2   Marital_Status       2016 non-null  object
3   Income               1995 non-null  object
4   Kidhome              2016 non-null  int64
5   Teenhome             2016 non-null  int64
6   Dt_Customer          2016 non-null  object
7   Recency              2016 non-null  int64
8   MntGoldProds         2016 non-null  int64
9   MntWines             2016 non-null  int64
10  MntFruits            2016 non-null  int64
11  MntFishProducts      2016 non-null  int64
12  MntSweetProducts     2016 non-null  int64
13  MntMeatProducts      2016 non-null  int64
14  NumDealsPurchases    2016 non-null  int64
15  NumWebPurchases      2016 non-null  int64
16  AcceptedCmp3         2016 non-null  int64
17  NumCatalogPurchases  2016 non-null  int64
18  NumStorePurchases    2016 non-null  int64
19  NumWebVisitsMonth    2016 non-null  int64
20  AcceptedCmp2         2016 non-null  int64
21  AcceptedCmp4         2016 non-null  int64
22  AcceptedCmp5         2016 non-null  int64
23  AcceptedCmp1         2016 non-null  int64
24  Complain             2016 non-null  int64
25  target               2016 non-null  object
dtypes: int64(21), object(5)
memory usage: 409.6+ KB
```

Figure 1: Overview of features

2 Data Processing & Feature Selection

Data processing include conversion of Income variable to integer form, removing data, adjusting the missing values of Income using the group average of the Education feature, one hot encoding for the Education and Marital Status variable. Feature Selections using Cramer V metrix for categorical variable and Krushal Wallis H test for Numerical variable.

3 Methods

3.1 Train-Test Split

The dataset is randomly split into training and testing sets by a factor of 0.3, i.e 70% data (1411 rows) is selected for training, and 30% data (605 rows) is selected for testing the model.

3.2 Models Used For Classification

Given problem is a Classification problem as there are two classes (0 and 1). Hence, the main classification algorithms used for this problem are:

- K-Nearest Neighbors
- Decision Tree Classifier
- Gaussian Naive Bayes
- Logistic Regression
- Random Forest Classifier
- Support Vector Classifier

The complete project can be found [here](#).

3.3 Hyperparameter Tuning

Specific parameters known as hyperparameters can be utilized to adjust how a machine learning algorithm behaves. These are given to the model and initialized prior to training. A hyperparameter is a parameter that controls how the learning process is carried out. Hyperparameters are fine-tuned by choosing the optimal values in order to increase performance and improve the assessment metric. One of the simplest methods for hyper-parameter tweaking is grid search. Its implementation is therefore quite simple. All possible permutations of a model's hyperparameters are utilized to adjust it, and the best-performing variations are selected.

Table 1: Hyper-parameters of different models

Models	Hyper-parameters Space	Best Hyper-parameter features
K-NN	<ul style="list-style-type: none">• n-neighbours: [5,7,9,11,...,38,39,40]• weights : ['uniform', 'distance'],• 'metric' : ['minkowski', 'euclidean', 'manhattan']	<ul style="list-style-type: none">• n-neighbours: [5]• weights : ['distance'],• 'metric' : ['minkowski']
Decision tree	<ul style="list-style-type: none">• criterion: ['gini', 'entropy']• max features: ['auto', 'sqrt', 'log2', None]• max depth: [15, 30, 45, 60]• ccp alpha: [0.009, 0.005, 0.05]	<ul style="list-style-type: none">• criterion: ['entropy']• max features: [sqrt]• max depth: [45]• ccp alpha: [0.009]
Random forest	<ul style="list-style-type: none">• criterion: ['gini', 'entropy']• n estimators: [int(x) for x in np.linspace(start = 200, stop = 300, num = 100)]• max depth: [10, 20, 30, 50, 100, 200]	<ul style="list-style-type: none">• criterion: ['entropy']• n estimators: [235]• max depth: [10]
Gaussian Naive bayes	<ul style="list-style-type: none">•	<ul style="list-style-type: none">• var smoothing : [0.0]
Logistic Regression	<ul style="list-style-type: none">• C: np.linspace(start = 0.1, stop = 10, num = 100)• penalty: ['l1', 'l2', 'elasticnet']• solver: ['newton-cg', 'lbfgs', 'liblinear']	<ul style="list-style-type: none">• C: [9.9]• solver: ['liblinear']
SVM	<ul style="list-style-type: none">• C : np.logspace(-2, 7, num=25, base=2)• gamma: [1, 0.1, 0.01, 0.001]• kernel: ('linear', 'rbf', 'polynomial', 'sigmoid')	<ul style="list-style-type: none">• C : [1.189207115002721]• gamma: [1]• kernel: ['linear']

4 Evaluation Criteria

Following performance metrics used are accuracy, macro-averaged precision, recall, and f-measure since this is a classification problem. These measures are described as:

- Precision = $\frac{\text{No of correctly predicted positive points}}{\text{total predicted positive points}} = \frac{TP}{TP + FP}$
- Recall = $\frac{\text{No of correctly predicted positive points}}{\text{total actual positive points}} = \frac{TP}{TP + FN}$
- Accuracy = $\frac{\text{No of correctly predicted data points}}{\text{total number of data points}} = \frac{TP + TN}{TP + FP + FN + TN}$
- f1_score = $\frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$

For two classes for example, Positive and Negative then the following are defined :

- True Positives(TP): It is the case where we predicted Positive and the real output was also Positive.
- True Negatives(TN): It is the case where we predicted Negative and the real output was also Negative.
- False Positives(FP): It is the case where we predicted Positive but it was actually Negative.
- False Negatives(FN): It is the case where we predicted Negative but it was actually Positive

5 Analysis of Results

Table 2 shows the recall, precision, accuracy and f measure for all the classification models used in this project

Table 2: Performance Of Different Classifiers Using All Terms

Classifier	Precision	Recall	Accuracy	F-measure
K-NN	0.665	0.584	0.834	0.601
Decision tree	0.639	0.691	0.776	0.654
Random forest	0.851	0.715	0.894	0.759
Gaussian Naive bayes	0.636	0.675	0.783	0.649
Logistic Regression	0.691	0.783	0.804	0.714
SVM	0.842	0.659	0.883	0.738

6 Discussions and Conclusion

Random forest provides the best f-measure as well as accuracy. Hence, I'll use it.

6.1 Result on Test Data Sample

On using the Random forest predicted class labels for the test sample.

