Yash Jain MDS202048
Yash Raj MDS202049

DMML Assignment 1

June 2021

OVERVIEW

We have built three classifiers for the **Bank Marketing Data Set**: a **Decision Tree** (DT), a **Random Forest Classifier** (RF) and a **Naive Bayes Classifier** (NB)

METRICS USED

Since this is a **highly imbalanced** data with only 11.2% of the data accounting for the positive instances i.e. people who subscribe to a term deposit, accuracy won't be the best metric to evaluate the performance of a classifier as in most cases (in this dataset) it will perfectly identify the negatives which are in abundance resulting in unreasonably high score. We are more interested in the positive instances hence Precision, Recall and F1 Score make more sense. Metrics used are:

1. Accuracy
2. Precision
3. Recall
4. F1 Score

FEATURE SELECTION

We removed the "**contact**" column since everyone had either a telephone or cellular and it wouldn't affect the result in any way. Similarly "**month**" and "**day_of_week**" were also removed because of less significance on the outcome. It may have affected people with seasonal jobs but since the jobs listed aren't seasonal, we ignored it.

Next, we checked the correlation of every feature with the rest of the features to identify features that are highly correlated. We found that **'emp.var.rate**,' '**cons.price.idx**,' '**euribor3m**', and '**nr.employed**' were **highly correlated** with each other showing correlation >0.7 which makes sense because these indicators are related to each other. We decomposed these to form a new feature.

We assigned all the categorical features, numerical values. We assigned categories on **'age**,' **'pdays'** and **'duration'** based on how much each of the categories contributed to the positive class. We also dropped the unknowns from the table.

After categorising we checked the correlation again and found that **'pdays', 'previous**' and **'poutcome'** were highly correlated so we decomposed them to form a new feature.

To deal with the class imbalance, we tried both **downsampling** and **upsampling** to make it more balanced. We ran the classification on the upsampled dataset with approx 54k data points.

CLASSIFIERS:

Decision Tree Classifier

```
DT = DecisionTreeClassifier(criterion="entropy",max_depth=5,max_leaf_nodes=20)
```

We tried the DT classifier for different depths but to avoid overfitting we went with 'max_depth' of 5 and restricted the number of leaf nodes to 20. It took around 0.14 seconds to fit the data.

Random Forest Classifier

```
RF = RandomForestClassifier(n_estimators=100,n_jobs=-1, max_depth=8)
```

We set the number of trees for the RF Classifier to 100 and max depth to 8. The time taken is somewhere between 1.5-2.5 seconds which is definitely higher than Decision Tree and it is computationally more expensive than DT Classifier.

Naive Bayes Classifier

```
NB = GaussianNB(var_smoothing=6.579332246575682e-07)
```

Using GridSearchCV, we found the best var_smoothing parameter value for our Naive Bayes Classifier. The search took about a minute and the classifier took around 0.04 seconds which is the fastest among the three classifiers.

COMPARISON:

Our main objective was to get high recall for the classifiers because we are interested in the positive instances i.e. people who subscribe to term deposit. We can afford to have high recall in lieu of high precision because high recall means less chances of False Negatives. We wouldn't want to classify a potential customer in the negative class since it's good for the bank that more people subscribe to a term deposit.

Based on the metrics. Both the Decision Tree and Random forest gave similar accuracy whereas the accuracy for Naive Bayes was quite low. On increasing the depth of the decision tree to 8, all 4 metrics seemed to converge at 0.85. But with the risk of overfitting we went with the smaller value 5 for DT.

	Accuracy	Precision	Recall	F1-Score
Decision Tree	**0.8527**	**0.8207**	**0.8992**	**0.8581**
Random Forest	**0.8662**	**0.8410**	**0.9001**	**0.8696**
Naive Bayes	**0.7994**	**0.7513**	**0.8894**	**0.8147**

Since Random Forest is less prone to overfitting, we chose the depth to be 8. Random Forest gave balanced results in all the metrics. On increasing the depth to an unusually high number, all the 4 metrics gave 0.90+ scores which may not be good when it comes to generalising the model. We chose the depth that seemed the best to us considering we had only 12 features. It outperformed Decision Tree with depth 5 and Naive Bayes in all four metrics.

Naive Bayes gave the lowest recall among the three classifiers and at the same time had the lowest F1 Score due to low precision. It gave quite low accuracy and precision as compared to DT and RF. Among all the three classifiers, Random Forest gave the most balanced performance and according to us it is a good choice for the problem on this dataset.

Note: The link to the features and parameter selection with outputs and visualization part is given below: https://github.com/yjwam/DMML-Assignment-1