Data Mining and Machine Learning

Assignment 1

Roudranil Das
MDS202227
roudranil@cmi.ac.in

Shreyan Chakraborty
MDS202237
shreyanc@cmi.ac.in

Contents

Task 1

1.1 Data preprocessing

Our data preprocessing was carried out in the following 3 steps:

1. Analysis of the variables.
2. Feature selection.
   a) Removal of highly correlated features.
   b) Removal of categorical features which are not contributing to the predictive accuracy.
3. Encoding of categorical variables.

A preliminary analysis of the variables showed that some 6 categorical features had missing values in them, encoded as `unknown`. We also examined the distribution of the numeric variables and the number of occurances of each unique category in all the cateogical columns.

1.1.1 Categorical Feature Selection

- The `contact` attribute noted the medium of communication with the client - either mobile or telephone. Method of contact does not make any difference in whether the client subscribes to the term deposit or not.
- Out of all columns with missing (unknown) values, the column `default` had the highest percentage of missing values, roughly 20%. Moreover if we consider the fraction of people who subscribed to yes for each of the unique categories in the column `default`, we see that there is no significant difference in the fractions. Hence we decide to drop this attribute.
- The attribute `day_of_week` was also dropped because the day on which the person was contacted was unlikely to have a big effect on the person subscribing to a term deposit. However note that the attribute `month` may be important in this sense, which is why we kept `month`.

1.1.2 Numerical Feature Selection

- An examination of the `pdays` column yielded that 39673 records out of 41188 had the value `999`, meaning that the corresponding client was not contacted in a previous campaign. All other values were between 0 and 27 days, but with extremely rare occurances. Thus this column turned out to have very little predictive effect and hence we decided to drop this column.
- The `previous` attribute noted the number of times a client was contacted previously in a past campaign. We found that 35563 out of 41188 clients have not been contacted previously. Following a similar argument as the one for `pdays` we decide to drop this column as well.
- We checked for correlation among the 5 consumer and market indices: `emp.var.rate`, `cons.price.idx`, `cons.conf.idx`, `euribor3m`, `nr.employed`. We observed the following:
  - `euribor3m` is highly correlated with `nr.employed` and `emp.var.rate`
  - `emp.var.rate` is highly correlated with `cons.price.idx`

  In order to reduce autocorrelation we decide to drop the `emp.var.rate` and `nr.employed` columns.

1.1.3 Categorical Feature Encoding

- The target variable `y` was label encoded with 0 and 1 corresponding to No and Yes.
- `job`, `marital`, `education` are target encoded, that is encoded with the mean of the targets of the rows with the corresponding categories. This was because we observed that falling in either of these categories had a big effect on whether a person subscribed to a term deposit or not. We have experimented with both target and one-hot encoding, and target encoding produced fractionally better results. We decided to stick with it as it is intuitively easier to explain than one hot encoding.
- We one-hot encoded `month` as the months should not have any inherent ordering among themselves. Similarly we have one-hot encoded `housing` and `loan`.

1.2 Results with Decision Trees

Baseline results with decision trees, validated with 5-Fold stratified cross validation over the entire dataset are the following:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.89
Average	0.50	0.52	0.51

We used `OPTUNA` for hyperparameter optimisation to get the following best parameters given below. The hyperparameter optimisation was done in conjunction with the same 5-fold stratified cross validation.

`max_depth`	5
`min_samples_split`	4
`min_samples_leaf`	2

Using these parameters our results improved to:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.91
Average	0.64	0.50	0.56

Average time taken to fit the model: 0.214 seconds.
Space taken to train the model: 281.85 MiB.
Average time taken to tune hyperparameters: 17.34 seconds.

1.3 Results with Naive Bayes

Since we had continuous values in all our columns, we decided to choose the `GaussianNB` model. Using it our results (validated by the same cross validation framework) are:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.88
Average	0.45	0.43	0.44

Average time taken to fit the model: 0.029 seconds.
Space taken to train the model: 281.14 MiB.

Task 2

2.1 Data preprocessing

Our data preprocessing had 4 main problems to resolve:

1. Presence of columns with names

2. Encoding of categorical variables
3. Converting continuous variable to categorical
4. Scaling numerical variables

We defined the target variable as `Hit` if the revenue is more than the budget and `Flop` otherwise. We have retained the

2.1.1 Dealing with columns with names

The dataset contained 4 columns with names: `Movie Name`, `Lead Star`, `Director` and `Music Director`. It is reasonable to expect that success or failure of a movie depends on the presence of certain prominent names, either as lead stars or directors or as music directors. However that assumption is based on the prior information that these certain individuals have delivered certain high grossing and popular movies in the past. This information can certainly be expressed in terms of parameters such as number of hits delivered by them in the past 2-3 years, gross revenue generated by their past movies of the same genre, or even gross audience rating (like those by Rotten Tomatoes or IMDB) of their movies in the recent past. The presence of such parameters would help to quantify the impact such names would have on the success of their films. In our dataset the absence of these variables means that these attributes with names are nothing but categorical columns with way too many unique categories.

Dealing with such categorical data is problematic in 2 ways:

- Encoding is extremely difficult.
  - If we one hot encode all the names, then we would end up with a highly sparse matrix of roughly 4000 columns, as some columns such as that of movie name or music director name have unique values in almost all rows.
  - In the absence of any further information, we cannot assume any inherent ordering within the names, and hence ordinal encoding them is out of the question.
- Even if we managed to devise a suitable encoding, any mapping would be difficult to extend to any new data point, which may have a director, lead star and a movie name we have never seen before.

For these reasons we have decided to completely do away with these 4 columns, as even though they have some useful information, we don't have any means currently to exploit that.

2.1.2 Categorical Feature Encoding

The features `whether remake`, `whether franchise`, `new actor`, `new director` and `new music director` had only two categories and hence we encoded them as 1 and 0.

2.1.3 Continuous to Categorical Encoding

The feature `number of screens` has extremely scattered value points.Moreover it has a highly positively skewed distribution with most values being 1.So we divided the entire distribution into three sub-divisions.

- All values less than or equal to 0.25 quantile of the distribution,categorized as "Low".
- All values greater than 0.25 quantile and less than or equal to 0.75 quantile of the distribution,categorized as "Moderate".
- All values greater than 0.75 quantile of the distribution,categorized as "High".

2.1.4 Scaling of Numerical Variable

The feature `budget(inr)` has extremely scattered value points.So it was scaled down using `MinMaxScaler` function.

2.2 Results with Decision Trees

Baseline results after spliting our dataset into train data consisting of 80% of the data and test data consisting of 20% of the data are as follows:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.90
Macro Average	0.86	0.87	0.87
Weighted Average	0.90	0.90	0.90

Next we perform `GridSearchCV` for hyper-paramter optimization and got better improved results.

`max_depth`	4
`min_samples_split`	2
`min_samples_leaf`	2

Using these parameters we got the better improved result as follows:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.91
Macro Average	0.87	0.89	0.88
Weighted Average	0.91	0.91	0.91

For preventing over generalization of our model and meeting an optimized acceptance between train and test dataset we carried out CCP-Cost Complexity Pruning. We got value of $\alpha = 0.0105$. Corresponding to value of alpha we got the following result as follows:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.90
Macro Average	0.86	0.89	0.88
Weighted Average	0.91	0.90	0.90

Average time taken to fit the model: 0.014 seconds.
Space taken to train the model: 274.38 MiB.
Average time taken to tune hyperparameters: 2.471 seconds.
Total time taken for cost complexity pruning: 0.102 seconds.

2.3 Results with Naive Bayes

Since most of the encoding done was one hot encoding and pre-processed dataset contained mostly binary data, we decided to choose the code `MultinomialNB` model.

We used 3 fold stratified cross validation over the entire dataset and got the result as follows:

Metric	Precision	Recall	F1-score
Accuracy	N.A	N.A	0.80
Average	0.68	0.58	0.62

Average time taken to fit the model: 0.0019 seconds.
Space taken to train the model: 277.20 MiB.

Conclusion

Comparing the performance of both the models we observe that the performance of the models depends largely on the feature engineering and hyperparameter tuning.

In both cases we had data with highly imabalanced target classes. As a result we have that decision trees performed better than Naive Bayes models.