Content

Problem Statement

Highlights (novel algorithms, insights from the project)

Dataset Pre-processing (including cleaning, data exploration, feature engineering etc)

Training and Testing Procedure

1. Algorithm
    a. Decision Tree
    b. Naive Bayes
    c. SVM
    d. Neural Net
    e. Random Forest

Experimental Study and Analysis

1. Results of each Algorithm
    a. Decision Tree
    b. Naive Bayes
    c. SVM
    d. Neural Net
    e. Random Forest

Summary of Project Achievements

Future Directions for further Improvements

Member work distribution:

Sandareka Wickramanayake(Neural Net)
Han Xi(Decision Tree)
Nguyen Thanh Toan(Random Forest, cforest, SVM)
Gao Xiang(Random Forest, cforest, Naïve Bayes)

1. Problem Statement

Marketing selling campaigns constitute a typical strategy using artificial intelligence techniques to support decisions and enhance business. Companies use direct marketing to contact with some specific customers to make their product more targeting. By communicating with customers through various channels, the company can get various information from them. Technology enables rethinking marketing by focusing on maximizing customer lifetime value through the evaluation of available information and customer metrics, thus allowing us to build longer and tighter relations in alignment with business demand. In addition, the task of selecting the best set of clients for a banking institution who are more likely to subscribe a product is considered significantly meaningful and important.

The problem to address in this project is to predict the success of direct marketing campaigns of a Portuguese banking institution for selling bank long-term deposits. The dataset includes 20 attributes describing bank client, product and social and economic attributes. Our desired target or output variable is the fact if the customer subscribe for the deposit or not. The given variables can be categorized into four categories as follows:

- Information about the client: id, age, job, marital, education, default, housing, loan,
- Information related to last contact of the current campaign : contact, month, day_of_week, duration.
- Social and economic context attributes: emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed
- Other attributes: campaign, pdays, previous, poutcome

The dataset is coming with many challenges. One is the dataset is unbalanced, only 11% is related to positive class. Further, the dataset contains “unknown” values.

In this project, we first explore and preprocess the dataset such as getting statistical parameters for each attribute, making plots and handling missing data. Then we use one part of the giving training data to train and another part to test the models. Moreover, we analyze the data with 7 predictive models, namely Decision Tree(DT), Random Forest(RF), Conditional Random Forests (cforest), Naive Bayes, Neural Network(NN), support vector machine(SVM) and a Deep learning approach. Then we compare the results of prediction of different models and optimize them by improving some parameters. Finally, those models are evaluated by Matthews correlation coefficient (MCC) and model fine tuning is performed.

2. Highlights

In this project, we have tried a bunch of classification approaches, namely decision tree, Bayesian classifier, random forest, conditional random forest or cforest, SVM and neural network. By comparing the results of classifiers to each other, we focus more on random forest and cforest which gave us the best results.

Firstly, data preprocessing is one of the most important part. In the preprocessing procedure, we conduct several techniques, including missing value elimination, feature engineering, label balancing, categorical and numerical feature combination. One major aspect about the preprocessing is solving imbalanced. We have tried out undersampling, oversampling, ROSE approaches to balance the number of positive and negative tests.

Secondly, by working in R we realize the importance of documentation of parameters by looking at https://www.rdocumentation.org/. One classification algorithm can not be used for all classification problems. By trying different parameters, we also found that parameter have an effect on the validation results. Therefore, we integrate parameter tuning algorithm to generate best parameter combinations.

Thirdly, we try 7 different algorithms: Decision Tree(DT), Random Forest(RF), Conditional Random Forests (cforest), Naive Bayes, Neural Network(NN), support vector machine(SVM) and a Deep learning approach. By comparing different algorithms, we found cforest generate best results. We conduct two validations: local validation and kaggle validation, where Matthews correlation coefficient (MCC) are used as the performance matrix.

In summary, four preprocessing techniques has been used in this data set. We compare 7 different algorithms, and find cforest generate best results based on MCC matrix.

3. Data Exploration

- Dataset

The following table shows the variables present in the dataset.

Variable Name	Description	Variable Type
id	ID of the entry	Integer
age	Age of the customer	Neumeric
job	Type of job Available job types are admin, blue-collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed and unknown	Categorical
marital	Marital status - divorced, married, single, unknown	Categorical
education	Education status - basic.4y, basic.6y, basic.9y, high school, illiterat, professional course, university degree,'unknown	Categorical
default	Has credit in default?	Categorical
housing	Has housing loan?	Categorical
loan	Has personal loan?	Categorical
contact	Contact communication type, if it is cellular or telephone	Categorical
month	Last contact month of year	Categorical
day_of_week	Last contact day of the week	Categorical
duration	Last contact duration in seconds	Categorical
compaing	Number of contacts performed during this campaign and for this client	Numeric
pdays	Number of days that passed by after the client was last contacted from a previous campaign. Attribute value is 999 if the client is not contacted before	Numeric
previous	Number of contacts performed before this campaign and for this client	Numeric
poutcome	Outcome of the previous marketing campaign. If it was a success, a failure or if information is not available	Categorical
emp.var.rate	Employment variation rate - quarterly indicator	Numeric

cons.price.idx	Consumer price index - monthly indicator	Numeric
cons.conf.idx	Consumer confidence index - monthly indicator	Numeric
euribor3m	Euribor 3 month rate - daily indicator	Numeric
nr.employed	Number of employees - quarterly indicator	Numeric

- **Data Exploration**

Before trying out any predictive model, it is a best practice to investigate some of the statistical properties of the data, to get a better grasp of the problem. A first idea of the statistical properties of the data can be obtained through a summary of its descriptive statistics:

```
       id              age                 job            marital                    education
 Min.   :    0   Min.   :17.00   admin.     :7778   divorced: 3400   university.degree  :9110
 1st Qu.: 7722   1st Qu.:32.00   blue-collar:6926   married :18775   high.school        :7160
 Median :15445   Median :38.00   technician :5107   single  : 8654   basic.9y           :4561
 Mean   :15445   Mean   :40.02   services   :2985   unknown :   62   professional.course:3945
 3rd Qu.:23168   3rd Qu.:47.00   management :2175                    basic.4y           :3102
 Max.   :30890   Max.   :98.00   retired    :1287                    basic.6y           :1714
                                 (Other)    :4633                    (Other)            :1299
    default          housing          loan            contact          month       day_of_week
 no     :24426   no     :13945   no     :25510   cellular :19592   may    :10382   fri:5921
 unknown: 6463   unknown:  755   unknown:  755   telephone:11299   jul    : 5416   mon:6386
 yes    :    2   yes    :16191   yes    : 4626                     aug    : 4642   thu:6477
                                                                   jun    : 3965   tue:6001
                                                                   nov    : 3078   wed:6106
                                                                   apr    : 1925
                                                                   (Other): 1483
    duration         campaign          pdays          previous            poutcome
 Min.   :   0.0   Min.   : 1.000   Min.   :  0   Min.   :0.0000   failure    : 3221
 1st Qu.: 102.0   1st Qu.: 1.000   1st Qu.:999   1st Qu.:0.0000   nonexistent:26623
 Median : 180.0   Median : 2.000   Median :999   Median :0.0000   success    : 1047
 Mean   : 258.5   Mean   : 2.572   Mean   :962   Mean   :0.1743
 3rd Qu.: 319.0   3rd Qu.: 3.000   3rd Qu.:999   3rd Qu.:0.0000
 Max.   :4199.0   Max.   :56.000   Max.   :999   Max.   :7.0000

  emp.var.rate      cons.price.idx  cons.conf.idx      euribor3m      nr.employed     y
 Min.   :-3.40000   Min.   :92.20   Min.   :-50.80   Min.   :0.634   Min.   :4964   no :27461
 1st Qu.:-1.80000   1st Qu.:93.08   1st Qu.:-42.70   1st Qu.:1.344   1st Qu.:5099   yes: 3430
 Median : 1.10000   Median :93.75   Median :-41.80   Median :4.857   Median :5191
 Mean   : 0.08514   Mean   :93.58   Mean   :-40.49   Mean   :3.626   Mean   :5167
 3rd Qu.: 1.40000   3rd Qu.:93.99   3rd Qu.:-36.40   3rd Qu.:4.961   3rd Qu.:5228
 Max.   : 1.40000   Max.   :94.77   Max.   :-26.90   Max.   :5.045   Max.   :5228
```

Figure 1: The summary of training data

This summarization immediately gives an overview of the statistical properties of the data. In the case of nominal variables (which are represented by factors in R data frames), it provides frequency counts for each possible value. For numeric variables, R gives us a series of statistics like their mean, median, quartiles information and extreme values. These statistics provide a first idea of the distribution of the variable values. In the event of a variable having

some unknown values, their number is also shown following the string NAs. By observing the difference between medians and means, as well as the inter-quartile range, we can get an idea of the skewness of the distribution and its spread. Moreover, some graphs can capture the information better.

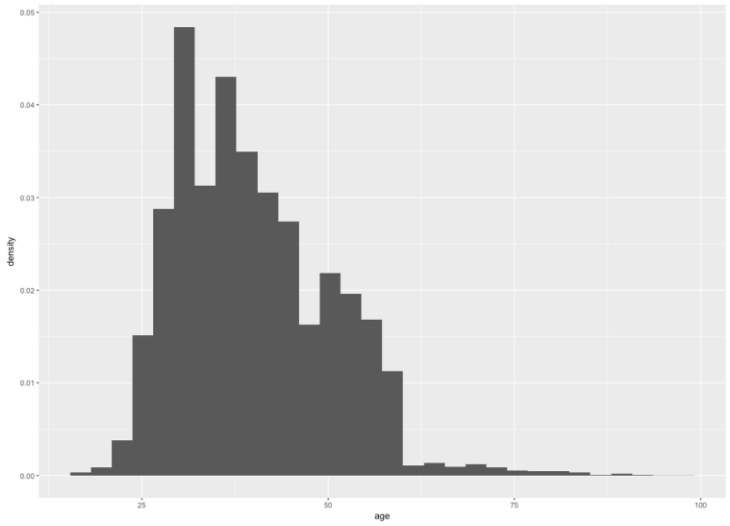

Figure 2: The histogram of variable age

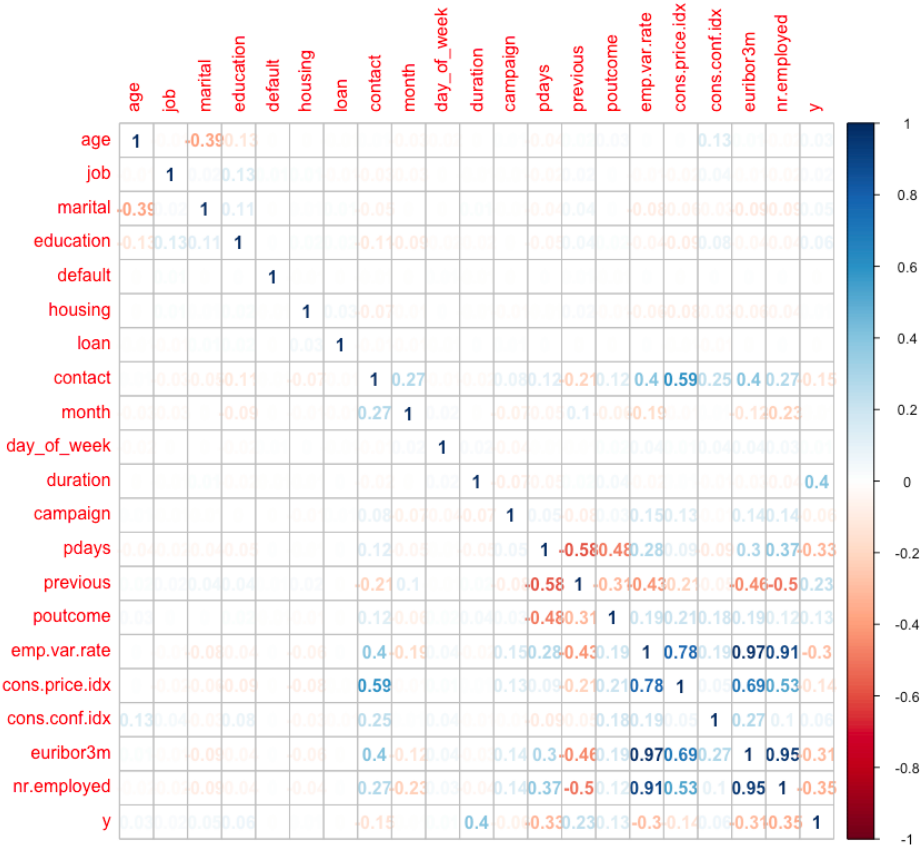

Figure 3: Plot of correlation of all attributes

Another example (Figure 4) showing this kind of data inspection can be achieved with box plot, this is the box plot for the promising values:

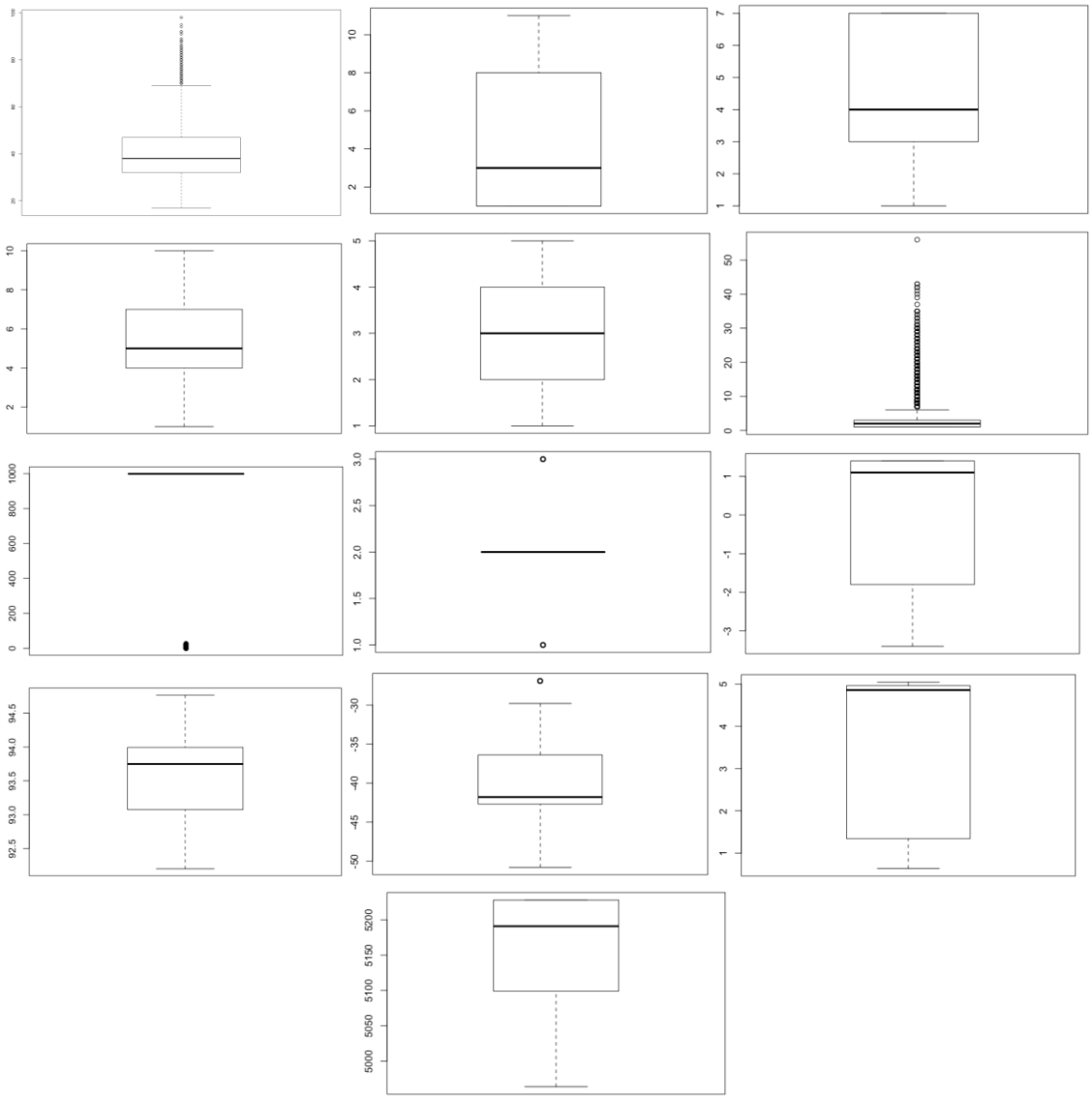

Figure 4 : Box plot for the following promising attributes: age, job, education, month, day_of_week, campaign, pdays, poutcome, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed

The analysis of Figure 5 tells us that the distribution of promising variables. Sometimes when we encounter outliers, we are interested in inspecting the observations that have these "strange" values. We can use box plot to find outliers.

- **Data Preprocessing**

- Handling missing values

There are several attributes with unknown values. This situation is rather common in real-world problems. There are various strategies to handle a dataset with missing values. we use the most common one: replacing the missing value by median for numeric columns and the most frequent value (the mode) for categorical variables.

- Feature engineering

There are 20 attributes in total but some are useless and some are very important. For example, the “id” is a variable in the process of modeling. We need remove attribute “id” to clean the data. In addition, we transfer the character to factor and categorical values to be integer values for further processing. Except the useless one, there are also differences between the importance of attributes. In the process of random forest modeling, the features we choose for the modeling are important features rather than all of them. To use the meaningful features, we also use pca for preprocessing.

- An unbalanced dataset

The class distribution in the dataset is unbalanced. Only 3430 (11.10%) instances are related with success stories of the marketing campaign. However, machine learning algorithms such as Neural Networks prefer balanced dataset because unbalanced cause for biased predictions and misleading accuracies. The techniques to tackle unbalanced data issues are undersampling, oversampling, synthetic data generation and cost sensitive learning. In this project, to balance training data we used ROSE (Random Over Sampling Examples) package in R. It helps to generate artificial data based on sampling methods. Since oversampling and undersampling can cause repeated information and deprived information with respect to the original data, we used inbuilt ROSE function of R ROSE package. This function synthetically data and make the data balanced. The generated new data set will contains same amount of data as the original dataset and the distribution of the each class is roughly 0.5. To compare effect of data balancing see Fig. 5. It shows the change in MCC score of the validation dataset over training epoch for Neural Network training.

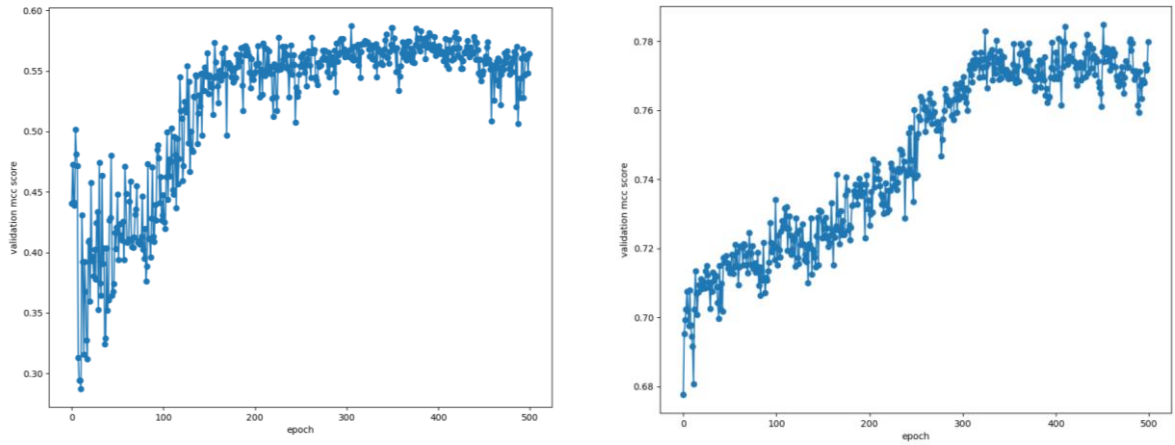

Figure 5: Validation data set MCC socre variation before balancing the dataset (left) and after balancing the dataset

- Standardization data

Another issue addressed by preprocessing is varied scales of various attributes. This does not have an impact on models based on decision trees as they look for information gain from each attribute individually at each internal node. However, addressing this is very important for some predict models such as Neural Networks as attributes in a relatively larger scale can dominate the prediction. To address this after above preprocessing steps, data set was scaled such that each column is having zero mean and 1 standard deviation.

- Dataset is a combination of continuous and categorical variables

According to Table I this data contains both continuous and categorical data. Even though this doesn't have an impact on algorithms based on decision trees, this should be especifically addressed for Neural Networks. To transform categorical data so that we can use those with MLP there are many approaches. One is to use numeric encoding. However, this works only if order has a meaning for the categorical data values. Otherwise the distance among values will give false interpretation. For example we can use a numeric encoding to represent day of the week, Monday-1, Tuesday-2, Wednesday-3, Thursday-4, Friday-5, Saturday-6 and Sunday-7. In here even though the distance between Monday and Tuesday and distance between Monday and Sunday should be same that information is not preserved. The issue is same if we use binary encoding. Therefore in this project to encode categorical attributes for MLP we used one hot representation. For one categorical attribute one-hot mapping introduces n number of new attributes where n equals to number of levels for the attribute realization.

4. Training and Testing Procedure

To evaluate the performance of each model, we conduct two validation: local validation and kaggle validation. For local validation, we split a small testing set from original training set, where the remaining training set are used for model training and selected testing set for model evaluation. For kaggle validation, we use all the original training set for training procedure and upload the predicate results of testing set to kaggle. Generally, the MCC of local validation and kaggle validation is similar.

To evaluate developed model for their prediction accuracy we used a validation dataset. The training data set is divided into two subsets for training and for validation with ratios by 75:25. As the label is imbalance (Y:N = 9:1), so we can not perform fully random selection, otherwise the distribution of selected training set and validation set will be different. To deal with this problem, we randomly split positive item and negative separately, and then form the validation set, while keeping the distribution of selected training set and validation set in consistency. We used Matthews correlation coefficient (MCC) as our evaluation matrix locally, which is the evaluation matrix in kaggle.

Then, if the MCC of current experiment is better than the best MCC, the model is trained with the whole data set and applied to the test set and finally the result is submitted to kaggle. However, we sometimes submit our submissions just for comparing results of different classifiers to each other since the results of SVM, for example, is significantly worse than the current best result at the time of that experiment.

Analyzed data mining models

- Decision trees

Decision tree learning uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modelling approaches used in statistics.
In data mining, decision trees can be described also as the combination of mathematical and computational techniques to aid the description, categorization and generalization of a given set of data.
Data comes in records of the form:
$$(\mathbf{x}, Y) = (x_1, x_2, x_3, \ldots, x_k, Y)$$
The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of the features, x1, x2, x3 etc., that are used for that task.

- Naive Bayes

In this model, given a problem instance to be classified, represented by a vector

$$\mathbf{x} = (x_1, \ldots, x_n)$$

representing some *n* features (independent variables), it assigns to this instance probabilities

$$p(C_k \mid x_1, \ldots, x_n)$$

For each of k possible outcomes or class C_k.
As it assump all the features are independent, so

$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\ &\propto p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,. \end{aligned}$$

We could use the training set to figure out the prior probability, and then, for each test item, we will predict its label based on:

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}}\ p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

- Random Forest

Decision trees are a popular method for various machine learning tasks. Random Forest is one of the decision tree.

In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set X = x1, ..., xn with responses Y = y1, ..., yn, bagging repeatedly (B times) selects a random sample with replacement of the training set and fits trees to these samples:

For b = 1, ..., B:

1. Sample, with replacement, n training examples from X, Y; call these Xb, Yb.
2. Train a classification or regression tree fb on Xb, Yb.

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x':

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b(x')$$

or by taking the majority vote in the case of classification trees.

- Neural Network

Neural Networks' capability to model very complex functions has caused for its immense success in myriad of fields. A simple form of Multi-Layer perceptron (MLP) will have an input layer, a hidden layer and an output layer. While input layer contains number of neurons equal to input dimension, the output layer contains number of neurons equal to number of classes. The number of neurons in the hidden layer is a hyper parameter fine-tuned by the user. The neurons in each layer of a Neural Network receive an affine transformation of the output of the previous layer and output activation values depending on the used nonlinear function.

The mathematical operations of a Neural Network can be expressed as follows,

$h_i = f(WX_{i-1} + b)$

Where h_i is the activations of the ith layer, f is the nonlinear activation function $W$is the weight matrix between i and $i-1$ layer and b is the bias term.

If we want to build a model with a complex non-linear function we can stack multiple hidden layers and build a deeper network. The intention of increasing number of hidden layers is to allow the model to learn a more complex function.

Since our problem, predict success of bank marketing campaign consist of 20 parameters and it seemed to be a non-linear binary classification problem, we tried out a deep MLP.

5. Experimental Study and Analysis

Decision Tree

The rpart package found in the R tool can be used for classification by decision trees and can also be used to generate regression trees. Recursive partitioning is a fundamental tool in data mining. First, we will use a classification tree to predict the possibility for subscribing the deposit. After handling data sets with missing values, we need to remove the attribute “id” for the reasons mentioned before. To grow a tree, we use

> rpart (formula, data=, method=, control=)

formula : is in the format outcome ~ predictor1+predictor2+predictor3+ect.

data= : specifies the data frame

method= : "**class**" for a classification tree , "**anova**" for a regression tree

control= : optional parameters for controlling tree growth.

The instructions necessary to obtain a regression tree are presented below:

> library(rpart)

> fit <- rpart(y ~ .,method="class", data=cleandata)

The training dataset is divided into two parts. 75% of them is used as training and 25% of them is used as validation. The decision tree is evaluated by 25% of training dataset as the follows:

Precision: 0.632

Recall: 0.488

F: 0.275

Area under the curve (AUC): 0.837

Score uploaded on Kaggle: 0.49619

```
Classification tree:
rpart(formula = y ~ ., data = cleandata, method = "class")

Variables actually used in tree construction:
[1] duration    nr.employed poutcome

Root node error: 3430/30891 = 0.11104

n= 30891

        CP nsplit rel error  xerror     xstd
1 0.071720      0   1.00000 1.00000 0.016099
2 0.024344      2   0.85656 0.86268 0.015080
3 0.018076      4   0.80787 0.80787 0.014643
4 0.010000      6   0.77172 0.78746 0.014474
```

Figure 6: Summary of the decision tree

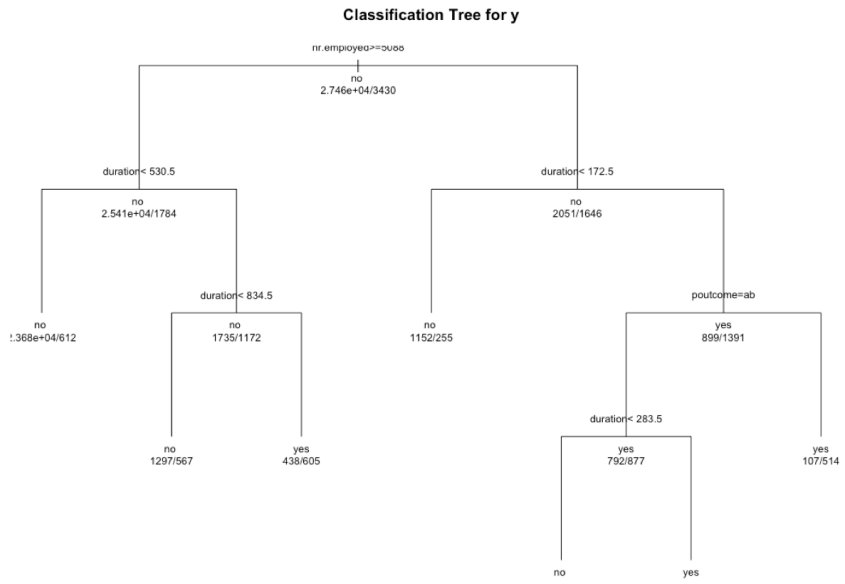

Figure 7: Decision tree for the training dataset

Naive Bayes

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. For example, the algorithm assumes that attributes such as job and education are independent from each other in predicting whether a customer will open a bank account or not.

We have tried this Naive Bayes strategies, but we found the result is not very promising. Just as we mentioned, this algorithm assume that all the features are independent. However, in this dataset, we found the feature does not satisfied this property. For example, we found strong relationship between Age and Marital. In summary, Naive Bayes is not a good choice for this dataset.

SVM

The first SVM model is the default one, training with the R command:

svm(y ~ ., train)

with train denotes the training data. The result is really low, just **0.44721**. Therefore, we just experiment with one another SVM model

smv(y ~ ., cost=10, kernel=”polynomial”, degree=3)

which also returned a bad result, **0.44046**. Hence, we decided to stop trying with SVM model.

Random Forest

Firstly, we train with all attributes by randomForest function

randomForest(factor(y) ~ ., data = train)

which gave a really good result, 0.55137, compared to other methods. To improve our results, the importances of attributes is considered since random forest can extract the order of importances. Then, not all attributes are considered in the subsequent tests, we choose the first 14 attributes in the importance ranking, namely age, job, education, month, day_of_week, campaign, pdays, poutcome, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed. Below is the importance ranking of all attributes, extracted by random forest classifier.

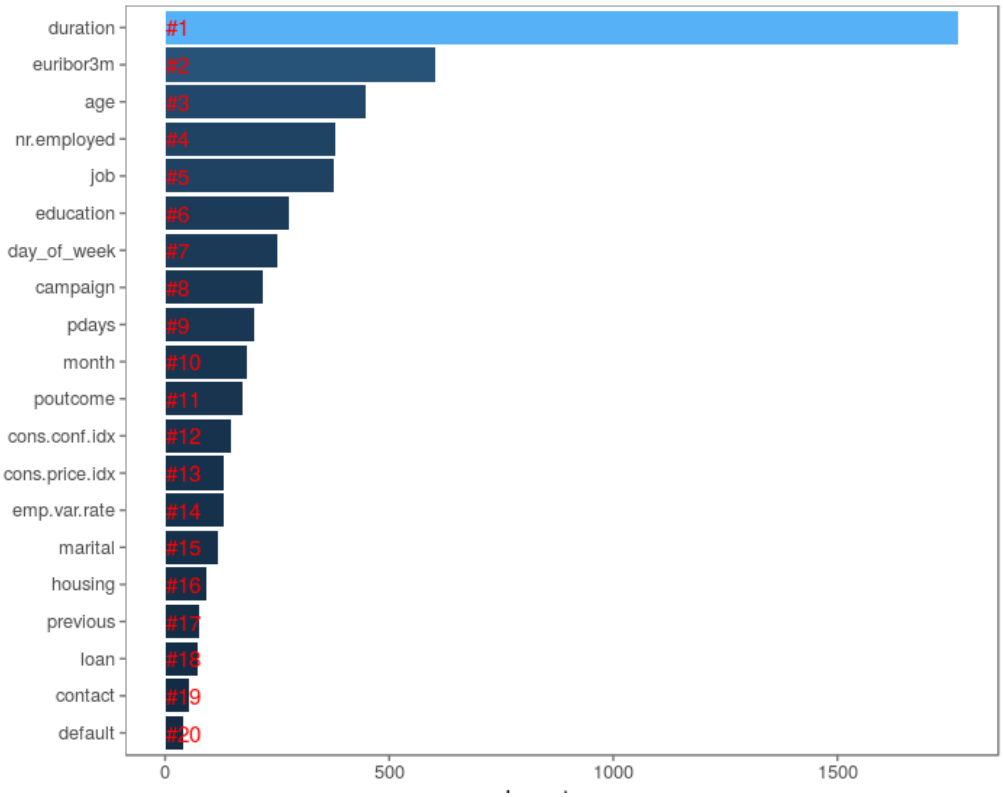

Figure 6 : Attribute importance ranking by random forest.

Secondly, we tune in attributes by some functions. Parameters are very important in random forest algorithm. Generally, *mtry* and *ntree* are the top two important parameters. To figure out the best parameters for this data set, we conducted some parameter tune algorithms.

Firstly, the range of parameters are provided manually. For *mtree*, we set the range from 500 to 2000. For *mtry*, we set its range from 2 to 10, while *nodesize* from 10 to 50. To find best results, the random controller are used to search different parameter combinations for evaluation. For each parameter combination, a resampling method from mlr package of R are used. Finally, we select the best parameter combination to train the model and then generate predictive labels.

Conditional Random Forest cforest

Traditional decision tree uses Gini index to decide which attribute to split. Therefore, it tends to select attributes that have many possible splits or many missing values. To avoid this limitation, conditional decision trees utilize significance test to determine slitting attributes.
The function cforest is implemented in **partykit** package. Then, the parameters are modified according to the documentation in rdocumentation.org. The changing of parameters are almost the same as random forest but got a limitation on training time. The training time of cforest is significantly much larger than random forest, about 30 minutes for 80% training set and about an hour for the whole training set. Hence, we just experiment with limited time but manage to get the best result by cforest. The syntax of cforest is almost the same as random forest, as we just change from **randomForest.importance** to **cforest.importance** in the method parameter.

Neural Network and Deep Learning

Before applying a Neural Network for this dataset preprocessing phase was essential to handle following issues with the dataset:

- Dataset is highly imbalanced
- Presence of categorical variables - (This has to be especially addressed for neural networks.)
- Attributes are in different scales

How we handled these issues was discussed in data preprocessing section.

For the implementation of the deep learning model we used "Tensorflow" deep learning framework. The validation dataset performance was measured using MCC matrix. 10% of the given training dataset was randomly selected as the validation set. The deep learning model consists of seven hidden layers each with 256 hidden neurons each using tanh activation function. The model is trained as batches of the training dataset using Adam optimization technique. To improve training procedure weights are initialized using xavier initialization and decaying learning rate is used. To prevent the model being overfit to training data dropout was used. When dropout is applied in training some neurons of the network are turned off with some probability p. This can be regarded as sampling a neural network from the full neural network. Dropout helps to improve MCC score of validation dataset.

Another, practical technique to improve performance of a classification model is to use an ensemble of models. For this project also we tried an ensemble of neural networks. This also helped to improve MCC score of the validation dataset a little bit (See. ……..)

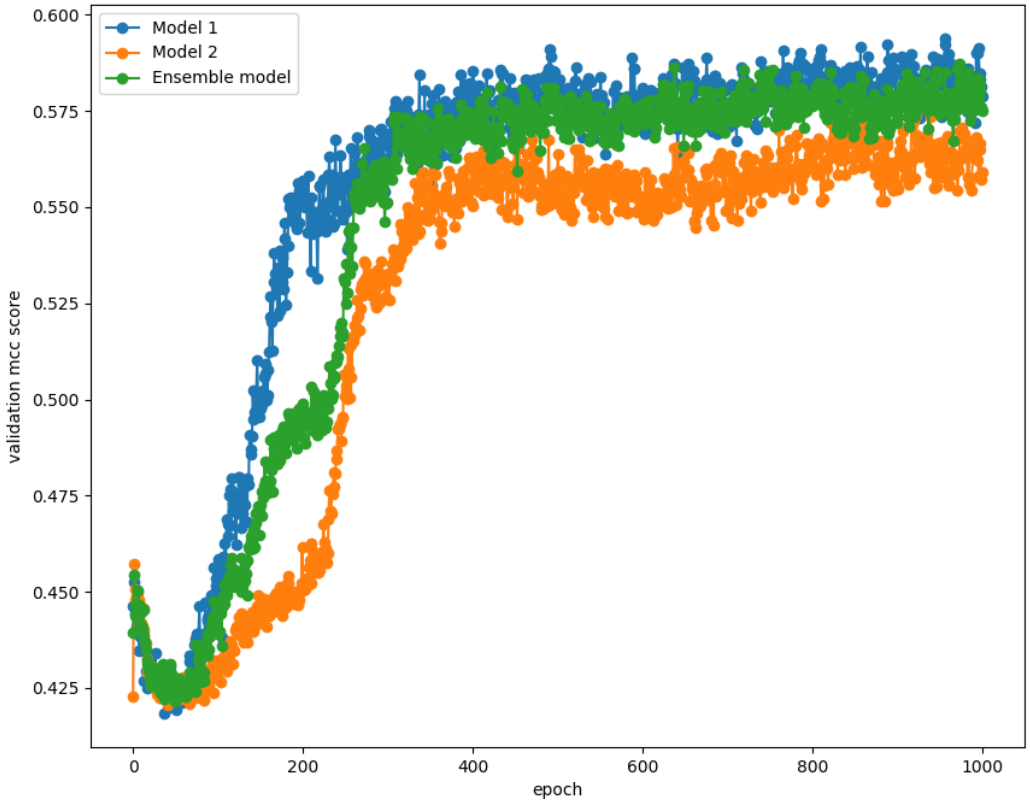

Figure 8 : Validation dataset MCC score variation with epoch for single models and ensemble model

6. Summary of Project Achievements

The second column of the following table shows the Area Under the Curve (AUC) metric for the validation dataset for each predictive model we tried out. AUC was used to compare accuracy of different predictive models. The third column shows the results of the test dataset we achieved when we submit our results to Kaggle.

Predictive Model	**AUC**	**Results for the test dataset from Kaggle - MCC**
Decision Trees	0.837	0.49619
Naive Bayes	0.709	0.42943
SVM	0.667	0.44721
Random Forest	0.862	0.61059
C-Random Forest	0.718	0.61346
Neural Networks - single MLP model	0.8672	0.54685
Neural Networks - ensemble of MLP models	0.8765	0.55693

Based on this results we selected C-Random Forest as the best model to predict success of direct marketing campaign of this bank.

7. Future Directions for further Improvements

- Following a more rigorous feature engineering process

Selecting the most important features related for the prediction is imperative to build a better prediction model. In this project we used importance score given by the Random Forest algorithm to decide what are the important features. However, we can further analyze these given features if we can consult a domain expert on that. As pointed out by some research papers, we can make a questionnaire and consult domain experts to identify more important features affecting the success of marketing campaign. This can be combined with an embedded feature selection approach to extract optimal subset of features. Using recently proposed Deep Learning based feature engineering approaches would be another promising direction.

- Following a better data preprocessing steps

We hope that our predictive results can be further improved by more insightful data preprocessing. Even though we have applied many preprocessing techniques, we believe that there is still room to improve preprocessing customized for each predictive model. Especially we could look at how we can combine features or how we can use dimensionality reduction techniques to derive new features.

- Explore more predictive models

We would explore other predictive models such as recent Deep Learning models for this predictive task. Further, we would try out more advanced model ensemble techniques to improve prediction results.