A Machine Learning Model for Marketing

Project Overview

Businesses today often launch marketing campaigns to boost the sale of their products and services. While digital marketing becomes popular and have many advantages over traditional marketing, traditional marketing methods are still providing physical customer experience difficult to be offered by digital marketing and could not be completely replaced by digital marketing[1]. One of the drawbacks of traditional marketing is that it is typically more expensive than digital marketing since it involves one or multiple types of activities such as phone calls, customer visits, or physical prints etc. These activities often require significant efforts and investment from businesses. Therefore, for traditional marketing, it is important to target marketing activities towards desirable customers who are more likely to buy products and services than others. It will not be cost-effective if marketing campaign targets are simply randomly chosen without going through thorough review and selection.

Machine learning can provide a data-driven approach to help marketing campaign more targeted to desirable customers. In this project, using bank telemarketing as an example of traditional marketing, a machine learning model was developed and demonstrated its effectiveness in maximizing business return while minimizing marketing effort.

Problem Statement

In financial industry, banks rely on telemarketing to gain more term deposit, which involves bank agents calling clients directly[2]. Ideally banks want to only contact those clients who will subscribe in order to keep the scale and cost of their marketing campaign under control. However, even though they have a lot of information about their clients and economic environment, they do not know who will subscribe after being contacted. Trying to contact all the clients will be costly, unrealistic, and ineffective.

A machine learning algorithm can be developed based on historical data[3]. This algorithm will predict which clients likely subscribe term deposit before bank agents make phone calls. It will be also able to help banks to decide how many clients need to be contacted in order to meet their business target, based on which banks can plan the scope, budget and resources of marketing campaign accordingly.

Evaluation Metrics

The evaluation metric will include Accuracy, Recall, Precision, ROC AUC, and Cumulative Gain, all of which are appropriate in the context of telemarketing. Accuracy is not a good metrics here and only for

reference use since the dataset is imbalanced and predominantly biased towards negative class (non-subscriber). Instead, Recall should be emphasized as well for both training and evaluating model as telemarketing campaign tries to identify more clients who will subscribe. ROC AUC and Cumulative Gain were used in a similar study by Moro et al[2].

The definition of evaluation metrics is listed below:

Recall = True Positive / (True Positive + False Negative). It measures how many clients will be predicted accurately to subscribe term deposit among all the clients who actually subscribe. Higher Recall value is preferred for a good model.

Precision = True Positive / (True Positive + False Positive). It measures how many clients actually subscribe among all the clients who are predicted to subscribe. Higher Precision is preferred for a good model.

ROC curve is the curve generated by plotting True Positive Rate (TPR = Recall) against False Positive Rate (FPR). It maps the relationship between TPR and FPR according to the probability threshold for binary classification. Instead of default threshold value 0.5, probability threshold can be adjusted to achieve the best balance between TPR and FPR under specific business context. AUC measures the area under the ROC curve and is between 0.5 and 1.0. AUC = 0.5 reflects a random classifier not able to make prediction at all. AUC = 1.0 represents a perfect classifier where TPR = 1 and FPR = 0. A classifier is better if AUC is higher (equivalent to higher TPR and lower FPR).

Cumulative Lift is calculated in the following way based on created predictive model[4]:

1) Calculate the **predicted** class probability for each sample;
2) Rank these probabilities in decreasing order;
3) Build deciles with each group having 10% of the total sample population;
4) Calculate the response rate at each decile for Responders (clients who **actually** subscribe).
5) The cumulative response rate can be compared to cumulative population percentage visualized by graphs (An example below, Figure 1).

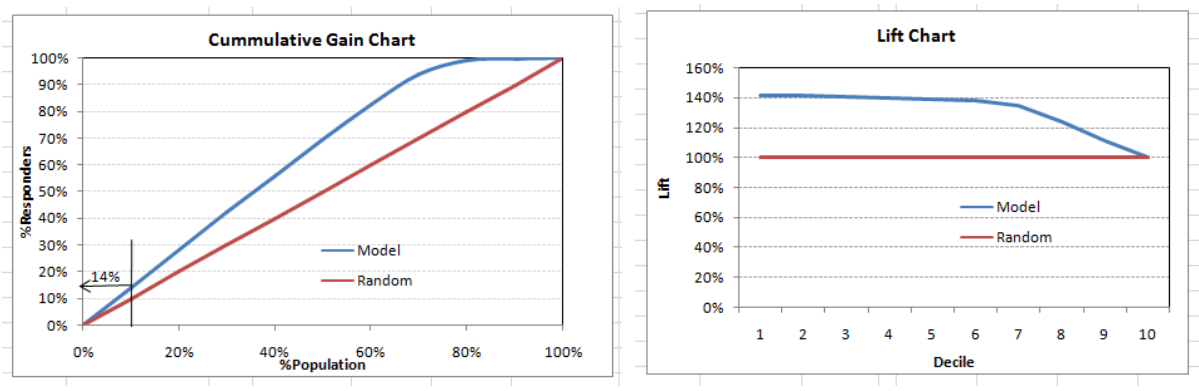

Figure 1 Example of Cumulative Gain Chart and Lift Chart

Cumulative Gain measures the cumulative percentage of clients who subscribe out of the total number of clients who subscribe relative to the cumulative percentage of all the clients (including who do or do not subscribe). In a random model, if X% of client population is contacted by phone calls, the same X% of clients who actually subscribe is expected. In a good model, clients who subscribe are more likely to be called so that the cumulative percentage from good model, Y%, will become higher than X%. Comparing Y% and X% will tell the performance of the model. The bigger Y% is compared to X%, the better is the model. Lift is another way to quantify Cumulative Gain by dividing Cumulative Gain by corresponding cumulative population% (Figure 1). Cumulative Gain and Lift are very important in the domain of telemarketing or similar marketing situations. It is not realistic to reach out to every client to find who will deposit. It is also not realistic to have a perfect predictive model to identify all the clients who will deposit from the client base. Instead, the success metrics should be to contact more clients who will subscribe while the total number of phone calls does not increase. In this way, marketing return is maximized with marketing investment minimized.

Analysis

Data Exploration

The dataset contains 41188 samples with 20 features and 1 class label[3]. Among 20 features, 10 are numeric and the other 10 are categorical. The class label is "yes" (subscribe) or "no" (not subscribe) so it is a binary classification label.

This dataset was originally described by Moro et al. in 2014[2]. The original dataset has bigger sample size (52,944) and more features (150). Moro et al. reduced the number of features through feature selection and used only 22 features to create predictive models. In UCI repository, the list of features was changed again from what Moro et al. used in 2014[3]. In particular, one feature in UCI depository dataset, "duration', was noted that it should be only included for benchmark purpose but should not be used to generate a realistic predictive model[3]. Therefore, 19 features will be used for model generation in the solution provided here. Table 1 lists all the features, data type and definition.

Table 1 Feature Information

Feature	Data Type	Definition
age	Numeric	Age of client
duration	Numeric	Last contact duration in seconds
campaign	Numeric	Number of contacts performed during this campaign and for this client
pdays	Numeric	Number of days that passed by after the client was last contacted from a previous campaign
previous	Numeric	Number of contacts performed before this campaign
emp.var.rate	Numeric	Employment variation rate – quarterly indicator
cons.price.idx	Numeric	Consumer price index – monthly indicator
cons.conf.idx	Numeric	Consumer confidence index – monthly indicator
euribor3m	Numeric	Euribor 3 month rate – daily indicator
nr.employed	Numeric	Number of employees – quarterly indicator
job	Categorical	Type of job
marital	Categorical	Marital status
education	Categorical	Education

Feature	Data Type	Definition
default	Categorical	Has credit in default?
housing	Categorical	Has housing loan?
loan	Categorical	Has personal loan?
contact	Categorical	Contact communication type
month	Categorical	Last contact month of year
day_of_week	Categorical	Last contact day of the week
poutcome	Categorical	Outcome of the previous marketing campaign
y	Categorical	Has the client subscribed a term deposit? (binary: 'yes', 'no')

The dataset is a CSV file and was imported into Jupyter Notebook as a Pandas dataframe. After examining the dataframe for summary information, there is no missing value for any feature (total 41188 non-null int64, object, float64). The dataset on UCI Repository has been preprocessed and anything without data was already labeled as 'unknown'. The descriptive statistics for each numerical feature was shown in Table 2.

Table 2 Descriptive Statistics Analysis of Dataset

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
count	41188.00000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000
mean	40.02406	258.285010	2.567593	962.475454	0.172963	0.081886	93.575664	-40.502600	3.621291	5167.035911
std	10.42125	259.279249	2.770014	186.910907	0.494901	1.570960	0.578840	4.628198	1.734447	72.251528
min	17.00000	0.000000	1.000000	0.000000	0.000000	-3.400000	92.201000	-50.800000	0.634000	4963.600000
25%	32.00000	102.000000	1.000000	999.000000	0.000000	-1.800000	93.075000	-42.700000	1.344000	5099.100000
50%	38.00000	180.000000	2.000000	999.000000	0.000000	1.100000	93.749000	-41.800000	4.857000	5191.000000
75%	47.00000	319.000000	3.000000	999.000000	0.000000	1.400000	93.994000	-36.400000	4.961000	5228.100000
max	98.00000	4918.000000	56.000000	999.000000	7.000000	1.400000	94.767000	-26.900000	5.045000	5228.100000

Exploratory Visualization

By examining the histogram of all numerical features and their descriptive statistics (Table 2), although most of features are not distributed normally, there are few features with extremely skewed distribution except 'pdays'. However, UCI Repository intentionally used '999' for 'pdays' to arbitrarily represent clients not previously contacted[3] and is not really outliers. Thus, no data transformation was performed.

This dataset is heavily biased for class label ('y') (Figure 2). 88.7% of samples (36548) are labeled as "no" and only 11.3% (4640) are labeled as "yes". In order to avoid the bias during model learning, upsampling was performed to elevate the number of positive class to be the same as that of negative class in training dataset[5].

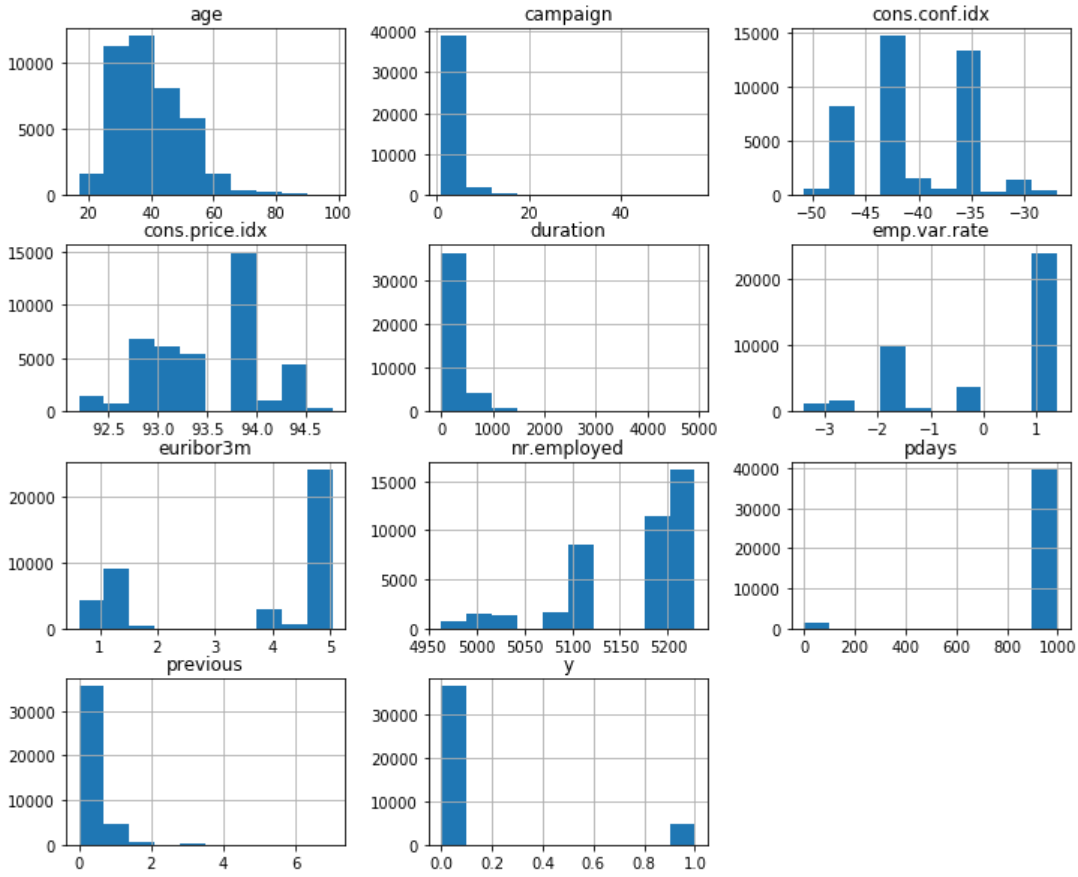

Figure 2 Histogram of Numerical Features

Bar charts were created for categorical features ('job' visualization as an example in Figure 3). OneHot Encoding was performed for all the categorical features.

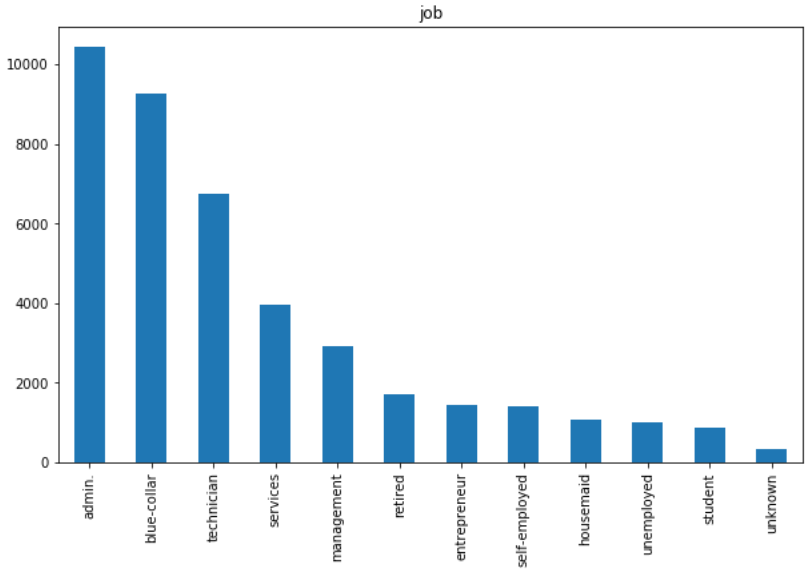

Figure 3 Bar Chart of Job Feature

Correlation matrix analysis was performed for all numerical features with Pearson's r^2 values calculated (Table 3). Scatter Matrix plots were also generated for easy visualization (Figure 4). There are a few features showing correlation with y label and are expected to have greater contribution to classification algorithm than others. They include 'duration', 'pdays', 'previous', 'emp.var.rate', 'euribor3m', and 'nr.employed'. For 'duration', while it shows the strongest relationship with y label, it will not be included in training as discussed in the section of Data Exploration. The correlation among 'emp.var.rate', 'euribor3m', and 'nr.employed' are very strong (Pearson's r^2 are close to 1), which is consistent with the fact that they are indicators for macro-economic environment as defined in Table 1. While all of them will be included for training classifiers, these strong relationships need to be considered when algorithms are evaluated and interpreted.

Table 3 Pearson's r^2 from Correlation Matrix Analysis

	Age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	y
Age	1.000000	-0.000866	0.004594	-0.034369	0.024365	-0.000371	0.000857	0.129372	0.010767	-0.017725	0.030399
Duration	-0.000866	1.000000	-0.071699	-0.047577	0.020640	-0.027968	0.005312	-0.008173	-0.032897	-0.044703	0.405274
Campaign	0.004594	-0.071699	1.000000	0.052584	-0.079141	0.150754	0.127836	-0.013733	0.135133	0.144095	-0.066357
Pdays	-0.034369	-0.047577	0.052584	1.000000	-0.587514	0.271004	0.078889	-0.091342	0.296899	0.372605	-0.324914
Previous	0.024365	0.020640	-0.079141	-0.587514	1.000000	-0.420489	-0.203130	-0.050936	-0.454494	-0.501333	0.230181
emp.var.rate	-0.000371	-0.027968	0.150754	0.271004	-0.420489	1.000000	0.775334	0.196041	0.972245	0.906970	-0.298334
cons.price.idx	0.000857	0.005312	0.127836	0.078889	-0.203130	0.775334	1.000000	0.058986	0.688230	0.522034	-0.136211
cons.conf.idx	0.129372	-0.008173	-0.013733	-0.091342	-0.050936	0.196041	0.058986	1.000000	0.277686	0.100513	0.054878
euribor3m	0.010767	-0.032897	0.135133	0.296899	-0.454494	0.972245	0.688230	0.277686	1.000000	0.945154	-0.307771
nr.employed	-0.017725	-0.044703	0.144095	0.372605	-0.501333	0.906970	0.522034	0.100513	0.945154	1.000000	-0.354678
Y	0.030399	0.405274	-0.066357	-0.324914	0.230181	-0.298334	-0.136211	0.054878	-0.307771	-0.354678	1.000000

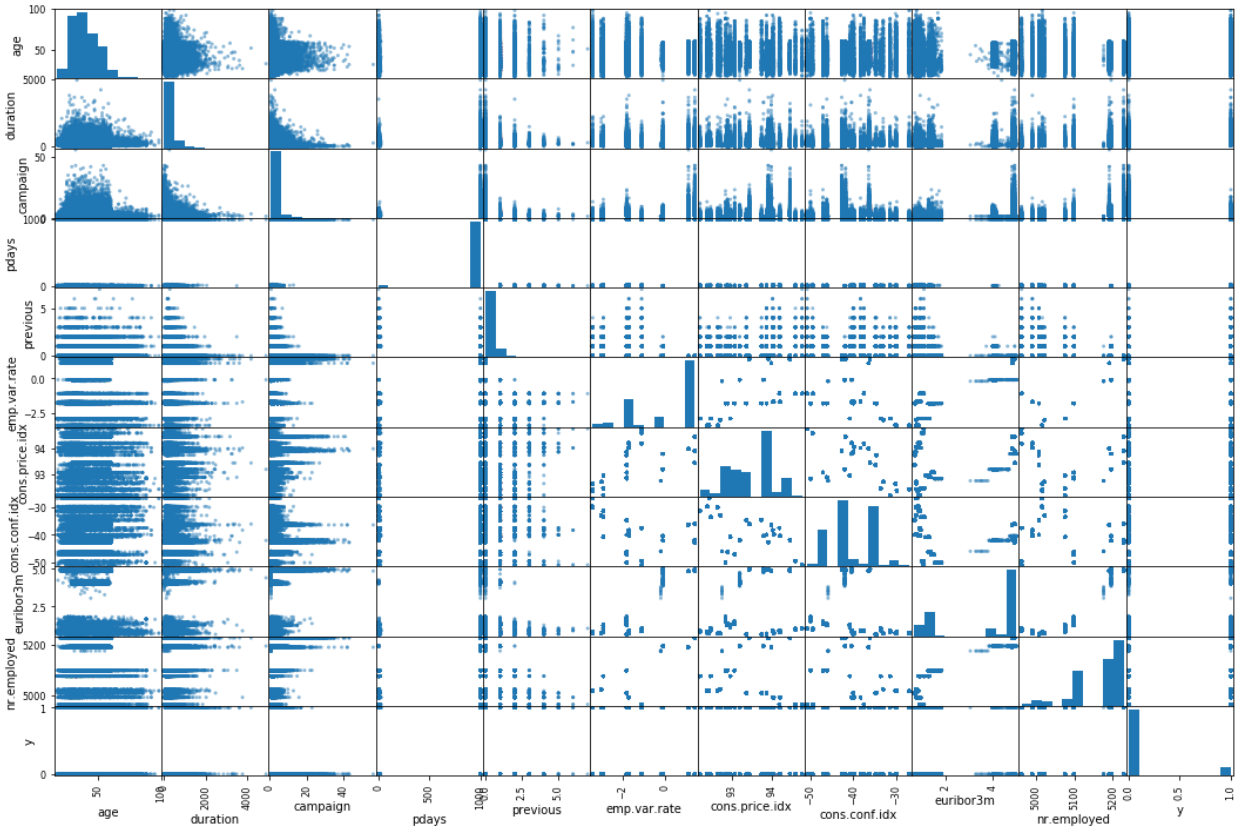

Figure 4 Scatter Matrix for Feature Relationship Visualization

Algorithms and Techniques

Several classification algorithms are applicable for the binary classification problem here. Since ROC and Cumulative Gain analysis will be performed as metrics evaluation, an ideal classifier should provide class probability prediction for each client. All the algorithms chosen below meet this criterion. Also the training size is relatively large (can reach ~70,000 after upsampling is performed on rare positive class) so the chosen classifier should handle large training size well. Algorithms such as SVM are slow to train and are not considered in this project. The chosen algorithms are listed below with training parameters listed.

1) Gaussian Naïve Bayes: it does not require training parameter to tune. It can quickly establish a benchmark model as baseline.
2) Decision Tree: easy to extract feature importance and help explain the model.
    a. max_depth: how deep is the decision tree allowed to split?
    b. min_samples_leaf: minimum number of samples allowed on each tree leaf
    c. min_samples_split: minimum number of samples on each leaf required to split to next level
3) Logistic Regression: can handle larger training size and smaller feature size well.
    a. Learning factor C: determine how fast the algorithm learns or how big is the gradient descent step.
    b. Penalty: regularization factor L1 or L2 to reduce overfitting
4) Random Forest: this ensemble model may result in a stronger classifier.
    a. max_depth
    b. min_samples_leaf
    c. min_samples_split
5) Neural network: may result in a model handling non-linear separation well.
    a. Layer architecture
    b. Activation function
    c. Dropout

Benchmark Model

A few models were generated in R by Moro et al. in 2014, including Logistic Regression, Decision Tree, SVM, and Neural Network[2]. However, these models did not reflect true telemarking process since "duration" feature was included and bank agents do not know the duration of phone calls before they call clients[3]. This is one of the drawbacks of these R models and limits their practical application. For such reason, UCI repository asks to exclude this feature from a realistic prediction model[3]. These models were trained on a bigger dataset with some of features different from what will be used in this project. They also did not use special handling strategy on imbalanced dataset. This project will establish learning algorithm under different data scenario and balanced model sampling strategy in Python. Therefore, the established R models are not ideal benchmark models.

A new benchmark model will be created with Gaussian Naïve Bayes classifier in this project. This incredibly simple algorithm can quickly establish a baseline performance for improvement.

Methodology

Data Preprocessing

The dataset has gone through the following data preprocessing:

1) Binary encoding y class label to 0 (no) and 1 (yes).
2) Use MinMaxScaler to scale numeric features since algorithms such as logistic regression and neural network require feature scaling.
3) OneHotEncoding multi-class categorical features.
4) Drop the 'duration' feature based on the request from NCI Repository to develop a realistic prediction model.
5) Perform stratified shuffle split based on y label class to create balanced train and test dataset (0.8:0.2) since y label is heavily biased. This project will use k-fold cross-validation strategy for most of classification algorithms to fully leverage the entire train dataset for learning so there is no validation dataset created initially. However, designated validation data set is created for neural network since keras module does not support k-fold cross validation.
6) Perform upsampling for training set to randomly increase the sample size of positive class (yes, 1) to match that of negative class (no, 0)[5]. The test dataset remains untouched.

Implementation

Define Functions for Modeling and Graphing

1) fit_classifier: grid search using 10-fold cross validation, fit model, and report accuracy, precision, recall, ROC AUC score, and best parameters on train dataset;
2) classifier_test: test accuracy, precision, recall, ROC AUC score on test dataset;
3) roc_curve_plot: make ROC curve graph;
4) lift_input: create data input for Cumulative Gain and Lift graph analysis;
5) Gain_plot: make cumulative gain graph;
6) lift_chart: calculate Lift and make a Lift graph;
7) y_predict_threshold: classify clients based on population sampling requirement.

Define and Train Models

After considering the optimization goal (identify more clients who are likely to subscribe term deposit), training optimization metric was decided to be 'recall'. There is some risk that models may be optimized so well on recall that too many false positive will be predicted and impact marketing efficiency. This concern will be firstly addressed to some degree by cross validation and quick analytical performance verification on test dataset. It will be finally addressed by evaluating cumulative lift and lift gain, which are the ultimate marketing goal. When each model was trained, 2-3 critical parameters were used in grid search for initial optimization. After model evaluation selects the best algorithm, more model parameters will be used for further fine tuning and improvement (Table).

When performing training, algorithms such as Gaussian Naïve Bayes, Decision Tree, Logistic Regression, Random Forest were imported from sckit-learn. Neural Network learning was performed in Keras using Tensorflow as backend. After training and achieving the best estimators, the performance of each model was quickly verified in test dataset by accuracy, precision, recall, and roc_auc with recall as the primary metric. There was no further training parameter tuning after validation on test dataset.

1) Created a Gaussian Naïve Bayes classifier as a bench model. Parameter tuning is not needed.
2) Created a Decision Tree classifier. 'max_depth' (best = 7), 'min_samples_leaf' (best = 2), and 'min_sample_split' (best = 2) were progressively tuned while comparing parameter by examining best_parameter output. The top 10 important features were plotted by bar graph.
3) Created a Logistic Regression classifier. 'C' (learning factor, best = 0.0007) and 'penalty' (best = L1) were progressively tuned while comparing parameter by examining best_parameter output.
4) Created a Random Forest ensemble classifier. 'max_depth' (best = 6), 'min_samples_leaf' (best = 2), and 'min_sample_split' (best = 2) were progressively tuned while comparing parameter by examining best_parameter output.
5) Created a Neural Network (NN) classifier.
    a. Shuffle splitted train dataset to smaller train set and validation set (0.8:0.2).
    b. Defined NN sequential layer architecture (128x64x1, 64x32x1, 64x1, 48x1, 32x1), activation functions ('relu' and 'sigmoid' for hidden layers and 'sigmoid' for output layer), epoch = 10, 20, 30, 40, dropout (0.1, 0.2). Since overfitting was the main issue, then dropout 0.2 was used throughout the training later.
    c. Defined 'recall' as model optimization target metric. Since there is no built-in 'recall' metric in keras Sequential model, a deprecated 'recall' metric code was downloaded from internet[6] and imported as an external custom function.
    d. Trained model and saved the best model to local disk. The best model has 64x1 sequential layers, 'sigmoid' activation function for both hidden layer and output layer, epoch = 20, dropout = 0.2, with other parameters default. NN model appears too complex for this problem and was very prone to overfit training set.
    e. Loaded the best model from local disk and verified performance in test dataset.

When comparing all the model performance between train and test set, most of algorithms have comparable performance on the primary metric 'Recall', presumably due to success on using 10-fold cross validation during training (Table 4).

Table 4 Initial Model Evaluation

Algorithm	Accuracy		Precision		Recall		ROC AUC	
	Train	Test	Train	Test	Train	Test	Train	Test
Gaussian Naïve Bayes	0.7176	0.7586	0.7388	0.2714	0.6730	0.6789	0.7176	0.7238
Decision Tree	0.7483	0.8383	0.8234	0.3692	0.6321	0.6142	0.7483	0.7405
Logistic Regression	0.7165	0.7348	0.7227	0.2552	0.7026	0.7058	0.7165	0.7221
Random Forest	0.7380	0.8055	0.7815	0.3217	0.6606	0.6552	0.7380	0.7399
Neural Network	0.7401	0.5691	0.7408	0.1818	0.7398	0.8071	0.7401	0.6730

Model Performance Evaluation

When evaluating model performance using ROC curve and Cumulative Gain analysis, original imbalanced training set was used with test set to reflect the low subscribe rate in natural setting.

1) Loaded all the trained models from local disk.
2) Plotted ROC curves for training set and test set by using ROC plotting function (Figure 5). While most of the models have similar area under curve between training and test sets, neural network model overfitted slightly on training set. Overall all models perform very similarly.

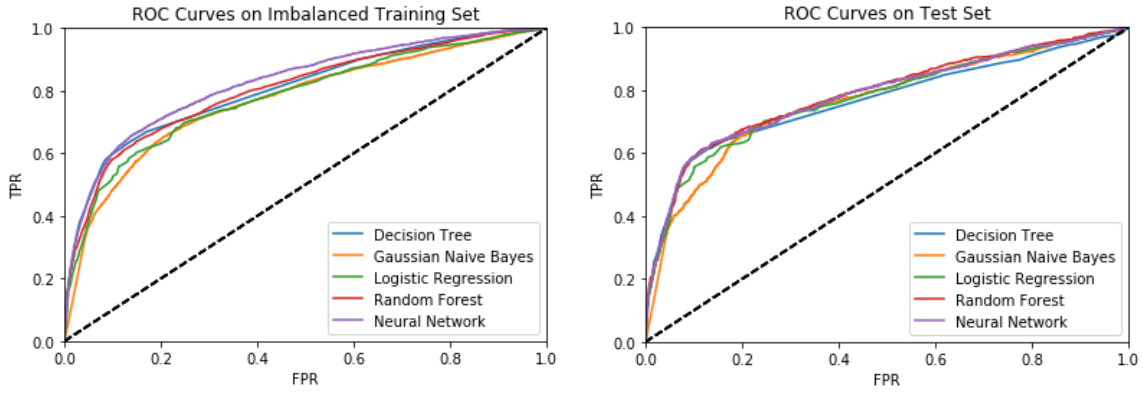

Figure 5 Model Evaluation: ROC Curves on Imbalanced Training and Test Set

3) Plotted Cumulative Gain curves on train dataset, original imbalanced train dataset and test dataset by using Cumulative Gain plotting function (Figure 6). Decision tree model showed larger subscribe% gain as indicated by area under curve than other models. Neural network overfitted slightly on training data.

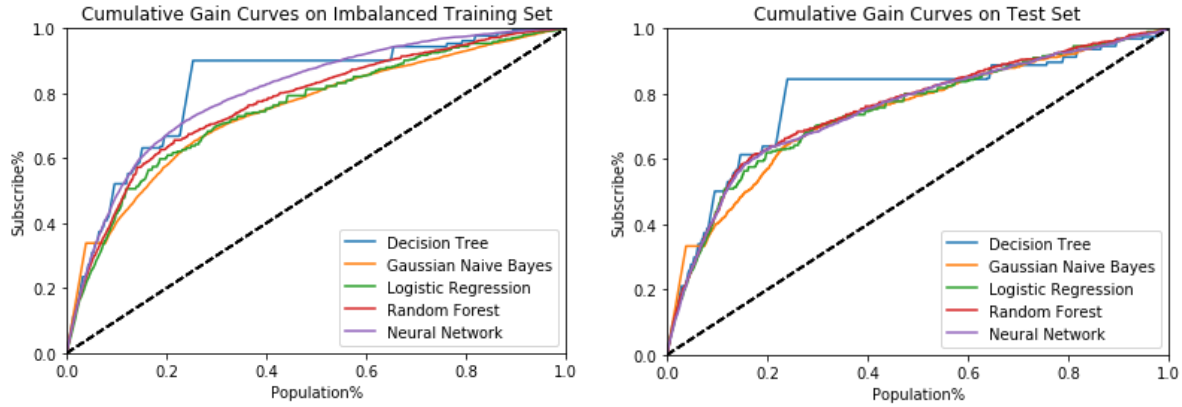

Figure 6 Cumulative Gain Curves on Imbalanced Training and Test Set

4) Plotted Lift curves for all the models on imbalanced training set and test set by using Lift plotting function (Figure 7). For every model, the general trend is that Lift will descend from high to low. However, for decision tree, there are some steep spikes in the areas of low population %. This indicates the quick increase of Lift within small population and correlates the elevation of subscribe% for decision tree model shown in Cumulative Gain plot (Figure 6).

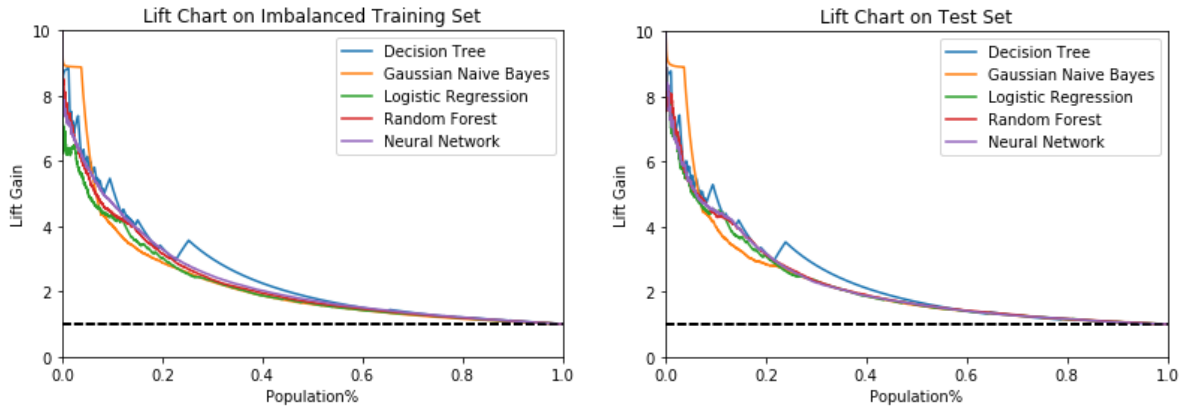

Figure 7 Lift Chart on Imbalanced Training and Test Set

Model Improvement and Evaluation

1) Used grid search on train dataset with expanded model parameter optimization and resulted in 2 improved decision tree models.
2) Plotted ROC and Cumulative Gain curves with original and improved decision tree models as well as benchmark model Gaussian Naïve Bayes. The best decision tree model was determined primarily by Cumulative Gain analysis.

Confusion Matrix Analysis

1) Assumed telemarketing effort was targeted to 20% of total clients.
2) Defined prediction probability threshold and made predictions on who should be contacted.
3) Performed Confusion Matrix plotting to compare subscribe rate among different models. The confusion matrix plotting function was downloaded from scikit-learn documentation and imported as an external function[7].

Refinement

After initial models were created and compared, decision tree was determined to be the best algorithms. Further fine tuning of decision tree model parameters was performed by expanding the grid search parameter pool from 3 to 8 and 10-fold cross validation enabled (Table 5). By looking at the 'recall' metric reported from test dataset, improvement model was likely to be the best decision model to pursue. The result was further confirmed by Cumulative Gain analysis (Table 7 and Figure 9).

Table 5 Model Parameters Optimization on Decision Tree

Algorithm	Original	Improvement (Final)	Improvement II
Criterion	Gini	Gini	Gini
Splitter	Best	Random	Random
Max_depth	7	6	5
Min_samples_leaf	2	2	3
Min_samples_split	2	2	2
Max_features	None	Log2	Log2
Class_weight	None	None	None
Presort	False	True	True
Others	Default	Default	Default

Results

Model Evaluation and Validation

The final decision tree model was firstly evaluated on test dataset with accuracy, precision, recall and ROC AUC (Table 6). Although it has very low accuracy and precision, the final model meets optimization target for the highest Recall. Apparently the model sacrificed precision in order to achieve high Recall and its performance needs to be confirmed further to see whether it is worth the sacrifice.

Table 6 Accuracy, Precision, Recall, ROC AUC Analysis on Test Set

Algorithm	Original	Improvement (Final)	Improvement II
Accuracy	0.8383	0.6653	0.6681
Precision	0.3692	0.2127	0.2122
Recall	0.6142	0.7295	0.7177
ROC AUC	0.7405	0.6934	0.6898

ROC curves on imbalanced training set and test set were created to compare decision tree models to benchmark model, Gaussian Naïve Bayes (Figure 8, Improvement II model was not shown since it basically overlaps with final model). Consistent with ROC AUC values, final model actually performs the worst. Therefore, by the standard of overall prediction capability, final model is not the best even when compared to benchmark model.

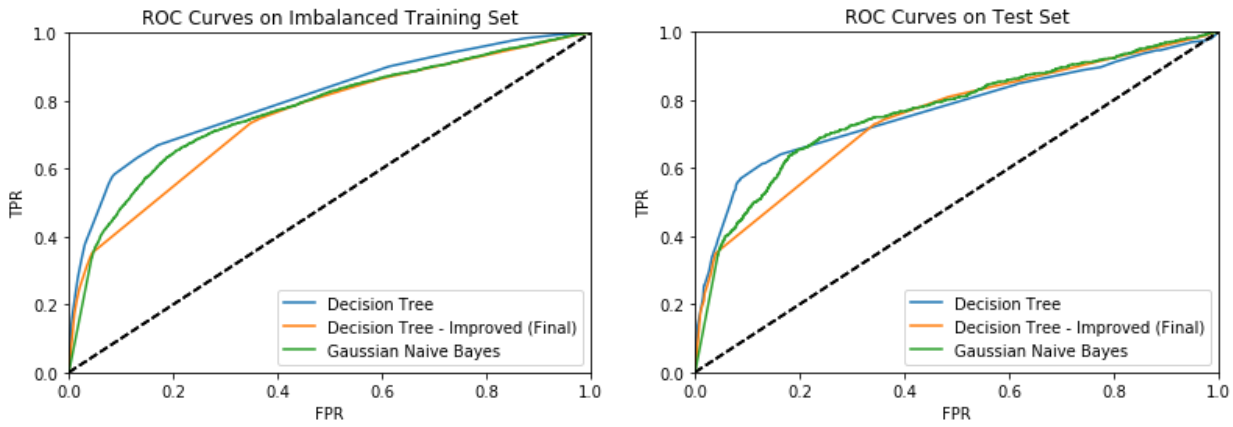

Figure 8 ROC Curves on Imbalanced Training and Test Sets - Decison Tree vs Benchmark

However, when evaluating performance on Cumulative Gain in both imbalanced training set and test set (Table 7), final model not only achieved the highest area under curve score (AUC) but also performed better on predicting higher probability value for true subscribers, as evidenced by the long steep curve at the region of 0 – 20% client population (high prediction probability area). It can predict the highest percentage of clients who actually subscribe (>70%) in the top <20% clients who are predicted to subscribe (Figure 9). This means that telemarketing effort only needs to target the top <20% clients to gain >70% of return. This effect is not affected by the manipulation of dataset and does not indicate overfitting because even though the model was trained on the upsampled training set, both original imbalanced training set and test set performed similarly.

Table 7 Area Under Curve of Cumulative Gain Curves - Decision Tree v.s. Benchmark

Model	Original		Improvement (Final)		Improvement II		Naïve Bayes	
Dataset	Train	Test	Train	Test	Train	Test	Train	Test
AUC score	0.8238	0.7847	0.7990	0.7958	0.7860	0.7833	0.7468	0.7422

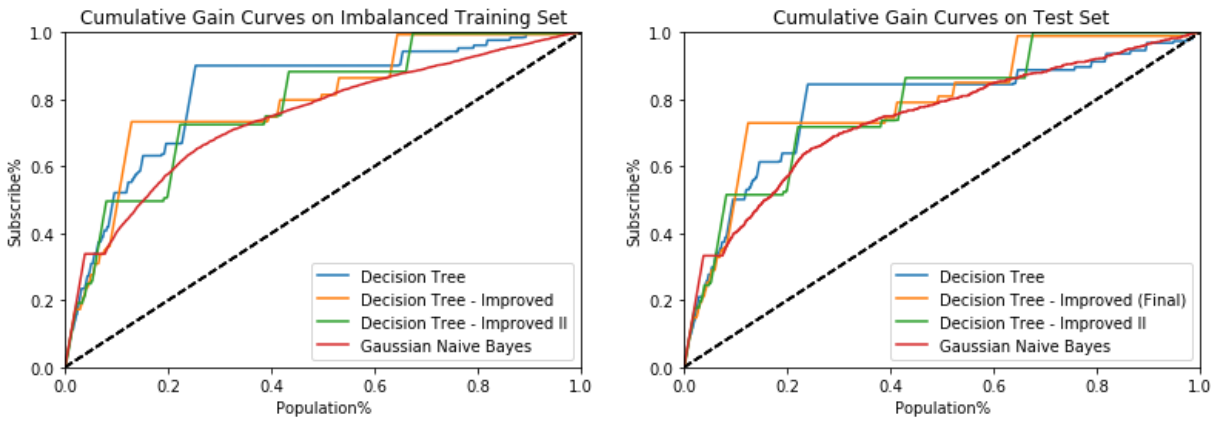

Figure 9 Cumulative Gain Analysis on Imbalanced Training and Test Set - Decision Tree v.s. Benchmark

Justification

To further visualize the benefit of final model, a marketing simulation was carried out by setting budget for telemarketing effort (contact 20% of clients) and calculate subscribe rate. A function was defined to perform this simulation, which basically retrieves probability threshold based on top 20% cutoff then uses this threshold to classify each client. Confusion Matrix analysis indicates that with only 20% clients contacted, final model can identify 73% of subscriber while benchmark model can only identify 57% (Figure 10). This is clearly aligned with business expectation: use minimal effort and budget to maximize marketing return.

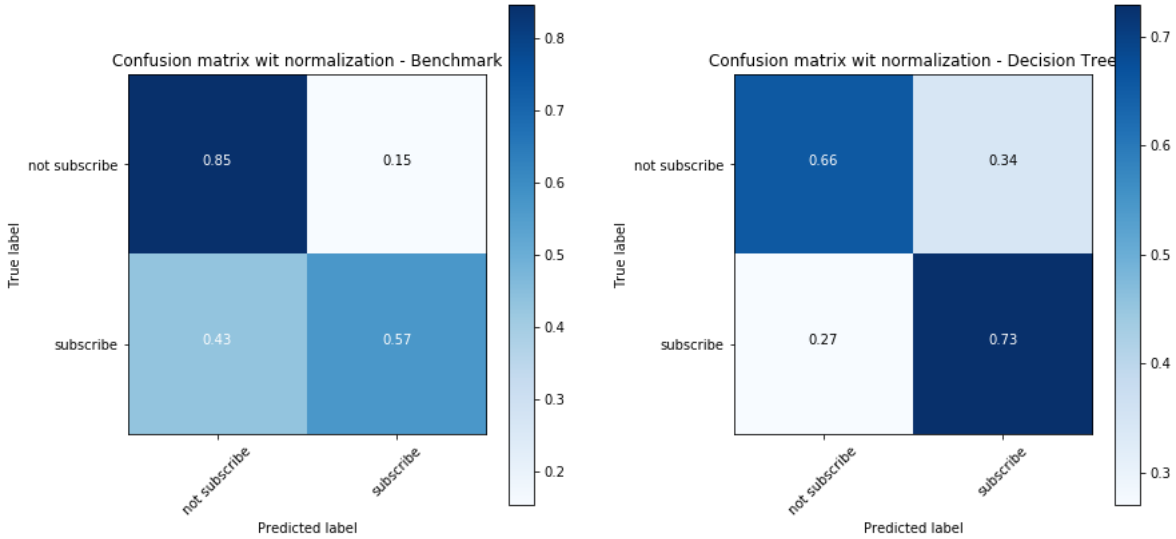

Figure 10 Confusion Matrix on Test Set - Decision Tree v.s. Benchmark

Conclusion

Free-Form Visualization

A decision tree model was developed to predict clients who will subscribe term deposit after telemarketing. This model was shown to be able to identify more clients who will subscribe than benchmark model within only a small proportion of total clients contacted, as demonstrated by the Cumulative Gain curves and Area Under Curve (AUC) in Figure 11.

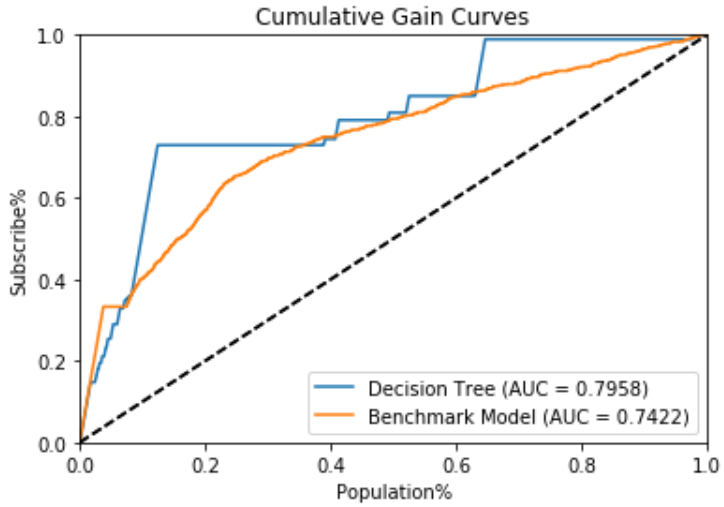

Figure 11 Cumulative Gain Graph – Decision Tree v.s. Benchmark

A detailed quantitative comparison of Cumulative Gain was shown in Table 8. In the first 30% clients, decision tree model clearly predicts more clients who actually subscribe term deposit than benchmark model and a previously published Neural Network model developed in R^2. In practice, bank only needs to contact as few as 10% of clients to get 51.5% of subscribers and 20% of clients to get 73% of subscribers. In contrast, benchmark model needs to contact ~20% and ~40% respectively while NN R model needs to contact ~30% and ~50% respectively. That is more than 50% saving on telemarketing efforts. The efficiency of decision tree model decreases to similar level as benchmark and NN R model when more clients are contacted (>40-50%).

Table 8 Client Subscribe Rate Identified by Predictive Models

Clients Contacted	Decision Tree	Benchmark	NN R Model[2]
10%	51.5%	40.1%	19.2%
20%	73.0%	57.0%	35.6%
30%	73.0%	69.6%	51.3%
40%	74.5%	75.0%	67.7%
50%	80.9%	79.4%	78.9%
60%	85.0%	84.8%	86.4%
70%	98.9%	88.3%	91.4%
80%	98.9%	92.1%	not reported
90%	98.9%	96.6%	not reported
100%	100%	100%	100%

Analysis of feature importance revealed the top 10 features contributing to 99.4% of the final model (Figure 12). These include macro-economic environment ('emp.var.rate', 'cons.conf.idx'), contact method ('contact'), timing of phone contact ('month', 'days_of_week'), time interval from last campaign contact ('pdays', 'poutcome'), employment status ('job'), age ('age'), debt situation ('loan'). Among them, 'contact', 'days_of_week', 'pdays', 'poutcome', and 'age' were not used in the NN R model previously published[2]. These new features may contribute to the performance improvement from the models developed in this project, in addition to other algorithm implementation such as upsampling of imbalanced training set and using Recall as optimization metric. In addition, only 'emp.var.rate' contributes significantly to the final model, even though it has very strong correlation with 'euribor3m' and 'nr.employed' (Pearson's r^2 close to 1), as mentioned in the section of Exploratory Visualization. Thus, the final model is not biased towards this category of features.

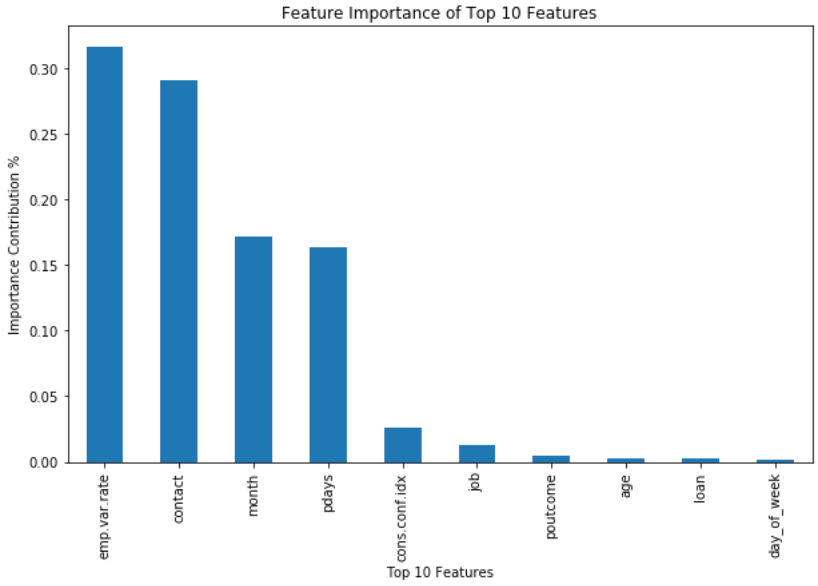

Figure 12 Feature Importance Ranking for Top 10 Features

Reflection

In this project, based on the analysis of the characteristics of data input, key data preprocessing steps such as feature selection (drop 'duration' feature), feature scaling, stratified shuffle split, and upsampling of imbalanced training set were implemented before algorithm training. Several classifiers were selected based on data type, size of feature and dataset. Recall instead of accuracy was used as training metric after evaluating the nature of business application (telemarketing). The performance of each model was evaluated and validated in original imbalanced training set and test set by recall, ROC, and Cumulative Gain/Lift. Decision tree was determined to be the best model and further optimized with expanded model parameters. Finally, a marketing simulation was performed to demonstrate that final model can identify the highest percentage of clients who subscribe term deposit with only a small percentage of clients contacted by phone calls.

An important concept for predictive modeling learned from this project is that model evaluation needs to carefully consider the context of business problems and practical solutions. That understanding led to using Recall as model training metric. Another critical aspect of this project is to understand Cumulative Gain and Lift concepts and develop functions to calculate and visualize them. As the result, the end solution in this project was measured and confirmed by relevant business metric. Final model developed in this project should be an enhancement to the solution provided by Moro et al.[2] in 2 important ways:

1) Firstly, the solution out of this project is a more realistic predictive model since it does not rely on 'duration' feature which is not practical to know before telemarketing campaign starts.
2) Secondly, decision tree model from this project is more powerful. In contrast to the Lift of <20% with 10% clients for all the models described by Moro et al.[2], 52% of Lift is demonstrated.

Improvement

Based on the analysis on Cumulative Gain and Lift, the final selected model, Decision Tree, does better on predicting higher probability for clients who likely subscribe term deposit than other models. This essentially suggests that business application discussed here, telemarketing, may benefit greater from a model enabling wider separation boundary between binary classes. While Support Vector Machine is suited for this purpose, due to the training complexity of SVM for this relatively large dataset, SVM algorithm training and optimization were not continued in this project (initial attempt was stopped due to long computation waiting hours). However, it will be interesting to see whether SVM model can make future improvement.

Recall was used as the target metric for model training in this project. However, it only serves as a surrogate metric for initial model screening. The ultimate metrics used for measuring model performance are Cumulative Gain and Lift. If a custom metric based on Cumulative Gain and Lift is defined for model parameter grid search and cross validation, it is possible to achieve better models.

References

[1] https://www.digitaldoughnut.com/articles/2016/july/digital-marketing-vs-traditional-marketing

[2] http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&cad=rja&uact=8&ved=0ahUKEwjQzZn-8IfWAhVFRiYKHb8fD1IQFgg5MAM&url=http%3A%2F%2Fmedia.salford-systems.com%2Fvideo%2Ftutorial%2F2015%2Ftargeted_marketing.pdf&usg=AFQjCNFNRnh5tcG6mYfNR1MneIb7YDct9A

[3] https://archive.ics.uci.edu/ml/datasets/Bank+Marketing#

[4] https://www.analyticsvidhya.com/blog/2016/02/7-important-model-evaluation-error-metrics/

[5] https://elitedatascience.com/imbalanced-classes

[6] https://github.com/fchollet/keras/commit/a56b1a55182acf061b1eb2e2c86b48193a0e88f7

[7] http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html