Portuguese Bank Marketing Classification Report

Machine Learning Engineer Nanodegree

Shen Bing
September 29th, 2016

I. Definition

Project Overview

Telemarketing is widely used in bank marketing. Banks offer credit cards, loans and financial products through phone calls. Sometimes, these telemarketing campaigns randomly making phone calls to clients, resulting in a very low success rate [1]. Machining learning algorithms enable us to create client segments so that telemarketing campaigns can be more efficient by targeting clients more likely to subscribe products [2]. By deciding client segments, classification algorithms would be applied. There are numbers of classification algorithms such as Logistic Regression, SVM, Decision Tree and etc, all of which have their advantages and disadvantages.

In this project, the goal is to build a data-driven model using input variables to decide customers' labels (campaign success or failure). The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required [3]. I used "bank-additional-full.csv", one of the four datasets from Bank Marketing Data Set of UCI's machine learning repository. It has 41188 examples and 20 inputs, ordered by date [2] (from May 2008 to November 2010). The clients have labels indicating subscribing or not subscribing bank term deposit.

Problem Statement

This project is a supervised binary classification problem. All clients can be classified as subscribing bank term deposit or not subscribing. The goal is to use the given features, build a proper model and decide which category a client belongs to.

By exploring the dataset, basic information about the dataset can be obtained. Then the dataset should be prepared by converting non-numeric binary variables into binary variables and categorical variables into dummy variables. For models like SVM and Logistic Regression, data needs to be standardized. After that, the full dataset should be split into training set and testing set. Several models should be applied on the training set. After the models are trained, the performance is evaluated on the testing set. By comparing metrics of each model, Gridsearch should be applied in order to tune parameters. After models are tuned, again, metrics need to be

compared. Finally I will validate the performance of the classifiers using tuned parameters on training and testing set of different sizes.

Metrics

Since, it is a binary classification problem; I will use ROC curve (receiver operating characteristic curve) and PR curve (Precision-Recall curve) to evaluate models. Precision-recall is typically going to be more useful when negative samples dwarf the number of positives. Although ROC tends to be an easier explanation for evaluating models, precision-recall reveals more information when dealing with imbalanced data sets [4]. When curves are so close to each other, AUC (Area under the curve) and F1 score are introduced. Besides, training and testing time is also an important metric.

(1) ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP+TN}}$$

A perfect classification is achieved at (0, 1) point. (1, 0) point represents the worst classifier. A random guess would give a point along a diagonal line [5]. When the curve is close to the upper left corner, it means that the model perform well.

(2) AUC is the size of the area under the ROC curve. When using normalized units, AUC is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming "positive" ranks higher than "negative") [5]. The larger AUC is, the better a model performs.

(3) PR curve is created by plotting the precision against the recall. When the curve is close to the upper right corner, it means that the model perform well [6].

$$\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \qquad \mathrm{recall} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}$$

(4) F-score is the harmonic mean of precision and recall. It's defined as follows [8]:

$$\mathrm{F1} = \frac{2 * \mathrm{recall} * \mathrm{precision}}{\mathrm{recall} + \mathrm{precision}}$$

(5) Time to train the model and make predictions is also taken into consideration.

II. Analysis

Data Exploration

The dataset is downloaded from UCI's machine learning repository. There are 21 total features and 41188 clients, which are ordered by date. It is an imbalanced dataset with 36548 clients not subscribed a term deposit, while 4640 clients subscribed a term deposit [3].

As is shown in Table 1, feature are "age", "job", "marital", "education", "default", "housing", "loan",

“contact”, “month”, “day_of_week”, “duration”, “campaign”, “pdays”, “previous”, “poutcome”, “emp.var.rate”, “cons.price.idx”, “cons.conf.idx”, “euribor3m”, “nr.employed” and “y”, in which “age”, “duration”, “campaign”, “pdays”, “previous”, “emp.var.rate”, “cons.price.idx”, “cons.conf.idx”, “euribor3m” and “nr.employed” are continuous variables. Others are discrete variables (categorical variables and non-numeric binary variables). The first 20 features are input variables. The last feature “y” is an output variable showing whether a client subscribing bank term deposit or not.

Classificaion	Feature names
Input(continuous)	age, duration, campaign, pdays, previous, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed
Input(discrete)	job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome
output	y

Table 1 Classification for all features

There are more details about variables [2]:

Input variables:

Bank client data:

(1) age (numeric)
(2) job : type of job (categorical: "admin.", "blue-collar", "entrepreneur", "housemaid", "management", "retired", "self-employed", "services", "student", "technician", "unemployed", "unknown")
(3) marital : marital status (categorical: "divorced", "married", "single", "unknown"; note: "divorced" means divorced or widowed)
(4) education (categorical: "basic.4y", "basic.6y", "basic.9y", "high.school", "illiterate", "professional.course", "university.degree", "unknown")
(5) default: has credit in default? (categorical: "no", "yes", "unknown")
(6) housing: has housing loan? (categorical: "no"," yes", "unknown")
(7) loan: has personal loan? (categorical: "no", "yes", "unknown")

Related with the last contact of the current campaign:

(8) contact: contact communication type (categorical: "cellular", "telephone")
(9) month: last contact month of year (categorical: "jan", "feb", "mar", ..., "nov", "dec")
(10) day_of_week: last contact day of the week (categorical: "mon", "tue", "wed", "thu", "fri")
(11) duration: last contact duration, in seconds (numeric).

Other attributes:

(12) campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
(13) pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
(14) previous: number of contacts performed before this campaign and for this client (numeric)
(15) poutcome: outcome of the previous marketing campaign (categorical: "failure",

"nonexistent", "success")

Social and economic context attributes
(16) emp.var.rate: employment variation rate - quarterly indicator (numeric)
(17) cons.price.idx: consumer price index - monthly indicator (numeric)
(18) cons.conf.idx: consumer confidence index - monthly indicator (numeric)
(19) euribor3m: euribor 3 month rate - daily indicator (numeric)
(20) nr.employed: number of employees - quarterly indicator (numeric)

Output variable (desired target):
(21) y: has the client subscribed a term deposit? (binary: "yes", "no")

Table 2 gives some examples of the dataset. For some categorical attributes, there are several missing values. 10700 samples have one or more missing values. However, there is no need to delete these samples; instead these missing values can be coded with the "unknown" label which can be treated as a possible class label.

Discrete values such as housing and loan, are non-numeric binary variables should be converted to binary variables like 0 or 1. Categorical variables like job should be converted into dummy variables.

	Sample 0	Sample 1	Sample 2	Sample 3	Sample 4
age	56	57	37	40	56
job	housemaid	services	services	admin.	services
marital	married	married	married	married	married
education	basic.4y	high.school	high.school	basic.6y	high.school
default	no	unknown	no	no	no
housing	no	no	yes	no	no
loan	no	no	no	no	yes
contact	telephone	telephone	telephone	telephone	telephone
month	may	may	may	may	may
day_of_week	mon	mon	mon	mon	mon
duration	261	149	226	151	307
campaign	1	1	1	1	1
pdays	999	999	999	999	999
previous	0	0	0	0	0
poutcome	nonexistent	nonexistent	nonexistent	nonexistent	nonexistent
emp.var.rate	1.1	1.1	1.1	1.1	1.1
cons.price.idx	93.994	93.994	93.994	93.994	93.994
cons.conf.idx	-36.4	-36.4	-36.4	-36.4	-36.4
euribor3m	4.857	4.857	4.857	4.857	4.857
nr.employed	5191	5191	5191	5191	5191
y	no	no	no	no	no

Table 2 Five samples of the dataset

Table 3 reveals that for continuous features they are not in the same scale. So standardization is needed when applying models like SVM and Logistic Regression.

	count	mean	std	max	min	25%	50%	75%
age	41188	40.02	10.42	98.00	17.00	32.00	38.00	47.00
duration	41188	258.29	259.28	4918.00	0.00	102.00	180.00	319.00
campaign	41188	2.57	2.77	56.00	1.00	1.00	2.00	3.00
pdays	41188	962.48	186.91	999.00	0.00	999.00	999.00	999.00
previous	41188	0.17	0.49	7.00	0.00	0.00	0.00	0.00
emp.var.rate	41188	0.08	1.57	1.40	-3.40	-1.80	1.10	1.40
cons.price.idx	41188	93.58	0.58	94.77	92.20	93.08	93.75	93.99
cons.conf.idx	41188	-40.50	4.63	-26.90	-50.80	-42.70	-41.80	-36.40
euribor3m	41188	3.62	1.73	5.05	0.63	1.34	4.86	4.96
nr.employed	41188	5167.0	72.25	5228.1	4963.6	5191.0	5191.0	5228.1

Table 3 Description of continuous features

According to a previous study [3], these 21 features are selected from the original 150 features. Using business intuitive knowledge, it has already ruled out a group of related features. Besides, other highly related features are ruled out by an automated selection approach. So I decided not to rule out any features.

Exploratory Visualization

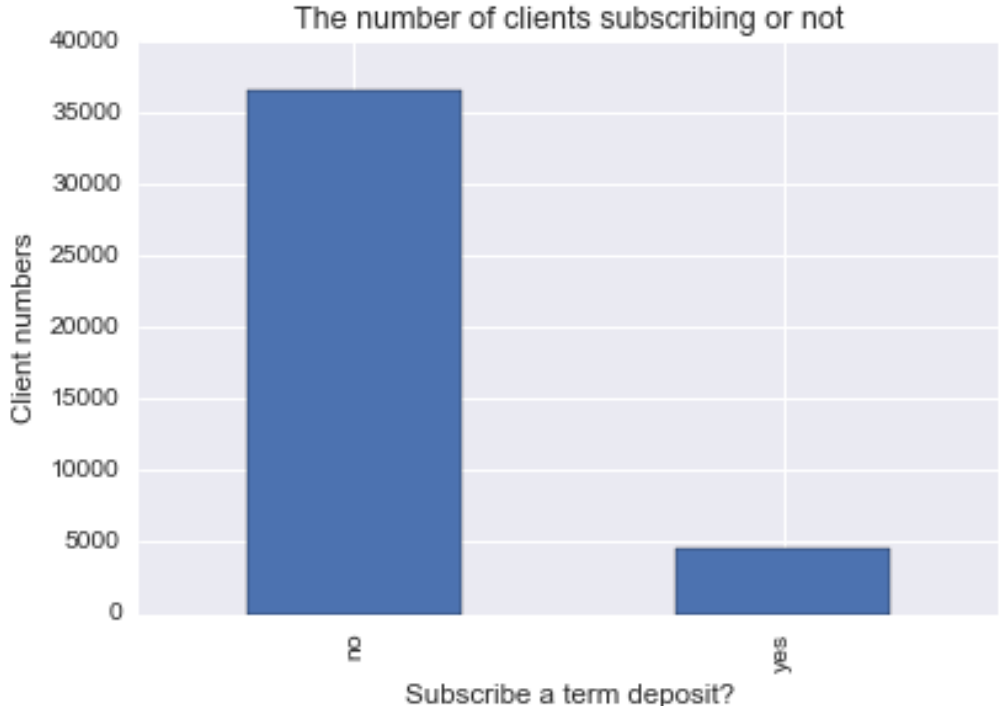

Figure 1 The number of clients subscribing a term deposit or not

As it is shown in Fig. 1, 36548 clients did not subscribed a term deposit, while 4640 clients subscribed a term deposit. Only 11.27% of the clients said yes to the campaign. It indicted that the dataset is imbalanced.

I used StratifiedShuffleSplit from sklearn.cross_validation to split the dataset. The training set has 30479 samples, in which 27045 samples are no and 3434 samples are yes. The testing set has 10709 samples, in which 9503 samples are no and 1206 samples are yes.

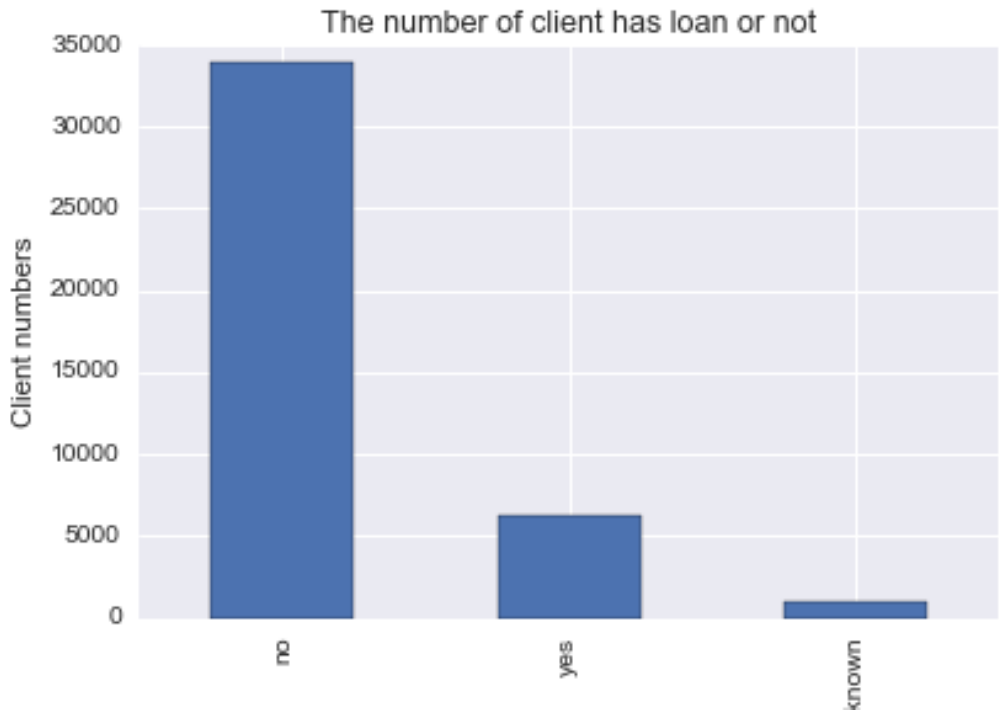

Figure 2 The number of clients have loan or not

Fig 2 shows that apart from 990 clients' status are unknown, 33950 clients have no loan and 6248 clients have personal loan. 82% of clients have no personal loan, while 15% of clients have personal loan.

Fig 3 shows that most of samples are from 24 years old to 60 years old. According to Tukey's Method for identifying outliers, samples over the age of 70 might be outliers. But in fact it is a sign of skewed distribution [7]. It is a positive skew that the tail on the right side is longer than the left side and the mass of the distribution is concentrated on the left of the figure. The mean is 40.02, while median is 38. So I used log transformation to make the feature obey a normal distribution.

Fig 4 shows that duration presents characteristics of skewed distribution. Most of the samples are under 1000 seconds and they are concentrated around 180 seconds. The long tail extends to the right. The mean is 258.29, while the median is 180. The log transformation is applied to make the feature obey a normal distribution.

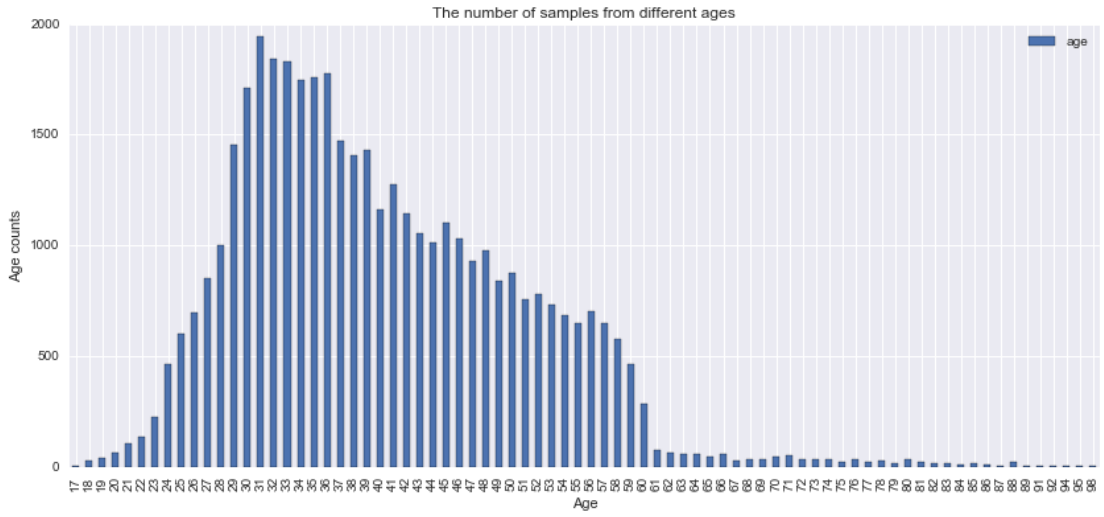

Figure 3 The number of samples from different ages

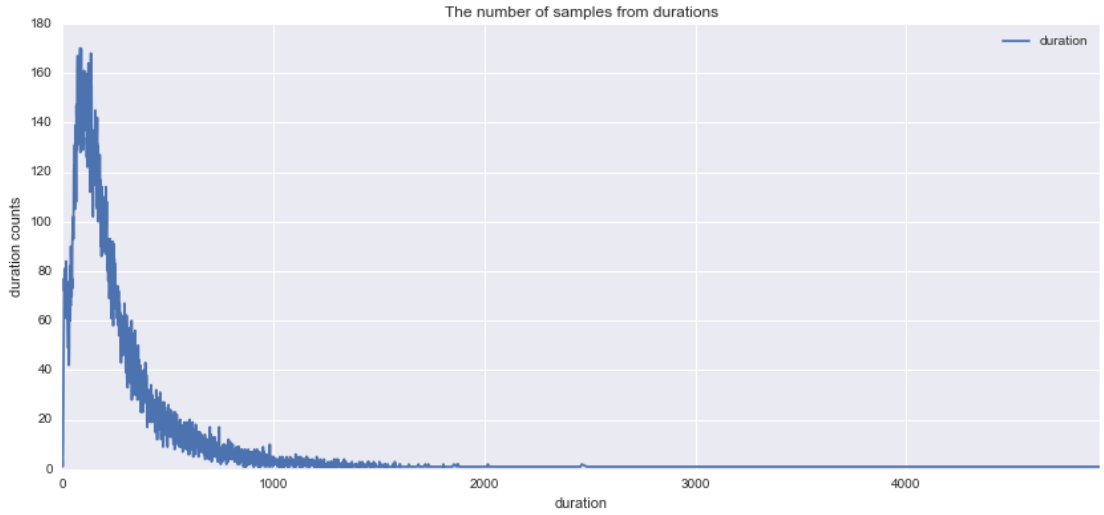

Figure 4 The number of samples from different durations

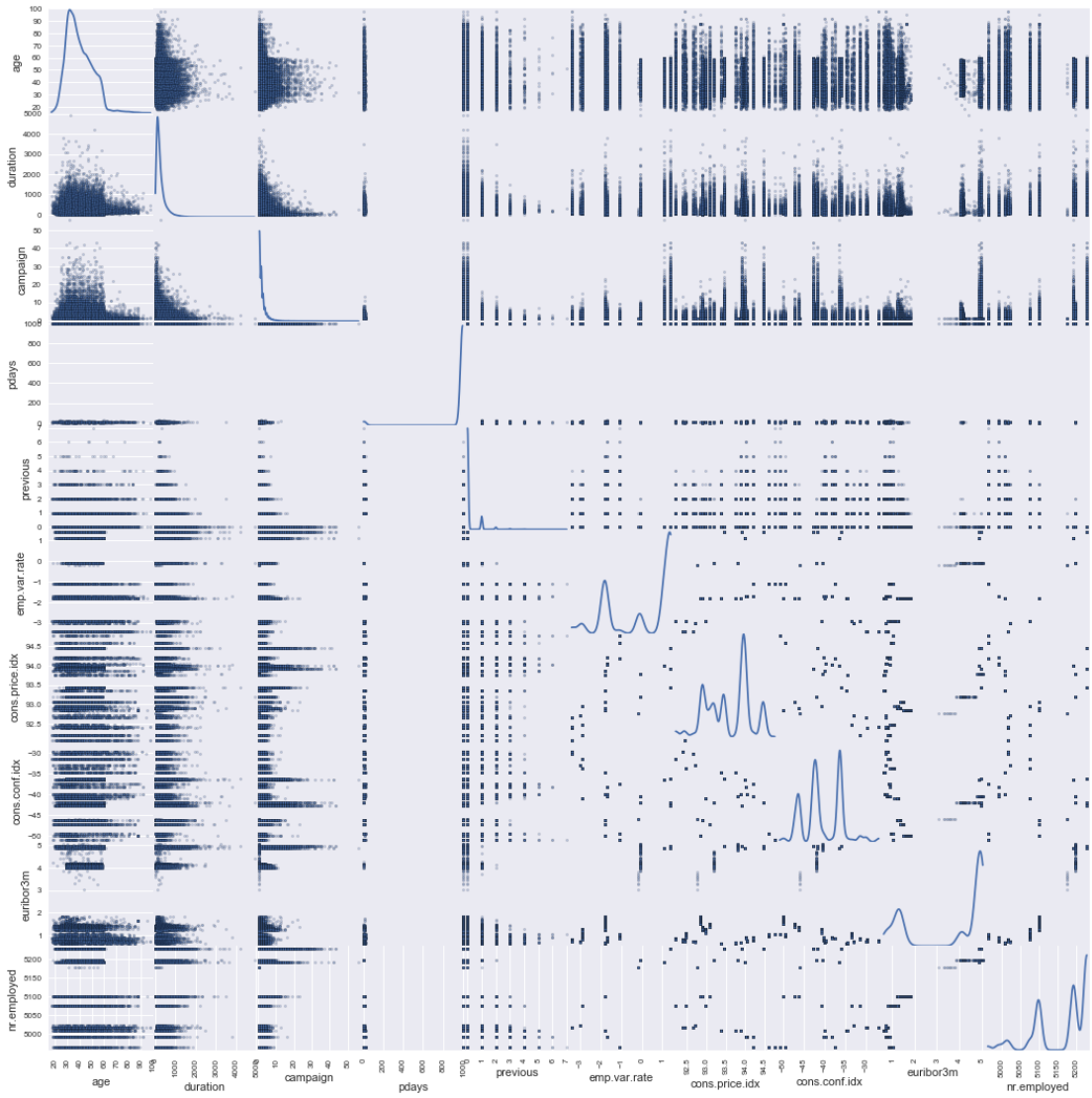

Figure 5 a scatter matrix for each pair of continuous features in the data

According to the scatter matrix, it seems that none of the continuous features has linear relationship between two features.

Algorithms and Techniques

I used Python scikit-learn package to do my work. The dataset should be preprocessed by converting non-numeric binary variables into binary variables and categorical variables into dummy variables. Since skewness was found of some numerical features, the log transformation was applied. For SVM and Logistic Regression, I used StandardScaler to standardize data. Since it is an imbalanced dataset, I used StratifiedShuffleSplit to split the data into training set and testing set. Since the training set is imbalanced, I also introduced a third party package imblearn.under_sampling to do under sampling for Random Forest Classifier. Because it is a binary classification problem, I will use SVM, Logisitc Regression and Random Forest Classifier models. After choosing one final model, I will use gridsearch to find best parameters.

(1) SVM [9]

Pros: It is a novel learning method with a solid theoretical basis. SVM is effective in high dimensional spaces, which the number of samples is small. It can also work on non-linear problem and generalize well. It works well in complicated domains where there is a clear margin separation.

Cons: The computational complexity depends on the number of support vectors so when the data set is large, it will take much longer time to compute. If there was too much noise in the data set, SVM wouldn't work well.

(2) Logistic Regression [10]

Pros: there is no need to make prior assumptions so that it can avoid the problem of inaccurate assumptions. I can get the approximate probability prediction as well as predictions of categories.

Cons: When the feature space is very large, it will not perform well. It is easy to be under fitted and can't handle a large number of features. It can only deal with the two classification problem, and for nonlinear features, there is a need for conversion.

(3) Random Forest Classifier [11]

Pros: for imbalanced classification data sets, it can balance error. It has high prediction accuracy, and is not easy to be over fitting. It is also not sensitive to noise, with good noise tolerance ability.

Cons: It is not always works well on small and low dimension data sets. Execution speed is much slower than Decision Tree.

Benchmark

In an article Sergio Moro et al. published in 2014, they used Neural Network model to achieve the best AUC that is around 0.8. A random guess will give an AUC of 0.5. So my project will try to find AUC higher than 0.8. Besides, for imbalanced dataset, PR curve is better than ROC to show the performance of a classifier. So a model with a higher F1 score is better describe the dataset.

III. Methodology

Data Preprocessing

Firstly, there are several missing values in some categorical attributes, all coded with the "unknown" label. There is no need to delete these samples; instead these missing values can be treated as a possible class label. The dataset contains non-numeric binary variables and categorical variables, which cannot be used directly. Then I used replace and get dummies to turn these variables into dummy variables. Since skewness is found of several numerical features, the log transformation is applied to make these features obey a normal distribution. For some variables are in different scales, when applying models like SVM and Logistic Regression, the data

needs to be standardized. I used StandardScaler to standardize data. The dataset was split by StratifiedShuffleSplit from sklearn.cross_validation. Since the training set is imbalanced, I also introduced a third party package imblearn.under_sampling to do under sampling for Random Forest Classifier.

Implementation

I used SVM, Logistic Regression and Random Forest Classifier to train the data, respectively.

(1) For SVM, svm was imported from sklearn. Then the model was trained on train set and all parameters are default, i.e. C=1, kernel='rbf', gamma='auto', class_weight=None.

(2) For Logistic Regression, I imported LogisticRegression from sklearn.linear_model and parameters were default, i.e. penalty='l2', dual=False, C=1.0, class_weight=None, solver='liblinear'

(3) For Random Forest Classifier, I firstly imported RandomUnderSampler from a third party package called imblearn.under_sampling, which is used for under sampling the training data to make the data set more balanced. I imported RandomForestClassifier from sklearn.ensemble to train the model using default parameters, i.e. n_estimators=10, criterion='gini', max_depth=None.

After models were trained, they were tested on the testing set. After that, I used roc_curve, precision_recall_curve, auc and F1 score from sklearn.metrics to evaluate models. I plotted roc curves and precision recall curves for three models using matplotlib.pyplot. Also, by recording start time and end time, I could calculate how much time it takes to train and test. Finally, GridSearchCV was imported from sklearn.grid_search to find the best parameters for Logistic Regression and Random Forest Classifier. For Logistic Regression, I decided to tune "C" and "class_weight". For Random Forest Classifier, I decided to tune "n_estimators" and "max_depth". Since it takes too much time to use GridSearch for SVM, I decided only tuning "class_weight" for SVM.

At first, three models were fitted to the train set with default parameters. Then after tuning models, tuned parameters are recorded in the Table 4 below.

Models	SVM	Logistic Regression	Random Forest
Untuned parameters	C=1 kernel='rbf' gamma='auto' class_weight=None	penalty='l2' dual=False C=1.0 class_weight=None solver='liblinear'	n_estimators=10 criterion='gini' max_depth=None
Tuned parameters	class_weight={'yes':2.3}	C=1 class_weight={'yes':3.0}	n_estimators=13 max_depth=14

Table 4 Parameters for SVM, Logistic Regression and Random Forest

Table 5 shows metrics for models using default parameters. The complexity of SVM is O (n^2) -O (n^3), where n is the amount of training samples. In this data set, the training set has 30479 samples and the testing set has 10709 samples, so SVM's training and testing time are the longest. Both Logistic Regression and Random Forest have a relatively short training and testing time. The three models have almost the same AUC. AUC of SVM, Logistic Regression and Random Forest Classifier are 0.9363, 0.9424 and 0.9264.

Metric	SVM	Logistc Regression	Random Forest
Training Time	50.7900	1.4870	0.1460
Test Time	11.7230	0.0030	0.0430
AUC	0.9363	0.9424	0.9264
F1 score	0.4944	0.5412	0.5832

Table 5 Metrics for three models using default parameters

A good model should offer the best compromise between a desirable a high TPR and low FPR, that is, the curve should be near the upper left corner. The advantage of the ROC curve is that the domain user can select the best TPR and FPR trade-off that serves its needs. Fig.6 shows that the Logistic Regression ROC curve is related with the highest area (AUC) and outperforms all other methods within most all of the FPR range.

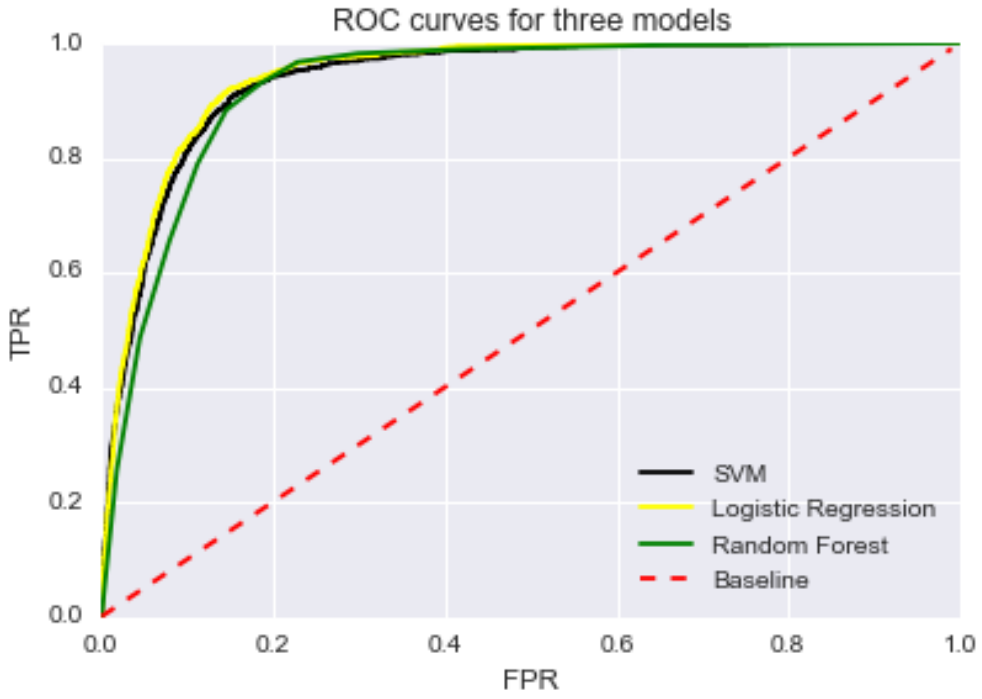

Figure 6 ROC curves for three models

A good model should have both high precision and high recall, which means most of samples are labeled correctly. When PR curve is near the upper right corner, it means the model has a good performance on this dataset. Fig.7 indicate that we consider models perform well by checking ROC curve and AUC, but PR curve shows that they did not, because our dataset is imbalanced.

Random Forest Classifier's F1 score is 0.5832, but F1 score on the training set is 0.9927, indicating the model is overfitting. For Logistic Regression model and Random Forest Classifier, F1 scores on the training set and testing set are roughly the same. So there is no overfitting on these two models. In order to make a better performance, we need to tune parameters to achieve higher F1 score.

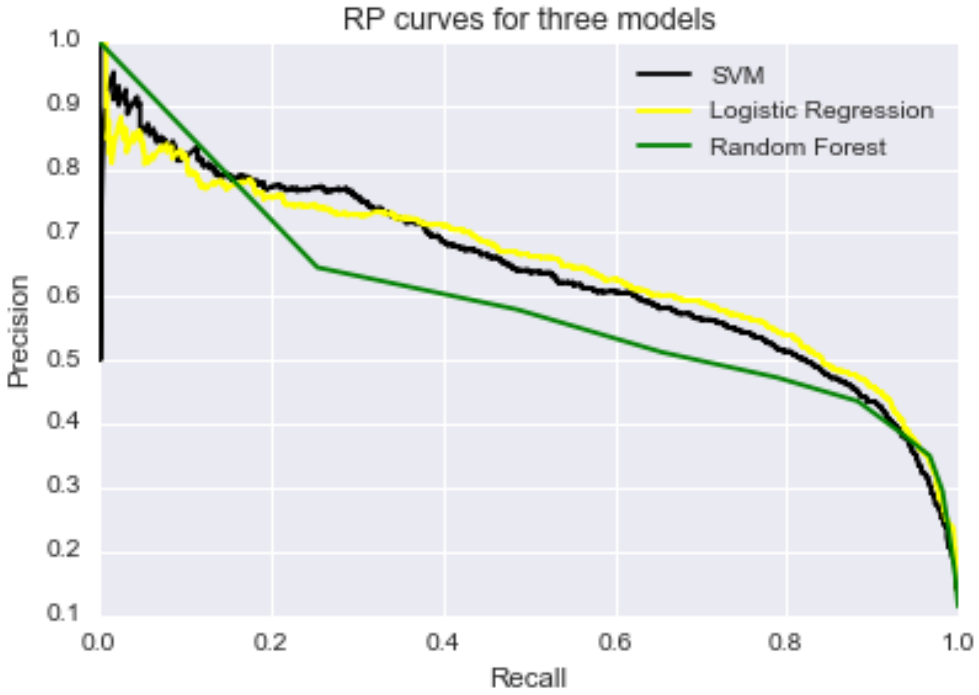

Figure 7 Precision Recall curves for three models

Then three models were fitted to the train set with tuned parameters.
Table 6 shows metrics for models using tuned parameters are as follows:

Metric	SVM	Logistc Regression	Random Forest
Training Time	53.3190	1.7330	0.2790
Test Time	11.5000	0.0030	0.0520
AUC	0.9401	0.9427	0.9352
F1 score	0.6260	0.6471	0.5665

Table 6 Metrics for three models using tuned parameters

Tuned parameter for SVM is class_weight = {'yes': 2.3}. Tuned parameters for Logistic Regression are C=1 and class_weight={'yes': 3.0}. Tuned parameters for Random Forest Classifier are max_depth=14 and n_estimators=13.

After the models are tuned, AUC hasn't changed much. AUC of SVM, Logistic Regression and Random Forest Classifier are 0.9401, 0.9427 and 0.9352. Fig. 8 also indicts that ROC curves are still the same as unturned models.

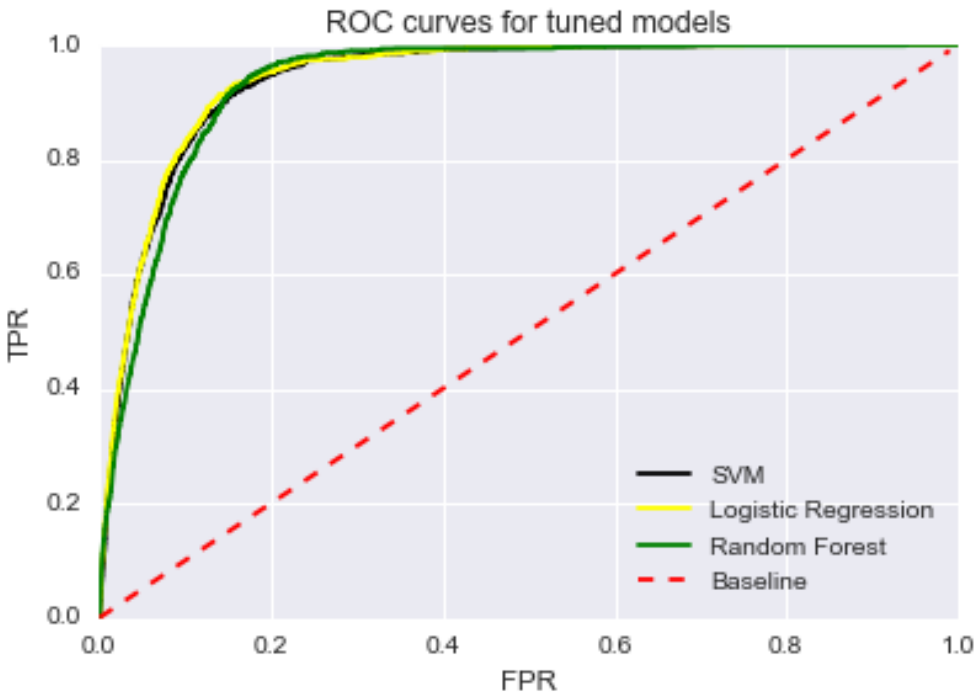

Figure 8 ROC curves for tuned models

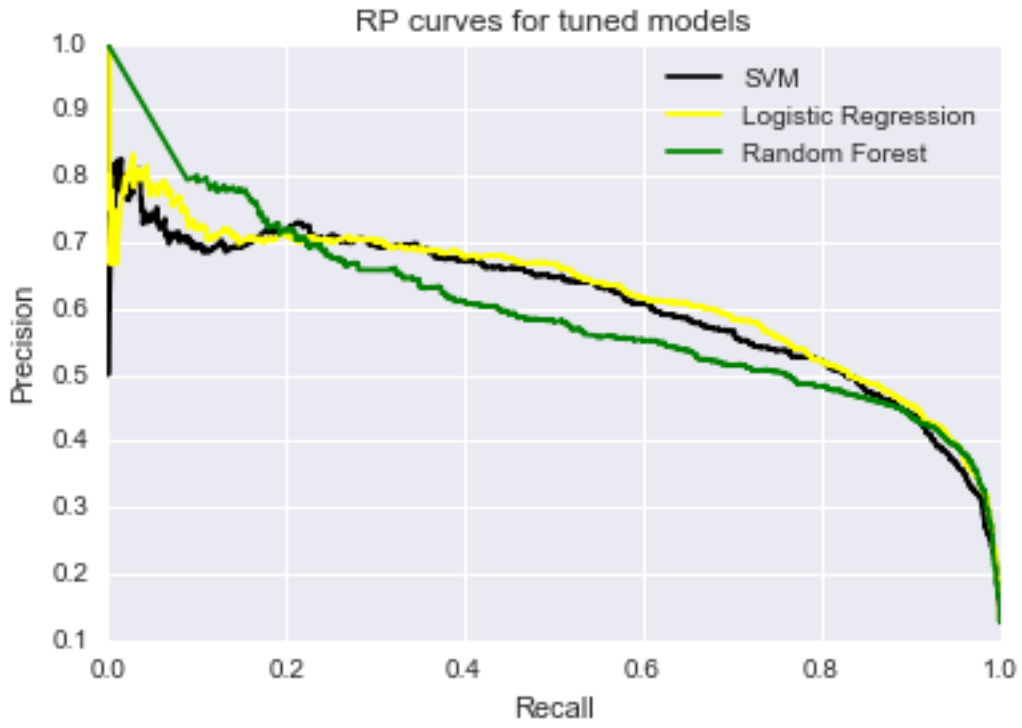

Figure 9 Precision Recall curves for three models

Refinement

AUC of SVM, Logistic Regression and Random Forest Classifier are 0.9363, 0.9424 and 0.9264.

However, F1 score of SVM, Logistic Regression and Random Forest Classifier are 0.4944, 0.5412 and 0.5832.

Metrics/Parameters	SVM	Logistic Regression	Random Forest
AUC (untuned)	0.9363	0.9424	0.9264
AUC (tuned)	0.9401	0.9427	0.9352
F1 score (untuned train)	0.5936	0.5193	0.9927
F1 score (untuned)	0.4944	0.5412	0.5832
F1 score (tuned train)	0.7442	0.6249	0.9581
F1 score (tuned test)	0.6260	0.6471	0.5665
Parameters (untuned)	class_weight=None	class_weight=None, C=1.0	n_estimators=10 max_depth=None
Parameters (tuned)	class_weight={'yes': 2.3}	class_weight={'yes': 3.0}, C=1	n_estimators=13 max_depth=14

Table 7 Comparisons of metrics and parameters for three models between untuned and tuned

Since it's time-consuming to use SVM model when the data set is large, I decided not to tune it's parameters by GridSearch. I just tried to adjusted parameters by setting class_weight to {'yes':2.3}, the F1 score increased from 0.4944 to 0.6260. I decided to use Grid Search on Logistic Regression and Random Forest Classifier. GridSearchCV was imported from sklearn.grid_search to find the best parameters for models. Logistic regression has the highest AUC of all three models, but its F1 score was not as good as Random Forest Classifier. So I tried to improve its F1 score by tuning parameters. The parameters I tried to tune are "C" and "class_weight", where C=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10) and class_weight = {'yes':1.5}, {'yes':2.0}, {'yes':2.5}, {'yes':3.0}, {'yes':3.0}, {'yes':3.5}. The best parameters are C=1 and class_weight={'yes': 3.0}, whose F1 score is 0.6471. I also tried to tune parameters for Random Forest Classifier by GridSearch. The parameters I tried to tune are 'n_estimators': (5,6,7,8,9,10,11,12,13,14), 'max_depth': (6,7,8,9,10,11,12,13,14,15). The best parameters are n_estimators = 13, max_depth = 14. The F1 score is 0.5665.

All of the models' AUCs are over 0.8. After tuning the models, AUCs are still the same. However, SVM and Logistic Regression's F1 scores are improved. Random Forest Classifier has a relatively low F1 score and it suffers from overfitting, so it won't be the final model. Comparing AUC and F1 score, SVM performs as well as Logistic Regression, but the long training and testing time makes it not as good as Logistic Regression. In conclusion, Logistic Regression is the final model with both the highest AUC and F1 score and short training and testing time.

IV. Result

Model Evaluation and Validation

Logistic Regression is the final model with both the highest AUC and F1 score and short training and testing time. The metrics indict that Logistic Regression has the best performance on the

dataset. I used 10-fold cross-validation to avoid randomness. Table 3 shows that F1 score is relatively stable under different testing sizes. So the tuned Logistic Regression model is robust and can generalize well.

Testing set	10%	15%	20%	25%	30%	35%	40%	45%	50%
F1 (train)	0.6276	0.6285	0.6288	0.6267	0.6275	0.6295	0.6225	0.6294	0.6357
F1 (test)	0.6598	0.6456	0.6305	0.6438	0.6323	0.6290	0.6377	0.6280	0.6234

Table 8 F1 score under different testing set

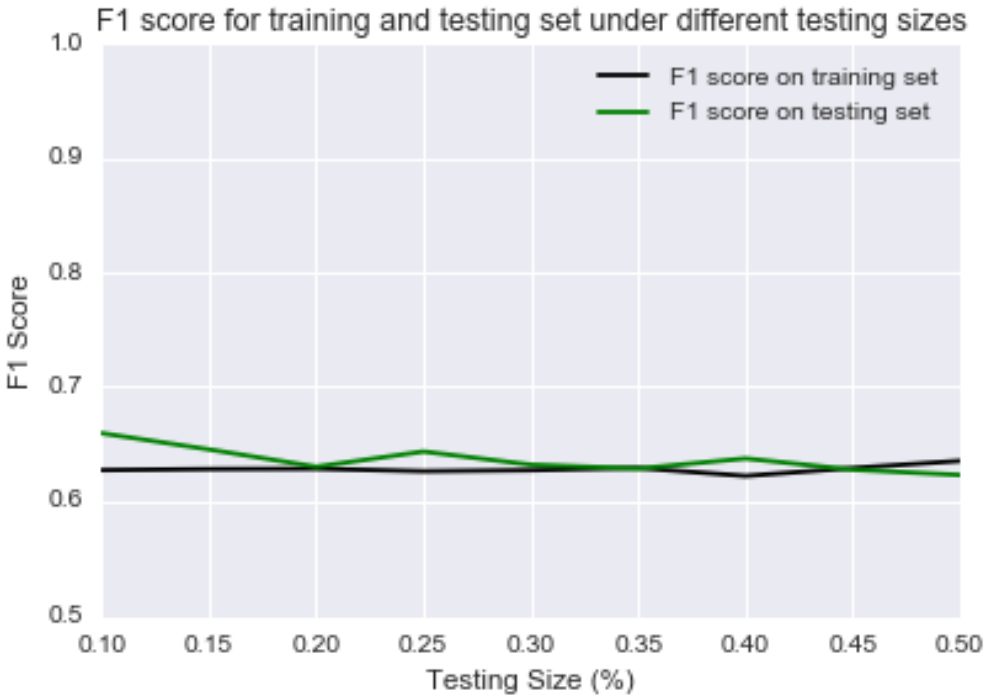

Figure 10 F1 score for training and testing set under different testing size

I also increase the parameter C of logistic regression to increasing complexities of the model. C=1, 2, 3, 4, 5, 6, 7, 8, 9. Table 9 indicates that both AUC and F1 score are relatively the same under different C values. So the tuned Logistic Regression model is robust.

C	1	2	3	4	5	6	7	8	9
AUC	0.9427	0.9427	0.9427	0.9428	0.9428	0.9428	0.9428	0.9428	0.9428
F1 score	0.6471	0.6471	0.6471	0.6471	0.6474	0.6474	0.6474	0.6474	0.6474

Table 9 F1 score and AUC using Logistic Regression under different C

Justification

All three models' AUC is higher than 0.8. After using grid search to find best parameter, Logistic Regression has the highest F1 score, 0.6471. Although SVM has nearly the same AUC and F1 score as Logistic Regression, it is very time consuming. Furthermore, Random Forest Classifier

suffers from overfitting. So Logistic Regression is the best model for this dataset.

V. Conclusion

Free-Form Visualization

For all three models, Random Forest Classifier gives us a way to understand how input features affected outputs. Fig.10 shows exhibits the top 8 relative input importance bar plot.

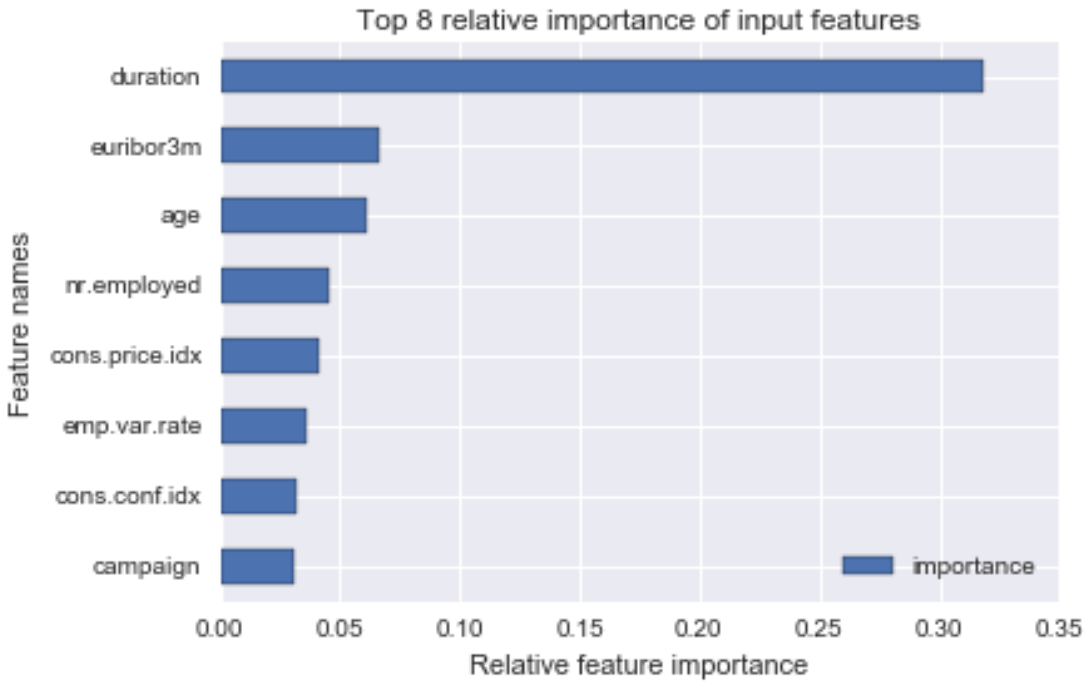

Figure 11 Top 8 relative importance of input features for Random Forest Classifier

Duration highly affects the output target, which has a relative importance of 32%. Three month Euribor rate (euribor3m), computed by the European Central Bank (ECB), a publicly available and widely used index, was considered the second most relevant attribute, with a relative importance around 7%. Next comes age is 6% and the number of employees (nr.employed) is 5%. It's interesting to find that 5 out of 8 most important features are social and economic context attributes.

I also take a look at the coefficients of Logistic Regression. Table 10 shows the features' names and coefficients of Logistic Regression.

Feature names	Positive coefficient	Features names	Negative coefficient
duration	2.0281	emp.var.rate	-2.506465
euribor3m	1.1472	month_may	-0.223905
cons.price.idx	0.8979	nr.employed	-0.223526
month_aug	0.3047	pdays	-0.182242
month_mar	0.2432	month_nov	-0.134859
contact_cellular	0.1195	contact_telephone	-0.119463
poutcome_success	0.1166	poutcome_failure	-0.09718
job_retired	0.1011	month_jun	-0.071861

Table 10 Features of top 8 positive and negative coefficien

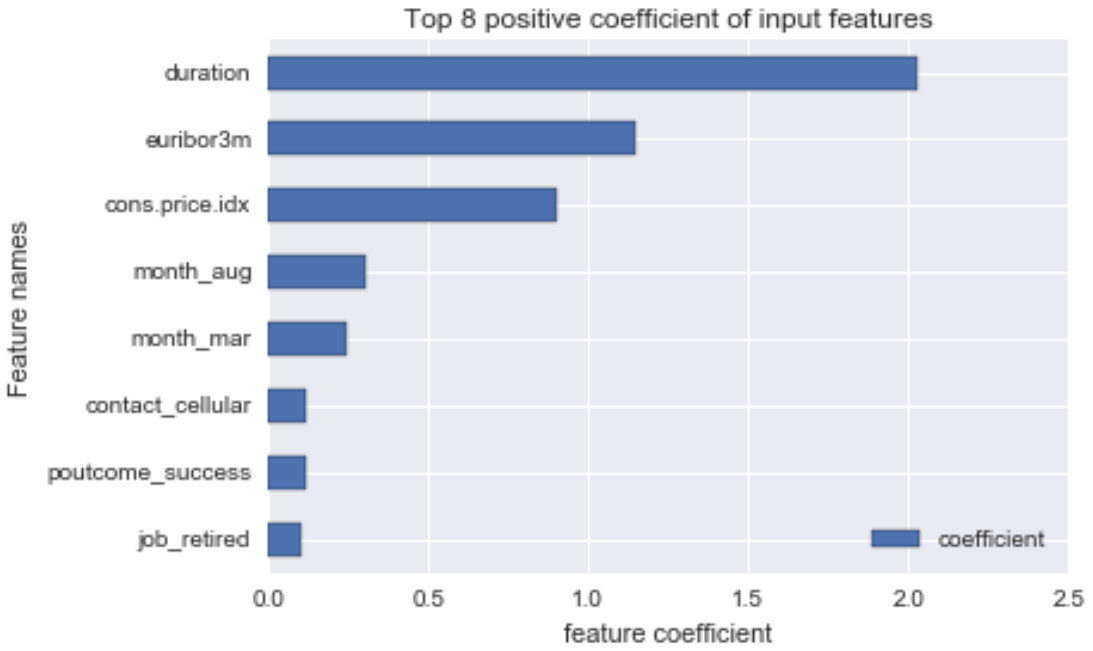

Figure 12 Top 8 positive coefficients of input features for Logistic Regression

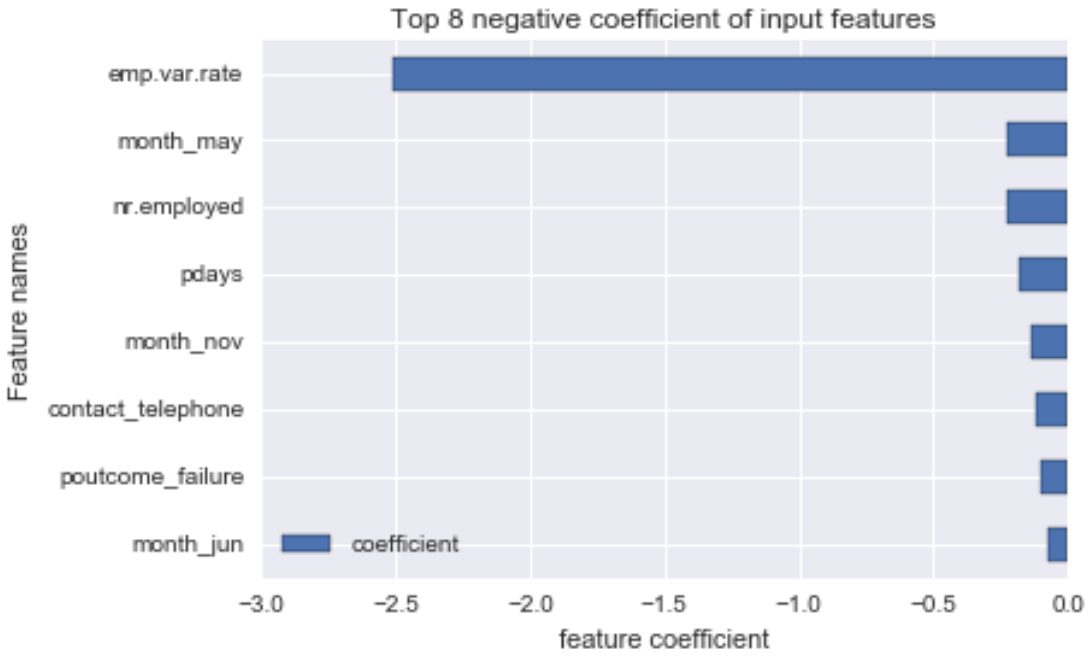

Figure 13 Top 8 negative coefficients of input features for Logistic Regression

Reflection

This project is a supervised binary classification problem. The dataset is downloaded from UCI's machine learning repository. There are 21 total features and 41188 clients, which are ordered by date. It is an imbalanced dataset. I applied SVM, Logistic Regression and Random Forest Classifier to the dataset. After tuning parameters, I found that Logistic Regression performed best. The AUC is 0.9427 and F1 score is 0.6471. After that, I applied tuned Logisitc Regression model to dataset of different testing sizes. F1 scores were relatively the same. Besides, I also exhibit top 8 respective input importance for Random Forest Classifier. And I showed Top 8 positive and negative coefficients of input features for Logistic Regression.

Improvement

For further improvements, Neural Network model could be applied to the dataset.
When focusing on the bank telemarketing, current bank context favors more sensitive models, which is model with higher recall score. The cost of each call will be keep at a low level when communication centers are at a relatively large scale. More importantly, the 2008 financial crisis strongly increased the pressure for Portuguese banks to increase long term deposits. Under these circumstances, it is better to produce more successful sells even if this involves wasting some effort on contacting non-buyers. So it's a good choice to evaluate models using recall score.

References

[1] https://en.wikipedia.org/wiki/Telemarketing

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

[3] http://dx.doi.org/10.1016/j.dss.2014.03.001

[4]https://www.kaggle.com/forums/f/15/kaggle-forum/t/7517/precision-recall-auc-vs-roc-auc-for-class-imbalance-problems/41179

[5] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

[6] https://en.wikipedia.org/wiki/Precision_and_recall

[7] https://en.wikipedia.org/wiki/Skewness

[8] http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[9] http://scikit-learn.org/stable/modules/svm.html#svm-classification

[10] http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

[11] http://scikit-learn.org/dev/modules/ensemble.html#forest