An Application of Classification Models in Bank Marketing

Amanda Shu

June 2020

Abstract

This paper explores the classification problem in bank marketing. I compare performances of the models Naive Bayes, K Neighbors, logistic regression, support vector machines, decision tree, random forest, and AdaBoost.

1 Introduction

To reach out to new customers, firms often use telemarketing, where sales representative call prospective customers over the phone to sell products or services. These telemarketing campaigns, however, need to be effective in that they are able to sell enough of their products to customers to meet business goals. It is in the firm's best interest to target the audience that is most likely to subscribe, so that they can save money and focus their telemarketing resources towards those people.

Implementing a machine learning model that is able to accurately predict whether or not a person is going to subscribe would be helpful for firms to find that target audience to call. In this paper, I look at bank marketing campaign data from a Portuguese banking institution. I aim to compare different classification models and find the best performing model. A better understanding of what models performs the best could help banking institutions improve their telemarketing effects.

In section II, I describe the data set and the methods for the preprocessing and feature selection of the data. In section III, I run various classification models and report my results. The full code behind these two sections can be found on Github[1].

2 Data

2.1 Dataset Description

The data comes from the marketing campaign of a Portuguese bank, collected from May 2008 to November 2010 [1]. This is a publicly available dataset that can be found on the UCI Machine Learning Repository[2]. Each row in this dataset relates to a particular bank client, and the outcome variable y is a binary variable that describes whether or not the client subscribed to a term deposit. Of all clients, approximately 11.3% of them subscribed. There are no null values in any of the features. In total, there are 41,188 instances and 20 features. Table 1 shows the descriptive statistics for the numerical features, and Table 2 shows the four most common values in each of the categorical features.

Table 1: Basic Statistics of Numerical Features

Variable	Mean	SD	Min.	Max.
age	40.02	10.42	17	98
duration	258.29	259.28	0	4918
campaign	2.57	2.77	1	56
pdays	962.48	186.91	0	999
previous	0.17	0.49	0	7
emp.var.rate	0.08	1.57	-3.40	1.4
cons.price.idx	93.58	0.58	92.2	94.77
cons.conf.idx	-40.50	4.63	-50.8	-26.9
euribor3m	3.62	1.73	0.63	5.045
nr.employed	5167.04	72.25	4963.6	5228.1

Table 2: Top 4 Common Values of Cat. Features

Variable	Values
job	admin, blue-collar, technician, services, ...
marital	married, single, divorced, unknown
education	university.degree, high.school, basic.9yr, professional.course, ...
default	no, unknown, yes
housing	yes, no, unknown
loan	no, yes, unknown
contact	cellular, telephone
month	may, july, aug, june, ...
dayofweek	thurs, mon, wed, tue, ...
poutcome	nonexistent, failure, success

[1] https://github.com/amandashu/Bank-Marketing

[2] https://archive.ics.uci.edu/ml/index.php

2.2 Data Preprocessing

Since most machine learning models cannot deal with categorical data, the categorical features must be converted into numerical values. First, all categorical variables except *education* (*job, marital, default, housing, loan, contact, month, dayofweek, poutcome*) were one-hot encoded. This means that the original variable is dropped and replaced with a binary variables for each unique value of that variable. For example the *month* feature is replaced with a feature for each month of the year, and for those new features the value is 1 if the client was contacted in that particular month.

Unlike the other features, the *education* feature has an implicit ordering (i.e. college education is higher than high school). Thus ordinal encoded was performed as such: 'unknown' maps to -1, 'illiterate' 0, 'basic.4y' 1, 'basic.6y' 2, 'basic.9y' 3, 'high.school' 4, 'professional.course' 5, and 'university.degree' 6.

Since the numerical features have a large variety of ranges, the numerical features are all standardized to have a mean of 0 and standard deviation of 1. It is especially important to standardize data before using support vector machines (which will later be used), since this model finds the hyperplane with the maximum maximum margin, and this calculation is affected by the scale of input features. Table 3 summarizes all the preprocessing steps described above. The resulting data after preprocessing has 56 columns.

Table 3: Preprocessing

Step	Variables
one-hot encoding	job, marital, default, housing, loan, contact, month, dayofweek, poutcome
ordinal encoding	education
standarization	age, duration, campaign, pdays, previous, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, nr.employed

2.3 Feature Selection

Feature selection is important as it reduces the possibility of overfitting, where the model fits the training data too well and does not generalize to unseen data. Thus I will run each model twice: one with all input features and another with certain selected features. The selected features will be decided based on statistical tests (chi-squared and F-test).

I selected features from the categorical variables by comparing chi-squared test results. The chi-squared test determines if the frequency of categories for a categorical variable is significantly different than the expected frequencies when partitioned into groups (subscribed or not). Figure 1 shows the features importances for each categorical feature. The selected categorical features (highlighted in blue) are the one hot encoded *contact, default, poutcome, job, marital*, and the ordinal encoded *education*.

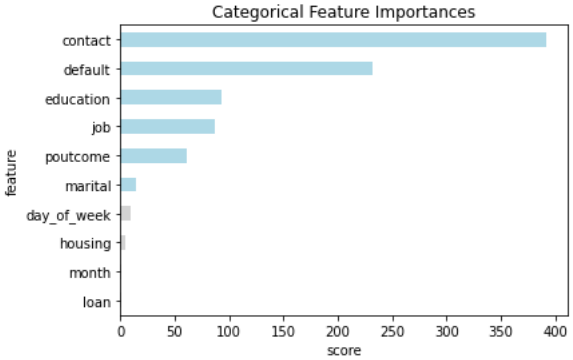

Figure 1: Categorical Feature Importances

I selected features from the numerical variables by comparing F-squared test results. The F-test tests if two samples' variances are significantly different from each other. Figure 2 shows the features importances for each numerical feature. The selected numerical features are *duration, nr.employed, pdays, euribor.3m, emp.var.rate, previous*, and *cons.price.idx*.

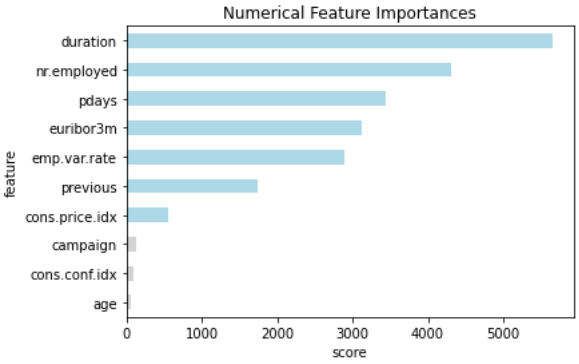

Figure 2: Numerical Feature Importances

With only the selected categorical and numerical features, the resulting table has 32 columns.

3 Results

In this section I detail the results of the various classification models and compare their performances.

F1 score: The F1 score is the harmonic mean between precision (number of correctly predicted positive values over total predicted positive values) and recall (number of correctly predicted positive values over total actual positive values). I choose to use this metric to compare my models with since I don't necessarily want to minimize false positives or false negatives in particular (which is what precision and recall minimizes respectively). In other words, I view predicting a client to subscribe when the client does not in reality and predicting a client to not subscribe when in reality the client does as having equal cost, since neither case has a stronger consequence over the other. The F1 score is ideal then since it balances precision and recall. Additionally, this data has class imbalance as there are approximately 89% true negatives, so using a metric such as accuracy would not be ideal.

Precision Recall AUC: The Precision Recall curve is a plot of the recall on the x axis and precision on the y axis at different probability thresholds. The model is perfect at the point (1,1) and a good model will have a high AUC, which is the area under the precision recall curve. I choose this metric to compare between the models because of the class imbalance as mentioned before. Also, unlike the F1 score, the PR AUC tells me how good a model is over different probability thresholds.

3.1 Model Results

Naive Bayes Classifier: I use the Naive Bayes classifier as my baseline model. The baseline model is used to have a basis for comparison of metrics. Naive Bayes classifier makes the strong assumption that the features are independent conditional on the class label, making it a simple model that can be used for a baseline model. It also does not have many parameters to tune. The F1 score for the Naive Bayes classifier is 0.324 with all the features and increases to 0.416 when ran with only selected features. The PR AUC is 0.435. I expect the F1 score and PR AUC for all the other models to be higher than these baselines.

K Neighbors Classifier: The next classifier I use is the K neighbors classifier, which assigns points the majority class of its nearest K neighbors. I use the default of k=5 and uniform weights. With this model the F1-score is 0.449 for all features and 0.507 with selected features, which is higher than the baseline model. The AUC is 0.533.

Logistic Regression: Logistic regression uses a logistic function to classify a binary label. I first tune the model parameter C, which is inverse regularization, so smaller values will reduce overfitting. Figure 3 is the validation curve, which shows the training and cross validation F1 scores for the C values 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, and 10000. The F1-score with all the features in the model is 0.413 and 0.485 for the selected features. The AUC is 0.564.

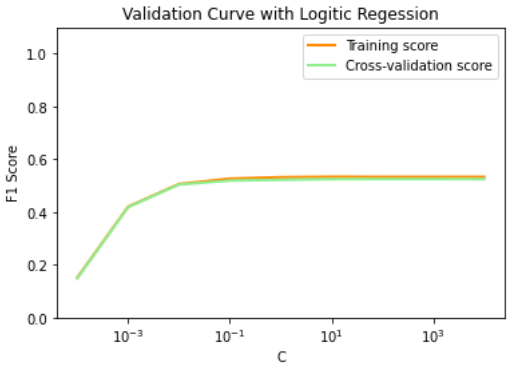

Figure 3: Validation Curve

Support Vector Machine: Support vector machine uses kernel functions to map data to higher dimensional spaces and fits the maximum margin hyperplane to the new space. Since the support vector machine is computationally expensive, I use a smaller subset of the bank data to do parameter tuning. I perform cross validation with parameter values 0.01, 0.1, 1, 10, and 100 for C (regularization parameter) and linear, poly, and rbf for the kernel type. For the model with all the features, the values C=10 and linear kernel were the best. The F1 score is 0.404. For the model with selected features, the values C=10 and rbf kernel performed the best. The F1 score and PR AUC for the second model are 0.456 and 0.577 respectively.

Decision Tree Classifier: In decision tree classifier, each node is a rule on an attribute that splits the data further. The best splitting rule is decided by the largest difference in Gini index, and the tree is split until the stopping criterion is met. I performed cross validation with the min_samples_split (minimum number of samples required to split an internal node) parameter values 5, 20, 50, and 100 and the max_depth (maximum depth of the tree) parameter values 4, 6, 8, 10, 12, and 14. The best performing parameters according to the F1 score is max_depth=6 and min_samples_split=100. The F1 score for the model with all features is 0.570 and with selected features is 0.56. The PR AUC is 0.574. Figure 4 shows the first four levels of the second decision tree model.

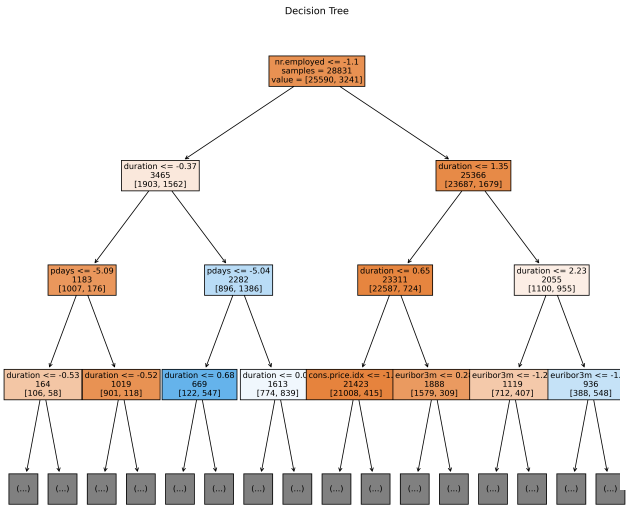

Figure 4: Decision Tree

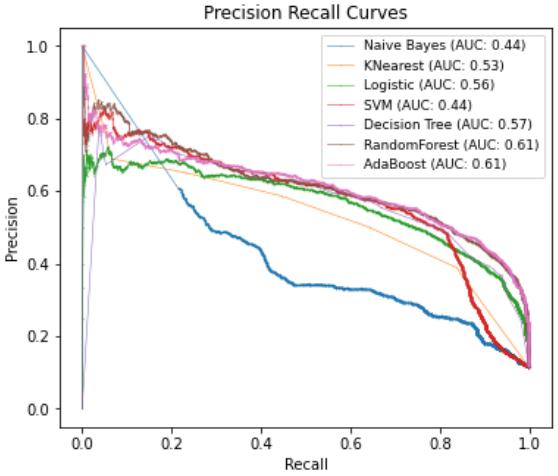

Figure 5: Precision Recall Curves

Random Forest Classifier: The random forest classifier averages predictions from decision tree models trained on bootstrapped data. This model is generally better than a simple decision tree as unlike the decision tree it is not prone to overfitting. I again perform cross validation with the parameter values 5, 20, 50, and 100 for min_samples_split and 4, 6, 8, 10, 12, and 14 for max_depth. The F1 score for the model with all features is 0.382 and with selected features is 0.504. The PR AUC is 0.616.

AdaBoost Classifier: Adaboost is an ensemble method in which each weak classifier is improved upon based on mistakes of the previous model. I perform cross validation to tune the learning_rate parameter with the values 0.01, 0.1, and 1, and I find that a learning rate of 0.01 is the best. I run 300 boosted decision tree classifiers with the same tuned parameters as I found with my decision tree classifier. The F1 score for the model with all features is 0.510 and with selected features is 0.580. The PR AUC for the second model is 0.610.

3.2 Discussion

Table 4 compares the scores between all the models. I included the accuracy score to show that accuracy would not be a good metric to compare models with; the accuracy score for all models are high to due class imbalance. I also note that between models that included all features and models that include only selected features, the model with selected features have a better F1 score. This is a clear demonstration of the importance of feature selection.

Based on the results, the best model is AdaBoost since it has a high F1 score and AUC. It improves on the baseline model F1 score by 35%. However, I note that the scores not very high (around 0.6), so perhaps more improvements could be made to the data preprocessing, such as oversampling the data to balance the classes or collecting better features.

4 Conclusion

In this paper, I compared different classification models on bank marketing campaign data and found that the AdaBoost classifier performed the

Table 4: Comparing Scores

	All Features		Selected Features		
Model	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**PR AUC**
Naive Bayes	0.65	0.32	0.76	0.42	0.44
K Neighbor	0.90	0.45	0.90	0.51	0.53
Logistic	0.9	0.41	0.91	0.49	0.56
SVM	0.9	0.4	0.91	0.46	0.58
Decision Tree	0.91	0.57	0.91	0.56	0.57
Random Forest	0.90	0.38	0.91	0.5	0.62
AdaBoost	0.91	0.51	0.91	0.58	0.61

best. My results also reiterated the importance of feature selection, as performing feature selection improved the metrics.

This paper's process and results could be helpful for firms looking to implement machine learning models to improve their own telemarketing campaigns. It's important to note that although my results show the AdaBoost classifier is the best model, the result is particular to this dataset and the preprocessing I performed. Thus, in real world applications, firms should tune models for their own customer data. If they are satisfied with their model performance, they can also build their own web applications in which sales representatives enter customer information and decide whether it is worth calling that customer. An interesting topic to study would be to see if an implementation of a web application for sales professionals would improve firms' sales numbers.

References

[1] S. Moro, P. Cortez and P. Rita. *A Data-Driven Approach to Predict the Success of Bank Telemarketing.* Decision Support Systems, Elsevier, 62:22-31, June 2014