Predicting the Success of Telemarketing in Subscription of a Term Deposit Using Machine Learning Methods

Arden Kim alkim@wisc.edu
Bella Wu zwu363@wisc.edu
Emma Chen wchen436@wisc.edu
Mengwei Sun msun247@wisc.edu

Abstract

This project aims to help banks predict whether a client will subscribe to a term deposit marketed through the channel of phone calls as well as improve the marketing conversion rate by targeting a certain population of clients. The dataset used[2] is from a direct marketing campaign of a Portuguese banking institution, with 41188 observations, 20 features, and a binary target variable of a client's subscription response. With logistic regression as the baseline model, this project compares KNN, Adaboost, Xgboost, Decision Tree, and Random Forest in terms of accuracy, recall, precision, and F1 scores. The results show that a balanced Decision Tree classifier is highly recommended for prediction and banks should aim at the population with higher income, more advanced educational background, and age group of twenties, thirties, sixties and seventies when advertising through phone calls.

1. Introduction

A term deposit is a type of deposit in a fixed term with a fixed interest rate at a financial institution. It is a significant component of a bank's inflow which funds loans, credits, and other cash outflows. Therefore, a bank needs to increase its cash flow and stabilize its liquidity, and banks usually employ various marketing strategies to solicit additional term deposits. For example, a majority of banks use targeted advertising, such as telemarketing, to identify prospective clients, forecast success ratios, and improve marketing efficiency. In this way, banks can learn clients' behaviors through data-driven approaches, identify prospective clients, and predict subscription rates of term deposits marketed directly via phone calls.

Specifically, our group is interested in optimizing telemarketing in banking, which has been one of the long-established methods. As some people may reach out to banks for more information about stocks and investments, most people are unaware of other low-risk options, term deposit as an example. Telemarketing allows banks to contact a wider range of customers and provide interactive services for customers.

Our project's objective is to build machine learning classifiers to predict whether a client will subscribe to a term deposit marketed through the channel of phone calls and help banks in targeting specific customers to improve the conversion rate, so that banks could apply a marketing strategy in reducing costs, increasing profits, and providing and fulfilling needs for people who are more willing to subscribe a term deposit. The classifiers may also apply to other marketing strategies focusing on the same targeted audience, not solely to telephone marketing. For example, email marketing may be used as an alternative to attract a large number of clients. As emails became more popular in recent years, applying the classifiers in this project to email marketing and other digital marketing strategies will be extremely meaningful. The classifiers in this project is trained on a dataset[2] related to the direct marketing campaign of a Portuguese banking institution and retrieved from the UCI Machine Learning Repository. The project will apply Logistic Regression, K-Nearest Neighbors (KNN), Adaboost, Xgboost, Decision Tree, Random Forest for constructing prediction models, with accuracy, precision, recall, F1 scores, ROC AUC as the performance metrics.

2. Related Work

Sergio Moro et al.[1] used data-driven approaches to analyze the original dataset of 150 variables retrieved from a telemarketing campaign of a Portuguese bank to predict the success of term deposit subscription. They selected 22 features out of the 150 features, including the month when the call is made. Their training set and test set were splited based on time order, with the test set containing contacts from July 2012 to June 2013. They applied Decision Trees, Neural Networks (NN), Support Vector Machine (SVM), and Logistic Regression to the dataset and compared each model based on the Area of the Receiver Operating Characteristic Curve (AUC) and the Area of Lift Cumulative Curve (ALIFT). NN achieved the highest performance with an AUC score of 0.8 and ALIFT of 0.78. Sergio Moro et al.[1] also conducted knowledge extraction analysis on NN and Decision Trees. They found that the 3 month Euribor rate was most relevant to term deposit subscription among 22 features. They explained that the Euribor rate negatively affected savings because people might regard the lower Euribor rate as a signal of economic downturn and more prudent about investments.

3. Proposed Method

3.1. Methods

3.1.1 Logistic Regression

Logistic Regression is one of the simplest and basic machine learning models for binary classification with great interpretability so this is adopted as the baseline model. By applying the sigmoid function, this classifier has an 'S' shaped graph in the range of 0 and 1, which indicates the probability of a binary event.

3.1.2 K-Nearest Neighbor

The k-Nearest Neighbor (KNN) is a machine learning algorithm that classifies a query point based on the class of its neighbors. It evaluates the classification based on the number of neighbors, k, according to the decision boundary which could calculated by different distance metrics.

3.1.3 Decision Tree

Decision Tree works as a hierarchy of decisions to predict class or value, learning decision rules from training data. *Gini* and *Entropy* are the two types of splitting criterion for measuring the quality of a split. In this project, the weighting scheme in the Decision Tree classifier is set to be balanced to avoid the unbalanced distribution of the target variable.

3.1.4 Random Forest

Random Forest is an ensemble method that combines and implements the concepts of bagging and Decision Tree, where different bootstrap samples from the training data are fitted with Decision Trees. At each node, random feature subsets are derived from deciding upon the optimal split to reduce variance and improve accuracy. Similar to Decision Tree, Random Forest also includes the weighting scheme set as balanced here, to balance the distribution of the target variable.

3.1.5 Adaboost

Adaboost, which stands for adaptive boosting, is an ensemble method that tries to boost the weak learner according to the mistakes of previous models. The weak learner is usually a stump, which stands for a tree with only one node and two leaves. An ensemble of stumps is conducted with differently assigned weights for each stump based on the classification performance.

3.1.6 Xgboost

Xgboost is also an ensemble that combines the outputs from individual trees. It uses boosting by combining weak learners sequentially so that each new tree corrects the errors of the previous one and this process continues until no further improvements can be made.

3.2. Evaluation of Performance of Models

In order to better understand and evaluate the performances of classifiers used, The confusion matrix is applied to all models for reference. Five different performance metrics are considered including accuracy, precision, recall, F1 score, and Area under Receiver Operating Characteristic curve (ROC AUC). Specifically, accuracy, precision, recall, F1 score, and AUC-ROC are taken into account for model sensitivity, while emphasizing F1 and recall due to the imbalanced target variable.

3.2.1 Confusion Matrix

Confusion Matrix contains values of true positives, true negatives, false positives, and false negatives as indicated in Figure 1. Under the context of our project, true positives are cases where classifiers correctly predict a client subscribing to the term deposit, and in contrast true negative are cases where classifiers correctly predict a client not subscribing to the term deposit. When a classifier has a high false negative rate, then it fails to correctly predict the class of not subscribing; when a classifier has a high false positive rate, it fails to correctly predict the other class.

Figure 1. Confusion Matrix

		Predicted condition	
		Positive	Negative
True condition	Positive	TP	FN
	Negative	FP	TN

TP: True Positive, TN: True Negative, FN: False Negative, FP: False Positive

3.2.2 Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy indicates the number of correctly predicted labels over all predictions, implying the number of correctly classified data. However, it is not a recommended measure of performance in cases where dataset labels are imbalanced with high risks of leading to misleading results.

3.2.3 Precision

$$Precision = \frac{TP}{TP + FP}$$

Precision represents the number of true positives over predicted positives, suggesting the classifier precision on forecasting the clients who will subscribe without making false predictions to those who will not.

3.2.4 Recall

$$Recall = \frac{TP}{TP + FN}$$

Similar to the concept of precision, recall shows the percentage of predicting true positives. However, it indicates the number of correctly predicted positive cases over all the actual positives. Contrasting to precision, it suggests how well the model catches all clients who will subscribe without missing potential clients.

3.2.5 F1 Score

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

The F1 score is the weighted average of precision and recall. The higher the F1 score, the better our model predicts clients who will subscribe when false positives and false negatives are low.

3.2.6 ROC AUC

Lastly, the Receiver Operating Characteristic (ROC) curve plots for false positive rates against true positive rates (recall), where false positive rates are the proportion of misclassified negative class. The Area under the ROC curve (ROC-AUC) measures the area under the ROC-AUC curve and ranges from 0 to 1.

4. Experiment

4.1. Data Overview

The dataset[2] is about a direct marketing campaign of a Portuguese banking institution and is retrieved from the UCI Machine Learning Repository. There are 41188 data points and 20 features in total, including 10 categorical features and 10 numerical features about clients' backgrounds, general economic environments, and details of marketing campaigns. The target variable `y` is binary and represents whether the client has subscribed to a term deposit or not. This variable is highly unbalanced, with 88.73% of data points belonging to the class 'no', indicating that a client has not subscribed to a term deposit.

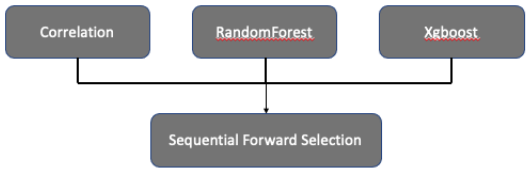
4.2. Data Cleaning

There are several missing values in features `job`, `marital`, `education`, `default`, `housing`, `loan` in the dataset. It is paramount to handle the missing values appropriately before training the models. For features with less than 1000 missing values, including `job`, `marital`, `housing`, `loan`, our group decides to drop those data points since 1000 is insignificant compared to the size of the dataset. On the other hand, for variables with more than 1000 missing values like `education`, `default`, our group members regard missing values as a new factor level to preserve more information than simply dropping those. Since some classifiers such as KNN is sensitive to distance, feature scaling is also performed on the dataset to avoid such problems.

Before model training, the binary feature `duration` should be dropped. This feature represents the last contact duration and the value is unknown until contact with a client, such as a phone call, is performed. Since our objective is to train a model capable of predicting subscriptions before contacting clients, and if the client never received such a phone call, this feature is useless. Binary features like `y`, `loan`, `housing` are converted to features with levels 1, 0. Our group performs ordinal encoding on ordinal features `month`, `day_of_week` and changes the rest of the categorical features to dummy variables.

After data cleaning, there are 39803 data points and 37 features in total with a binary target variable.

Figure 2. A flow chart of the feature selection process



Next, different feature selection methods are applied to improve computational efficiency and reduce the model's generalization error by removing irrelevant features or noise. Moreover, since our primary goal is to help with client profiling and identify prospective clients, there should not be a large number of features. As shown in Figure 2, our group utilizes Pearson's correlation, Random Forest, and Xgboost to calculate feature importance and selects three sets of top ten important features respectively. In the three sets of ten features, there are 17 distinct features, and again the Sequential Forward Selection (SFS) is operated on these 17 features to select the top ten important features as indicated in Figure 2. The ultimate dataset with feature selection contains `housing`, `pdays`, `previous`, `emp.var.rate`, `cons.conf.idx`, `euribor3m`, `nr.employed`, `contact_telephone`, `poutcome_nonexistent`, `poutcome_success`.

In training and testing the classifiers, it is necessary to prepare the train and test data split. As stratification ensures the same distribution of class classes in both sets, our group uses a stratified split to split the dataset into a 75% training set with 29852 observations and a 25% testing set with 9951 observations. Due to the unbalanced dataset, it is challenging to evaluate model performance, and thus we resampled a balanced test set with only 2244 observations from the test set obtained above. After training the baseline model, which will be discussed later, the performances of models trained with training set with feature selection and without feature selection are similar. Therefore, in the following analysis, training sets with or without feature selection are applied to interpret whether feature selection affects performance.

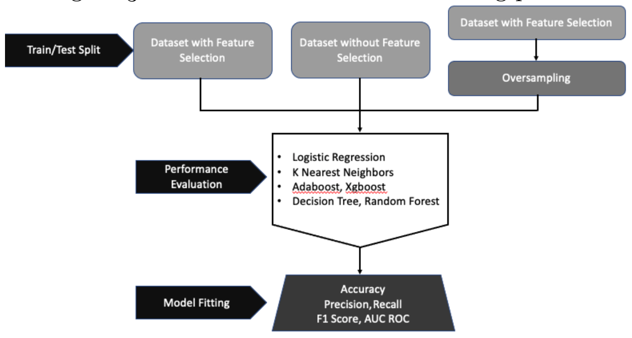
While the test set is balanced thanks to resampling, the training set mentioned above is still unbalanced. To further tackle the problem, we applied oversampling on the training set for the minority class, which is the class when the client has made a deposit subscription. In this way, both sets are corrected to have a balanced distribution. In the following study, three different training sets are used when running the model as demonstrated in Figure 3:

- The training set with feature selection
- The training set without feature selection
- The oversampled training set with feature selection

4.3. Methods

All of the classifiers used in this report are tuned by the grid search except Logistic Regression classifier

Figure 3. A flow chart of classifier training process



which involves zero hyperparameter. The hypterparameters of each final classifier is indicated in Table 1

4.3.1 Logistic Regression

As mentioned above, three Logistic Regression models are built and the Logistic Regression model with the complete set of features serves as our baseline model so that our group could compare the effect of feature selection.

4.3.2 KNN

An optimal KNN is constructed with hyperparameters chosen by grid search with 3-Fold Cross-Validation. To solve problems of computation inefficiency, the number of neighbors (k), is chosen in a range of values from 50 to 300 with a step size of 50. Uniform and Distance weighting schemes are experimented with along with

Table 1. Hyperparameters in each classifier after grid search

Classifier	Hyperparameters
KNN	n_neighbors=300, weights='uniform', metric='manhattan'
Adaboost	n_estimators=800, learning_rate=0.1
Xgboost	n_estimators=100, max_depth=4, learning_rate=0.1
Decision Tree	criterion='entropy', max_depth=6, min_samples_leaf=4, min_samples_split=2, class_weight="balanced"
Random Forest	criterion='gini', max_depth=6, max_features='sqrt', n_estimators=200, class_weight="balanced"

Minkowski, Euclidean, and Manhattan distance metrics. The optimal KNN model is shown in Table 1.

4.3.3 Decision Tree

For Decisition Tree, information criteria, including *Entropy* and *Gini*, are examined. The maximum depth of trees is chosen from a sequence of integer values from 1 to 20. The hyperparameter *min_samples_leaf* and *min_samples_split* is respectively chosen from 1 to 10 and from 1 to 5. The optimal Decision Tree is shown in Table 1.

4.3.4 Random Forest

In Random Forest, the estimator number and maximum depth are respectively selected from 200 to 500 and 4 to 8. The maximum number of features allowed for each tree is determined among three options: no restrictions(auto), square root of the total number of features in each run (sqrt), or 20% of variables in each run(log2). Two information criteria, including *Gini* and *Entropy*, will be examined. The optimal Decision Tree is shown in Table 1.

4.3.5 Adaboost

For Adaboost, estimator number is chosen from 10, 50, 100, 500, 800, and 1000, and the learning rate is picked from values of 0.001, 0.01, and 0.1. The result from the hyperparameter tuning shows that the optimal Adaboost model uses 800 estimators and a learning rate of 0.1.

4.3.6 Xgboost

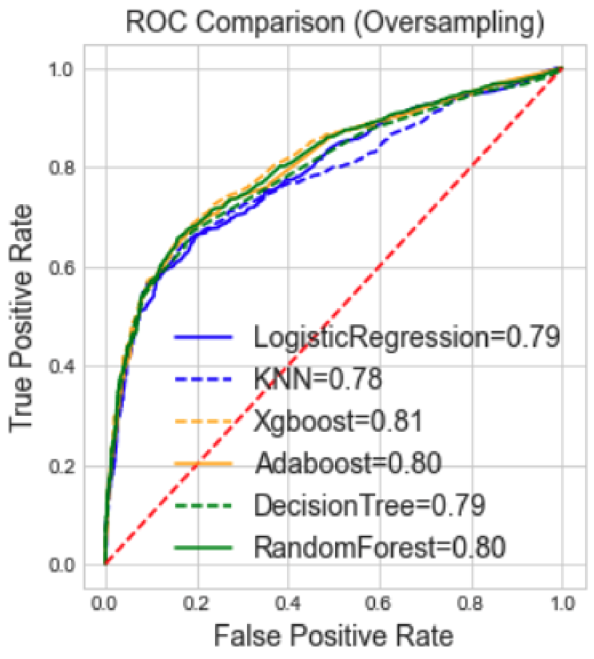
In Xgboost, the number of estimators is chosen from 50, 100, 500, 1000, 2000, and the learning rate is chosen from 0.001, 0.01, and 0.1. The maximum depth is chosen from values of 4, 6, 8, and 10. The optimal Xgboost classifier is shown in Table 1.

5. Results

In terms of the five performance metrics, using the ten selected features yields similar performance compared with using all features for all types of classifiers when trained on the ordinary stratified training sample according to Figure 5, especially for Adaboost, Xgboost, and Random Forest. There is a small negative effect of feature selection on Logistic Regression while feature selection positively affects KNN and Decision Tree though to a small extent. For the bank to collect data more easier and identify client more efficiently, it is crucial to control the number of features in our models without compromising predictive performance. Using only the ten selected features decreases model complexity while ensuring original predictive performance. Therefore, discussions below will exclude the classifiers trained on set without feature selection and only consider classifiers trained on only the ten selected features with or without oversampling.

Table 2 demonstrates all 5 performance metrics for each classifier. Obviously, the values of precision and ROC AUC of different classifiers have little variation and this is due to the fact that these two performance metrics are biased when the dataset is imbalanced. This pattern is more distinct when taking a look at Figure 4 where different colors represent the ROC of each classifier trained on the oversampling set and the red dashed line in middle of the plot is the reference line of 0.5 ROC AUC score. All of the lines are almost overlapping each other, and therefore it is not meaningful to evaluate ROC. In the case of the imbalanced dataset, using performance metrics like recall and F1 score could minimize the bias compared with ROC_AUC and precision.

Figure 4. ROC AUC for each classifier trained on oversampling training set



As mentioned earlier, the test set is resampled to have a balanced distribution due to our imbalanced target variable. An oversampling method is also applied to the training set and changes the classification weight for random forest and decision tree classifiers. For classifiers including Logistic Regression, Adaboost, and Xgboost, the performances significantly improve

Figure 5. F1 score in each classifier

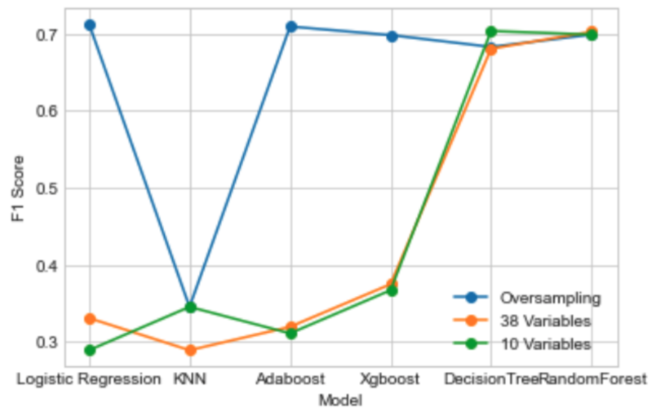


Table 2. Performance metrics for each classifier

Classifier	Accuracy	Precision	Recall	F1	ROCAUC
Logistic Regression[ε]	0.593	0.930	0.201	0.331	0.786
Logistic Regression[b]	0.580	0.937	0.171	0.289	0.785
Logistic Regression[c]	0.734	0.772	0.662	0.713	0.785
KNN[a]	0.580	0.937	0.171	0.289	0.794
KNN[b]	0.598	0.933	0.212	0.346	0.803
KNN[c]	0.598	0.933	0.212	0.346	0.779
Adaboost[a]	0.590	0.943	0.193	0.320	0.803
Adaboost[b]	0.588	0.946	0.186	0.311	0.802
Adaboost[c]	0.743	0.815	0.628	0.710	0.801
Xgboost[a]	0.611	0.953	0.234	0.375	0.811
Xgboost[b]	0.607	0.941	0.228	0.367	0.809
Xgboost[c]	0.738	0.822	0.607	0.698	0.807
Decision Tree[a]	0.731	0.836	0.574	0.681	0.778
Decision Tree[b]	0.740	0.816	0.619	0.704	0.789
Decision Tree[c]	0.732	0.834	0.578	0.683	0.791
Random Forest[a]	0.744	0.838	0.605	0.703	0.804
Random Forest[b]	0.740	0.832	0.601	0.698	0.807
Random Forest[c]	0.740	0.831	0.602	0.698	0.805

[a] Trained on the training set with feature selection
[b] Trained on the training set without feature selection
[c] Trained on the oversampling training set with feature selection

after being trained on an oversampled sample according to Table 2. The improved performance is clearly represented in Figure 5 where the blue line of models with oversampling is above the green line and the orange line. For instance, compared with training without oversampling, the values of accuracy, recall, and F1 scores of Logistic Regression, Adaboost, and Xgboost nearly double after being trained with an oversampling training set. The classifiers mentioned above are susceptible to unbalanced data because, by default, more weights are given to the majority class, and predictions will be biased toward the majority class. What's worse, training the above three classifiers on an imbalanced dataset would result in low recall scores, implying higher odds of misidentifying prospect clients as non-prospect clients. For classifiers including Decision Tree and Random Forest, setting a balance weight scheme by adding hyperparameter *class_weight* as 'balanced' automatically offset the impacts of the unbalanced dataset so that the five scores are similar regardless of whether oversampling is used in training samples. Classes are weighted inversely to their proportions in a balanced Decision Tree. A Random Forest classifier with a balanced weight scheme applies an oversampling strategy to get each bootstrap sample. Correcting the unbalanced distribution, both balanced Decision Tree and balanced Random Forest are resistant to imbalance dataset no matter what training samples are used.

While performances for Logistic regression, Adaboost, and Xgboost are satisfactory after correcting for bias due to an unbalanced dataset, the performances of KNN are consistently unsatisfactory compared with the baseline model. In Table 2, the highest accuracy and F1 score KNN achieved are 0.59 and 0.346 respectively. The bad performances of KNN are not unexpected for several reasons. First, KNN, a lazy and instance-based algorithm, is too simple to perform well on a large dataset because this classifier stores all points in the training set and compares each new point with all the training points for classification, resulting in enormous computation costs and degraded performances. Second, although there are only ten selected features, KNN may still be cursed by dimensionality. In the higher dimension space, less similar points around the query point will be considered and eventually negatively impact the performance. Due to poor performances, the KNN classifier will not be further considered in later discussion.

In addition to predictive performance, the interpretability of the model should also be examined since a model of higher interpretability helps banks make informative decisions. Because Adaboost, Xgboost, and Random Forest are black-box methods whose internal working details are unrevealed, they are not interpretable and fail to provide insights into the banking system. Investigations into Decision Tree splitting nodes and the statistically important parameters of Logistic Regression offer perceptions into the banks' marketing strategies. The root of the obtained Decision Tree model is the `nr.employed`, or the number of employments. The root could explain that the increase in employment signifies the economic growth and thus, society is more optimistic about making investments including the term deposit. Furthermore, based on the coefficients of the improved Logistic Regression model, variables including `nr.employed`, `emp.var.rate`, and `euribor3m` have relatively larger effects on the odds of subscribing to term deposits. `emp.var.rate` (employment variation rate) measures the degree of shifts in the labor market, and if the labor market shift enormously, the odds of term deposit subscription might decrease because people are more hesitant about any investments, including term deposits. `euribor3m` (Euribor 3 month rate) is the basic reference interest rate for the economy where a higher Euribor 3 month rate often indicates a higher interest rate for investments, stimulating more savings. As implied by the Decision Tree and Logistic Regression, banks should employ more targeting advertisements via phone when the general economic environment is optimistic with higher number of employments, less employment shifts and higher Euribor rate and vice versa.

In addition to predicting the success of phone-based marketing campaigns, identifying individuals who are more likely to make term deposits might be handy for a higher client conversion rate. In Figure 6, administrators, technicians, students, and the retired group have larger proportions of people subscribing to term deposits. Administrators and technicians are high-income groups with extra assets to be invested. The retired groups, who might have life-long cumulative savings and fewer incentives for expenses, might consider low-risk investments like term deposits. Students might start learning about investment with lower-risk options and there are numerous student friendly term deposit options. Population groups in the twenties, thirties, sixties, and seventies have larger proportions of people subscribing to term deposits as shown in Figure 7. People in their sixties and seventies might have already retired, while people in their twenties and thirties might start considering accumulating their assets through deposit approaches. Furthermore, according to Figure 8, people with higher educational backgrounds show more interest in term deposit subscriptions since higher educational backgrounds may be highly correlated to higher-income status, more knowledge about investments, and more general acceptance of risks, thus more likely to subscribe to term deposits.

Though all the models are not time-consuming, the time complexity is worth considering, especially when banks put the classifiers in practice after acquiring a

Figure 6. Bar Chart for Different Jobs Subscribing Term Deposit

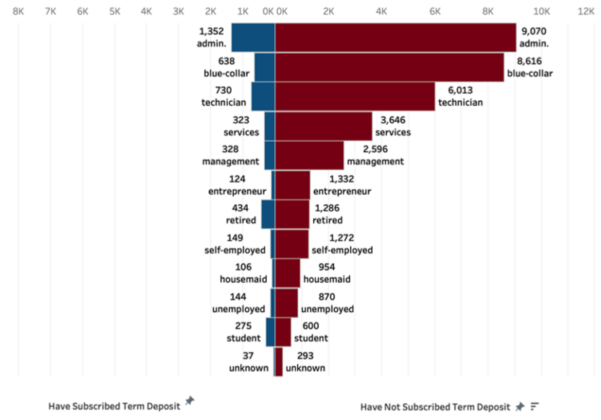


Figure 7. Bar Chart for Different Age Groups Subscribing Term Deposit

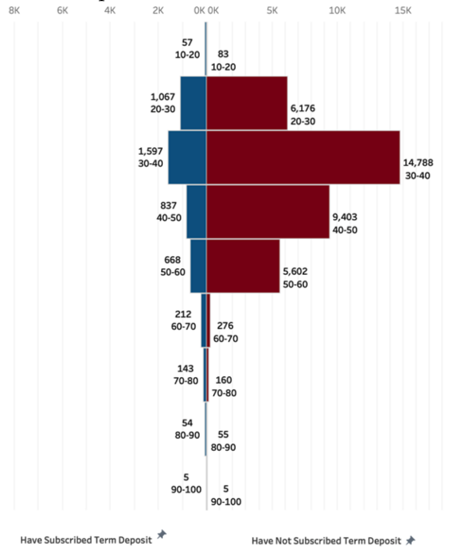
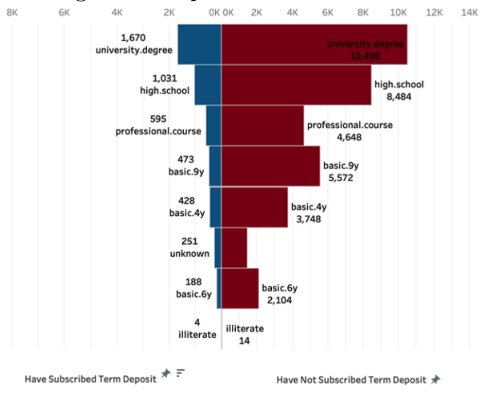
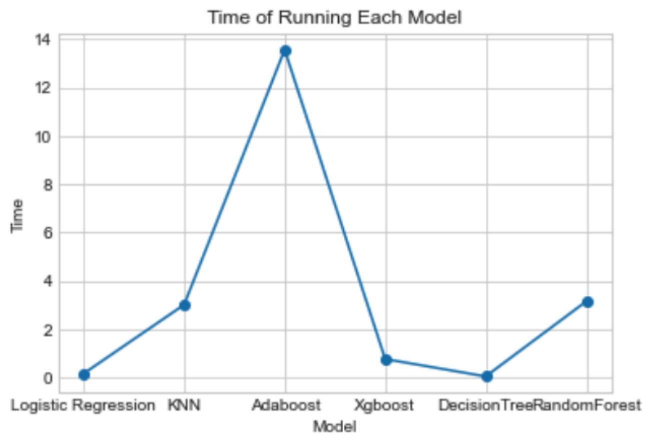


Figure 8. Bar Chart for Different Education Levels Subscribing Term Deposit



larger dataset. Figure 9 shows the runtime for each classifier, where Adaboost is the most computationally expensive classifier among all these models, probably because weight error rates are computed at the end of each round to updating weights sequentially. In comparison, Logistic Regression and Decision Tree are the least time-consuming classifiers. Decision Tree splits into smaller sets at each node and processes further splitting parallel and Logistic regression is fast because it only requires solving the optimization problem of a strictly convex function from a mathematical perspective.

Figure 9. Time for running each classifier



This project has some limitations and improvements that could be made in future studies. Decision Tree classifier could be further improved by pruning since the current final model contains 60 leaves and more than 100 nodes implying the tree model grows horizontally. Furthermore, the dataset used in this project is an incomplete subset from the dataset used in Sergio Moro et al.[1] research. Features such as time are not included in this dataset. If the time of the data points were known, a more reasonable approach of training classifiers on earlier data points and testing on more recent data points might be taken.

In conclusion, the project's goal is achieved with the balanced Decision Tree classifier as the recommended classifier for its consistently high performances, resistance to the unbalanced dataset, computational efficiency, and interpretability for simple predictive purposes. To achieve a high client conversion rate, banks should send advertisements specifically to the population with higher income, more advanced educational background, and age groups of the twenties, thirties, sixties, and seventies.

6. Conclusion

This project accomplishes its primary goal of building a simple, robust predictive model for subscription response to term deposits to help with client profiling, client identification, and marketing efficiency. The obtained Decision Tree have an accuracy of 0.74, F1 score of 0.7, and ROC-AUC of 0.79 with the advantages of time efficiency, stability, interpretability, and accuracy for basic predictive purposes and is highly recommended for future implementation among all classifiers. The ROC-AUC score achieved is close to the AUC score of 0.8 achieved by Sergio Moro et al.[1]. Aside from great predictive quality, Logistic Regression and Decision Tree could offer insightful feedback for the banks for knowledge extraction purposes with `emp.var.date`, euribor3m, and nr.employed identified as the three most relevant features. The positive effect of euribor3m in our study contradicts the negative effect of euribor3m in Sergio Moro et al.[1] study probably because Moro had access to the specific month when each contact is made, incorporating the time lag effect of euribor3m into classifiers. Although Adaboost also has great performances, it is the most time-consuming classifier, which could be problematic in reality. Regarding marketing efficiency, it is beneficial for banks to make telephone calls to the population with higher income, more advanced educational background, and age group of the twenties, thirties, sixties, and seventies.

To further improve model performance, more data should be collected though different banking system in different nations across a wider time range. Future studies should investigate how the features affect the subscription responses and provide constructive suggestions into marketing schemes such as the frequency of calls.

7. Contributions

Mengwei found the original dataset online, cleaned up the data, prepared training, and tested the data split for further analysis. Bella and Emma worked on coding classifiers together and parameter tuning. Arden worked on visualization, evaluation metrics, and feature selection. For the report, Arden and Emma worked on the Abstract, Introduction, Related Work, and Proposed Method section. Bella finished writing the experiment section and conclusion. Mengwei completed the results and discussion with the help of other group members.

References

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

[2] UCI Machine Learning Repository: Bank Marketing Data Set. (n.d.). Retrieved December 3, 2021, from https://archive.ics.uci.edu/ml/datasets/Bank+Marketing?package=regselamp;version=0.2.