COMPSCI 9114A
Introduction to Data Science
Final Project Report

Bank Direct Marketing Classifier

Presented by:

Zixuan Wang (zwang863, 250834887)

Bowen Wang (bwang489, 251139226)

Runcong Wu (rwu252, 251148344)

Abstract

Background:
Implementing a correct marketing strategy is critical to promote products and services sales. Compared to traditional mass marketing, direct marketing has the advantages of building personal relationships with customers in a very efficient way.

Objectives:
To develop predictive models for bank deposit subscription campaigns. Predicting customers who subscribe to a term deposit can be useful to identify characteristics and traits that affect success. Our goal is to target prospects that have a high probability of subscription and reduce the resources used in direct marketing.

Method:
Data from direct marketing campaigns of a Portuguese banking institution were split into training and testing sets. Three supervised learning methods were used to train classifiers in this study, including Logistic Regression, XGboosting, and Random Forest, plus a Dummy Classifier. 10-fold cross-validation on the training data set was used for model selection. The selection criteria were based on the recall/sensitivity and area under the curve (AUROC). We selected one model for each criterion, then we evaluated the model performance on test data.

Results:
We selected two models based on the cross-validation scores. Random Forest has the best cross-validation score on recall/sensitivity, with an average of 0.810. XGboosting had the best cross-validation score on AUROC, with an average of 0.929. The accuracy on test data for XGboosting and Random Forest are 0.866 (95% confidence interval: 0.859, 0.873) and 0.822 (0.813, 0.829) respectively. Recall/sensitivity is 0.823 (0.815,0.831) for XGboosting and 0.805 (0.797, 0.813) for Random Forest. Specificity is 0.872 (0.865, 0.879) for XGboosting and 0.824 (0.816, 0.832) for Random Forest. Precision is 0.462 (0.451, 0.472) for XGboosting and 0.379 (0.369, 0.388) for Random Forest. AUROC is 0.925 (0.919, 0.930) for XGboosting and 0.898 (0.891, 0.904) for Random Forest.

Conclusion:
The Random Forest classifier has decent overall scores in all 4 metrics (accuracy, recall, specificity, AUROC) on test data by default cut-off value (0.5), especially in the recall. This can be particularly useful to enhance the campaign success rate. However, we can lower down the cut-off value from 0.5 to 0.1, then XGboosting can achieve a better performance from all aspects than Random Forest.

Introduction

Direct marketing and mass marketing are widely used in different industries, including banking, insurance, and retailing, to promote products and services. Implementing a mass marketing strategy in highly competitive sectors can result in a very low response rate (< 1%), it takes more resources and therefore more expensive (Moro et al., 2011). Direct marketing aims to improve the response rate and utilize the resource more efficiently than mass marketing (Moro et al., 2014). Enterprises use a data-driven process to study customer behaviors, then target the group of customers who are most likely to respond to the campaigns (Miguéis et al., 2017). It is worth mentioning that a 1% boost in response rate can generate an additional profit of €500,000 (Baesens et al., 2002). Therefore, our objective is to develop classifiers that can be used to predict customers with a high probability of subscription to direct marketing campaigns in a data-driven approach.

Data

Data Description

The data is related to direct marketing campaigns of a Portuguese banking institution (Moro et al., 2011). Bank agents are asked to contact the client through phone calls, and usually, the same client will be contacted more than once. The purpose is to assess if the product (bank term deposit) would be subscribed or not. The file contains all examples from May 2008 to November 2010 ordered by date. There are 45211 instances and 17 attributes including the output attribute.

17 attributes including 7 numeric variables which are: *age*, *balance*, *day*, *duration*, *campaign*, *pdays* (-1 means client was not previously contacted), and *previous*. The rest of the attributes are categorical features including the type of job contains 12 categories, marital status contains 3 categories, education contains 4 categories, contact communication type contains 3 categories, last contact month of the year contains 12 categories and outcome of the previous marketing campaign contains 4 categories, and the rest four features are binary features including *default*, *housing*, *loan*, and lastly the output variable *y* (Moro et al., 2011). More details are described in Table I.

Data Preprocessing

All data pre-processing and analyses were done in python 3.8 (Hilpisch, 2015). There are no missing values in this dataset. No features were dropped before modeling, and we performed feature selection in a data-driven fashion. However, some values are extreme in certain variables such as balance and duration. For example, some clients have an extremely high balance; some phone call duration is extremely long. Thus, we will standardize features by removing the mean and scaling to unit variance before applying Logistic Regression. The effect of standardization is shown in Table II. Another problem we encountered was a mass of "unknown" values in *poutcome* (previous marketing campaign), *job, education,* and *contact* since the clients were not contacted previously. As a result, we decided to treat "unknown" as a level in these categorical variables instead of missing values.

There were 9 categorical variables excluding the output variable. We used a function in the pandas package to obtain dummy variables (Nelli, 2015). We used the option to drop the first level so that we obtained k-1 dummy variables for each categorical feature with k levels. Eventually, we have 42 variables after preprocessing.

Data Splitting

We split 80% of our data into the training set to fit models and the rest 20% of the data into the testing set to estimate the predictions. Our training dataset has 36168 rows of data and our testing dataset has 9043 rows of data. We used 10-fold cross-validation on the training dataset to fit each model. The selection criteria were based on the recall/sensitivity and area under the curve (AUROC). We are going to select one model for each criterion and then do evaluations based on the model performance on the testing data set.

Methods

For our objective, we chose three models: Logistic Regression, XGboosting, and Random Forest, plus a Dummy Classifier to compare with. We chose Logistic Regression because it deals with binary output variables. XGboosting is chosen because it performs well in imbalanced datasets. Bagging reduces the variance in decision trees; however, trees are highly correlated with each other. Thus, it is not the optimal method and we decided to use Random Forest because it highly reduces the variance in decision trees and correlations between trees at the same time. The packages used for each model can be found in Table III

Logistic Regression: We tried different penalties to obtain the best model. Additionally, we compared two solvers, one is sklearn's Limited-memory Broyden Fletcher Goldfarb Shanno (lbfgs) solver and the other is Newton's method solver. We eventually decided to implement the final model using ridge regression (l2 penalty) and "lbfgs" solver with penalty strength c equal to 5.0.

XGboosting: Secondly, we used the XGboosting method because it performs well even on imbalanced datasets. We decided to use a "binary:logistic" learning task, 300 boosting rounds (estimators) with a maximum depth of 10, and a learning rate of 0.1. The random state was set to 100 to ensure reproducible results.

Random Forest: Random Forest method corrects the problem of overfitting training sets which are often caused in decision trees. It firstly constructs multitudinous decision trees at training time. After that, it returns the class that is the mode of the classification of the individual trees (Ho, 1998). We decided to use 500 estimators with a maximum of 3 features shown at each split and no restriction on maximum depth. Out-of-bag scores were used to estimate the general accuracy. The minimal impurity decrease was set to 0.01%. The random state was set to 100 to ensure reproducible results.

Dummy Classifier: Dummy Classifier was used to serve as a baseline to the above three classifiers. It predicted the most frequent class in the data set. There were more negative examples in both our training set and testing set; thus, the negative prediction is favored in the Dummy Classifier.

Evaluation Metrics

For each of the models we used above, we calculated the accuracy, precision, recall/specificity, and area under the curve (AUROC). The best model will be selected by comparing the performances of each model with these evaluation metrics. We will focus primarily on recall because we want to cover more truly positive cases to increase the response rate the most. We also consider precision as an important metric if our budget is limited. The accuracy will not be the main focus because our dataset is highly imbalanced. Additionally, we constructed a 95% confidence interval for each model by applying the central limit theorem since the size of our dataset is relatively large.

Results

According to the cross-validation, all of our three models (Logistic Regression, XGboosting, Random Forest) have over 0.9 AUROC scores, 0.9083, 0.9295, 0.9035 respectively. However, the difference between recall scores of the cross-validation is more noticeable, 0.3502, 0.4929, 0.8192 respectively. To narrow the candidate of our model, we decided to use the model with the highest score (XGboosting and Random Forest) from the two cross-validation methods for our further experiment.

After training the selected two models (XGboosting and Random Forest) on the training dataset, and test on the testing dataset, we calculated the accuracy, recall, precision, specificity as our performance metrics of each model. The details of the results are shown in Table IV. Our XGboosting model with default decision threshold doesn't perform well as the recall is 0.472; however, when we decrease the decision threshold to 0.1, it has better performance: accuracy (XGboosting: 0.866 vs Random Forest: 0.822), recall (0.823 vs 0.805), precision (0.462 vs 0.379), specificity (0.872 vs 0.824), and AUROC (0.924 vs 0.897). The confusion matrix and ROC plot of XGboosting are shown in Table V. The confusion matrix, ROC plot, and feature importance plot of the Random Forest are shown in Table VI.

The 95% confidence interval of the models is very narrow. Details are shown in Table VII. For the XGboosting model, CI of accuracy is [0.859,0.873], CI of recall is [0.815, 0.831], CI of specificity is [0.865, 0.879], CI of precision is [0.451, 0.472], and CI of AUROC is [0.9186, 0.9296]. For the Random Forest model, CI of accuracy is [0.813,0.829], CI of recall is [0.797, 0.813], CI of specificity is [0.816, 0.832], CI of precision is [0.369, 0.389] and CI of AUROC is [0.8913, 0.9038].

Discussion

We develop a predictive model to predict whether a client would subscribe to a term deposit by giving the account info and promoting history. The model is expected to be applicable to increase response rate in direct marketing campaigns with optimal budget management for the banking industry.

According to the results of our models, classifiers based on decision tree algorithm generally have better performance than Logistic Regression. The main reason could be there are too many features in our dataset and there could be some relationship between each feature that makes them not independent. For example, the number of days that passed since the last contacted “pdays” is -1 (not previously contacted) would result in “poutcome” to be “unknown”. The Logistic model requires independent predictors and there should be a linear relationship between the predictors and the response variable. However, in our data, both of the assumptions are violated, which makes the Logistic Regression have a bad performance in our case.

As for XGboost and Random Forest classifiers, XGboosting has slightly better performance in our case. Compare the mechanism of these two classifiers, they are both a set of decision trees. However, XGboosting builds one tree at a time and combines results along the way while Random Forest builds each tree independently and combines results at the end of the process. Both of them have advantages in dealing with multiple features and a large volume of data, but XGboosting is better at dealing with a non-noisy dataset like our case.

Finally, from the metrics, our model can easily achieve 80% recall but the precision (0.462) is not as good as the recall. There’s always a trade-off between precision and recall in real cases, while precision measures the portion of predicted positives that are truly positive and recall measures the portion of true positives that are correctly predicted (Moro et al., 2011). Low precision indicates the clients tagged positive by our model have a greater chance to not subscribe to our promotion. Increasing the decision threshold can significantly improve the precision but lower the recall which means the bank may miss many potential subscribers. Another way to improve precision is to add more positive data to the training dataset. As only about 10% of our training data is tagged as “positive”, adding more positive data would help with balancing the dataset and make the metrics score much better.

Limitation and Future Works

We are going to discuss the limitations of our study from two aspects. One problem is our obsolete data from May 2008 to November 2010. This can lead to inaccurate predictions. Including recent data will improve the performance of our classifiers. The second limitation is our data is heavily imbalanced. We may use under-sampling and over-sampling methods in future studies to achieve better model performance (Miguéis et al., 2017).

References

Baesens B., Viaene S., Poel D. V., Vanthienen J., & Dedene G. (2002). Bayesian neural network learning for repeat purchase modelling in direct marketing. *European Journal of Operational Research, 138*(1), 191–211. https://doi.org/10.1016/S0377-2217(01)00129-1

Hilpisch, Y. (2015). Derivatives Analytics with Python: Data Analysis, Models, Simulation, Calibration and Hedging. In *Derivatives Analytics with Python* (1st ed.). John Wiley & Sons, Incorporated.

Ho, T. K. (1998). The Random Subspace Method for Constructing Decision Forest. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 20 (8): 832-844. doi: 10.1109/34.709601

Miguéis V. L., Camanho A. S., & Borges J. (2017). Predicting direct marketing response in banking: comparison of class imbalance methods. *Service Business, 11*(4), 831–849.   https://doi.org/10.1007/s11628-016-0332-3

Moro S., Laureano R. M., & Cortez P. (2011). Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. *In P. Novais et al. (Eds.), Proceedings of the European Simulation and Modelling Conference - ESM'2011*, pp. 117-121, Guimarães, Portugal, EUROSIS.

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

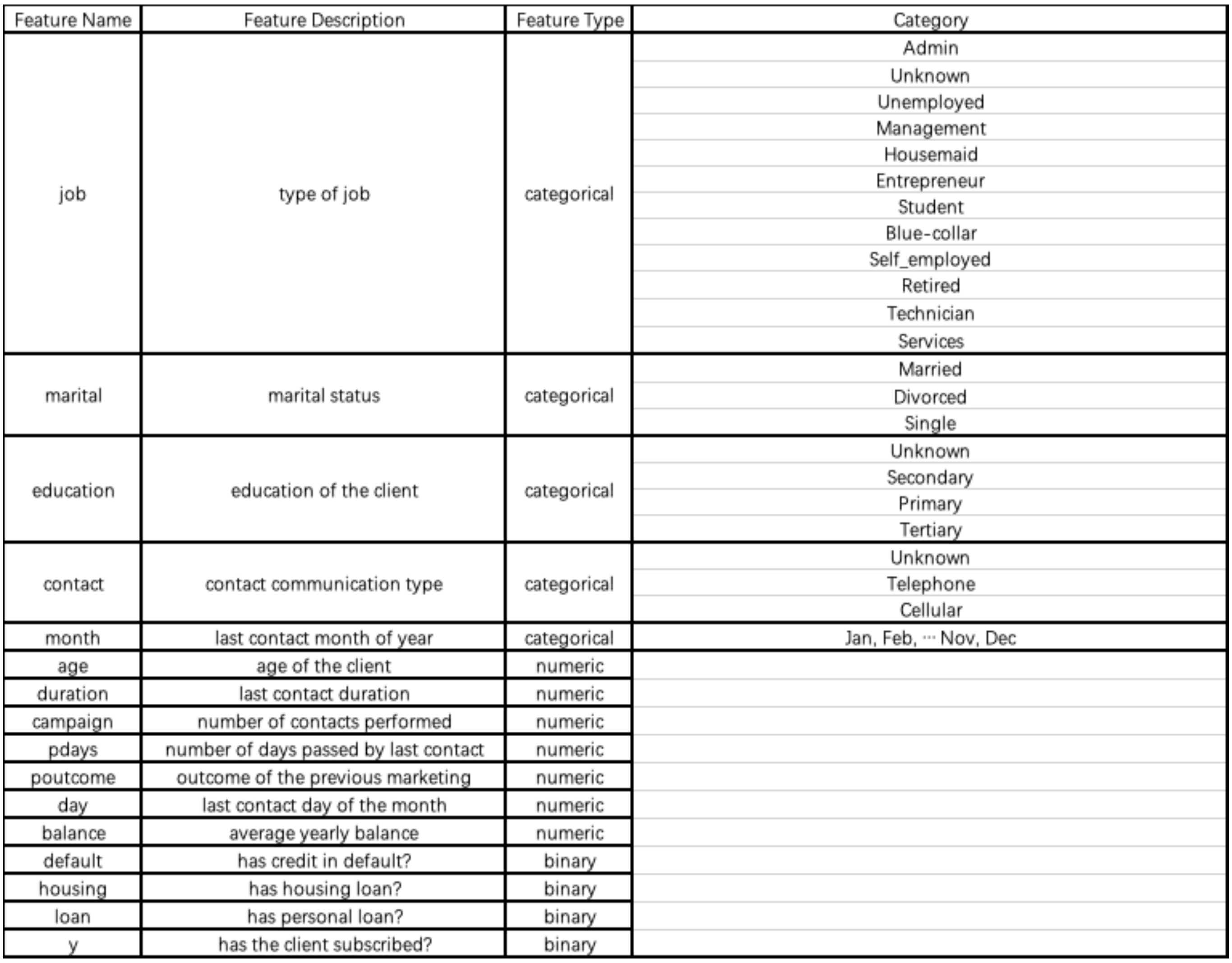
Nelli, F. (2015). Python Data Analytics: Data Analysis and Science Using Pandas, Matplotlib and the Python Programming Language. In *Python Data Analytics* (1st ed.). Apress L. P.

Trappenberg, T. (2019). Machine learning with sklearn. In *Fundamentals of Machine Learning*. Oxford University Press. https://doi.org/10.1093/oso/9780198828044.003.0003

Wang, D. (2020). Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost. *Pattern Recognition Letters, 136,* 190–197. https://doi.org/10.1016/j.patrec.2020.05.035

Appendix

1. Data Description



Feature Name	Feature Description	Feature Type	Category
job	type of job	categorical	Admin
			Unknown
			Unemployed
			Management
			Housemaid
			Entrepreneur
			Student
			Blue-collar
			Self_employed
			Retired
			Technician
			Services
marital	marital status	categorical	Married
			Divorced
			Single
education	education of the client	categorical	Unknown
			Secondary
			Primary
			Tertiary
contact	contact communication type	categorical	Unknown
			Telephone
			Cellular
month	last contact month of year	categorical	Jan, Feb, ··· Nov, Dec
age	age of the client	numeric	
duration	last contact duration	numeric	
campaign	number of contacts performed	numeric	
pdays	number of days passed by last contact	numeric	
poutcome	outcome of the previous marketing	numeric	
day	last contact day of the month	numeric	
balance	average yearly balance	numeric	
default	has credit in default?	binary	
housing	has housing loan?	binary	
loan	has personal loan?	binary	
y	has the client subscribed?	binary	

Table I: Description of Variables

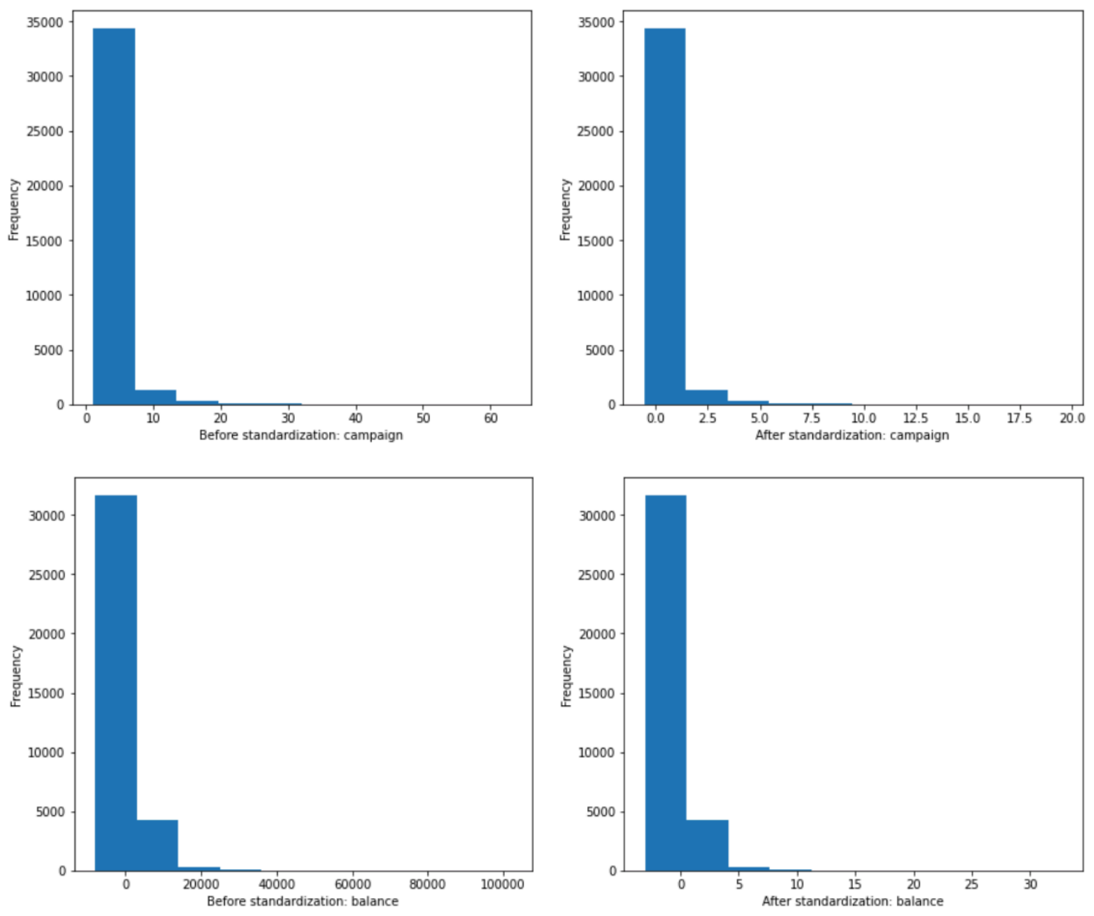


Table II: Effect of Standardization

2. Package Used

Model	Package Used
Logistic Regression	sklearn.linear_model.LogisticRegression
XGboosting	xgboost.XGBClassifier
Random Forest	sklearn.ensemble.RandomForestClassifier
Dummy Classifier	sklearn.dummy.DummyClassifier

Table III: Package Used

3. Model Performance Metrics

	CV						
Model	AUROC	Recall	Accuracy	Recall	Precision	Specificity	AUROC
Logistic	0.9083	0.3502					
XGboosting(cutoff = 0.5)	0.9295	0.4929	0.8980	0.4720	0.5800	0.9540	0.9259
XGboosting(cutoff = 0.1)			0.8660	0.8230	0.4620	0.8720	0.9241
Random Forest	0.9035	0.8192	0.8220	0.8050	0.3790	0.8240	0.8980

Table IV: Performance Metrics

	Predicted Positive	Predicted Negative
Observed Positive	502	364
Observed Negative	561	7616

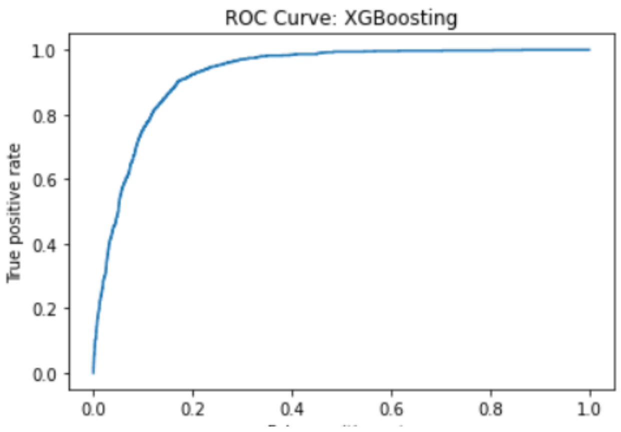
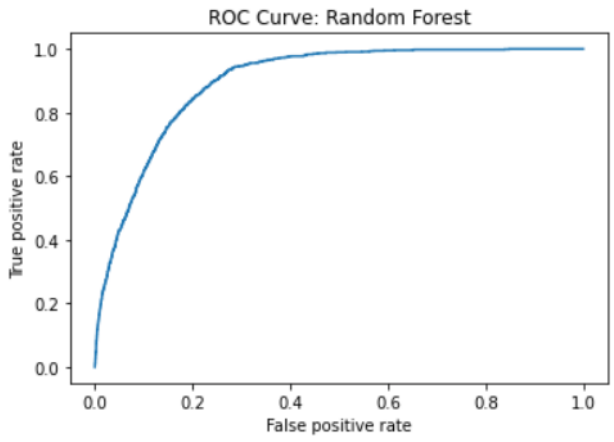


Table V: Confusion Matrix and ROC Plot of XGboosting

	Predicted Positive	Predicted Negative
Observed Positive	856	1404
Observed Negative	207	6576



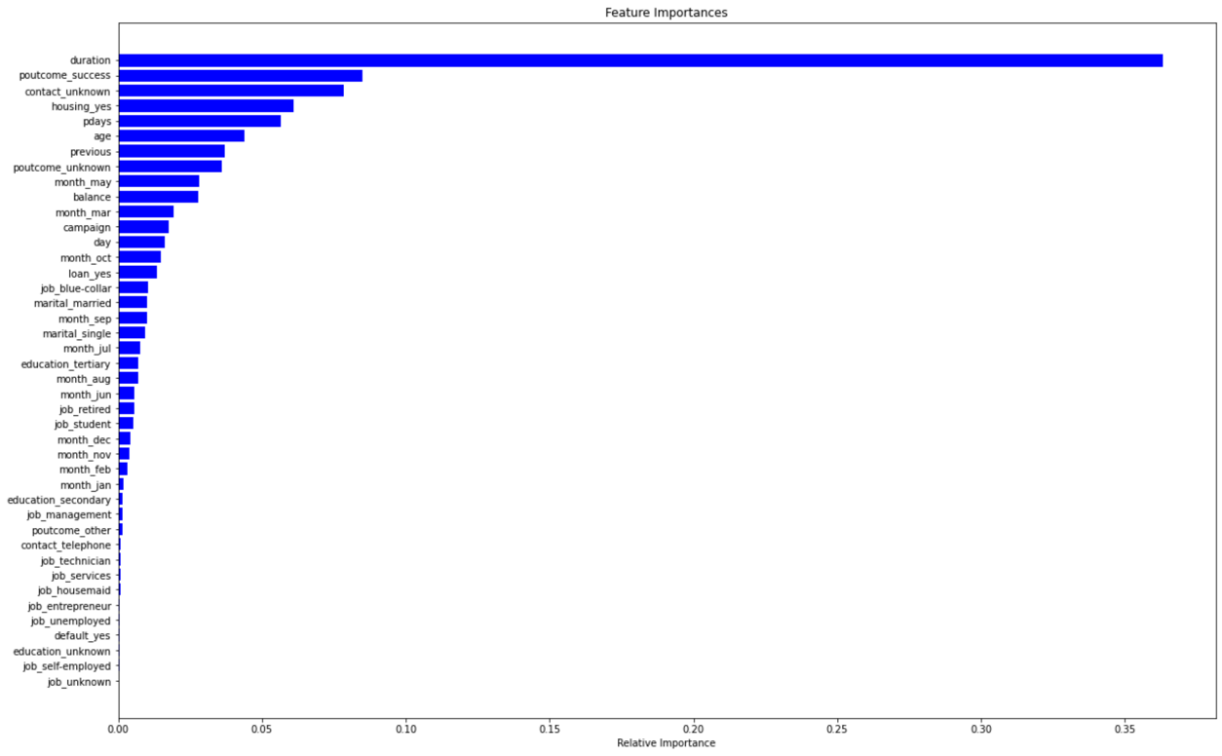


Table VI: Confusion Matrix ROC Plot and Feature Importance Plot of Random Forest

	95% Confidence Interval				
Model	**Accuracy**	**Recall**	**Specificity**	**Precision**	**AUROC**
XGboosting(cutoff = 0.1)	[0.859, 0.873]	[0.815, 0.831]	[0.865, 0.879]	[0.451, 0.472]	[0.9186, 0.9296]
Random Forest	[0.813,0.829]	[0.797, 0.813]	[0.816, 0.832]	[0.369, 0.389]	[0.8913, 0.9038]

Table VII: Confidence Interval