DIRECT
MARKETING
CAMPAIGN

June 16, 2021

Bank Marketing Analysis

Nicholas Sohn

Nicholas Sohn
(213)448-7090
sohnnick@gmail.com

501 W Olympic Blvd,
Los Angeles, CA 90015
www.nicksohn.com

I. ABSTRACT

Motivation

Data has never been so important in bank marketing. As part of the rapid technological shifts in banking, increased data collection yields three key benefits:

- Improve relationships between customers and traditional agencies.
- Empowering marketers with more information on customers.
- Boost social media presence and offer new channels for targeted ads.

Despite these changes towards digital advertising, telephone communication still stands as a critical component of B2B marketing and sales. As such, targeted telemarketing through data analytics has never been more important. This report outlines the most important customer features and creates a prediction model that will result in a term deposit for the bank.

Business Problem

The digital revolution has made bank marketing more competitive and finance companies must be able to adapt and find solutions to stay relevant. Marketers are not only pressed for time but face fierce competition from emerging fintech firms. As a result, bank managers and marketers must make swift decisions to keep up with new marketing techniques and reach the right audience.

Opportunity

Obtaining term deposits is pivotal for the success of finance companies. Term deposits allow financial institutions to hold onto a deposit for a specific amount of time and unlike the traditional savings account, it cannot be pulled out till its maturity. This enables banks to invest in higher gaining financial products or make loans to clients and customers with interest. Simply put, the larger the term deposit, the more revenue the bank makes, and in turn, the larger the profit. Consequently, identifying significant characteristics for customers, and prediction algorithms, can help both the efficiency of marketing efforts and increase revenue gains for the bank. A successful predictor can minimize time challenges and improve decision-making for bank marketers.

II. DATASET

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rate	cons
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
2	37	services	married	high.school	no	yes	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1	
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999	0	nonexistent	1.1	

The dataset is composed of **41,188 rows and 21 columns**. Every row describes an interaction with a client, and oftentimes, more than one contact with the same client was required. Each column represents a descriptive characteristic of the client interaction. Of those 21 columns, there is one response variable: “y” - whether or not the client has subscribed to the term deposit.

Bank Client Data	Last Contact	Other	Social and Economic Context
age job marital education default housing loan	contact month duration	campaign pdays previous poutcome	emp.var.rate cons.price.idx cons.conf.idx euribor3m nr.employed

The independent variables can be split into four segments: **bank client data, related with the last contact of the current campaign, other attributes, and social and economic context attributes.**

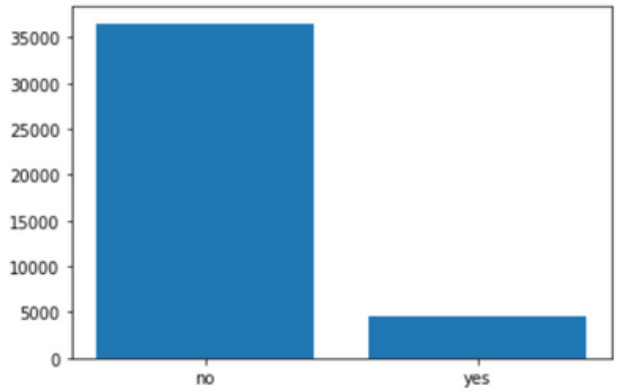
```
df_bank.isnull().sum()
```

```
age               0
job               0
marital           0
education         0
default           0
housing           0
loan              0
contact           0
month             0
day_of_week       0
duration          0
campaign          0
pdays             0
previous          0
poutcome          0
emp.var.rate      0
cons.price.idx    0
cons.conf.idx     0
euribor3m         0
nr.employed       0
y                 0
dtype: int64
```

Pandas methods reveal that the the dataset has no null values and therefore, does not require imputing missing values.

III. EXPLORATORY DATA ANALYSIS

Frequency by Class



A simple bar chart reveals that there are **36,548 "no" and 4,640 "yes"** in terms of term deposits. This suggests that the dataset is imbalanced and may be subject to inaccuracies in terms of true positive rates (the probability that an actual yes will be predicted yes).

Summary Statistics

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

Exploring further, we look at the summary statistics for numeric variables regarding mean, standard deviations, minimums, maximums, and quartiles. Out of the ten continuous input variables, duration, pdays, and nr.employed showcase the highest standard deviation, and thus have the most variability.

When looking into the dataset documentation, we observe that the number of days that passed by after the client was last contacted from a previous campaign, denoted by pdays, has 999 when a client has never been contacted. Thus its high variability may largely be accounted for by an unrepresentative input method. This will be further explained in the data encoding portion.

III. EXPLORATORY DATA ANALYSIS

Summary Statistics by Class

y	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
no	39.911185	220.844807	2.633085	984.113878	0.132374	0.248875	93.603757	-40.593097	3.811491	5176.166600
yes	40.913147	553.191164	2.051724	792.035560	0.492672	-1.233448	93.354386	-39.789784	2.123135	5095.115991

In terms of the mean values between the classes (yes or no), we can make preliminary hypotheses regarding the impact of specified on the outcome. A larger difference in means between the “yes” and “no” class suggests that the feature may be a significant in determining the outcome of the term deposit.

Observation 1: There appears to be a significant difference in duration means between the “yes” and “no” classes.

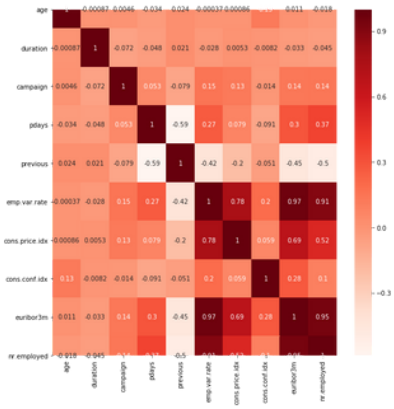
Hypothesis 1: A client who is interested in the financial institution will be on call for longer, perhaps to inquire and gain information about benefits and risks associated with a term deposit. As such, I hypothesize that **duration of the call has a significant impact on the term deposits.**

Observation 2: There appears to be a significant class difference in emp.var.rate means, relative to the overall standard deviation.

Hypothesis 2: Employment variation rate refers to the number of employees being hired and fired, a reflection of the overall economy. When the economy is in a recession, both individuals and institutions are more conservative with investments. Hence, I hypothesize that the **employment variation rate has a significant impact on the term deposits.**

III. EXPLORATORY DATA ANALYSIS

Correlation Heat Map

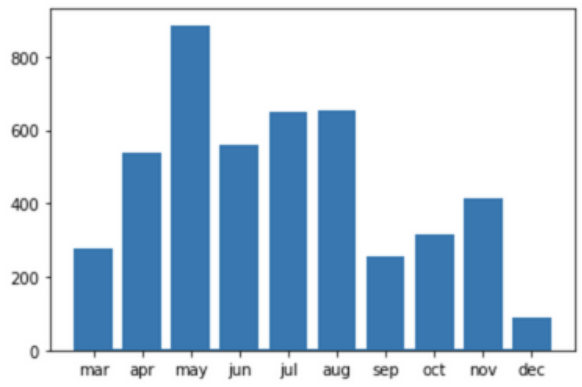


As part of the traditional data analysis pipeline, it is important to look at pairwise correlations between variables. Correlation indicates that as one variable changes, the other variable tends to change in a specific direction.

I have outlined a heatmap, showcasing the correlation between the continuous variables. It appears that the **Euribor 3 month rate, denoted euribor3m, is highly correlated** with nr.employed (coef = 0.95) and emp.var.rate (coef = 0.97).

Term Deposits Frequency by Month

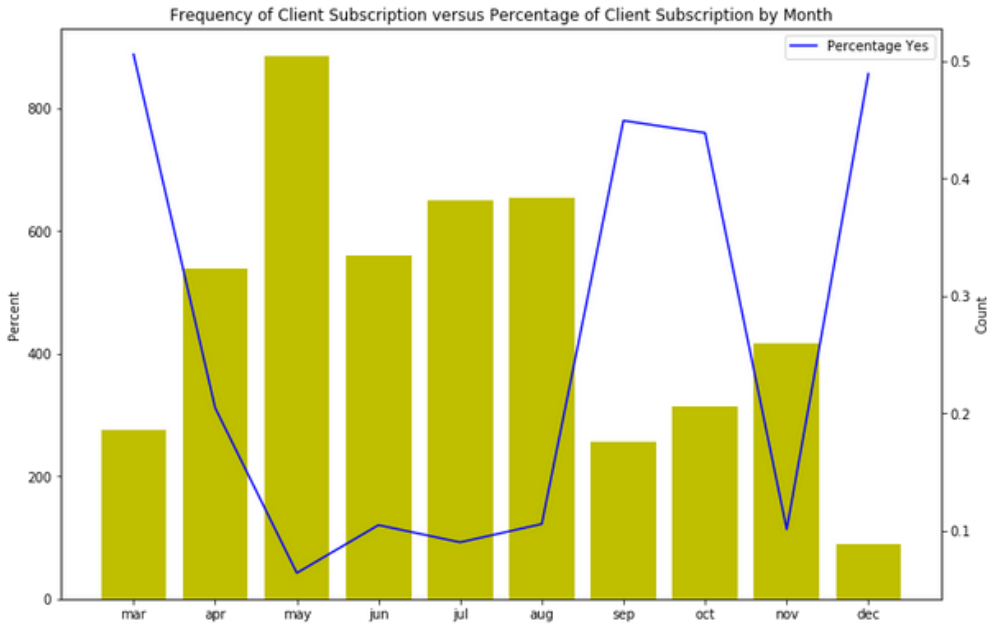
Revenues for most industries are cyclical and seasonal and banks are no exception. Therefore, it is important to plot the number of successful term deposits with respect to the month.



The **peak month for "yes" term deposits from clients occur in May**, whereas the lowest "yes" term deposits occurred in December. In summary, the highest volume of "yes" term deposits take place around late Spring to early Summer and the lowest volume of "yes" term deposits occur in the Winter.

III. EXPLORATORY DATA ANALYSIS

Term Deposit Frequency and Proportion by Month



Despite high volumes of “yes” term deposits occur around late Spring, when plotted against the proportion of “yes” over the total number of calls made that month, we see a different story. **The lowest percentage of “yes” occurs in May, where we previously observed the highest volume of “yes” term deposits.**

Similarly, the highest percentage of “yes” term deposits occur in December and March, the lowest two months in terms of “yes” volume. As a bank marketer, time is strapped and efficiency is crucial. Therefore, looking into defining features that yield the highest probability of a “yes” classification is a more effective metric to analyze.

IV. ENCODING CATEGORICAL VARIABLES

Since Python machine learning algorithms cannot interpret categorical data, I used integer encoding to map the categories with its respective numeric value.

Encoding Ordinal Variables

Ordinal variables are similar to categorical variables in that they are discrete and not numeric. For instance, the number of contacts performed during the campaign is a numeric variable since it is quantifiable whereas marital status is categorical since it can't be performed with mathematic calculations such as mean. **Ordinal variables, however, are categorical variables with an order or a hierarchical structure**, such as month and day of the week.

I created a mapping such that March, the earliest month, is encoded 0, February is encoded 1, so on and so forth. Similarly, I mapped days of the week such that Monday is 0, Tuesday is 1, etc.

Earlier, I observed that for the pdays feature, "999" represented clients who have not been contacted previously. This may throw off the distribution of the continuous variables. Therefore, I categorized the pdays into five intervals:

Interval / Category	Integer Code
never contacted	0
0 - 6	1
7 - 13	2
14 - 20	3
21 - 27	4

Encoding Nominal Variables

Nominal variables are discrete variables that do not have any particular order. Therefore I utilized the enumerate function in order to integer code. The figure below outlines each nominal variable and their corresponding integer values:

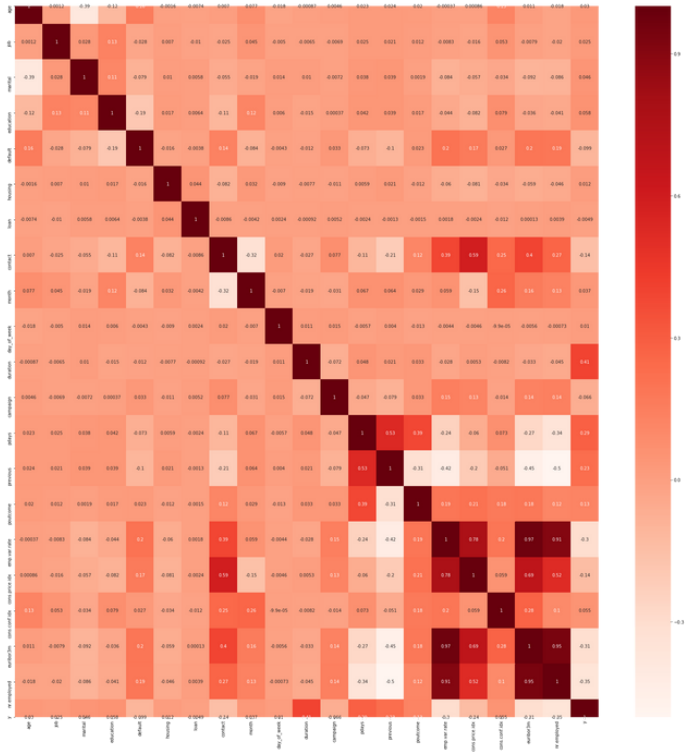
```
{'job': {0: 'admin.',
  1: 'blue-collar',
  2: 'entrepreneur',
  3: 'housemaid',
  4: 'management',
  5: 'retired',
  6: 'self-employed',
  7: 'services',
  8: 'student',
  9: 'technician',
  10: 'unemployed',
  11: 'unknown'},
 'marital': {0: 'divorced', 1: 'married', 2: 'single', 3: 'unknown'},
 'education': {0: 'basic.4y',
  1: 'basic.6y',
  2: 'basic.9y',
  3: 'high.school',
  4: 'illiterate',
  5: 'professional.course',
  6: 'university.degree',
  7: 'unknown'},
 'default': {0: 'no', 1: 'unknown', 2: 'yes'},
 'housing': {0: 'no', 1: 'unknown', 2: 'yes'},
 'loan': {0: 'no', 1: 'unknown', 2: 'yes'},
 'contact': {0: 'cellular', 1: 'telephone'},
 'poutcome': {0: 'failure', 1: 'nonexistent', 2: 'success'},
 'y': {0: 'no', 1: 'yes'}}
```

V. FEATURE SELECTION

Curse of Dimensionality

The curse of dimensionality refers to a set of problems that arise from datasets with a high number of features. Datasets with a large number of features are difficult to fathom from a business intelligence perspective and difficult to make decisions with. In addition, computational processes take longer. Most importantly, however, is that datasets with more features are more likely to overfit. Predictors must perform well on both the existing datasets (training) as well as unknown or new datasets (test). With high dimensional datasets, models are more likely to perform exceedingly well on existing datasets but perform poorly on new datasets. In data analytics, we seek to find a healthy balance of both.

Drop Correlated Features



During the data exploration portion, I came across a highly correlated feature: euribor3m. Features with high correlation to other features are deemed redundant and contributes to the aforementioned curse of dimensionality. As such, I simply removed the euribor3m variable from the dataset.

V. FEATURE SELECTION

Univariate Feature Selection

Categorical features rely on the Chi-Square statistic to test independence. A low Chi-Square value indicates that the term-deposit outcome is independent from the specified explanatory variable. A high Chi-Square value indicates that the feature is not independent from the term-deposit outcome, and thus a significant predictor of the given binary output.

	Feature	Chi2	P-Val
0	job	78.441531	8.240144e-19
1	marital	19.384622	1.068641e-05
2	education	128.643391	8.116637e-30
3	default	248.289367	6.128837e-56
4	housing	7.249350	7.092667e-03
5	loan	2.532007	1.115582e-01
6	contact	429.340446	2.261727e-95
7	poutcome	70.423866	4.783767e-17
8	day_of_week	3.453692	6.311065e-02
9	month	36.396948	1.609551e-09
10	pdays	3883.537814	0.000000e+00

Through the univariate feature selection technique, I select the six best categorical features based on the Chi-Square metric. The six features with the highest significance are **job, education, default, contact, poutcome, and pdays**. We will filter the dataset to remove categorical features outside of this subset.

Variance Threshold Feature Selection

Variance threshold is a baseline feature selection technique. By default, it removes all features with zero variance. Features with higher variance tend to contain more useful information whereas features with lower variance tend to showcase low predictive power.

Of the eight continuous variables used to train the Variance Threshold function, six features variances above the 0.8 threshold: **age, duration, campaign, emp.var.rate, cons.conf.idx, and nr.employed**. Whereas features previous and const.price.idx revealed variance lower than 0.3. I removed the features with variances lower than 0.8.

VI. ANALYSIS

No Free Lunch Theorem

The No Free Lunch Theorem states that all optimization algorithms perform equally when their performance is averaged across all problem sets. In short, it means that **there is no single best optimization algorithm.** Therefore, it is always best to utilize a variety of prediction algorithms, compare the outcomes, and evaluate the best model(s) to use. As data scientists, it is not feasible to apply all possible algorithms to a dataset, and thus, data scientists must make assumptions about which algorithm to use. In this report, I utilized three common classification predictors: **Naive-Bayes Classifier, Logistic Regression Classifier, and Decision Tree Classifier.**

Metrics for Success

Accuracy is a common metric used to measure predictive power. It is calculated by simply dividing the number of correctly classified data points over the number of instances in the dataset. However, accuracy can be misleading as it does not provide sufficient information on positive and negative rates.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Precision is defined as the number of true positives divided by the number of true positives and false positives. False positives are cases where the model labels the instance positive when in actuality, it is negative. In summary, it is the proportion of correctly identified positives divided by all predicted positives.

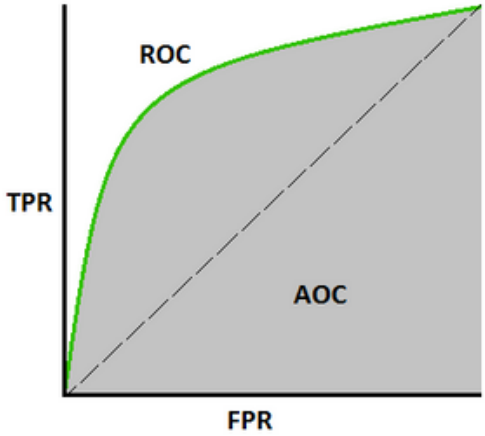
Recall is defined as the number of true positives divided by the number of true positives and false negatives. False negatives are cases where the model labels the instance negative when in actuality, it is positive. In summary, it is the proportion of correctly identified positives divided by all actual positives.

Recall is an important measure for medical data where a test for a disease should minimize false negatives (or maximize recall). For instance, if you test negative for COVID-19 but actually have the virus, it will do more harm to society.

VI. ANALYSIS

Metrics for Success (cont.)

In the case of bank term deposits, we want to maximize precision or minimize false positives. We want to correctly identify the clients who are most likely to make term deposits. If we predict an individual to be in the "yes" category, only to reveal that they were unwilling to make a term deposit, then it would be a waste of both time and money for the bank.



In essence, the **AUC** tells us how much the model is capable of distinguishing between classes. The higher the AUC, the better the model predicts "no" term deposits as "no" and "yes" term deposits as "yes".

VI. ANALYSIS

Naive Bayes Classification

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

The Naive-Bayes classifier is a probabilistic machine learning model that is grounded on Baye's theorem and is used for a classification task. Let's explore a simple example for term deposits. Let's say we want to predict the probability that a client would make a term deposit given their marital status. We would calculate, for instance, the proportion of clients who are married given that they are in the "yes" class. Then we would apply Baye's theorem to obtain the probability that the client would make a term deposit given that they are married. After applying the Naive-Bayes algorithm to the dataset, I obtained these performance metrics:

Test Accuracy: 0.8844
Train Accuracy: 0.8795

We actually see something that is fairly uncommon in data analytics, a test accuracy that is higher than the training accuracy. This means the algorithm will perform better on new information than existing information.

	precision	recall	f1-score	support
0	0.95	0.92	0.93	9141
1	0.47	0.58	0.52	1156
accuracy			0.88	10297
macro avg	0.71	0.75	0.72	10297
weighted avg	0.89	0.88	0.88	10297

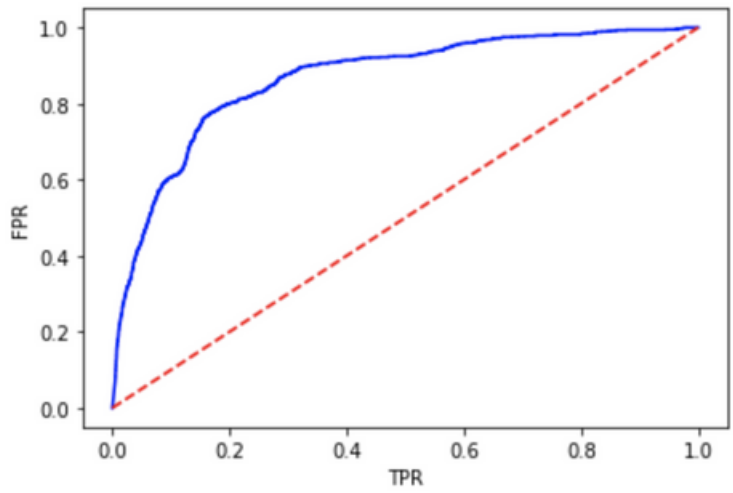
When taking a deeper look, we see that the precision for "no" is quite high but the precision for identifying "yes" is lower than 50%. Our aim is to maximize precision for "yes", thus this model is not an effective predictor with respect to the business goals of the bank.

VI. ANALYSIS

Naive Bayes Classification (Cont.)

```
AUC: 0.8663673195295994

Text(0, 0.5, 'FPR')
```



The AUC curve, with a value of 0.8664, reveals that **the model is overall able to distinguish between the term deposit "yes" classes and "no" classes**. Nonetheless, the Naive-Bayes classifier faces several limitations. The first being that the algorithm does not handle classification with continuous and categorical input variables well. Secondly, the algorithm assumes that features are independent of each other, which is likely not the case in the natural world. Lastly, Naive-Bayes does not offer an intrinsic method to evaluate feature importance, a critical component in understanding the characteristics of clients who are likely to make term deposits. Despite these restraints, it sets a benchmark for model comparison when applying different algorithms.

VI. ANALYSIS

Logistic Regression

The Logistic Regression classifier is similar to the Naive-Bayes classifier. It is grounded on the linear regression function with the application of a sigmoid function. The key takeaway is that the logistic regression function contains both feature importance as well as a probability assigned to “yes” or “no” term deposits. If the probability is greater or equal to 0.5, then it is assigned a “yes”.

Logistic Regression (SMOTE)

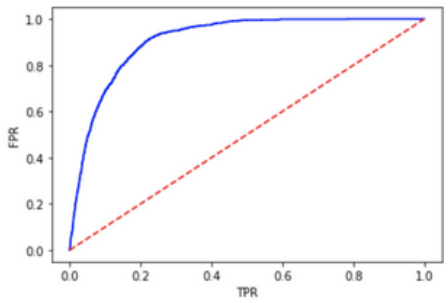
Previously we noticed that there is a significantly larger sample of “no” than “yes”. This may cause dataset imbalances and consequently, yield poor performance for the minority class. Therefore, I applied the Synthetic Minority Oversampling Technique (SMOTE). **SMOTE synthesizes new examples for the minority class.** After applying the Naive-Bayes algorithm to the dataset, I obtained these performance metrics:

Test Accuracy: 0.8289
Train Accuracy: 0.8296

	precision	recall	f1-score	support
0	0.98	0.83	0.90	9141
1	0.38	0.84	0.52	1156
accuracy			0.83	10297
macro avg	0.68	0.83	0.71	10297
weighted avg	0.91	0.83	0.85	10297

Despite relatively decent training and test accuracies, further analysis shows that the **precision for “yes” term deposits is exceedingly low at 0.38**. This is lower than the precision obtained from the Naive-Bayes classifier.

```
AUC: 0.9100831494589381
Text(0, 0.5, 'FPR')
```



The model AUC appears to be higher than the Naive-Bayes classifier at 0.91, indicating overall better performance when distinguishing outcome classes.

VI. ANALYSIS

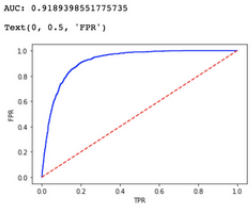
Logistic Regression (Without SMOTE)

We must compare the SMOTE technique performance with a dataset without SMOTE. After applying the Logistic Regression classifier to the non-SMOTE dataset, I obtained these performance metrics:

Test Accuracy: 0.9003
Train Accuracy: 0.8972

Again, we notice a unique case where the test accuracy is higher than the training accuracy. Furthermore, we obtain accuracy metrics improvements across the training and test dataset.

	precision	recall	f1-score	support
0	0.92	0.98	0.95	9152
1	0.61	0.28	0.38	1145
accuracy			0.90	10297
macro avg	0.76	0.63	0.66	10297
weighted avg	0.88	0.90	0.88	10297



The precision score for “yes” term deposits (0.61) has shown a significant increase. It has broken the 50% barrier mark that I have previously struggled to obtain. The AUC score of 0.9189 is greater than both the SMOTE model and the Naive-Bayes model.

From this, we realize that **SMOTE yielded lower-performing metrics including accuracy, precision, and AUC**. Therefore, I moved forward to my next prediction algorithm without any sampling techniques.

L1 and L2 Regularized Logistic Regression

Earlier in the report, I briefly explained how datasets are subject to overfitting, performing well on existing datasets but performing poorly on new datasets. **In order to mitigate overfitting, regularization techniques are utilized.** Loss functions are used to evaluate the actual outcome with respect to the predicted function. By altering the loss function, we are able to shrink the weight of certain features and minimize overfitting.

VI. ANALYSIS

L1 Logistic Regression

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |w_i|$$

In L1 Regularized Logistic Regression, we alter the loss function such that the absolute weight of a feature shrinks.

L2 Logistic Regression

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2$$

In L2 Regularized Logistic Regression, we alter the loss function such that the squared weight of a feature shrinks.

There is a parameter we must adjust: **lambda**. Lambda is a scalar that shrinks the absolute weight of features by altering the loss function hyperparameter. In order to find the optimal lambda, I conducted 5-fold cross-validation on the training set. Reported below are the average test errors based on the lambdas:

```
KFold(n_splits=5, random_state=None, shuffle=False)
K = 1e-05 :  0.8917483958002468
K = 0.0001 :  0.8929785097457522
K = 0.001 :  0.8948884029601197
K = 0.01 :  0.9020102498405846
K = 0.1 :  0.904438130991305
K = 1 :  0.9078047382263812
K = 10 :  0.9086464188506784
K = 100 :  0.9088082520966833
K = 1000 :  0.9086140092401445
K = 10000 :  0.9084198082971012
K = 100000 :  0.9086140459144533
```

Optimal Lambda: 100
Test Accuracy: 0.9139
Train Accuracy: 0.9092
Term-Deposit Precision: 0.7
AUC: 0.929

```
KFold(n_splits=5, random_state=None, shuffle=False)
K = 1e-05 :  0.8938848995681916
K = 0.0001 :  0.8948560509806421
K = 0.001 :  0.897963774270468
K = 0.01 :  0.9029813855354746
K = 0.1 :  0.9039525421871122
K = 1 :  0.9036935534581227
K = 10 :  0.9038554233784364
K = 100 :  0.903725931633535
K = 1000 :  0.9039201744900739
K = 10000 :  0.9038230556813979
K = 100000 :  0.9038554286176232
```

Optimal Lambda: 1000
Test Accuracy: 0.9035
Train Accuracy: 0.9036
Term-Deposit Precision: 0.64
AUC: 0.9231

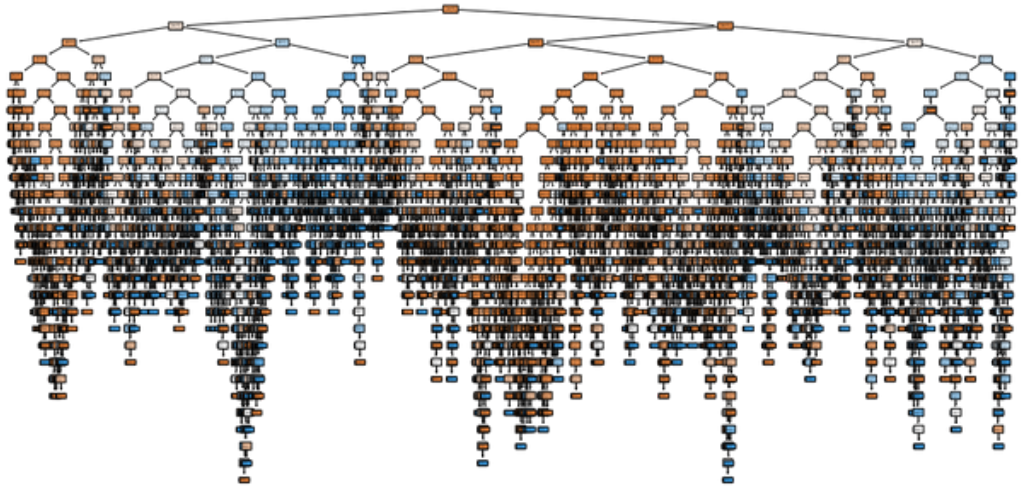
We see that the **clear winner thus far is L1 Logistic Regression**. It beats out every prior classifier in terms of test accuracy, training accuracy, and term-deposit precision. The only exception is AUC, which L2 Logistic Regression scored higher on. Despite this, we discovered that the regularization techniques were extremely successful in reducing overfitting and raising precision.

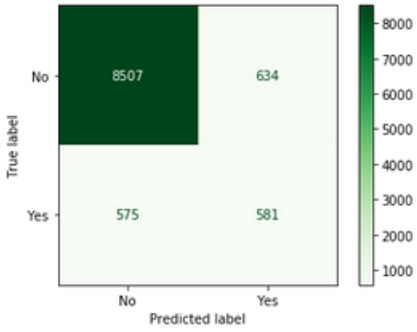
VI. ANALYSIS

Decision Tree Classifier

A decision tree can be used visually and explicitly to represent decisions and decision-making. It is often simpler to understand and relies on logical decisions to split nodes.

Decision trees are subject to overfitting and therefore, we must apply further feature selection. Recursive feature selection begins by obtaining p-values for all features. Then recursively dropping the highest p-value, or least significant, features. The selected features from this process are **job, education, contact, poutcome, age, duration, campaign, emp.var.rate, cons.conf.idx, and nr.employed**.





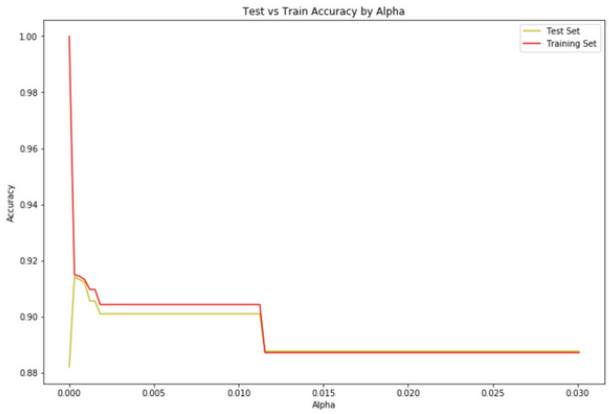
	precision	recall	f1-score	support
0	0.94	0.93	0.93	9141
1	0.48	0.50	0.49	1156
accuracy			0.88	10297
macro avg	0.71	0.72	0.71	10297
weighted avg	0.89	0.88	0.88	10297

Right off the bat, we notice that the **decision tree is essentially unfathomable** and contains too many rules. This is further supported by the discrepancy between the test accuracy (0.8826) and training accuracy (0.9999). In addition, this appears to be a poor fit model as the "yes" term-deposit precision rate is below 50%. This suggests we must reduce the size of the tree to create a more optimal model.

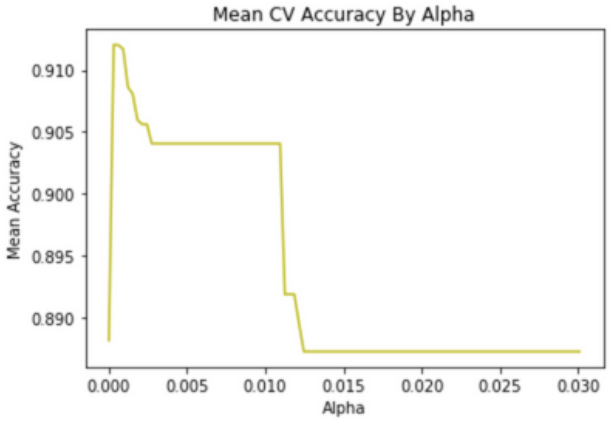
VI. ANALYSIS

Pruned Decision Tree Classifier

Similar to regularized logistic regression, we must alter the loss function with a parameter alpha such that a larger tree results in a larger loss. Finding the optimal alpha enables us to not only reduce the size of the tree but also reduce overfitting.



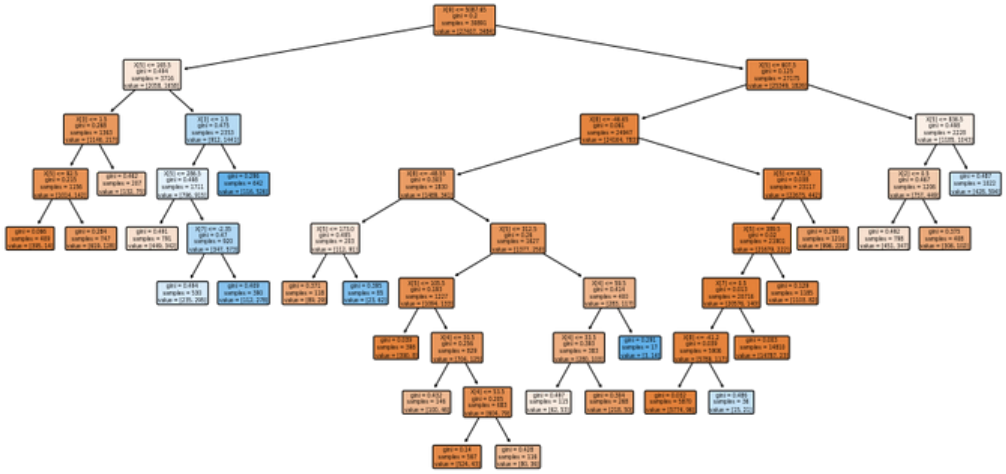
By plotting the train and test accuracies with respect to the hyperparameter alpha, we see that the train and test accuracies converge on the lower end of 0 to 0.005.



After applying 5-fold cross-validation on the training set, the highest test mean accuracy based on the **hyperparameter alpha is 0.0003**. When applying the decision tree classifier with the obtained hyperparameter, we see a simpler tree.

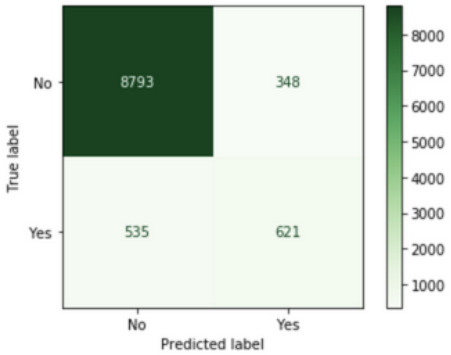
VI. ANALYSIS

Pruned Decision Tree Classifier (Cont.)



From a short glance, we can assume that this **model will be less overfit since it has fewer rules.** The metrics below support this case:

Optimal Alpha: 0.0003
Test Accuracy: 0.9142
Train Accuracy: 0.9149
Term-Deposit Precision: 0.64
AUC: 0.9328



	precision	recall	f1-score	support
0	0.94	0.96	0.95	9141
1	0.64	0.54	0.58	1156
accuracy			0.91	10297
macro avg	0.79	0.75	0.77	10297
weighted avg	0.91	0.91	0.91	10297

Overall, we see that we have reduced overfitting, increased term-deposit precision, raised AUC, and improved test accuracy by pruning.

VII. EVALUATION

Prediction Algorithm	Test Error	Train Error	Term-Deposit Precision	AUC
Naïve-Bayes	0.8844	0.8795	0.47	0.8664
Logistic Regression	0.9003	0.8972	0.61	0.9189
Logistic Regression w/ SMOTE	0.8289	0.8296	0.38	0.91
L1 Logistic Regression	0.9139	0.9092	0.7	0.929
L2 Logistic Regression	0.9035	0.9036	0.64	0.9231
Decision Tree	0.8826	0.9999	0.48	N/A
Pruned Decision Tree	0.9142	0.9149	0.64	0.9328

From this spreadsheet above, we see that every algorithm performed well in terms of AUC, meaning that the predictor effectively distinguishes between the "no" term-deposit and "yes" term-deposit.

With regards to overfitting, only the Decision Tree classifier showcased a large discrepancy between its test and training error. This was later solved with pruning.

Among the test error successes, L1 Logistic Regression, L2 Logistic Regression, and Pruned Decision Tree were the most successful. Furthermore, its term-deposit precision rates were the highest among the predictors. Therefore, I recommend bank marketers to utilize regularized Logistic Regression and pruned Decision Tree classifiers for future efforts.

The feature selection methods showcased which features were most important in predicting the bank term-deposit outcome (yes or no). Out of 20 input variables, we filtered down to 12 features:

Categorical Variables: job, education, default, contact, poutcome, pdays
Continuous Variables: age, duration, campaign, emp.var.rate, cons.conf.idx, nr.employed

From our original hypothesis, both call duration and the employment variation rate were believed to be significant factors for determining the term-deposit outcome. We see that that to be true with regards to Chi-Square tests and variance. Bank marketers should keep a watchful eye on economic indicators that may affect client likelihood for investments. Furthermore, call duration should be tracked and monitored to further understand how long a "yes" client will be on call for.