**Executive Summary**

The objective of the project is to predict “yes class” with high probability. The input data was noisy and unclean. We cleaned the data and built numerous models for experimentation and finally settled on two models a. Logistic Regression with an AUC score of 0.7 for better explainability and b. Neural Network model with an AUC score of 0.85 for higher predictability.

The input data for the model was downloaded from a reliable source at https://drive.google.com/drive/folders/1ubUZJMrn3mkrz9jc3w2rckkmNc2cwJbo. It consisted of 40,000 records and 100 features. With exploratory analysis, we identified numerous issues ranging from

1. Missing Values, b. Outliers
2. Feature Scale Issues d. Inadequate Feature Values/ Frequency
3. Data Distribution Issues e. Correlated variables
4. Low variance Variables f. Data representation Errors
5. Feature Value Normalization

We identified the data issues, primarily using statistical tools/ tests, and then cleaned them up for modeling. For modeling, the training data was split into train data and validation data. The models were trained on train data and their evaluation was done on the validation data to ensure the performance observed was unbiased by the training process.

We tested different models ranging from Logistic Regression, Decision Tree, Boosted Trees (AdaBoost, Gradient Boost, XGBoost) and finally settled on to Neural Network model.

**Model Comparison:** Logistic Regression(LR) and Neural Network(NN)

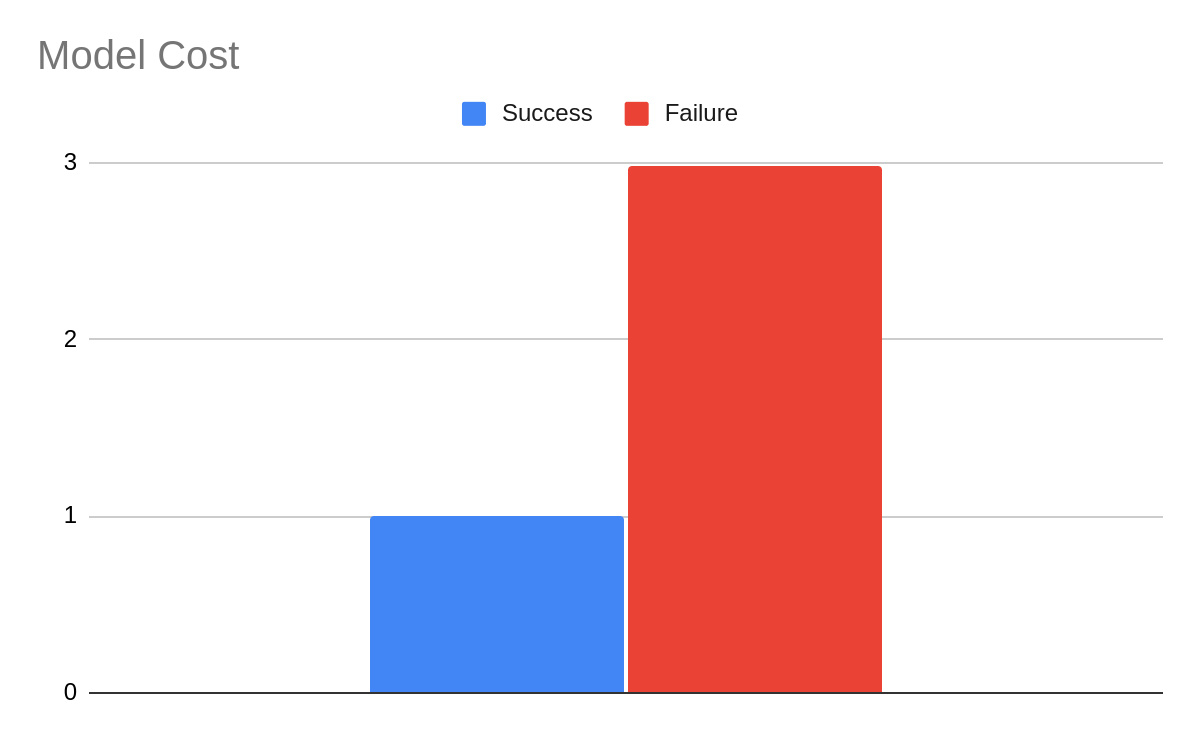
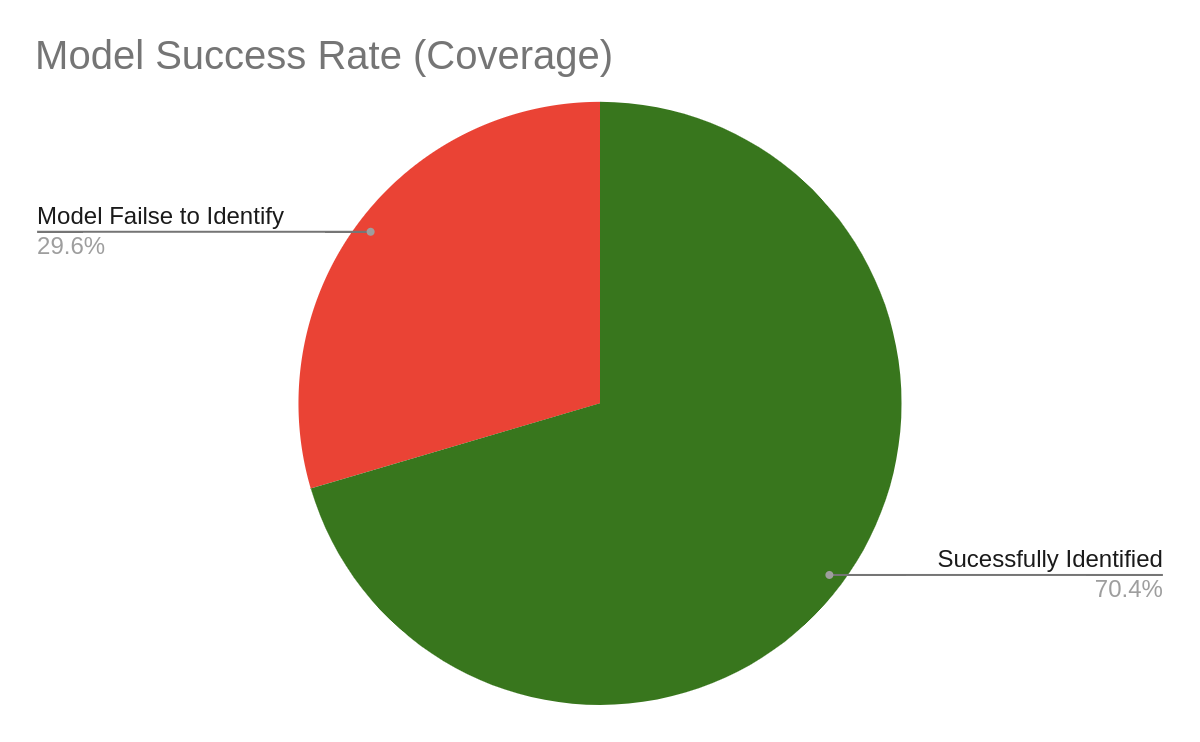
Logistic Regression were primarily selected from their explainability. It allows us to quantify the impact of each feature variable e.g In our model: i. People from Alaska(odds:3.47), and Washington(odds:2.95),.. are more likely to say yes compared to people from x33\_California(reference variable). ii. Day of week: People are more likely to say yes on x3\_Sat(odds: 1.21), x3\_Fri(odds: 1.25), and x3\_Sun(odds: 1.25) compared to Monday iii. x47: 1 unit increase in Scaled Number of X47 decreases yes likelihood by 58%.

These individual impacts can then be later used for business decision-making as well. However, the downside of the LR model is that they are not as powerful as other black box models i.e DNN. e.g In our modeling, with LR, we got an AUC of 0.70 but with DNN we were able to get an AUC score of 0.85. This also can be attributed to the fact the LR has difficulty representing AND, OR, and XOR interaction between variables. NN on the contrary can easily and automatically handle them. LR can handle AND interaction, however, they have to be manually fed to the model, while also accounting for not introducing multi-collinearity, which makes it a challenging and labor-intensive task. On the contrary, Neural Networks can compute complex and deep functions and select correct AND, OR, XOR, or any interaction terms implicitly and automatically and can provide greater predictability. However, the downside is that they are black box models and lack the explainability provided by models i.e LR.

We finally selected the Neural Networks model and expect it to perform better on the test set. The AUC score of 0.85 vs 0.7 on validation data and the plot of predictions across both LR and NN, gave us confidence that NN will outperform LR. Our NN model consists of 2 hidden layers, class weight, and output bias weight to account for the highly imbalanced nature of the classification problem. We estimate the AUC score of the test set to be equal to that of validation data or slightly lower i.e 0.8-0.85.

To demonstrate to our business partner that NN is better than LR, we can use Charts demonstrating what the accuracy and cost will be for each model. The precision of the model represents the accuracy of the model, while the recall represents the coverage of the model (i.e what % of the true class were captured)

E.g in the chart below, the model is able to capture 70% of customers who would open an account with us. However, only 1 out of 4 customers asked by the model to contact would actually open an account with us.



Depending upon the business preference, by choosing different threshold boundaries we can optimize for precision(higher accuracy) or recall(higher coverage).

We can use NN for its predictive power and use LR to understand the problem better. e.g

* People from Alaska(odds:3.47), Washington(odds:2.95),.. are more likely to say yes compared to people from x33\_California(reference variable)

x47: 1 unit increase in Scaled Number of X47 decreases yes likelihood by 58%.

More detailed steps and rationale behind each step, results, discussion, and conclusion are detailed in the Detailed report below

**Detailed Report**

**Title: Classification**

# 

# Introduction:

The objective of the project is two predict class yes class with high probability.

# 

# Methodology:

## 

## Data collection:

The data used for our project was downloaded from https://drive.google.com/drive/folders/1ubUZJMrn3mkrz9jc3w2rckkmNc2cwJbo

## 

## Exploratory Analysis: EDA

The training dataset consisted of 40,000 records and 100 features and the test data consisted of 10000 records. Exploratory Data analysis was done primarily using Statistical Tests/ Tools because of the large number of feature variables. Univariate and Bivariate plots and Tabular summaries were secondarily used. It helped us identify several data issues present in the Data and ensure the data quality i.e

1. Missing Values, b. Outliers
2. Feature Scale Issues d. Inadequate Feature Values/ Frequency
3. Data Distribution Issues e. Correlated variables
4. Low variance Variables f. Data representation Errors
5. Feature Value Normalization

It also helped us identify transformations that needed to be done on the features i.e

1. Categorical Encoding (Binary / Dummy Features)
2. Feature Transformation:
3. Eliminate Features

## 

## Reproducibility:

To ensure reproducibility, we have used the random seed to ensure that the randomization is the same across any number of runs. In addition, we have provided the code in a jupyter notebook alongside the Data, for anyone to rerun them and reproduce the results.

## 

## Preprocessing:

The Issues identified in the EDA led to the following preprocessing to

* Ensure Data Quality
* Ensure Modeling Data adheres to the upcoming ML model Data assumptions (e.g scaling- LR, no missing values- Sklearn-Trees, no correlated variables e.t.c )

It also helps the upcoming statistical model with better convergence and better predictive modeling power. The following preprocessing was done and applied to the data.

1. **Data Representation Errors:** 
   1. Removed %, $ from features to correctly represent them as their true numeric values. E.g $100 -> 100
2. **Missing Values**:
   1. Remove Variable: The variables with constant variable values or with the large number of missing values were removed. E.g X99-32% missing & rest constant value. X44, x57, and x30 had >80% of their data missing, and 'x55', and 'x52' had > 33%. With >33% missing, it's likely to add more noise and hence was removed.
   2. Remove Records with missing values on features: The column with the least amount of missing variables was 5%. Because eliminating 5% of records results in significant loss, we did not choose this option.
   3. Fill: Out of mean, median, mode, interpolation, and proportionate fills available, we choose proportionate fill. And because these records were in moderate size and treating them as the separate feature value was not justified. The following features were proportionately filled (X24-10% missing, X33 - 18% missing). We did a mode fill for X79 instead of mean because it was a binary variable and not continuous.

* Mean fill (5-33% values missing): ['x74', 'x49', 'x54', 'x95', 'x78', 'x16', 'x89', 'x14', 'x42', 'x85', 'x41', 'x45', 'x61', 'x96', 'x91', 'x80', 'x76', 'x75', 'x11', 'x64', 'x92', 'x26', 'x5', 'x83', 'x67', 'x63', 'x86', 'x38', 'x22', 'x68', 'x94', 'x88', 'x90', 'x73', 'x72', 'x71', 'x97', 'x70', 'x69', 'x82', 'x100', 'x84', 'x66', 'x7', 'x87', 'x81', 'x1', 'x62', 'x15', 'x25', 'x23', 'x21', 'x20', 'x18', 'x17', 'x13', 'x28', 'x12', 'x10', 'x9', 'x8', 'x6', 'x4', 'x27', 'x29', 'x58', 'x47', 'x56', 'x53', 'x2', 'x51', 'x50', 'x48', 'x46', 'x32', 'x43', 'x40', 'x37', 'x36', 'x35', 'x34', 'x19']
* Mode Fill : x79
  1. Keep as a separate class: We treated some missing values as a separate class because the missing value size was not significant enough to eliminate the feature completely but was significant enough to be randomly filled. More importantly in our case, missing values are likely to have represented some meaningful information e.g Vehicle missing could mean, a person does not drive any. Hence we kept missing values in X77 as a separate class.

1. **Inadequate Feature Variable Values**: Some values of the feature columns were too few. They can cause prediction reliability issues during inference. E.g only 1.1% of the data belonged to Utah. The model will hence have limited capability/performance when making predictions for Utah. While removal was an option because the state is a valid one, we included it in our prediction for now, but we must be careful with the inferences made for such few variable values.
2. **Outliers**: We detected outlier features using the Tukey method (x67, 54,49,95,74,58,79,..) with outlier count ranging from 0.18% to 13.36%. Of clipping, Replacement(mean, median, mode, model-interpreted ), transformation(log, binning) and Keep available, we choose to keep it because our model by itself is an outlier detection model (14%- yes class). Hence these outlier values are more likely to be signals for outlier detection rather than noise, hence we kept them for this iteration. In the future, we can test different outlier resolution strategies and test models.
3. **Normalisation:** We detected duplicate values representation for X3 (e.g sun, Sunday). Hence we normalized all duplicates in X3 wt its short form.
4. **Categorical Encoding**: Some Models i.e Regression, NN, SVM cannot handle categorical values and need to be converted into Dummy variables or Binary Encoded Variables. The following transformations were done:
   1. Dummy variables: If a feature has >2 categories, they were transformed into dummy variables. ("x3", "x60", "x65", "x77", "x33" )
   2. Binary Encoding: Features with only two categories i.e yes/no were converted into binary variables (i.e yes/ no -> 0/1) (x24", "x31", "x93").
5. **Skewed Data Distribution**: Skewed and non-zero mean-centered data causes challenges for models and their convergence. Of log transform, squaring, and sq/cube rooting, keep available, we choose to keep. X75 was flagged as skewed, but on inspection, we noted it followed normal distribution except for at ~ 29, ~1 s.d to right. Marking this as a separate categorical column and normalizing it and testing it will be a good future to do. Currently, we kept it as is.
6. **Scaling**: Having numerical features on different scales i.e 99999 vs 0 creates problems for models. They create slower non-convergence at best to worst performing model at worst. Hence we scaled the following features.

['x41', 'x50', 'x27', 'x58', 'x16', 'x100', 'x11', 'x85', 'x18', 'x22', 'x36', 'x53', 'x48', 'x34', 'x64', 'x86', 'x47', 'x72', 'x78', 'x38', 'x37', 'x2', 'x28', 'x40', 'x68', 'x67', 'x21', 'x89', 'x96', 'x49', 'x80', 'x61', 'x97', 'x17', 'x73', 'x25', 'x20', 'x92', 'x94', 'x13', 'x62', 'x83', 'x12', 'x90', 'x1', 'x9', 'x84', 'x19', 'x82']

Of diff scaling methods available ie min-max scaling, standardization, normalization, percentile-scaling we used percentile-based Robust scaler because it preserves outliers and is less outlier sensitive for non-outlier values.

Scaling makes prediction good. But because it increases model explainability/comprehension challenge later, keeping both scaled and original columns and later during modeling, experimenting using both feature variants alternatively is a good option. If there is no significant loss in prediction power, we can favor non-scaled columns for better and easier explainability, if explainability is preferable.

1. **Correlation**: Correlation creates problems for models. E.g For LR, they produce a numerically unstable model and can impact convergence. Hence we removed highly correlated variables. For our data, we observed no highly correlated variables.
2. **Variance**: Features with very low variance add very little information/ value to models. We checked and observed x7. But upon further analysis, we found out, it had nice variance but the low flagged variance was because of the scale [-0.03,0.03]. Hence we upscaled the variable by factor of 100.
3. **Multi-Collinearity**: ML Models i.e LR have no multi-collinearity data assumption. It leads to a numerically unstable model and for Lr in particular creates coefficient computation challenges. We removed multi-collinearity introduced with dummy variables for removing a reference variable i.e Remove-

## Modeling:

### DataSets:

We did a stratified split of the Train Data into 80% Train Data and 20% Validation Data. The final Test set was used for the final prediction.

**Imbalanced Data**: We also created synthetic data for training data only (not validation/ test data)’s minority class using SMOTE to create a balanced Dataset. Synthetic Balanced / Unbalanced Data both were tested and experimented with during the later modeling stage alongside the class\_weighting.

### Evaluation Metrics:

Because we had imbalanced data (yes-14% vs no-86%), we carefully choose metrics that are able to handle them well i.e AUC, recall, precision, f1-score,auc\_roc, balanced accuracy. Amongst them, we focussed more on AUC as requested and also because of the lack of problem description, we did not know if recall or precision was more important.

### Model Building / Validation:

We started by building a benchmark model (Tree model) and then into building more complex explainable models (Logistic Regression), Boosted Trees (AdaBoost, Gradient Boost, XGBoost), and Neural Networks while benchmarking them against the original Benchmark model. More complex but unexplainable models i.e (Deep Neural Networks ) although more powerful and have higher predictive power were excluded from our current scope because of a lack of explainability.

### Model Comparision

We choose Two models Logistic Regression and Neural Network finally as our comparative models. The logistic regression model provides good explainability. It allows us to quantify the impact of each parameter e.g

* People from Alaska(odds:3.47), Washington(odds:2.95),.. are more likely to say yes compared to people from x33\_California(reference variable)
* Day of week: People are more likely to say yes on x3\_Sat(odds: 1.21), x3\_Fri(odds: 1.25), x3\_Sun(odds: 1.25) compared to Monday
* x47: 1 unit increase in Scaled Number of X47 decreases yes likelihood by 58%.

However, the downside of the Logistic Regression model is that they are not as powerful as other black box models i.e DNN. e.g In our modeling, with Logistic Regression, we got an AUC of 0.70 but with DNN we were able to get an AUC score of 0.85. This also can be attributed to the fact that Logistic Regression has difficulty coming with AND, OR, and XOR logics, which Neural Networks can easily and automatically handle.

PS: Logistic Regression can handle ANDs, however, they have to be manually fed to the model, while also accounting for not introducing multi-collinearity, which makes it a challenging and labor-intensive task.

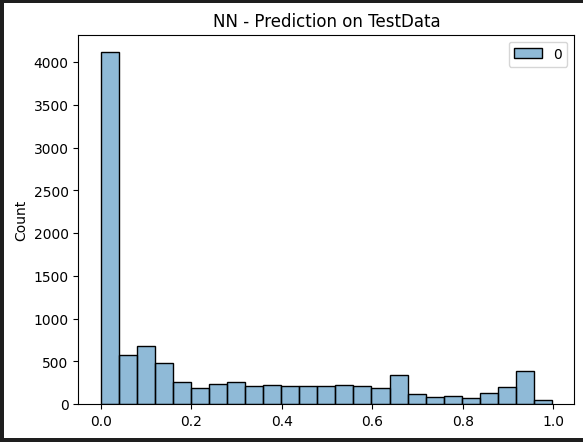
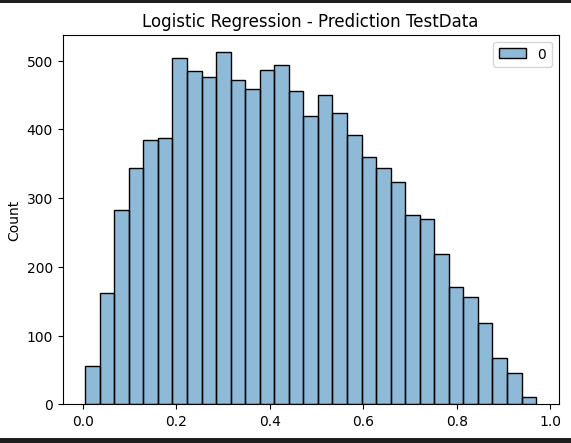
On the contrary, Neural Networks can compute complex and deep functions and can provide greater predictability. However, they lack the explainability provided by the Logistic Regression Models.

## 

## Results:

The following result were obtained for our models.

| DNN : Validation AUC at epoch 479 : | 0.85 |
| --- | --- |
| Logistic Regression AUC (Validation Data) | 0.699 |



Results for all other models are as follows:

| **model\_name** | **roc\_auc** | **recall** | **precision** | **accuracy** | **balanced\_accuracy** | **f1** |
| --- | --- | --- | --- | --- | --- | --- |
| Simple Tree Unbalanced | 0.56 | 0.27 | 0.25 | 0.77 | 0.56 | 0.26 |
| Simple Tree balanced | 0.56 | 0.26 | 0.25 | 0.78 | 0.56 | 0.26 |
| Boosted Tree (AdaBoost)-LR\_1.5 | 0.79 | 0.22 | 0.53 | 0.86 | 0.59 | 0.31 |
| Boosted Tree (AdaBoost)-LR\_1.7 | 0.79 | 0.23 | 0.51 | 0.86 | 0.6 | 0.32 |
| Boosted Tree (GradientBoost)-n\_estimator\_100-L... | 0.72 | 0.23 | 0.37 | 0.83 | 0.58 | 0.28 |
| LR-l2-0.01 | 0.76 | 0.69 | 0.27 | 0.69 | 0.69 | 0.39 |
| LR-l2-1 | 0.76 | 0.69 | 0.27 | 0.69 | 0.69 | 0.39 |
| LR-l1-0.1 | 0.76 | 0.7 | 0.27 | 0.68 | 0.69 | 0.39 |
| LR-l1-0.1-SMOTE | 0.76 | 0.7 | 0.27 | 0.68 | 0.69 | 0.39 |

## Model Selection:

We finally selected the Neural Networks with AUC: 0.85 for non-glm model with 2 hidden layers, class weight, and output bias to account for highly imbalance nature of classification problem.

And the Logistic Regression model with AUC: 0.7 as our glm model.

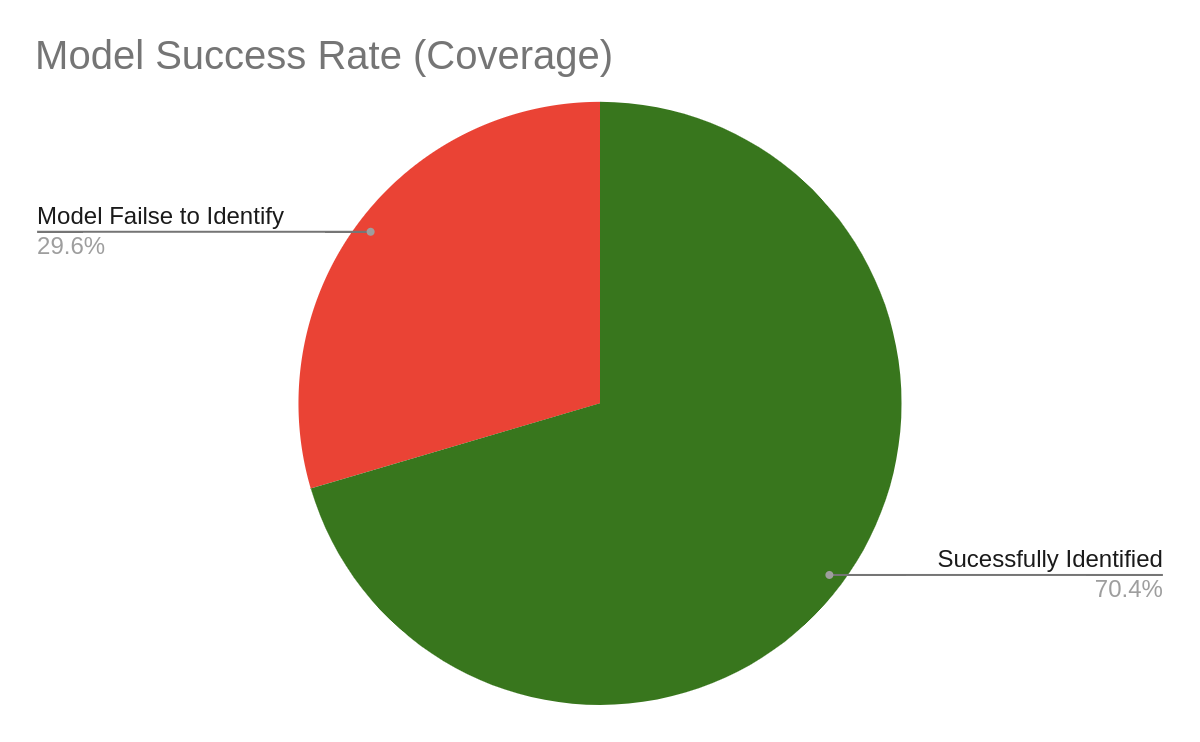
## Model Performance Estimate on Test Data:

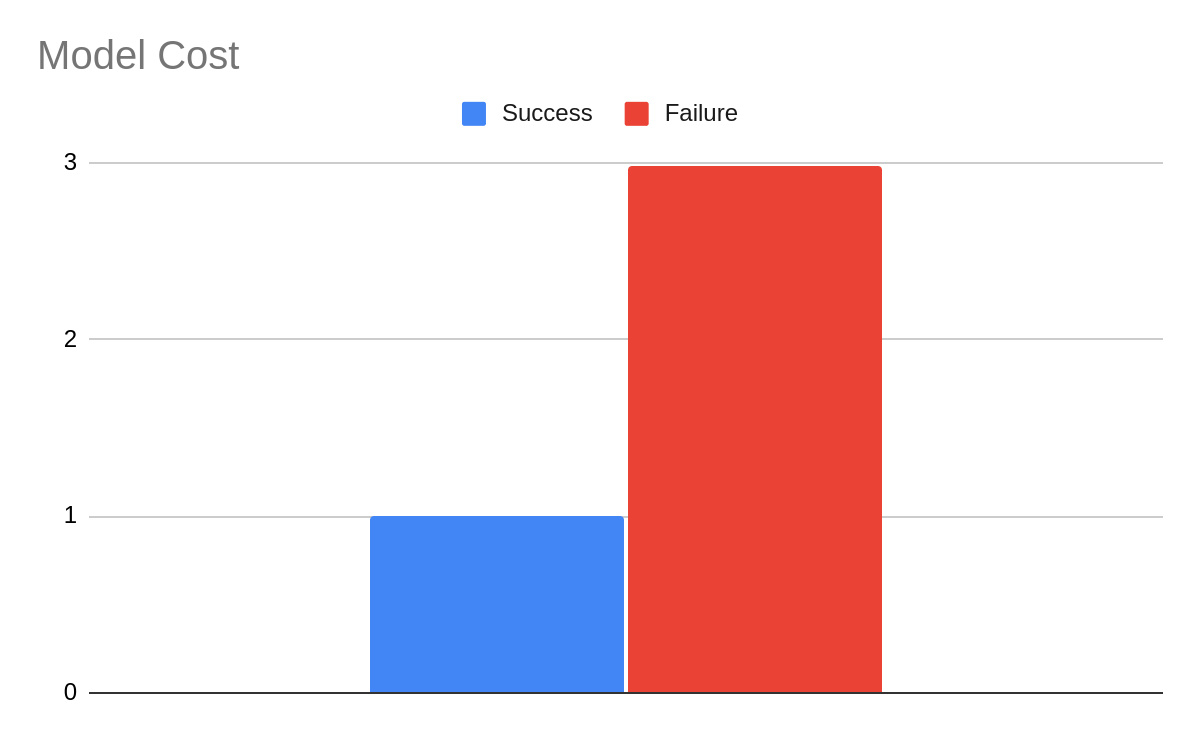
After observing the large AUC difference of 0.7 vs 0.85 on validation data observed between the Logistic Regression and Neural Network model and the plot of predictions from both the model, our estimate is that the NN’s will perform significantly better than Logistic Regression on Test set. And since the validation data was unseen data, we estimate the AUC score of validation data i.e 0.85 to be reflected on the Test set as well.

## Model Performance Explanation (Layman):

We can use Charts to illustrate the benefit and cost implication for each of the models as they are much more intuitive to the business person and layman users. Precision projects the accuracy of the model in predicting class “yes” (i.e benefit) and recall projects coverage of the model(cost).

E.g in the chart below, the model is able to capture 70% of customers who would open an account with us. However, only 1 out of 4 customers asked by the model to contact would actually open an account with us.





## Conclusion:

Depending upon the business preference, by choosing different threshold boundaries, a business may get higher recall or precision.

* E.g if the task is telemarketing then, a business can capture 86% of Customers(more coverage/ recall preferred) at the cost of a 1:4 success rate(0.03 threshold) i.e out of 5 calls, only 1 lead to success and 4 are failed conversion calls..
* E.g if a task is a fraud, then a business can trade off the % of successful customers captured (recall) with precision

However if the interest of the modeling is the understandability of the problem to shape other business decisions down the line, then it can be done using logistic regression. For example, based on our LR modeling, we obtained the following many positively impacting factors and negatively impacting factors. The top 2 sampled positive and negative impacting factors are

**Top 2 Positive Impacting Features**

- State: People from Alaska(odds:3.47), Washington(odds:2.95),.. are more likely to say yes compared to people from x33\_California(reference variable)

- other positive odds state :

x33\_West\_Virginia, x33\_Mississippi, x33\_Rhode\_Island, x33\_Nebraska, x33\_Indiana, x33\_South\_Dakota, x33\_Maine, x33\_Iowa, x33\_Ohio, x33\_DC, x33\_Kentucky, x33\_Missouri, x33\_South\_Carolina, x33\_Florida, x33\_Illinois, x33\_Georgia, x33\_Vermont, x33\_North\_Carolina, x33\_Kansas, x33\_Montana, x33\_Washington, x33\_Idaho, x33\_Oregon, x33\_Alaska

- Day of the week: People are more likely to say yes on x3\_Sat(odds: 1.21), x3\_Fri(odds: 1.25) , x3\_Sun(odds: 1.25) compared to Monday.

**6.2 Top 2 Negative Impacting Features**

- x47: 1 unit increase in Scaled Number of X47 decreases yes likelihood by 58%.

- State: People from Wyoming are less likely to say yes (odds: 0.7) compared to people from x33\_California(reference variable)

## Future Work:

Given the limited time frame, not all potential action items could be pursued. However, we have identified and listed them for future work

* Include Interaction Terms in Logistic Regression
* Analyse model parameters against the EDA bivariate plots and discuss/ validate.
* Test more different Preprocessing Strategies and their impact on models. (e.g outlier removal, clipping)
* More variable slicing/ segmentation for significant p-values
* More Complex Modeling (Deep Neural Network, ConvNets) for more predictive power.
* Automated Hyper Parameter Tuning