

## Carolyn Massa - Assignment completed 3.3.2020

### Learning Objective

- Practice identifying which supervised and unsupervised learning techniques are best suited for your Capstone Project data.
- Utilize supervised and unsupervised learning techniques to build predictive models.

To begin, I painstakingly read what seemed to be an infinite # of articles (+ youtube videos) on models and algorithms as well as the parameters and hyperparameters and how to select the options to improve your model's objective. Below is what I learned and I will both cite and paraphrase my sources.

The first statistical test I ran on my 10,000 observations was a Chi Squared test.



A **chi-square** ( $\chi^2$ ) statistic is a test that measures how expectations compare to actual observed data (or model results). The data used in calculating a **chi-square** statistic must be random, raw, mutually exclusive, drawn from independent variables, and drawn from a large enough sample.

After I ran my Chi Squared test on my raw data I reviewed how to preprocess my data in order to improve model accuracies:

Predictive Models:

**Scaling the Data Set:** I use the min/max operations to scale my "continuous variables" to eliminate unnecessary variances. Min/Max is also known as "**Normalization**". This formula behind this is below: These "normalization"

**Normalization Formula**

$$X_{\text{normalized}} = \frac{(X - X_{\text{minimum}})}{(X_{\text{maximum}} - X_{\text{minimum}})}$$


techniques help in comparing corresponding normalized values from two or more different data sets in a way that it eliminates the effects of the variation in the scale of the data sets i.e. a data set with large values can be easily compared with a data set of smaller values. I used it to process my data as I will apply Logistic Regression, Support Vector Machine Learning and a Random Forest Method for my algorithm and I want to eliminate

variants in my data. SVM is intrinsically two-class. For multiclass problem you will need to reduce it into multiple binary classification problems. For example, if I am working with and comparing two different data sets and one has much larger values than the other, I would want them to be standardized between a range of 0 to 1. The Normalization techniques are not typically used in Random Forest or K Nearest neighbors as one is working with decision trees which are not affected by scaling your data.

### Modeling Pipeline

Next, I build a "**Data Pipeline**" In Python which allows the me to transform data from one representation to another through a series of steps. In other words, to ensure my hot encoding, min/max normalization and both my categorical and continuous variables continue in the test and train modeling.

## Model Fitting

For the Bank Churn dataset, since it is relatively small (10,000 records and 13 variables which I split into 2 sets: Training (70%) and Testing (30%) I chose the following 3 algorithms to build my model:

**1) Logistic Regression:** Logistic Regression is one of the basic and popular algorithms to solve a classification problem. It is named as '**Logistic Regression**', because it's underlying technique is quite the same as Linear Regression. The term "Logistic" is taken from the **Logit function** that is used in this method of classification which uses the Sigmoid function.

An explanation of logistic regression can begin with an explanation of the standard logistic function. The logistic function is a Sigmoid function, which takes any real value between zero and one. It is defined as

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$$

My logistic regression parameter choices:

```
log_primal = LogisticRegression(C=100, class_weight=None, dual=False,  
    fit_intercept=True, intercept_scaling=1, max_iter=250, multi_class='warn', n_jobs=None,  
    penalty='l2', random_state=None, solver='lbfgs', (I chose lbfgs as it is used in Multi Class  
problems which can handle Binary Classes and supports both L1 and L2 penalties so it minimizes both total  
squared error and absolute value of error) tol=1e-05, verbose=0, warm_start=False)
```

\* **Error Sum of Squares. Error Sum of Squares (SSE)** SSE is the **sum** of the **squared** differences between each observation and its group's mean. It can be used as a measure of variation within a cluster. Stanford.edu

**2) Support Vector Machine (SVM)** is a supervised machine learning algorithm which can be used for both classification and regression challenges. SVM is mostly used in classification problems. *It is a **classifier** formally defined by a separating hyperplane which is a subspace of one dimension less than its ambient space.* For my model, I performed a "CrossValidation" Grid Search to determine the best parameters.

In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a coordinate. SVM is better suited as I need a way to separate my data into CHURN or NO CHURN and Logistic Regression uses a straight line. I used a large C to output low bias and high variance and to instill a regulation parameter and achieve a higher level of accuracy. I chose an RBF (Radial Basis Function) Kernel. It is a **stationary** kernel, which means that it is invariant to translation. I read on "stackexchange" that RBF is a safe default as it contains one parameter and is relatively smooth.

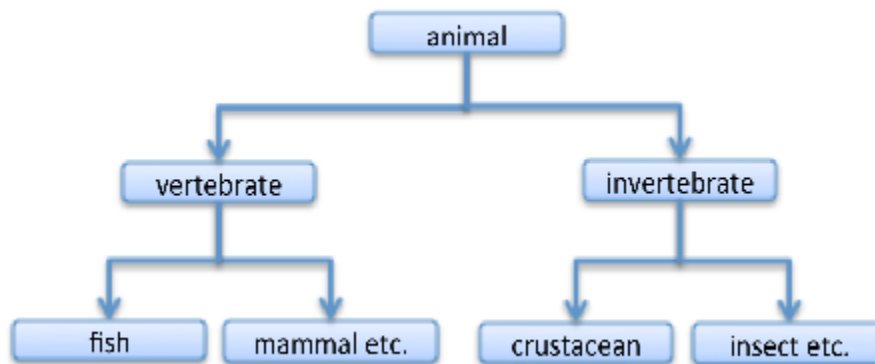
BF will work on the *difference* of the two vectors and will scale the same in all directions. Gamma is a parameter of a Gaussian Kernel which is used on non - linear data to find the dimensions of classification. To "raise" the points you use the RBF kernel, gamma controls the shape of the "peaks" where you raise the points. A small gamma gives you a *pointed bump in the higher dimensions*, a large gamma gives you a softer, broader bump.

A small gamma will give you low bias and high variance while a large gamma will give you higher bias and low variance.

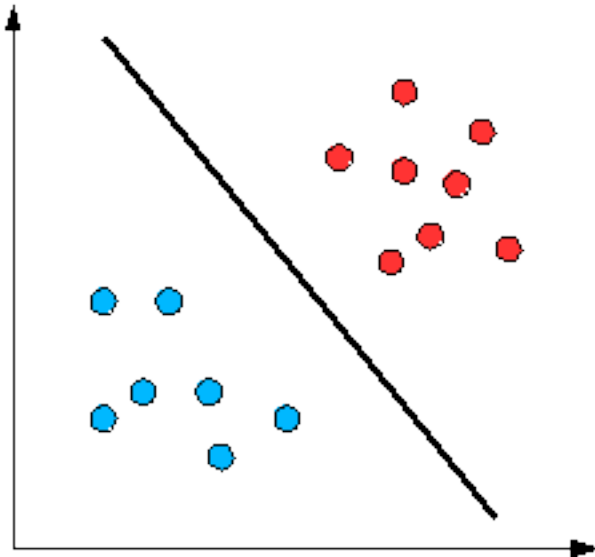
### My SVM Gridsearch Cross Validation choices:

```
GridSearchCV(cv=3, error_score='raise-deprecating',
             estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', (grid search chose this as it is a multi-class problem and ovr = "one v
             ersus the rest"), degree=3,
             gamma='auto_deprecated', kernel='rbf', max_iter=-1,
             probability=False, random_state=None, shrinking=True,
             tol=0.001, verbose=False),
             iid='warn', n_jobs=None,
             param_grid={'C': [0.5, 100, 150], 'gamma': [0.1, 0.01, 0.001],
                         'kernel': ['rbf'], 'probability': [True]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

Example below of an SVM (reference: Toward Data Science)



Therefore the hyperplane of a two dimensional space below (fig.2) is a one dimensional line dividing the red and blue dots.



*Deciding on a kernel – I cite this simple explanation from [“Toward Data Science”](#)*

**Simple Example:**  $x = (x_1, x_2, x_3)$ ;  $y = (y_1, y_2, y_3)$ . Then for the function  $f(x) = (x_1x_1, x_1x_2, x_1x_3, x_2x_1, x_2x_2, x_2x_3, x_3x_1, x_3x_2, x_3x_3)$ , the kernel is  $K(x, y) = \langle f(x), f(y) \rangle$ .

Let's plug in some numbers to make this more intuitive: suppose  $x = (1, 2, 3)$ ;  $y = (4, 5, 6)$ . Then:

$$f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)$$

$$f(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)$$

$$\langle f(x), f(y) \rangle = 16 + 40 + 72 + 40 + 100 + 180 + 72 + 180 + 324 = 1024$$

A lot of algebra. Mainly because  $f$  is a mapping from 3-dimensional to 9 dimensional space.

Now let us use the kernel instead:

$$K(x, y) = (4 + 10 + 18)^2 = 32^2 = 1024$$

Same result, but this calculation is so much easier.

**The Kernel Relation to SVM:** now how is related to SVM? The idea of SVM is that  $y = w \phi(x) + b$ , where  $w$  is the weight,  $\phi$  is the feature vector, and  $b$  is the bias. if  $y > 0$ , then we classify datum to class 1, else to class 0. We want to find a set of weight and bias such that the margin is maximized. Previous answers mention that kernel makes data linearly separable for SVM. I think a more precise way to put this is, *kernels do not make the data linearly separable. The feature vector  $\phi(x)$  makes the data linearly separable*. Kernel is to make the calculation process faster and easier, especially when the feature vector  $\phi$  is of very high dimension (for example,  $x_1, x_2, x_3, \dots, x_D, x_1^2, x_2^2, \dots, x_D^2$ ).

The **significance level**, also denoted as **alpha** or  $\alpha$ , is the probability of rejecting the null hypothesis when it is true. For example, a **significance level** of 0.05 indicates a 5% risk of concluding that a difference exists when there is no actual difference.

**3) Random Forest** works well with a mixture of numerical and categorical features which the Bank Churn data has. When features are on the various scales, it is also fine. Roughly speaking, with Random Forest you can use the data

as it is. Random Forest uses a large # of trees, works with missing values and is often considered to be a highly accurate model for both regression and classification problems.

I chose the following hyperparameters:

Research randomseed

```
RF = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',(the Gini parameter will quantitatively evaluate how good a split is. Gini Impurity is the probability of incorrectly classifying a randomly chosen element in the dataset if it were randomly labeled according to the class distribution in the dataset. When training a decision tree, the best split is chosen by maximizing the Gini Gain, which is calculated by subtracting the weighted impurities of the branches from the original impurity.) ,max_depth=8,(these are the maximum features when the tree decides to make a split) max_features=6,
max_leaf_nodes=None,min_impurity_decrease=0.0,
min_impurity_split=None,min_samples_leaf=1,
min_samples_split=3,min_weight_fraction_leaf=0.0
n_estimators=50,(This will determine the # of trees in my model) n_jobs=None,
oob_score=False, random_state=None, verbose=0,warm_start=False)
```

## Classification Reports:

The last line gives a weighted average of precision, recall and f1-score where the weights are the support values.

The total is just for total support which is 5 here.

The f1-score gives you the harmonic mean of precision and recall. The scores corresponding to every class will tell you the accuracy of the classifier in classifying the data points in that class compared to all other classes.

The support is the number of samples of the true response that lie in that class which for this study are 2995.

Source: [sklearn documentation](#).

## Logistic Regression

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.83      | 0.96   | 0.89     | 2411    |
| 1            | 0.57      | 0.20   | 0.29     | 584     |
| accuracy     |           |        | 0.81     | 2995    |
| macro avg    | 0.70      | 0.58   | 0.59     | 2995    |
| weighted avg | 0.78      | 0.81   | 0.78     | 2995    |

## SVM Grid

---

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.88      | 0.98   | 0.93     | 2411    |
| 1            | 0.86      | 0.45   | 0.59     | 584     |
| accuracy     |           |        | 0.88     | 2995    |
| macro avg    | 0.87      | 0.71   | 0.76     | 2995    |
| weighted avg | 0.88      | 0.88   | 0.86     | 2995    |

---

## Random Forest

---

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.86      | 0.98   | 0.92     | 2411    |
| 1            | 0.85      | 0.36   | 0.50     | 584     |
| accuracy     |           |        | 0.86     | 2995    |
| macro avg    | 0.85      | 0.67   | 0.71     | 2995    |
| weighted avg | 0.86      | 0.86   | 0.84     | 2995    |

---

I review my scores by my Model using the F1-score as it takes both Churn and Non-Churn predictions into account and I applied it to an uneven class distribution. I will look at how well the mode did at predicting “Churn” which is represented by the “1” in my report.

**Accuracy** - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you must look at other parameters to evaluate the performance of your model.

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

**Precision** - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all bank customers that are labeled as having remained at the bank, how many remained with the bank? High precision relates to the low false positive rate.

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

**Recall (Sensitivity)** - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers are: Of all the bank customers that truly remained with the bank, how many did we label?

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

**F1 score** - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

Source = [Exsilio](#)

Logistic Regression results: The accuracy for our LR model is 0.81 which means our model is approx. 81% accurate. We have 0.78 precision score which is also strong as well as a recall of 0.81 which is good for this model as it's above 0.5. Our F1 score is 0.90.

Support Vector Machines: The accuracy for our SVM model is 0.88 which means our model is approx. 88% accurate. We have 0.88 precision score which is also strong as well as a recall of 0.88 which is good for this model as it's above 0.5. Our F1 score is 0.86.

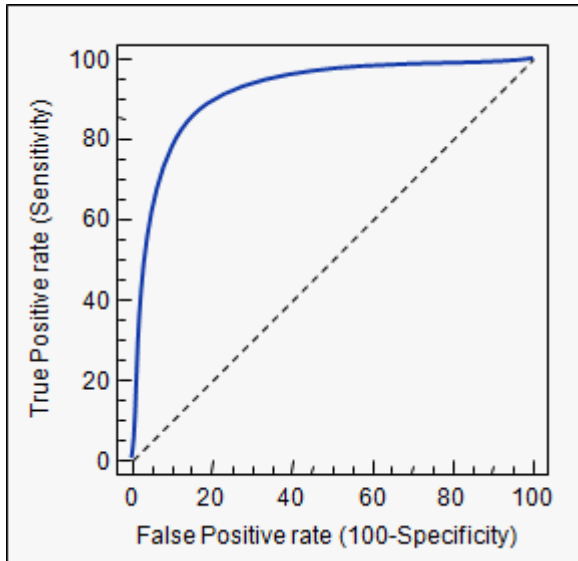
Random Forest Model: The accuracy for our RF model is 0.86 which means our model is approx. 86% accurate. We have 0.86 precision score which is also strong as well as a recall of 0.86 which is good for this model as it's above 0.5. Our F1 score is 0.84.

As we see from the Confusion Matrix Below that our RF Model resulted in the following:

From the sample of 2995 the RF model found 2387 predictions to be true positives which is a "NO CHURN" result and 329 as a True Negative which is a "Churn" result.

| Confusion Matrix for Random Forest   |                    |          |  |
|--|--------------------|----------|--|
| Actual   | Predicted by Model |          |  |
|  | Positive           | Negative |  |
|  | TP = 2373          | FN = 38  |  |
|  | FP = 375           | TN = 209 |  |
|  |                    |          |  |
| We then add the TP and the TN then divide by our sample of 2995 to get .86 which matches our weighted average in our RF Classification Report. |                    |          |  |

## ROC CHART



A **ROC curve (receiver operating characteristic curve)** is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

**True Positive Rate (TPR)** is a synonym for recall and is therefore defined as follows:

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

**False Positive Rate (FPR)** is defined as follows:

$$FPR = \frac{FP}{FP + TN}$$

**Fig 14:** Illustration of ROC Chart (Source = Wikipedia)

A ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and the AUC represent degree or measure of **separability**. It tells how much a model is capable of distinguishing between classes. The higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. By analogy, Higher the AUC, better the model is at distinguishing between patients with disease and no disease.

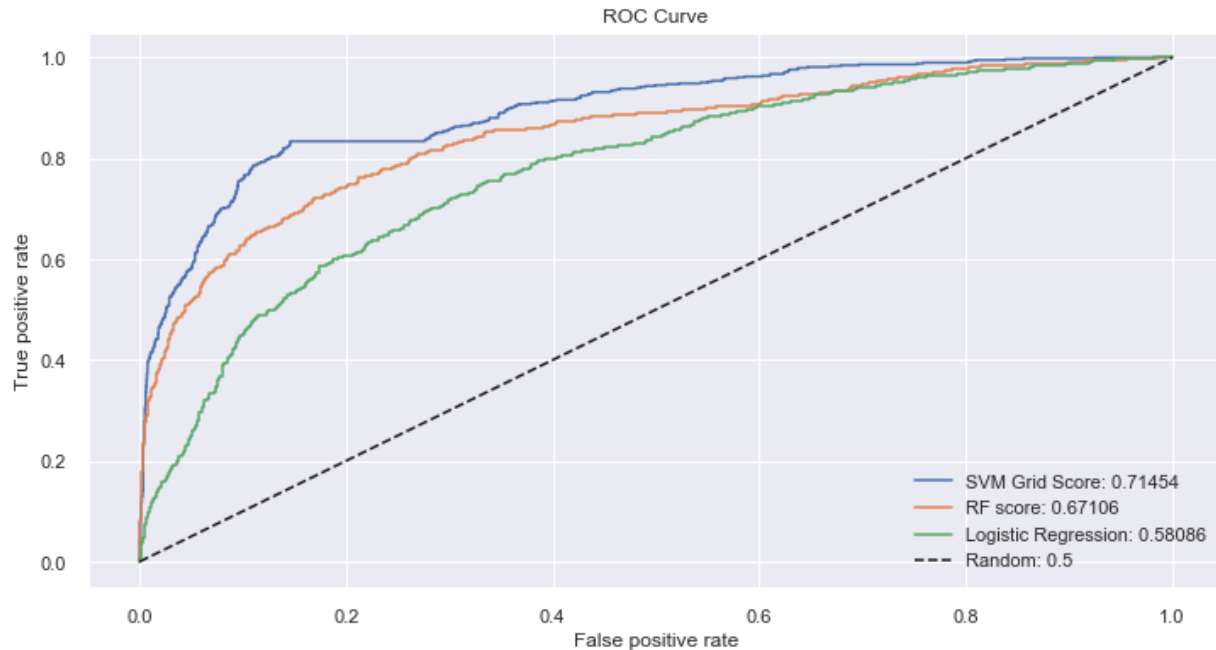
The ROC curve is plotted with TPR against the FPR where TPR is on y-axis and FPR is on the x-axis.

Source: [Toward Data Science](#)

To evaluate effectiveness of my model I will be using the following metrics.

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





**Fig 15 above:** ROC/AUC compares the effectiveness of the Logistic Regression, the SVM and RF models. This accuracy is measured by the area under the ROC curve. An area of 1, for instance, is a perfect score while .5 is meaningless. So, for all the customers, .67 of the instances are correctly classified in the **RF model** .71 are correctly classified in the SVM model, and only .58 of the instances in the study are correctly classified in the Logistic Regression model.

Here is a rough guide for classifying the accuracy of the ROC/AUC using a diagnostic test is the traditional academic point system:

- .90-1 = excellent (A)
- .80-.90 = good (B)
- .70-.80 = fair (C)
- .60-.70 = poor (D)
- .50-.60 = fail (F)

Source [UNMC](#)

**Now that we now the accuracies of our 3 Models let's look at how each Data Point contributes to the level of Churn for our Bank.**

**Fig 8 below:** Pearson Correlation

I rank each Variable by its significance in determining if the customer churns or not  
It looks like Age and Balance/Salary Ration played importance in determining Churn.

| Legend of Feature Importance |                                     |
|------------------------------|-------------------------------------|
| 1.                           | Age 1 (0.315051)                    |
| 2.                           | Credit Score 0 (0.214699)           |
| 3.                           | Tenure 2 (0.209859)                 |
| 4.                           | Balance/Salary Ratio 6 (0.041458)   |
| 5.                           | Has Credit Score 9 (0.038024)       |
| 6.                           | Balance 3 (0.037885)                |
| 7.                           | Credit Score Given Age 8 (0.037067) |
| 8.                           | Tenure By Age 7 (0.036401)          |
| 9.                           | Estimated Salary 5 (0.035074)       |
| 10.                          | NumofProducts 4 (0.034483)          |

**P-Value Test- Next I need to either Accept or Reject my Null Hypothesis which is** The smaller the p-value the

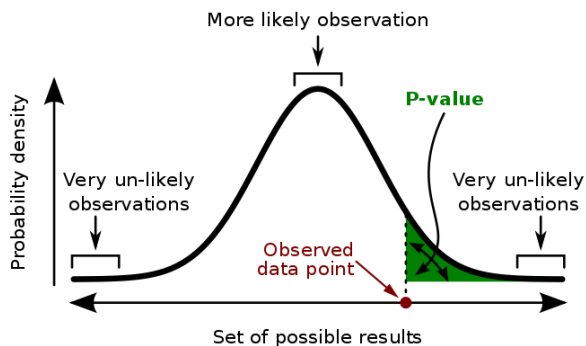
higher the significance because it tells the investigator that the hypothesis under consideration may not adequately explain the observation. The null hypothesis is rejected. If any of these probabilities is less than or equal to a small, fixed but arbitrarily pre-defined threshold value. This is referred to as the level of significance and is set by the researcher before examining the data and is arbitrary. It typically ranged from .05 to .001. Here I review the results and determine that since "Balance" and "BalanceSalaryRatio" are below .05 I will reject my Null Hypothesis.

Important:

**$\Pr(\text{observation} \mid \text{hypothesis}) \neq \Pr(\text{hypothesis} \mid \text{observation})$**

The probability of observing a result given that some hypothesis is true is *not equivalent* to the probability that a hypothesis is true given that some result has been observed.

Using the p-value as a "score" is committing an egregious logical error: **the transposed conditional fallacy.**



A **p-value** (shaded green area) is the probability of an observed (or more extreme) result assuming that the null hypothesis is true.

The **Chi-square test** is intended to **test** how likely it is that an observed distribution is due to chance. It is also called a "goodness of fit" statistic, because it measures how well the observed distribution of data fits with the distribution that is expected if the variables are independent.

Below I performed a Chi Squared Test to test my Null Hypothesis which was **"Age is not a determining factor to whether a Customer leaves the bank or not"**

We see from the results below the "p-value" is less than our significance of .05 so we reject the HO and go with the HA which is **"Age has a determining factor if the customer leaves the bank"**

```
contingency_table :-
  Exited    0    1
```

Gender

Female 3404 1139

Male 4559 898

*We notice more females have left the bank than males*

**Observed Values :- Our observed values (actual values are different from our "Expected" Values if there were no relationship)**

```
[[3404 1139]
```

```
[4559 898]]
```

**Expected Values :-**

```
[[3617.5909 925.4091]
```

```
[4345.4091 1111.5909]]
```

Degree of Freedom:- 1

chi-square statistic:- 113.44910030392086

critical\_value: 3.841458820694124

p-value: 0.0

Significance level: 0.05

Degree of Freedom:

```
1
chi-square statistic: 113.44910030392086
critical_value: 3.841458820694124
p-value: 0.0
Reject H0, There is a relationship between 2 categorical variables
Reject H0, There is a relationship between 2 categorical variables
```

Hypothesis “ Bank Balance has little effect on if the person left the bank or not”

Bank balance details of those who exited the bank (observable only)

```
count      2037.000000
mean      91108.539337
std       58360.794816
min        0.000000
25%       38340.020000
50%       109349.290000
75%       131433.330000
max       250898.090000
Name: Balance, dtype: float64
```

Bank balance details of those who did not exit the bank (observable only)

```
count      7963.000000
mean      72745.296779
std       62848.040701
min        0.000000
25%        0.000000
50%       92072.680000
75%      126410.280000
max       221532.800000
Name: Balance, dtype: float64
```

Here are my results of a T test using the Bank Balance versus Exited variable

```
t=-2.262, df=198, cv=1.653, p=0.025
Reject the null hypothesis that the means are equal.
Reject the null hypothesis that the means are equal.
```

We notice the p- value (alpha) is less than the .05 significance threshold so we reject the Null Hypothesis and go with the alternative hypothesis “Bank Balance” DOES affect if the customer left the bank or not.