This is a **predictive analytics** project where we will try to predict future events based on the historical data using machine-learning algorithms. The objective is to find out why the best employees are leaving by using this human-resources database with the following data items:  1. Satisfaction Level, 2. Last evaluation, 3. Number of projects, 4. Average monthly hours, 5.Time spent at the company, 6.Whether they have had a work accident 7. Whether they have had a promotion in the last 5 years, 8.Sales, and 9.Salary 10. Whether the employee has left

**Some general statistics of the dataset.**

SUMMERY STATISTICS OF NUMERIC VARIABLES:

left=0 ---------------------------------------------------------------------------------------------------- ---

Variable N Mean Std Dev Minimum Maximum

---------------------------------------------------------------------------------------------------------------

satisfaction\_level 11428 0.6668096 0.2171038 0.1200000 1.0000000

last\_evaluation 11428 0.7154734 0.1620050 0.3600000 1.0000000

number\_project 11428 3.7866643 0.9798838 2.0000000 6.0000000

average\_montly\_hours 11428 199.0602030 45.6827312 96.0000000 287.0000000

time\_spend\_company 11428 3.3800315 1.5623480 2.0000000 10.0000000

----------------------------------------------------------------------------------------------------------------

left=1 --------------------------------------------------------------------------------------------------------

Variable N Mean Std Dev Minimum Maximum

----------------------------------------------------------------------------------------------------------------

satisfaction\_level 3571 0.4400980 0.2639334 0.0900000 0.9200000

last\_evaluation 3571 0.7181126 0.1976734 0.4500000 1.0000000

number\_project 3571 3.8555027 1.8181654 2.0000000 7.0000000

average\_montly\_hours 3571 207.4192103 61.2028254 126.0000000 310.0000000

time\_spend\_company 3571 3.8765052 0.9776979 2.0000000 6.0000000

---------------------------------------------------------------------------------------------------------------

FREQUENCYS OF CATAGORICAL VARIABLES:

Cumulative Cumulative

sales Frequency Percent Frequency Percent

---------------------------------------------------------------------------------------------------------------

IT 1227 8.18 1227 8.18

RandD 787 5.25 2014 13.43

accou 767 5.11 2781 18.54

hr 739 4.93 3520 23.47

manag 630 4.20 4150 27.67

marke 858 5.72 5008 33.39

produ 902 6.01 5910 39.40

sales 4140 27.60 10050 67.00

suppo 2229 14.86 12279 81.87

techn 2720 18.13 14999 100.00

Cumulative Cumulative

salary Frequency Percent Frequency Percent

----------------------------------------------------------------------------------------------------------------

high 1237 8.25 1237 8.25

low 7316 48.78 8553 57.02

medium 6446 42.98 14999 100.00

Cumulative Cumulative

Work\_accident Frequency Percent Frequency Percent

-------------------------------------------------------------------------------------------------------------------

0 12830 85.54 12830 85.54

1 2169 14.46 14999 100.00

promotion\_last\_ Cumulative Cumulative

5years Frequency Percent Frequency Percent

-------------------------------------------------------------------------------------------------------------------

0 14680 97.87 14680 97.87

1 319 2.13 14999 100.00

**1. Data Processing:**

After importing the data remove test set without replacing 70/30 splits. Our data set has several different datatypes that were categorized according to the following schema:

|  |  |  |  |
| --- | --- | --- | --- |
| **Data**  Data can be classified into two different types. | | | |
| **Categorical** Values or observations that can be sorted into groups or categories. Bar charts and pie graphs are used to graph categorical data. | | **Numerical** Values or observations that can be measured. And these numbers can be placed in ascending or descending order. Scatter plots and line graphs are used to graph numerical data. | |
| **Nominal** Values or observations can be assigned a code in the form of a number where the numbers are simply labels. You can **count but not order** or measure nominal data. Examples: Sex, and eye color. True/False | **Ordinal** Values or observations can be ranked (put in order) or have a rating scale attached. You can **count and order**, but not measure, ordinal data. Example: house numbers and swimming level. | **Discrete** Values or observations that is counted as **distinct and separate** and can only take particular values. Examples: the number of kittens in a litter; number of threads in a sheet, number of stars given for an energy rating. | **Continuous** You can **measure** continuous data. Values or observations **may take on any value** within a finite or infinite interval. Examples: height, time and temperature. Age. |

Source: <http://www.abs.gov.au/websitedbs/CaSHome.nsf/Home/CaSQ+3B+NUMERICAL+DATA:+WHAT'S+THE+DIFFERENCE+BETWEEN+DISCRETE+AND+CONTINUOUS>



**Dummy Variables:**

Two of categorical variables in our data set needed to be converted into dummy variables. Dummy variables are created in this situation to ‘trick’ the regression algorithm into correctly analyzing attribute variables.

**Standardization:**

The standardization or **z**-**score** is a very useful statistic that prevents the different scaling of variables from having an undue impact on the final model. We standardize all the continuous predictors. from our data set because it allows us to calculate the probability of a employee left of stayed within our normal distribution and also to compare between employee left of stayed that are from different normal distributions.

**2. Tools / Methods**

We will use classification techniques and models here to make our predictions. The goal is to achieve the highest possible accuracy and test the predictive technique on the test data. If our technique is accurate then our prediction results on the test data (which is randomly drawn from the available data) will generate exactly the same values.

**Logistic Regression (Sohana, Bruce)**

Linear regression models with lm() function is great when we have continuous response variables. But in our case, the response variable left is a factor (categorical) variable. We will use glm() function to see the relationship between the dependent and the independent variables, and then use the predict() function on the test data to get the probability for the next employee to leave.

**Support Vector Machine** (**SVM**) **(Greg)**

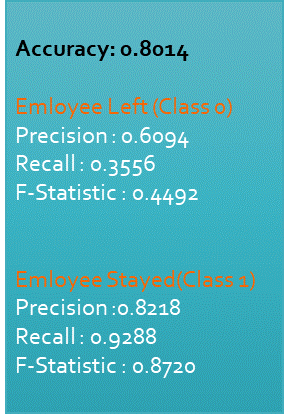
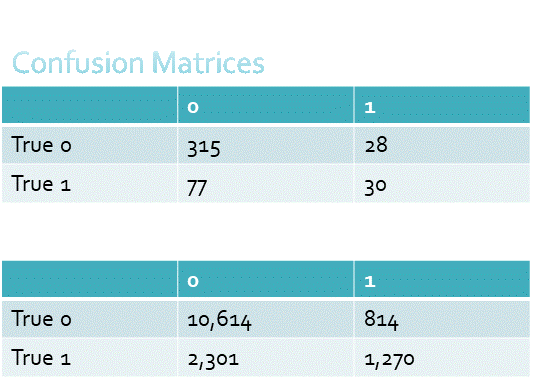
The support vector machine is very useful for creating the most optimal boundary to differentiate between different classes. The SVM will ignore outliers and concentrate on only the relevant points. The fitsvm() function in Matlab will allow our team to create this model and generate the statistics necessary to compare this model’s results to other classification model results.

**3. Model Generation:**

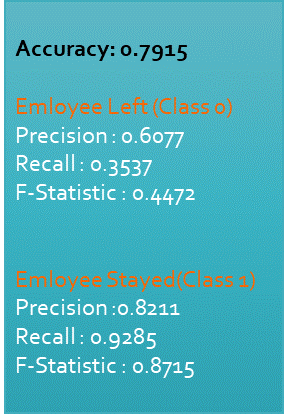
**Logistic Regression (Sohana Bruce):**

Since our target variable is categorical, logistic regression is appropriate. Logistic regression assigns an observation to a particular group based on the probability that the observation belongs to a particular group. The probability/likelihood is generated by a logistic function. The MatLab generalized linear model classifies the data using a Log-Likelihood or Maximum Log-Likelihood function.

After loading and transforming the data, we ran a logistic regression analysis and generated the following results:



Note the slightly unbalanced confusion matrix, which implies that the accuracy of the model may not be reliable. We evaluated the model for outliers found 1,313 observations at the standard cutoff of >3x the Cook’s value mean—too many to eliminate—so settled on 5x, which eliminated 638 values. Lasso regularization found that 14 of the 18 predictors could be retained, as shown by the Lasso plot below:



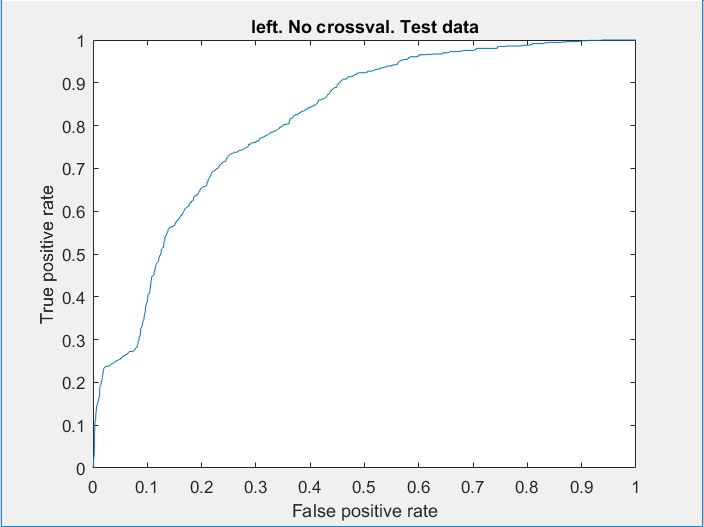
Surprisingly, the revised model did not perform as well as the original. We next turned to the support vector machine.

**SVM (Greg and Muntaser):**

Due to our data set high dimensions I hypothesized that an SVM algorithm would be a great wat to separate the data into two classes. The SVM will construct the best possible decision boundary to separate the two classes. Due to the high dimensionality of the data the box constraint and kernel scale parameters will be utilized. As the research ensued it was discovered that MATLAB has an Objective Function to minimalize the classification error at a given range of box and kernel parameters. This function was utilized. However, the function did come with a run time penalty. The study began with 70/30 split between training and test data. A validation set was not used.

The initial test set with the default box constraint did not produce optimal results. The average likelihood of predicting a record as class 1 was 69% with the prediction of class 0 at 81%.

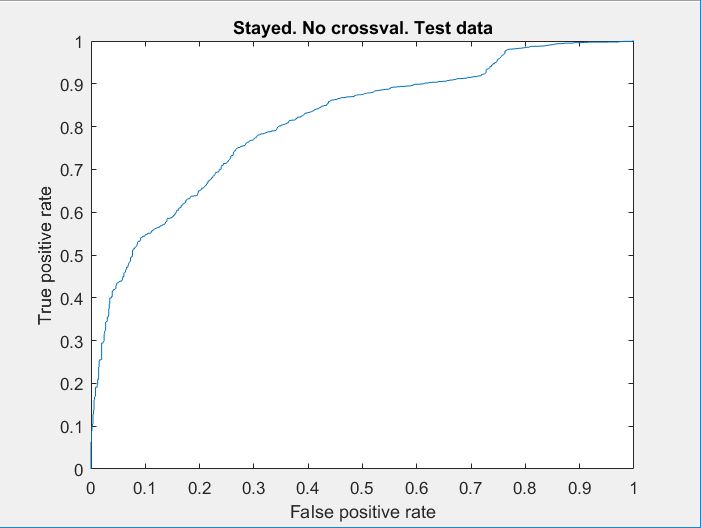
**Class 1:**



Accuracy: .7793 Precision: .8034

F1: .8666 Recall: .9405

**Class 0:**



Accuracy: .7793 Precision: .8057

F1: .8667 Recall: .9410

Various folds were tested to in an attempt to find the optimal split. The data set was not large and I believe this limited any information gain with an increase in folds. A fold of 15 did maintain a slightly better running average.

5-fold average F1 score: .8665

10-fold average F1 score: .8665

15-fold average F1 score: .8667

Initially the box constraint and kernel parameter were hand tested. This laborious technique did not seem to produce the desired result. Therefore, MATLAB’s Bayesian Optimization Function was utilized.

#A range of values was selected

sigma = optimizableVariable('sigma',[1e-2,1e2],'Transform','log');

box = optimizableVariable('box',[1e-2,1e2],'Transform','log');

#A partition was created with kfold equal to 15

c = cvpartition(10499,'kfold',15);

#The SVM model was build implementing our parameter range

minfn = @(z)kfoldLoss(fitcsvm(x,y,'CVPartition',c,...

'KernelFunction','rbf','BoxConstraint',z.box,...

'KernelScale',z.sigma));

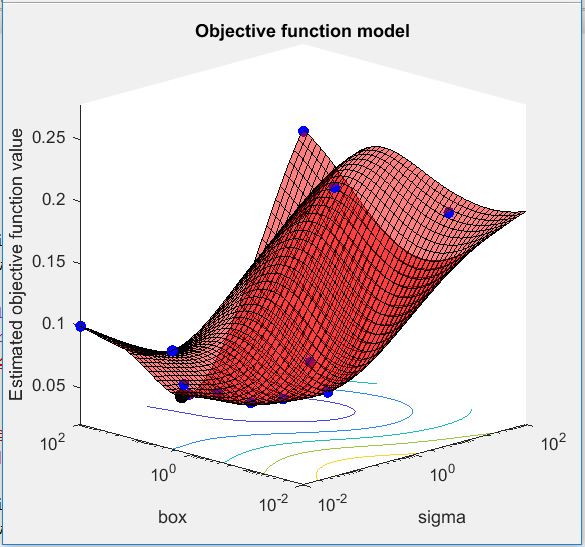
#Bayesian Optimaztion assumes that new trials will improve on current observations

#The function returns 30 possible solutions labeled best and acceptable

results = bayesopt(minfn,[sigma,box],'IsObjectiveDeterministic',true,...

'AcquisitionFunctionName','expected-improvement-plus')

Bayesian Optimization Function attempting to minimize loss



The final model with optimized parameters produced astounding results.

ROC: Box = 10.368 Sigma = 1.039

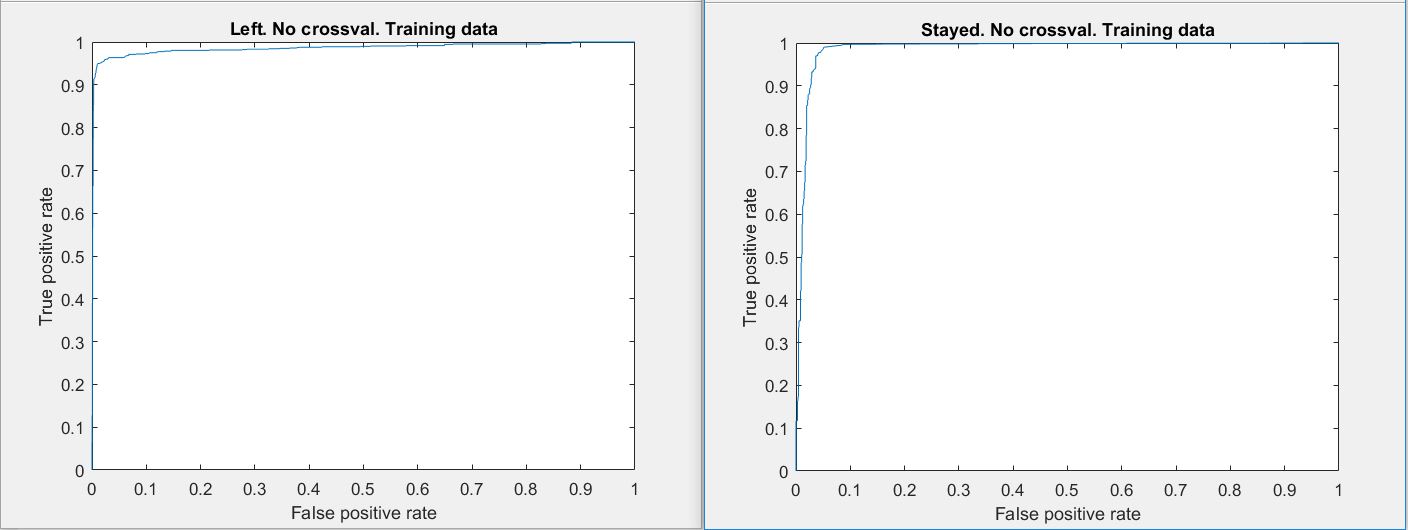
Positive Class:

* Accuracy = .9764
* Precision = .9845
* Recall = .9845
* F1 = .9845
* Misclassification Rate = .0236 or 2.36%
* Average likelihood of 1 prediction: .9987
  + Average likelihood of 0 prediction: .9949

Negative Class:

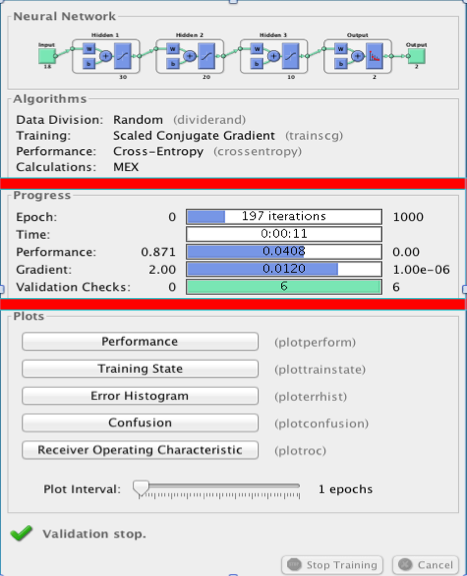
* Accuracy = .9765
* Precision = .9506
* Recall = .9506
* F1 = .9506
* Misclassification Rate = .0236 or 2.36%

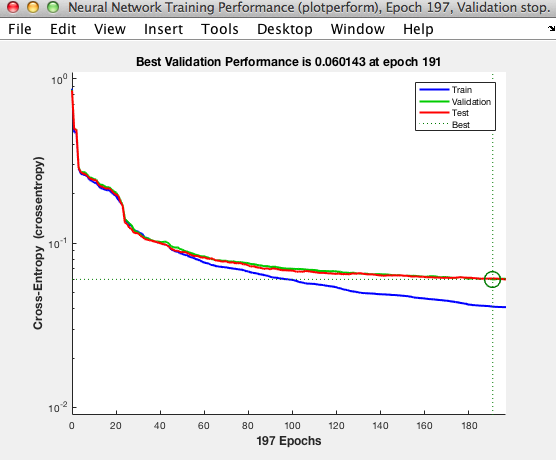
The SVM model produced the most accurate results.



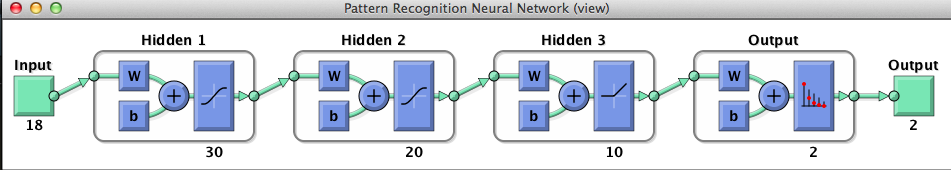
**Neural Network (Sohana):**

We are predicting, “Why are our best and most experienced employees leaving prematurely”. So, created a Pattern Recognition Network since the performance evaluation in this network is based on Cross-Entropy and transfer function “Softmax”. We used hidden LayerSize = [30,20,10].as increasing layer size reduces the error rate in large scale than increasing number of nueron. Here our purpose was to improve prediction quality.. The machine took 197 epoch to measure the number of times all training vector are used to update weight and reached best validatie performance.

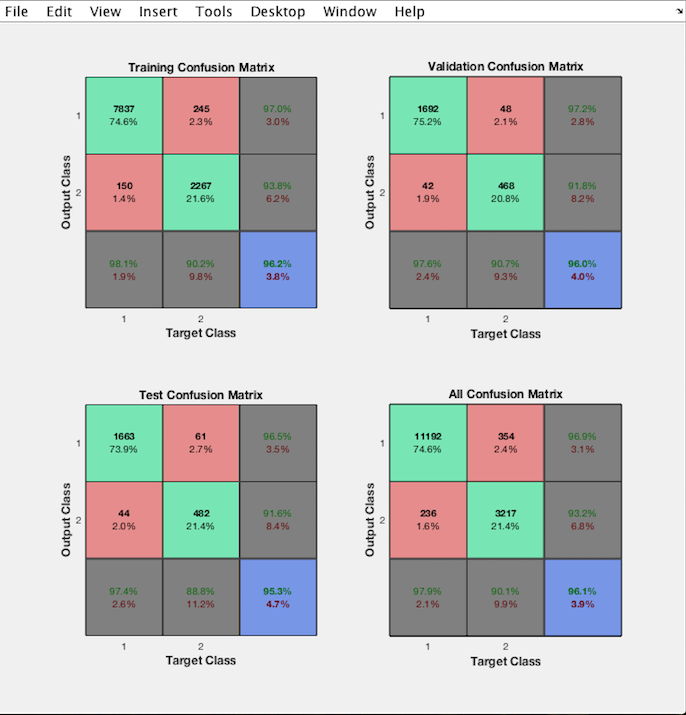
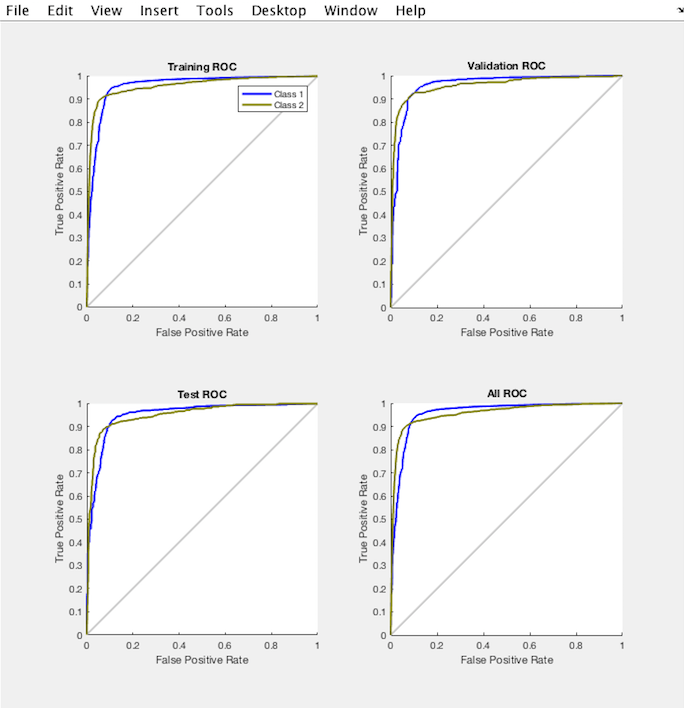




We setup division of data for training 70%, for validation 15%, and testing 15% then train and test the network.



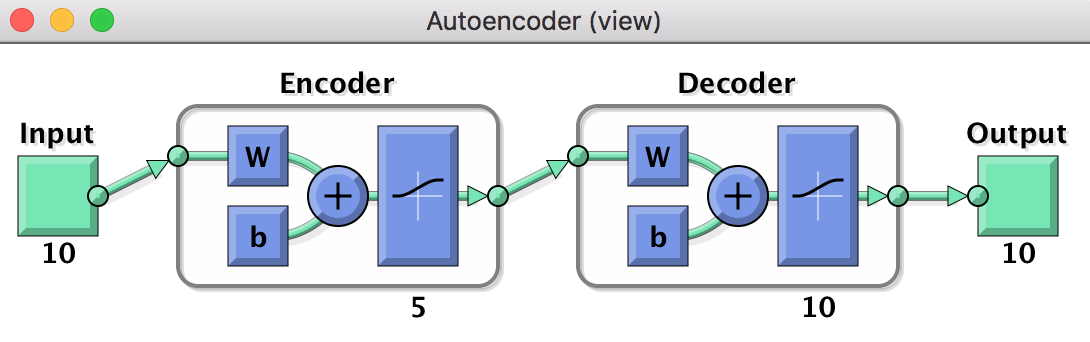
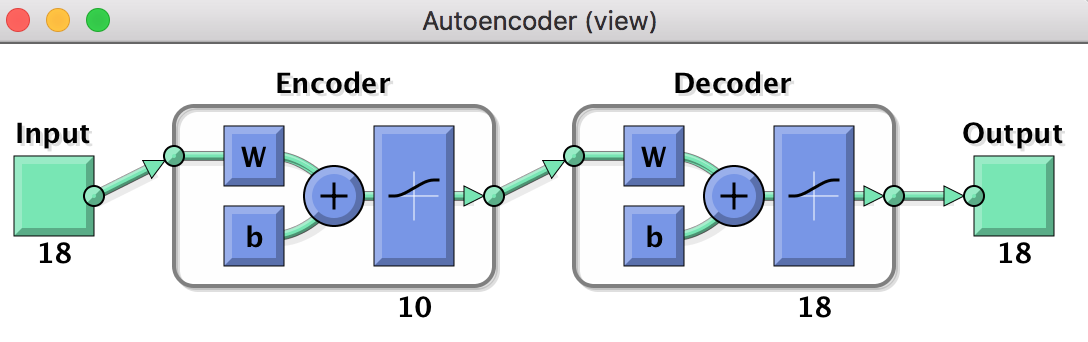
Before the machine was taking longer epoch than 197 and we realized training over longer epoch does not reduce error effectively; it slowing down the learning because of vanishing gradient decent (VGD). So we set the last layer to ReLU to turn off the neuron with gradient decent zero. By using ReLU we achieve epoch quickly (at 197) as can be seen above.



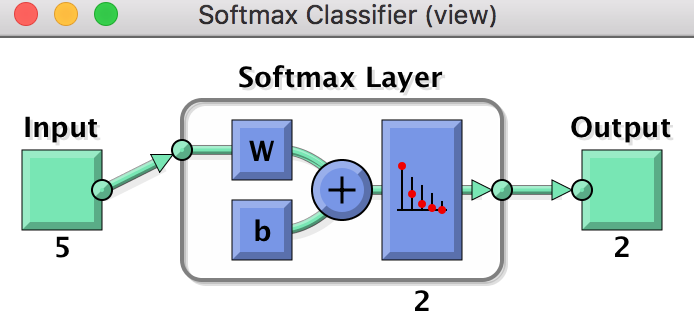
We found 96.1% overall accuracy for this model. For class 1 the precision 97.9% recall 96.9% and F1 97.4% and for class 2 the precision 90.1%, recall 93.2% and F1 91.6%. Also, ROC and AUC curve for both classes are looking good.

A**utoencoder (Samia)**

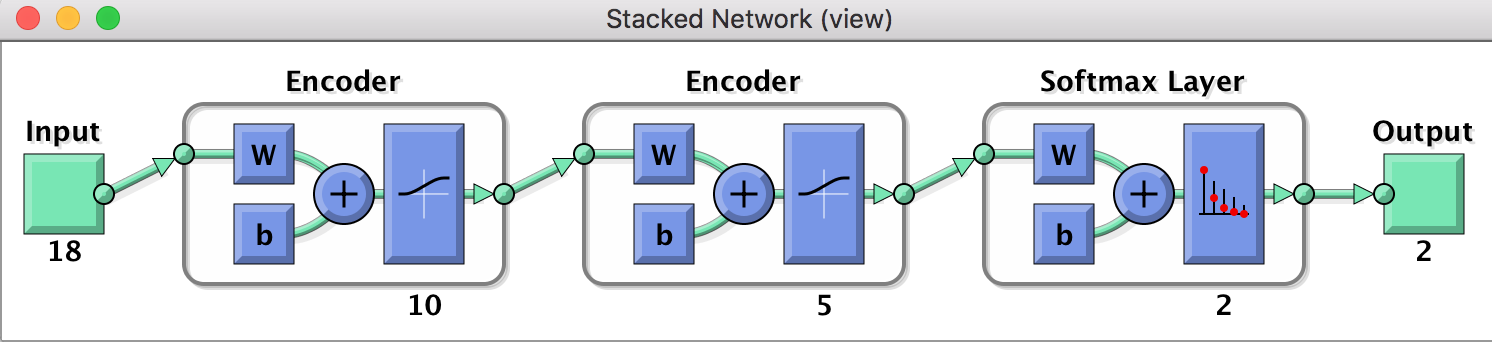
Although we got very good result by using SVM, we decided to try autoencoder on our dataset. I used autoencoder with two hidden layers to classify the training data. In first layers I used 10 neurons, and 5 neurons in second layer.

****

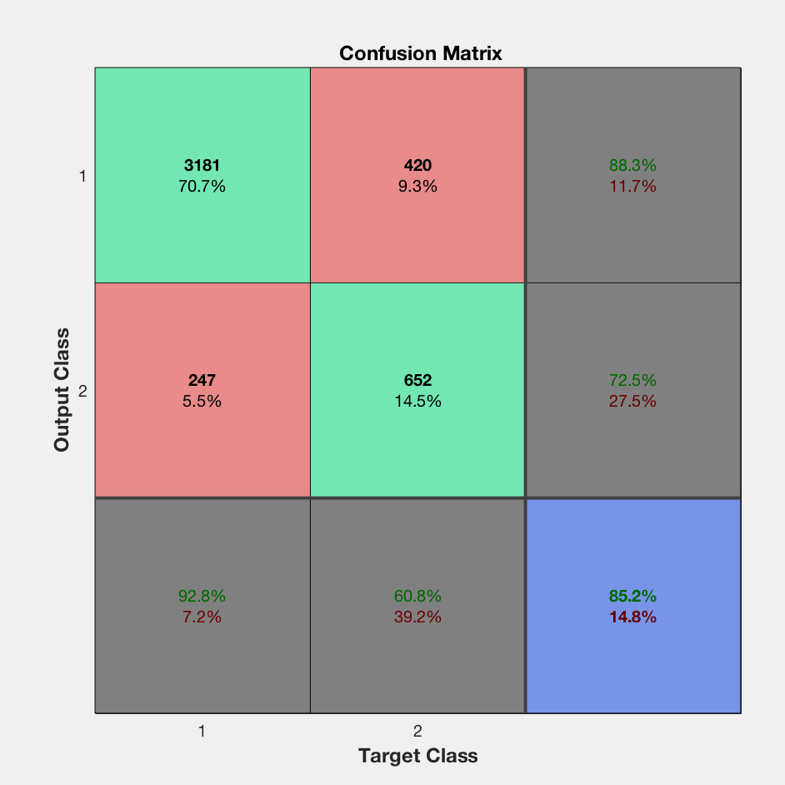
Then I trained softmax layer to classify the 5-dmentionla featurevectors of the second layer by using labels for the training data.



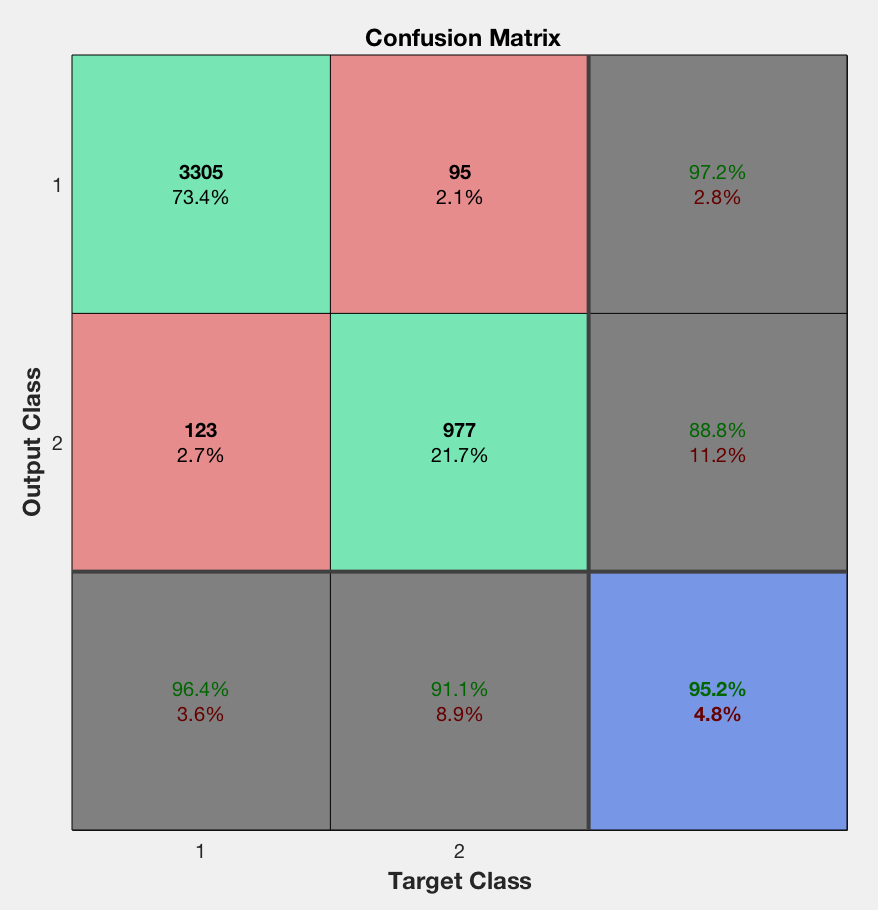
to form the DNN, I stacked the encoder of the first and second layers with the softmax layer.



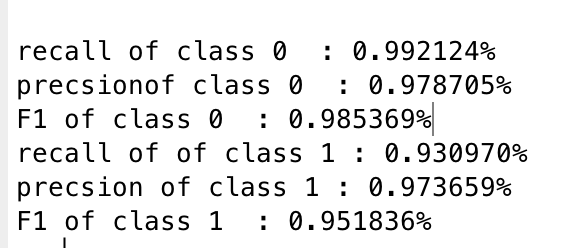
after forming the DNN, I could compute the results on the test data. In the blew picture is the result with the confusion matrix before doing fin-tuning.



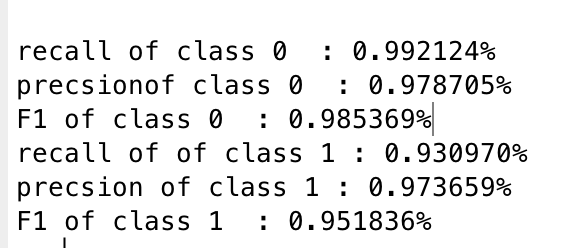
as in the above picture the accuracy before fine-tuning is 85%. To get better accuracy I performed fine-tuning and visualized the result with confusion matrix.

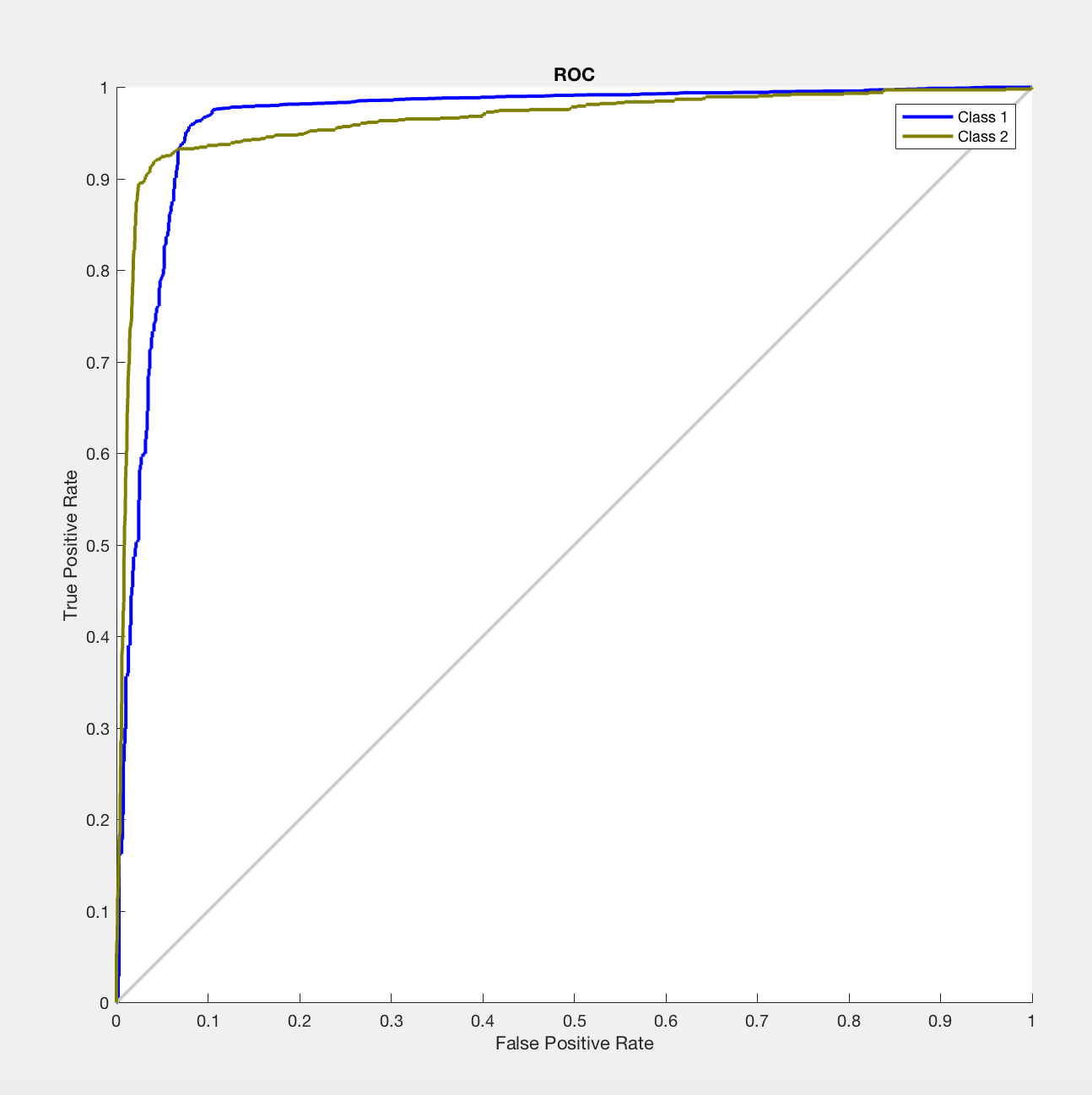


the accuracy after fine-tuning increased by 10 % and became 95%. To know more detail about how well the autoencoder classify our training data, I computed precision, Recall, and plot ROC curve for both classes. The below picture is for class 0 which represent stayed employee.



Below is the result for class 1 which represent left employee, and ROC for both classes





the autoencoder works very well for both classes. It classifies our test data to both classes with quality measures around 93% which considered very high quality.

**3. Model Validation:**

|  |  |  |
| --- | --- | --- |
| Models | Employee Stayed  (Class 0) | Employee Left  (Class 1) |
| Logistic regression | |  |  | | --- | --- | | Accuracy | **0.8215** | | Precision | 0.6094 | | Recall | 0.3556 | | F-Statistic | 0.4492 | | |  |  | | --- | --- | | Accuracy | **0.8215** | | Precision | 0.8218 | | Recall | 0.9288 | | F-Statistic | 0.8720 | |
| SVM | |  |  | | --- | --- | | Accuracy | .**9764** | | Precision | .9506 | | Recall | .9506 | | F-Statistic | .9506 | | |  |  | | --- | --- | | Accuracy | **.9765** | | Precision | .9845 | | Recall | .9845 | | F-Statistic | .9845 | |
| Neural network | |  |  | | --- | --- | | Accuracy | **0.961** | | Precision | 0.901 | | Recall | 0.932 | | F-Statistic | 0.9162 | | |  |  | | --- | --- | | Accuracy | **0.961** | | Precision | 0.979 | | Recall | 0.969 | | F-Statistic | .9740 | |
| Auto-encoder | |  |  | | --- | --- | | Accuracy | **95%** | | Precision | 97% | | Recall | 99% | | F-Statistic | 98% | | |  |  | | --- | --- | | Accuracy | **95%** | | Precision | 97% | | Recall | 93% | | F-Statistic | 95% | |

**4. Make Prediction:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SVM | |  |  | | --- | --- | | Accuracy | .**9765** | | Precision | .9506 | | Recall | .9506 | | F-Statistic | .9506 | | |  |  | | --- | --- | | Accuracy | **.9765** | | Precision | .9845 | | Recall | .9845 | | F-Statistic | .9845 | |

**5. Conclusion / Lessons Learned**

Our task was to build a classification model that could predict whether employees would be likely to stay or leave. The support vector machine model generated the best accuracy, precision, and recall, with the best F-Statistic for employees who left. Scaling the data was important to successful modeling, as was persistence in several alternative model parameters to come up with the best result.

**6. References**

The Element of Statistical Learning, by T. Hastie, R. Tibshirani, J. Friedman, Springer, 2013.

Deep Learning, by Ian Goodfellow, Yoshua Bengio, Aaron Courville, MIT Press, 2016

Learning From Data, by Y. S. Abu-Mostafa, M. Magdon-Ismail, H. T. Lin, AMLBook, 2012.

Matlab, Statistics and Machine Learning Toolbox, http://www.mathworks.com/help/stats/