**IN402 Unit 10 Assignment**

Laurence T. Burden

Purdue Global University

IN402: Modeling and Predictive Analysis

Steward Huang

January 21, 2024

The dataset provided contains details for banking customers with a target variable of whether each customer remained or left, also known as churn (Iyyer, 2019). There are a total of 14 features across 10,000 observations. The target variable is titled “Exited.”

The code starts by dropping columns that hold no analytical value. These include Geography, Gender, RowNumber, CustomerId, and Surname. The primary features analyzed are CreditScore, Age, Tenure, Balance, and EstimatedSalary. The variance of each of these columns are displayed.

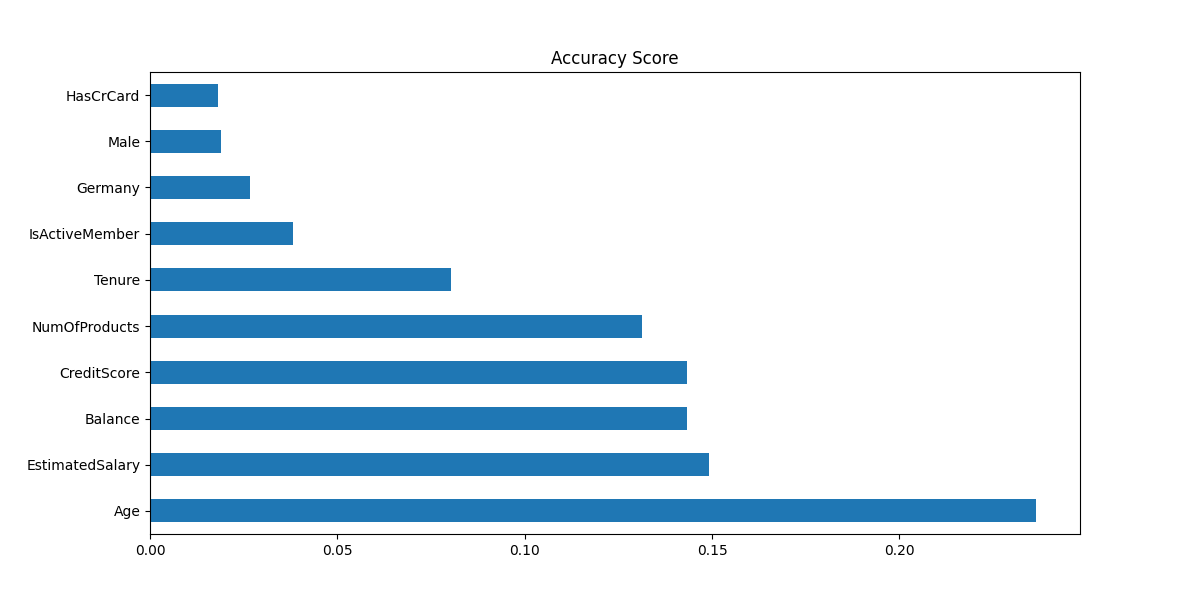
The high variance of some of the columns will not affect a logistical regression model. On the other hand, the data will need to go through feature scaling to have a useful SVM model. The individual ML models are created next using a train set of 8,000 values and a test set of 2,000.

The logistic regression model is created first. It has an accuracy of 0.811, which is a good accuracy score. The confusion matrix show that 1,526 true positives were predicted and 96 true negatives were predicted. However, 69 false positives and 309 false negatives were predicted. This could be an acceptable score based on the information not being used for life-or-death decisions.

The SVM model does not fair as well as the logistic regression model. It has a decent accuracy score of .7975, but the precision and recall scores are both zero. This means that no actual positives were identified correctly. For this reason, it would not be wise to use this model.

The random forest model has an accuracy score of 0.864. This is better than the logistic regression model. The precision of the model is 0.73, but the recall is only 0.51. This means that the model has a close to 50% chance to incorrectly identify someone as a potential churn. This could be acceptable by the company, because they would still be attempting to keep actual member who are poised to churn. Code and output can be found in Appendix A.

**Figure 1**



**References**

Iyyer, Shruti. (2019). *Churn Modelling* [Data set]. <https://www.kaggle.com/datasets/shrutimechlearn/churn-modelling?resource=download>

**Appendix A**

**Output**

Unit 10 Assignment / Module 6 Part 3 Competency Assessment Output

01/21/2024 12:57:28

Shape of Dataset:

(10000, 14)

Variance of CreditScore is 9341.860156575658

Variance of Age is 109.99408416841685

Variance of Tenure is 8.364672627262726

Variance of Balance is 3893436175.990742

Variance of EstimatedSalary is 3307456784.134512

X\_train Length: 8000

X\_test Length: 2000

LR Prediction Accuracy:

0.811

LR Confusion Matrix:

[[1526 69]

[ 309 96]]

LR Classification Report:

precision recall f1-score support

0 0.83 0.96 0.89 1595

1 0.58 0.24 0.34 405

accuracy 0.81 2000

macro avg 0.71 0.60 0.61 2000

weighted avg 0.78 0.81 0.78 2000

Predict with SVM Model:

accuracy: 0.7975

precision: 0.0

recall 0.0

SVM Classification Report:

precision recall f1-score support

0 0.80 1.00 0.89 1595

1 0.00 0.00 0.00 405

accuracy 0.80 2000

macro avg 0.40 0.50 0.44 2000

weighted avg 0.64 0.80 0.71 2000

RF Evaluate Model:

precision recall f1-score support

0 0.89 0.95 0.92 1595

1 0.73 0.51 0.60 405

accuracy 0.86 2000

macro avg 0.81 0.73 0.76 2000

weighted avg 0.85 0.86 0.85 2000

Accuracy Score:

0.864

**Code**

#######################################################  
# Author: Laurence Burden  
# For: Purdue University Global  
# IN402 - Modeling and Predictive Analysis  
#  
# Unit 10 Assignment / Module 6 Part 3 Competency Assessment  
#  
# Classification Model Selection  
###################################################################  
  
# Import packages  
import sys  
import pandas as pd  
from datetime import datetime  
from statistics import variance  
from sklearn.model\_selection import train\_test\_split  
from sklearn.preprocessing import StandardScaler  
from sklearn.linear\_model import LogisticRegression  
from sklearn import metrics  
from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score  
from sklearn.svm import SVC  
from sklearn import metrics  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.metrics import classification\_report, accuracy\_score  
import matplotlib.pyplot as plt  
  
# Ignoring warnings  
if not sys.warnoptions:  
 import warnings  
  
warnings.simplefilter("ignore")  
  
# Output Header  
print('Unit 10 Assignment / Module 6 Part 3 Competency Assessment Output\n')  
  
print(datetime.now().strftime("%m/%d/%Y %H:%M:%S"), '\n')  
  
# Import the dataset  
# Importing the dataset to a pandas DataFrame  
df = pd.read\_csv('Churn\_Modelling.csv')  
print('Shape of Dataset: ')  
print(df.shape)  
print()  
  
# Wrangle the data  
# Drop columns with no analytical value  
df = df.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1)  
  
# Convert the categorical columns into dummy columns  
# and drop the original categorical columns  
geography = pd.get\_dummies(df.Geography).iloc[:, 1:]  
gender = pd.get\_dummies(df.Gender).iloc[:, 1:]  
  
# Drop columns with non-numeric data  
df = df.drop(['Geography', 'Gender'], axis=1)  
  
# Join the dummy columns into the main dataset  
# Add columns with converted dummy values  
df = pd.concat([df, geography, gender], axis=1)  
  
# Split the dataset into target and feature subsets.  
X = df.drop(['Exited'], axis=1)  
y = df.loc[:, 'Exited']  
  
# Select features  
# Check the variance in the numeric variables  
creditScore = df['CreditScore']  
age = df['Age']  
tenure = df['Tenure']  
balance = df['Balance']  
estimatedSalary = df['EstimatedSalary']  
  
# Display the parameter variances  
print("Variance of CreditScore is %s " % (variance(creditScore)))  
print("Variance of Age is %s " % (variance(age)))  
print("Variance of Tenure is %s " % (variance(tenure)))  
print("Variance of Balance is %s " % (variance(balance)))  
print("Variance of EstimatedSalary is %s " % (variance(estimatedSalary)))  
print()  
  
# Split the dataset into training and testing subsets.  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)  
  
# Display size of training set  
print('X\_train Length: %s' % (len(X\_train)))  
  
# Display size of test set  
print('X\_test Length: %s' % (len(X\_test)))  
print()  
  
# Conduct feature scaling (required by SVM)  
# feature scaling is required by SVC  
sc = StandardScaler()  
X\_train = sc.fit\_transform(X\_train)  
X\_test = sc.transform(X\_test)  
  
# Model using Logistic Regression  
# Build the model  
lr\_model = LogisticRegression()  
  
# Fit the model  
result = lr\_model.fit(X\_train, y\_train)  
  
# Predict using the model  
prediction\_test = lr\_model.predict(X\_test)  
  
# Evaluate the model  
# Print the prediction accuracy  
print('LR Prediction Accuracy: ')  
print(metrics.accuracy\_score(y\_test, prediction\_test))  
print()  
  
# Display the confusion matrix  
print('LR Confusion Matrix: ')  
print(confusion\_matrix(y\_test, prediction\_test))  
print()  
  
# Display the classification report  
print('LR Classification Report: ')  
print(classification\_report(y\_test, prediction\_test))  
print()  
  
# To get the weights of all the variables  
weights = pd.Series(lr\_model.coef\_[0], index=X.columns.values)  
weights.sort\_values(ascending=False)  
  
# Model using SVM  
# Build the model  
svm\_model = SVC(kernel="linear")  
  
# Fit the model  
# Train the model  
svm\_model.fit(X\_train, y\_train)  
  
# Predict using the model  
print('Predict with SVM Model:')  
svm\_prediction = svm\_model.predict(X\_test)  
print()  
  
# Evaluate the model  
print("accuracy: ", metrics.accuracy\_score(y\_test, y\_pred=svm\_prediction))  
  
# Precision score  
print("precision: ", metrics.precision\_score(y\_test, y\_pred=svm\_prediction))  
  
# Recall score  
print("recall", metrics.recall\_score(y\_test, y\_pred=svm\_prediction))  
print()  
  
# Display classification report  
print('SVM Classification Report: ')  
print(metrics.classification\_report(y\_test, y\_pred=svm\_prediction))  
print()  
  
# Model using RandomForestClassifier  
# Build the model  
rf\_model = RandomForestClassifier(n\_estimators=200, random\_state=0)  
  
# Fit the model  
rf\_model.fit(X\_train, y\_train)  
  
# Predict using the model  
rf\_predictions = rf\_model.predict(X\_test)  
  
# Evaluate the model  
print('RF Evaluate Model: ')  
print(classification\_report(y\_test, rf\_predictions))  
print()  
  
# Display the accuracy score  
print('Accuracy Score: ')  
print(accuracy\_score(y\_test, rf\_predictions))  
print()  
  
# Display the accuracy score  
plt.figure(figsize=(12, 6))  
feat\_importances = pd.Series(rf\_model.feature\_importances\_, index=X.columns)  
feat\_importances.nlargest(10).plot(title="Accuracy Score", kind='barh')  
plt.show()