State Farm Insurance

Classification Exercise Modeling Summary

**Name**: Anitha Ganapathy

**Email id**: [Anitha.g19@gmail.com](mailto:Anitha.g19@gmail.com)

# Introduction

The task at hand was to use the sample dataset to build a binary classification model. The data provided was raw data which needed to be pre-processed before building a binary classification model that predict the label probabilities. Exploratory data analysis was done on the data based on which the pre-processing steps were built to clean the data. Baseline model used was a Logistic regression and many experiments were performed to build a better performing model. The complete exercise with the models and AUC score is listed here.

# Pre-Processing

The following steps were executed to clean the data:

* To avoid data leakage, the data was split into train and test with a 70% for the training data.
* The EDA steps started with how evenly the label was distributed. The finding was that the data was unevenly distributed with 85% to 15% for the labels 0 and 1.
* The datatypes distribution was found next to identify the numerical and categorical data.
* Depending on the finding the pipeline for the classification was built to clean the data and then go on to use SMOTE (***Synthetic Minority Oversampling Technique***) for handling the imbalanced dataset.

# Classification Models

The requirement was to build a baseline model of Logistic Regression model to predict the probabilities. Then move on to experiment with other models and find best model to predict. The comparison is based on the ROC AUC metrics. The list of the models and the ROC AUC score is:

# Graphical user interface, text, application Description automatically generated

# Comparison of the Modeling Approaches

The baseline model, Logistic regression did a descent job of the classification with AUC ROC value of 0.763847 but the Neural Network was a close competition with an AUC of 0.763615. Generally Neural network perform better at classification problem.

## Logistic Regression

1. The raw and dirty dataset are not even close to the cleaned dataset that we use for academic learning. Most cases the raw dataset would be imbalanced and may or may not be ready for machine learning. The datasets may not be linearly separable (SVM are used for linearly separable datasets) as in our case. Logistic regression works great with small nearly linearly separable datasets. The logistic regression model tries to predict the probability of the label that its trying to predict P(Y =1| Given\_X) based on the input features provided while training.
2. Compared to a neural network, Logistic Regression is a special case of a Neural Network with no hidden layers, that uses the sigmoid activation function and uses the softmax with cross entropy loss.
3. Logistic regression is a linear classifier model.

#### Pros

* Works great for small datasets
* Easy to perform cross validation as the dataset is small.

#### Cons

* The options to fine tune the model is very limited. The grid search can be performed with only the model parameters.

## Neural Networks

1. Neural Networks can be thought of as a stack of multiple logistic regression model that behave in a feed forward network learning from the previous network in the pipeline.
2. The loss or the error is then back propagated to the previous layers to learn from the previous run of the full stack of models.
3. The main objective is to minimize the loss and maximize the accuracy of model.
4. To make a neural network non-linear, we just need to add at least one hidden layer with a non-linear activation function, like a ReLU or a sigmoid.

#### Pros

* The neural network can work with ease when we want to work with huge, non-linear and data that has very complex relationship.
* Very easy to scale to build a complex network that is needed for the solution that we are trying to solve.
* The possibilities and options to fine tune the model is easy and unlimited.
* The weights of the network can be initialized using many methods, an example being kernel\_initializer="he\_normal"

#### Cons

1. The networks are very easy to overfit and hence need some kind of regularization like adding a dropout layer in between the stack of models.

# Best performing model

The neural network definitely is the best performing model, provided we take care of few things like data leakage, overfitting, batch size, initializer and many more.

# Further improvements

The model can be further improved performing some feature engineering. Few things that I think I could have done are:

* I noticed that the state of California had the most data from that state, I could have created a feature that says California\_no\_yes.
* The months of December, July, August, January had the most data again, I could have created a feature as max\_4\_months\_no\_yes
* The automobile brand Ford had again the most data in that bucket, so new feature ford\_no\_yes.
* I could have created the stats like the mean, median of the numerical data for each row and made it available as a feature.
* Allstate and progressive insurance has the majority cut of the data, so allstate\_prog\_no\_yes feature.

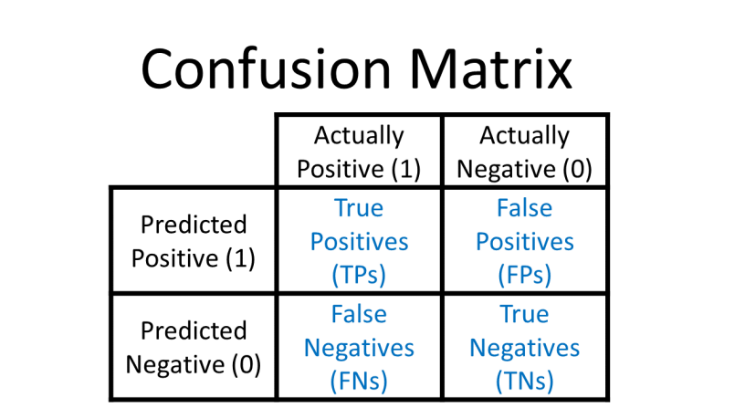
# Estimate of AUC for the test set for each model.

Graphical user interface, text, application

Description automatically generated

# How you would demonstrate to a business partner that one model is better than the other without using a scoring metric.

I would prefer to use the confusion matrix to explain which model would work better against the dataset. Its is easy to explain the term like false positive and true positive.



In case the metrics is not allowed to be used to explain to a business partner, then I would start by saying that the percentage of the success and failures with respect to the predictions that a model would make.

I would say that a model A will make less mistakes than model B.