**Introduction**

Credit card fraud is a problem for many people across the United States,

with 45,428 cases of credit card fraud reported to the Federal Trade Commission in 2017.

With datasets available today, neural networks can be trained to identify trends

in fraudulent transactions and non-fraudulent transactions.

The output of these networks can be used to notify consumers of

suspicious activity on their accounts and prevent further losses.

**The Dataset**

For this case study, the dataset consists of over 280,000 credit card transaction data points accumulated over 2 days in European countries. Each data point holds basic information about the transaction as well as obfuscated data resulting from a PCA transformation, which can be seen in Table {sample-points}. The data prefixed with a “V” is the transformed data.

One of the interesting metrics of the dataset is the ratio of fraudulent to non-fraudulent transactions. With fraudulent transactions only making up 0.172% of the 280,000+ transactions, this poses a challenge for a neural to make accurate identifications for fraudulent cases. The fraudulent cases are also the most interesting result from the network, as simply identifying the 99.82800% of non-fraudulent cases is trivial.

**Methods**

**Deep Learning Framework**

TensorFlow r1.8 is used to carry the back end of the neural network, providing definitions for layers, optimization functions, back propagation, and running of the network. The Pandas package is used heavily for the data pre-processing and matrix manipulations.

**Data Pre-Processing**

The dataset is provided in a .csv format, so the first pre-processing step is to read the data points

into the Pandas package. Once the data is read in, the “Class” property is changed into two properties, named “Fraud” and “NonFraud” for easier determination of the output of the network. The fraudulent and non-fraudulent cases are then split into two lists for division of training and testing data. 75\% of each case is used as training data, while the remaining 25\% is used for testing data. Both the training and test data are further divided into various separate lists for result processing later in the network.

To overcome the skewed nature of the dataset, augmentation of the fraudulent cases is required. Augmentation was accomplished by sampling the fraudulent cases in both the training and testing sets 600 times, to obtain a ratio between the cases of around 50:50.

**Neural Network Architecture**

**Layer Definitions**

The neural network is defined by an input layer, three hidden layers, and an output layer, as can be seen in Figure {network-figure}. Each layer also includes a bias input. The input layer is the width of each data point (30), with each layer doubling the width of the previous layer until the output layer, which is of width 2. A fully connected network approach is also used.

**Weight and Bias Initializations**

Each layer’s weights are initialized using a random generated number from a normal distribution with a standard deviation of 0.1 and centered around zero. The bias weights are initialized in a similar fashion.

**Activation Functions**

Activation functions used between layers are rectified linear units (RELU), as well as a linear activation function leading to the output layer.

**Output Interpretation**

The cost function used for interpretation of the network output is mean squared error (MSE). The accuracy of the output is determined by selecting the index of the output node with higher activation and comparing it to index of the output node of the expected result.

**Experiments**

**Sauces**

FTC Report Number:

https://www.ftc.gov/policy/reports/policy-reports/commission-staff-reports/consumer-sentinel-network-data-book-2017/report-categories