Credit Card Fraud Detection With a Feed Forward Neural Network

Abstract

Today, credit card transactions are a considerable part of how purchases are made. Due to the ability for credit cards to be able to make large purchases, some have looked into ways to steal a person's credit card in order to make illegal purchases. This is called credit card fraud. Banks have put in place techniques in order to minimize the damage of credit card fraud such as rule-based systems, anomaly detection, and predictive model systems. This paper will examine the effectiveness of differently-tuned classification supervised learning models in predicting whether a purchase that was made was fraudulent or not. A European dataset was used of purchases made with credit cards that were found to be either fraudulent or not for training the model. The model would look for patterns in the features of the data in order to identify whether a purchase was fraudulent. As a baseline, a binary classification model was used for comparison. Then, seven differently-tuned feed-forward neural networks were trained on the data for fraud detection. Overall, the models were effective at detecting fraud. A logistic regression model had an accuracy of 96.46%. Using feed-forward neural networks, the results could be improved (1st model: 99.79%, 2nd model: 99.92%, 3rd model: 99.95%, 4th model: 99.95%). The high performance of these models displays that fraud detection can be accomplished accurately. By training models on large datasets, banks can detect when fraud happens and minimize the damage to both themselves and the user.

Background

In the modern day, credit cards have become an integral part of financial transactions in daily life. In the U.S. alone, 84 percent of the adult population have at least one credit card, which adds up to a total of 572 million credit cards in the population. In place of physical cash, credit cards are used for purchases. This is done by establishing a line of credit with a bank. Credit is where a person borrows money up to a maximum threshold for purchases, which is paid back by the person each month. As you establish trust with the bank through regular payments, the maximum threshold increases. While credit cards are an effective way to make transactions, there can be major consequences when stolen.

When a credit card is stolen, it is considered credit card fraud. Fraud occurs when a person physically or virtually steals an individual's credit card account to make unauthorized purchases. Some examples of methods for stealing a credit card include: physically stealing the card, using a skimming device in public places like gas stations, and hacking into financial databases. Unlike cash where people on average have a smaller amount of money on hand, credit cards can make a large amount of purchases before reaching the spending threshold. For personal victims, this can result in large amounts of unwanted costs and reduce the bank's trust in their credit if fraud goes unnoticed. As for banks, when fraud is discovered, they are liable to pay for the credit without reimbursement. They will also have to contact the police for investigation and settle disputes in charges with credit card users. To limit the damage of credit card fraud, techniques have been developed for detecting when fraud occurs.

Current Solutions

Current solutions for detecting and preventing fraudulent activities use multiple techniques. Each current solution has its strengths and weaknesses. The current solutions are Rule-based systems, Anomaly Detection, and Predictive Model Systems.

Rule-based systems are like the first line of defense when dealing with fraud. Rule-based systems rely on predefined rules to identify suspicious transactions, which flags them for manual review. These rules have criteria like transaction amounts that go over a certain limit. These are great for dealing with fraud patterns that already exist but struggle with new and evolving types of fraud.

Anomaly detection uses machine learning algorithms to detect deviations from an individual's typical spending behavior, analyzing patterns to uncover anomalies. It builds a profile of a user's typical behavior and flags any activity that goes away from the norm. It's similar to systems that raise an alarm when it spots something out of the ordinary. This could be a sudden increase in spending or a purchase that was made in an unfamiliar place. Its ability to learn and adapt makes it great against evolving fraud schemes.

Predictive model systems use analytics and historical data to create real-time fraud detection models, generating alerts for specific fraudulent activities. Predictive models analyze patterns from past transactions, predictive models can find potential fraud before it happens. This model continually learns and improves, which makes them great over time. This approach can be important and Predictive models are great for finding complex fraud schemes that might not be immediately obvious. These systems often utilize both supervised and unsupervised machine learning approaches to enhance their accuracy and effectiveness in combating fraud.

Introduction

Many of the techniques used above use a form of machine learning. Machine learning is the ability for computers to identify patterns in data and use them to make predictions. Models follow two types of paradigms, supervised and unsupervised. Supervised models is an algorithm trained on labeled data that consists of features in order to predict an outcome. Unsupervised learning is a machine learning algorithm for finding patterns in data without labeled inputs. A common type of supervised learning model is called classification. This is where the model predicts the category of a data item based on the pattern analysis of featured data. This paper will be utilizing classification-supervised learning models in order to predict whether a purchase is fraudulent or not. It will examine the effectiveness of differently-tuned classification models based on the accuracy and loss function of each model.

Methods

The dataset used contains information about credit card transactions from European cardholders in 2023, as well as whether or not they were fraudulent. This dataset has 31 features in total and 568,630 observations. The features include the unique ID of each transaction, the transaction amount, the class indicating whether or not that transaction is fraud and 28 distinct features denoted as V1-V28. V1-28 are different types of metadata for each transaction, such as location and timestamp information, which helps in understanding the contextual aspects of each transaction. For privacy and security reasons, V1-28 are anonymized, which is why they appear as normalized integer values. Closer inspection of the

V1-28 features does indeed show that they are normalized, as viewing their mean and standard deviation shows that they correspond with the statistical requirements of being classed as normalized. However, some of their histograms, shown below in Figures 1 and 1.1, show a good normal distribution while others are a bit skewed. For that reason, these features were standardized regardless during preprocessing. A class imbalance was checked for as well. The number of fraudulent and non-fraudulent transactions was evenly split 50%. The features used for prediction were V1-28 and the transaction amount. ID was not used due to not being a meaningful feature.

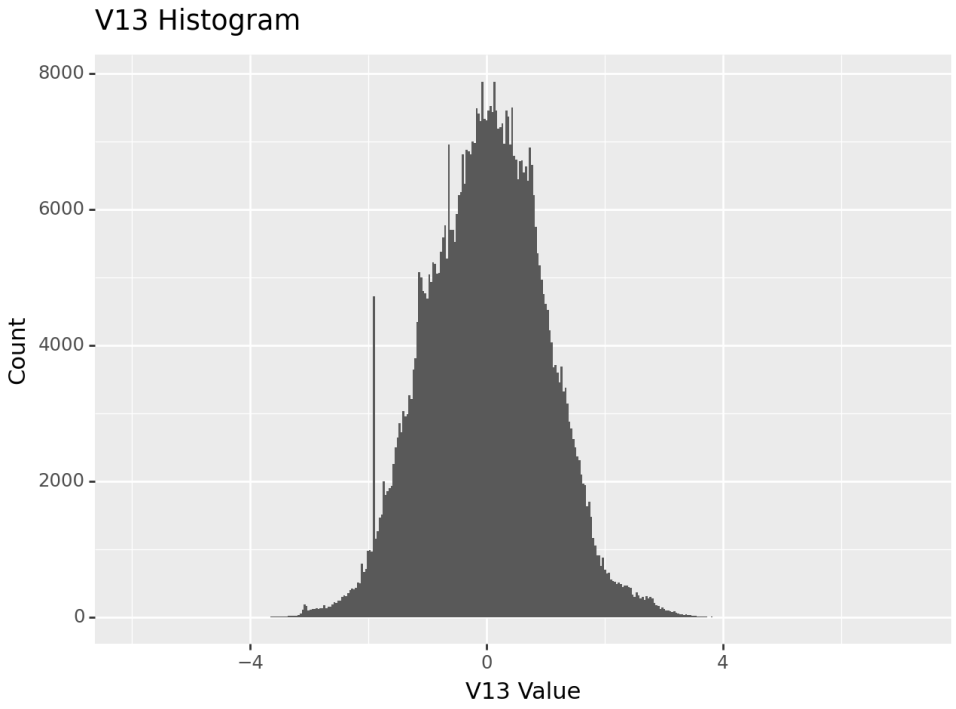

Figure 1 - Example of Feature with Good Normal Distribution

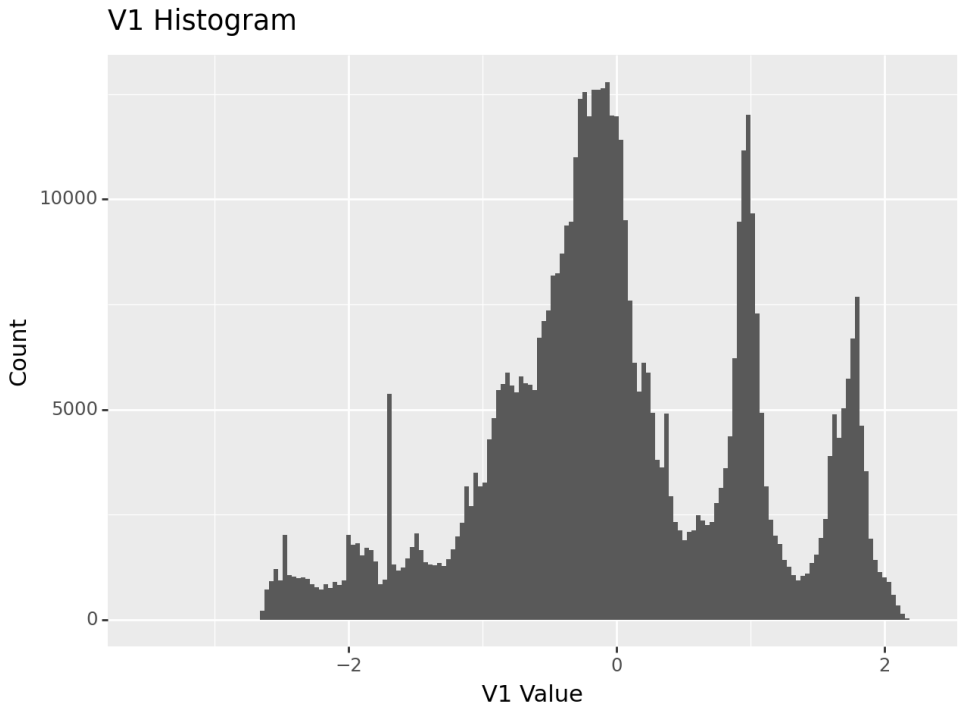

Figure 1.1 - Example of Feature With a Somewhat Skewed Distribution

Due to the dataset using a binary classification approach, this simplifies the complex array of transaction data into two primary categories. Transactions are labeled as “0” when they are non-fraudulent, and transactions labeled as “1” when fraudulent. The binary classification system is important for improving the efficiency and accuracy of fraud detection. It simplifies the process of flagging transactions that warrant further investigation, allowing for a more focused and effective approach to identifying and preventing fraudulent activities.

A total of seven different feed-forward neural network models were created and tested, as well as a basic logistic regression model for comparison. The initial three models were for getting a baseline both in terms of model architecture and performance. The fourth model was the final model overall based on what was learned from the previous three models. The remaining three models were applied regularization to the final overall model to see if they would have any beneficial effects on the model’s performance. Due to the abundance of training data, transfer learning was never utilized.

The first model was very basic, having only three layers. Two intermediate layers with 16 nodes each and ReLu activation functions, and a final binary classification layer with a sigmoid activation function. The model compiled with RMSP as the optimizer, binary cross entropy as the loss function, and accuracy as the primary performance metric. The model was trained for 20 epochs with a batch size of 256. This first model provided the initial baseline for how the future model architectures would be constructed.

The second model focused on increasing the model complexity of the initial model, as well as optimizing it a bit. Two more intermediate layers were added, each having 32 nodes. Every layer used the same activation functions as before. The optimizer was changed to Adam from RMSP, and the model was trained for 25 epochs.

The third model ramped up model complexity even further, for reasons to be discussed in the results section. The third model had eight intermediate layers, each with 256 nodes, and a final binary classification layer; activation functions still being the same as the previous models. The model used the Adam optimizer as well and trained for 20 epochs with a batch size of 512.

The fourth model, or the final overall model, had three intermediate layers and a classification layer. The intermediate layers had 32 nodes each and used the ReLu activation function. The binary classification layer used the sigmoid activation function. The model was compiled with the Adam optimizer, binary cross entropy as the loss function, and accuracy as the main performance metric. The model was trained on 20 epochs with a batch size of 512. The reduced model complexity, but not too reduced, and larger batch size allowed this final model to achieve the goal of fast prediction time and high accuracy. The other three models that were created and trained on were primarily for testing regularization methods on the final model. The first test was with dropout and L1 regularization, a more aggressive regularization method. The second test was with dropout and L2 regularization, a less aggressive regularization method. The final test was applying dropout exclusively on the model.

Results

The first unoptimized feed forward model achieved a test loss and accuracy of 0.009 and 99.79% respectively. A basic logistic regression model achieved a test accuracy of 96.46% accuracy. So why use a feed forward neural network when a basic model can achieve comparable results in almost instantaneous speeds? At face value, a 3% difference in accuracy may not seem like a lot. However, given the large scale of credit card fraud damage, the 3% difference in performance is justifiable; especially when considering the costs associated with credit card fraud. That 3% means less customer disputes, hundreds of thousands or even millions of dollars in damages saved, and better security and confidence with both the credit card users and companies. Also to be considered, is that the first model is very basic and unoptimized, so there's room for improvement.

Looking at the first model’s performance more closely reveals that the model was unable to overfit, indicating more room for complexity. This is shown in figure 2 and 2.1 looking at the model’s training and validation loss and accuracy over 20 epochs. The model trained very fast on the dataset and had diminishing returns early on.

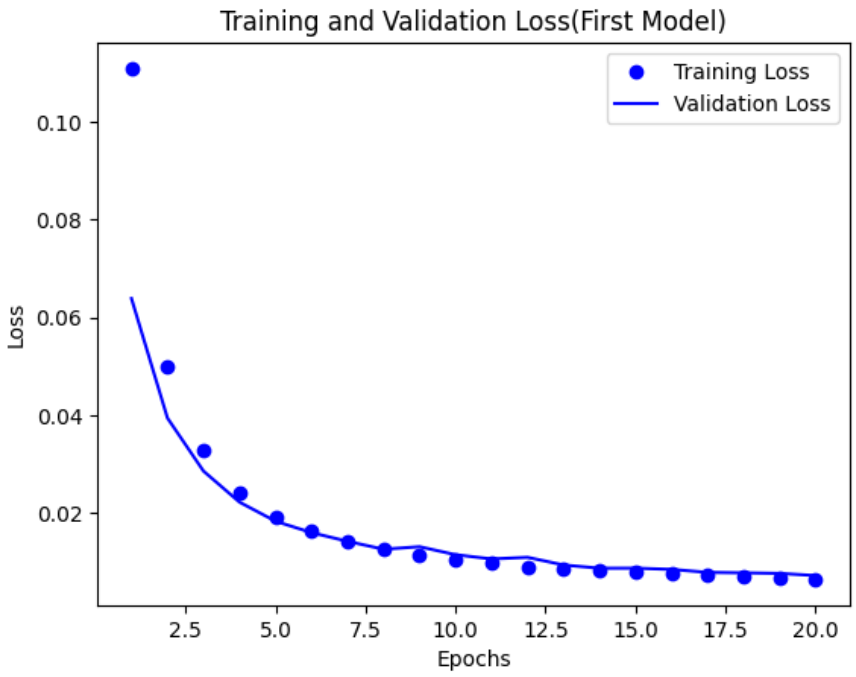

Figure 2 - First Model Training and Validation Loss Over 20 Epochs

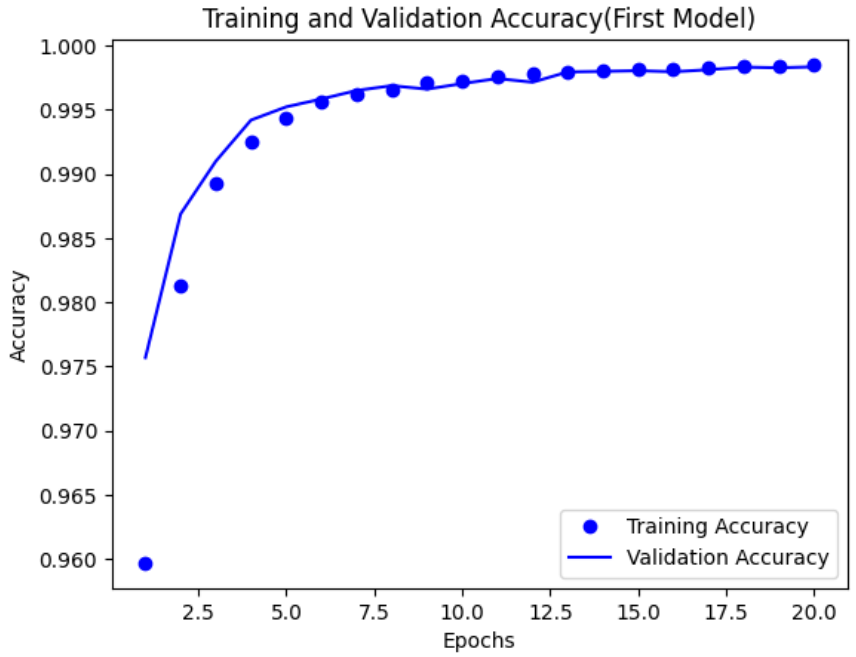

Figure 2.1 - First Model Training and Validation Accuracy Over 20 Epochs

The second model focused on increasing model complexity. The test loss and accuracy ended up being 0.003 and 99.92% respectively. The model almost fully trained on the data in even less epochs, and reached diminishing returns even earlier as well as shown below in

figure 2.2 and 2.3. Due to this model still not being able to overfit, the third model heavily ramped up model complexity to see if the model is capable of overfitting.

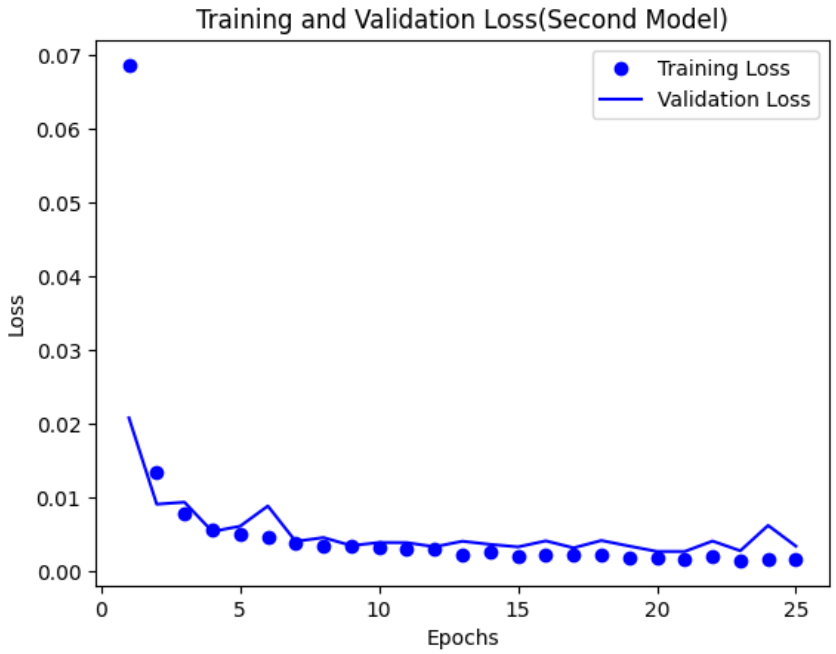

Figure 2.2 - Second Model Training and Validation Loss Over 25 Epochs

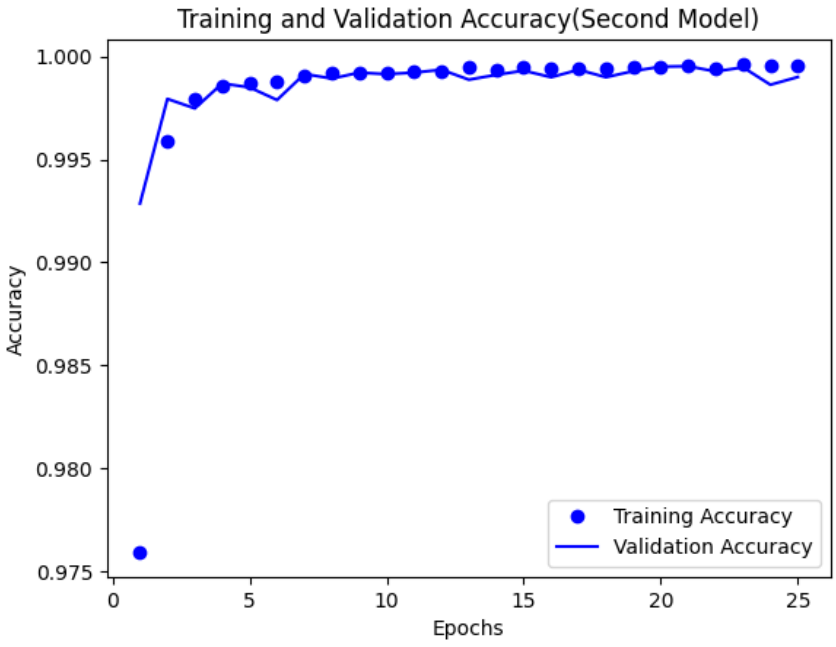

Figure 2.3 - Second Model Training and Validation Accuracy Over 25 Epochs

The third model achieved a test loss and accuracy of 0.002 and 99.95% respectively. Figures 2.4 and 2.5 display the training and validation loss and accuracy respectively over 20 epochs. The metrics are a lot choppier overtime, compared to the previous models, and the

model is still unable to overfit. The inability for the model to overfit is likely due to the large abundance of training data, allowing it to effectively learn the patterns of the data. Also, increasing model complexity led to very minimal returns to model performance. Because of this, the objective shifted to creating a model with a balance between high performance and fast prediction speed to allow for real-time detection; target accuracy being 99.9%.

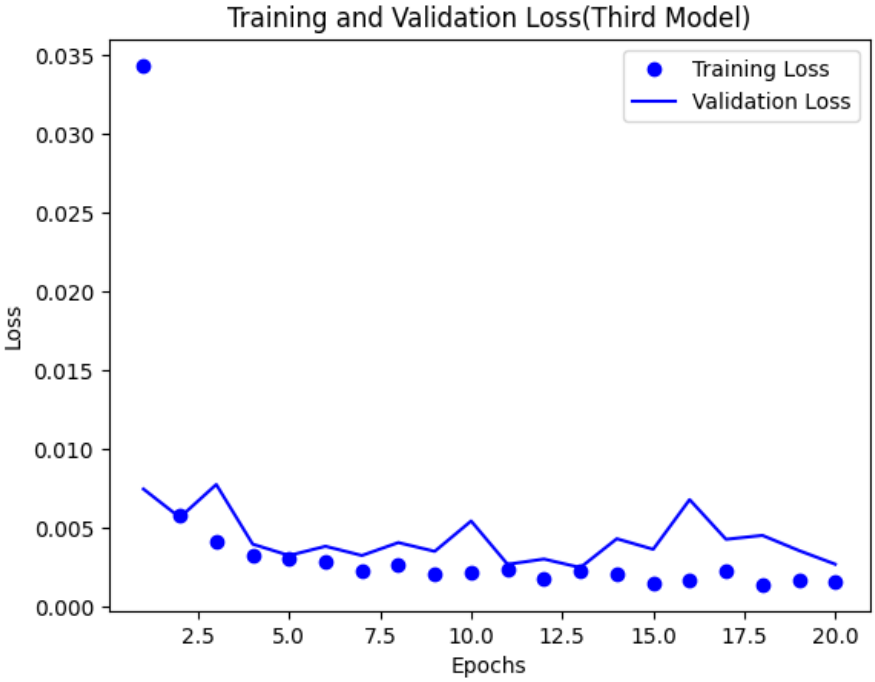

Figure 2.4 - Third Model Training and Validation Loss Over 20 Epochs

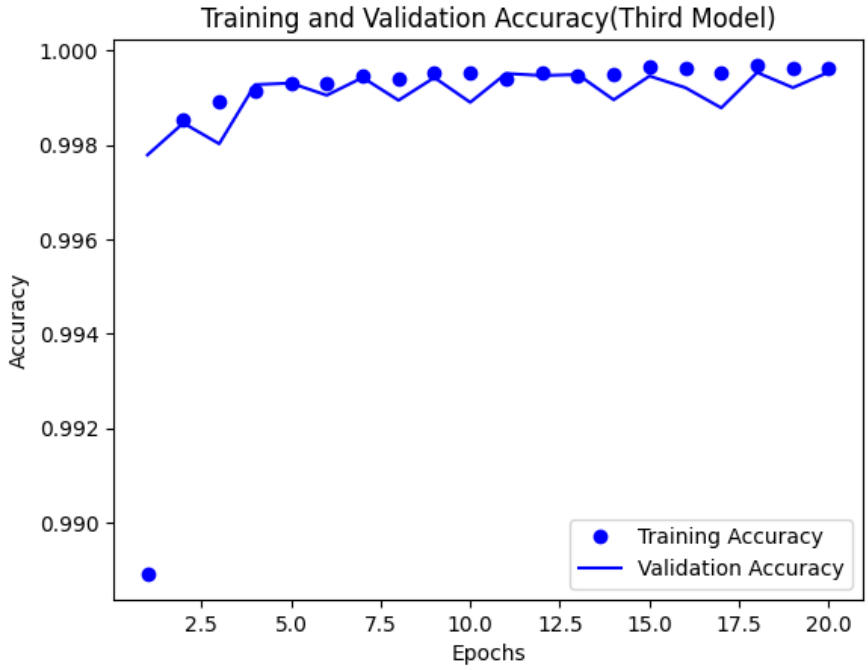

Figure 2.5 - Third Model Training and Validation Accuracy Over 20 Epochs

The final overall model had a test loss and accuracy of 0.002 and 99.95% respectively. The model's prediction time was 5 seconds, however, this was for the entire test dataset so it would likely be much faster for only one observation. Regularization was not previously applied on any models, mainly due to lack of overfitting, but was done so for this final model to see if it would have any positive effects. Table 1 below shows the various performance metrics of different regularization tests compared to the base final model.

Table 1 - Summary of Performance Metrics for Final Model and Regularization Tests

	Test Loss	**Test Accuracy**	**Prediction Time**
Base Final Model	0.002	99.95%	5s
Dropout + L1	0.215	96.05%	9s
Dropout + L2	0.079	98.64%	8s
Dropout Only	0.005	99.91%	7s

Overall, regularization didn't have any positive effect on model performance, so it was never considered for the final model. This makes sense, especially considering the model is unable to overfit on the data, so it is only natural for regularization to not help with model performance. Perhaps the biggest drawback of regularization in this case, if used, would be the longer prediction time; possibly preventing real-time detection. Something important to note as well is that all prediction time tests were done on Google Colab. If the prediction time tests were performed on stronger hardware, then the prediction time would likely be instant. However, Google Colab works as a useful baseline, as it likely wouldn't be possible(or cost-effective) to implement potentially expensive hardware on every credit card transaction device.

Conclusion

Overall, the feed-forward neural network model performed exceptionally well in predicting credit card fraud. Having a dataset with a large number of observations assisted in the model's performance as well, as it allowed the model(s) to learn what is and isn't credit card fraud and their respective patterns. Given that the final model is able to predict at a relatively fast speed, this allows for real-time detection capabilities with very high accuracy. Going forward, it would be useful to analyze other performance metrics to see if there is an imbalance in fraudulent or non-fraudulent predictions. Along with this, further speeding up prediction times would be further beneficial. One should consider that not every credit card transaction device will have advanced hardware such as GPUs installed into them.

References

Balaban, D. (2023, May 1). *Top 10 Credit Card Fraud Detection Solutions in 2023*. CybeReady. https://cybeready.com/top-10-credit-card-fraud-detection-solutions-in-2023

Clearly Payments. (2023). *Credit Card Fraud in 2023 - Credit Card Processing and Merchant Account*. Clearly Payments. https://www.clearlypayments.com/blog/credit-card-fraud-in-2023/

Elgiriyewithana, N. (2023). *Credit Card Fraud Detection Dataset 2023* [Dataset containing credit card transactions from European cardholders in 2023.]. Kaggle. https://www.kaggle.com/datasets/nelgiriyewithana/credit-card-fraud-detection-dataset-2023

Hare, V. (2022, February 17). *The Cost of Credit Card Fraud - tokenex*. TokenEx. https://www.tokenex.com/blog/vh-the-cost-of-credit-card-fraud/

Kovalenko, O. (2023). *Credit Card Fraud Detection Using Machine Learning*. SPD Technology. https://spd.tech/machine-learning/credit-card-fraud-detection/

White, A. (2023, November 14). *Credit Card Fraud: How It Happens and How to Protect Yourself*. CNBC. https://www.cnbc.com/select/credit-card-fraud/