Capstone Proposal: Maxfield Thompson

Domain Background:

The detection of anomalies is important to many industries and businesses. Anomalous behavior, search engine requests, or website history could suggest criminal motives. Anomalous activity in a company's server logs could be the activity of a hacker or an intruder. And anomalous activity in credit card purchases could be evidence of identity theft.

Using machine learning to detect anomalies is challenging because in the classic supervised learning environment, labeled anomalies are usually very rare. This makes a class imbalance problem when training. Furthermore, in many situations people want to find anomalies without having first identified any anomalies. That is, people don't have any labeled anomalies in their datasets, but instead have to find them without any previous examples. How then does one find anomalies using machine learning if no anomalies are labeled? This leads me to the point of this project: unsupervised anomaly detection using autoencoders.

Here are some papers on the subject:

https://link.springer.com/chapter/10.1007/978-3-319-13563-2_27

http://dl.acm.org/citation.cfm?id=2689747

http://digital-library.theiet.org/content/journals/10.1049/el.2016.0440

Problem Statement:

I will use deep learning, specifically an autoencoder, to detect anomalies in credit card purchases. This will be unsupervised anomaly detection. By unsupervised, I mean that I will not use any of the anomaly labels in my training. I will, however, use the labels to help check how well my autoencoder is performing. In a real world situation, where I didn't know what was anomalous, I wouldn't be able to check my autoencoder in this way, but for sake of this project, I need some type of metric to check how well I'm doing.

In this problem, I can use precision and recall as metrics to test how well my autoencoder is performing. These metrics are quantifiable, measureable, and replicable. Lastly, my autoencoder can be reproduced by using my code, so anyone who's interested could reproduce my results.

Dataset and Inputs:

https://www.kaggle.com/dalpozz/creditcardfraud

I will be using a dataset that contains credit card transactions from European cardholders in September 2013. In this dataset there are 284,807 transactions and 492 incidents of fraud. This means 0.172% of transactions are fraudulent (anomalous). The specific features of this dataset have been transformed by a PCA transformation. This means that I don't know the meaning of 28 of the features, they are simply labeled V(then the feature number) with some numerical score. However, the features 'Time' and 'Amount' have been left in the dataset unchanged. Finally, there is a feature 'Class' which labels the transaction as fraudulent or not, but I will only be using this feature to check my results not for training the model.

Benchmark Model:

First, I can use the labels in my dataset as a reference for how well my model is performing. But I can also use basic supervised methods as a check for how well my model is performing. However, the supervised methods will most certainly perform better. I am tackling this problem in an unsupervised way (rare for this dataset), and my results will be different from supervised methods. I am trying to solve a much harder problem than the classic supervised approach. Never-the-less I can still use other's supervised methods as a comparison, and also use the labels as a means of checking my performance.

Evaluation Metrics:

The evaluation metric the people who released the dataset recommend is using the Area Under the Precision-Recall Curve (AUPRC). Furthermore, they say that confusion matrix accuracy is not appropriate for this problem. So I will take their word for it and use AUPRC as my evaluation metric.

Project Design:

To begin the project, I will first examine the distribution of all my variables to see if any of them are skewed. If they are, I will scale the in a way that encourages a more normal distribution. Next, I'll check the range of my variables and see if any numerical scaling is needed. Then, as pure exploratory analysis, I'll look at the covariance matrix of my variables and visualize these results to see how my variables are correlated. Once all that preprocessing is complete, I will begin to build the autoencoder.

For the autoencoder, I will input each one of these transaction vectors into the network and train the model to reproduce the input after it has gone through a bottle neck in the network. Then I will measure the reconstruction error of this outputted vector as a metric of anomalousness. Finally, I will set some type of threshold using statistical methods to decide what level of reconstruction suggests a given transaction is anomalous. This process will be iterative as there are many parameters to optimize in an autoencoder. I will grid search through many types of optimizers, activation functions, loss functions, compression layer sizes, and total network structures to find the optimal autoencoder to complete this task. Finally, I'll use AUPRC as my evaluation metric to determine how well my model did.