International Journal of Scientific Research & Engineering Trends
Volume 6, Issue 6, Nov-Dec-2020, ISSN (Online): 2395-566X

Credit Card Fraud Detection using Autoencoders

Saatwik Bisaria, Aryan Aditya Singh, Sarita Yada
saatwik.bisaria@gmail.com, aryanadi2017@gmail.com,
Department of Information Technology
Bharati Vidyapeeth College of Engineering
New Delhi,India

***Abstract*- There are certain characteristics of fraudulent transactions that differentiate them from legitimate ones. Machine Learning algorithms recognize patterns in the data points which allow them to detect fraud transactions from legitimate ones, based on thousands of pieces of information, that sometimes may seem completely unrelated to a human being. In Machine Learning, problems like fraud detection are usually framed as classification problems —predicting a discrete class label output given a data observation. Autoencoders are special type of neural network architectures in which the output is same as the input. Autoencoders are trained in an unsupervised manner in order to learn the extremely low-level representations of the input data. These low-level features are then deformed back to project the actual data. An autoencoder is a regression task where the network is asked to predict its input (in other words, model the identity function). These networks have a tight bottleneck of a few neurons in the middle, forcing them to create effective representations that compress the input into a low-dimensional code that can be used by the decoder to reproduce the original input.**

***Keywords*- Autoencoders, Fraud Detection, Neural Network, Unsupervised Learning**

I. INTRODUCTION

With the evolution of modern technology, credit card fraud is soaring and has become an easy point of target for fraud. Credit card fraud is a critical issue in financial services, causing losses of billions of dollars every year. Due to the lack of confidentiality and eminently imbalanced publicly available data sets, the design of fraud detection algorithms is a rigorous task and there is a lack of real-world transaction data sets. In this paper, we apply unsupervised machine learning algorithms to detect fraudulent credit card transactions using data sets. In addition, we compare and discuss the performance of autoencoders in the literature for the classifier implemented in this paper.

Credit card fraud can be done in many ways- Using lost or stolen cards, by searching or stealing data from the seller, production of false or counterfeit cards by cloning the original card, by modifying the magnetic strip present in card containing user data by phishing. With the unrelenting advancement of fraud strategies, it is crucial to develop effective models to combat these frauds only at their initial stage. But the major challenge to develop such a model is that the number of fraudulent transactions among the total number of transactions is very small and therefore the task of finding a fraudulent transaction among the genuine transactions is tedious. Detecting fraud is a complex computational task. The number of parameters that need to be selected, grouped and classified is huge and the classification of parameters will decide the success any fraud detection techniques.
In addition, transactions cannot be purely classified as fraud because actual fraud existing systems; just find the probability that the transaction is a scam based on an extensive study of customer behavior, spending habits and also analyze previously committed frauds and various other features and parameters to observe them their patterns. Hence, the main problems that every credit card fraud detection system face is that they have one a very limited period of time in which to perform a decision to designate a transaction as fraudulent or genuine; and, it must process a huge number of parameters in training and decision making.

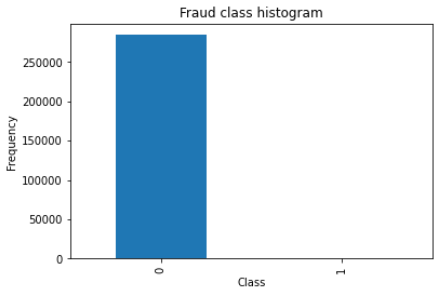
Fraud detection involves monitoring the activities of user groups to evaluate, perceive or avoid any reprehensible behavior, including fraud, intrusion and illegal payments. This is a very relevant problem that requires the attention of communities such as machine learning and data science, which can automate the solution to the problem. From a learning perspective, this problem is particularly challenging because it has various factors such as class imbalance. The number of valid transactions goes far beyond the fraudulent ones. Similarly, the transaction data points change their statistical properties over time. Some of the currently used approaches to detection of such fraud are:
• Artificial Neural Networks
• Fuzzy Logic
• Logistic Regression
• Support Vector Machines
• Bayesian Networks
• K-Nearest Neighbor.

1. Dataset

This method is commonly used to model and analyze data with small sample sizes. Unlike parametric models, nonparametric models do not require the modeler to make any assumptions about the distribution of the population, and so are sometimes referred to as a distribution-free method.

The datasets contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

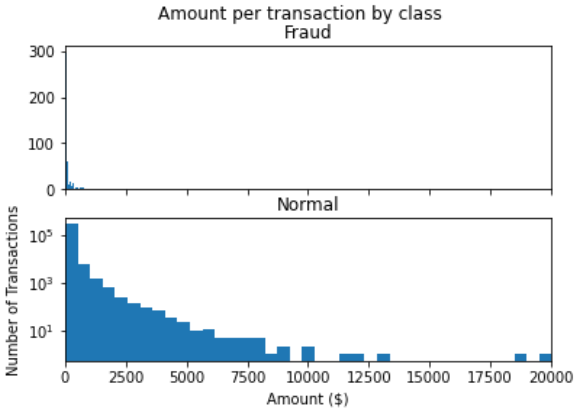
This dataset is highly unbalanced. Since providing transaction details of a customer is considered to issue related to confidentiality, therefore most of the features in the dataset are transformed using principal component analysis (PCA). V1, V2, V3,..., V28 are PCA applied features and rest i.e., ‘time’, ‘amount’ and ‘class’ are non-PCA applied features.



Put here fig.title.name

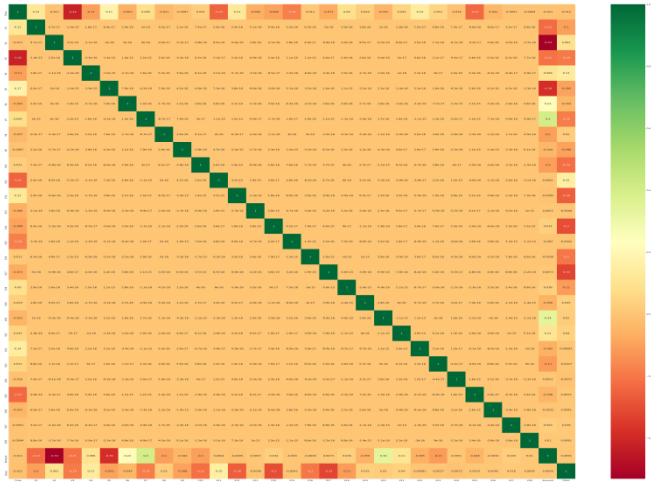
Histogram depicting the imbalance of classes in the dataset.

International Journal of Scientific Research & Engineering Trends
Volume 6, Issue 6, Nov-Dec-2020, ISSN (Online): 2395-566X



Put here fig.title.name

Histogram depicting amount of transaction for each class of transaction – Normal and Fraud in the dataset.



Put here fig.title.name

This figure is a correlation heatmap. This is a monochromatic scale representation which shows a 2D correlation matrix between two or more than two features present in the used dataset.

2. Unsupervised Classification

Unsupervised classification techniques are based on the analysis of data and categorization the data for the same samples. The goal in such unsupervised learning problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine how the data is distributed in the space, known as density estimation.

In some pattern recognition problems, the training data consists of a set of input vectors x without any corresponding label values. To put forward in simpler terms, for a n-sampled space with data points *X1* to *Xn,* class labels are not provided for each sample point. Hence, unsupervised learning is used to understand the structure of the data points, create data distribution to summarize and identify the classes of data. Unsupervised Learning can be classified into two categories:

2.1 Parametric Unsupervised Learning

In this case, we assume a parametric distribution of data. It assumes that sample data comes from a population that follows a probability distribution based on a fixed set of parameters. Parametric Unsupervised Learning involves construction of Gaussian Mixture Models and using Expectation-Maximization algorithm to predict the class of the sample in question. This case is much harder than the standard supervised learning because there are no answer labels available and hence there is no correct measure of accuracy available to check the result.

2. 2 Non-parametric Unsupervised Learning

In non-parameterized version of unsupervised learning, the data is grouped into clusters, where each cluster denotes something about categories and classes present in the data. This method is commonly used to model and analyze datasets with small sample sizes. Unlike parametric models, nonparametric models do not require any assumptions about the distribution of the population, and so are sometimes referred to as a distribution-free method.

II. METHODOLOGY

Autoencoder is an unsupervised artificial neural network that learns how to efficiently compress and encode data then learns how to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible. This neural network model is trained to attempt to copy its input to its output. Internally, it has a hidden layer *h* that describes a code used to represent the input. The network may be viewed as consisting of two parts:

1. An encoder function that produces latent representation

$$h=f(x)$$

2.A decoder function that produces a reconstruction

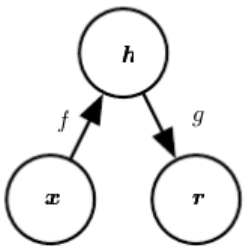
$$r=g(h).$$

If an autoencoder succeeds in simply learning to fit

$$g(f(x)) =x$$

every where, then it is not useful. Instead, autoencoders are designed to be unable to learn to copy perfectly. Usually they are restricted in ways that allow them to copy only approximately, and to copy only input that resembles the training data. Because the model is forced to prioritize which aspects of the input should be copied, it often learns useful properties of the data.

The general architecture of autoencoders is presented in figure given below.

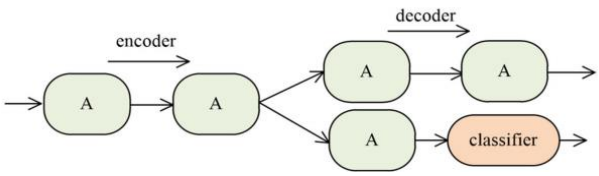


Put here fig.title.name

The general structure of an autoencoder, mapping an input x to an output (called reconstruction) *r* through an internal representation or code *h*. The autoencoder has two components: the encoder f (mapping x to h) and the decoder g (mapping h to r).

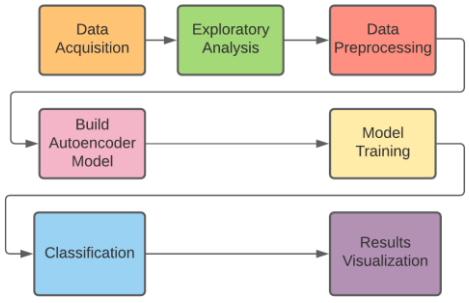
We created an autoencoder model in which we showed the model only one class of records. The model tried to learn the best representation of that class. The same model generated the representations of that class and we expected them to be different from the other class.

To aid our methodology we used Keras, an open-source neural-network library written in Python. It is capable of running on top of TensorFlow. We chose Keras as it follows best practices for reducing cognitive load: it offered consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear and actionable feedback upon user error.



Put here fig.title.name

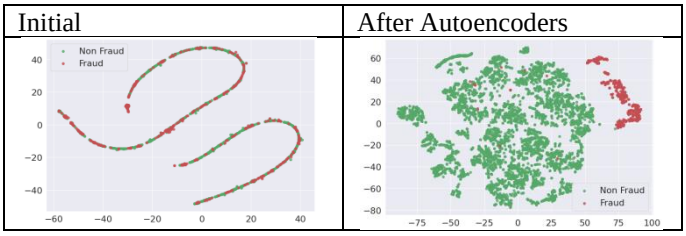
III. IMPLEMENTATION



Put here fig.title.name

We import the python modules and loaded the dataset. We performed exploratory analysis on our dataset to obtain valuable insights on the data. This included checking for null values, plotting histograms and correlation heatmap to represent the relation of variables with the target class. We preprocessed the data to work better with the neural networks.

After preprocessing, the data and converting it into usable form, an autoencoder model is built with encoder input network and a decoder output network. The encoder network generates a latent representation of the input data and the decoder layer generates an output. The features were fed into the autoencoder network of the single class to make it able to differentiate between two classes. The model learned the best representation of that class and generated the representations of that class and we visualized the results using T-SNE plot to allow autoencoder model to separate fraud transaction from normal transactions, which are represented in different colors.



Put here fig.title.name

Figure denoting the initial representations v/s the output latent representations of autoencoders. We can observe that now fraud and non-fraud transactions are pretty visible and are linearly separable. Hence, we have successfully separated fraudulent transactions with the normal ones.

IV. CONCLUSION AND FUTURE WORK

It is clear that credit card fraud is a serious crime. In this paper we provided an approach for fraud detection using of autoencoders and we finally observed that use of autoencoders is a viable and feasible option in the detection of credit card fraud. As for the future work, preparing a balanced dataset with unhidden features would make this more effective as it would enable us to differentiate between various features and their impact on deciding whether the transaction is fraud or not.

REFERENCES

The following resources were referred to while making this project.

1. Zaslavsky V. & Strizhak A. 2006. 'Credit card fraud detection using self-organizing maps'. Information and Security, 18; 48-63.
2. Thomas, L.C., Edelman, D.B., & J.N Crook. 2002. Credit Scoring and its Applications, SIAM Monographs on

Mathematical Modeling and Computation, Philadelphia. European e-Business Market Watch. 2005. ICT Security, e-Invoicing and e-Payment Activities in European Enterprises, Special Report, September.\

3. S. Wheeler R, " Multiple algorithms for fraud detection. Knowledge-Based Systems," no. S0950-7051(00)00050-2, 2000.
4. Credit Card Fraud Detection using Machine Learning Algorithms, Vaishnavi Nath Dornadula, Geetha S, VIT, Chennai-600127
5. Melo-Acosta, German E., et al. "Fraud Detection in Big Data Using Supervised and Semi-Supervised Learning Techniques." 2017 IEEE Colombian Conference on Communications and Computing (COLCOM), 2017, doi:10.1109/colcomcon.2017.8088206.
6. W. Yu and N. Wang, "Research on Credit Card Fraud Detection Model Based on Distance Sum," 2009 International Joint Conference on Artificial Intelligence, Hainan Island, 2009, pp. 353-356.
7. Awoyemi, John O., et al. "Credit Card Fraud Detection Using Machine Learning Techniques: A Comparative Analysis." 2017 International Conference on Computing Networking and Informatics (ICCNI), 2017, doi:10.1109/iccni.2017.8123782.