Capstone Project

Final Project Report

Project Title: Distributed patterns to detect anomalies in time-series data

Project S. No.: 3

NU Faculty Mentor: Prof. Deeksha Arya

Team Members: Ashwin Pilgaonkar, Ankit Kumar Singh, Surajnath Sidh, Saumya Gupta, Pranavi Padegal, Jeel Dharmesh Shah

S. No.	Name of Student*	Contribution in Project
1.	Ashwin Pilgaonkar	Research, Building Neural Network with Tensorflow & Evaluation of Network
2.	Ankit Kumar Singh	Data exploration and Feature Engineering
3.	Surajnath Sidh	Research, Building Neural Network with Tensorflow & Evaluation of Network
4.	Saumya Gupta	Data exploration and Feature Engineering
5.	Pranavi Padegal	Data exploration and Feature Engineering
6.	Jeel Dharmesh Shah	Data exploration and Feature Engineering

* In alphabetical order

Table of Contents

Abstract

Goal

To build an Anomaly detector to detect anomalies in time-series data. We have chosen credit card fraud detection for our project.

About the Problem

Credit card fraud detection relies on the analysis of recorded transactions. Transaction data is mainly composed of a number of attributes (e.g. credit card identifier, transaction date, recipient, amount of the transaction). Automatic systems are essential

since it is not always possible or easy for a human analyst to detect fraudulent patterns in transaction datasets, often characterized by a large number of samples, many dimensions and online updates. Also, the cardholder is not reliable in reporting the theft, loss or fraudulent use of a card.

In the following we will now discuss advantages and disadvantages of Expert Driven and Data Driven approaches to fraud detection.

Introduction

What is an Anomaly?

"Anomalies are patterns in data that do not conform to a well defined notion of normal behavior"(Chandola et al, 2009). Or it can be thought of being generated from a process that is different from the process that generated most of the observations.

As we can see in the graph below, the points O1 and O2 are away from the regions N1 and N2 where most of the points are lying. The same is the case with point in region O3 even though the points constituted form a region, they are still isolated from the regions where normal data lies. There is a possibility that these points are anomalies in the 2D space described here.

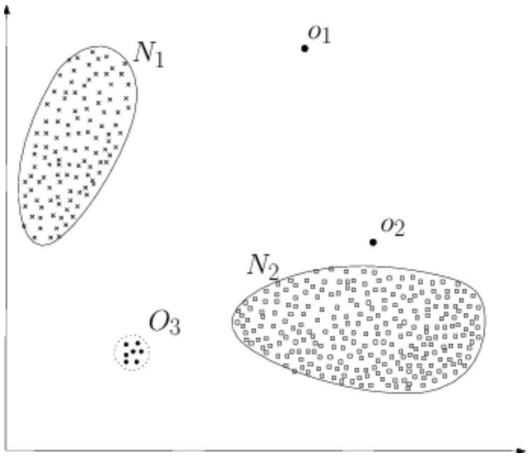

Fig.: Example Anomaly in 2D plane

There may be a variation in the types of outliers because anomalies may also be of many forms. For example in the case of data regarding credit card,

outliers may be caused by either frauds or government corruption or it could also be due to a malfunction in machines and sensors.

Types of anomalies

Point Anomaly

The examples for this are the points O1 and O2 as well as the points in region O3. A point anomaly is detected when an individual instance of observation can be taken to be anomalous as compared to the rest of the data.

Contextual Anomaly

A contextual anomaly must have features with respect to some context attribute (for example: time, space, etc). Whether it is an anomaly or not is determined with respect to that specific context. For example, if we have the data about the amount of transactions as well as the information about the time at which the transaction took place; a transaction of big amount would be considered to be anomalous in the month of june but the same amounted transaction would be listed under normal data if it was made in duration of Christmas.

Collective Anomaly

In this case, some data instances that are related may be anomalous as compared to the entire dataset, but individually, they may not be anomalous. In other words, a low or a high data instance value may not be an anomaly but its persistence is. For example, the sales of a supermarket may fluctuate with time but if say, the sales are low for a greater period of time, then that data must be anomalous.

In Contextual and Collective anomalies, the instances are not considered to be i.i.d.

How do we detect them?

Contextual anomaly detection will not be a convenient way for our data, since the day is a collection of transactions from only 2 days and thus it will be difficult to add the temporal aspect. Thus, the anomalies to be detected will be considered to be point anomalies. However, time will be kept as a feature so some light will be given to contextual information even though it will not be directly modeled.

One way to automatize the detection is by means of Data Driven methods, i.e. setting up an FDS based on Machine Learning able to learn from data in a supervised or unsupervised manner which patterns are the most probably related to a fraudulent behavior.

With Machine Learning we let the computers to discover fraudulent patterns in the data. Data Driven approaches have advantages and disadvantages, for instance with Machine Learning algorithms we can: i) learn complex fraudulent configurations (use all features available), ii) ingest large volumes of data, iii) model complex distributions, iv) predict new types of fraud (anomalies from genuine patterns) and v) adapt to changing distribution in the case of fraud evolution. However, they have also some drawbacks: i) they need enough samples, ii) some models are black box, i.e. they are not easily interpretable by investigators and they do not provide an understanding of the reason why an alert is generated.

Objectives

Detect fraud in credit card transaction data. Using Machine Learning to learn from data in a supervised manner which patterns are the most probably related to a fraudulent behavior.

Work Done

Challenges in Anomaly Detection

The most simple and straightforward method for detection of anomalies would to define a specific region for all the normal data and any instances found outside this

region may be classified as an anomaly. However, there are many challenges that we face, as listed below:

- Defining a clear boundary between normal and abnormal data and modeling a region that covers all normal instances is extremely difficult.
- If the anomalies are a result of malicious actions. The intend may also be to make the anomalies look normal.
- With advancement in time, the whole notion of what is normal data may change.
- The definition of anomaly may be different for different applications and there cannot just be a single algorithm as a solution for handling all of them equally.
- Noisy instances may appear in the data and these are difficult to be differentiated from anomalies.

Dataset selected

The dataset has been collected during a research collaboration of Worldline and the Machine Learning Group(http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection. It is a September 2013 dataset that shows the transactions made by European credit card holders. Out of the two days that the data was collected, 0.172% was found to be fraudulent(492 out of 284,807). This shows that the data is highly unbalance

Exploring the Data

Schema of dataset in csv format is : *Time, V1, V2, V3, …., V28, Amount, Class*
Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Attribute 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The Attribute 'Amount' is the transaction Amount. Attribute 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Exploratory analysis of dataset

Using pandas isnull().sum() the number of null values in each attribute was found. Fortunately the data has no missing values.

Comparing time between fraudulent and normal transactions.(matplotlib was used to plot the histogram)

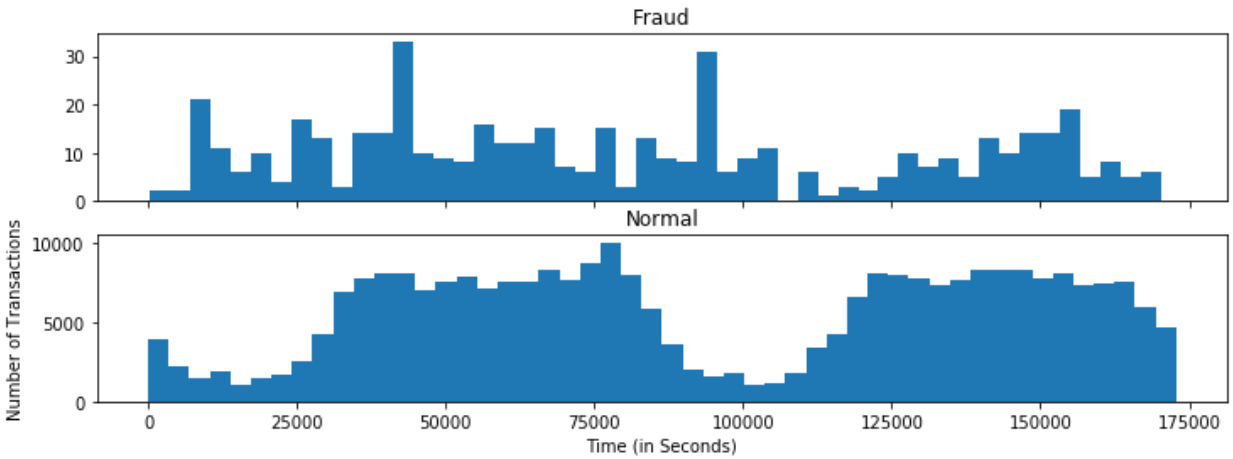

Fig: Number of Transactions vs Time for Normal and Fraud Transactions

The 'Time' feature looks pretty similar across both types of transactions. You could argue that fraudulent transactions are more uniformly distributed, while normal transactions have a cyclical distribution. This could make it easier to detect a fraudulent transaction during at an 'off-peak' time.

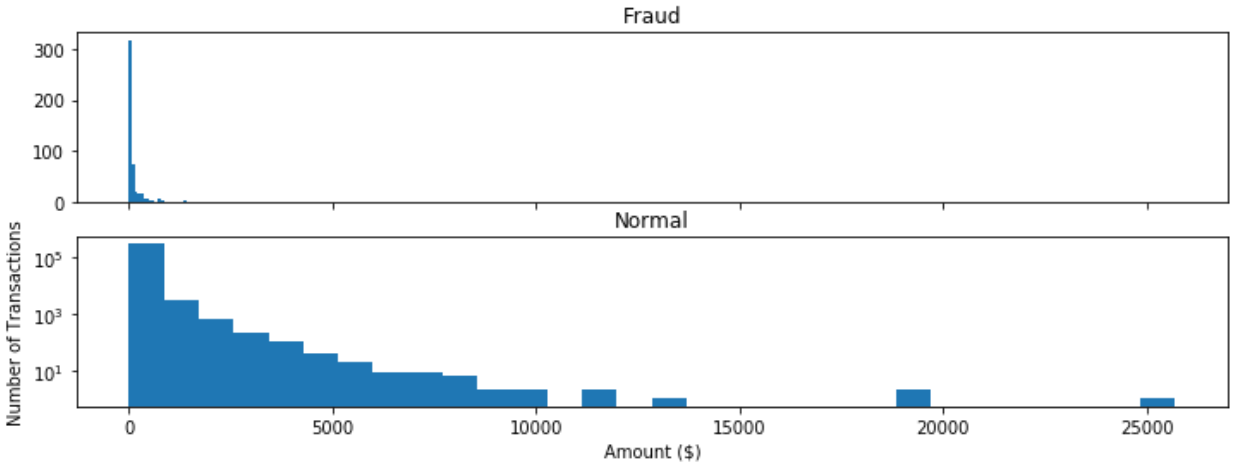

Fig: Number of Transactions vs Amount(in $) for Normal and Fraud Transactions

Most transactions are small amounts, less than $100. Fraudulent transactions have a maximum value far less than normal transactions, $2,125.87 vs $25,691.16.

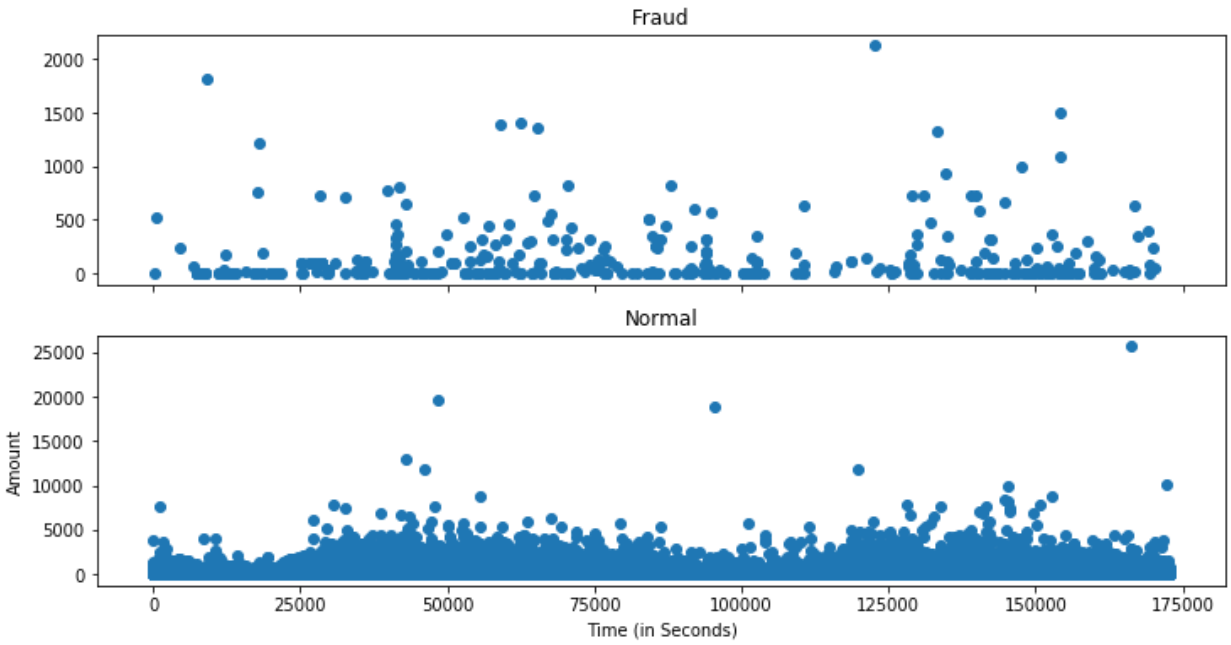

Fig: Amount(*In $*) vs Time for Normal and Fraud Transactions

We found nothing too useful here.

Feature Engineering

All anonymized features were taken and a histogram for each of the features was plotted as shown in the figure below.

Drop less important features

We will drop all the features that have similar distributions between fraud and non-fraud transactions because these will not help us in distinguishing between the two. We compared the distributions using np.percentile() If the distributions were too similar, We dropped the feature

Based on the plots, new features are created to identify values where fraudulent transaction are more clear to see.

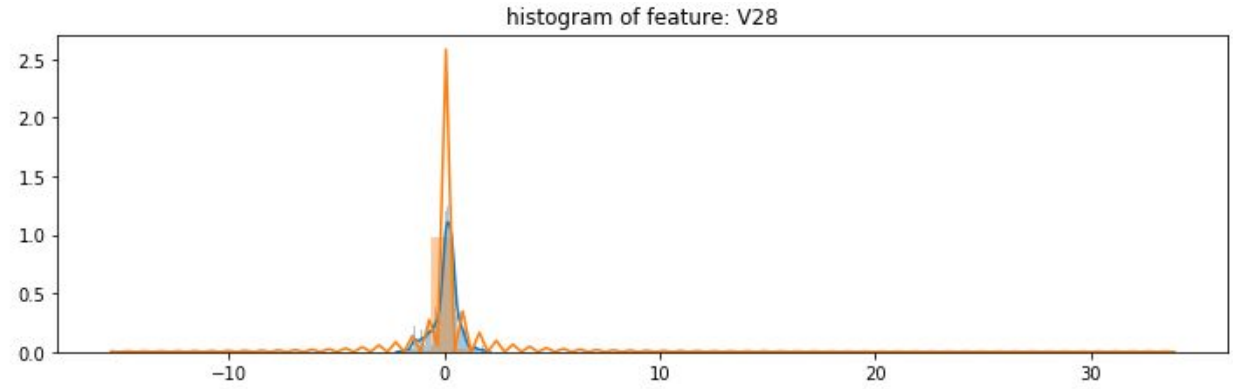

Fig: Less important feature

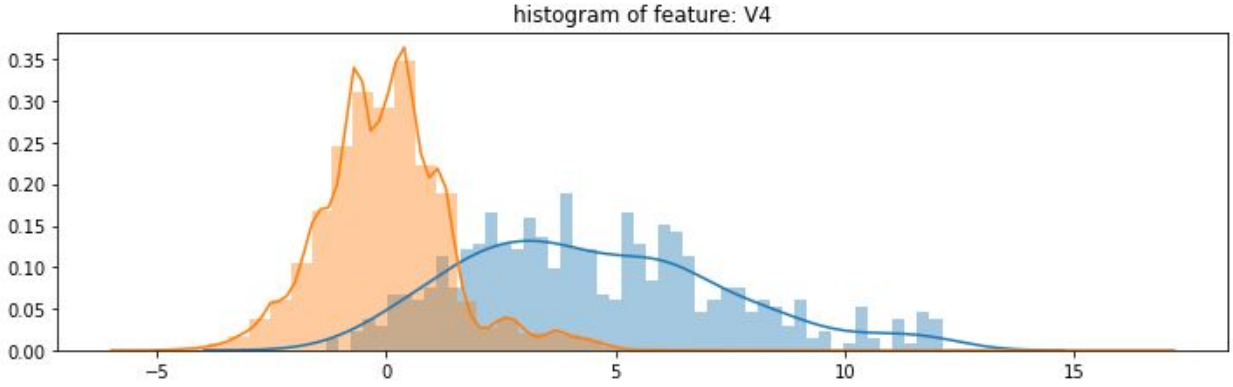

Fig: More important feature

Account for unbalanced classes in data

By dividing the number of transactions by those that are fraudulent, ratio will equal the value that when multiplied by the number of fraudulent transactions will equal the number of normal transactions.

Due to the imbalance in the data, ratio will act as an equal weighting system for the model.

Experiments and Results

Prepare training and testing data

Training Set have 80% of the fraudulent and normal transactions in training data

Testing Set (Validation Set) have 20% of the fraudulent and normal transactions to test data

Shuffle the data frames to ensure that the training is done in a random order and neural network does not learn about inherent order of data.

Neural Network

We are using multilayer structure, with 1 Input and 1 Output layer and 4 Hidden layer with Cross Entropy as Cost function and Adam Optimizer (an optimizer from Gradient Descent family) as our Optimizer

Neural Net training

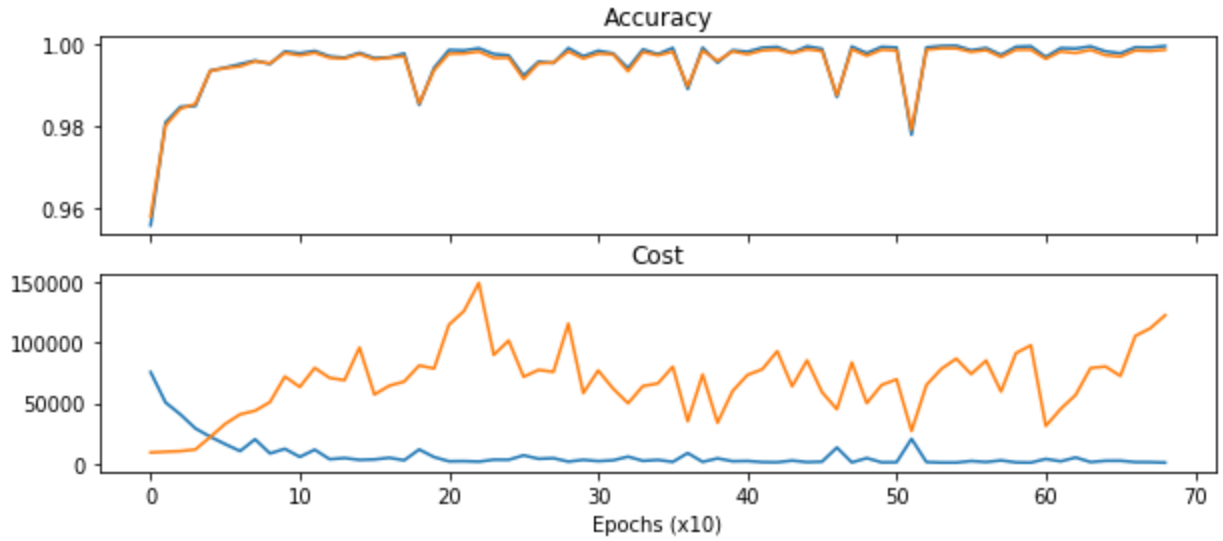

Fig: Neural Net training over time

We took care of unbalanced data and used F1 score along with Accuracy measure to calculate final accuracy.

Results

- F1-Score = 0.999542623931
- Testing Accuracy = 0.999087

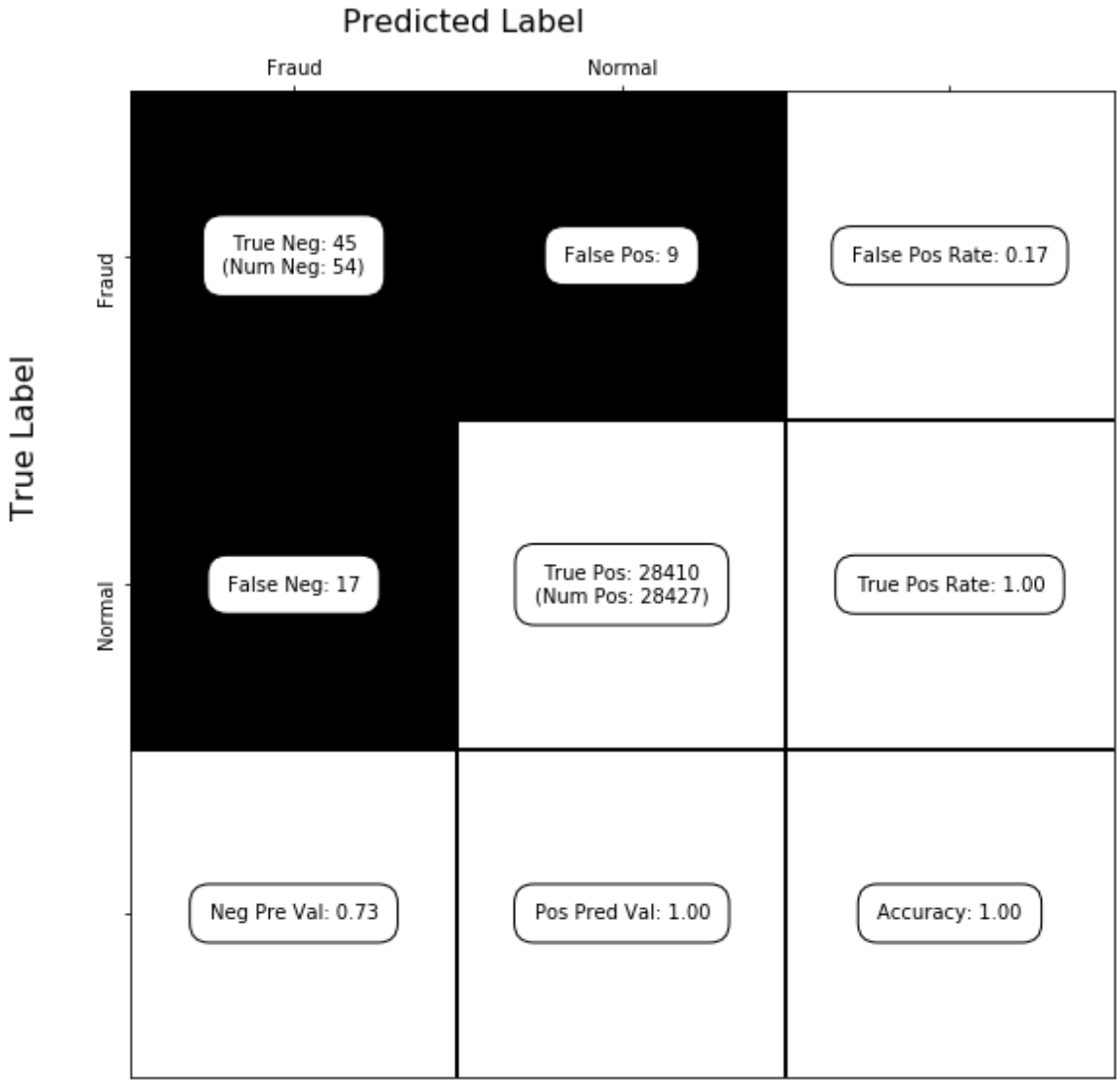

Fig: F1 - Score

Conclusions and Possible Future Work

We have achieved F1-Score of 0.99 along with Accuracy of 99.9% on a biased dataset. our false positive rate is 0.17 and Our false negative rate us 0.73.

We can work on bringing our false negative rate down but we kept our model simple due to reasons for example Risk of Overfitting

We can use ensemble(multiple models and use vote from all the models for our final decision) to improve our F1 score and Accuracy and minimize false positive rate and our false negative rate

We can use Meta ML(finding hyperparameters for our models used in our ensemble using Machine Learning)

References

Chandola, Varun, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey." ACM computing surveys (CSUR) 41.3 (2009)

Dal Pozzolo, Andrea, and Gianluca Bontempi. "Adaptive machine learning for credit card fraud detection." (2015) (Ph.D Thesis)

Diederik P. Kingma, Jimmy Ba "Adam: A Method for Stochastic Optimization" arXiv:1412.6980 [cs.LG] (2017)

Matplotlib: Python plotting — Matplotlib 2.1.0 documentation. (n.d.). Retrieved from https://matplotlib.org/

NumPy — NumPy. (n.d.). Retrieved from http://www.numpy.org/

Python Data Analysis Library — pandas: Python Data Analysis Library. (n.d.). Retrieved from http://pandas.pydata.org/

TensorFlow. (n.d.). Retrieved from https://www.tensorflow.org/