1. Credit Card Fraud Detection Intuitions

What is Credit Card Fraud?

Credit card fraud is when someone uses another person's credit card or account information to make unauthorized purchases or access funds through cash advances. Credit card fraud doesn’t just happen online; it happens in brick-and-mortar stores, too. As a business owner, you can avoid serious headaches – and unwanted publicity – by recognizing potentially fraudulent use of credit cards in your payment environment.

Three challenges surrounding credit card fraud

1. It's not always easy to agree on ground truth for what "fraud" means.
2. Regardless of how you define ground truth, the vast majority of charges are not fraudulent.
3. Most merchants aren't experts at evaluating the business impact of fraud.

Problem Statement:

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be a fraud. This model is then used to identify whether a new transaction is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications.

Observations

* Very few transactions are actually fraudulent (less than 1%). The data set is highly skewed, consisting of 492 frauds in a total of 284,807 observations. This resulted in only 0.172% fraud cases. This skewed set is justified by the low number of fraudulent transactions.
* The dataset consists of numerical values from the 28 ‘Principal Component Analysis (PCA)’ transformed features, namely V1 to V28. Furthermore, there is no metadata about the original features provided, so pre-analysis or feature study could not be done.
* The ‘Time’ and ‘Amount’ features are not transformed data.
* There is no missing value in the dataset.

Why does class imbalanced affect model performance?

* In general, we want to maximize the recall while capping FPR (False Positive Rate), but you can classify a lot of charges wrong and still maintain a low FPR because you have a large number of true negatives.
* This is conducive to picking a relatively low threshold, which results in the high recall but extremely low precision.

What is the catch?

* Training a model on a balanced dataset optimizes performance on validation data.
* However, the goal is to optimize performance on the imbalanced production dataset. You ultimately need to find a balance that works best in production.
* One solution to this problem is: Use all fraudulent transactions, but subsample non-fraudulent transactions as needed to hit our target rate.

Business questions to brainstorm:

Since all features are anonymous, we will focus our analysis on non-anonymized features: Time, Amount

1. How different is the amount of money used in different transaction classes?
2. Do fraudulent transactions occur more often during a certain frames?

# Credit Card Fraud Detection using CNN

## Classification using CNN

It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase. In this project we are going to build a model using CNN which predicts if the transaction is genuine or fraudelent.

### Dataset

We are going to use the Credit Card Fraud Detection Dataset from kaggle. It contains anonymized credit card transactions labeled as fraudulent or genuine. You can download it from [here](https://www.kaggle.com/mlg-ulb/creditcardfraud).

The datasets contains 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.

It contains only numerical input variables which are the result of a PCA transformation. The only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-senstive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

### Build CNN

A Sequential() model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

Conv1D() is a 1D Convolution Layer, this layer is very effective for deriving features from a fixed-length segment of the overall dataset, where it is not so important where the feature is located in the segment. In the first Conv1D() layer we are learning a total of 32 filters with size of convolutional window as 2. The input\_shape specifies the shape of the input. It is a necessary parameter for the first layer in any neural network. We will be using ReLu activation function. The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.



BatchNormalization() allows each layer of a network to learn by itself a little bit more independently of other layers. To increase the stability of a neural network, batch normalization normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. It applies a transformation that maintains the mean output close to 0 and the output standard deviation close to 1.

Dropout() is used to randomly set the outgoing edges of hidden units to 0 at each update of the training phase. The value passed in dropout specifies the probability at which outputs of the layer are dropped out.

Flatten() is used to convert the data into a 1-dimensional array for inputting it to the next layer.

Dense() is the regular deeply connected neural network layer. The output layer is also a dense layer with 1 neuron because we are predicting a single value as this is a binary classification problem. Sigmoid function is used because it exists between (0 to 1) and this facilitates us to predict a binary input.