Intermediate Status Report

Group 3

Table of contents

We are working on the credit card fraud detection project. This document summarizes our progress in the project so far.

1) Difficulties encountered -

We are working with a real life IEEE data set for credit card fraud detection. The difficulties faced so far are listed below:

- Understanding of the features as most features have masked meanings - Listing some of the complex features: card feature is distributed over 6 different columns, in which card1,2,5 are numerical and rest are categorical
- The data is broken into two different files namely “Transaction” and “Identity” , where for many transactions the identity data is missing. 314 columns out of 394 columns have missing data.
- The dataset is highly skewed with very large number of transactions in genuine class. We only have 3.5 % of the Fraudulent transactions in the data as shown in the fig:1.To solve this we have done down sampling on the dataset to get balanced classes.
- TransactionDT: timedelta is only a time from a particular reference but not the actual time, we had to convert this delta(difference) into actual days and hours to analyse the distribution of fraud transaction over different dates as shown in fig:2

- Most of the D (time delta) features in the dataset are NAN.

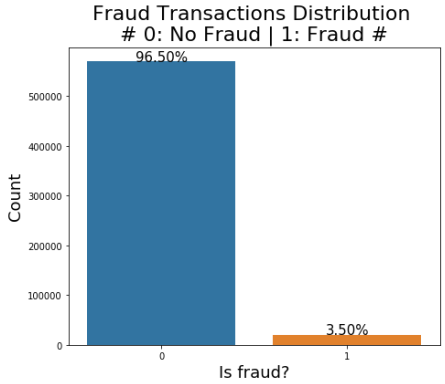


Fig: 1

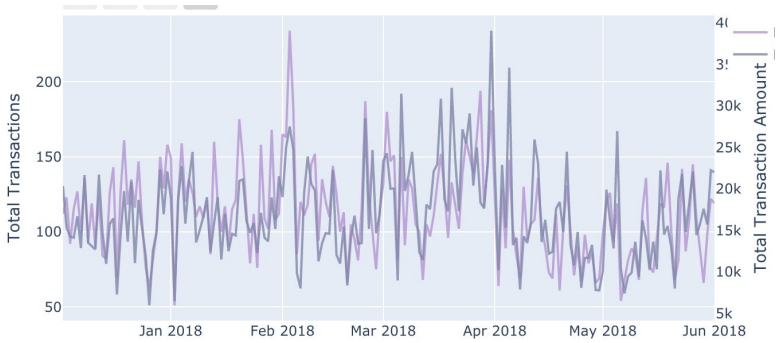


Fig: 2

	TransactionID	id_01	id_02	id_03	id_04	id_05	id_06	id_07	id_08	id_09	...	id_31	id_32	id_33	id_34
0	2987004	0.0	70787.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	samsung browser 6.2	32.0	2220x1080	match
1	2987008	-5.0	98945.0	NaN	NaN	0.0	-5.0	NaN	NaN	NaN	...	mobile safari 11.0	32.0	1334x750	match
2	2987010	-5.0	191631.0	0.0	0.0	0.0	0.0	NaN	NaN	0.0	...	chrome 62.0	NaN	NaN	NaN
3	2987011	-5.0	221832.0	NaN	NaN	0.0	-6.0	NaN	NaN	NaN	...	chrome 62.0	NaN	NaN	NaN
4	2987016	0.0	7460.0	0.0	0.0	1.0	0.0	NaN	NaN	0.0	...	chrome 62.0	24.0	1280x800	match

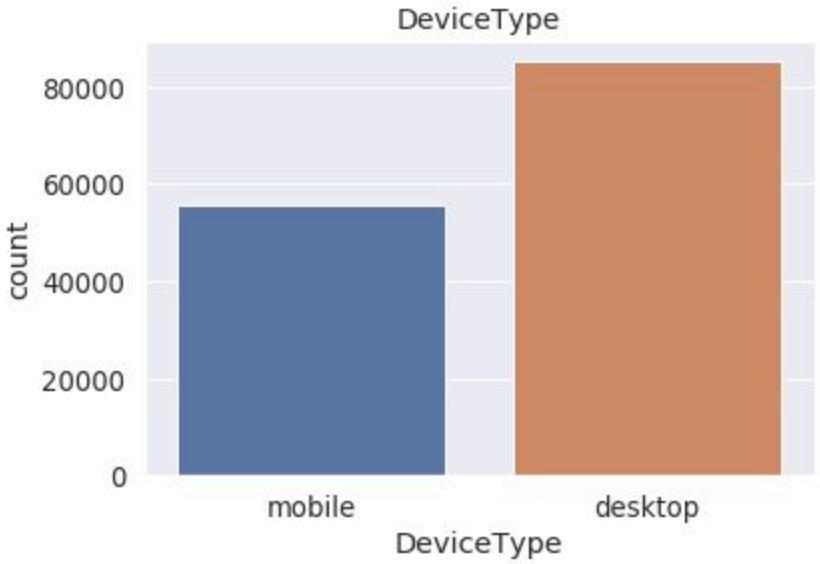
Fig: 3 showing many columns having missing data in identity data

2) Progress

Data Analysis: The data set we are dealing with is huge and is divided into four parts. The four parts are listed as:

a. Test_identity
b. Test_transaction
c. Train_identity
d. Train_transaction

The tables identity and transaction are joined by key TransactionId. The target variable isFraud is a binary value and is present in the transaction file. Many of the features are masked for keeping the data anonymized. Here is a summary of the first few columns our transactions data before preprocessing:



	count	mean	std	min	25%	50%	75%	max
TransactionID	590540.0	3.282270e+06	1.704744e+05	2987000.000	3134634.750	3282269.500	3429904.25	3.577539e+06
isFraud	590540.0	3.499001e-02	1.837546e-01	0.000	0.000	0.000	0.00	1.000000e+00
TransactionDT	590540.0	7.372311e+06	4.617224e+06	86400.000	3027057.750	7306527.500	11246620.00	1.581113e+07
TransactionAmt	590540.0	1.350272e+02	2.391625e+02	0.251	43.321	68.769	125.00	3.193739e+04
card1	590540.0	9.898735e+03	4.901170e+03	1000.000	6019.000	9678.000	14184.00	1.839600e+04
card2	581607.0	3.625555e+02	1.577932e+02	100.000	214.000	361.000	512.00	6.000000e+02
card3	588975.0	1.531949e+02	1.133644e+01	100.000	150.000	150.000	150.00	2.310000e+02
card5	586281.0	1.992789e+02	4.124445e+01	100.000	166.000	226.000	226.00	2.370000e+02
addr1	524834.0	2.907338e+02	1.017411e+02	100.000	204.000	299.000	330.00	5.400000e+02
addr2	524834.0	8.680063e+01	2.690623e+00	10.000	87.000	87.000	87.00	1.020000e+02
dist1	238269.0	1.185022e+02	3.718720e+02	0.000	3.000	8.000	24.00	1.028600e+04
dist2	37627.0	2.318554e+02	5.290535e+02	0.000	7.000	37.000	206.00	1.162300e+04

➔ **Memory reduction:** As the data set is huge we have to use some memory reduction functions. Using the memory reduction function we could reduce the memory usage from **1959.88 Mb to 650.48 Mb** (66.8% reduction).

➔ **Data cleaning:** To do the data cleaning part we have removed the duplicate rows from the dataset. Also the data has many rows with NaN which needs to be replaced with 0 in this case. Since most of the features are masked to numerical, we have no insights what the original features were. Hence we would just replace them with 0 for now. In the initial dataset, there are 414 cells with value NaN. We used dataframe fillna function to replace those null values with 0.

➔ **Data Transformation:** The next step in this process would be to transform all the categorical features to numerical so that it fits the data models that we will create later on. There are few techniques to do this but label encoding and one hot encoding is one of the most popular ones. In order to transform a category to number, we used sklearn label encoder which simply iterate through all the categories and map them to a number. The only problem is if we feed this new data to some of the algorithms, there is a possibility that the algorithm favor one over the other. For example if we map dog:1 and cat:2, some ML algorithms favor dogs over cats or vice versa due to the order of them. But in reality the order of those categories have no effect on the outcome.

With our dataset, we first filtered all the features that are categorical and separated them from the rest of the data. In our dataset, 31 features were categorical or had categorical values in one of their instances. Using this newly created dataframe, we used sklearn label encoder to transform the data to numerical. We will need to run one hot encoding on our categorical data as well to prevent from the problem discussed earlier. Once data transformed, then we combined the features back to our original dataset.

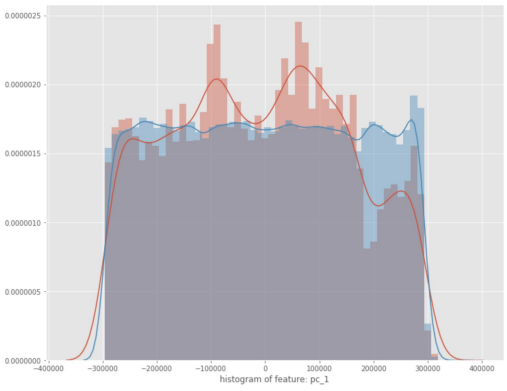
➔ **Standardize data:** Most of the data is already standardized except the amount and date. In order to have the right scale for our ML algorithms, we need to reshape the data to get them ready for next steps. To do this, we used StandardScaler in our project to reshape both transaction amount and transaction date.

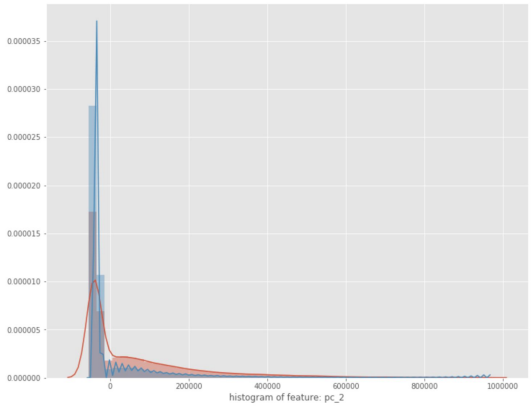
➔ **Dimensionality Reduction:** As we discussed in the beginning of this section, our data consists of 403 features. This much complexity and dimension can slow down any models at the time of training and also introduces the problem of overfitting the model. This can result in poor accuracy and very slow training process. This is when we need to use dimensionality reduction to overcome the curse of dimensionality. PCA is the most widely used tool that can be helpful in this step to prioritize the features that have the most weight on the outcome.

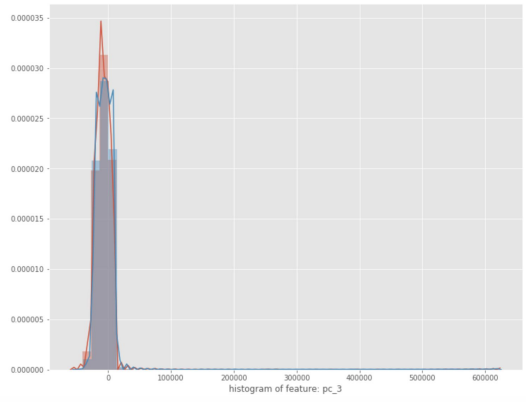
Using sklearn PCA, we limited our dataset to filter for top 10 principal component so that we can have a robust model when it comes to training. The future work in this step would be to find the optimal number of principal components that need to be used to have a better accuracy at the end. Here is a summary of our data after preprocessing:

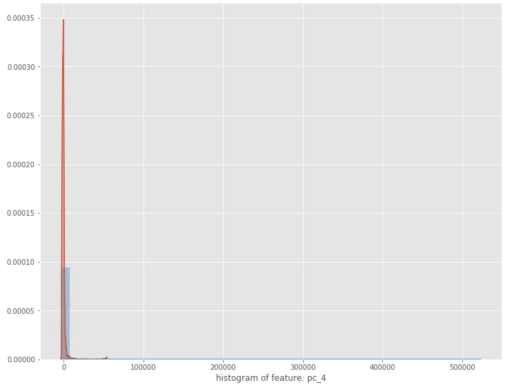
	count	mean	std	min	25%	50%	75%	max
pc_1	590540.0	-1.036872e-09	170842.694501	-296685.999527	-147832.402741	-503.311439	146547.294091	345074.295269
pc_2	590540.0	-2.170408e-11	107705.632207	-55351.329552	-45307.515924	-37193.835258	-29050.389806	964741.833269
pc_3	590540.0	8.692406e-12	54597.574715	-54675.035773	-14474.842929	-5855.012910	3012.393833	622600.638529
pc_4	590540.0	-8.547353e-12	14568.393310	-3145.025784	-1530.081789	-731.104391	75.881885	523046.806580
pc_5	590540.0	7.464482e-13	4894.204188	-8778.853287	-4255.112012	208.518284	3869.111791	9521.329098
pc_6	590540.0	4.570193e-13	2482.843559	-104446.240778	-119.012118	-28.829423	88.994463	170040.753696
pc_7	590540.0	-4.304819e-13	1979.885211	-127453.021948	-136.504051	-103.062557	-28.583112	172669.975757
pc_8	590540.0	4.548657e-15	1717.354581	-24529.778405	-130.254167	-101.153535	-60.390526	303968.132186
pc_9	590540.0	1.158712e-12	1566.977520	-31295.129471	-94.166169	-27.412229	45.448625	41021.290602
pc_10	590540.0	-1.340900e-13	1434.337909	-64858.751913	-137.656203	-80.837037	-26.336444	108004.036441
isFraud	590540.0	3.499001e-02	0.183755	0.000000	0.000000	0.000000	0.000000	1.000000

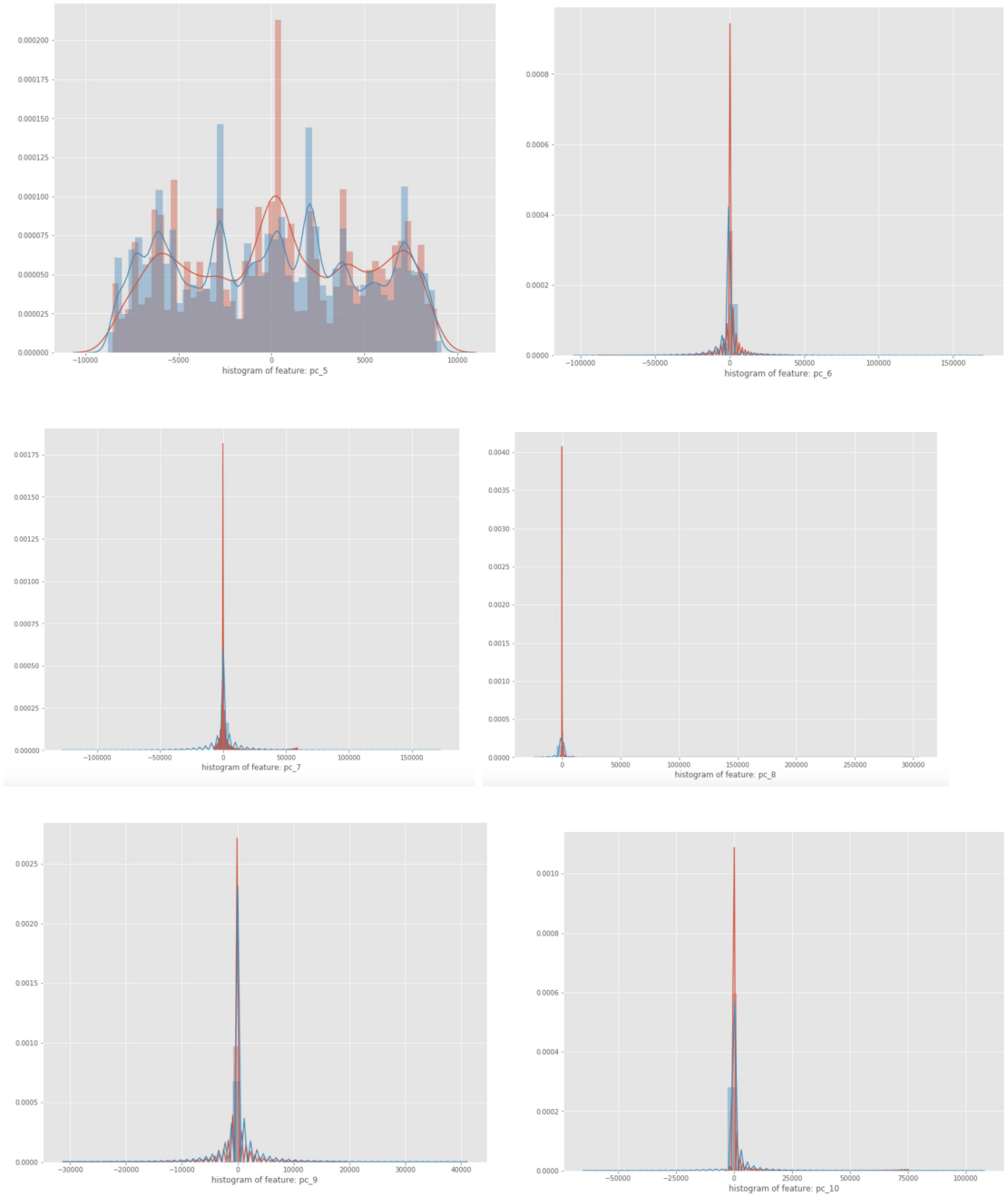
Here is the histogram of each principal component:



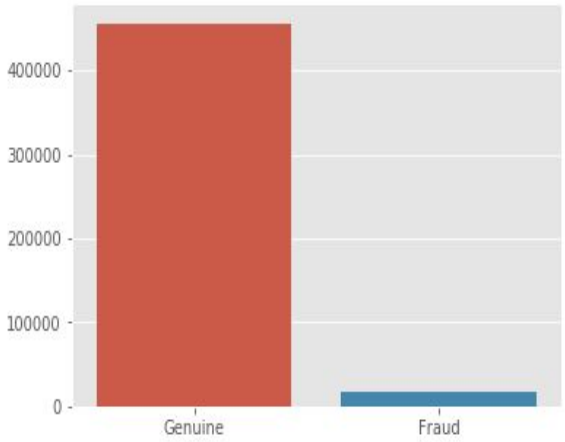


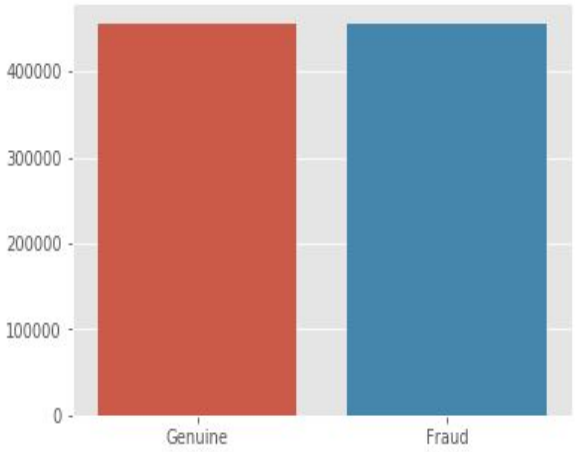






➔ **Dealing with Class Imbalance:** The dataset we are using is highly imbalanced with only 5% of transactions are being classified as fraudulent. So we are using Oversampling Technique to match number of fraudulent transactions with the genuine transactions to improve accuracy in prediction





Before Resampling with Class imbalance

After Resampling with balanced data

3) Results - We have applied 3 models till now on our dataset.

- Naive Bayes : This classification model helped us in achieving an accuracy of 56.6 %. We are aiming on higher accuracy so this model may be too simple for this complex dataset.
- Logistic Regression: Then we applied the logistic regression model to our dataset and the accuracy improved significantly to 75%. This accuracy score is also not enough for our problem statement.
- Random Forest: Using the random forest method we further increased the accuracy of the prediction to 81.7%.

4) Remaining Tasks

- We will be working on how different number of principal component features can affect the model prediction values.
- Since our data is very complex we will try more exploratory data analysis techniques and extract more important features.
- We have still achieved the best accuracy as 81.7% , we be trying to improve this accuracy to around 95%. Also we will try to measure and compare different evaluation metrics.
- We will also work on applying some unsupervised learning technique to our problem and compare the results with supervised learning.

References:

1. https://www.kaggle.com/c/ieee-fraud-detection/data