Credit Card Fraud Detection Using Machine Learning

Swaroop K
Dept. of ISE
SDMCET
Dharwad, India

Amruta D
Dept. of ISE
SDMCET
Dharwad, India

Sanath J
Dept. of ISE
SDMCET
Dharwad, India

Pooja G
Dept. of ISE
SDMCET
Dharwad, India

***Abstract*— In todays world, the most easiest mode of payment is credit card for both online and offline. It helps in providing cashless shopping across the globe. Fraud event occurs only during online payment as credit card number is sufficient to make transaction which will be on the credit card to make online payment but for offline payment password will be asked so during offline transaction frauds cannot occur. In the existing system of detecting fraud transaction, the fraud is detected after the transaction is done. Companies have a detailed analysis of transactional and fraud data. Frauds tends to appear in patterns. In billions of credit card transactions, it is quite difficult to analyse each in isolation. Having predictive algorithms can help to detect fraudulent transactions. this is how data mining comes into play. Data consists of combination of continuous data and nominal data. We can use variety statistical tests to prevent fraud events. Detecting credit card fraud is still not a perfect science. While fraud is still a major financial issue to banks, the distribution of fraud to non-fraudulent transactions is severely skewed towards non-fraudulent transactions. Out of an estimated 12 billion transaction made annually 10 million are fraudulent (this shows every transaction in 1200 is fraudulent transaction).To analyse and predict fraud events we have used local outlier factor and isolation forest algorithms and thus calculated number of fraud transactions. We have calculated the accuracy and number of errors of both the algorithms.**

Keywords: Credit card ,Isolated forest ,Local outlier factor, Fraud detection, Data mining.

I. INTRODUCTION

In daily routine we use credit cards to buy goods and services using online transaction or physical card for offline transaction .In credit card based purchase, the card holder issues his card to merchant to do payment .the person has to steal the card to make the transaction fraudulent . If the user is not aware of loss of card it leads to financial loss to the user as well as credit card company. When the payment mode is online, attackers require only little information for doing false transaction. Example card number. The only way to detect these kind of fraud is to analyse the spending patterns on every card and irregularities are figured with respect to normal pattern. Fraud which is detected using existing purchase data of card holder is way to reduce the rate of frauds. Every card holder is characterised by patterns containing information about distinctive purchase category the time since the last buying, money spent and other things. Falsehood from such patterns is sensed as fraud. Fraud in finance is an ever growing issue, resulting in far reaching consequences. Fraud can be defined as criminal cheating with an aim of financial gain. With an emergence of internet, it has lead to increase in credit card transactions .As credit card is most prevailing method, as it attracts more discounts and offers in both stores and e-commerce, it is more vulnerable to fraud events. Credit card fraud detection is the science and the art of detecting unusual activity in credit transactions . Fraud occurs when the credit card information of the individual is stolen and used to make unauthorized purchases and or withdrawals from the original holders account .A major challenge to credit fraud detection research is the availability of the real world data due to privacy and legal concerns. Online Shopping is one of the largest and fast growing trend and mode of payment will be by using credit card, debit card and net banking. Online payment does not require physical card. If credit card details is known to others that will become a major risk. Currently, card holder will come to know only after the fraud transaction is carried out. No mechanism exist to track fraud transaction. In this project, that is exactly what we are going to be doing as well. Using a dataset of nearly 28,500 credit card transactions and multiple unsupervised anomaly detection algorithms, we are going to identify transactions with a high probability of being credit card fraud. Furthermore, using metrics such as precision, recall, and F1-scores, we will investigate why the classification accuracy for these algorithms can be misleading. In addition, we will explore the use of data visualization techniques common in data science, such as parameter histograms and correlation matrices, to gain a better understanding of the underlying distribution of data in our data set.

II. LITERATURE SURVEY

In [2] the authors begin by explaining the method used for transactions through credit cards. They have proposed a system in which they integrate their algorithm with the payment gateway to detect fraudulence in real time. The authors used 7 techniques to develop the algorithm, which are Neural Networks, Rule Induction, Case-based reasoning, Genetic Algorithms, Inductive Logic Programming, Expert Systems, Regression. The authors determined, the ANN method would best serve this problem statement. The output of the neural network will be in the form of probability which tells the degree of a transaction being fraudulent. Neural network are trained on information based on the various categories about the card holder such as profession of the card holder, earnings, about the large amount of purchased are placed. The system will use back propagation learning algorithm in this phase to train the network. Depending on the numeric value of probability between 0 and 1, a transaction will be classified into one of the following categories: Non-Fraudulent , Doubtful , Suspicious and Fraudulent. This system being developed will particularly focus on the merchant side of the industry which will be beneficial to the merchant by reducing the merchant’s losses which he has to bear if a transaction is fraudulent. Therefore it is limited by the availability of Merchant side transaction data which is hard to obtain on scale.

Authors focused on the Chinese market as it is rapidly growing and fast paced[3]. The authors proposed a data mining technique using outlier detection using distance sum to identify fraud transactions. The authors preferred to use this method over traditional statistical methods like Regression and Discriminant analysis because outlier detection method is independent of the dataset distribution. The paper used Euclidean distance formula to calculate distance sum to detect outliers. The authors calculated a threshold value for distance, if the distance is above said threshold, the object is classified as an anomaly, or in this case, a fraud transaction. The authors collected data from a domestic bank in China, with 16000 observations. The authors achieved a highest accuracy of 89.4% for threshold value of 12. This method is highly dependent on the nature of distribution of the data, and may vary for data sources of different banks.

III. SYSTEM DESIGN

The fraud detection module will work in the following steps:

1)The Incoming set of transactions and amount are treated as credit card transactions.

2)The credit card transactions are given to machine learning

algorithms as an input.

3)The output will result in either fraud or valid transaction by

analyzing the data and observing a pattern and using machine

learning algorithms such as local outlier factor and isolation forest to do anomaly detection.

4)The fraud transactions are given to alarm which alerts the

user that fraud transaction has occurred and the user can block the card to prevent further financial loss to him as well as the credit card company.

5)The valid transactions are treated as genuine transactions.

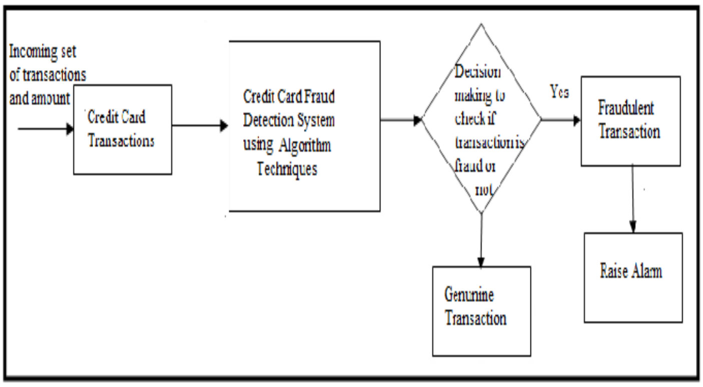
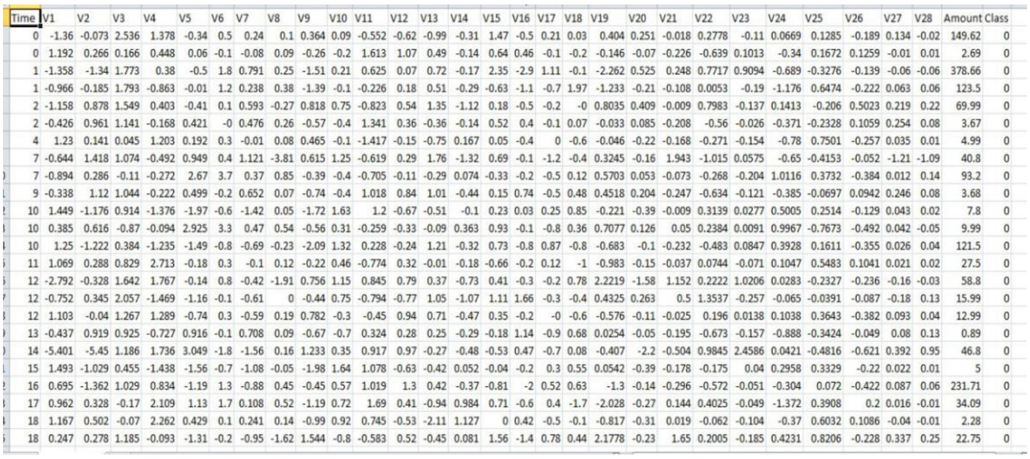

Incoming set of transactions and amount
Credit Card Transactions
Credit Card Fraud Detection System using Algorithm Techniques
Decision making to check if transaction is fraud or not
Yes
Fraudulent Transaction
Raise Alarm
Genuine Transaction

Figure 1 . System block diagram of credit card fraud detection

IV. SOFTWARE IMPLEMENTATION

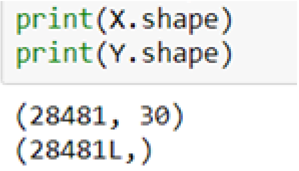
We collected the dataset from Kaggle [1].we collected the source code from GitHub[4]. The datasets contains transactions made by credit cards in september2013 by European cardholders shown in figure 2.

We imported libraries and printed the versions in our code and then we imported necessary packages. we loaded the dataset from the csv file using pandas. we explored the dataset. we have 31 different columns as shown in figure 3.v1 to v28 are the result of PCA dimensionality reduction to protect sensitive information in our dataset like we don’t want to expose identity and location of an individual. class 0 indicates valid transaction and class 1 indicates fraud transaction. we have 284807 transactions with 31 columns. further while exploring dataset we noticed that mean values are close to 0 shown in figure 4 it means there are more valid transactions than fraud transactions in our dataset.in order to save time and computational requirements as it is a large dataset we will take only 10% of the data.so now we have 28401 transactions left. now visually we plot histogram of each parameter to check if there are any unusual parameters as shown in Figure 5.Now we calculated number of fraud and valid cases and outlier fraction by dividing the number of fraud transactions with number of valid transactions as shown in Figure 7. We constructed correlation matrix with heat map to know if there is any strong co relationship between different variables in our dataset as shown in Figure 6.It also says if there is any strong linear relationship and also to know which all features are important for overall classification. But we found that most of the values were close to 0 so hence there was no strong relationships between v parameters. We need to format our dataset. We get all columns from data frame, filter columns to remove data that we don't want. We store variable we will be predicting on i.e. X has columns except class label and Y is what we want i.e. it is 1 dimensional array that has class label for samples as shown in Figure 8.This is unsupervised learning as it is normally detected so we do not want labels to be fed into our network.



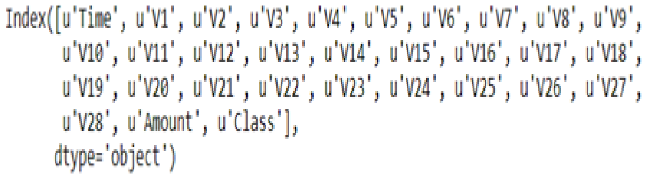
Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
0	-1.36	-0.073	2.536	1.378	-0.34	0.5	0.24	0.1	0.364	0.09	-0.552	-0.62	-0.99	-0.31	1.47	-0.5	0.21	0.03	0.404	0.251	-0.018	0.2778	-0.11	0.0669	0.1285	-0.189	0.134	-0.02	149.62	0
0	1.192	0.266	0.166	0.448	0.06	-0.1	-0.08	0.09	-0.26	-0.2	1.613	1.07	0.49	-0.14	0.64	0.46	-0.1	-0.2	-0.146	-0.07	-0.226	-0.639	0.1013	-0.34	0.1672	0.1259	-0.01	0.01	2.69	0
1	-1.358	-1.34	1.773	0.38	-0.5	1.8	0.791	0.25	-1.51	0.21	0.625	0.07	0.72	-0.17	2.35	-2.9	1.11	-0.1	-2.262	0.525	0.248	0.7717	0.9094	-0.689	-0.3276	-0.139	-0.06	-0.06	378.66	0
1	-0.966	-0.185	1.793	-0.863	-0.01	1.2	0.238	0.38	-1.39	-0.1	-0.226	0.18	0.51	-0.29	-0.63	-1.1	-0.7	1.97	-1.233	-0.21	-0.108	0.0053	-0.19	-1.176	0.6474	-0.222	0.063	0.06	123.5	0
2	-1.158	0.878	1.549	0.403	-0.41	0.1	0.593	-0.27	0.818	0.75	-0.823	0.54	1.35	-1.12	0.18	-0.5	-0.2	-0	0.8035	0.409	-0.009	0.7983	-0.137	0.1413	-0.206	0.5023	0.219	0.22	69.99	0
2	-0.426	0.961	1.141	-0.168	0.421	-0	0.476	0.26	-0.57	-0.4	1.341	0.36	-0.36	-0.14	0.52	0.4	-0.1	0.07	-0.033	0.085	-0.208	-0.56	-0.026	-0.371	-0.2328	0.1059	0.254	0.08	3.67	0
4	1.23	0.141	0.045	1.203	0.192	0.3	-0.01	0.08	0.465	-0.1	-1.417	-0.15	-0.75	0.167	0.05	-0.4	0	-0.6	-0.046	-0.22	-0.168	-0.271	-0.154	-0.78	0.7501	-0.257	0.035	0.01	4.99	0
7	-0.644	1.418	1.074	-0.492	0.949	0.4	1.121	-3.81	0.615	1.25	-0.619	0.29	1.76	-1.32	0.69	-0.1	-1.2	-0.4	0.3245	-0.16	1.943	-1.015	0.0575	-0.65	-0.4153	-0.052	-1.21	-1.09	40.8	0
7	-0.894	0.286	-0.11	-0.272	2.67	3.7	0.37	0.85	-0.39	-0.4	-0.705	-0.11	-0.29	0.074	-0.33	-0.2	-0.5	0.12	0.5703	0.053	-0.073	-0.268	-0.204	1.0116	0.3732	-0.384	0.012	0.14	93.2	0
9	-0.338	1.12	1.044	-0.222	0.499	-0.2	0.652	0.07	-0.74	-0.4	1.018	0.84	1.01	-0.44	0.15	0.74	-0.5	0.48	0.4518	0.204	-0.247	-0.634	-0.121	-0.385	-0.0697	0.0942	0.246	0.08	3.68	0
10	1.449	-1.176	0.914	-1.376	-1.97	-0.6	-1.42	0.05	-1.72	1.63	1.2	-0.67	-0.51	-0.1	0.23	0.03	0.25	0.85	-0.221	-0.39	-0.009	0.3139	0.0277	0.5005	0.2514	-0.129	0.043	0.02	7.8	0
10	0.385	0.616	-0.87	-0.094	2.925	3.3	0.47	0.54	-0.56	0.31	-0.259	-0.33	-0.09	0.363	0.93	-0.1	-0.8	0.36	0.7077	0.126	0.05	0.2384	0.0091	0.9967	-0.7673	-0.492	0.042	-0.05	9.99	0
10	1.25	-1.222	0.384	-1.235	-1.49	-0.8	-0.69	-0.23	-2.09	1.32	0.228	-0.24	1.21	-0.32	0.73	-0.8	0.87	-0.8	-0.683	-0.1	-0.232	-0.483	0.0847	0.3928	0.1611	-0.355	0.026	0.04	121.5	0
11	1.069	0.288	0.829	2.713	-0.18	0.3	-0.1	0.12	-0.22	0.46	-0.774	0.32	-0.01	-0.18	-0.66	-0.2	0.12	-1	-0.983	-0.15	-0.037	0.0744	-0.071	0.1047	0.5483	0.1041	0.021	0.02	27.5	0
12	-2.792	-0.328	1.642	1.767	-0.14	0.8	-0.42	-1.91	0.756	1.15	0.845	0.79	0.37	-0.73	0.41	-0.3	-0.2	0.78	2.2219	-1.58	1.152	0.2222	1.0206	0.0283	-0.2327	-0.236	-0.16	-0.03	58.8	0
12	-0.752	0.345	2.057	-1.469	-1.16	-0.1	-0.61	0	-0.44	0.75	-0.794	-0.77	1.05	-1.07	1.11	1.66	-0.3	-0.4	0.4325	0.263	0.5	1.3537	-0.257	-0.065	-0.0391	-0.087	-0.18	0.13	15.99	0
12	1.103	-0.04	1.267	1.289	-0.74	0.3	-0.59	0.19	0.782	-0.3	-0.45	0.94	0.71	-0.47	0.35	-0.2	-0	-0.6	-0.576	-0.11	-0.025	0.196	0.0138	0.1038	0.3643	-0.382	0.093	0.04	12.99	0
13	-0.437	0.919	0.925	-0.727	0.916	-0.1	0.708	0.09	-0.67	-0.7	0.324	0.28	0.25	-0.29	-0.18	1.14	-0.9	0.68	0.0254	-0.05	-0.195	-0.673	-0.157	-0.888	-0.3424	-0.049	0.08	0.13	0.89	0
14	-5.401	-5.45	1.186	1.736	3.049	-1.8	-1.56	0.16	1.233	0.35	0.917	0.97	-0.27	-0.48	-0.53	0.47	-0.7	0.08	-0.407	-2.2	-0.504	0.9845	2.4586	0.0421	-0.4816	-0.621	0.392	0.95	46.8	0
15	1.493	-1.029	0.455	-1.438	-1.56	-0.7	-1.08	-0.05	-1.98	1.64	1.078	-0.63	-0.42	0.052	-0.04	-0.2	0.3	0.55	0.0542	-0.39	-0.178	-0.175	0.04	0.2958	0.3329	-0.22	0.022	0.01	5	0
16	0.695	-1.362	1.029	0.834	-1.19	1.3	-0.88	0.45	-0.45	0.57	1.019	1.3	0.42	-0.37	-0.81	-2	0.52	0.63	-1.3	-0.14	-0.296	-0.572	-0.051	-0.304	0.072	-0.422	0.087	0.06	231.71	0
17	0.962	0.328	-0.17	2.109	1.13	1.7	0.108	0.52	-1.19	0.72	1.69	0.41	-0.94	0.984	0.71	-0.6	0.4	-1.7	-2.028	-0.27	0.144	0.4025	-0.049	-1.372	0.3908	0.2	0.016	-0.01	34.09	0
18	1.167	0.502	-0.07	2.262	0.429	0.1	0.241	0.14	-0.99	0.92	0.745	-0.53	-2.11	1.127	0	0.42	-0.5	-0.1	-0.817	-0.31	0.019	-0.062	-0.104	-0.37	0.6032	0.1086	-0.04	-0.01	2.28	0
18	0.247	0.278	1.185	-0.093	-1.31	-0.2	-0.95	-1.62	1.544	-0.8	-0.583	0.52	-0.45	0.081	1.56	-1.4	0.78	0.44	2.1778	-0.23	1.65	0.2005	-0.185	0.4231	0.8206	-0.228	0.337	0.25	22.75	0

Figure 2 . Contents of dataset.



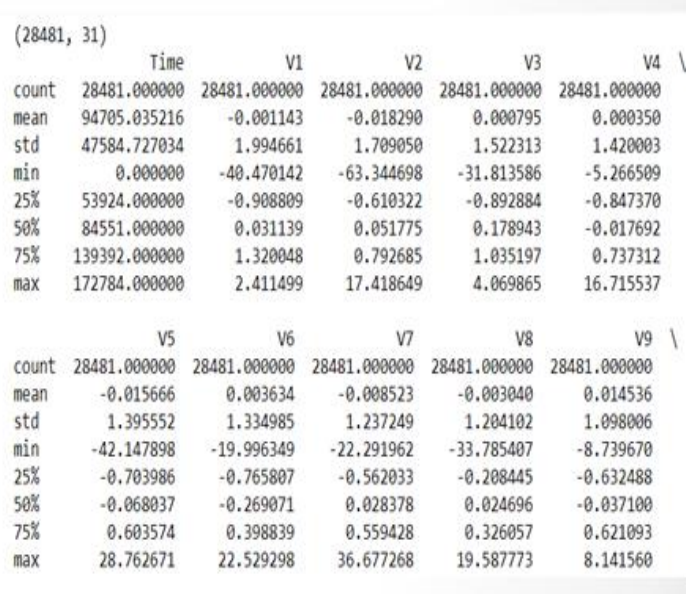
```
print(X.shape)
print(Y.shape)

(28481, 30)
(28481L,)
```

Figure 8 . Showing X and Y values.



```
Index([u'Time', u'V1', u'V2', u'V3', u'V4', u'V5', u'V6', u'V7', u'V8', u'V9',
       u'V10', u'V11', u'V12', u'V13', u'V14', u'V15', u'V16', u'V17', u'V18',
       u'V19', u'V20', u'V21', u'V22', u'V23', u'V24', u'V25', u'V26', u'V27',
       u'V28', u'Amount', u'Class'],
      dtype='object')
```

Figure 3 . Showing 31 columns of our dataset.



```
(28481, 31)
                Time            V1            V2            V3            V4  \
count   28481.000000  28481.000000  28481.000000  28481.000000  28481.000000
mean    94705.035216     -0.001143     -0.018290      0.000795      0.000350
std     47584.727034      1.994661      1.709050      1.522313      1.420003
min         0.000000    -40.470142    -63.344698    -31.813586     -5.266509
25%     53924.000000     -0.908809     -0.610322     -0.892884     -0.847370
50%     84551.000000      0.031139      0.051775      0.178943     -0.017692
75%    139392.000000      1.320048      0.792685      1.035197      0.737312
max    172784.000000      2.411499     17.418649      4.069865     16.715537

                  V5            V6            V7            V8            V9  \
count   28481.000000  28481.000000  28481.000000  28481.000000  28481.000000
mean       -0.015666      0.003634     -0.008523     -0.003040      0.014536
std         1.395552      1.334985      1.237249      1.204102      1.098006
min       -42.147898    -19.996349    -22.291962    -33.785407     -8.739670
25%        -0.703986     -0.765807     -0.562033     -0.208445     -0.632488
50%        -0.068037     -0.269071      0.028378      0.024696     -0.037100
75%         0.603574      0.398839      0.559428      0.326057      0.621093
max        28.762671     22.529298     36.677268     19.587773      8.141560
```

Figure 4 . Showing useful information such as mean,count of our dataset.

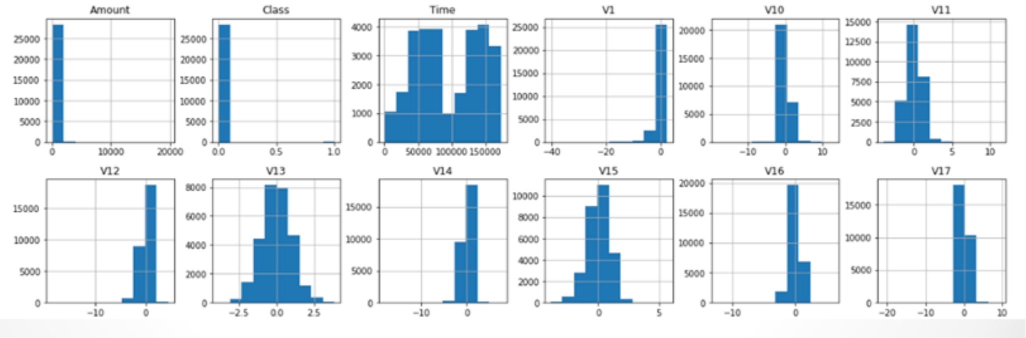
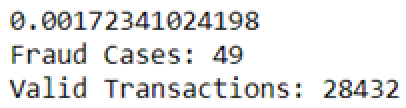


Figure 5 . Showing histogram of each parameter

.



```
0.00172341024198
Fraud Cases: 49
Valid Transactions: 28432
```

Figure 7 . Showing number of valid and fraud cases as well as outlier fraction.

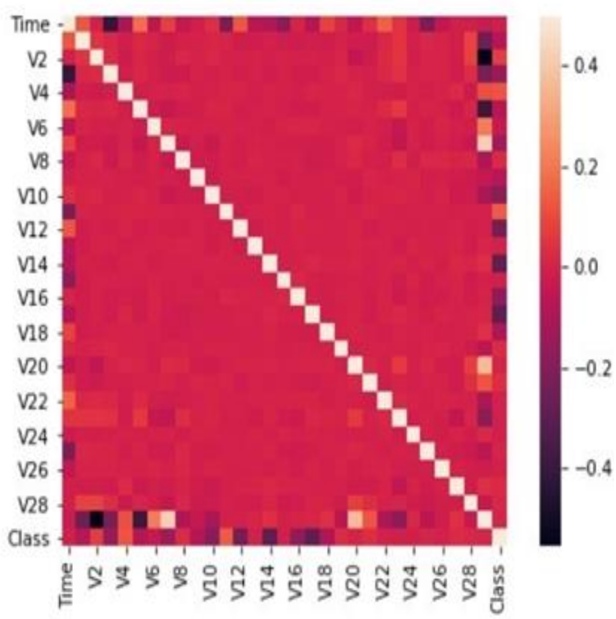


Figure 6 . Showing correlation matrix with heat map.

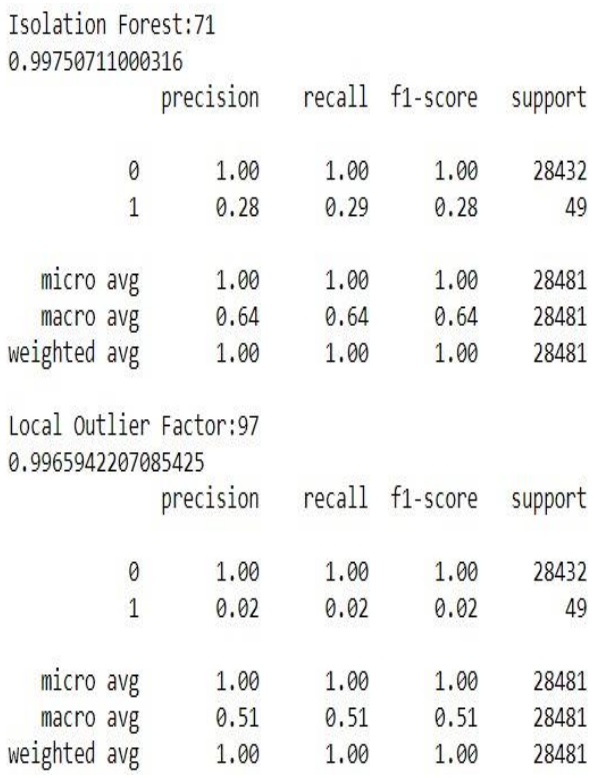
V. IMPLEMENTATION AND WORKING

Earlier SVM i.e. support vector machines were used for outlier detection but it took more time for complex datasets. Isolation forest and local outlier factor are anomaly detection methods provided by sk learn package. In local outlier factor method, the anomaly score of each sample is called Local Outlier Factor. It records the local deviation of density of a given sample with respect to its neighbors. The anomaly score depends on how isolated the object is with respect to the surrounding neighbor. In isolation forest algorithm, it separates observations by casually selecting a feature and then randomly selecting a split value between the highest and lowest values of the selected feature. Recursive partitioning is represented by tree structure so we should know the number of splitting to isolate the sample and that is equal to the path length from root to terminating node. This path length is a measure of normality and decision function. Random partitioning produces noticeably shorter paths for anomalies. Forest of random trees produce shorter path lengths for samples and are more prone to be anomalies. We get the y prediction values which will be negative for outlier and 1 for inlier. It is very useful information but we need to process it before we compare to class label .class label is 1 for fraud event and 0 for valid case. We take all inliers, classify them as o i.e. it indicates valid when we explored the dataset transactions. We take all outliers, classify them as 1 i.e. it indicates fraud transactions . We run classification metrics as it gives useful information such as method name, number of errors,precision,f1 and recall scores.

I. Results

For complex datasets like what we had isolation forest is good method as 30% of time it is able to detect fraud transactions in local outlier factor method, we have 97 total number of errors which is relatively high and accuracy of 99.65942207%.Precision and f1- score are not as good. For class 0 we have precision of 100% and for class 1 it is found to have very less amount of fraudulent transactions.

In Isolation forest method, we have 71 total number of errors which is relatively low and accuracy of 99.750711% For class 1 it is found to have 30% precision. f1 scores are good for isolation forest compared to local outlier factor method. Isolation forest method was able to produce better results as shown in Figure 9.



```
Isolation Forest:71
0.99750711000316
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     28432
           1       0.28      0.29      0.28        49

   micro avg       1.00      1.00      1.00     28481
   macro avg       0.64      0.64      0.64     28481
weighted avg       1.00      1.00      1.00     28481

Local Outlier Factor:97
0.9965942207085425
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     28432
           1       0.02      0.02      0.02        49

   micro avg       1.00      1.00      1.00     28481
   macro avg       0.51      0.51      0.51     28481
weighted avg       1.00      1.00      1.00     28481
```

Figure 9. Showing the method name, total number of errors,precision,f1,recall scores.

VI. CONCLUSION AND FUTURE WORK

We imported csv data set, preprocessed it, exploring and describing data. And plotting histogram to check unusual parameters. We did correlation matrix to know which parameters important for our class. Two algorithm used are

Isolation forest and local outlier factor to do anomaly detection. In the dataset. We realized the importance of understanding the data and precision.

We notice that Isolation Forest is good when compared to Local Outlier Factor in terms of accuracy, number of errors, precision, f1 and recall scores. In future, we can use Neural Networks to train our system for still higher accuracy [5]. We imported csv data set, preprocessed it, exploring and describing data. And plotting histogram to check unusual parameters. We did correlation matrix to know which parameters important for our class. Two algorithms used are Isolation forest and local outlier factor to do anomaly detection. In the dataset, We realized the importance of understanding the data and precision. Fraud detection is a complex issue that requires a substantial amount of planning before throwing machine learning algorithms at it. Nonetheless, it is also an application of data science and machine learning for the good, which makes sure that the customer’s money is safe and not easily tampered with. Future work will also include implementing the system by using neural networks to train the system for increasing efficiency. Having a data set with non-anonymized features would make this particularly interesting as outputting the feature importance would enable one to see what specific factors are most important for detecting fraudulent transactions.

Some of the advantages are:

- Reduction in number of fraud transactions.
- User can safely use his credit card for online transaction.
- Added layer of security.

Some drawbacks that can be further improved upon are:

- Machine learning algorithms work only for huge sets of data. For smaller amount of data the results may be not accurate. It takes a significant amount of data for machine learning models to become accurate. For large organizations, this data volume is not an issue but for others, there must be enough data points to identify legitimate cause and effect relations.

REFERENCES

[1] Datasets. (n.d.). Retrieved from https://www.kaggle.com/datasets

[2] A. Srivastava, M. Yadav, S. Basu, S. Salunkhe and M. Shabad, "Credit card fraud detection at merchant side using neural networks," *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 667-670.

[3] 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. doi: 10.1109/JCAI.2009.146

[4] Eduonix.(2018,July26).Eduonix/creditcardML.Retrievedfrom https://github.com/eduonix/creditcardML

[5] https://pythonprogramming.net/neural-networks-machine-learning-tutorial/