Turkish Journal of Computer and Mathematics Education Vol.12 No.12 (2021), 837-*846*

Research Article

Fraud Detection in Credit Card Transactions using Anomaly Detection

Asheesh Kumar Dwivedi[1], Ashish Kumar Rai[2], Ashish Kashyap[3]

[1]Dept. of CSE
Galgotias University
Greater Noida India
[2]Dept. of CSE
Galgotias University
Greater Noida India
ak2466719@gmail.com
[3]Dept. of CSE
Galgotias University
Greater Noida India
Asheeshkumardwivedi007@gmail.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

Abstract: Credit Card is a convenient payment mode. It is useful for both online and offline modes of payment. For online, we need to use the Credit Card Number. The Credit Card Number is sufficient for online transactions and that comes with a risk. We have fraud transaction detection systems but they can detect it only after the occurrence of transactions. The Organizations keep the detailed data consisting of genuine transactions as well as fraudulent transactions. The fraudulent are generally caught following a particular pattern. It is a difficult task to analyze each and every transaction data among about millions and billions of them. Predictive Algorithms could be a valuable asset for the detection of fraudulent transactions, here we need Data Mining. A variety of statistical tests could be used for the prevention of fraud events .However, we still have no perfect method for detecting fraudulent transactions. To, the banks, these frauds are a major financial issues. The detection of fraudulent transactions among the genuine transactions is totally skewed towards the latter. According the estimation, out of 12 billion transactions made in a year, 10 million are frauds. We are using isolation forest algorithm and local outlier factor algorithm to analyze and predict the frauds. The accuracy and errors of both the data has also been computed.

Keywords: Local Outlier Factor Algorithm, Isolation Forest Algorithm, Fraud Detection, Credit Card, Data Mining, Anomaly Detection

I. Introduction

In our day to day lives Credit Cards are used in daily lives to buy services and goods using online transactions or offline transactions. In an offline purchase , the customer uses his physical card to for the payment. If the transaction is to be made fraudulent, the attacker needs to steal the card. If the user is unaware of his lost card, it results in financial losses, for both the user and the credit card company. In case of an online payment, the attackers, need only little information to cause a fraud transaction. This 'little information' could be the card number. The sole method of detecting these types of fraud is examining the patterns of transactions of each card and realizing the abnormalities with respect to the normal pattern. The detected frauds with the help of the purchase data of the card user can be used to lessen the fraudulent transactions. Each and every Credit Card User has a specific pattern , that contains, information and data regarding purchase , the elapsed time since last buy, money used for the purchase etc. the irregularity from such pattern is recognized as fraudulent transaction. These Frauds are the issues, in finance, that can result in, many consequences. We can define fraud as a criminal cheating that aims financial gain. The internet's frequent use has resulted to, a hike in the online transactions using credit card. The Credit Card also attracts more vulnerable and fraud events. The fraud mainly takes place because many a times, the credit card detail and data of an individual is misappropriated, for making illegitimate acquisition of items, withdrawing money. Online shopping is one of the most popular trends and the various payment methods are net banking, debit card and credit card. They eliminate any need of any physical card. If others come to know the details, it becomes a risk. The card holder realizes the fraud only after it has occurred. No system/model actually exists for detecting a fraud transaction. In this project we use a dataset of about 29,000 transactions and more than one unsupervised anomaly detection algorithms to detect transactions with good chances of being fraudulent transactions. Also, we will be, using F1 scores, recall and precision to check the reason of the efficiency of classification of the algorithms being misleading. Further, we would be

exploring the data visualization techniques , which are commonly used in Data Science, like correlation matrices, histograms and parameters for acquiring much better understanding of the data in the dataset, used by us.

II. Literature Reviews

In [2] the authors have started, explaining the process involved in the credit card transactions. A system has been proposed in which their algorithms are integrated with the payment gateway for the detections of real time frauds. 7 techniques have been used by the author to develop the required algorithm. These techniques involve Neural Network, Case-based Reasoning , Inductive Logic Programming, Rule Induction, Genetic Algorithms, Regression and Expert Systems. It is also said that Artificial Neural Network would be the best to serve the problem statement. The output of the ANN, to tell the degree of transaction being fraudulent would be in the form of probability. The information, which is based on different categories about the card user like, profession, earnings, etc. is used to train the Neural Network. Back Propagation learning algorithm will be used by the system here to train the network. The Transaction is to be grouped among one of the mentioned categories: Fraudulent and Non-Fraudulent. This classification will take place depending on the numeric value between 0 and 1. This system under development is particularly beneficial for the merchants, by reducing their losses which, they have to face if the transaction occurred, is fraud. The Authors have also focused on the Chinese market due to its rapid growth and fast pace[3]. The authors have also proposed using outlier detection which uses distance sum to detect the fraudulent transaction. This is a data mining technique. This method is preferred over the traditional statistical methods like Discriminant analysis and Regression due to the independence of the outlier detection method, from the distribution of dataset. In this paper we are using Euclidean distance to calculate distance sum for the detection of the outliers. For distance, the they (authors) have computed a threshold value. The distance, if more than the calculated threshold, the object is classified as a fraudulent transaction. The data, having around 16,000 observations, has been accumulated from a Chinese bank,. The maximum accuracy of 89.4% has been recorded for the threshold value of 12. This process highly depends on the nature of the data distribution and may vary for the data of other banks.

Also in [4], the authors have tried to analyze, how algorithms like Random Forest, Decision Tree and Logistic Regression (in R language) perform on the dataset with approximately 2,85,000 transaction data of a dataset. When we implemented those algorithms on our dataset, we obtained the accuracies of Decision Tree, Random Forest and Logistic Regression as 94.3, 95.5 and 90 respectively. Random Forest is the most accurate technique among the three.

III. Challenges

Some of the challenges that we need to face are:-

1) Huge amount of data is processed everyday, so the system built must be fast enough to detect scam in time.
2) Data is imbalanced i.e. most of the transactions are genuine, which makes it difficult for detecting the fraud ones.
3) Data availability is a challenge because the data is mostly private.
4) The Data is misclassified, which is another major issue, as not every fraud is caught.
5) The Scammers use Adaptive techniques against the system.

A few ways to tackle the challenges:-

1) The system which is being used must be fast enough to detect the anomaly and distinguish it as a fraud, instantly.
2) For, protecting the privacy of the users, the dimensionality of the data can be reduced.
3) We can take a more trustworthy source, for double-checking the data, at least to train the model.
4) The system can be made simple and interpretable so that, when the attacker adapts to it with just some tweaks we can have a new system up and running to deploy.

IV. System Design

Our Fraud detection module works as follows:-

1) The transactions and amount incoming are considered credit card transactions

2) The incoming Transactions are used as an input to the machine learning algorithms.

3) By, examining data, and observing the, pattern and using machine learning algorithms such as isolation forest algorithm and local outlier factor algorithm for doing anomaly detection, the output will be resulting in either fraud or valid transaction.

4) Alarm takes the fraud transactions , to alert the user in case, a fraud transaction has taken place and the card could be blocked for avoiding further financial losses to the user and the company of the credit card.

5) The Genuine Transactions contain the true transactions.

V. Implementation The Software Model

1) The dataset has been collected from kaggle [1]. The source-code has been collected from github[5]. The contents of the dataset are credit card transactions made in the month of September ,in the year, 2013, by European card Holders as shown in figure 2.
2) The libraries have been imported and the versions have been printed in our documentation. Then the necessary packages have been imported.
3) Dataset has been loaded, using pandas, from the .csv file. After exploring through dataset, we found that it has 31 distinct columns as shown in figure 3.
4) To ensure the protection of sensitive information, in our dataset, like identity and location of an individual, PCA dimensionality reduction has been used, which has resulted to columns from V1 to V28.
5) Here valid transactions are indicated by class 0 and fraud transactions are detected by class 1.
6) The dataset contains 284807 rows with 31 columns. After examining the dataset further, we saw the mean values, being near 0, (figure 4). This means that the amount of valid transactions is greater than the fraud ones in the dataset.
7) As it is a huge dataset, so in order to save time and computation, we took a small fraction (20%) of the data. So after this we had only 56961 transactions remaining.
8) After this, we plotted the histogram of each parameter (figure 5). Then, we computed, the fraudulent and the genuine cases, and the outlier fraction (number of fraud transactions divided by the valid ones) (figure 6).
9) Also, the correlation matrix was constructed along with the heat-map, to check, whether there was a strong correlation between the variables of the dataset (figure 8). It also determines which features are significant for the total classification. But it was seen that, the majority (values) were around 0 and hence, there wasn't any strong relationship among the V-parameters.
10) We filtered the columns to remove the unwanted data.
11) We only stored the variables, required for prediction i.e. X contains all columns, other than the class label and Y is what we are in the need of i.e. it it's a single-dimensional array containing label for the samples (figure 7). This method is Unsupervised learning , so we didn't want the labels.

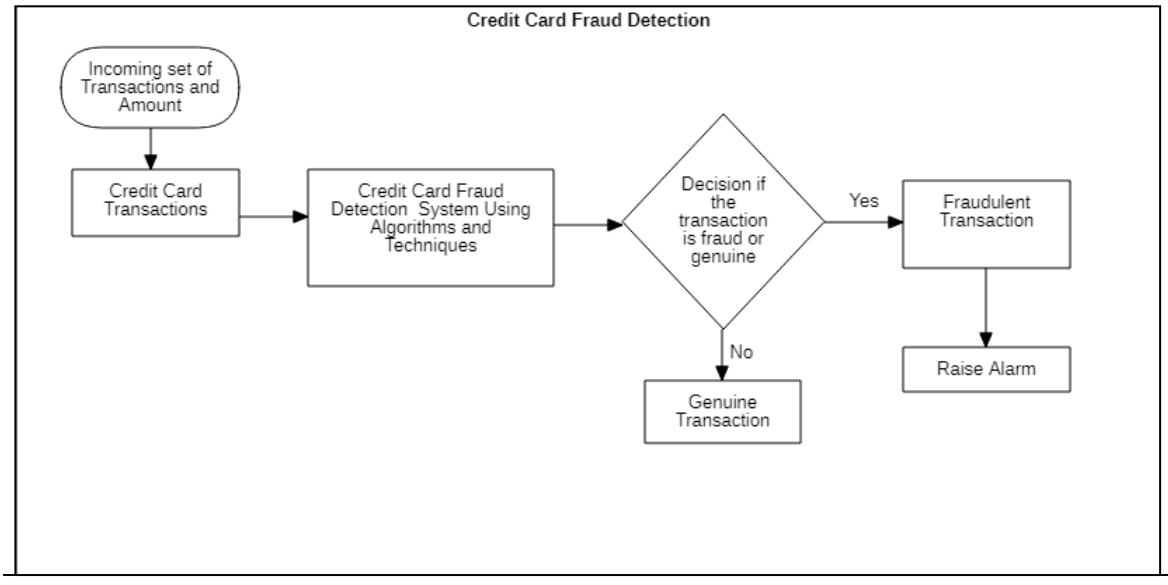

Figure 1. F Diagram of the Model

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	-0.01	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. The dataset

```
Index(['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'],
      dtype='object')
```

Figure 3. columns of the dataset

```
                V25           V26           V27           V28         Amount  \
count  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05  284807.000000
mean   5.340915e-16  1.687098e-15 -3.666453e-16 -1.220404e-16      88.349619
std    5.212781e-01  4.822270e-01  4.036325e-01  3.300833e-01     250.120109
min   -1.029540e+01 -2.604551e+00 -2.256568e+01 -1.543008e+01       0.000000
25%   -3.171451e-01 -3.269839e-01 -7.083953e-02 -5.295979e-02       5.600000
50%    1.659350e-02 -5.213911e-02  1.342146e-03  1.124383e-02      22.000000
75%    3.507156e-01  2.409522e-01  9.104512e-02  7.827995e-02      77.165000
max    7.519589e+00  3.517346e+00  3.161220e+01  3.384781e+01   25691.160000

               Class
count  284807.000000
mean        0.001727
std         0.041527
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000

[8 rows x 31 columns]
```

Figure 4. Useful Information(count,mean etc) of the data in dataset

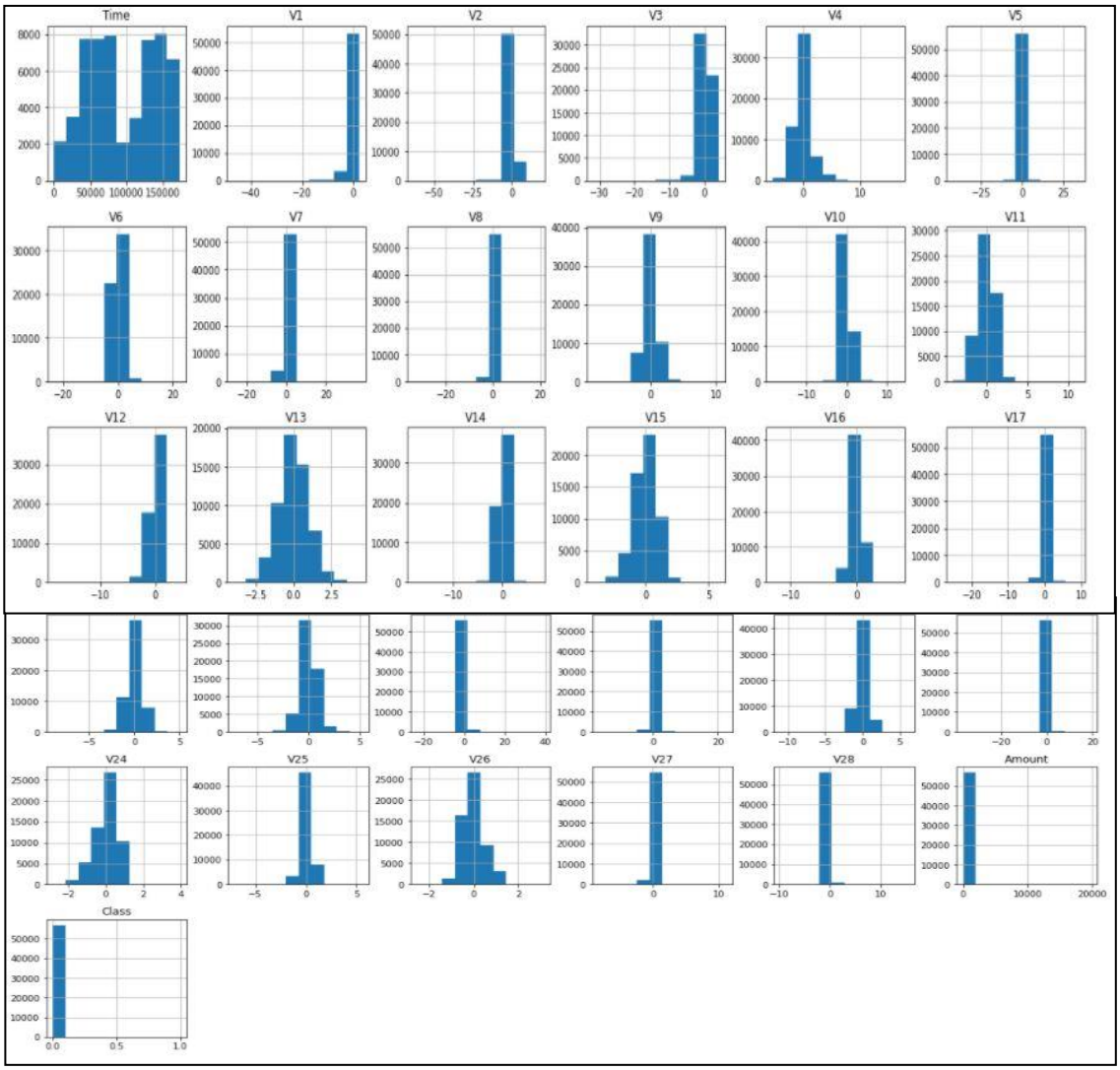

Figure 5. Histogram of every Parameter

```
0.0015296972254457222
Fraud Cases 87
Valid Cases 56874
```

Figure 6. The valid cases, fraud cases and outlier fraction

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

(56961, 30)
(56961,)
```

Figure 7. The X (all the columns other than class label)
and Y (array having class labels for samples)

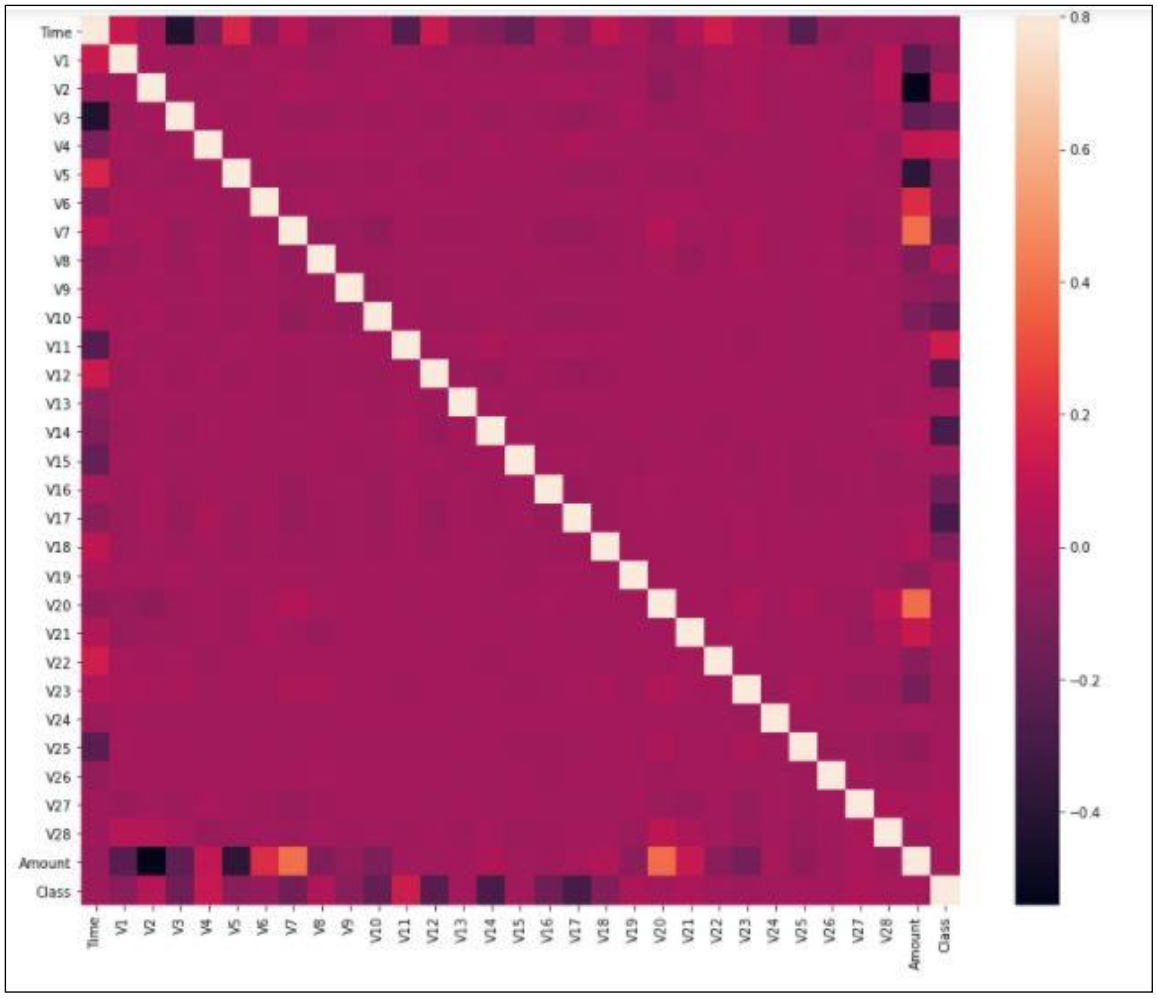

Figure 8. The Correlation-Matrix along with the Heat-Map

VI. Process and Working

Previously, Support Vector Machines (SVM) were relied on for the detection of outlier, but it was time-consuming when it came to complex datasets. Isolation forest and Local Outlier Factor are provided by sklearn package and are Anomaly Detection Methods. The Score of Anomaly of a sample is called Local Outlier Factor in case of the Local Outlier Factor Algorithm. The main significance of the local outlier factor is, that it records the local deviation of density of the sample in relation to its neighbor. However, in case of the Isolation Forest Algorithm, its use is that it separates observations by haphazardly choosing a feature , then haphazardly choosing a split value between the maximum and minimum values of the chosen feature. The Tree Structure is used for representing, the recursive partitioning, for us to understand the number of splitting, for the sample isolation and is equal to the path length, from root to the terminal node, which is the measure, of decision function and normality. The shorter paths, for the anomalies, can be produced by Random Partitioning. For the samples, Forest of random trees produce shorter paths and they are more reclining to be anomalous. The y prediction values, that we get, would be negative for the outlier and for the inlier, 1. But we need to process this information, before the comparison of it to the class label, where class label 1 represents fraud event and 0 represents genuine events. Classification metrics is run. It provides necessary details, such as precision, method name, recall and f1 scores and number of errors.

VII. Evaluation metrics

To classify the transactions as fraudulent and genuine we use different standards apart from accuracy like :--

- Precision
- Recall

- F1-Scores
- Support

These Standards however, are dependent on the 'Actual Class' and the 'Predict Class', so we are using a confusion matrix(figure 9.) of 2x2 to understand more.

		Predicted Class	
		Negative	Positive
Actual Class	Negative	True Negative (TN)	False positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Figure 9. The Confusion Matrix

True Positive: The values of actual class as well as the predicted class are 'YES'.
True Negative: The values of both actual class and predicted class are 'NO'.
False Positive: The value of the actual class is 'NO' and the value of the predicted class is 'YES'.
False Negative: The value of the actual class is 'YES' and the value of the predicted class is 'NO'.

When there is a contradiction between, the Actual and the Predicted Classes, this results in the False Positive and the False Negative classes.

The Standards of Correctness are calculated as follows :--
Precision: Precision= (TP)/(TP x FP)
Recall: Recall= (TP)/(TP + FP)
F1-Score:
F1-Score= 2 x (Recall x Precision)/(Recall + Precision)
Support: It is the number of actual occurrences of any class.

Results

In complex datasets, like the one we have used, isolation forest proves to be a good method as in 30% of all times, it can detect fraudulent transactions.

In case of Local Outlier factor Algorithm, the total number of errors is 173, and that's comparatively high, and it is 99.696% (approx.) accurate. f1-score and precision are not that good. We have a precision of 100% for class 0 and very less amount of fraudulent transactions are found for class 1.

In case of Isolation Forest Algorithm, the total number of errors is 127, and that's relatively low, and it is 99.777% (approx.) accurate. We get 30% precision for class 1. F1-scores are better than those of the local outlier factor algorithm.

Isolation Forest Method has given us better results.
We have also compared our methods, Isolation Forest Algorithm and Local Outlier Factor Algorithm .

```
IsolationForest : 127
0.997770404311722
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56874
           1       0.27      0.28      0.27        87
```

Figure 10. The Results of the Isolation Forest Algorithm (0 states the valid transactions and 1 states the fraud transactions)

```
Local Outlier Factor : 173
0.9969628342199048
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56874
           1       0.01      0.01      0.01        87
```

Figure 11. The Results of Local outlier factor algorithm (0 states the valid transactions and 1 states the fraud transactions)

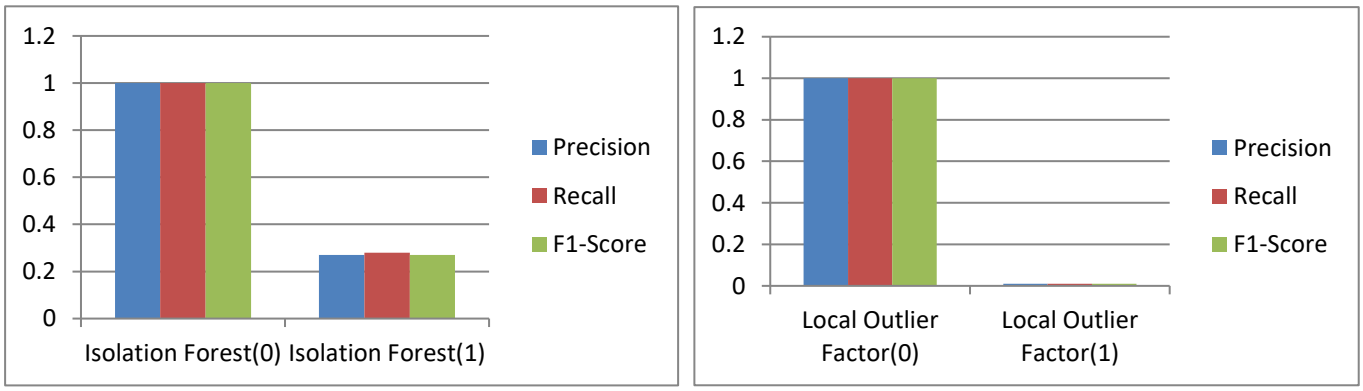

Figure 12. Variation charts for the Isolation Forest Algorithm and the Local Outlier factor algorithms

Algorithm	**Accuracy(%)**
Random Forest	95.5
Decision Tree	94.3
Logistic Regression	90
Isolation Forest	99.77
Local Outlier Factor	99.69

Table 1. Comparison of Different Algorithms

Figure 13. Graphical representation of the Comparison

Conclusion

The dataset of type (.csv) was imported, pre-processed, explored and described, histogram was plotted, to check the unusual parameters. Correlation matrix has been done to know the important parameters for the class. The algorithms being used by us are Isolation Forest Algorithm and Local Outlier Factor Algorithm for doing the anomaly detection. We have also understood the significance of examining, precision and data.

We have also noticed that, compared to local outlier factor, Isolation Forest has relatively better efficiency, precision, f1 and recall scores. Neural Networks could be used in future to train the system for being more accurate [5]. Fraud detection in credit card needs a lot of planning, before applying, the algorithms of Machine Learning to it. Hence, we can say that it is a complex issue. However, it makes sure that the card user's finance is safe . So, we can also say that, it is the application of machine learning and data science, made for the welfare of the people.

Our Proposed methods gave us the highest accuracies(table 1 and figure 12).

Implementation of the system, using neural networks, for training the system, to obtain better accuracy, will be included in the Future Work.

The following are the advantages:--

1) Reduced number of fraud transactions.
2) Credit Cards can be safely used, for the online transactions, by the user.
3) There is more security.

There are a few disadvantages, they are as follows:-

1) Huge Datasets are good for the machine learning algorithms to work. For less amount of data, the sresult might be inaccurate.
2) Quite a lot of data, would be needed for the machine learning algorithms to be more accurate.

REFERENCES

1. Dataset collected from https://www.kaggle.com/datasets
    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*
2. *Development (INDIA.com)*, 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. "Ensemble learning for credit card fraud detection," by I Sohony, R Pratap, and U Nambiar, 2018.
5. Eduonix.(2018,July26).Eduonix/creditcardML.
6. Retrieved from https://github.com/eduonix/creditcardML
7. https://pythonprogramming.net/neural-networks-machine-learningtutorial/
8. "Credit Card Fraud Detection Using Machine Learning methodologies" by H. A. Shukur ,2019.
9. "Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy", IEEE , 2018.
10. "nilsonreport.com." https://nilsonreport.com/upload/content_promo/The_Nilson_Report_10- 17-2016.pdf [Accessed 6 December 2020].
11. "Comparative Analysis of Machine Learning Algorithm through Credit Card Fraud Detection" by
12. R Banerjee, G Bourla, S Chen, S Purohit, and J Battipagli, 2018.
13. "Credit Card Fraud Detection using Local Outlier Factor", *Int. J. Pure Appl. Math.,* by D Tripathi, T Lone, Y Sharma, and S Dwivedi, 2018.
14. "Credit Card Fraud Detection Using AdaBoost and Majority Voting", *IEEE Access*, by C P Lim, M Seera, A K Nandi, K. Randhawa, and C. K. Loo,2018.

15. "Local outlier factor", *En.wikipedia.org*, 2020 https://en.wikipedia.org/wiki/Local_outlier_factor.
16. [Accessed 06 December 2020].
17. "Isolation forests for anomaly detection improve fraud detection.", *Blog Total Fraud Protection*, 2019 [Online].
18. https://blog.easysol.net/using-isolation-forests-anamoly-detection/ [Accessed 06 December 2020].
19. "Credit Card Fraud Detection", *Ijarcce*, vol. 5, I. Trivedi, M. M, and M. Mridushi,, 2016.