Fraud Detection using Machine Learning

- Anish Akode

(Dt: 10-05-2022)

1. *Problem Statement*

The world is moving forward with digital modes of payments and have seen a lot of traction. Customers benefit from digital payments because they make financial transactions easier. However, this also invited many fraudsters to make use of gaps and dupe the customers in different ways.

According to a study by Experian, over 90% of consumers around the world rely on online payments for purchasing goods and services. This increase in online payments, however, brings with it an increase in transaction fraud.

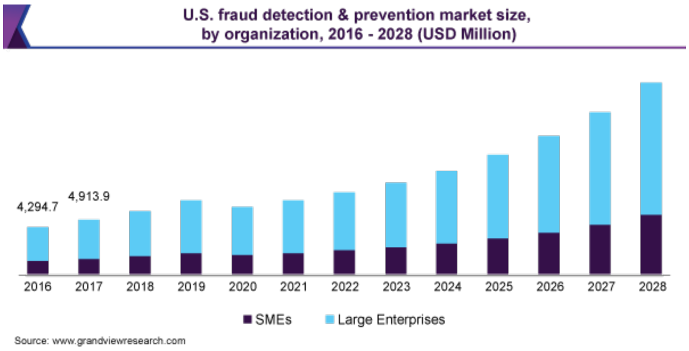
What is payment fraud?

Payment fraud occurs when someone steals another person's payment information and uses it to make unauthorized transactions or purchases. The actual cardholder or owner of the payment information then notices their account being used for transactions or purchases they did not authorize, and raises a dispute.

Here the aim is to build a machine learning model which can predict whether a fraud can occur. This helps people to be warned before hand during a transaction and can help to improve the security of payment/transactions from one domain to another domain.

2. *Market Need Assessment*

The global fraud detection and prevention market size was valued at USD 20.98 billion in 2020 and is expected to grow at a compound annual growth rate (CAGR) of 15.4% from 2021 to 2028. The growing concerns regarding digital frauds, despite technological advancements facilitating ease of payment options or data access, calls for the deployment of fraud detection solutions.



Source: www.grandviewresearch.com

3. *Target Specifications and Characterization*

As mentioned in the problem statement most the victims to fraud are lower to middle income countries so there is a need for a cheap and quick predictor of frauds. A computer-based model can never be as accurate, however here the aim to detect and prevent fraud happening.

Fraud detection and prevention is not a static process. There's no starting and ending point. Rather it's an ongoing cycle which involves monitoring, detection, improvements in the model. Here the main moto is to detect the fraud happening pre hand and prevent it. The model is trained on the features:

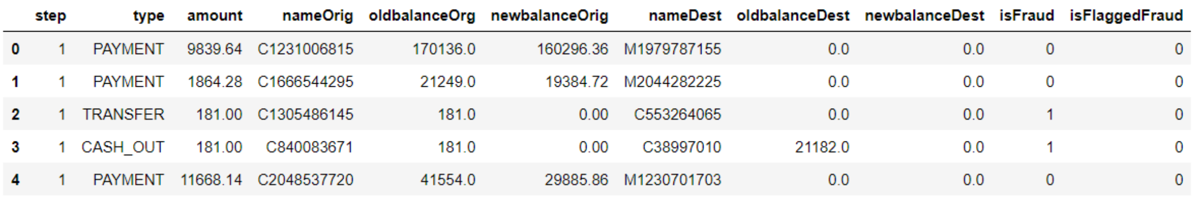
- step: represents a unit of time where 1 step equals 1 hour
- type: type of online transaction
- amount: the amount of the transaction
- oldbalanceOrg: balance before the transaction
- oldbalanceDest: initial balance of recipient before the transaction

and many more. If the input is aligning towards a fraud transaction, then the model will detect using the feature variables in the input and predict the fraud.

4. *External References*

The dataset is taken from Kaggle.com which contains 63,62,620 records, which are the previous entries whether the fraud occurred or not.

Here is the sample dataset:



	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0	0
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0	0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1	0
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1	0
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0	0

5. *Benchmarking*

In real world there is no perfect computer-based model which can predict the fraud accurately. As there is no starting and ending point for the fraud detection, rather it is a cycle which continues and help the model to improve based on previous experience of fraud occurred.

I can say that the model which I have trained on the dataset provided above, has a pretty fine accuracy rate and recall rate. Therefore, I say that this model can be used for fraud prediction.

6. *Applicable Patents*

As this is an open source and real term model used for social development, there are no applicable patents. As it helps in the welfare of the society it must be available for free. Since it is free to use and a continuously improving service using the previous data, there must some data security between the developer and the consumer.

7. *Applicable Regulations*

- Regulations related to banking sector
- Ownership regulations

8. *Applicable Constraints*

1. Machine Learning Expert: There are many patterns in which the fraud can happen, there should be an ML expert who can supervising, tunning and updating the model.
2. Data Analyst: The role of data analyst is to process the previous data and find some patterns and predict the future results.
3. Software Developer: To get the model reachable to the public, there are mobile and web platforms. To handle those issues there must be a software developer who can handle the bug issues etc.
4. Data Engineer: He is the one who handles the servers and networks to ensure their smooth functioning. As we present our product on various platform, we need a server to communicate with the users and preserve the transaction details.

9. *Business Opportunity*

As the previous mentioned reports speak that there might be a growth of fraud of nearly 15.4% from 2021 to 2028. 15% is not a less value and we cannot image the data been stolen or money within this 15%. Machine learning models are able to learn from patterns of normal behaviour. They are very fast to adapt to changes in that normal behaviour and can quickly identify patterns of fraud transactions.

I can say that with the help of this product/model we can try reducing the frauds happening but cannot assure to make fraud-free system. With this product not only higher-class people benefit, but it is a universal model where there are no kind of barrier involved.

10. *Concept Generation*

Fraud has touched nearly every area of business – from data breaches that affect end customers' privacy rights and payment security, to ransom attacks that demand vast sums of money from organizations.

During the COVID-19 phase all we into remote working, distance learning and home isolations. Due to which everything went online and the new normal rendered individuals and organizations exponentially more vulnerable to cyberattacks, creating a field day for cybercriminals. Fraudulent activity became a lot more complex to detect on time and prevent before it took a toll on business.

11. *Concept Development*

The concept behind using machine learning in fraud detection is that fraudulent transactions have specific features that legitimate transactions do not. Based on this assumption, machine learning algorithms detect patterns in financial operations and decide whether a given transaction is legitimate. Machine learning fraud detection algorithms are way more effective than humans. They can process a raft of information faster than a team of the best analysts ever could.

ML algorithms can spot patterns that seem unrelated or go unnoticed by a human. By exploring and studying tons of cases of fraudulent behavior, ML algorithms determine the most stealthy fraudulent patterns and remember them forever.

To detect fraud, a machine learning model first needs to have some input data. The model analyses all the data gathered, segments, and extracts the required features from it. Next, the machine learning model receives training sets that teach it to predict the probability of fraud. Finally, it creates fraud detection machine learning models.

12. *Final Product Prototype*

Back End

i. In this area mainly the Data Engineer, ML Engineer work.
ii. To store datasets, data graphs, payments info etc.
iii. The main component of this product should be kept safe.
iv. Previous model should be kept updated with the new upcoming dataset.
v. System Administrator should see upon the databases.

Front End

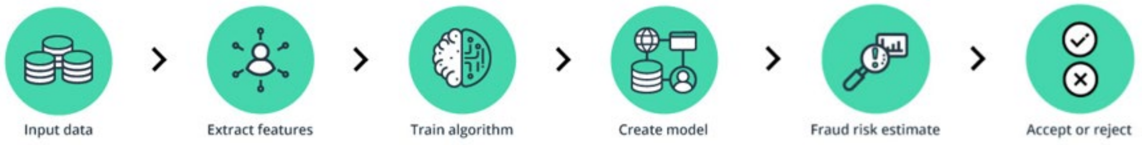
i. In this area mainly the Software developer and UI/UX team work.
ii. As in the backend the model will get updated, so there should be change at the front end.
iii. SDE should be responsible for updating the website/app on which our model is running.
iv. UI/UX designers should create an attractive and easy go design for the product.

13. *Product details*

How does it work?

The first step, data input, differs for Machine and humans. In the case of humans, they struggle to handle and understand large datasets and require a lot of time for this to happen, whereas for machines they understand pretty easily and requires very less amount of time to learn. The more data an ML model receives, the better it can learn its fraud detection skills.

Feature extraction is the next step. At this point, features describing good customer behaviour and fraudulent behaviour are added. These feature variables determine the output for an input given to the model. Based on the complexity of the fraud detection system, the list of investigated features can differ.



Next, a training algorithm. Here the data is split into train and test datasets. The train data set is used to train the model i.e., with the help of the algorithm it will find some patterns regarding the given problem. The test data set is given to the model when the training of the model is successfully done. Here the train and test dataset should be kept separate. The more data a business can provide for a training set, the better the ML model will be.

After the training of the model is done successfully, we need to test how our model is performing on the test dataset. Such that we can know how accurately our model is predicting the results. If our model is preforming pretty well then, we can proceed forward or else we need to tune the model for better results.

Finally, when the evaluation of the model is over, the company receives a fraud detection machine learning model suitable for their business. This model can detect fraud in next to no time with high accuracy. To be effective a machine learning model needs to be constantly improved and updated. Payment fraud detection can be eliminated for a while using ML. But sooner or later, fraudsters will come up with new tricks to game the system unless you keep it updated.

Data Source

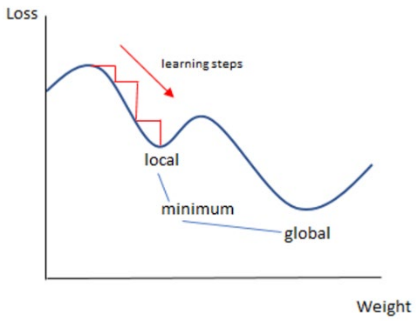
The dataset is taken from Kaggle.com as mentioned before, the dataset contains various features which help the model to train on. On the basis of the data present the model is trained and will help the model to update in the future.

Algorithms

Stochastic Gradient Descent (SGD):

Stochastic Gradient Descent is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing.

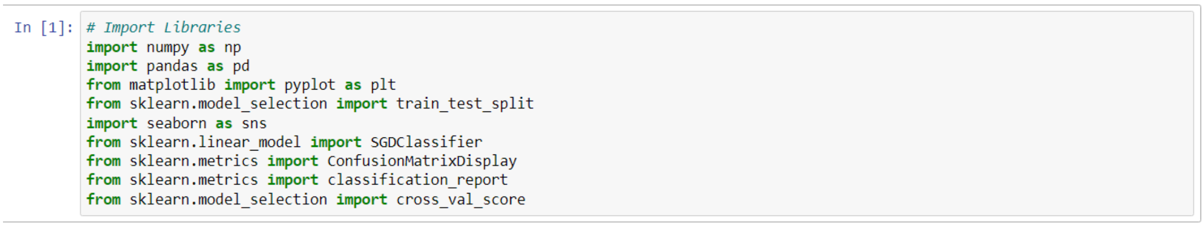


Team required to develop

- Machine Learning Engineer
- Data Engineer
- Software Developer
- UI/UX Developer

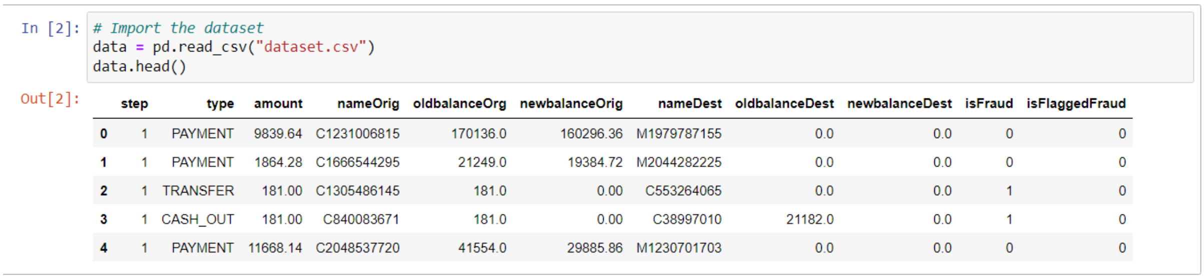
14.Code Implementation

Importing the required Modules and libraries required



```
In [1]: # Import Libraries
        import numpy as np
        import pandas as pd
        from matplotlib import pyplot as plt
        from sklearn.model_selection import train_test_split
        import seaborn as sns
        from sklearn.linear_model import SGDClassifier
        from sklearn.metrics import ConfusionMatrixDisplay
        from sklearn.metrics import classification_report
        from sklearn.model_selection import cross_val_score
```

Importing the dataset from the local machine using pandas and making a data frame.

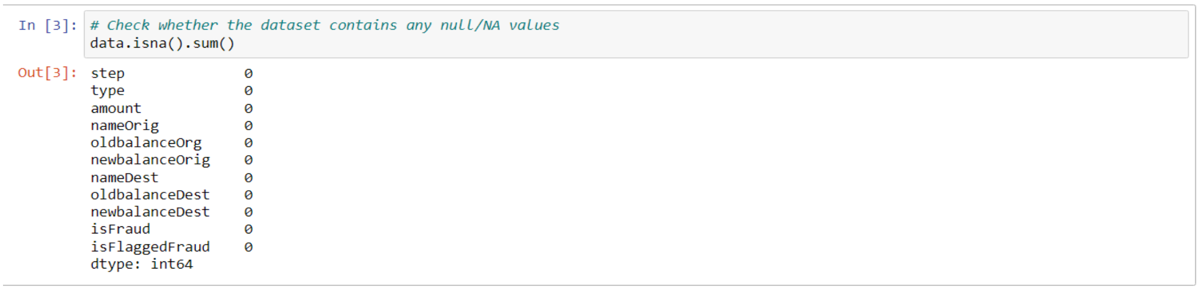


```
In [2]: # Import the dataset
        data = pd.read_csv("dataset.csv")
        data.head()
```

Out[2]:

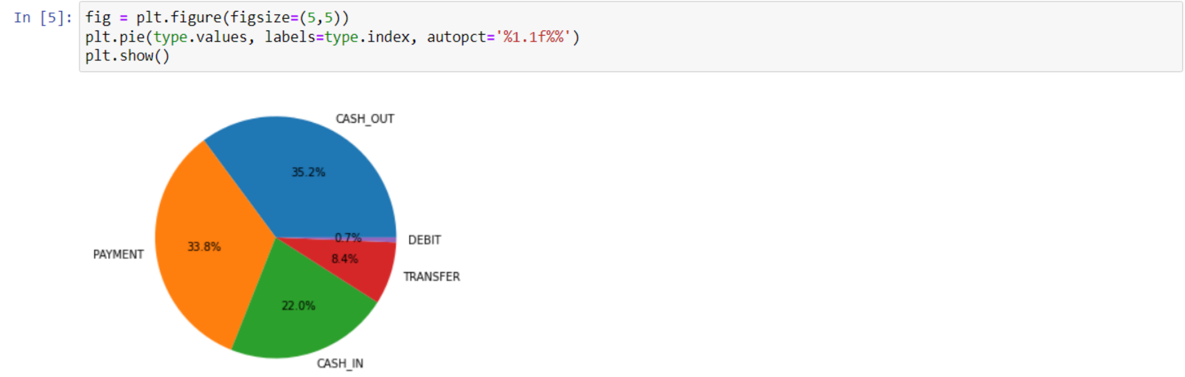
	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0	0
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0	0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1	0
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1	0
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0	0

Check whether there are any missing (NULL/NA) values present in the data frame



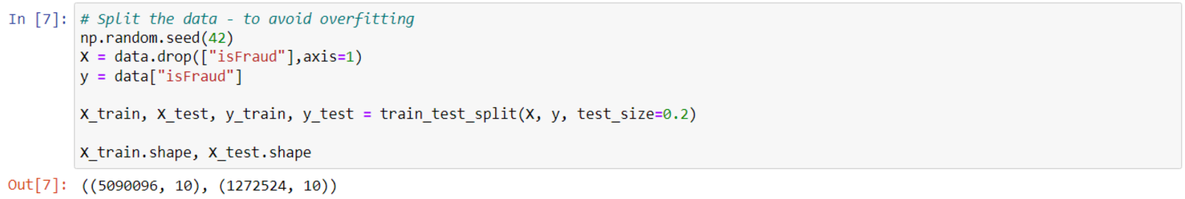
```
In [3]: # Check whether the dataset contains any null/NA values
        data.isna().sum()

Out[3]: step              0
        type              0
        amount            0
        nameOrig          0
        oldbalanceOrg     0
        newbalanceOrig    0
        nameDest          0
        oldbalanceDest    0
        newbalanceDest    0
        isFraud           0
        isFlaggedFraud    0
        dtype: int64
```

Analyse the data using graphs and charts



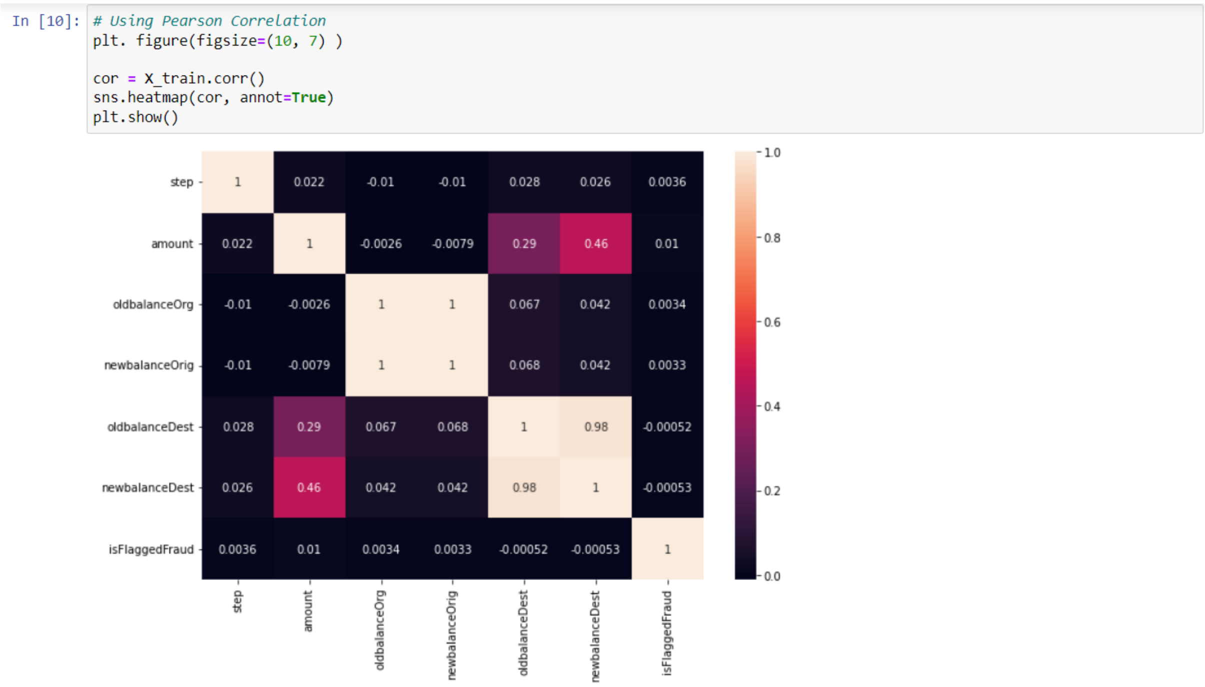
```
In [5]: fig = plt.figure(figsize=(5,5))
        plt.pie(type.values, labels=type.index, autopct='%1.1f%%')
        plt.show()
```

Split the data into train and test to avoid over fitting of the model



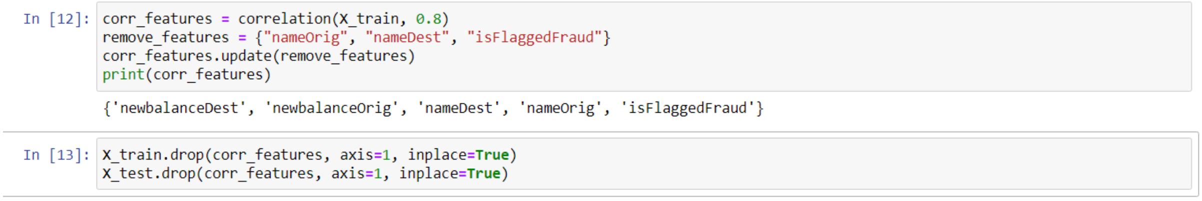
```
In [7]: # Split the data - to avoid overfitting
        np.random.seed(42)
        X = data.drop(["isFraud"],axis=1)
        y = data["isFraud"]

        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

        X_train.shape, X_test.shape

Out[7]: ((5090096, 10), (1272524, 10))
```

Check for any correlation among the feature variable. Correlation is a statistical measure that expresses the extent to which two variables are linearly related.



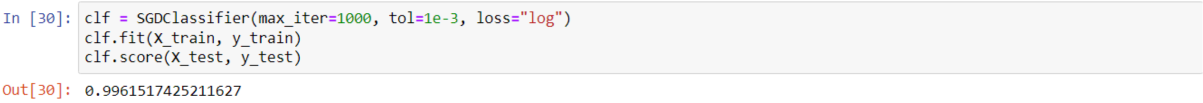
```
In [10]: # Using Pearson Correlation
         plt. figure(figsize=(10, 7) )

         cor = X_train.corr()
         sns.heatmap(cor, annot=True)
         plt.show()
```

Drop the correlation features and non-determining features from the data frame.



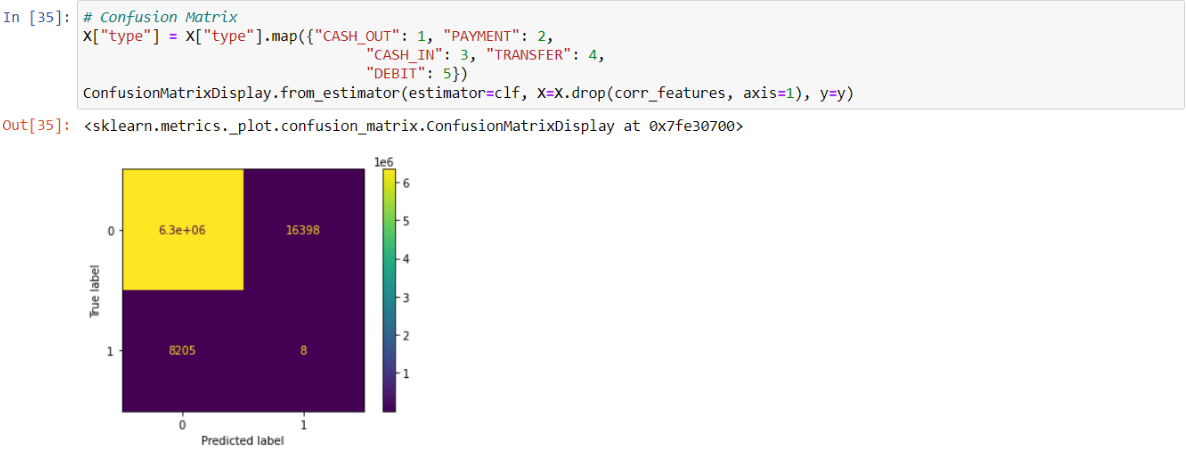
```
In [12]: corr_features = correlation(X_train, 0.8)
         remove_features = {"nameOrig", "nameDest", "isFlaggedFraud"}
         corr_features.update(remove_features)
         print(corr_features)

         {'newbalanceDest', 'newbalanceOrig', 'nameDest', 'nameOrig', 'isFlaggedFraud'}

In [13]: X_train.drop(corr_features, axis=1, inplace=True)
         X_test.drop(corr_features, axis=1, inplace=True)
```

Train your ML Model (SGD Classifier)



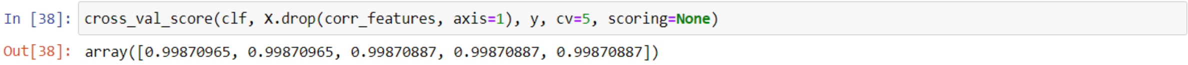
```
In [30]: clf = SGDClassifier(max_iter=1000, tol=1e-3, loss="log")
         clf.fit(X_train, y_train)
         clf.score(X_test, y_test)

Out[30]: 0.9961517425211627
```

Use Evaluation metrices to Evaluate the model performance.



```
In [35]: # Confusion Matrix
         X["type"] = X["type"].map({"CASH_OUT": 1, "PAYMENT": 2,
                                     "CASH_IN": 3, "TRANSFER": 4,
                                     "DEBIT": 5})
         ConfusionMatrixDisplay.from_estimator(estimator=clf, X=X.drop(corr_features, axis=1), y=y)

Out[35]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fe30700>
```

Use Cross Validation to test your model with different data division of train and test data.



```
In [38]: cross_val_score(clf, X.drop(corr_features, axis=1), y, cv=5, scoring=None)

Out[38]: array([0.99870965, 0.99870965, 0.99870887, 0.99870887, 0.99870887])
```

Here we can see our model is performing pretty well, the reason behind this is because of the size of the dataset i.e., the dataset has 63,62,620 records which is a pretty huge number. Due to large dataset the model is able to find pattern among the data and can predict the output pretty well.

GitHub - Code

15.Conclusion

The ML model detects potentially fraudulent activity and flags that activity for review. Accurately detecting fraudulent and non-fraudulent transactions given various users transaction data can be expected by the model which we have built. Hoping in the future that the rate of fraud transaction will go down and maintain some privacy among the people.

References

- Fraud Detection
- Scikit-Learn
- Kaggle.com