PREVENTING ONLINE PAYMENT FRAUD: A MACHINE LEARNING APPROACH FOR PREDICTIVE MODELING AND RISK ASSESSMENT

Venkata Ravi Kiran Kolla
Sr. Research Scientist and Software Engineer
venkat0634@gmail.com

ABSTRACT

The size of online transactions is always growing due to the Internet's quick growth of technology. The problem of associated network transaction fraud has also gotten worse at the same time. The properties of the network transaction—low cost, extensive reach, and high frequency—make it more difficult to identify fraud. The practice of making payments online is becoming increasingly popular as we move closer to modernity. Online payments are particularly advantageous for the buyer since they save time and address the issue of free money. We also don't need to bring any cash with us. But as we all know, good things often come with negative things.

Fraud can occur with any payment app when using the online payment method. Online Payment Fraud Detection is crucial as a result.

INTRODUCTION

Mobile payments are becoming among the most used payment options. The internet trading platform regularly sees thousands of transactions. Some criminals have the chance to commit crimes thanks to the prevalence of network transactions. The risk of theft of personal property exists in the complex network environment, endangering not only consumer interests but also negatively affecting the healthy growth of the network economy. Consequently, one of the important techniques for addressing the issue of network transaction fraud is transaction fraud detection.

The majority of statistical and multidimensional analysis approaches are used in traditional fraud detection.

It is challenging to discover the laws concealed behind the transaction records because these approaches are verification ones. Big data technology and machine learning algorithms offer effective approaches for detecting transaction fraud. Machine learning is able to express significant aspects across a big amount of data in a way that traditional statistical methods cannot. By employing the appropriate machine learning technique, we may create a model based on the transaction data already available to realise the identification of network transaction fraud and so lessen the damage brought on by fraud.

As a component of overall fraud prevention, payment fraud detection automates and assists in reducing the manual components of a screening/checking process. Because it is impossible to know with certainty the validity of an intention behind an application or transaction, it is a difficult problem.

Methodology

Below are all the columns from the dataset I am using:

step: represents a unit of time where 1 step equals 1 hour type: type of online transaction amount: the amount of the transaction nameOrig: customer starting the transaction oldbalanceOrg: balance before the transaction newbalanceOrig: balance after the transaction nameDest: recipient of the transaction oldbalanceDest: initial

balance of recipient before the transaction newbalanceDest: the new balance of recipient after the transaction isFraud: fraud transaction

The objective of the fraud detection methodology is to develop classification models that will help identify fraud in electronic transactions.

Finding the data that should be properly taken into account is part of the data selection process.

The method used for the prediction model is **Decision Tree Algorithm**.

The most effective and well-liked technique for categorization and prediction is the decision tree. A decision tree is a tree structure that resembles a flowchart, in which each leaf node (terminal node) bears a class label, each internal node implies a test on an attribute, and each branch shows the test's result.

CONSTRUCTION OF DECISION TREE:

By dividing the source set into subgroups based on an attribute value test, a tree can be "trained". It is known as recursive partitioning to repeat this operation on each derived subset. When the split no longer improves the predictions or when the subset at a node has the same value for the target variable, the recursion is finished. Decision tree classifier building is ideal for exploratory knowledge discovery because it doesn't require parameter configuration or domain understanding. High-dimensional data can be handled via decision trees. Decision tree classifiers are often accurate. A popular inductive method for learning classification information is decision tree induction.

DECISION TREE REPRESENTATION:

Decision trees categorise instances by arranging them in a tree from the root to a leaf node, which gives the instance's categorization. As seen in the above diagram, to classify an instance, one tests the attribute given by the root node of the tree before continuing down the branch of the tree that corresponds to the attribute's value. The subtree rooted at the new node is then subjected to the same procedure once more.

GINI INDEX:

The Gini Index is a number that measures how accurately a split is between the groups that are categorised. A score between 0 and 1, where 1 represents a randomly distributed distribution of the elements within classes, is evaluated using the Gini index. In this situation, we wish to have a low Gini index score. The assessment statistic we'll use to assess our decision tree model is the Gini Index.

STRENGTHS OF DECISION TREE METHOD:

- Decision trees are capable of producing clear rules.
- Without requiring a lot of computing, decision trees conduct classification.
- Both continuous and categorical variables are capable of being handled by decision trees.
- Which fields are crucial for classification or prediction can be clearly shown in decision trees.

Weakness of Decision Tree Method:

- For estimating situations when the objective is to forecast the value of a continuous characteristic, decision trees are less suitable.
- In classification problems with multiple classes and a limited number of training samples, decision trees are prone to errors.
- The training of decision trees can be computationally expensive. A decision tree's growth requires extensive computing work. Each candidate splitting field at each node must first be sorted in order to determine which split is optimal. Some algorithms employ combinations of fields, hence it is necessary to look for the best combining weights. Due to the need to create and evaluate numerous candidate sub-trees, pruning algorithms can also be costly.

Predictive and descriptive techniques are particularly intriguing in the context of fraud since they make it evident what criteria were used to produce their predictions, which is a crucial tool for comprehending fraud-related trends.

Appendices

```
In [8]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline
```

```
In [21]: data = pd.read_excel("onlinefraud.xlsx")
         print(data.head())
```

```
   step      type    amount     nameOrig  oldbalanceOrg  newbalanceOrig  \
0     1   PAYMENT   9839.64  C1231006815       170136.0       160296.36
1     1   PAYMENT   1864.28  C1666544295        21249.0        19384.72
2     1  TRANSFER    181.00  C1305486145          181.0            0.00
3     1  CASH_OUT    181.00   C840083671          181.0            0.00
4     1   PAYMENT  11668.14  C2048537720        41554.0        29885.86

     nameDest  oldbalanceDest  newbalanceDest  isFraud  isFlaggedFraud
0  M1979787155             0.0             0.0        0               0
1  M2044282225             0.0             0.0        0               0
2   C553264065             0.0             0.0        1               0
3    C38997010         21182.0             0.0        1               0
4  M1230701703             0.0             0.0        0               0
```

```
In [22]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 11 columns):
 #   Column          Non-Null Count    Dtype
---  ------          --------------    -----
 0   step            1048575 non-null  int64
 1   type            1048575 non-null  object
 2   amount          1048575 non-null  float64
 3   nameOrig        1048575 non-null  object
 4   oldbalanceOrg   1048575 non-null  float64
 5   newbalanceOrig  1048575 non-null  float64
 6   nameDest        1048575 non-null  object
 7   oldbalanceDest  1048575 non-null  float64
 8   newbalanceDest  1048575 non-null  float64
 9   isFraud         1048575 non-null  int64
 10  isFlaggedFraud  1048575 non-null  int64
dtypes: float64(5), int64(3), object(3)
memory usage: 88.0+ MB
```

```
In [23]: data.describe()
```

Out[23]:

	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
count	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1.048575e+06	1048575.0
mean	2.696617e+01	1.586670e+05	8.740055e+05	8.838049e+05	9.781600e+05	1.114193e+06	1.069067e-03	0.0
std	1.562325e+01	2.649408e+05	2.971725e+06	3.008246e+06	2.296779e+06	2.416554e+06	3.268351e-02	0.0
min	1.000000e+00	1.000000e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.0
25%	1.500000e+01	1.214907e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.0
50%	2.000000e+01	7.634333e+04	1.600200e+04	0.000000e+00	1.263772e+05	2.152604e+05	0.000000e+00	0.0
75%	3.900000e+01	2.137619e+05	1.366420e+05	1.746000e+05	9.158235e+05	1.149608e+06	0.000000e+00	0.0
max	9.500000e+01	1.000000e+07	3.892642e+07	3.894823e+07	4.205466e+07	4.216816e+07	1.000000e+00	0.0

In [24]:
```
obj = (data.dtypes == 'object')
object_cols = list(obj[obj].index)
print("Categorical variables:", len(object_cols))

int_ = (data.dtypes == 'int')
num_cols = list(int_[int_].index)
print("Integer variables:", len(num_cols))

fl = (data.dtypes == 'float')
fl_cols = list(fl[fl].index)
print("Float variables:", len(fl_cols))
```

```
Categorical variables: 3
Integer variables: 0
Float variables: 5
```

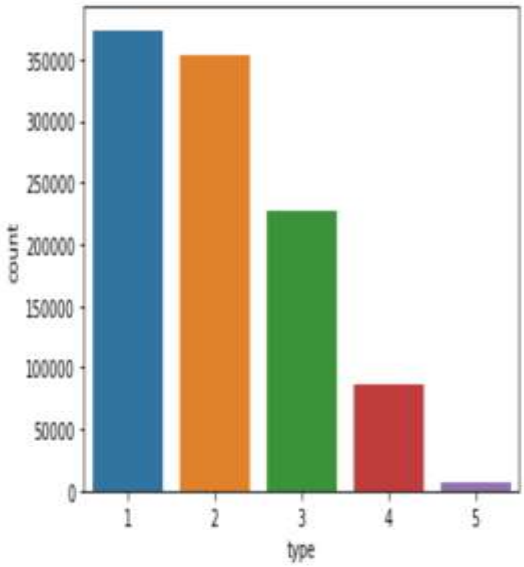
In [40]:
```
data["type"] = data["type"].map({"CASH_OUT": 1, "PAYMENT": 2,
                                 "CASH_IN": 3, "TRANSFER": 4,
                                 "DEBIT": 5})
data["isFraud"] = data["isFraud"].map({0: "No Fraud", 1: "Fraud"})
print(data.head())
```

```
   step  type    amount     nameOrig  oldbalanceOrg  newbalanceOrig  \
0     1     2   9839.64  C1231006815       170136.0       160296.36
1     1     2   1864.28  C1666544295        21249.0        19384.72
2     1     4    181.00  C1305486145          181.0            0.00
3     1     1    181.00   C840083671          181.0            0.00
4     1     2  11668.14  C2048537720        41554.0        29885.86

     nameDest  oldbalanceDest  newbalanceDest   isFraud  isFlaggedFraud
0  M1979787155             0.0             0.0  No Fraud               0
1  M2044282225             0.0             0.0  No Fraud               0
2   C553264065             0.0             0.0     Fraud               0
3    C38997010         21182.0             0.0     Fraud               0
4  M1230701703             0.0             0.0  No Fraud               0
```

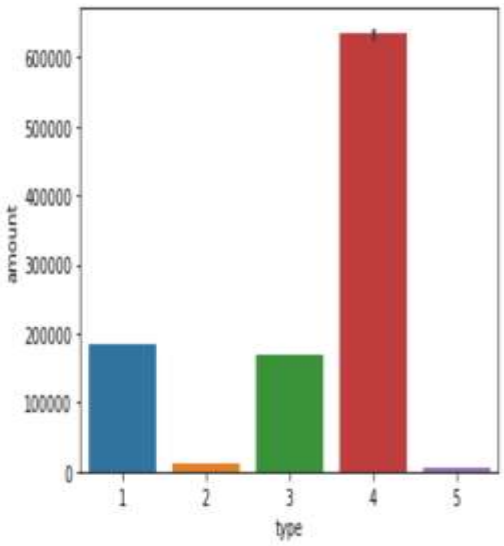
```
In [43]: sns.countplot(x='type', data=data)
```

```
Out[43]: <AxesSubplot:xlabel='type', ylabel='count'>
```




```
In [44]: sns.barplot(x='type', y='amount', data=data)
```

```
Out[44]: <AxesSubplot:xlabel='type', ylabel='amount'>
```

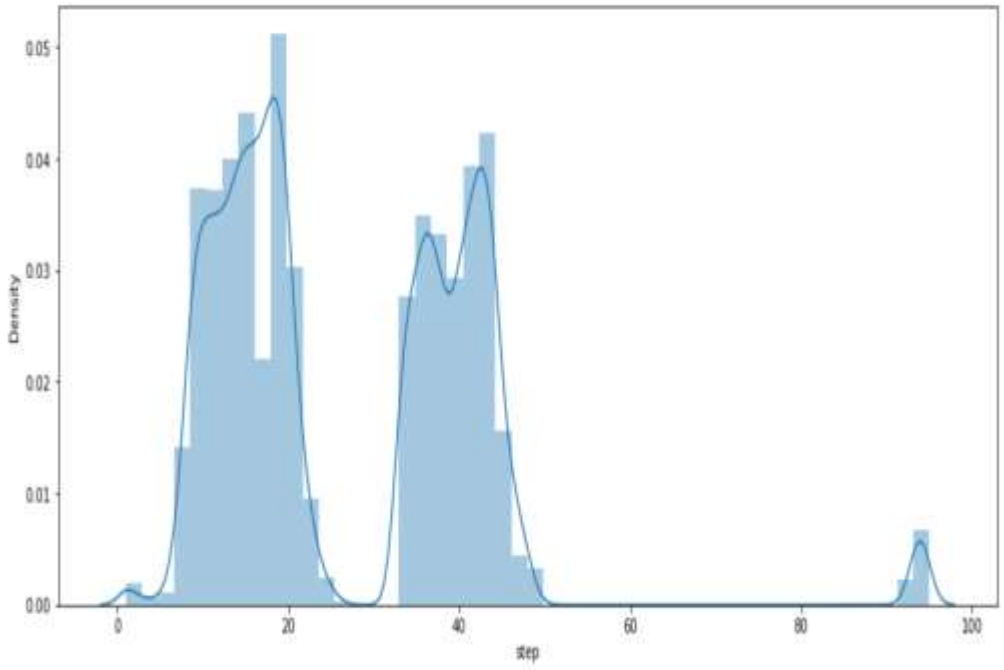


```
In [45]: data['isFraud'].value_counts()

Out[45]: No Fraud    1047433
         Fraud          1142
         Name: isFraud, dtype: int64
```

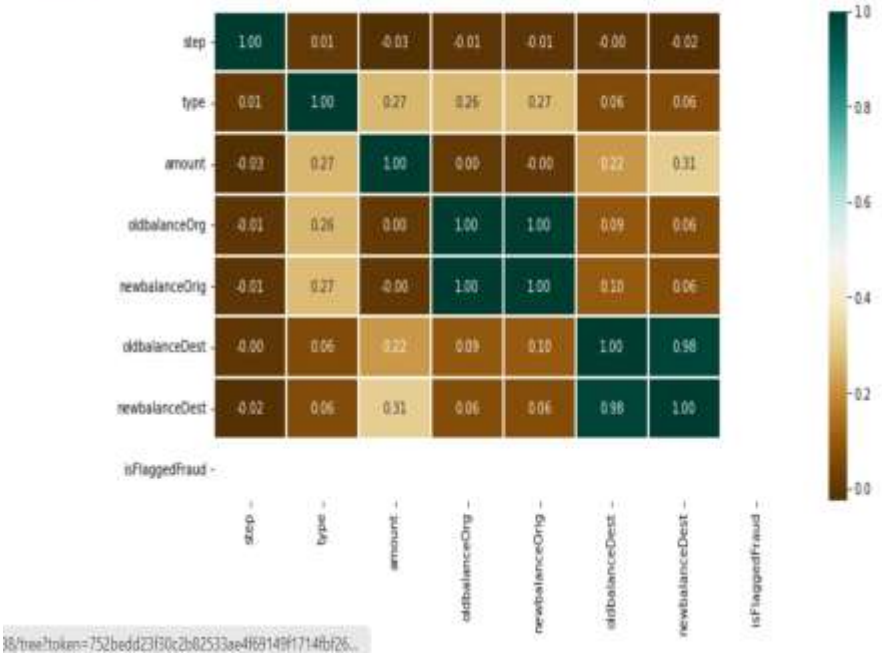
```
In [46]: plt.figure(figsize=(15, 6))
         sns.distplot(data['step'], bins=50)
```

```
Out[46]: <AxesSubplot:xlabel='step', ylabel='Density'>
```




```
In [48]: plt.figure(figsize=(12, 6))
         sns.heatmap(data.corr(),
                     cmap='BrBG',
                     fmt='.2f',
                     linewidths=2,
                     annot=True)
```

Out[48]: <AxesSubplot:>



38/tree?token=752bedd23f30c2b82533ae4f69149f1714fbf26...

In [49]:
```
type_new = pd.get_dummies(data['type'], drop_first=True)
data_new = pd.concat([data, type_new], axis=1)
data_new.head()
```

Out[49]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud	2	3	4	5
0	1	2	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	No Fraud	0	1	0	0	0
1	1	2	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	No Fraud	0	1	0	0	0
2	1	4	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	Fraud	0	0	0	1	0
3	1	1	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	Fraud	0	0	0	0	0
4	1	2	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	No Fraud	0	1	0	0	0

In [50]:
```
X = data_new.drop(['isFraud', 'type', 'nameOrig', 'nameDest'], axis=1)
y = data_new['isFraud']
```

In [51]:
```
X.shape, y.shape
```

Out[51]: ((1048575, 11), (1048575,))

```
In [52]: from sklearn.model_selection import train_test_split
         x = np.array(data[["type", "amount", "oldbalanceOrg", "newbalanceOrig"]])
         y = np.array(data[["isFraud"]])

In [53]: from sklearn.tree import DecisionTreeClassifier
         xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.10, random_state=42)
         model = DecisionTreeClassifier()
         model.fit(xtrain,ytrain)
         print(model.score(xtest,ytest))

         0.9994087241793664

In [54]: features = np.array([[4, 9000.60, 9000.60, 0.0]])
         print(model.predict(features))

         ['Fraud']
```

CONCLUSION

As fraud is part of operational risk, the results have far-reaching implications for regularity issues. A complicated problem like online payment fraud detection requires a comprehensive understanding of the problem. A prerequisite for this is access to a complete dataset. For the evaluation of our method, we used a real dataset obtained from a private bank. Regardless of the algorithm chosen, feature extraction is an essential part of developing an effective fraud detection method. The known fraud cases are only used in the ensemble aggregation step when the base learning programs are combined to form the final predictive function. Our framework opens exciting directions for future research.

REFERENCES

[1] Online Payment Fraud Detection using Machine Learning in Python - GeeksforGeeks

[2] T. P. Bhatla, V. Prabhu, and A. Dua. Understanding Credit Card Frauds

[3] E. Kurshan, H. Shen: Graph Computing for Financial Crime and Fraud Detection: Trends, Challenges and Outlook.

[4] Dheepa V, Dhanapal R: Analysis of credit card fraud detection methods. International journal of recent trends in engineering, 2009

[5] Fan, W., Miller, M., Stolfo, S., Lee, W. & Chan, P. (2001). Using Artificial Anomalies to Detect Unknown and Known Network

Intrusions. Proc. of ICDM01 , 123-248.

[6] Fanning, K., Cogger, K. & Srivastava, R. (1995). Detection of Management Fraud: A Neural Network Approach. Journal of Intelligent Systems in Accounting, Finance and Management 4: 113-126.

[7] Fawcett, T. (2003). "In Vivo" Spam Filtering: A Challenge Problem for KDD. SIGKDD Explorations 5 (2): 140-148.

[8] Fawcett, T. (1997). AI Approaches to Fraud Detection and Risk Management: Papers from the 1997 AAAI Workshop. Technical Report WS-97-07. AAAI Press.

[9] Fawcett, T. & Provost, F. (1999). Activity monitoring: Noticing Interesting Changes in Behavior. Proc. of SIGKDD99 , 53-62.

[10] Fawcett, T. & Provost, F. (1997). Adaptive Fraud Detection. Data Mining and Knowledge Discovery 1 (3): 291-316.

[11] Foster, D. & Stine, R. (2004). Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy.Journal of American Statistical Association 99 : 303-313.

[12] Ghosh, S. & Reilly, D. (1994). Credit Card Fraud Detection with a Neural Network. Proc. of 27th Hawaii International Conference on Systems Science 3 : 621-630.

[13] Goldberg, H., Kirkland, J., Lee, D., Shyr, P. & Thakker, D. (2003). The NASD Securities Observation, News Analysis \& Regulation System (SONAR). Proc. of IAAI03.

[14] Goldenberg, A., Shmueli, G., Caruana, R. & Fienberg, S. (2002). Early Statistical Detection of Anthrax Outbreaks by Tracking Over-the-Counter Medication Sales. Proc. of the National Academy of Sciences , 5237-5249.

[15] Green, B. & Choi, J. (1997). Assessing the Risk of Management Fraud through Neural Network Technology. Auditing 16 (1): 14-28.

[16] Hawkins, S., He, H., Williams, G. & Baxter, R. (2002). Outlier Detection Using Replicator Neural Networks. Proc. of DaWaK2002 170-180.

[17] He, H., Graco, W. & Yao, X. (1999). Application of Genetic Algorithms and k -Nearest Neighbour Method in Medical Fraud Detection. Proc. of SEAL1998 , 74-81.

[18] He, H., Wang, J., Graco, W. & Hawkins, S. (1997). Application of Neural Networks to Detection of Medical Fraud. Expert Systems with Applications 13 : 329-336.

[19] Heino, J. & Toivonen, H. (2003). Automated Detection of Epidemics from Usage Logs of a Physicians' Reference Database. Proc. of PKDD2003 , 180-191.

[20] Hodge, V. & Austin, J. (2004). A Survey of Outlier Detection Methodologies. Artificial Intelligence Review 22 : 85-126.

[21] Hollmen, J. & Tresp, V. (1998). Call-based Fraud detection in Mobile Communication Networks using a Hierarchical Regime- Switching Model. Proc. of Advances in Neural Information Processing Systems .

[22] Hutwagner, L., Thompson, W. & Seeman, M. (2003). The Bioterrorism Preparedness and Response Early Abberation Reporting System (EARS).

[23] Journal of Urban Health: Bulletin of the New York Academy of Medicine 80 (2): 89-96.

[24] Jensen, D., Rattigan,M.Blau, H. (2003). Information Awareness: A Prospective Technical Assessment. Proc. of SIGKDD03 , 378-387.

[25] Julisch, K. & Dacier, M. (2002). Mining Intrusion Detection Alarms for Actionable Knowledge. Proc. of SIGKDD02 , 366-375.

[26] Kim, H., Pang, S., Je, H., Kim, D. & Bang, S. (2003). Constructing Support Vector Machine Ensemble. Pattern Recognition 36 : 2757-2767.