Detecção de fraude de pagamentos online com aprendizado de máquina

Autor: Ernesto Villafuerte Oyola

A introdução de sistemas de pagamento online ajudou muito na facilidade de pagamentos. Mas, ao mesmo tempo, aumentou em fraudes de pagamento. Fraudes de pagamento online podem acontecer com qualquer pessoa que use qualquer sistema de pagamento, especialmente ao fazer pagamentos com cartão de crédito. É por isso que a detecção de fraudes de pagamento online é muito importante para as empresas de cartão de crédito.

No aprendizado de máquina (machine learning), que vamos utilizar aqui para realizar a detecção de fraudes, temos três subcategorias importantes: O supervisionado, não supervisionado e aprendizado por reforço. O aprendizado de máquina supervisionado consiste no uso de **conjuntos de dados rotulados** para treinar algoritmos, permitindo que eles classifiquem dados ou façam previsões com precisão.

Tipos de Problemas:

O aprendizado supervisionado aborda dois tipos principais de problemas:

- **Classificação**: Atribui dados de teste a categorias específicas (por exemplo, classificar spam em uma caixa de entrada).
- **Regressão**: Compreende a relação entre variáveis dependentes e independentes (por exemplo, prever receita de vendas).

O problema de detecção de fraude é tipicamente um problema de classificação e um algoritmo para resolução deste tipo de problema que iremos utilizar aqui é o de “árvores de decisão” (“decision tree”) embora posteriormente possamos usar um outro algoritmo também de aprendizado de máquina supervisionado para comparar a performance, caso achemos que seja necessário.

Detecção de fraude de pagamentos online com aprendizado de máquina

Para identificar fraudes de pagamento online com aprendizado de máquina, precisamos treinar um modelo de aprendizado de máquina para classificar pagamentos fraudulentos e não fraudulentos. Para isso, precisamos de um conjunto de dados contendo informações sobre fraudes de pagamentos online, para que possamos entender que tipo de transações levam à fraude. Para esta tarefa, coletei um **conjunto de dados** do Kaggle, que contém informações históricas sobre transações fraudulentas que podem ser usadas para detectar fraudes em pagamentos online. Abaixo estão todas as colunas do conjunto de dados que estou usando aqui:

1. step: representa uma unidade de tempo em que 1 passo (step) é igual a 1 hora
2. type: Tipo de transação online
3. value: o valor da transação
4. nameOrig: cliente que inicia a transação
5. oldbalanceOrg: saldo antes da transação
6. newbalanceOrig: saldo após a transação
7. nameDest: destinatário da transação
8. oldbalanceDest: saldo inicial do destinatário antes da transação
9. newbalanceDest: o novo saldo do destinatário após a transação
10. isFraud: transação fraudulenta

Detecção de fraudes em pagamentos online usando Python

Importando as bibliotecas Python necessárias e o conjunto de dados necessários para esta tarefa:

```
import pandas as pd
import numpy as np
data = pd.read_csv("credit card.csv")
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
```

Agora, vamos dar uma olhada se esse conjunto de dados tem algum valor nulo ou não:

```
print(data.isnull().sum())
```

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

Portanto, esse conjunto de dados não tem nenhum valor nulo. Antes de prosseguir, agora, vamos dar uma olhada no tipo de transação mencionada no conjunto de dados:

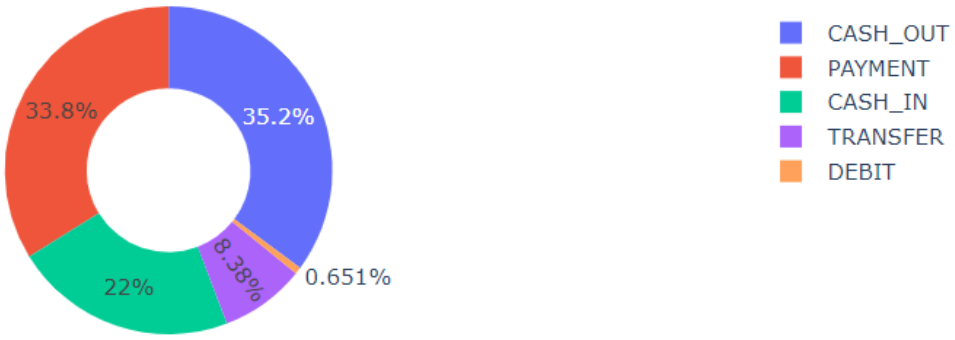
```
# Explorando o tipo de transação
print(data.type.value_counts())
```

```
type
CASH_OUT    2237500
PAYMENT     2151495
CASH_IN     1399284
TRANSFER     532909
DEBIT         41432
Name: count, dtype: int64
```

```
type = data["type"].value_counts()
transactions = type.index
quantity = type.values

import plotly.express as px
figure = px.pie(data,
        values=quantity,
        names=transactions,hole = 0.5,
        title="Distribuição do Tipo de Transação")
figure.show()
```

Distribuição do Tipo de Transação:



Excluindo colunas do tipo texto que não são relevantes

```
data = data.drop(columns=["nameOrig", "nameDest"])
data.head()
```

	step	type	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
0	1	2	9839.64	170136.0	160296.36	0.0	0.0	0	0
1	1	2	1864.28	21249.0	19384.72	0.0	0.0	0	0
2	1	4	181.00	181.0	0.00	0.0	0.0	1	0
3	1	1	181.00	181.0	0.00	21182.0	0.0	1	0
4	1	2	11668.14	41554.0	29885.86	0.0	0.0	0	0

Agora vamos transformar os recursos categóricos em numéricos.

```
data["type"] = data["type"].map({"CASH_OUT": 1, "PAYMENT": 2,
                                 "CASH_IN": 3, "TRANSFER": 4,
                                 "DEBIT": 5})
```

Agora vamos dar uma olhada na correlação entre os recursos dos dados com a coluna **isFraud**:

```
# Checking correlation
correlation = data.corr()
print(correlation["isFraud"].sort_values(ascending=False))
```

```
isFraud           1.000000
amount            0.076688
isFlaggedFraud    0.044109
step              0.031578
type              0.016171
oldbalanceOrg     0.010154
newbalanceDest    0.000535
oldbalanceDest   -0.005885
newbalanceOrig   -0.008148
Name: isFraud, dtype: float64
```

Verificamos que a correlação maior com a coluna isFraud é das colunas amount e isFlaggedFraud.

Aqui, também transformarei os valores da coluna isFraud em rótulos No Fraud e Fraud para ter uma melhor compreensão da saída:

```
data["isFraud"] = data["isFraud"].map({0: "No Fraud", 1: "Fraud"})
print(data.head())
```

	step	type	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	\
0	1	2	9839.64	170136.0	160296.36	0.0	
1	1	2	1864.28	21249.0	19384.72	0.0	
2	1	4	181.00	181.0	0.00	0.0	
3	1	1	181.00	181.0	0.00	21182.0	
4	1	2	11668.14	41554.0	29885.86	0.0	

	newbalanceDest	isFraud	isFlaggedFraud
0	0.0	No Fraud	0
1	0.0	No Fraud	0
2	0.0	Fraud	0
3	0.0	Fraud	0
4	0.0	No Fraud	0

Modelo de detecção de fraude de pagamentos online

Agora vamos treinar um modelo de classificação para classificar transações fraudulentas e não fraudulentas. Antes de treinar o modelo, dividirei os dados em conjuntos de treinamento e teste:

Dividindo os dados em x e y

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

Agora vamos treinar o modelo de detecção de fraude de pagamentos online:

```
# Treinando o modelo de machine learning DecisionTree
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))
```

```
0.999746959585831
```

Podemos perceber que o score (performance) do modelo é praticamente de 100%, por isso não vamos tentar implementar outro modelo para comparação porque o nosso modelo é muito bom.

Agora vamos classificar se uma transação é uma fraude ou não, alimentando uma transação no modelo:

```
# prediction
#features = [type, amount, oldbalanceOrg, newbalanceOrig]
features = np.array([[4, 9000.60, 9000.60, 0.0]])
print(model.predict(features))
```

```
['Fraud']
```

Resumo

É assim que podemos detectar fraudes em pagamentos online com aprendizado de máquina usando Python. A detecção de fraudes de pagamento online é uma das aplicações da ciência de dados em **finanças**.

O notebook jupyter com o código em Python do projeto, 100% funcional e testado, se encontra disponível em https://github.com/evillafuerte/fraud_detection .

Autor: Ernesto Villafuerte Oyola
Analista de Dados
Mestrado em Computação (Inteligência Artificial) - COPPE Sistemas
Consultor e Professor

ernesto.villafuerte@aomega.com.br
https://www.aomega.com.br