Prevendo transaçẽs fraudulentas

O objetivo desse notebook é criar um modelo de regressão logistica que consiga classificar trasançẽs fraudulentas

Os dados usados tem como fonte https://www.kaggle.com/datasets/vardhansiramdasu/fraudulent-transactions-prediction?resource=download&select=Fraud.csv

In [1]:

```python
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
import pandas as pd
```

In [2]:

```python
data = pd.read_csv('Fraud.csv')
display(data.head())
data.shape
```

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

```
(6362620, 11)
```

In [3]:

```python
# Codificando variáveis categóricas
encoder = LabelEncoder()
for col in data.columns:
    if data[col].dtype == 'object':
        data[col] = encoder.fit_transform(data[col])

data.head()
```

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud
0	1	3	9839.64	757869	170136.0	160296.36	1662094	0.0	0.0	0
1	1	3	1864.28	2188998	21249.0	19384.72	1733924	0.0	0.0	0
2	1	4	181.00	1002156	181.0	0.00	439685	0.0	0.0	1
3	1	1	181.00	5828262	181.0	0.00	391696	21182.0	0.0	1
4	1	3	11668.14	3445981	41554.0	29885.86	828919	0.0	0.0	0

In [4]:

```python
# Separando as features e o alvo

target = data['isFraud']
features = data.drop(['isFraud'], axis=1)
```

In [5]:

```python
# Reescalando as features
scaler = MinMaxScaler()
scaled_features = scaler.fit_transform(features)

scaled_features[0:5]
```

```
array([[0.00000000e+00, 7.50000000e-01, 1.06437179e-04, 1.19287344e-01,
        2.85534757e-03, 3.23275647e-03, 6.10534018e-01, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 7.50000000e-01, 2.01662565e-05, 3.44544714e-01,
        3.56616357e-04, 3.90938877e-04, 6.36919204e-01, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 1.00000000e+00, 1.95790998e-06, 1.57737720e-01,
        3.03767521e-06, 0.00000000e+00, 1.61508705e-01, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 2.50000000e-01, 1.95790998e-06, 9.17358931e-01,
        3.03767521e-06, 0.00000000e+00, 1.43880992e-01, 5.94973445e-05,
        0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 7.50000000e-01, 1.26216397e-04, 5.42391788e-01,
        6.97389810e-04, 6.02719283e-04, 3.04485335e-01, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00]])
```

In [8]:

```python
# Dividindo os dados em treino e teste
features_train, features_test, target_train, target_test = train_test_split(scaled_features, target, test_size=0.2, random_state=42)
```

In [9]:

```python
# Criando o modelo
model = LogisticRegression()
model.fit(features_train, target_train);
```

In [10]:

```python
# Avaliando o modelo

def evaluate_logistic_regression(model, features, target):
    predictions = model.predict(features)
    accuracy = model.score(features, target)
    precision = precision_score(target, predictions)
    recall = recall_score(target, predictions)
    f1 = f1_score(target, predictions)
    conf_matrix = confusion_matrix(target, predictions)
    confusion_matrix_data_frame = pd.DataFrame(conf_matrix, index=['True', 'False'], columns=['Predicted True', 'Predicted False'])
    display(confusion_matrix_data_frame)
    print(f'Accuracy: {accuracy:.4}\nPrecision: {precision:.4}\nRecall: {recall:.4}\nF1:{f1:.4}')

evaluate_logistic_regression(model, features_test, target_test)
```

	Predicted True	Predicted False
True	1270903	1
False	1507	113

```
Accuracy: 0.9988
Precision: 0.9912
Recall: 0.06975
F1: 0.1303
```