```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
```

```
df=pd.read_csv("//content/drive/MyDrive/archive (15).zip")
```

```
df
```

	Index	Incoming Calls	Answered Calls	Answer Rate	Abandoned Calls	Answer Speed (AVG)	Talk Duration (AVG)	Waiting Time (AVG)	Service Level (20 Seconds)
0	1	217	204	94.01%	13	0:00:17	0:02:14	0:02:45	76.28%
1	2	200	182	91.00%	18	0:00:20	0:02:22	0:06:55	72.73%
2	3	216	198	91.67%	18	0:00:18	0:02:38	0:03:50	74.30%
3	4	155	145	93.55%	10	0:00:15	0:02:29	0:03:12	79.61%
4	5	37	37	100.00%	0	0:00:03	0:02:06	0:00:35	97.30%
...	...	...	...	...	...	...	...	...	...
1246	1247	191	184	96.34%	7	0:00:07	0:02:50	0:01:56	92.55%
1247	1248	212	209	98.58%	3	0:00:10	0:02:51	0:01:45	89.10%
1248	1249	210	203	96.67%	7	0:00:12	0:03:22	0:03:52	85.24%
1249	1250	167	159	95.21%	8	0:00:16	0:03:16	0:02:42	83.03%
1250	1251	12	11	91.67%	1	0:00:03	0:02:12	0:00:58	91.67%

1251 rows × 9 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1251 entries, 0 to 1250
Data columns (total 9 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Index                       1251 non-null   int64
 1   Incoming Calls              1251 non-null   int64
 2   Answered Calls              1251 non-null   int64
 3   Answer Rate                 1251 non-null   object
 4   Abandoned Calls             1251 non-null   int64
 5   Answer Speed (AVG)          1251 non-null   object
 6   Talk Duration (AVG)         1251 non-null   object
 7   Waiting Time (AVG)          1251 non-null   object
 8   Service Level (20 Seconds)  1251 non-null   object
dtypes: int64(4), object(5)
memory usage: 88.1+ KB
```

```
df.isnull().sum()
```

	0
Index	0
Incoming Calls	0
Answered Calls	0
Answer Rate	0
Abandoned Calls	0
Answer Speed (AVG)	0
Talk Duration (AVG)	0
Waiting Time (AVG)	0
Service Level (20 Seconds)	0

dtype: int64

```
df.dropna(inplace=True)
```

```
X = df[['Incoming Calls', 'Answered Calls']]
y = df['Answer Rate']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
y_pred = model.predict(X_test)
```

```
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

     100.00%       0.85      0.52      0.65        21
      51.72%       0.00      0.00      0.00         0
      53.46%       0.00      0.00      0.00         0
      56.29%       0.00      0.00      0.00         1
      56.95%       0.00      0.00      0.00         1
      67.49%       0.00      0.00      0.00         1
      72.97%       0.00      0.00      0.00         1
      73.77%       0.00      0.00      0.00         1
      74.08%       0.00      0.00      0.00         0
      74.71%       0.00      0.00      0.00         1
      75.00%       0.00      0.00      0.00         0
      77.08%       0.00      0.00      0.00         1
      77.84%       0.00      0.00      0.00         1
      77.92%       0.00      0.00      0.00         0
      78.95%       0.00      0.00      0.00         0
      79.07%       0.00      0.00      0.00         1
      79.31%       0.00      0.00      0.00         1
      79.60%       0.00      0.00      0.00         1
      79.92%       0.00      0.00      0.00         0
      80.00%       0.00      0.00      0.00         1
      80.17%       0.00      0.00      0.00         1
      80.55%       0.00      0.00      0.00         0
      81.03%       0.00      0.00      0.00         0
      81.05%       0.00      0.00      0.00         1
      81.14%       0.00      0.00      0.00         1
      81.25%       0.00      0.00      0.00         0
```

```
          81.44%       0.00      0.00      0.00         0
          81.99%       0.00      0.00      0.00         1
          82.41%       0.00      0.00      0.00         1
          82.61%       0.00      0.00      0.00         1
          83.42%       0.00      0.00      0.00         1
          84.00%       0.00      0.00      0.00         0
          84.02%       0.00      0.00      0.00         1
          84.08%       0.00      0.00      0.00         1
          84.46%       0.00      0.00      0.00         0
          84.62%       0.00      0.00      0.00         1
          84.73%       0.00      0.00      0.00         1
          84.75%       0.00      0.00      0.00         1
          84.77%       0.00      0.00      0.00         0
          84.83%       0.00      0.00      0.00         0
          84.91%       0.00      0.00      0.00         0
          85.05%       0.00      0.00      0.00         0
          85.10%       0.00      0.00      0.00         0
          85.24%       0.00      0.00      0.00         0
          85.65%       0.00      0.00      0.00         1
          85.78%       0.00      0.00      0.00         0
          85.79%       0.00      0.00      0.00         1
          85.85%       0.00      0.00      0.00         1
          85.99%       0.00      0.00      0.00         1
          86.07%       0.00      0.00      0.00         0
          86.14%       0.00      0.00      0.00         1
          86.28%       0.00      0.00      0.00         1
          86.34%       0.00      0.00      0.00         0
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
accuracy
```

```
0.18326693227091634
```

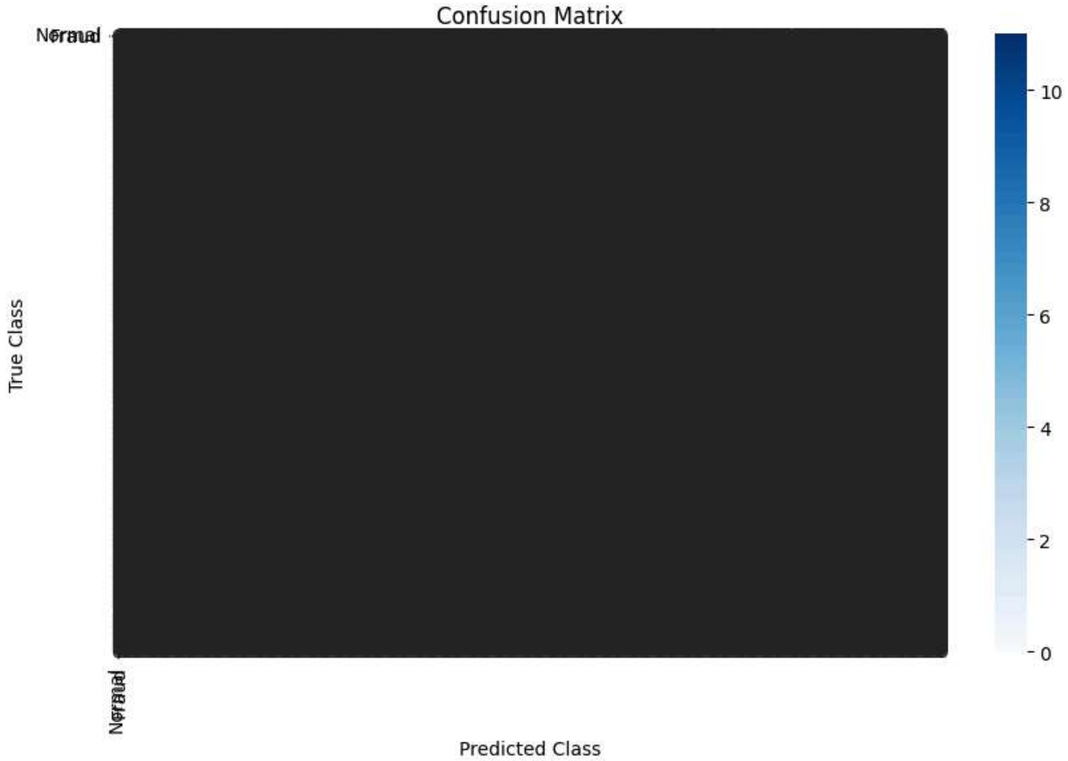
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix


cm = confusion_matrix(y_test, y_pred)
LABELS = ['Normal', 'Fraud']

print("Confusion Matrix:")
print(cm)

plt.figure(figsize=(10, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap='Blues', xticklabels=LABELS, yticklabels=LABELS)
plt.title("Confusion Matrix")
plt.ylabel("True Class")
plt.xlabel("Predicted Class")
plt.show()
```

```
Confusion Matrix:
[[11  0  0 ...  1  0  1]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]]
```



```
new_call = [[10, 5]]
new_call_scaled = scaler.transform(new_call)
prediction = model.predict(new_call_scaled)

if prediction[0] == 1:
    print("The call is predicted to be fraud.")
```