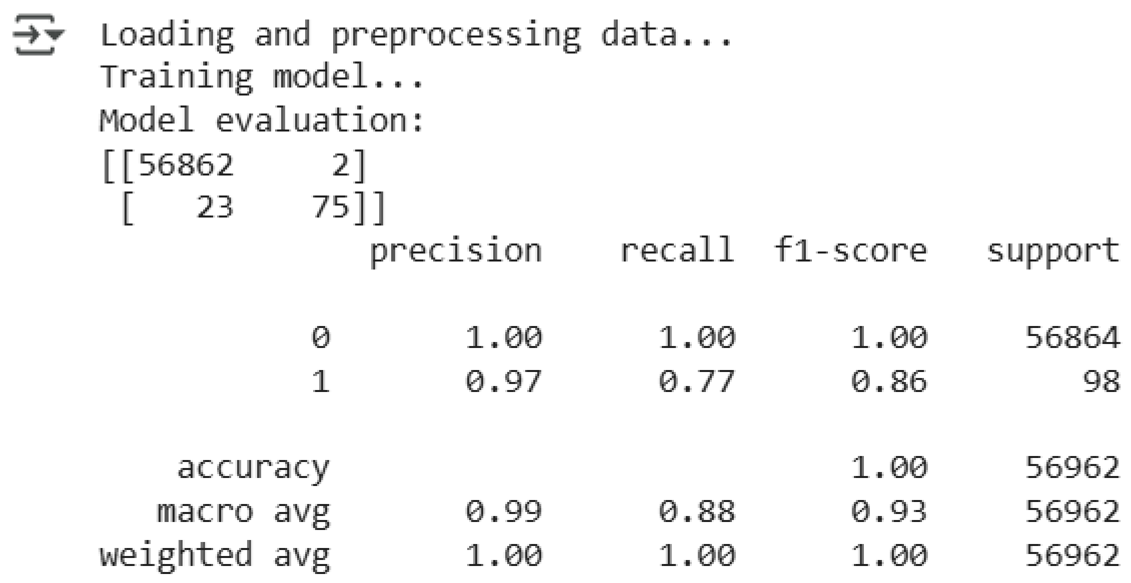
Guarding transactions with AI powered credit card fraud detection and prevention.

Source code

```
import pandas as pd import numpy as np from flask
import Flask, request, jsonify from sklearn.ensemble
import RandomForestClassifier from
sklearn.preprocessing import StandardScaler from
sklearn.model_selection import train_test_split from
sklearn.metrics import classification_report,
confusion_matrix import joblib import os # Load and
preprocess dataset print("Loading and preprocessing
data...") data = pd.read_csv("creditcard.csv") # Feature
engineering scaler = StandardScaler()
data['normalizedAmount'] =
scaler.fit_transform(data['Amount'].values.reshape(-1, 1))
data = data.drop(['Time', 'Amount'], axis=1) # Prepare
features and labels X = data.drop('Class', axis=1) y =
data['Class'] # Split into training/testing X_train, X_test,
y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42) # Train model print("Training model...")
model = RandomForestClassifier(n_estimators=100,
random_state=42) model.fit(X_train, y_train) # Save model
and scaler joblib.dump(model, 'fraud_model.pkl')
joblib.dump(scaler, 'scaler.pkl') # Evaluate print("Model
evaluation:") y_pred = model.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred)) # Set up Flask
API app = Flask(__name__) model =
joblib.load('fraud_model.pkl') scaler =
joblib.load('scaler.pkl') @app.route('/predict',
methods=['POST']) def predict(): try: data =
request.get_json() features = pd.DataFrame([data]) if
'Amount' in features: features['normalizedAmount'] =
scaler.transform([[features['Amount'][0]]]) features =
features.drop(['Amount'], axis=1) if 'Time' in features:
features = features.drop(['Time'], axis=1) prediction =
model.predict(features)[0] result = "Fraudulent" if
prediction == 1 else "Legitimate" return
jsonify({"prediction": result}) except Exception as e: return
jsonify({"error": str(e)}) if __name__ == 'main':
print("Starting API on http://localhost:5000 ...")
app.run(debug=True)
```

OUTPUT



```
Loading and preprocessing data...
Training model...
Model evaluation:
[[56862     2]
 [   23    75]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56864
           1       0.97      0.77      0.86        98

    accuracy                           1.00     56962
   macro avg       0.99      0.88      0.93     56962
weighted avg       1.00      1.00      1.00     56962
```