# credit\_card\_fraud\_detection.py

import pandas as pd

import numpy as np

from sklearn.ensemble import RandomForestClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification\_report, confusion\_matrix

# 1. Load Data

print("Loading dataset...")

data = pd.read\_csv('creditcard.csv') # Make sure this file is in the same directory

# 2. Preprocess Data

print("Preprocessing data...")

scaler = StandardScaler()

data['NormalizedAmount'] = scaler.fit\_transform(data[['Amount']])

data = data.drop(['Time', 'Amount'], axis=1)

# 3. Prepare Features and Target

X = data.drop('Class', axis=1)

y = data['Class']

# 4. Split into Train and Test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.3, random\_state=42, stratify=y

)

# 5. Train Random Forest Model

print("Training model...")

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

# 6. Evaluate Model

print("Evaluating model...")

y\_pred = model.predict(X\_test)

print("\nConfusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

# 7. Real-Time Transaction Prediction Function

def detect\_fraud(transaction\_features):

"""

transaction\_features: list or np.array of features (should match the number of columns in X)

"""

if isinstance(transaction\_features, list):

transaction\_features = np.array(transaction\_features)

transaction\_df = pd.DataFrame([transaction\_features], columns=X.columns)

prediction = model.predict(transaction\_df)

return "⚠️ Fraudulent Transaction" if prediction[0] == 1 else "✅ Legitimate Transaction"

# 8. Test With a Sample Transaction

print("\nTesting single transaction...")

sample\_transaction = X\_test.iloc[10] # Random test transaction

result = detect\_fraud(sample\_transaction)

print(f"Transaction Prediction: {result}")