AI-Powered Credit Card Fraud Detection and Prevention

This project demonstrates a basic machine learning pipeline to detect fraudulent transactions using the popular Kaggle Credit Card Fraud Detection dataset. We use a Random Forest Classifier and evaluate the model based on precision, recall, and accuracy due to class imbalance.

Source Code (Python)

```python
# Import Libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Load Dataset
df = pd.read_csv("creditcard.csv")  # Download from https://www.kaggle.com/mlg-ulb/creditcardfraud

# Explore Dataset
print(df.head())
print(df['Class'].value_counts())  # Check imbalance

# Visualize Class Distribution
sns.countplot(x='Class', data=df)
plt.title("Class Distribution (0 = Normal, 1 = Fraud)")
plt.show()

# Feature Scaling (important for ML models)
scaler = StandardScaler()
df['Amount'] = scaler.fit_transform(df[['Amount']])
df = df.drop(['Time'], axis=1)

# Split Data
X = df.drop('Class', axis=1)
y = df['Class']
```

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

# Train Model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predictions
y_pred = model.predict(X_test)

# Evaluation
print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Simulated Output

Class Distribution (imbalanced):

```
0   284315  # Non-fraudulent transactions
1      492  # Fraudulent transactions
```

Confusion Matrix:

```
[[56857    3]
 [   30   73]]
```

Accuracy Score:

0.9994

Classification Report:

```
           precision   recall  f1-score   support

        0      1.00      1.00      1.00     56860
        1      0.96      0.71      0.82       103

 accuracy                          1.00     56963
macro avg      0.98      0.86      0.91     56963
weighted avg   1.00      1.00      1.00     56963
```