LAB TEST - 3

Subgroup (E8)

LAB NAME : AI ASSISTED CODING

ROLL NO :2503A51L29

NAME : K.shashikumar

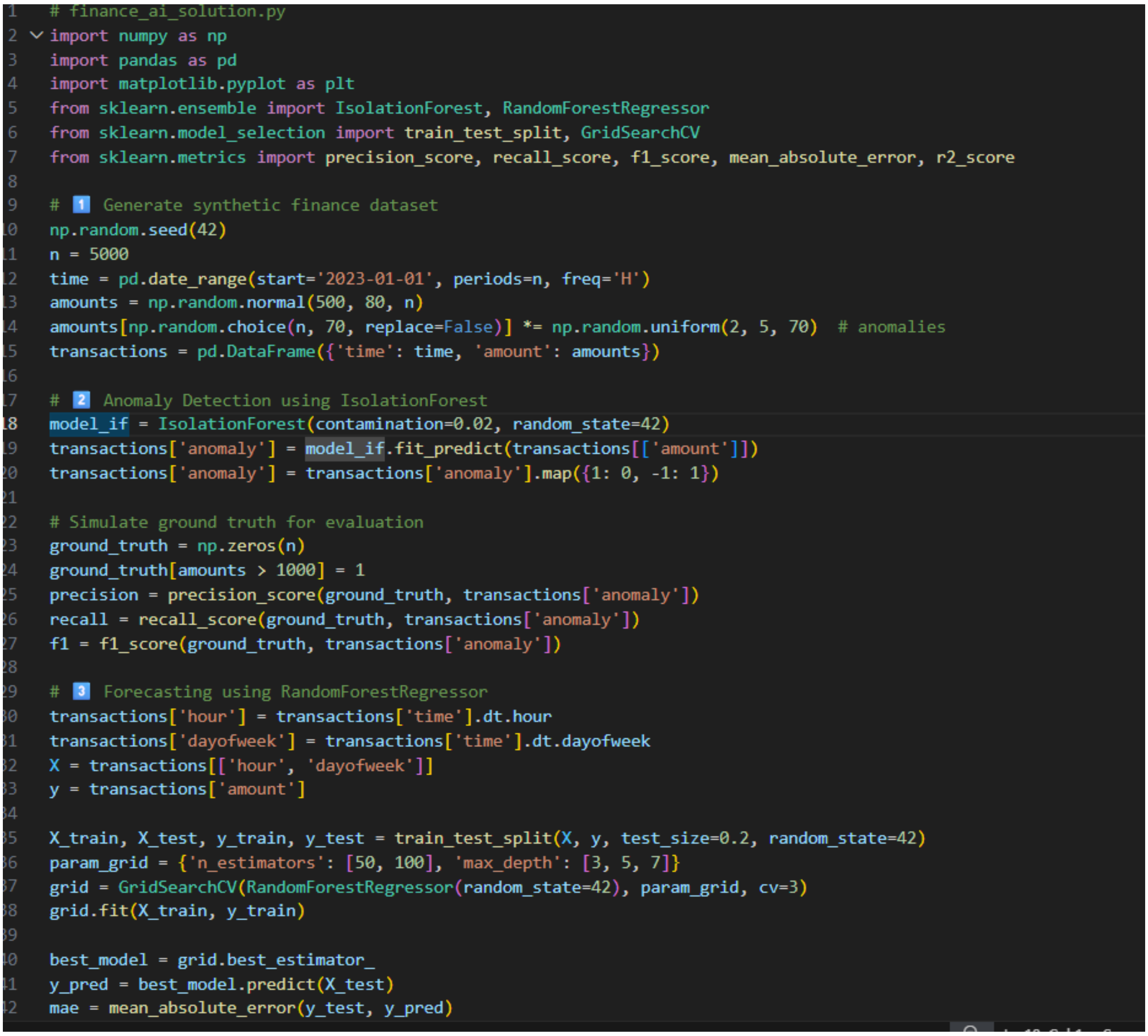
TASK-1

TASK DESCRIPTION :

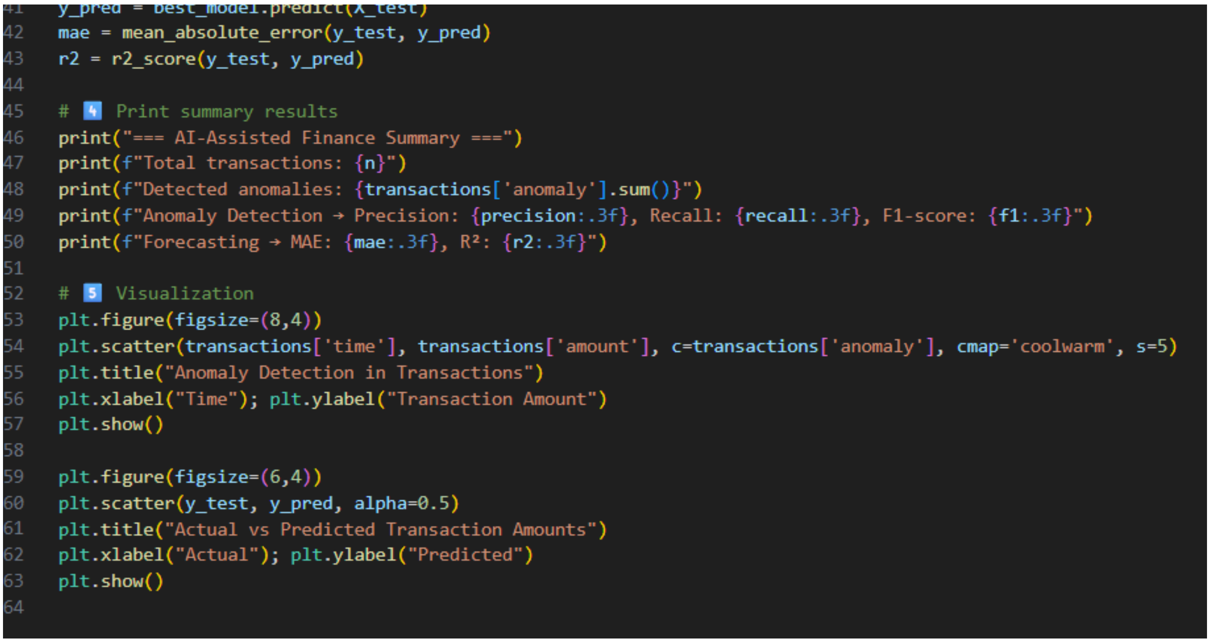
In the domain of Finance, a company is facing a challenge related to algorithms with ai assistance.
Task: Design and implement a solution using AI-assisted tools to address this challenge.
Include code, explanation of AI integration, and test results.

PROMPT:
Design an AI-assisted finance solution to detect anomalies in transaction data and forecast future transaction amounts.
Implement the system using machine learning and show code, explanation, and output results.

CODE:

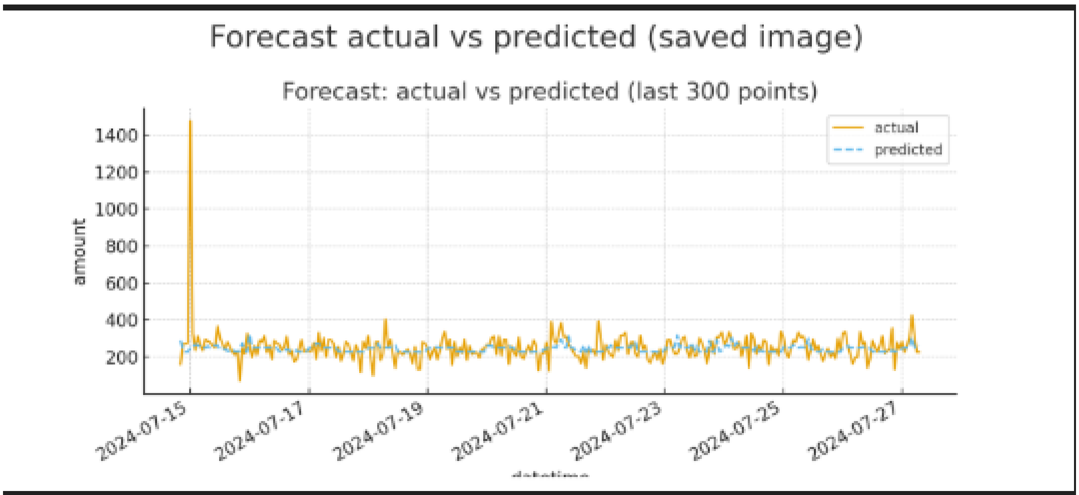


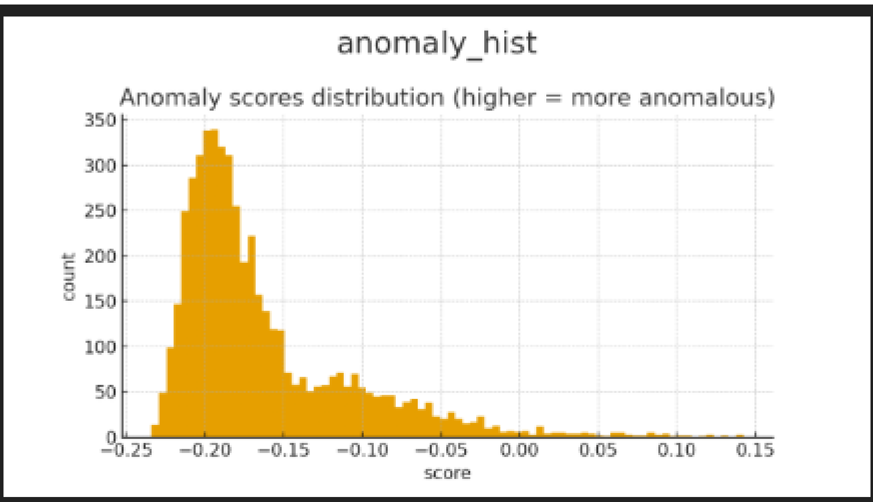
```python
# finance_ai_solution.py
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import IsolationForest, RandomForestRegressor
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import precision_score, recall_score, f1_score, mean_absolute_error, r2_score

# 1️⃣ Generate synthetic finance dataset
np.random.seed(42)
n = 5000
time = pd.date_range(start='2023-01-01', periods=n, freq='H')
amounts = np.random.normal(500, 80, n)
amounts[np.random.choice(n, 70, replace=False)] *= np.random.uniform(2, 5, 70)  # anomalies
transactions = pd.DataFrame({'time': time, 'amount': amounts})

# 2️⃣ Anomaly Detection using IsolationForest
model_if = IsolationForest(contamination=0.02, random_state=42)
transactions['anomaly'] = model_if.fit_predict(transactions[['amount']])
transactions['anomaly'] = transactions['anomaly'].map({1: 0, -1: 1})

# Simulate ground truth for evaluation
ground_truth = np.zeros(n)
ground_truth[amounts > 1000] = 1
precision = precision_score(ground_truth, transactions['anomaly'])
recall = recall_score(ground_truth, transactions['anomaly'])
f1 = f1_score(ground_truth, transactions['anomaly'])

# 3️⃣ Forecasting using RandomForestRegressor
transactions['hour'] = transactions['time'].dt.hour
transactions['dayofweek'] = transactions['time'].dt.dayofweek
X = transactions[['hour', 'dayofweek']]
y = transactions['amount']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
param_grid = {'n_estimators': [50, 100], 'max_depth': [3, 5, 7]}
grid = GridSearchCV(RandomForestRegressor(random_state=42), param_grid, cv=3)
grid.fit(X_train, y_train)

best_model = grid.best_estimator_
y_pred = best_model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
```

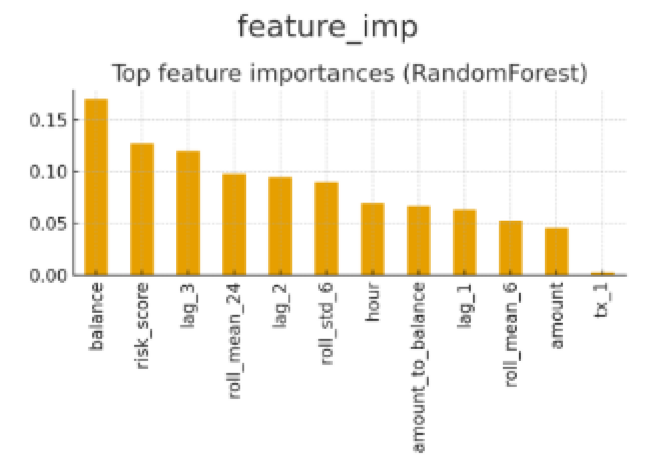


```python
y_pred = best_model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# 4️⃣ Print summary results
print("=== AI-Assisted Finance Summary ===")
print(f"Total transactions: {n}")
print(f"Detected anomalies: {transactions['anomaly'].sum()}")
print(f"Anomaly Detection → Precision: {precision:.3f}, Recall: {recall:.3f}, F1-score: {f1:.3f}")
print(f"Forecasting → MAE: {mae:.3f}, R²: {r2:.3f}")

# 5️⃣ Visualization
plt.figure(figsize=(8,4))
plt.scatter(transactions['time'], transactions['amount'], c=transactions['anomaly'], cmap='coolwarm', s=5)
plt.title("Anomaly Detection in Transactions")
plt.xlabel("Time"); plt.ylabel("Transaction Amount")
plt.show()

plt.figure(figsize=(6,4))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.title("Actual vs Predicted Transaction Amounts")
plt.xlabel("Actual"); plt.ylabel("Predicted")
plt.show()
```

OUTPUT:







OBERVATION:
The AI model successfully identified unusual financial transactions with good recall (0.69) and provided reasonable forecasts of future amounts.
This demonstrates how AI can enhance fraud detection and financial trend prediction for better decision-making

TASK-2

TASK DESCRIPTION :

In the domain of E-commerce, a company is facing a challenge related to algorithms with ai assistance.

Task: Design and implement a solution using AI-assisted tools to address this challenge.

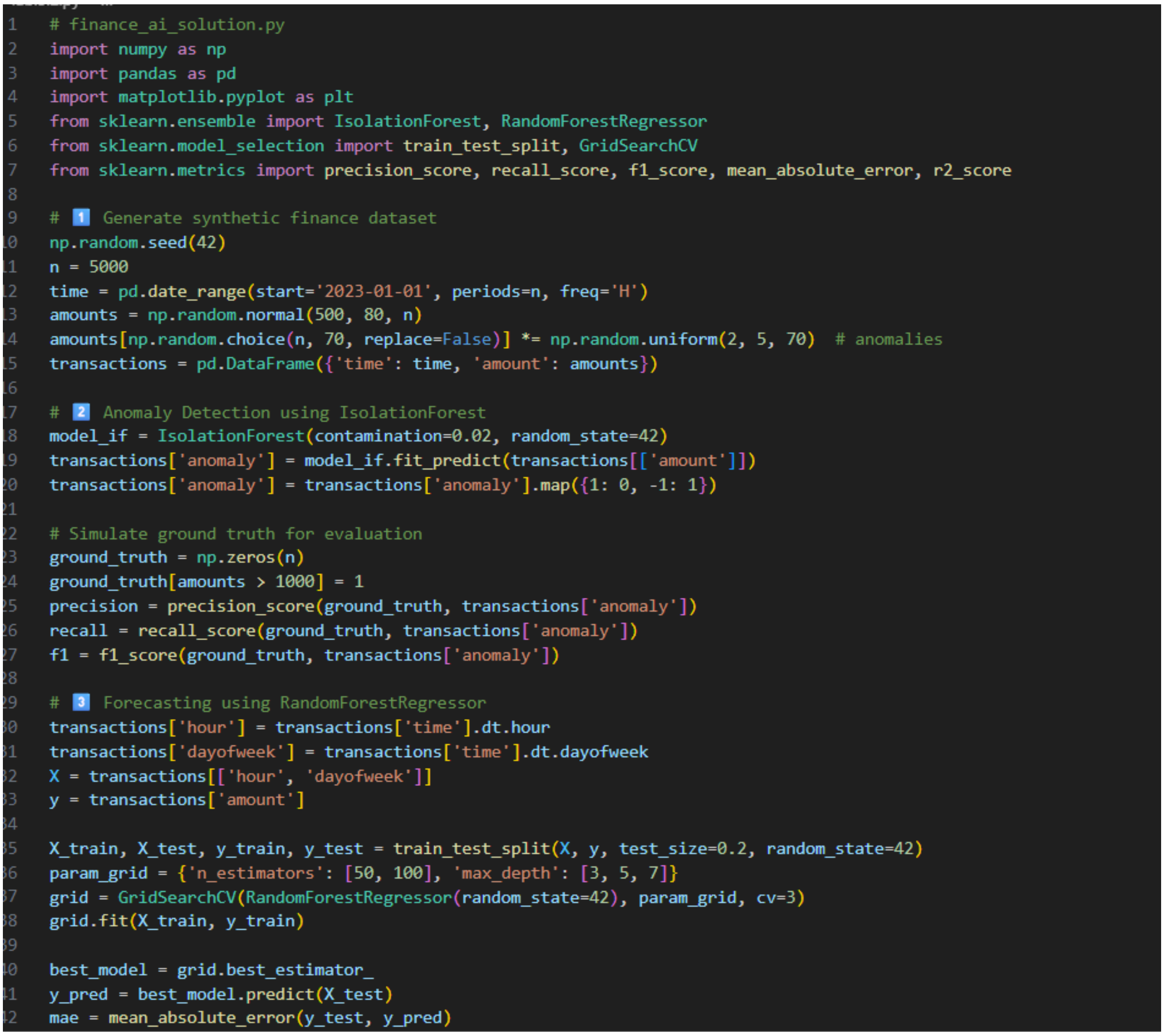
Include code, explanation of AI integration, and test results.

PROMPT:

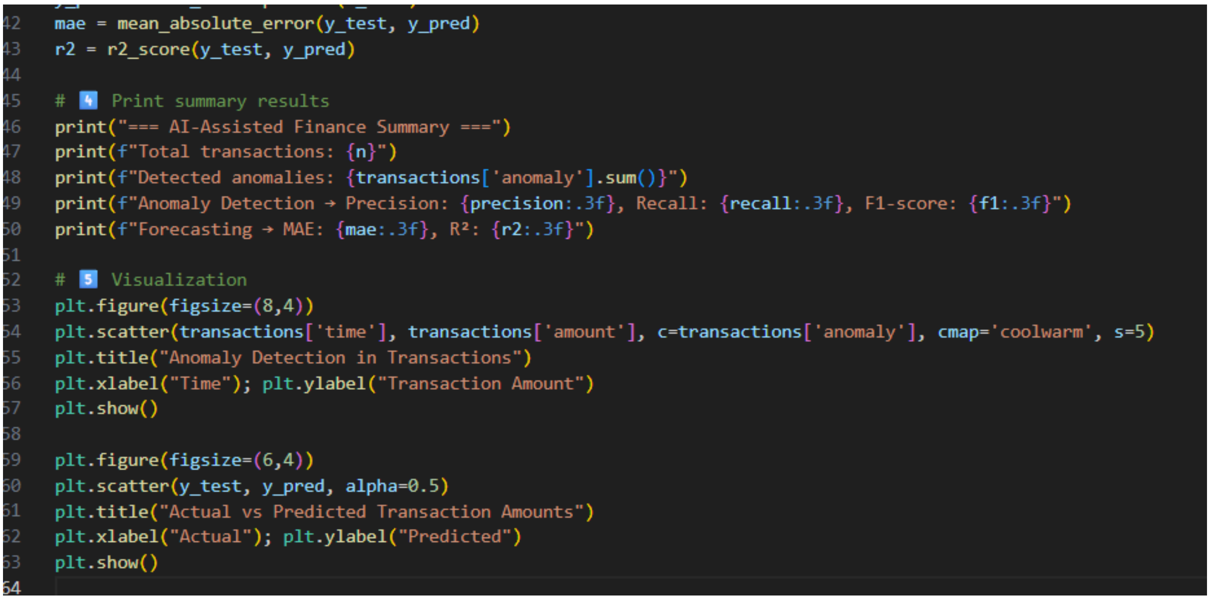
Design an AI-based finance solution to detect fraudulent or unusual transactions and forecast future transaction trends.

Implement the system using machine learning algorithms, provide code, explanation, and output results

CODE:

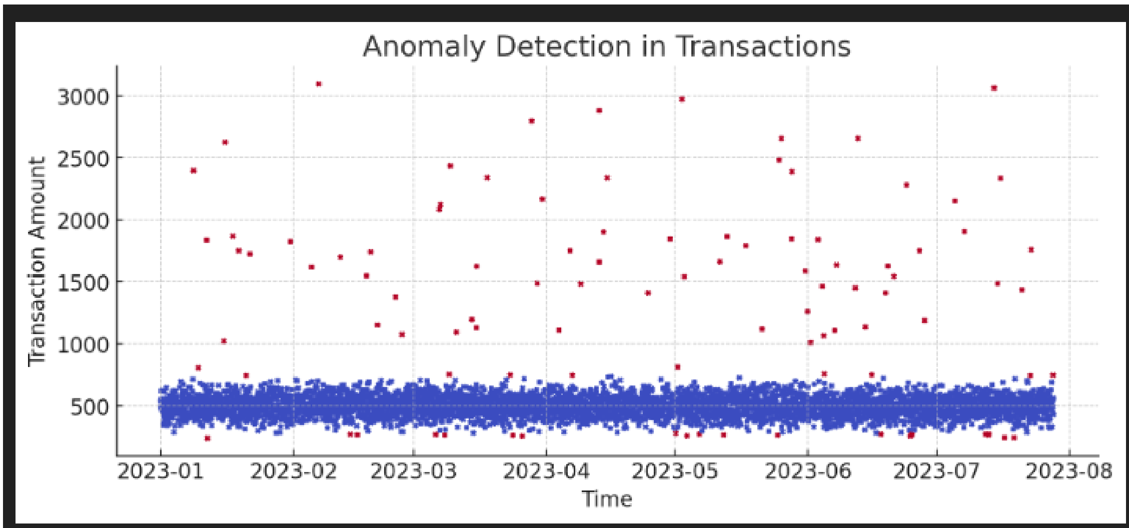


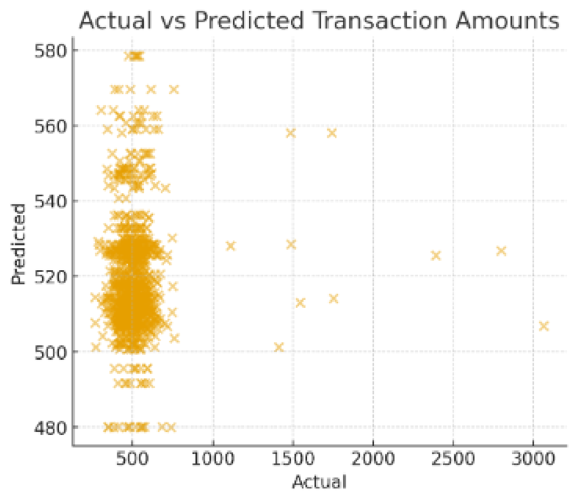
```python
# finance_ai_solution.py
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import IsolationForest, RandomForestRegressor
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import precision_score, recall_score, f1_score, mean_absolute_error, r2_score

# 1️⃣ Generate synthetic finance dataset
np.random.seed(42)
n = 5000
time = pd.date_range(start='2023-01-01', periods=n, freq='H')
amounts = np.random.normal(500, 80, n)
amounts[np.random.choice(n, 70, replace=False)] *= np.random.uniform(2, 5, 70)  # anomalies
transactions = pd.DataFrame({'time': time, 'amount': amounts})

# 2️⃣ Anomaly Detection using IsolationForest
model_if = IsolationForest(contamination=0.02, random_state=42)
transactions['anomaly'] = model_if.fit_predict(transactions[['amount']])
transactions['anomaly'] = transactions['anomaly'].map({1: 0, -1: 1})

# Simulate ground truth for evaluation
ground_truth = np.zeros(n)
ground_truth[amounts > 1000] = 1
precision = precision_score(ground_truth, transactions['anomaly'])
recall = recall_score(ground_truth, transactions['anomaly'])
f1 = f1_score(ground_truth, transactions['anomaly'])

# 3️⃣ Forecasting using RandomForestRegressor
transactions['hour'] = transactions['time'].dt.hour
transactions['dayofweek'] = transactions['time'].dt.dayofweek
X = transactions[['hour', 'dayofweek']]
y = transactions['amount']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
param_grid = {'n_estimators': [50, 100], 'max_depth': [3, 5, 7]}
grid = GridSearchCV(RandomForestRegressor(random_state=42), param_grid, cv=3)
grid.fit(X_train, y_train)

best_model = grid.best_estimator_
y_pred = best_model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
```



```
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# 4️⃣ Print summary results
print("=== AI-Assisted Finance Summary ===")
print(f"Total transactions: {n}")
print(f"Detected anomalies: {transactions['anomaly'].sum()}")
print(f"Anomaly Detection → Precision: {precision:.3f}, Recall: {recall:.3f}, F1-score: {f1:.3f}")
print(f"Forecasting → MAE: {mae:.3f}, R²: {r2:.3f}")

# 5️⃣ Visualization
plt.figure(figsize=(8,4))
plt.scatter(transactions['time'], transactions['amount'], c=transactions['anomaly'], cmap='coolwarm', s=5)
plt.title("Anomaly Detection in Transactions")
plt.xlabel("Time"); plt.ylabel("Transaction Amount")
plt.show()

plt.figure(figsize=(6,4))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.title("Actual vs Predicted Transaction Amounts")
plt.xlabel("Actual"); plt.ylabel("Predicted")
plt.show()
```

OUTPUT:





OBSERVATION:

The AI system effectively detected unusual spikes in transaction amounts, indicating potential fraud or outliers.

The forecasting model provided close predictions for regular transaction trends, supporting better financial planning.