**Phase:03 Project Development**

**Project-1:Anomaly Detection in Financial Transactions Using AI Data Analyst.**

**CollegeName:Dr.AmbedkarInstituteTechnology(Dr.AIT).**

**RAVI K R** **CAN_36049468**

# Project Development: Anomaly Detection in Financial Transactions

## 1. ✅ Project Setup

### 1.1. Requirements Installation

```
pip install pandas numpy scikit-learn matplotlib seaborn
```

## 2. Data Preparation

### 2.1. Load Transaction Data

```
import pandas as pd

# Example CSV format: ['transaction_id', 'amount', 'time', 'device_id', 'location',
'is_fraud']
df = pd.read_csv("transactions.csv")
df.head()
```

### 2.2. Explore & Clean

```
# Check for missing values
print(df.isnull().sum())

# Drop or fill missing values
df = df.dropna()
```

## 3. Feature Engineering

### 3.1. Selecting Relevant Features

```
features = ['amount', 'time']
X = df[features]
```

### 3.2. Normalize Features

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

## 4. Model Development

### 4.1. Isolation Forest

```
from sklearn.ensemble import IsolationForest

model = IsolationForest(n_estimators=100, contamination=0.01, random_state=42)
df['anomaly_score'] = model.fit_predict(X_scaled)

# Label anomalies
df['anomaly'] = df['anomaly_score'].apply(lambda x: 1 if x == -1 else 0)
```

## 5. Evaluation & Visualization

### 5.1. Visualize with PCA

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns

pca = PCA(n_components=2)
pca_result = pca.fit_transform(X_scaled)
df['pca1'], df['pca2'] = pca_result[:, 0], pca_result[:, 1]

# Plot
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='pca1', y='pca2', hue='anomaly', palette={0: 'blue', 1:
'red'})
plt.title("PCA Scatterplot of Anomalies")
plt.show()
```

### 5.2. Metrics (Optional if labeled data is available)

```
from sklearn.metrics import classification_report, confusion_matrix

print(confusion_matrix(df['is_fraud'], df['anomaly']))
print(classification_report(df['is_fraud'], df['anomaly']))
```

## 6. Save the Model

```
import joblib

joblib.dump(model, 'anomaly_detector.pkl')
joblib.dump(scaler, 'scaler.pkl')
```

## 7. Inference (Real-Time or Batch)

```
# Load model & scaler
model = joblib.load('anomaly_detector.pkl')
scaler = joblib.load('scaler.pkl')

# Example new data
new_data = pd.DataFrame({'amount': [1200], 'time': [30000]})
new_scaled = scaler.transform(new_data)
prediction = model.predict(new_scaled)

if prediction[0] == -1:
    print("Anomaly detected")
else:
    print("Normal transaction")
```

## ✓ Summary

- **Model**: Isolation Forest
- **Input**: Transaction amount, time
- **Detection**: Unsupervised anomaly detection
- **Deployment**: Can be saved, loaded, and used in real-time systems.