Anomaly Detection Hackathon(Code)

GitHub Repository:

https://github.com/Arif2455/AI_Anomaly_Detection_Hackathon

1) IMPORTS & SETUP :

```
import os

# create a project folder

if not os.path.exists("project"):

   os.makedirs("project")


print("📁 Project folder created:", os.listdir())


from google.colab import files

uploaded = files.upload()


import shutil

shutil.move("81ce1f00-c3f4-4baa-9b57-
006fad1875adTEP_Train_Test.csv",

          "project/TEP_Train_Test.csv")


print("File moved to project folder")

print(os.listdir("project"))
```

```
import pandas as pd
df = pd.read_csv("project/TEP_Train_Test.csv")
print(" Dataset Loaded! Shape:", df.shape)
df.head()
```

2) EXPLORATORY DATA ANALYSIS (EDA) :

```
# basic info
print(" Columns:", df.columns.tolist())
print(" Data Types:")
print(df.dtypes)

# check missing values
print("\n Missing Values per Column:")
print(df.isnull().sum())

# summary statistics
df.describe().T

import matplotlib.pyplot as plt
numeric_cols = df.select_dtypes(include=['int64','float64']).columns
first_col = numeric_cols[0]

plt.figure(figsize=(12,5))
plt.plot(df[first_col][:500])
plt.title(f"Sample of {first_col}")
```

```
plt.xlabel("Time Index")
plt.ylabel(first_col)
plt.show()

from sklearn.ensemble import IsolationForest
X = df[numeric_cols].fillna(0)

iso = IsolationForest(contamination=0.05, random_state=42)
df["anomaly"] = iso.fit_predict(X)
print(df["anomaly"].value_counts())

# visualize anomalies
plt.figure(figsize=(12,5))
plt.plot(df[first_col][:500], label="Data")
plt.scatter(df[df["anomaly"]==-1].index[:50],
        df[df["anomaly"]==-1][first_col][:50],
        color="red", label="Anomaly")
plt.legend()
plt.show()

from sklearn.covariance import EllipticEnvelope
ell = EllipticEnvelope(contamination=0.05, random_state=42)
df["ell_anomaly"] = ell.fit_predict(X)
print(df["ell_anomaly"].value_counts())
```

3) MODEL TRAINING & VISUALIZATION :

```
# count anomalies
anomaly_counts = df[df["anomaly"]==-
1][numeric_cols].count().sort_values(ascending=False)
print(anomaly_counts)


# time-based features (if available)
import matplotlib.pyplot as plt
if "timestamp" in df.columns or "time" in df.columns:
    ts_col = "timestamp" if "timestamp" in df.columns else "time"
    df[ts_col] = pd.to_datetime(df[ts_col], errors="coerce")
    df["hour"] = df[ts_col].dt.hour
    df["hour"].value_counts().sort_index().plot(kind="bar",
figsize=(10,5))
    plt.title("Flights per Hour")
    plt.show()


# anomaly vs normal
plt.figure(figsize=(12,5))
plt.plot(df[first_col][:1000], label="Data")
```

```
plt.scatter(df[df["anomaly"]==-1].index[:1000],
            df[df["anomaly"]==-1][first_col][:1000],
            color="red", label="Anomaly")
plt.legend()
plt.show()

# bar chart
counts = df["anomaly"].value_counts()
plt.bar(["Normal Flights", "Anomalous Flights"], counts.values,
color=["green", "red"])
plt.title("Normal vs Anomalous Flights")
plt.show()
```

4) EVALUATION :

```
from sklearn.metrics import classification_report, confusion_matrix

print("Isolation Forest Report:")
print(classification_report(df["anomaly"], df["anomaly"]))

print("Elliptic Envelope Report:")
print(classification_report(df["ell_anomaly"], df["anomaly"]))

print("Confusion Matrix (IsoForest):")
print(confusion_matrix(df["anomaly"], df["anomaly"]))
```

5) SAVING ARTIFACTS :

```
# save submission
df.to_csv("hackathon_submission.csv", index=False)
```