import streamlit as st

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.ensemble import IsolationForest

st.set\_page\_config(layout="wide")

st.title("🚨 Real-Time Anomaly Detection in MFT Logs")

@st.cache\_data

def load\_data():

n\_samples = 500

anomalies = 10

np.random.seed(42)

data = {

"file\_size\_MB": np.concatenate([np.random.normal(5, 1, n\_samples), np.random.normal(50, 5, anomalies)]),

"transfer\_time\_sec": np.concatenate([np.random.normal(10, 2, n\_samples), np.random.normal(100, 10, anomalies)]),

"hour\_of\_day": np.concatenate([np.random.normal(14, 3, n\_samples), np.random.normal(3, 1, anomalies)]),

"unique\_dest\_ips": np.concatenate([np.random.normal(1, 0.5, n\_samples), np.random.normal(5, 1, anomalies)]),

"user\_id\_encoded": np.concatenate([np.random.normal(1000, 50, n\_samples), np.random.normal(5000, 100, anomalies)]),

}

df = pd.DataFrame(data)

df["label"] = [0] \* n\_samples + [1] \* anomalies

return df

df = load\_data()

model = IsolationForest(contamination=0.02, random\_state=42)

df["anomaly\_score"] = model.fit\_predict(df.drop(columns=["label"]))

df["detected\_anomaly"] = df["anomaly\_score"].apply(lambda x: 1 if x == -1 else 0)

# Metrics

true\_positives = sum((df["label"] == 1) & (df["detected\_anomaly"] == 1))

false\_positives = sum((df["label"] == 0) & (df["detected\_anomaly"] == 1))

false\_negatives = sum((df["label"] == 1) & (df["detected\_anomaly"] == 0))

precision = true\_positives / (true\_positives + false\_positives + 1e-5)

recall = true\_positives / (true\_positives + false\_negatives + 1e-5)

st.metric("Precision", f"{precision:.2f}")

st.metric("Recall", f"{recall:.2f}")

# Chart

st.subheader("Anomaly Visualization")

fig, ax = plt.subplots(figsize=(10, 6))

sns.scatterplot(

x="file\_size\_MB", y="transfer\_time\_sec",

hue="detected\_anomaly", style="label", palette="coolwarm", data=df, ax=ax

)

plt.title("Anomaly Detection: File Size vs Transfer Time")

plt.grid(True)

st.pyplot(fig)

st.subheader("Sample Logs")

st.dataframe(df.head(20))

Got it — let's solve this! Since you're having trouble downloading directly from here, I’ll give you the full source code below. You can **copy and paste** it into your own .py file (for example: mft\_anomaly\_streamlit\_app.py) on your machine.

**✅ Step 1: Copy the code below into a .py file**

import streamlit as st

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.ensemble import IsolationForest

st.set\_page\_config(layout="wide")

st.title("🚨 Real-Time Anomaly Detection in MFT Logs")

@st.cache\_data

def load\_data():

n\_samples = 500

anomalies = 10

np.random.seed(42)

data = {

"file\_size\_MB": np.concatenate([np.random.normal(5, 1, n\_samples), np.random.normal(50, 5, anomalies)]),

"transfer\_time\_sec": np.concatenate([np.random.normal(10, 2, n\_samples), np.random.normal(100, 10, anomalies)]),

"hour\_of\_day": np.concatenate([np.random.normal(14, 3, n\_samples), np.random.normal(3, 1, anomalies)]),

"unique\_dest\_ips": np.concatenate([np.random.normal(1, 0.5, n\_samples), np.random.normal(5, 1, anomalies)]),

"user\_id\_encoded": np.concatenate([np.random.normal(1000, 50, n\_samples), np.random.normal(5000, 100, anomalies)]),

}

df = pd.DataFrame(data)

df["label"] = [0] \* n\_samples + [1] \* anomalies

return df

df = load\_data()

model = IsolationForest(contamination=0.02, random\_state=42)

df["anomaly\_score"] = model.fit\_predict(df.drop(columns=["label"]))

df["detected\_anomaly"] = df["anomaly\_score"].apply(lambda x: 1 if x == -1 else 0)

# Metrics

true\_positives = sum((df["label"] == 1) & (df["detected\_anomaly"] == 1))

false\_positives = sum((df["label"] == 0) & (df["detected\_anomaly"] == 1))

false\_negatives = sum((df["label"] == 1) & (df["detected\_anomaly"] == 0))

precision = true\_positives / (true\_positives + false\_positives + 1e-5)

recall = true\_positives / (true\_positives + false\_negatives + 1e-5)

st.metric("Precision", f"{precision:.2f}")

st.metric("Recall", f"{recall:.2f}")

# Chart

st.subheader("Anomaly Visualization")

fig, ax = plt.subplots(figsize=(10, 6))

sns.scatterplot(

x="file\_size\_MB", y="transfer\_time\_sec",

hue="detected\_anomaly", style="label", palette="coolwarm", data=df, ax=ax

)

plt.title("Anomaly Detection: File Size vs Transfer Time")

plt.grid(True)

st.pyplot(fig)

st.subheader("Sample Logs")

st.dataframe(df.head(20))

Step 1: Copy the code above into a .py file

**✅ Step 2: Run it in Streamlit**

1. Open Terminal / Command Prompt.
2. Navigate to the folder where you saved the file.
3. Run:

streamlit run mft\_anomaly\_streamlit\_app.py

Let me know if you'd like to add real-time log ingestion, Slack/email alerts, or connect it to a real file transfer system for live testing.

**To run the app:**

1. Make sure you have Streamlit installed:

bash

CopyEdit

pip install streamlit

1. Run the app:

bash

CopyEdit

streamlit run mft\_anomaly\_streamlit\_app.py