```
!pip install pandas pyarrow fastparquet
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: pyarrow in /usr/local/lib/python3.11/dist-packages (18.1.0)
Collecting fastparquet
  Downloading fastparquet-2024.11.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (4.2 kB)
Requirement already satisfied: numpy≥1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil≥2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz≥2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: tzdata≥2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: cramjam≥2.3 in /usr/local/lib/python3.11/dist-packages (from fastparquet) (2.9.1)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from fastparquet) (2025.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from fastparquet) (24.2)
Requirement already satisfied: six≥1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil≥2.8.2→pandas)
Downloading fastparquet-2024.11.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 13.5 MB/s eta 0:00:00
Installing collected packages: fastparquet
Successfully installed fastparquet-2024.11.0
```

```
import os
from google.colab import drive
drive.mount('/content/drive')

folder_path = "/content/drive/My Drive/GenAI/"
```

```
Mounted at /content/drive
```

```
import pandas as pd

# Read a Parquet file
dataset_path = folder_path + "data/unsupervised learning/dataset.parquet"
print(os.path.exists(dataset_path))

df = pd.read_parquet(dataset_path)  # Uses pyarrow or fastparquet

# Display the first few rows
print(df.head())
```

```
True
           ssn            cc_num    first   last gender      city state  \
0  367-85-9826  4361337605230458  Kristie  Davis      F  Chandler    OK
1  367-85-9826  4361337605230458  Kristie  Davis      F  Chandler    OK
2  367-85-9826  4361337605230458  Kristie  Davis      F  Chandler    OK
3  367-85-9826  4361337605230458  Kristie  Davis      F  Chandler    OK
4  367-85-9826  4361337605230458  Kristie  Davis      F  Chandler    OK

     zip  city_pop                     job         dob      acct_num  \
0  74834      7590  Chief Strategy Officer  1987-06-12  349734538563
1  74834      7590  Chief Strategy Officer  1987-06-12  349734538563
2  74834      7590  Chief Strategy Officer  1987-06-12  349734538563
3  74834      7590  Chief Strategy Officer  1987-06-12  349734538563
4  74834      7590  Chief Strategy Officer  1987-06-12  349734538563

                          trans_num  trans_date trans_time   unix_time  \
0  c036244703adb9d5392f4027d9d4b38d  2021-07-31   02:30:01  1627678801
1  42f000b0b3b0ef534e5b8ef9ec1db13a  2021-08-01   22:37:41  1627837661
2  543037b1baf088961e58d00b705f4bcc  2021-08-01   23:02:09  1627839129
3  00a4e08643edebf9277c2967676f6a26  2021-08-01   22:27:24  1627837044
4  492c4412815306718f686fc5b459a285  2021-12-02   02:28:51  1638392331

          category     amt  is_fraud               merchant
0     grocery_pos  337.54         1          fraud_Kovacek
1   personal_care   21.13         1          fraud_Bradtke
2   personal_care   22.61         1    fraud_Kozey-Kuhlman
3  health_fitness   17.32         1            fraud_Hills
4        misc_pos   75.82         0  fraud_Kemmer-Buckridge
```

ISOLATION FOREST

Useful for fraud detection purposes and is unsupervised learning

```
from sklearn.ensemble import IsolationForest

# Select relevant features
```

```
features = ["gender", "amt", "unix_time", "category", "merchant", "city_pop"]
df_selected = df[features]

# Encode categorical features
df_selected = pd.get_dummies(df_selected)

# Train Isolation Forest
# Assigning random state to give same results everytime
model = IsolationForest(contamination=0.4, random_state=42)  # 2% expected fraud
model.fit(df_selected)

# Predict fraud scores (-1 = anomaly, 1 = normal)
df["fraud_score"] = model.predict(df_selected)
df["fraud_detected_isoforest"] = (df["fraud_score"] == -1).astype(int)  # Convert to 0/1


from datetime import datetime
date_str = "2024-03-23 00:30:00"
unix = datetime.strptime(date_str, "%Y-%m-%d %H:%M:%S").timestamp()

new_transaction = pd.DataFrame([{
    "gender": "M",
    "amt": 500,
    "unix_time": unix,
    "category": "Entertainment ej eoijeo ijoeij oiejoij ",
    "merchant": "Amazon",
    "city_pop": 5000000000
}])

new_transaction_encoded = pd.get_dummies(new_transaction)

# Ensure all columns match the training dataset
missing_cols = set(df_selected.columns) - set(new_transaction_encoded.columns)

missing_df = pd.DataFrame(0, index=new_transaction_encoded.index, columns=list(missing_cols))
new_transaction_encoded = pd.concat([new_transaction_encoded, missing_df], axis=1)


# Reorder columns to match training data
new_transaction_encoded = new_transaction_encoded[df_selected.columns]


# Predict fraud score (-1 = fraud, 1 = normal)
fraud_score = model.predict(new_transaction_encoded)[0]

# Convert to readable format
fraud_detected = 1 if fraud_score == -1 else 0

print("Fraud Detected:", fraud_score, fraud_detected)
```

```
Fraud Detected: -1 1
```

```
import joblib

# Save the trained Isolation Forest model
model_path = folder_path + "data/unsupervised learning/isolation_forest_model.joblib"
print(os.path.exists(model_path))

joblib.dump(model, model_path)
print("Model saved successfully!")
```

```
False
Model saved successfully!
```

AUTO ENCODER

Deep learning unsupervised model

```
!pip install tensorflow keras
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.11/dist-packages (2.18.0)
Requirement already satisfied: keras in /usr/local/lib/python3.11/dist-packages (3.8.0)
Requirement already satisfied: absl-py ⩾ 1.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse ⩾ 1.6.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers ⩾ 24.3.25 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (25.2.10
```

```
Requirement already satisfied: gast≠0.5.0,≠0.5.1,≠0.5.2,≥0.2.1 in /usr/local/lib/python3.11/dist-packages (from tenso
Requirement already satisfied: google-pasta≥0.1.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: libclang≥13.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum≥2.3.2 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from tensorflow) (24.2)
Requirement already satisfied: protobuf≠4.21.0,≠4.21.1,≠4.21.2,≠4.21.3,≠4.21.4,≠4.21.5,<6.0.0dev,≥3.20.3 in /usr/l
Requirement already satisfied: requests<3,≥2.21.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from tensorflow) (75.1.0)
Requirement already satisfied: six≥1.12.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor≥1.1.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions≥3.6.6 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (4.1
Requirement already satisfied: wrapt≥1.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.2)
Requirement already satisfied: grpcio<2.0,≥1.24.3 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.71.0)
Requirement already satisfied: tensorboard<2.19,≥2.18 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.18
Requirement already satisfied: numpy<2.1.0,≥1.26.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.0.2)
Requirement already satisfied: h5py≥3.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.13.0)
Requirement already satisfied: ml-dtypes<0.5.0,≥0.4.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.4.
Requirement already satisfied: tensorflow-io-gcs-filesystem≥0.23.1 in /usr/local/lib/python3.11/dist-packages (from tens
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from keras) (13.9.4)
Requirement already satisfied: namex in /usr/local/lib/python3.11/dist-packages (from keras) (0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.11/dist-packages (from keras) (0.14.1)
Requirement already satisfied: wheel<1.0,≥0.23.0 in /usr/local/lib/python3.11/dist-packages (from astunparse≥1.6.0→ten
Requirement already satisfied: charset-normalizer<4,≥2 in /usr/local/lib/python3.11/dist-packages (from requests<3,≥2.2
Requirement already satisfied: idna<4,≥2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3,≥2.21.0→tensorf
Requirement already satisfied: urllib3<3,≥1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3,≥2.21.0→t
Requirement already satisfied: certifi≥2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3,≥2.21.0→t
Requirement already satisfied: markdown≥2.6.8 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,≥2.18→
Requirement already satisfied: tensorboard-data-server<0.8.0,≥0.7.0 in /usr/local/lib/python3.11/dist-packages (from ten
Requirement already satisfied: werkzeug≥1.0.1 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,≥2.18→
Requirement already satisfied: markdown-it-py≥2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich→keras) (3.0.0
Requirement already satisfied: pygments<3.0.0,≥2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich→keras) (2.1
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py≥2.2.0→rich→k
Requirement already satisfied: MarkupSafe≥2.1.1 in /usr/local/lib/python3.11/dist-packages (from werkzeug≥1.0.1→tensor
```

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler

# Select relevant features for Autoencoder
df_selected = df[features]

# Encode categorical features
df_selected = pd.get_dummies(df_selected)

# Normalize numerical features
scaler = StandardScaler()
batch_size = 10000
df_scaled_list = []

for i in range(0, len(df_selected), batch_size):
    batch = df_selected.iloc[i : i + batch_size]
    df_scaled_list.append(scaler.fit_transform(batch))

df_scaled = df_scaled_list[0]  # Start with the first batch

for batch in df_scaled_list[1:]:
    df_scaled = np.concatenate((df_scaled, batch), axis=0)  # Incrementally add batches


import tensorflow as tf
from tensorflow import keras
from keras.models import Model
from keras.layers import Input, Dense

# Define input size
input_dim = df_scaled.shape[1]

# Build Autoencoder model
input_layer = Input(shape=(input_dim,))
encoded = Dense(8, activation="relu")(input_layer)
encoded = Dense(4, activation="relu")(encoded)
decoded = Dense(8, activation="relu")(encoded)
decoded = Dense(input_dim, activation="sigmoid")(decoded)

autoencoder = Model(input_layer, decoded)
autoencoder.compile(optimizer="adam", loss="mse")

# Train the autoencoder
```

```
autoencoder.fit(df_scaled, df_scaled, epochs=50, batch_size=32, shuffle=True)



# Reconstruct transactions
reconstructed = autoencoder.predict(df_scaled)

# Compute reconstruction errors
mse = np.mean(np.abs(df_scaled - reconstructed), axis=1)

# Set a threshold for fraud (e.g., top 5% of errors)
threshold = np.percentile(mse, 99.6)

# Detect fraud (1 = fraud, 0 = normal)
df["fraud_detected_autoencoder"] = (mse > threshold).astype(int)



similarity_percentage = (df["fraud_detected_isoforest"] == df["fraud_detected_autoencoder"]).mean() * 100
print(f"Similarity between Isolation Forest and Autoencoder fraud detection: {similarity_percentage:.2f}%")
```