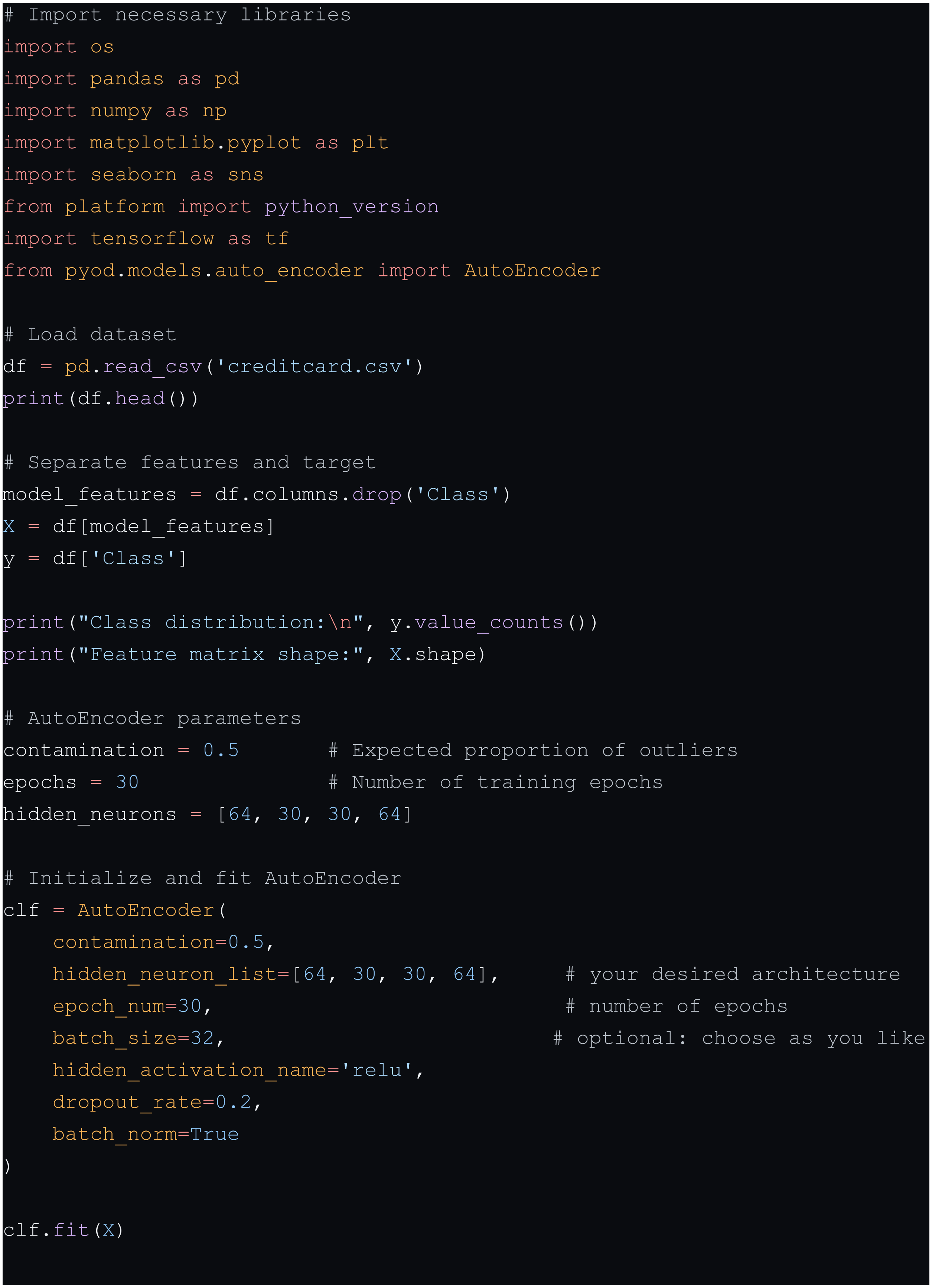


```
# Import necessary libraries
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from platform import python_version
import tensorflow as tf
from pyod.models.auto_encoder import AutoEncoder

# Load dataset
df = pd.read_csv('creditcard.csv')
print(df.head())

# Separate features and target
model_features = df.columns.drop('Class')
X = df[model_features]
y = df['Class']

print("Class distribution:\n", y.value_counts())
print("Feature matrix shape:", X.shape)

# AutoEncoder parameters
contamination = 0.5       # Expected proportion of outliers
epochs = 30               # Number of training epochs
hidden_neurons = [64, 30, 30, 64]

# Initialize and fit AutoEncoder
clf = AutoEncoder(
    contamination=0.5,
    hidden_neuron_list=[64, 30, 30, 64],     # your desired architecture
    epoch_num=30,                            # number of epochs
    batch_size=32,                          # optional: choose as you like
    hidden_activation_name='relu',
    dropout_rate=0.2,
    batch_norm=True
)

clf.fit(X)
```

```
# Predict outliers
outliers = clf.predict(X)
anomaly_indices = np.where(outliers == 1)[0]
print("Anomaly indices:", anomaly_indices)

# Example prediction on a sample
sample_idx = 4920
sample = X.iloc[[sample_idx]]
print("Sample features:\n", sample)
print("Sample actual class:", y.iloc[[sample_idx]])
print("Predicted class:", clf.predict(sample, return_confidence=False))
print("Prediction confidence:", clf.predict_confidence(sample))

# Get labels and decision scores
y_pred = clf.labels_  # Predicted labels for all data
y_scores = clf.decision_scores_  # Outlier scores (higher = more abnormal)

print("First 5 predicted labels:", y_pred[:5])
print("First 5 anomaly scores:", y_scores[:5])

# Plot anomaly scores with model threshold
plt.figure(figsize=(15, 8))
plt.plot(y_scores, label='Anomaly Scores')
plt.axhline(y=clf.threshold_, color='r', linestyle='dotted',
label='Threshold')
plt.xlabel('Instances')
plt.ylabel('Anomaly Scores')
plt.title('Anomaly Scores with Auto-Calculated Threshold')
plt.legend()
plt.show()

# Plot anomaly scores with custom threshold
custom_threshold = 50
plt.figure(figsize=(15, 8))
plt.plot(y_scores, color='green', label='Anomaly Scores')
plt.axhline(y=custom_threshold, color='r', linestyle='dotted',
label='Custom Threshold')
plt.xlabel('Instances')
plt.ylabel('Anomaly Scores')
plt.title('Anomaly Scores with Modified Threshold')
plt.legend()
plt.show()

# Plot training loss history
plt.figure(figsize=(15, 8))
pd.DataFrame(clf.history_).plot(title='AutoEncoder Training Loss')
plt.show()

# Scatter plot of transactions with anomaly scores
plt.figure(figsize=(15, 8))
sns.scatterplot(
    x='Time',
    y='Amount',
    hue=y_scores,
    size=y_scores,
    palette='RdBu_r',
    data=df,
    legend='full'
)
plt.xlabel('Time (seconds elapsed from first transaction)')
plt.ylabel('Transaction Amount')
plt.title('Transaction Scatter Plot Colored by Anomaly Scores')
plt.legend(title='Anomaly Scores')
plt.show()
```

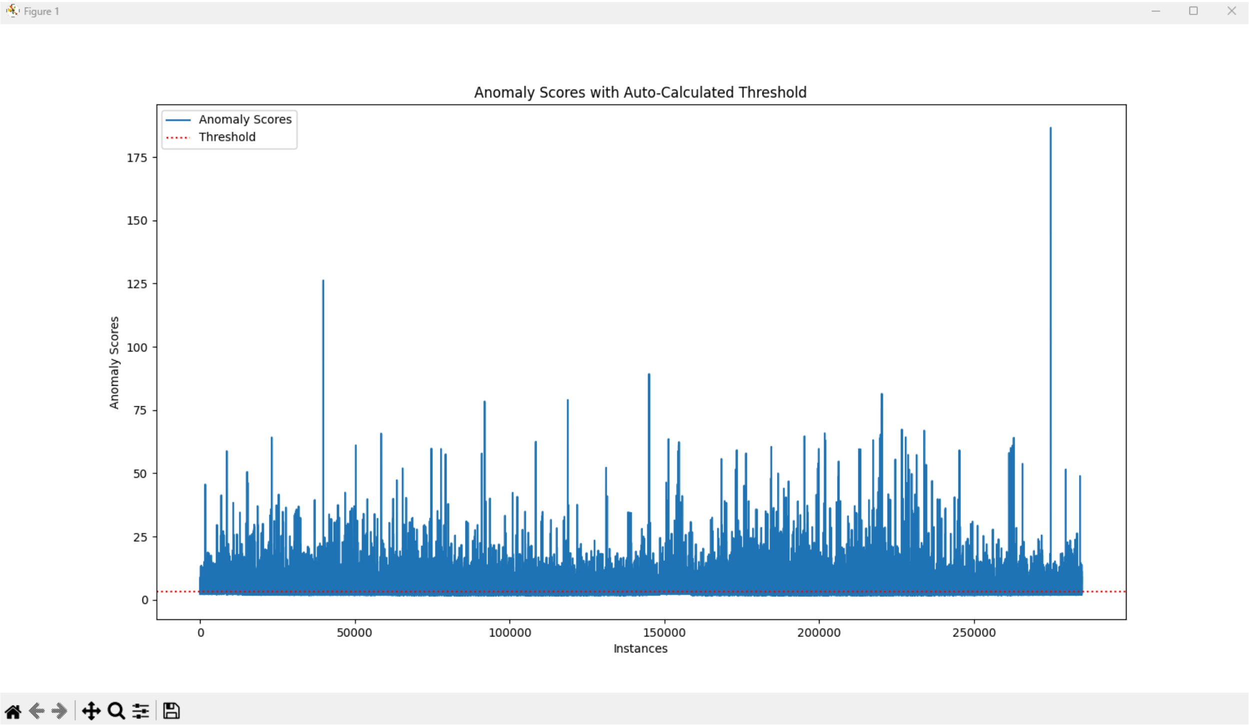


Figure 1
Anomaly Scores with Auto-Calculated Threshold
Anomaly Scores
Threshold
Anomaly Scores
0
25
50
75
100
125
150
175
0
50000
100000
150000
200000
250000
Instances

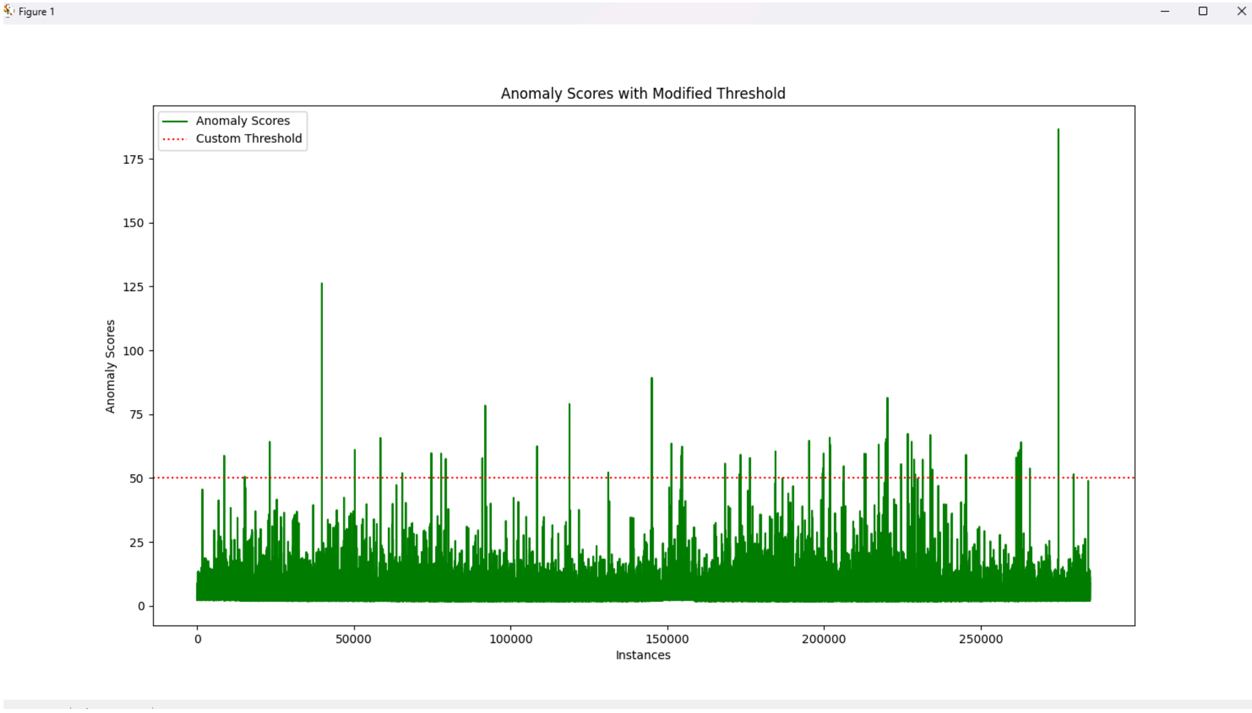
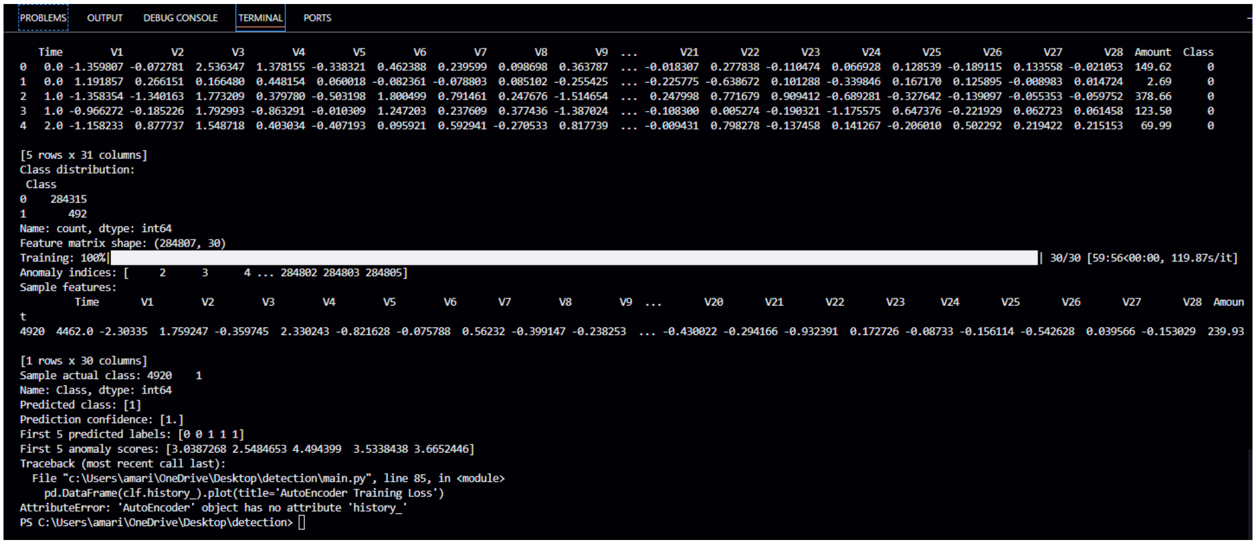


Figure 1
Anomaly Scores with Modified Threshold
Anomaly Scores
Custom Threshold
Anomaly Scores
0
25
50
75
100
125
150
175
0
50000
100000
150000
200000
250000
Instances



```
   Time        V1        V2        V3        V4        V5        V6        V7        V8        V9  ...       V21       V22       V23       V24       V25       V26       V27       V28  Amount  Class
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599  0.098698  0.363787  ... -0.018307  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053  149.62      0
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803  0.085102 -0.255425  ... -0.225775 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724    2.69      0
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461  0.247676 -1.514654  ...  0.247998  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752  378.66      0
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609  0.377436 -1.387024  ... -0.108300  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458  123.50      0
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941 -0.270533  0.817739  ... -0.009431  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153   69.99      0

[5 rows x 31 columns]
Class distribution:
 Class
0    284315
1       492
Name: count, dtype: int64
Feature matrix shape: (284807, 30)
Training: 100%|██████████████████████████████████████████████████| 30/30 [59:56<00:00, 119.87s/it]
Anomaly indices: [     2      3      4 ... 284802 284803 284805]
Sample features:
         Time       V1        V2        V3        V4        V5        V6       V7        V8        V9  ...       V20       V21       V22       V23      V24       V25       V26       V27       V28  Amoun
t
4920  4462.0 -2.30335  1.759247 -0.359745  2.330243 -0.821628 -0.075788  0.56232 -0.399147 -0.238253  ... -0.430022 -0.294166 -0.932391  0.172726 -0.08733 -0.156114 -0.542628  0.039566 -0.153029  239.93

[1 rows x 30 columns]
Sample actual class: 4920    1
Name: Class, dtype: int64
Predicted class: [1]
Prediction confidence: [1.]
First 5 predicted labels: [0 0 1 1 1]
First 5 anomaly scores: [3.0387268 2.5484653 4.494399  3.5338438 3.6652446]
Traceback (most recent call last):
  File "c:\Users\amari\OneDrive\Desktop\detection\main.py", line 85, in <module>
    pd.DataFrame(clf.history_).plot(title='AutoEncoder Training Loss')
AttributeError: 'AutoEncoder' object has no attribute 'history_'
PS C:\Users\amari\OneDrive\Desktop\detection> 
```