ELEC 472 - Artificial Intelligence: Lab 4

Objectives

This laboratory activity is focused on PCA and Feature Extraction.

Submission and Deliverable

Please submit the following through onQ:

- ***one report*** (with ***8*** parts) – the report must be created in MS Word (or latex) and converted to PDF prior to submission,
- ***three*** Python files (named: lab4_PCA.py, lab4_data_visulaization_modified.py, and lab4_feature_extraction_modified.py).

1. PCA

Write all the PCA-related code in a file called ***lab4_PCA.py*** (which you need to submit). The goal of the first step is to read and visualize the data. Download the file named **dataset.csv** from OnQ. The dataset contains readings from 4 accelerometers placed on the shoulder, hip, arm, and leg, each with 3 axes, resulting in 12-dimensional data. The first 6 columns contain metadata and can be ignored for PCA.

Run the following code:

```
# If you do not have pandas installed, please install it using `pip install pandas`
import pandas as pd
from matplotlib import pyplot as plt
import numpy as np

dataset_full=pd.read_csv('dataset.csv', delimiter=';')

# We want to take a subset of the dataset (the entire dataset is too large).
x = np.array(dataset_full.values[155750:156250, 6:18], dtype=float)
x_names=list(dataset_full.columns)[6:18]

print('feature shape: ', x.shape)
print('feature names: ', x_names)

# We can plot the 12 dimensions.
plt.figure()
t=[i for i in range(x.shape[0])]
for k in range(len(x_names)):
    plt.plot(t, x[:, k], label=x_names[k])

plt.legend(loc='upper right')
plt.title('Feature visualization')
plt.savefig('fig1.pdf')
```

Report - Part 1: Take a screenshot of the plot and include it in the ***Report***.

Next, we aim to normalize the data. To this end, we can use the following code:

```
# Calculate the mean and standard deviation:
m_x = np.mean(x, axis=0)
s_x = np.std(x, axis=0)

# Perform normalization
x_bar = (x - m_x) / s_x

plt.figure()
t=[i for i in range(x_bar.shape[0])]
for k in range(len(x_names)):
    plt.plot(t, x_bar[:, k], label=x_names[k])

plt.legend(loc='upper right')
plt.title('Normalized feature visualization')
plt.savefig('fig2.pdf')
```

Report - Part 2: Take a screenshot of the plot and include it in the ***Report***. See the "range" of the data before and after normalization. State your observation.

Next, we aim to reduce the dimensionality of the dataset since we expect there to be redundancy. For instance, the movements of the two arms during walking are typically similar and highly correlated. To eliminate this redundancy and simplify our data, we will use PCA.

Report - Part 3: Implement a Python function named ***pca*** to perform Principal Component Analysis with the following specifications:

Input:

- x: Input data of shape nxm, where n is the number of samples and m is the number of dimensions/features.

Output:

- pcs: The sorted eigenvectors representing the principal components.
- scores: The transformed data represented in the principal component space.
- explained: The variance explained by each principal component.

Below is a step-by-step guideline to implement your PCA function:

```
import numpy as np

def pca(x):
```

```
    # Step 1: Mean-center the data

    # Step 2: Compute the covariance matrix (use np.cov)

    # Step 3: Perform eigen decomposition (use np.linalg.eigh)

    # Step 4: Sort the eigenvalues AND eigenvectors in decreasing order
    # (Hint: use np.argsort on eigenvalues)

    # Step 5: Select the sorted eigenvectors as principal components (pcs)

    # Step 6: Compute the variance explained by each principal component (explained)

    # Step 7: Project the data onto principal component axes to obtain scores

    return pcs, scores, explained

pcs, scores, explained = pca(x)
```

Report - Part 4: Plot the transformed data (scores) obtained from PCA and include a screenshot of this plot in your ***Report***. By examining this plot, discuss how the first few principal components (new axes) effectively capture the most important aspects of the original data. Clearly state your observations in the ***Report***.

Report - Part 5: Plot the variance explained by each principal component (the explained parameter). Include this plot in your ***Report*** and briefly discuss your findings.

Report - Part 6: Plot the cumulative sum of the explained variance. Include this figure in your ***Report***. Using this plot, answer the following question clearly in your ***Report***:
"What is the minimum number of principal components required to capture at least 98% of the variance (information) from the original dataset?"

2. Feature Extraction

In this part of the lab, you will perform feature extraction on a Human Activity Recognition dataset from the UCI Machine Learning Repository:
https://archive.ics.uci.edu/dataset/240/human+activity+recognition+using+smartphones

Download the file ***raw_accelerometer_dataset.csv*** from OnQ to your working directory. This dataset is a smaller subset of the original dataset, containing data from a single accelerometer with three axes (X, Y, Z), along with labels corresponding to one of six activities: 'bike', 'sit', 'stairsdown', 'stairsup', 'stand', 'walk'. Use the provided file ***lab4_data_visulaization.py***, which imports and visualizes the X-axis data for you.

Report - Part 7: Modify the provided visualization code (lab4_data_visulaization.py) to additionally plot the Y and Z axes of the accelerometer data. Each axis (X, Y, and Z) should be visualized in a separate plot. Save your modified file as ***lab4_data_visulaization_modified.py***.

Include screenshots of all three plots (clearly labeled X, Y, and Z) in your report. Make sure to update each figure title appropriately to indicate the axis plotted.

Report - Part 8: Now you will extract features from the raw accelerometer signals. For time-series data, we typically extract statistical features using a sliding window approach. Common features include: Maximum, Minimum, Mean, Standard Deviation, Skewness, Entropy, and others. Download the provided file ***lab4_feature_extraction.py***, which is initially set up to extract only the maximum value from each sliding window for all three accelerometer axes. Now perform the following:

a) Modify this code to extract Maximum, Minimum, Mean, Standard Deviation, Skewness, Entropy. You are also encouraged to come up with other features in addition to these – but it is not mandatory to do so. Include screenshots of your modified feature extraction code in your ***Report***.

b) Create two sets of extracted features, one with a window size (segment size) of 500 samples and another with a window size of 250 samples. Briefly discuss how these two different window sizes affect your feature sets. Include screenshots of your modified feature extraction code in your ***Report***. Also, compare the differences and implications of the two window sizes.

Note: Ensure the file features_500.csv is generated by your script and stored automatically upon execution. Keep this file safe, as it will be used in the next lab session for training machine learning models.