Feature Extraction Project

Part I

Implement several feature extraction methods that reduce the dimensionality of the data from m to 2.

Input: two files: data and labels. The data is a comma separated matrix of size $n \times m$. Here the data points are the rows, not the columns. The "labels" is an array of size n. The labels can only take the values 1,2,3.

Output:
- A comma separated file containing the $n \times 2$ matrix of the reduced data.
- A comma separated file containing the two vectors v_1, v_2 as a matrix of size $2 \times m$.

Programs: Read the files for the matrices X^t and the labels y. Use these to calculate two vectors v_1, v_2, each of size m. Compute the projections of X^t on v_1, v_2 and use them to create the matrix D of size $n \times 2$.

Visualization: For small data you can visualize your results using the program **plot.py**.

Here are the programs that you are asked to implement:

1. **pca1.py** : use PCA without subtracting the mean.
2. **pca2.py** : use PCA with mean subtraction.
3. **scatter1.py** : minimize the within-class scatter.
4. **scatter2.py** : maximize the between-class scatter.
5. **scatter3.py** : maximize the ratio of between-class scatter and within-class scatter.

Try your programs on the following two datasets: **irises, wine**.

Part II

Feature extraction can be guided by various criteria. In this part our goal is to perform dimensionality reduction for nearest neighbor applications. Specifically, training is performed with the input being the same as in Part I. Then, given a test vector x of length m, it is reduced into two dimensions v_x, v_y. The program searches for the vector in the training data that is nearest to v_x, v_y and returns its index.

You are asked to implement such programs for two cases. In the first case all items in the training data are candidates for the nearest neighbor. In the second case the nearest neighbor must be of a specific label.

Programs

1. **reducedim1.py** : reduced dimension for the first case.
2. **reducedim2.py** : reduced dimension for the second case.
3. **nearest.py** : find nearest neighbor.

The arguments for **reducedim1.py** and **reducedim2.py** are the same as in Part I. The arguments for **nearest.py** are as follows:

```
nearest.py reduced.txt V.txt labels.txt label
```

Here **reduced.txt** and **V.txt** are the output files of the dimensionality reduction programs. **labels.txt** is the same as the input to the dimensionality reduction program. If Label is 1/2/3/ compute nearest neighbor of the appropriate label. Otherwise ignore the label in computing the nearest neighbor.

The output of the program is the row of the nearest neighbor in the **reduced.txt** file.

What you need to submit

- Source code and documentation for the python scripts.

You **must** be available to demonstrate your program to the TAs. Time slots and additional instructions will be announced later.

Deadline:

TBA.