

C S 487/519 Applied Machine Learning

Dimensionality reduction techniques

1 Objective

In this *individual* homework, you are required to understand and compare several dimensionality reduction techniques.

2 Requirements

2.1 Tasks

- (1) (36 points) Write code to conduct dimensionality reduction by
 - (a) (12 points) using Principal Component Analysis (PCA) approach offered by scikit-learn library,
 - (b) (12 points) using the Linear Discriminant Analysis method offered by scikit-learn library, and
 - (c) (12 points) using a kernel PCA method offered by scikit-learn library.
- (2) (22 points) Compare the performance of the different dimensionality reduction techniques. Please verify the performance of these techniques by feeding the dimension reduced data to a decision tree classifier. You may want to change different parameters (e.g., the number of components, kernels, etc.)
- (3) (22 points) The algorithms need to be tested using two datasets: (d1) the Iris dataset (`iris.data`) with description (`iris.names.txt`), which can be downloaded from this page, and (d2) the MNIST dataset, which can be loaded from sklearn.datasets using the function `fetch_mldata` (scikit-learn version before 0.19) or `fetch_openml` function (scikit-learn version from 0.20).

Note: The MNIST dataset has 70K instances. It might be too large for your computer (or our CS server) memory to do all calculations, especially for KernelPCA. To work on your homework without having this memory problem, you can extract a subset of the dataset by using the `train-test-split` method with the `stratify` option to get an `MNIST_subset`. Then, you can use this `MNIST_subset` as the MNIST and later split `MNIST_subset` to `X_train` and `X_test` for the rest of your calculations. If the subset has 1000-2000 rows, the model should work without the memory issue.
- (4) (15 points) Properly analyze the different algorithms behavior by applying the knowledge that we discussed in class. Such analysis should include classification accuracy (or F1, precision, recall, ROC, AUC) and running time. You can also use other metrics that look reasonable to conduct the analysis. Put the analysis to report.pdf file.
- (5) (5 points) Write a readme file `readme.txt` with detailed instructions to run your program.

2.2 Other requirements

- Your Python code should be written for **Python version 3.5.2 or higher**.
- Please write proper **comments** in your code to help the instructor and teaching assistants to understand it.
- Please properly organize your Python code (e.g., create proper classes, modules).
- You can put your code to Jupyter Notebook or a `.py` file.

3 Submission instructions

Put all your files (Python code, readme file, report, etc.) to a zip file named `hw.zip` and upload it to Canvas.

4 Grading criteria

- (1) **ZERO point will be given if your code does not work. Please do not submit code that you did not test and make sure it works.**
- (2) The score allocation has been put beside the questions.
- (3) FIVE points will be deducted if files are not submitted in the required format.
- (4) If the total points are more than 100. Your grades will be scaled to the range of $[0,100]$.
- (5) Please make sure that you test your code thoroughly by considering all possible test cases. For this homework, your code will NOT be tested using more datasets. Thus, it does not need to be flexible to accept different datasets as input.