Practical exercises for the course ‘Fundamentals of Machine Learning, unsupervised approach’

Instructions:

The project consists of three exercises each from the following three types:

1) Implement a simple ML method from scratch
2) Exploratory analysis of a dataset
3) Derivation of a theoretical results

You can choose one of the exercises from each category at your own taste.

You can use any programming language (with Python in preference, if you master it).

You can work in couples if you prefer.

I recommend using Google Colab environment and Jupyter notebooks, with test datasets being downloaded online or from your Google Drive.

Template Google Colab notebook with examples of loading the datasets can be found here:
https://colab.research.google.com/drive/1kIYTe2MH8iwNjmrpp9W31nIOVoQG49jJ?usp=sharing

You can use any available online tutorials and the description of the algorithms but you should be able to explain any line of code you write. For Part 1 exercises, the implementation must not be a copy-paste from the existing implementations, it must be original translation from any available pseudo-code algorithm description (which would be useful to show in comments)

You should prepare a report for your project, which you will defend. If you use Google Colab then the report can be the notebook itself (just share it with my gmail account Andrei.Zinovyev.U900@gmail.com)

The derivation of theoretical results can be done analytically or can be replaced by a demonstration using a computational experiment (or both, which will be the best)

The exercise is considered to be done if the tasks written with the font put in bold are done. The rest is for your discretion in case the obligatory tasks will appear too easy problems. Performing them will be a bonus for your final grade

** and *** denote exercises of increased difficulty level (if other problems look too simple)

You can suggest your own project based on the material studied in the class in one of the three categories (implementation of ML methods from scratch, exploratory analysis of a dataset, proof of a theoretical result), but ask for my agreement first

Part 1. Implementing one Machine Learning algorithm from scratch (in any programming language)

Exercise 1.1. Implementing k-means clustering algorithm

1) **Implement the basic version of the algorithm**
2) **Test it using iris and MNIST test datasets**
3) Implement a version with multiple initializations and finding the best solution
4) Think of the initialization strategy, compare it with a random one
5) Try to speed up the neighbor search, demonstrate when it can be advantageous

Exercise 1.2. Implementing hierarchical clustering algorithm

1) **Implement the basic version of the algorithm, having as input a distance matrix and one possible choice for the distance between clusters (or all of them with a possibility to choose)**
2) **Test it using iris dataset**
3) Add clustering quality check using silhouette
4) Compare different ways to compute distance between clusters, in terms of clustering results

Exercise 1.3. Implementing data whitening algorithm

1) Use MNIST dataset as example or any other dataset with the number of features > 10
2) **Use standard functions of scikit-learn to compute principal components, and suggest an appropriate method to choose the number of components**
3) **Implement the data whitening**
4) **Demonstrate that PCA cannot be applied to the whitened data**
5) Try to compare different dimensionality reduction approaches (e.g., tSNE) on the initial and whitened data

Exercise 1.4. Implementing several methods to choose the number of principal components

1) Use the available implementation of PCA in scikit-learn
2) **Use MNIST dataset as an example or any other dataset with the number of features >50, produce a scree plot for it**
3) **Implement Kaiser, based on explained fraction of variance,** broken stick distribution, **conditional number-based** and, may be, other methods for selecting the number of components
4) **Compare different methods on the same dataset**, compute the amount of variance explained in each choice
5) Choose a subset of features, duplicate them, study the effect on the estimating the number of components

Exercise 1.5. Implementing Kernel Density Estimate of Probability Density Function

1) **Use iris dataset as test dataset**
2) **Implement the KDE approach for the Gaussian kernel**
3) Reconstruct the probability density function for the complete dataset and for each class separately
4) Visualize the results of 3) using an approach of your choice (e.g., PCA)
5) Use the reconstructed probability density function to compute the conditional probability of a data point to belong to a particular class (suggestion: use logarithms of probabilities)

Exercise 1.6**. Using Principal Component Analysis as a special form of regression for predicting some data features from the others

1) **Use iris as example dataset or any other dataset**
2) **Use the standard implementation of PCA in scikit-learn in order to compute PCA of order k=1,2,3,4.**
3) **Predict one column out of four using three other columns and the computed principal components**
4) Investigate how the prediction quality depends on k, and on the column. As a measure of quality prediction use mean squared error or any other suitable error measure
5) Predict two columns from the other two using PCA
6) Split the dataset into the training and test parts, compute PCA of order k=1,2,3,4 on the training dataset and compute the test error. Check how the test error depend on the number of components *k*.

Exercise 1.7***. Implementing computation of the principal components

1) **Use iris and/or MNIST datasets as examples**
2) **Implement the iterative SVD-based first principal component computation using the material from the lectures**
3) **Compare the results with the standard implementation of PCA in scikit-learn**
4) Implement the computation of the second and all higher components
5) Generalize your implementation to the case of weighted data points (in test example, create weights which will be much higher for a particular class of data points)
6) Think how to generalize your implementation to the case of moderately missing data

Part 2. Analyses of datasets using unsupervised approach

Dataset 1. Fashion MNIST dataset

This is one of the standard datasets used to train ML and AI methods. Each object in the dataset represents an image of a fashion item (10 distinct labels). CSV version of the dataset is available from here: https://media.githubusercontent.com/media/fpleoni/fashion_mnist/master/fashion-mnist_train.csv

Dataset 2. Single cell dataset

This is a dataset describing development of a dentate gyrus, a part of the mouse brain. Each object is a brain cell characterized by the expression of ~10000 genes. The cells are labeled into 24 distinct cell types. Zipped CSV version of the dataset is available from here: https://www.ihes.fr/~zinovyev/FundamentalsOfAI2020_lectures/datasets/dentate_gyrus_sample3000.zip

Dataset 3. Analysis of a clinical dataset

This is a dataset describing complications of myocardial infarction. The variables of the dataset are described here : https://leicester.figshare.com/articles/Myocardial_infarction_complications_Database/12045261?file=22803572. The tab-delimited file of the dataset is available here: https://raw.githubusercontent.com/auranic/ClinTrajan/master/data/infarction/infarctus_na_v30s20.txt

Check this tutorial https://github.com/auranic/ClinTrajan/blob/master/tutorial/tutorial.md for example of use of this dataset in a research project.

Exercise:

1) **Choose one of the datasets 1-3 or suggest you own**
2) **Check if you have missing values and suggest a strategy for imputing them**
3) **Check if the dataset needs to be normalized in a particular way**
4) **Give an estimate of intrinsic dimensionality of the dataset** (if you want you can use skdim package https://github.com/j-bac/scikit-dimension/tree/master/skdim)
5) **Apply PCA and visualize the dataset in the space of two first principal components**
6) Investigate higher-order principal components, try to conclude if they contain useful information
7) Apply any non-linear manifold learning method and visualize the dataset
8) **Apply two or more different clustering methods using clustering quality criteria, compare the results as well as computational time**
9) Apply any matrix factorization method which is not PCA, compare it with PCA as well as computational time

Part 3. Derivation of theoretical results

Exercise 3.1. Demonstrate that the matrix of scalar products can be derived from the distance matrix via double-centering, in the case of Euclidean distance

Exercise 3.2. Demonstrate that the principal components are eigenvectors of empirical covariance matrix (use lecture slides). After the proof, consider the case when two eigenvalues are the same.

Exercise 3.3. Calculate the entropy of normal distribution. Calculate the entropy of uniform distribution with unit variance. Compare the results.

Exercise 3.4.** Prove the relation between k-means clustering and NMF:

Theorem: if ***H*** is orthogonal (***HH***^{T} is a diagonal matrix) then one of the possible solutions to the problem

$$||X - WH||^2 \rightarrow min, subject\ to\ W \geq 0, H \geq 0$$

is given by K-means clustering of columns in ***X. W*** are cluster centroids in this case.

Hint: use the article http://ranger.uta.edu/~chqding/papers/NMF-SDM2005.pdf

Exercise 3.5***. Demonstrate that

$$I(s_1, s_2, ..., s_m) = const - \sum_{i=1}^{m} J(s_i)$$

for whitened data distribution, where *I* is the mutual information, and *J* is negentropy. Hint: use this article : https://hal.archives-ouvertes.fr/hal-00417283/document