Master Degree in Artificial Intelligence
Statistical and Mathematical Methods for Artificial Intelligence

2023-2024

Application: SVD and PCA for Machine Learning

## Visualizing dyad.

Consider an image from `skimage.data`. For simplicity, say that $X \in \mathbb{R}^{m \times n}$ is the matrix representing that image. You are asked to visualize the dyad of the SVD Decomposition of $X$ and the result of compressing the image via SVD. In particular:

- Load the image into memory and compute its SVD;
- Visualize some of the dyad $\sigma_i u_i v_i^T$ of this decomposition. What do you notice?
- Plot the singular values of $X$. Do you note something?
- Visualize the $k$-rank approximation of $X$ for different values of $k$. What do you observe?
- Compute and plot the approximation error $||X - X_k||_F$ for increasing values of $k$, where $X_k$ is the $k$-rank approximation of $k$.
- Plot the compression factor $c_k = \frac{k(m+n+1)}{mn}$ for increasing $k$. What is the approximation error when the compressed image requires the same amount of informations of those of the uncompressed image (i.e. $c_k = 1$)?

## Classification of MNIST Digits with SVD Decomposition.

The task for this exercise is to learn the classification of MNIST digits by using SVD decomposition. Remember that, Given a matrix $X \in \mathbb{R}^{m \times n}$ and its SVD decomposition $X = USV^T$ we can prove that an orthogonal base for the space of the columns is given by the first $p$ columns of the matrix $U$, where $p = rank(X)$ is equal to the number of non-zero singular values of $A$.

We will make use of the space of the columns defined by the $U$ matrix and the following Theorem:

**Theorem 1.** *Let's consider $W$ a subspace of $\mathbb{R}^n$ where dim $W = s$ and $\{w_1, \dots, w_s\}$ an orthogonal base of $W$. Given a generic $y \in \mathbb{R}^n$ we have that the projection $y^\perp$ of $y$ onto $W$ has the following form:*

$$y^\perp = \frac{y \cdot w_1}{w_1 \cdot w_1} w_1 + \cdots + \frac{y \cdot w_s}{w_s \cdot w_s} w_s. \tag{1}$$

**Corollary 1.1.** *If $X \in \mathbb{R}^{m \times n}$ is a given matrix with SVD decomposition $X = USV^T$, since the $p = rank(X)$ is the dimension of the space defined by the columns of $X$ and the columns of $U$, $\{u_1, \dots, u_p\}$ are an orthonormal basis for that space, the projection of an $n$-dimensional vector $y$ on this space can be easily computed as:*

$$y^\perp = U(U^T y) \tag{2}$$

Thus, consider a binary classification problem, where we want to classificate if a given digit of dimension $m \times n$ represents the number 3 or the number 4. We will call refer to the class of the number 3 as $C_1$, and to the class of the number 4 as $C_2$. Suppose that $s_1$ is the number of elements in $C_1$, while $s_2$ is the number of elements in $C_2$.
If $X_1 \in \mathbb{R}^{mn \times s_1}$ is the matrix such that its columns are a flatten version of each digit in $C_1$, $X_2 \in \mathbb{R}^{mn \times s_2}$ is the matrix such that its columns are a flatten version of each digit in $C_2$, and consider

$$X_1 = U_1 S_1 V_1^T$$
$$X_2 = U_2 S_2 V_2^T$$

the SVD decomposition of the two matrices.

If $y$ in $\mathbb{R}^{m \times n}$ is a new, unknown digit, we can classify it by first flatten it to a vector of $\mathbb{R}^{mn}$, then we can project it to the spaces of $X_0$ and $X_1$ and call them

$$y_1^{\perp} = U_1(U_1^T y)$$
$$y_2^{\perp} = U_2(U_2^T y)$$

Thus, $y$ will be classified as $C_1$ if $||y - y_1^{\perp}||_2 < ||y - y_2^{\perp}||_2$ and vice versa will be classified as $C_2$ if $||y - y_2^{\perp}||_2 < ||y - y_1^{\perp}||_2$. We want to implement this idea on Python.

1. In the first exercise, we will implement the binary classification algorithm for the digits 3 and 4 of MNIST following the ideas explained above.
   - Load the MNIST dataset contained in `./data/MNIST.mat` with the function `scipy.io.loadmat`. This dataset, which is loaded in the form of a $256 \times 1707$ matrix $X$, contains the flattened version of 1707 $16 \times 16$ grayscale handwritten digits. Moreover, from the same file it is possible to load a vector $I$ of length 1707 such that the $i$-th element of $I$ is the true digit represented by the $i$-th image of $X$.
   - Visualize a bunch of datapoints of $X$ with the function `plt.imshow`.
   - Extract from $X$ those columns that corresponds to digits 3 or 4. Those digits represents the classes $C_1$ and $C_2$ defined above.
   - Split the obtained dataset in training and testing. From now on, we will only consider the training set. The test set will be only used at the end of the exercise to test the algorithm.
   - Create the matrices $X_1$ and $X_2$ defined above from $X$.
   - Compute the SVD decomposition of $X_1$ and $X_2$ with `np.linalg.svd(matrix, full_matrices=False)` and denote the $U$-part of the two decompositions as $U_1$ and $U_2$.
   - Take an unknown digit $y$ from the test set, and compute $y_1^{\perp} = U_1(U_1^T y)$ and $y_2^{\perp} = U_2(U_2^T y)$.
   - Compute the distances $d_1 = ||y - y_1^{\perp}||_2$ and $d_2 = ||y - y_2^{\perp}||_2$ and classify $y$ to $C_1$ if $d_1 < d_2$ and to $C_2$ if $d_2 < d_1$.
   - Repeat the experiment for different values of $y$ in the test set. Compute the misclassification number for this algorithm.
   - Repeat the experiment for different digits other than 3 or 4. There is a relationship between the visual similarity of the digits and the classification error?
   - Comment the obtained results.
2. The extension of this idea to the multiple classification task is trivial. Indeed, if we have more than 2 classes (say, $k$ different classes) $C_1, \dots, C_k$, we just need to repeat the same procedure as before for each matrix $X_1, \dots, X_k$ to obtain the distances $d_1, \dots, d_k$. Then, the new digit $y$ will be classified as $C_i$ if $d_i$ is lower that $d_j$ for each $j = 1, \dots, k$.
   Repeat the exercise above with a 3-digit example. Comment the differences.

## Clustering with PCA

The task for this exercise is to verify the ability of PCA in clustering data by projecting very high-dimensional datapoints to 2 or 3 dimensions. In particular, consider the dataset MNIST provided on Virtuale. This dataset contains images of handwritten digits with dimension $28 \times 28$, together with a number from 0 to 9 representing the label. You are asked to:

- Load the dataset in memory and explore its `head` and `shape` to understand how the informations are placed inside of it;
- Split the dataset into the $X$ matrix of dimension $d \times N$, with $d = 784$ being the dimension of each datum, $N$ is the number of datapoints, and $Y \in \mathbb{R}^N$ containing the corresponding labels;
- Choose a number of digits (for example, 0, 6 and 9) and extract from $X$ and $Y$ the sub-dataset containing only the considered digits. Re-call $X$ and $Y$ those datasets, since the originals are not required anymore;
- Set $N_{train} < N$ and randomly sample a training set with $N_{train}$ datapoints from $X$ (and the corresponding $Y$). Call them $X_{train}$ and $Y_{train}$. Everything else is the test set. Call it $X_{test}$ and $Y_{test}$.
- Implement the algorithms computing the PCA of $X_{train}$ with a fixed $k$. Visualize the results (for $k = 2$) and the position of the centroid of each cluster;
- Compute, for each cluster, the average distance from the centroid. Comment the result;
- Compute, for each cluster, the average distance from the centroid on the test set. Comment the results;
- Define a classification algorithm in this way: given a new observation $x$, compute the distance between $x$ and each cluster centroid. Assign $x$ to the class corresponding the the closer centroid. Compute the accuracy of this algorithm on the test set and compute its accuracy;
- Repeat this experiment for different values of $k$ and different digits. What do you observe?