Computer Science Department, Ben Gurion University of the Negev

Introduction to Data Science - Winter 2022-23

Home assignment 2

Due on Jan 8th.

December 12, 2022

1. **Maximum Likelihood estimation**
   Some background: The geometric distribution gives the probability that the first occurrence of success in k independent $Bernouli(\theta)$ trials with success probability θ (like coin tosses) is achieved at the k-th trial. If the probability of success on each trial is θ, then the probability that the k-th trial (out of k trials) is the first success is given by

$$\mathbb{P}(X = k) = (1 - \theta)^{k-1}\theta.$$

   (a) Let $x_1, x_2, x_3, ..., x_n$ be random i.i.d. samples, which are assumed to be from a $Geometric(\theta)$ distribution. That is, each x_i is the number of Bernouli trials it took to get success, like in the example in the next section. The parameter θ is unknown. Find $\hat{\theta}_{ML}$ the maximum likelihood estimator of θ based on such a random sample (derive a formula for the estimator).

   (b) Suppose that we got the samples: $(x_1, x_2, x_3, x_4, x_5, x_6) = 2, 3, 3, 3, 6, 2$. What is the maximum likelihood estimation for θ?

2. **Unsupervised Learning: clustering/classifying images of handwritten numbers**

   In this question we will classify handwritten digits from the MNIST data set. The data can be downloaded from: `http://yann.lecun.com/exdb/mnist/`
   Reading the data from Python can be obtained using this link `https://www.kaggle.com/hojjatk/read-mnist-dataset` or through the file on the course website.

   We will use the unsupervised learning algorithm K-means to classify the images, therefore, for learning the classifier we will use only the training data set (the one in `train-images-idx3-ubyte.gz`).

The images are black and white, containing 28×28 pixels. Most of the images contain a lot of background, which makes the computations a bit redundant. To make those computations more feasible on your computers, we will first apply a PCA dimensionality reduction to the data.

(a) Use the links above to read the data. As the images $\mathbf{x}_i$ in the data come in the range 0-255, you may divide them by 255 and reduce them by 0.5, to make them between -0.5 and 0.5. No need to submit anything written here.

(b) Write a program to reduce the dimension of the data (using PCA) into a dimension of size parameterized by p (later, we will set it to 40). That is:

- Put all the images in a matrix $X = [x_1, x_2, ..., x_m] \in \mathbb{R}^{784 \times m}$, and compute the so called covariance matrix: $\Theta = \frac{1}{m} X X^T \in \mathbb{R}^{784 \times 784}$. For this you should flatten each image $\mathbf{x}_i$ into a vector.
- Then, compute the eigendecomposition $\Theta = U \Sigma^2 U^T$. Plot the singular values in Σ and see that they are decaying - sort them if not.
  Remark: If you need to sort these eigenvalues - then you also need to sort the corresponding eignevectors, the columns of U, according to the descending eigenvalues. This should in principle be sorted, but may not be, depending on the actual code you're using.
- Define the matrix $U_p \in \mathbb{R}^{784 \times p}$ as the matrix with first p columns of U. E.g., something like: `Up = U[:,0:p]`. Make sure that U is sorted according to the descending singular vectors. Finally, define each image as the vector: $\mathbf{w}_i = U_p^T \mathbf{x}_i \in \mathbb{R}^p$.
- Plot an example of an image $\mathbf{x}_i$ and a reconstructed image from those reduced sized-p vectors: $\tilde{\mathbf{x}}_i = U_p \mathbf{w}_i$. Use $p = 40$ here. For plotting you'll need to reshape $\tilde{\mathbf{x}}_i$ back to be a 2D array.

In this section you need to submit the code and plots.

(c) Write a program for applying the Kmeans clustering algorithm using k clusters. Use random initialization (say, random values in [-0.5,0.5]). You should not use built-in codes.

(d) Using $k = 10$, classify the reduced images ($p = 40$) from the training data set.

(e) See which of the clusters you found corresponds to which digit. Assign a digit to a cluster that you found using the most common label in that cluster - use the train labels to determine that. No need to submit anything written here.

(f) Test your success: for each of the images in the test data, estimate its label using closest centroid (and its cluster's label) and check the percentage of true estimations using the test labels. Report your model's results.

(g) See if the process above is consistent or not with respect to the random initialization. Try this 3 times and report your results.

(h) Try to repeat your experiment using a smaller dimension $p = 12$. Are the results different?

(i) Try initializing each of the Kmeans centroids using the mean of 10 reduced images that you pick from each label. Are the results better now (use $p = 40$ again)?