ECE5258 Pattern Recognition (Fall 2021)
Mini-Project #3 (ver. 1.0)

Dr. Georgios C. Anagnostopoulos

Thursday 24th November, 2022

1 Objectives

The objective of Mini-Project (MP) III is to assess students' understanding in a variety of aspects pertaining to K-Means clustering.

Before putting together your work, carefully review the preparation guidelines (Section 3) and submission instructions (Section 4) that are provided.

2 Assignments

● **Task 1. [20 total points]**
In this task, we will examine some of the theoretical aspects of K-Means.

(a) **[10 points]** Intuitively, the result of K-Means clustering should be invariant to translations, rotations and reflections of the training set. Here, you asked to prove it. If $\mathbf{x} \in \mathbb{R}^D$ is any feature vector, then the transformed feature vector

$$\mathbf{x}' \triangleq \mathbf{H}\mathbf{x} + \mathbf{c} \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{D \times D}$ is an orthogonal matrix and $\mathbf{c} \in \mathbb{R}^D$ is a constant vector, is – potentially – a translated, reflected and rotated version of $\mathbf{x}$. Show that K-Means' average loss function remains invariant, when $\mathcal{D} \triangleq \{\mathbf{x}_n\}_{n=1}^N$ is replaced with the transformed training set $\mathcal{D}' \triangleq \{\mathbf{x}'_n\}_{n=1}^N$ by using the fact that $\mathbf{H}^{-1} = \mathbf{H}^T$ by virtue of its orthogonality.

(b) **[10 points]** Now, consider a feature transformation

$$\mathbf{x}' \triangleq \mathbf{A}\mathbf{x} \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{D \times D}$ is an invertible matrix. Show that transforming the data in this way is equivalent to K-Means clustering, but using Mahalanobis instead of Euclidean distances.

● **Task 2. [20 total points]**
This task pertains to the three, most commonly-used mean initialization methods for K-Means clustering, namely the *Random Partition*, *Forgy* [2] and *K-Means++* [1] methods. The first method randomly assigns the training samples to each cluster, ensures that no cluster is empty and, then, computes the resulting cluster means. Next, the *Forgy* method randomly initializes the means to

training samples without replacement, so that no means coincide. The latter method also initializes the means to distinct training samples, but does so with a probability that is proportional to the square Euclidean distance between the sample and the closest already-initialized mean vector. To complete this task you need to implement all three initialization methods. Also, for this task we will be using the *Three* dataset, whose samples are stored in `three.csv`. The dataset consists of 30 points on the plane that are equally distributed to 3 well-separated clusters.

(a) **[5 points]** For 1,000 initialization runs using the *Random Partition* method for $C = 3$ clusters on the *Three* dataset, construct a histogram with 100 bins that shows the empirical distribution of K-Mean's average loss function achieved via this initialization method.

(b) **[5 points]** Repeat the previous part, but using the *Forgy* method instead.

(c) **[5 points]** Repeat the previous part, but using the *K-Means++* method.

(d) **[5 points]** Produce a plot that superimposes all three histograms and provide the mean average losses for each method. Finally, comment on the initialization quality of these methods.

● **Task 3. [30 total points]**

In order to address the parts shown below, you will have to implement the training (with restarts) and prediction algorithm of K-Means clustering. By restarts, we mean to run K-Means training, each time using a different random initialization of means. Furthermore, part (b) asks you to identify the optimal number of clusters via the average of *silhouette* values of the training samples, which result from clustering the data using a varying number of clusters C. For more information about how these values are computed and how they are interpreted, please consult the Wikipedia article on silhouettes.

(a) **[10 points]** Cluster the *Three* dataset into $C = 3$ clusters using K-Means, which is initialized via the *K-Means++* method. Run the algorithm until convergence and, for each iteration, show the $\log_{10}$ value of the normalized average loss, which is defined as the average loss (at that iteration) divided by the average loss obtained for $C = 1$ (it equals the trace of the data's covariance matrix). Finally, provide a plot that shows how the data were clustered by coloring the data points according to which cluster they were assigned to. Also, show the final position of the cluster means.

(b) **[10 points]** One of the simplest methods of determining a suitable number of clusters for a given dataset is to compute the average silhouette value of the data, after they have been clustered in a particular way (in our case, using K-Means). Here, we will use the average silhouette value to determine the optimal number of clusters for the *Three* dataset. In order to do this, for each choice of $C \in \{1, 2, 3, 4, 5\}$, use the *K-Means++* initialization method, train K-Means using 10 restarts, retain the best clustering results (the ones featuring the lowest average loss value) and compute the data's corresponding average silhouette value. Finally, pick as the optimal clustering (and, hence, optimal C value) the one that yields the maximum average silhouette value. Show a graph of the average silhouette values obtained versus C and comment on your results.

(c) **[10 points]** Now, consider the *Atoll* dataset, whose samples are configured as two distinct clusters: a central cluster of 12 samples and a ring-shaped cluster of 48 samples. Run K-Means using the *K-Means++* initialization method and 10 restarts to cluster the data into $C = 2$ clusters and $C = 5$ clusters respectively. Then, for each of these cases, provide a plot that shows how the data were clustered by coloring the data points according to which cluster they were assigned to and show the final position of the cluster means. Provide pertinent comments on the obtained results.

● **Task 4. [30 total points]**

This task pertains to an application of K-Means clustering for lossy compression of (in our case, grayscale/single-channel) images, which falls under the umbrella of what is referred to as *vector quantization* (VQ). The idea goes as follows: an image to be compressed is partitioned into small, fixed *(height, width)* patches. Each patch is vectorized (*i.e.*, represented as a vector) and the collection of these vectors serves as the training set for K-Means clustering. Choosing a number of clusters C, one first runs K-Means (possibly, with restarts) to cluster these data. Next, each training set vector is replaced with its closest mean vector (this constitutes the *quantization* part). Finally, the vectors are reshaped back into patches and these patches are then used to reconstruct the image. We will be measuring the quality of the reconstructed image by computing the resulting *Peak Signal-to-Noise Ratio* (PSNR); you may want to consult Wikipedia's article on PSNR. In order to address this task, you have to implement this VQ technique.

(a) **[10 points]** Explain how one could use the above VQ scheme for lossy compression of single-channel images and justify why the ratio $\frac{C}{N}$ can be regarded as a rough measure of compression.

(b) **[20 points]** Consider the 390×490 image stored in `vangogh-starry-night.png`[1], which depicts the famous "*The Starry Night*" painting by the Dutch post-impressionist painter Vincent van Gogh. This part calls for using K-Means-based VQ to reconstruct the original image for a patch shape of $(5, 5)$ pixels; this should result in having $N = 7,644$ patches/training samples. Run K-Means with *K-Means++* initialization and 10 restarts for $C \in \{1, 10, 100, 1000\}$ and retain the clustering of the run with the lowest average loss[2]. For each one of these cases, display the reconstructed image and provide a plot that shows the trade-off between the resulting PSNR (measured in dB; reconstruction quality) and the $\log_{10}$ of the rough compression ratio $\frac{C}{N}$ (compression quality). Finally, comment on the results you've obtained.

References

[1] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. URL: https://dl.acm.org/doi/10.5555/1283383.1283494.

[2] Edward W. Forgy. Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[1] This is a slightly cropped and converted to grayscale version of a public domain image.

[2] Keep in mind that, the higher the C value, the longer the training times. For example, for $C = 100$, each run may take several tens of minutes to complete.

3 Preparation Guidelines

Below are some general guidelines that you are asked to follow, when compiling a project report. These guidelines greatly facilitate your work's assessment by a grader and, at the same time, aim at helping you sidestep some major pitfalls that would prevent you from receiving the maximum credit for your work.

- **Task Statements:** Before attempting to address a particular task, ensure that you completely understand what is asked from you to perform and/or to produce. When in doubt, ask your instructional staff for clarifications! Also, make sure you did not omit your response to any of the parts that you have attempted. Finally, make sure that it is crystal clear, which response corresponds to which task/part.

- **Derivations & Proofs:** If you provide handwritten derivations and/or proofs, make sure you use your best handwriting. Each derivation should have a logical and organized flow, so that it is easy to follow and verify.

- **Code & Data:** The code that you author should be as well-organized as possible and amply commented. This is very useful for assessing your work, as well as for you, while you are debugging/or modifying it, or when you have to go back to it in the near future. **Caution:** You are not allowed to use any code that you have not produced without having/obtaining explicit prior permission, in which case the source(s) you have obtained this code from must be clearly indicated via comments inside your code as well in your report. You are deemed to be plagiarizing, if you fail to do so, which may have dire consequences. Finally, if a task asks you to generate data, keep them organized in a separate folder and document, *e.g.*, in a text file, the specifics of how they were generated.

- **Figures, Plots & Tables:** Plots should have their axes labeled and, if featuring several visual elements such as curves or types of points on the same graph, an appropriate legend should be used. Whether figures or tables, each one of these elements should feature a caption with sufficient information on what is being displayed and how were these results obtained (*e.g.*, under what experimental conditions or settings, etc.). You should ask yourself the question: if someone comes across it, will they understand what is being depicted? Apart from a concise description, major, relevant conclusions stemming from the display should also be included in the caption text.

- **Observations, Comments & Conclusions:** When stating observations about a particular result, do not stop at the obvious that anyone can notice (*e.g.*, "*... we see that the curve is increasing.*"). Instead, assess whether the result is expected, either by theory or intuition (*e.g.*, "*... This is as expected, because X is the integral of ...*"), or, if it is unexpected, offer a convincing reasoning behind it (*e.g.*, "*... We expected a decreasing curve ... All points to that I must have not been calculating X correctly ...*"). The latter is more preferable (*i.e.*, expect partial credit) than stopping at the obvious, which happens to be wrong (*i.e.*, do not expect partial credit). Next, descriptions and comments on results should be sufficient. Be concise, but complete. Finally, conclusions that you draw must be well-justified; vacuous conclusions will be swiftly discounted.

4 Submission Instructions

Kindly adhere to the conventions and submission instructions outlined below. Deviations from what is described here may cause unnecessary delays, costly oversights and immense frustrations related to the assessment of your hard work.

First, store all your project deliverables in a folder named `lastname_mpX`, where `lastname` should be your last name and `X` should be the number of the MP, like 1, 2, etc. The folder name should be all lower case. For example, Anagnostopoulos' folder for MP 1 would be named: `anagnostopoulos_mp1`.

Secondly, your `lastname_mpX` folder should have the following contents:

- A signed & dated copy of the Work Origination Certification page in Adobe PDF format. You can either scan such a page after you complete, date and sign it, or do so electronically, as long as your signature is not typed. If this page is missing from your report, your MP work will not be considered for assessment (grading) and will be assigned a default total score of 0/100.

- An Adobe PDF document named **lastname_report.pdf**, where, again, "lastname" should be replaced by your last name in all lower case, *e.g.*, *anagnostopoulos_report.pdf*. This document should contain your entire Mini-Project report as a single document. This will be the document that will be graded. Also, here are some important things to adhere by:

  - Your responses to the project's tasks and parts should be given in their expected sequential order, *i.e.*, task 1 part (a), task 1 part(b), etc., followed by task 2 part (a), task part(b), and so on. If you did not attempt a particular part, list it in its expected order and state that you have not attempted it as your response.
  - For tasks and parts that require you to show analytical work (*e.g.*, a derivation/proof), you are not obliged to typeset it. While it would be nice to do so, such effort may turn out to be quite time-consuming. Instead, you can scan your work into an image, as long as it is legible and well organized with a clear logical flow. When scanning your hand-written work, use a relatively low-resolution (DPI) setting, so your resulting PDF document does not become too big in size, which may prevent you from uploading your work to Canvas. Use a scanner, not a photo taken by a mobile device.

- A folder named `src`, which should contain all your code (*e.g.*, MATLAB or Python scripts, Jupyter Notebooks, etc.) that you authored and used for producing your results and the data sets that you created for this Mini-Project, if applicable. It is best, if you named your scripts according to the task and or task/part pair they produce results for.

- An optional folder named `docs`, in which you can include a MS Word version of your report and other ancillary material connected in one way or another to your Mini-Project report.

Finally, when you are done putting together all required project materials, compress your folder called `lastname_mpX` into a single `ZIP` archive named `lastname_mpX.zip` and upload it to Canvas by the specified deadline.

WORK ORIGINATION CERTIFICATION

By submitting this document, I, ____, the author of this deliverable, certify that

1. I have reviewed and understood Regulation UCF 5.015 of the current version of UCF's Golden Rule Student Handbook available at http://goldenrule.sdes.ucf.edu/docs/goldenrule.pdf, which discusses academic dishonesty (plagiarism, cheating, miscellaneous misconduct, etc.)
2. The content of this Major Project report reflects my personal work and, in cases it is not, the source(s) of the relevant material has/have been appropriately acknowledged after it has been first approved by the course's instructional staff.
3. In preparing and compiling all this report material, I have not collaborated with anyone and I have not received any type of help from anyone but the course's instructional staff.

Signature ____ Date _______________