CSE 6140 / CX 4140 Assignment 4
due Sep 26, 2019 at 11:59pm on Canvas

1. Please upload a *single* PDF named `assignment.pdf` for all of your answers/report containing:
   (a) a typed preamble that contains:
      i. the list of people you worked with people for each question (if applicable),
      ii. the sources you used,
      iii. and if you wish, your impressions about the assignment (what was fun, what was difficult, why...);
   (b) your solutions for all problems, typed (not handwritten).
2. a single zip file named `code.zip` containing your code, README, and results for Programming problem. Please do *not* place your report in the zip file. Your report should be in the same pdf file as answers to the first two problems.
3. Not abiding by the submission instructions will cause you to lose up to all of the points for a problem.
4. If you do not understand the question, please ask on Piazza or come to office hours. Misunderstanding the question is not a valid excuse for losing points.

1 Divide and Conquer (10 pts)

You are given a sorted array $S[1 \dots n]$ with n distinct integers, i.e., $S[i] < S[i+1]$, for all $1 \leq i < n$. Design a divide-and-conquer algorithm to decide whether there exists an index k such that $S[k] = k$. Your algorithm should run in $O(\log n)$ time. Please provide a description of your algorithm and the pseudocode.

2 Greedy and Dynamic Programming (12 pts)

Consider the following game. You are given a sequence of n positive numbers $(a_1, a_2, ..., a_n)$. Initially, they are all colored black. At each move, you choose a black number a_k and color it and its immediate neighbors (if any) red (the immediate neighbors are the elements a_{k-1}, a_{k+1}). You get a_k points for this move. The game ends when all numbers are colored red. The goal is to get as many points as possible.

(a) Describe a greedy algorithm for this problem. Verify that it does not always maximize the number of points by giving a counter-example. (3 pts)

(b) Describe and analyze an efficient dynamic programming algorithm for this problem that runs in O(n) and returns optimal solutions. (9 pts)

3 Programming (28 pts)

Devan has recently earned $10,000 in cash by gambling (lucky him!). He wants to buy a brand new car. However, he doesn't have enough money yet and of course he is wise enough to not take the risk of gambling again. He usually takes the bus home while dreaming of his own car. "My dark gray car, you are the most beautiful car in the world", he imagines.

One day, while day-dreaming, his eyes fell on an advertisement which changed his life:

Do you want money to buy a car? or a home?
Join us today. Tomorrow is too late!
Pool-o-Pale Investment Co. Visit www.pool-o-pale.com.

He visits the website as soon as he gets home and reads the rules and regulations. He finds that he has to invest his money. They will pay his daily interest, a typical banking approach. He then finds a very appealing rule:

The interest rates are known in advance!

For example, tomorrow's interest rate is 3.5 %. This means that tomorrow, the bank will pay Devan 0.035×10000. The interest rate on the day after tomorrow is -2.1%, which means that they will claim 0.021×10000 of his money on that day. Devan gets excited about this: he can invest on the days with positive interest rates only! "That's great!", he thought, feeling that he was closer than ever to buying the car. But then, he goes on to read the next rule:

Every person can join the Pool-o-Pale once!

"What a bad rule!" he whispered disappointedly. This means that he has to join the plan on one given day, and remain so till some later day. Then, he will earn money at the rate equal to the summation of the interest rates on those days. "How can I earn as much money as possible? I wish I knew of an algorithm that finds the best investment period for me", he thinks. That night he slept while driving his dark gray car in his dreams.

Let's help Devan buy a car! Tomorrow morning, he is going to a branch of Pool-o-Pale. The manager will give him a spreadsheet containing the fixed interest rates from now until some days later. He has less than ten seconds to decide the interval he is going to invest within. You are going to help him find an efficient algorithm to accomplish this. You can, because you know how to design and analyze efficient algorithms!

Suppose the manager will give him a text file containing on its first line, n, the total number of days in the plan. Then, at line i he will receive a (positive or negative) real number indicating the daily interest rate, say a_i. If you really want to help him, you have to find the indices j and k such that $\sum_{j \leq i \leq k} a_i$ is maximum. Assume that there exists at least one day where the interest rate is positive (otherwise, there is no reason why Devan should invest). Remember you have only ten seconds.

You, as an algorithm expert, should try and analyze the following proposals:

- A brute-force approach for this problem seems very naive. You can design a faster algorithm. Believe in yourself! Implement a *divide-and-conquer* approach by splitting the array into two halves. The best solution will either be:
  - fully contained in the first half
  - fully contained in the second half
  - such that its start point is in the first half and its end point in the second half

The first two cases are handled recursively. The third one is a linear search.

- Secondly, you will implement a more clever solution by *dynamic programming*: Assume that the days are indexed by the set $I = \{1, ..., n\}$. Let $B(j)$ denote the maximum sum of interest rates that can be obtained if $j \in I$ is Devan's last day of investment. Derive the recurrence relation for $B(j)$.

You will help Devan by proposing and implementing the two aforementioned algorithms (divide-and-conquer and dynamic programming). Your algorithms should take inputs and generate outputs as follows.

Input: The first line contains two numbers. The first one, n, is the number of days. The second one, k, is the number of instances of the problem you should solve. Then, the next k lines contain n comma separated values. The input files are in `data.zip` with each named `<n>.txt`, e.g. 7.txt and have the form:

```
7,3
-1.5,3.4,-3.1,1.7,2.7,-4.8,3.4
-1.2,3.8,-6.1,9.7,2.8,-5.8,1.4
-3.5,6.4,-3.1,1.7,1.7,-1.8,3.2
```

Output: For each algorithm and each input file, you should produce an output file with k lines. Each line has four numbers. The first one is the maximum value of interest rate Devan will receive. The next two numbers are the indices of the the start and end days of optimal investment, and are in the range $[1, n]$. The last value is the running time of the corresponding algorithm in milliseconds. Values are separated by commas, and non-integer values are output with two decimal digits.

```
4.78,2,6,1554.21
...
...
```

Your should submit the output for both algorithms corresponding to the input files provided. Your outputs should be named as

`<GTusername>_output_<algorithm>_<n>.txt`

where `<algorithm>` should be either `dc` (for divide and conquer) or `dp` (for dynamic programming). Put all output files in a folder named `output`.

Sample data for debugging: We provide a sample input file with $(n = 10, k = 10)$ in `10.txt`, and the corresponding sample output file in `drobinson67_output_dc_10.txt`, inside `sample_input_and_output.zip`. If your algorithms are correct, they will have the same values for the first three columns of the sample output file. Note that you should not submit your output file for this sample dataset.

Implementation Instructions: Your implementation can be in either Python or C/C++. If you use Python please use a Python 3.x version [1] (e.g. Python 3.5).

We have provided starter files in C++ and Python, for both divide-and-conquer and dynamic programming. **We will call these programs, so do not change the file names, executable names, or command line arguments**. For C/C++ submissions, we will compile through the Makefile. A starter one is provided.

A penalty may be assessed if your submission fails to compile and/or run. Keep in mind that your submission will be run and graded on the instructors' environment. It is strongly encouraged that your code only uses standard libraries associated with each language–if you have any questions about this, please check with instructors before the deadline.

Deliverables: You should create a zip file for the programming portion of this assignment that includes the following:

[1] This is also timely, as in the real world Python 2 is officially nearing end-of-life: https://python3statement.org

- Source for the two algorithms, well structured and commented.
- A `README` A file providing any specific instructions for running your code. For example, information about interpreter/compiler version and/or external libraries used could be helpful for debugging your submission.
- A folder named `output` containing the corresponding output of the input files provided in. The output should be in the format described in the "Output" section above.

Your report should be part of your `assignment.pdf`. The report should include:

- A description and pseudocode of your divide and conquer algorithm.
- A description and pseudocode of your dynamic programming algorithm.
- Time and space complexity analysis of both algorithms.
- A single graph with two lines that shows how the average running time of your algorithms grows with n. For each input size, n, average the running time over the k instances in the input file for that value of n.
- Observations about your empirical results that tie back into your time and space complexity analysis.
- Discussion on how the two algorithms compare with each other in terms of the complexity and empirical performance.

CS 7641 CSE/ISYE 6740 Homework 1

Le Song

Deadline: A section: Sep. 26 Thursday, 11:55pm
Q section: Oct. 3 Thursday, 11:55pm

- Submit your answers as an electronic copy on Canvas.
- No unapproved extension of deadline is allowed. Zero credit will be assigned for late submissions. Email request for late submission may not be replied.
- For typed answers with LaTeX (recommended) or word processors, extra credits will be given. If you handwrite, try to be clear as much as possible. No credit may be given to unreadable handwriting.
- Explicitly mention your collaborators if any.
- Recommended reading: PRML[1] Section 9.1, 12.1

1 Probability [15 pts]

(a) Stores A, B, and C have 50, 75, and 100 employees and, respectively, 50, 60, and 70 percent of these are women. Resignations are equally likely among all employees, regardless of stores and sex. Suppose an employee resigned, and this was a woman. What is the probability that she has worked in store C? [5 pts]

(b) A laboratory blood test is 95 percent effective in detecting a certain disease when it is, in fact, present. The test also yields a false positive result for 1 percent of the healthy persons tested. That is, if a healthy person is tested then with probability 0.01 the test result will imply he has the disease. If 0.5 percent of the population actually has the disease, what is the probability a person has the disease given that his test result is positive? [5 pts]

[c-d] On the morning of September 31, 1982, the won-lost records of the three leading baseball teams in the western division of the National League of the United States were as follows:

Team	Won	Lost
Atlanta Braves	87	72
San Francisco Giants	86	73
Los Angeles Dodgers	86	73

Each team had 3 games remaining to be played. All 3 of the Giants games were with the Dodgers, and the 3 remaining games of the Braves were against the San Diego Padres. Suppose that the outcomes of all remaining games are independent and each game is equally likely to be won by either participant. If two teams tie for first place, they have a playoff game, which each team has an equal chance of winning.

(c) What is the probability that Atlanta Braves wins the division? [2 pts]

(d) What is the probability to have an additional playoff game? [3 pts]

[1] Christopher M. Bishop, Pattern Recognition and Machine Learning, 2006, Springer.

2 Maximum Likelihood [15 pts]

Suppose we have n i.i.d (independent and identically distributed) data samples from the following probability distribution. This problem asks you to build a log-likelihood function, and find the maximum likelihood estimator of the parameter(s).

(a) Poisson distribution [5 pts]

The Poisson distribution is defined as

$$P(x_i = k) = \frac{\lambda^k e^{-\lambda}}{k!} (k = 0, 1, 2, ...).$$

What is the maximum likelihood estimator of λ?

(b) Multinomial distribution [5 pts]

The probability density function of Multinomial distribution is given by

$$f(x_1, x_2, \ldots, x_k; n, \theta_1, \theta_2, \ldots, \theta_k) = \frac{n!}{x_1! x_2! \cdots x_k!} \prod_{j=1}^{k} \theta_j^{x_j},$$

where $\sum_{j=1}^{k} \theta_j = 1, \sum_{j=1}^{k} x_j = n$. What is the maximum likelihood estimator of $\theta_j, j = 1, \ldots k$?

(c) Gaussian normal distribution [5 pts]

Suppose we have n i.i.d (Independent and Identically Distributed) data samples from a univariate Gaussian normal distribution $\mathcal{N}(\mu, \sigma^2)$, which is given by

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

What is the maximum likelihood estimator of μ and σ^2?

3 Principal Component Analysis [20 pts]

In class, we learned that Principal Component Analysis (PCA) preserves variance as much as possible. We are going to explore another way of deriving it: minimizing reconstruction error.

Consider data points $\mathrm{x}^n (n = 1, ..., N)$ in D-dimensional space. We are going to represent them in $\{\mathrm{u}_1, ..., \mathrm{u}_D\}$ orthonormal basis. That is,

$$\mathrm{x}^n = \sum_{i=1}^{D} \alpha_i^n \mathrm{u}_i = \sum_{i=1}^{D} (\mathrm{x}^{n^T} \mathrm{u}_i) \mathrm{u}_i.$$

Here, α_i^n is the length when x^n is projected onto u_i.

Suppose we want to reduce the dimension from D to $M < D$. Then the data point x^n is approximated by

$$\tilde{\mathrm{x}}^n = \sum_{i=1}^{M} z_i^n \mathrm{u}_i + \sum_{i=M+1}^{D} b_i \mathrm{u}_i.$$

In this representation, the first M directions of u_i are allowed to have different coefficient z_i^n for each data point, while the rest has a constant coefficient b_i. As long as it is the same value for all data points, it does not need to be 0.

Our goal is setting u_i, z_i^n, and b_i for $n = 1, ..., N$ and $i = 1, ..., D$ so as to minimize reconstruction error. That is, we want to minimize the difference between x^n and $\tilde{\mathrm{x}}^n$ over $\{\mathrm{u}_i, z_i^n, b_i\}$:

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathrm{x}^n - \tilde{\mathrm{x}}^n\|^2.$$

(a) What is the assignment of z_j^n for $j = 1, ..., M$ minimizing J? [5 pts]

(b) What is the assignment of b_j for $j = M + 1, ..., D$ minimizing J? [5 pts]

(c) Express optimal $\tilde{\mathbf{x}}^n$ and $\mathbf{x}^n - \tilde{\mathbf{x}}^n$ using your answer for (a) and (b). [2 pts]

(d) What should be the $\mathbf{u}_i$ for $i = 1, ..., D$ to minimize J? [8 pts]

Hint: Use $S = \frac{1}{N} \sum_{n=1}^{N} (\mathrm{x}^n - \bar{\mathrm{x}})(\mathrm{x}^n - \bar{\mathrm{x}})^T$ for sample covariance matrix.

4 Clustering [20 pts]

[a-b] Given N data points $\mathrm{x}^n (n = 1, \ldots, N)$, K-means clustering algorithm groups them into K clusters by minimizing the distortion function over $\{r^{nk}, \mu^k\}$

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r^{nk} \|\mathrm{x}^n - \mu^k\|^2,$$

where $r^{nk} = 1$ if x^n belongs to the k-th cluster and $r^{nk} = 0$ otherwise.

(a) Prove that using the squared Euclidean distance $\|\mathbf{x}^n - \mu^k\|^2$ as the dissimilarity function and minimizing the distortion function, we will have

$$\mu^k = \frac{\sum_n r^{nk} \mathbf{x}_n}{\sum_n r^{nk}}.$$

That is, μ^k is the center of k-th cluster. [5 pts]

(b) Prove that K-means algorithm converges to a local optimum in finite steps. [5 pts]

[c-d] In class, we discussed bottom-up hierarchical clustering. For each iteration, we need to find two clusters $\{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_m\}$ and $\{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_p\}$ with the minimum distance to merge. Some of the most commonly used distance metrics between two clusters are:

- Single linkage: the minimum distance between any pairs of points from the two clusters, i.e.

$$\min_{\substack{i=1,\ldots,m \\ j=1,\ldots,p}} \|\mathrm{x}_i - \mathrm{y}_j\|$$

- Complete linkage: the maximum distance between any parts of points from the two clusters, i.e.

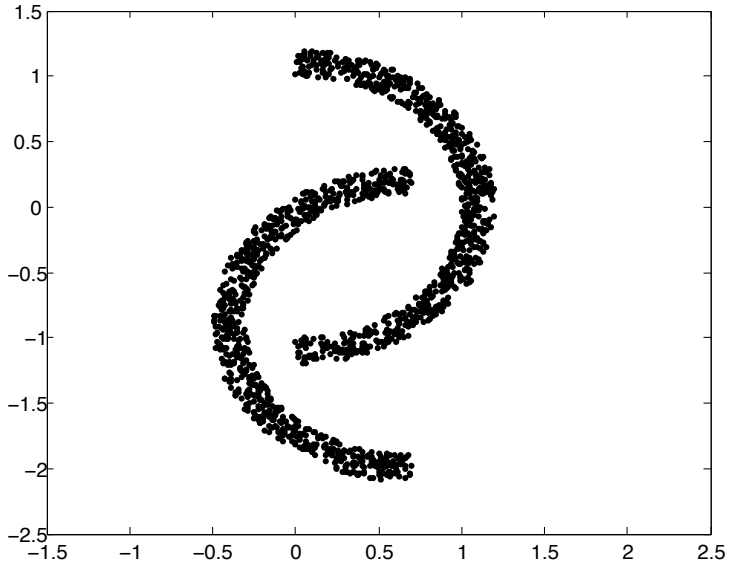
$$\max_{\substack{i=1,\ldots,m \\ j=1,\ldots,p}} \|\mathrm{x}_i - \mathrm{y}_j\|$$

- Average linkage: the average distance between all pair of points from the two clusters, i.e.

$$\frac{1}{mp}\sum_{i=1}^{m}\sum_{j=1}^{p}\|\mathrm{x}_i - \mathrm{y}_j\|$$

(c) When we use the bottom up hierarchical clustering to realize the partition of data, which of the three cluster distance metrics described above would most likely result in clusters most similar to those given by K-means? (Suppose K is a power of 2 in this case). [5 pts]

(d) For the following data (two moons), which of these three distance metrics (if any) would successfully separate the two moons? [5 pts]



5 Programming: Image compression [30 pts]

In this programming assignment, you are going to apply clustering algorithms for image compression. Before starting this assignment, we strongly recommend reading PRML Section 9.1.1, page 428 – 430.

To ease your implementation, we provide a skeleton code containing image processing part. `homework1.m` is designed to read an RGB bitmap image file, then cluster pixels with the given number of clusters K. It shows converted image only using K colors, each of them with the representative color of centroid. To see what it looks like, you are encouraged to run `homework1('beach.bmp', 3)` or `homework1('football.bmp', 2)`, for example.

Your task is implementing the clustering parts with two algorithms: *K-means* and *K-medoids*. We learned and demonstrated K-means in class, so you may start from the sample code we distributed.

The file you need to edit is `mykmeans.m` and `mykmedoids.m`, provided with this homework. In the files, you can see it calls Matlab function `kmeans` initially. Comment this line out, and implement your own in the files. You would expect to see similar result with your implementation of K-means, instead of `kmeans` function in Matlab.

K-medoids

In class, we learned that the basic K-means works in Euclidean space for computing distance between data points as well as for updating centroids by arithmetic mean. Sometimes, however, the dataset may work better with other distance measures. It is sometimes even impossible to compute arithmetic mean if a feature is categorical, e.g, gender or nationality of a person. With K-medoids, you choose a representative data point for each cluster instead of computing their average.

Given N data points $\mathrm{x}^n (n = 1, ..., N)$, K-medoids clustering algorithm groups them into K clusters by minimizing the distortion function $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r^{nk} D(\mathrm{x}^n, \mu^k)$, where $D(\mathrm{x}, \mathrm{y})$ is a distance measure between two vectors x and y in same size (in case of K-means, $D(x, y) = \|\mathrm{x} - \mathrm{y}\|^2$), μ^k is the center of k-th cluster; and $r^{nk} = 1$ if x^n belongs to the k-th cluster and $r^{nk} = 0$ otherwise. In this exercise, we will use the following iterative procedure:

- Initialize the cluster center μ^k, $k = 1, ..., K$.
- Iterate until convergence:
  - Update the cluster assignments for every data point x^n: $r^{nk} = 1$ if $k = \arg\min_j D(\mathrm{x}^n, \mu^j)$, and $r^{nk} = 0$ otherwise.
  - Update the center for each cluster k: choosing another representative if necessary.

There can be many options to implement the procedure; for example, you can try many distance measures in addition to Euclidean distance, and also you can be creative for deciding a better representative of each cluster. We will not restrict these choices in this assignment. You are encouraged to try many distance measures as well as way of choosing representatives.

Formatting instruction

Both `mykmeans.m` and `mykmedoids.m` take input and output format as follows. You should not alter this definition, otherwise your submission will print an error, which leads to zero credit.

Input

- `pixels`: the input image representation. Each row contains one data point (pixel). For image dataset, it contains 3 columns, each column corresponding to Red, Green, and Blue component. Each component has an integer value between 0 and 255.
- `K`: the number of desired clusters. Too high value of K may result in empty cluster error. Then, you need to reduce it.

Output

- `class`: cluster assignment of each data point in pixels. The assignment should be 1, 2, 3, etc. For $K = 5$, for example, each cell of class should be either 1, 2, 3, 4, or 5. The output should be a column vector with `size(pixels, 1)` elements.
- `centroid`: location of K centroids (or representatives) in your result. With images, each centroid corresponds to the representative color of each cluster. The output should be a matrix with K rows and 3 columns. The range of values should be [0, 255], possibly floating point numbers.

Hand-in

Both of your code and report will be evaluated. Upload `mykmeans.m` and `mykmedoids.m` files with your implementation. In your report, answer to the following questions:

1. Within the K-medoids framework, you have several choices for detailed implementation. Explain how you designed and implemented details of your K-medoids algorithm, including (but not limited to) how you chose representatives of each cluster, what distance measures you tried and chose one, or when you stopped iteration.

2. Attach a picture of your own. We recommend size of 320×240 or smaller.

3. Run your K-medoids implementation with the picture you chose above, with several different K. (e.g, small values like 2 or 3, large values like 16 or 32) What did you observe with different K? How long does it take to converge for each K?

4. Run your K-medoids implementation with different initial centroids/representatives. Does it affect final result? Do you see same or different result for each trial with different initial assignments? (We usually randomize initial location of centroids in general. To answer this question, an intentional poor assignment may be useful.)

5. Repeat question 2 and 3 with K-means. Do you see significant difference between K-medoids and K-means, in terms of output quality, robustness, or running time?

Note

- You may see some error message about empty clusters even with Matlab implementation, when you use too large K. Your implementation should treat this exception as well. That is, do not terminate even if you have an empty cluster, but use smaller number of clusters in that case.
- We will grade using test pictures which are not provided. We recommend you to test your code with several different pictures so that you can detect some problems that might happen occasionally.
- If we detect copy from any other student's code or from the web, you will not be eligible for any credit for the entire homework, not just for the programming part. Also, directly calling Matlab function `kmeans` or other clustering functions is not allowed.