

PROGRAMMING ASSIGNMENT 5: UNSUPERVISED LEARNING

10-301/10-601 Introduction to Machine Learning (Summer 2023)

<https://www.cs.cmu.edu/~hchai2/courses/10601/>

OUT: Thursday, July 13th

DUE: Thursday, July 20 at 11:59 PM

TAs: Alex, Andrew, Sofia, Tara, Markov, Neural the Narwhal

Summary In this assignment, you will develop cover the K-Means algorithm along with its variant, K-Means++. In the programming component, you will implement an end-to-end system for both unsupervised learning algorithms. Using the MNIST dataset, you will compare the performance of the these algorithms for different values of K.

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., “Jane explained to me what is asked in Question 2.1”). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the Academic Integrity Section on the course site for more information: <https://www.cs.cmu.edu/~hchai2/courses/10601/>
- **Late Submission Policy:** See the late submission policy here: <https://www.cs.cmu.edu/~hchai2/courses/10601/>
- **Submitting your work:**
 - **Programming:** You will submit your code for programming questions on the homework to Gradescope (<https://gradescope.com>). After uploading your code, our grading scripts will autograde your assignment by running your program on a virtual machine (VM). **You are only permitted to use the Python Standard Library modules and numpy.** When you are developing, check that the version number of the programming language environment (e.g. Python 3.9.12) and versions of permitted libraries (numpy 1.23.0) match those used on Gradescope. You have a **total of 10 Gradescope programming submissions**. Use them wisely. In order to not waste code submissions, we recommend debugging your implementation on your local machine (or the linux servers) and making sure your code is running correctly first before any Gradescope coding submission.
 - **Written:** For written problems such as short answer, multiple choice, derivations, proofs, or plots, please use the provided template. You must typeset your submission using L^AT_EX. If your submission is misaligned with the template, there will be a **2% penalty** (e.g., if the homework is out of 100 points, 2 points will be deducted from your final score). Each derivation/proof should be completed in the boxes provided. Do not move or resize any of the answer boxes. If you do

not follow the template, your assignment may not be graded correctly by our AI assisted grader.

For multiple choice or select all that apply questions, shade in the box or circle in the template document corresponding to the correct answer(s) for each of the questions. For L^AT_EX users, replace `\choice` with `\CorrectChoice` to obtain a shaded box/circle, and don't change anything else.

Instructions for Specific Problem Types

For “Select One” questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

- ☒ Henry Chai
- ☐ Marie Curie
- ☐ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

- ☒ Henry Chai
- ☐ Marie Curie
- ☒ Noam Chomsky

For “Select all that apply” questions, please fill in all appropriate squares completely:

Select all that apply: Which are scientists?

- ☒ Stephen Hawking
- ☒ Albert Einstein
- ☒ Isaac Newton
- ☐ I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are scientists?

- ☒ Stephen Hawking
- ☒ Albert Einstein
- ☒ Isaac Newton
- ☒ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

10-601

10-~~6~~301

Written Questions (2 points)

1 PCA (2 points)

For this section, refer to the PCA demo linked [here](#). In this demonstration, we have performed PCA for you on a [simple four-feature dataset](#). The questions below have also been added to the colab notebook linked for ease of access. Run the code in the notebook, then answer the questions based on the results.

- (1 point) Do you see any special relationships between any of the features? In particular, take a look at the `petal_length` feature. How would you describe its association with each of the **other features**? Select the correct statement with appropriate justification.
 - ☐ The features are highly correlated: we observe linearly proportional relationships where increases in `petal_length` often correspond to increases in another feature
 - ☐ The features are highly correlated: we observe that the color classes can be separated with decision boundaries along the `petal_length` axis.
 - ☐ The features are uncorrelated: we observe random noise as if the features were generated from independent distributions
 - ☐ The features are uncorrelated: we observe the “default $y = x$ ” relationship between features
- (1 point) If we wanted to find k principal components such that we preserve **at least** 95% of the variance in the data, what would be the value of k ? Hint: it is helpful here to look at the cumulative variance in the first k components, which we have calculated for you.

k

2 Empirical Questions (19 points)

The following questions should be completed after you work through the programming portion of this assignment. **For any plotting questions, you must title your graph, label your axes and provide units (if applicable), and provide a legend (if applicable) in order to receive full credit.**

Please submit computer-generated plots for all parts. We have provided you with the code to generate these plots in the handout.

1. K-means

The following questions should be completed after you work through the programming portion of this assignment. Use `mnist_train.csv` for the following questions.

- (a) (3 points) Run the K-Means algorithm with $K=2$. Include plots below of the cluster centers returned by the K-Means algorithm.

Your Answer

- (b) (3 points) Run the K-Means algorithm with $K=5$. Include plots below of the cluster centers returned by the K-Means algorithm.

Your Answer

- (c) (3 points) Run the K-Means algorithm with $K=10$. Include plots below of the cluster centers returned by the K-Means algorithm.

Your Answer

- (d) (2 points) Compare the performance of the three values of K and discuss which one is optimal and why.

Your Answer

- (e) (3 points) Run the K-Means++ algorithm with $K=5$. Include plots below of the cluster centers returned by the K-Means++ algorithm.

Your Answer

- (f) (3 points) Run the K-Means++ algorithm with $K=10$. Include plots below of the cluster centers returned by the K-Means++ algorithm.

Your Answer

- (g) (2 points) Compare the performance of the K-Means ++ algorithm with the K-Means algorithm using the loss values of both with $K=10$ and explain which algorithm performs better than the other and why.

Your Answer

3 Collaboration Questions

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found [here](#).

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details.
2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details.
3. Did you find or come across code that implements any part of this assignment? If so, include full details.

Your Answer

Programming (41 points)

4 The Task

Our goal in this assignment is to implement a K-Means algorithm to cluster images in the MNIST dataset.

5 The Datasets

Dataset We will be using a **subset** of the Modified National Institute of Standards and Technology (MNIST) database. This data includes images of all 10 handwritten digits; our subset will include 3000 of the available images in the dataset. We will evaluate your code on this subset of 3000 samples.

File Format The dataset consists of two csv files. One of these file uses a subset of the 3000 samples given to you. The digits in this subset are restricted to only 0 and 1. Each row contains 784 columns separated by commas. These represent the pixel values of the image.

6 Model Definition

In this assignment you will implement the K-means function to perform K-means clustering on the MNIST dataset. You will also implement the K-Means variant, K-Means++ on the same dataset. The input to your dataset will be N 784-dimensional data points and your output will be the cluster assignments of each of the data points in your training input.

Your code should be able to run with K-Means or K-Means++ invariably on any dataset we pass in according to the algorithm flag passed into the command line when your code is run. Furthermore, you should be able to assign cluster centers for any positive nonzero value of K which is passed in.

For K-Means, you will be asked to initialize the cluster centers to K random points. It is critical that in order for your code to run successfully, you do not change the random seed defined at the top of the file `kmeans.py`. For K-Means++, you must randomly assign your first cluster center, and then use the definition of the K-Means++ algorithm to define the subsequent K-1 cluster centers.

6.1 Command Line Arguments

The autograder runs and evaluates the output from the files generated, using the following command:

```
$ python3 kmeans.py [args...]
```

Where above `[args...]` is a placeholder for four command-line arguments: `<train_input>` `<K>` `<algorithm>` `<train_output>`. These arguments are described in detail below:

1. `<train_input>`: path to the training input `.csv` file (see Section 5)
2. `<K>`: integer specifying the number of cluster centers which should be initialized.
3. `<algorithm>`: integer taking value 0 or 1 that specifies whether to use K-MEANS or K-MEANS++—that is, if `algorithm==0` use K-Means, and if `algorithm==1` use K-Means++ initialization.
4. `<train_output>`: cluster assignments for all data points in `train_input`.

As an example, the following command line would run your program with `K=2` on the small data provided in the handout using K-Means.

```
python3 kmeans.py mnist_small_train.csv 2 0 small_train_out.txt
```

The command line arguments are parsed for you in `kmeans.py` using the Python builtin `argparse` package.

6.2 Sample Outputs

In the handout file we have also provided you with sample outputs for the small dataset. These files correspond to the expected output for the small dataset with $K=2$ and $K=5$ as well as `algorithm=0` and `algorithm=1`. Although you will not be required to output the final cluster centers after your k-means algorithm terminates, we have included output files containing the final cluster centers for the four settings described above to help you debug your code.

6.3 Gradescope Submission

You should only submit your `kmeans.py` file to Gradescope. Please do not use any other file name for your implementation. This will cause problems for the autograder to correctly detect and run your code.

Note: For this assignment, you may make up to **10** submissions to Gradescope before the deadline, but only your last submission will be graded.