CS 446 MJT — Homework 5

junzhew3 haoxuans8

Version 1

Instructions.

- Homework is due **Tuesday, April 16, at 11:59pm**; no late homework accepted.
- Everyone must submit individually at Gradescope under `hw5` and `hw5code`.
- The "written" submission at `hw5` **must be typed**, and submitted in any format Gradescope accepts (to be safe, submit a PDF). You may use LaTeX, markdown, google docs, MS word, whatever you like; but it must be typed!
- When submitting at `hw5`, Gradescope will ask you to mark out boxes around each of your answers; please do this precisely!
- Please make sure your NetID is clear and large on the first page of the homework.
- Your solution **must** be written in your own words. Please see the course webpage for full academic integrity information. Briefly, you may have high-level discussions with at most 3 classmates, whose NetIDs you should place on the first page of your solutions, and you should cite any external reference you use; despite all this, your solution must be written in your own words.
- We reserve the right to reduce the auto-graded score for `hw5code` if we detect funny business (e.g., rather than implementing an algorithm, you keep re-submitting the assignment to the auto-grader, eventually completing a binary search for the answers).
- There are **no regrade requests** on `hw5code`, which is the code auto-grader; however, you can re-submit and re-grade as many times as you like before the deadline! Start early and report any issues on piazza!
- Methods and functions in the template and utility code include docstrings describing the inputs and outputs. The autograder relies on correct implementations of these methods. Follow the docstrings to avoid failing tests.
- In this homework, you cannot use any of the SciPy or SciKit methods. Importing these libraries or their sublibraries, except for those already imported in `hw5_utils.py` will raise an error.

1. **k-means and PCA.**

Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ be n data points in a k-means problem. Let $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ denote the data matrix with rows $\boldsymbol{x}_1^\intercal, \ldots, \boldsymbol{x}_n^\intercal$. For centers $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^d$, define the matrix $\boldsymbol{C}$ with the centers as its rows. Let $\boldsymbol{A} \in \{0,1\}^{n\times k}$ denote an assignment matrix, meaning there is a single 1 per row, and all other entries are zeros. The notation $\mathcal{C}_{d,k} = \mathbb{R}^{k\times d}$ for matrices of k centers in d dimensions, and the notation $\mathcal{A}_{n,k} \subseteq \{0,1\}^{n\times k}$ for all possible assignment matrices. Let $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ be the full singular value decomposition of $\boldsymbol{X}$ with the diagonal of $\boldsymbol{S}$ being sorted in decreasing order. Define $\boldsymbol{V}_l$ as the first l columns of $\boldsymbol{V}$, corresponding to the l largest singular values.

(a) Prove that for any orthonormal matrix $\boldsymbol{M}$,

$$\|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 = \left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\,\boldsymbol{M}\boldsymbol{M}^\intercal\right\|_{\mathrm{F}}^2 + \left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\left(\boldsymbol{I}-\boldsymbol{M}\boldsymbol{M}^\intercal\right)\right\|_{\mathrm{F}}^2.$$

(b) Prove that for any fixed assignment matrix $\boldsymbol{A}$, the row space of the optimal $\boldsymbol{C}$ is a subset of the row space of $\boldsymbol{X}$.

Hint. The matrix $\boldsymbol{V}_r$ is orthonormal, where r denotes rank of $\boldsymbol{X}$.

(c) Prove that $\|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 \geq \sum_{i=k+1}^r s_i^2$, for any assignment matrix and any choice of the k centers, where $(s_i)_i$ is the decreasing sequence of singular values of $\boldsymbol{X}$.

(d) Let $(\boldsymbol{A}_l, \boldsymbol{C}_l) = \arg\min_{\substack{\boldsymbol{A}\in\mathcal{A}_{n,k}\\ \boldsymbol{C}\in\mathcal{C}_{d,k}}} \|\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\intercal - \boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2$. Then

$$\|\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\intercal - \boldsymbol{A}_l\boldsymbol{C}_l\|_{\mathrm{F}}^2 \leq \min_{\substack{\boldsymbol{A}\in\mathcal{A}_{n,k}\\ \boldsymbol{C}\in\mathcal{C}_{d,k}}} \|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 \leq \|\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\intercal - \boldsymbol{A}_l\boldsymbol{C}_l\|_{\mathrm{F}}^2 + \sum_{i=l+1}^r s_i^2.$$

Remark. This means that if PCA down to some dimension doesn't incur too much error, then we can solve k-means there without things changing too much.

Solution. *(Your solution here.)*

(a) Set $\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C} = \boldsymbol{N}$, so $\|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 = \|\boldsymbol{N}\|_{\mathrm{F}}^2 = tr(\boldsymbol{N}\boldsymbol{N}^T)$

$\boldsymbol{M}$ is orthonormal, $\boldsymbol{M}^T\boldsymbol{M} = \boldsymbol{I}$. So:

$$\left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\,\boldsymbol{M}\boldsymbol{M}^\intercal\right\|_{\mathrm{F}}^2 = tr(\boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) = tr(\boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) \tag{1}$$

$$\left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\left(\boldsymbol{I}-\boldsymbol{M}\boldsymbol{M}^\intercal\right)\right\|_{\mathrm{F}}^2 = tr((\boldsymbol{N}-\boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T)(\boldsymbol{N}^T-\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T)) \tag{2}$$

$$= tr(\boldsymbol{N}\boldsymbol{N}^T - \boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T - \boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T + \boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) = tr(\boldsymbol{N}\boldsymbol{N}^T - \boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) \tag{3}$$

So:

$$\left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\,\boldsymbol{M}\boldsymbol{M}^\intercal\right\|_{\mathrm{F}}^2 + \left\|(\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C})\left(\boldsymbol{I}-\boldsymbol{M}\boldsymbol{M}^\intercal\right)\right\|_{\mathrm{F}}^2 = tr(\boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) + tr(\boldsymbol{N}\boldsymbol{N}^T - \boldsymbol{N}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{N}^T) \tag{4}$$

$$= tr(\boldsymbol{N}\boldsymbol{N}^T) = \|\boldsymbol{N}\|_{\mathrm{F}}^2 = \|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 \tag{5}$$

So it is proved.

(b) According to the proof in the kmeans slide:

$$\min_{C \in C_{d,k}} \phi(\boldsymbol{C}; \boldsymbol{A}) = \sum_{j=1}^{k} \min_{\boldsymbol{\mu}_j \in \mathbb{R}^d} \sum_{i=1}^{n} A_{ij} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_j \right\| \tag{6}$$

Taking the gradient with respect to μ_j and setting to 0:

$$\sum_{i=1}^{n} 2A_{i,j} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_j \right) = 0, so : \boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \nVdash \left[A_{ij} = 1 \right]}{\sum_{i=1}^{n} \nVdash \left[A_{ij} = 1 \right]} = \text{mean} \left(\left\{ \boldsymbol{x}_i : A_{ij} = 1 \right\} \right) \tag{7}$$

So the row space $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^d$ of the optimal $\boldsymbol{C}$ is the mean of points in $\boldsymbol{X}$ with common assignment. So for any fixed assignment matrix $\boldsymbol{A}$, the row space of the optimal $\boldsymbol{C}$ is a subset of the row space of $\boldsymbol{X}$.

(c) Since $r(\boldsymbol{A}) \leq \boldsymbol{k}$ and $r(\boldsymbol{C}) \leq \boldsymbol{k}$, $r(\boldsymbol{AC}) \leq \boldsymbol{k}$.

since $\boldsymbol{X} = \boldsymbol{USV}^\top$, We assume rank k approximation to $\boldsymbol{X}$ is given by $\boldsymbol{X}_k = \sum_{i=1}^{k} s_i u_i v_i^\top$

By the triangle inequality with the spectral norm, we can know that if $\boldsymbol{X} = \boldsymbol{X}' + \boldsymbol{X}''$ then $\boldsymbol{s}_1(\boldsymbol{X}) \leq \boldsymbol{s}_1 \left(\boldsymbol{X}' \right) + \boldsymbol{s}_1 \left(\boldsymbol{X}'' \right)$. Suppose $\boldsymbol{X}'_k$ and $\boldsymbol{X}''_k$ respectively denote the rank k approximation to $\boldsymbol{X}'$ and $\boldsymbol{X}''$. So for any $i, j \geq 1$:

$$\begin{aligned}
\boldsymbol{s}_i \left(\boldsymbol{X}' \right) + \boldsymbol{s}_j \left(\boldsymbol{X}'' \right) &= \boldsymbol{s}_1 \left(\boldsymbol{X}' - \boldsymbol{X}'_{i-1} \right) + \boldsymbol{s}_1 \left(\boldsymbol{X}'' - \boldsymbol{X}''_{j-1} \right) \\
&\geq \boldsymbol{s}_1 \left(\boldsymbol{X} - \boldsymbol{X}'_{i-1} - \boldsymbol{X}''_{j-1} \right) \\
&\geq \boldsymbol{s}_1 \left(\boldsymbol{X} - \boldsymbol{X}_{i+j-2} \right) \left(\text{rank} \left(\boldsymbol{X}'_{i-1} + \boldsymbol{X}''_{j-1} \right) \leq \text{rank} \left(\boldsymbol{X}_{i+j-2} \right) \right) \\
&\geq \boldsymbol{s}_{i+j-1}(\boldsymbol{X})
\end{aligned}$$

since $r(\boldsymbol{AC}) \leq \boldsymbol{k}$, $\boldsymbol{s}_{k+1}(\boldsymbol{AC}) = 0$. So we can set $\boldsymbol{X}' = \boldsymbol{X} - \boldsymbol{AC}$ and $\boldsymbol{X}'' = \boldsymbol{AC}$, and for $i \geq 1, j = k+1$: $\boldsymbol{s}_i \left(\boldsymbol{X} - \boldsymbol{AC} \right) \geq \boldsymbol{s}_{k+i}(\boldsymbol{X})$. So:

$$\|\boldsymbol{X} - \boldsymbol{AC}\|_{\mathrm{F}}^2 = \sum_{i=1}^{r} \boldsymbol{s}_i(\boldsymbol{X} - \boldsymbol{AC})^2 \geq \sum_{i=k+1}^{r} \boldsymbol{s}_i(\boldsymbol{X})^2 = \sum_{i=k+1}^{r} s_i^2 \tag{8}$$

So the problem is proved.

(d) Since the row space of the optimal $\boldsymbol{C}$ is a subset of the row space of $\boldsymbol{X}$ and the $\boldsymbol{A}, \boldsymbol{C}$ are optimal,

$\boldsymbol{C V}_l \boldsymbol{V}_l^\top = \boldsymbol{C}$

Since $\boldsymbol{V}_l$ is orthonormal, using the conclusion from problem (a):

$$\begin{aligned}
\min_{\substack{\boldsymbol{A} \in \mathcal{A}_{n,k} \\ \boldsymbol{C} \in \mathcal{C}_{d,k}}} \|\boldsymbol{X} - \boldsymbol{AC}\|_F^2 &= \min_{\substack{\boldsymbol{A} \in \mathcal{A}_{n,k} \\ \boldsymbol{C} \in \mathcal{C}_{d,k}}} \left\| (\boldsymbol{X} - \boldsymbol{AC}) \boldsymbol{V}_l \boldsymbol{V}_l^\top \right\|_F^2 + \min_{\substack{\boldsymbol{A} \in \mathcal{A}_{n,k} \\ \boldsymbol{C} \in \mathcal{C}_{d,k}}} \left\| (\boldsymbol{X} - \boldsymbol{AC}) \left(\boldsymbol{I} - \boldsymbol{V}_l \boldsymbol{V}_l^\top \right) \right\|_F^2 \\
&\geq \min_{\substack{\boldsymbol{A} \in \mathcal{A}_{n,k} \\ \boldsymbol{C} \in \mathcal{C}_{d,k}}} \left\| (\boldsymbol{X} - \boldsymbol{AC}) \boldsymbol{V}_l \boldsymbol{V}_l^\top \right\|_F^2 = \min_{\substack{\boldsymbol{A} \in \mathcal{A}_{n,k} \\ \boldsymbol{C} \in \mathcal{C}_{d,k}}} \left\| \boldsymbol{X V}_l \boldsymbol{V}_l^\top - \boldsymbol{AC V}_l \boldsymbol{V}_l^\top \right\|_F^2 \\
&= \|\boldsymbol{X V}_l \boldsymbol{V}_l^\top - \boldsymbol{A}_l \, \boldsymbol{C}_l\|_F^2
\end{aligned}$$

Moreover:

$$\begin{aligned}
\min_{A\in\mathcal{A}_{\mathcal{A},k}} \|\boldsymbol{X}-\boldsymbol{A}\boldsymbol{C}\|_{\mathrm{F}}^2 &\leq \|\boldsymbol{X}-\boldsymbol{A}_l\boldsymbol{C}_l\|_F^2 \\
&= \left\|\boldsymbol{X}-\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\top+\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\top-\boldsymbol{A}_l\boldsymbol{C}_l\right\|_F^2 \\
&\leq \left\|\boldsymbol{X}-\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\top\right\|_F^2+\left\|\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\top-\boldsymbol{A}_l\boldsymbol{C}_l\right\|_F^2 \\
&= \left\|\boldsymbol{X}\boldsymbol{V}_l\boldsymbol{V}_l^\top-\boldsymbol{A}_l\boldsymbol{C}_l\right\|_F^2+\sum_{i=l+1}^{r} s_i^2
\end{aligned}$$

So the problem is proved.

2. k**-means.**

In this problem you will deal with the Iris dataset by Ronald Fisher. The dataset consists of measurements of various classes of Iris flowers and the goal is to group the points into classes. The dataset provides four features of the flowers. They are sepal length, sepal width, petal length and petal width, respectively, measured in centimeters. You can access these data by calling the method `load_iris_data()`. The function returns two NumPy arrays of shape $(150, 4)$ and $(150, 1)$. The first array contains 150 datapoints with 4 features each and the latter array contains the corresponding label of datapoints. Note that your solution is not allowed to directly invoke `sklearn` or `scipy`.

(a) Implement Lloyd's method inside the `k_means(X, k)` method, which takes as input a NumPy array of shape (n, d) of n datapoints of dimension d and a positive integer k. The method should return a $k \times d$ matrix in which each row corresponds to one of the k centers. For the initialization, use a random (valid) assignment matrix. You don't need to write anything in the hand-in solutions for this part.

(b) Implement the `get_purity_score(X, Y, C)` method. The method takes a data matrix $\boldsymbol{X}$ of shape (n, d), a label array $\boldsymbol{Y}$ of shape $(n, 1)$, and a matrix $\boldsymbol{C}$ of shape (k, d) with rows corresponding to the k centers. The purity score of a clustering is defined as the percentage of data points whose label matches the plurality label of the cluster they are placed in. The function should return a number in range $[0, 1]$, that is the purity score of the clustering induced on $\boldsymbol{X}$ by the centers $\boldsymbol{C}$. Ties can be broken arbitrarily. You don't need to write anything in the hand-in solutions for this part.

(c) Load the Iris dataset using the `load_iris_data()` method and apply your implementation of k-means to it with different values of k. Plot the purity score of the classification as a function of k. You can use the `line_plot(data1, ..., min_k=2, output_file='plot.pdf')` to draw the plot. This method takes as input an array of purity scores and prints the corresponding line plot to a PDF file. The optional argument `min_k` indicates the first label along the x axis and `output_file` indicates the output file location. Describe the behavior of purity score as k increases. At what point does purity reach 1?

(d) In the next three parts, you will use k-means centers to learn features of a dataset. For a data matrix $\boldsymbol{X}$ of shape (n, d) and centers matrix $\boldsymbol{C}$ of shape (k, d), let $\boldsymbol{A}$ be the (n, k) matrix of assignments in the i-th row equals $\boldsymbol{e}_j^\top$ if the j-th center is the closest center to the i-th datapoint. You may break ties arbitrarily. Run the `logistic_regression(X, Y)` method given in `hw5_utils.py` on the rows of $\boldsymbol{A}$. The output of the function is of type `sklearn.linear_model.LogisticRegression` and in particular supports the method `predict_proba(x)` that returns the probability of a given data point `x` having each of the four labels. See the documentation at `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`. Implement the function `classify_using_k_means(X, Y, k)` that takes and input the data matrix $\boldsymbol{X}$ and labels $\boldsymbol{Y}$ and the integer k indicating the number of centers, and returns a function f that takes a point $x \in \mathbb{R}^d$ and returns a label in $\{0, 1, 2, 3\}$. You don't need to write anything in the hand-in solutions for this part.

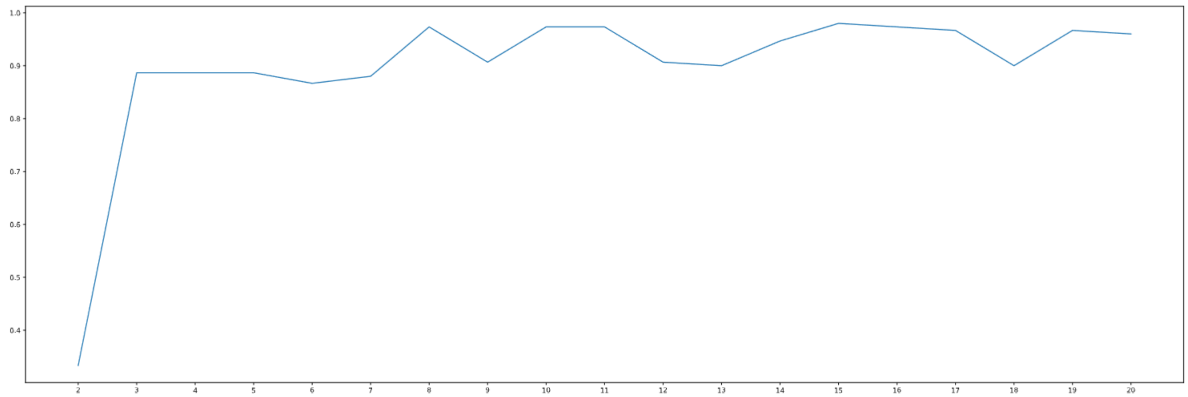
(e) In this part you will repeat the previous problem, except instead of the matrix $\boldsymbol{A}$, you should construct the matrix $\boldsymbol{A}_l$ that has exactly l ones in each row i, that correspond to the l closest centers to the i-th datapoint. Ties can be broken arbitrarily. Implement `classify_using_k_means(X, Y, k, l=1)` as in the previous part, for all positive integer l. You don't need to write anything in the hand-in solutions for this part.

(f) The method `load_iris_data(ratio=0)` takes a value `ratio` in range $[0, 1]$ and returns the matrices $\boldsymbol{X}_{\text{train}}$, $\boldsymbol{X}_{\text{test}}$, $\boldsymbol{Y}_{\text{train}}$ and $\boldsymbol{Y}_{\text{test}}$ where $\boldsymbol{X}_{\text{test}}$, $\boldsymbol{Y}_{\text{test}}$ is a test set with $\lfloor n \times \text{ratio} \rfloor$ points and $\boldsymbol{X}_{\text{train}}$, $\boldsymbol{Y}_{\text{train}}$ is the training dataset and contains the rest the points. Use this method to train a classifiers using `classify_using_k_means(X, Y, k, l)` with training/test ratio of 0.8 and $k \in 2, 3, \ldots, 20$. Plot the training error and test error of the classifier against k once for $l = 1$ and once for $l = 3$, and describe your explanation for the trends in a few sentences. Again, you can use the method `line_plot(data1, ..., min_k=2, output_file='plot.pdf')` to plot the data.

(g) Train a k-means model with 4 centers. Plot the data and group them by the closest center. To draw the plot, you can use the method `scatter_plot_2d_project(X1, ..., output_file='output.pdf', ncol=3)`. This method takes multiple matricies `X1, ...` of size $n_i \times d$ each corresponding to one cluster. It generates $\binom{d}{2}$ plots, one for every pair of dimensions. The output is then saved to the file indicated by `output_file`.

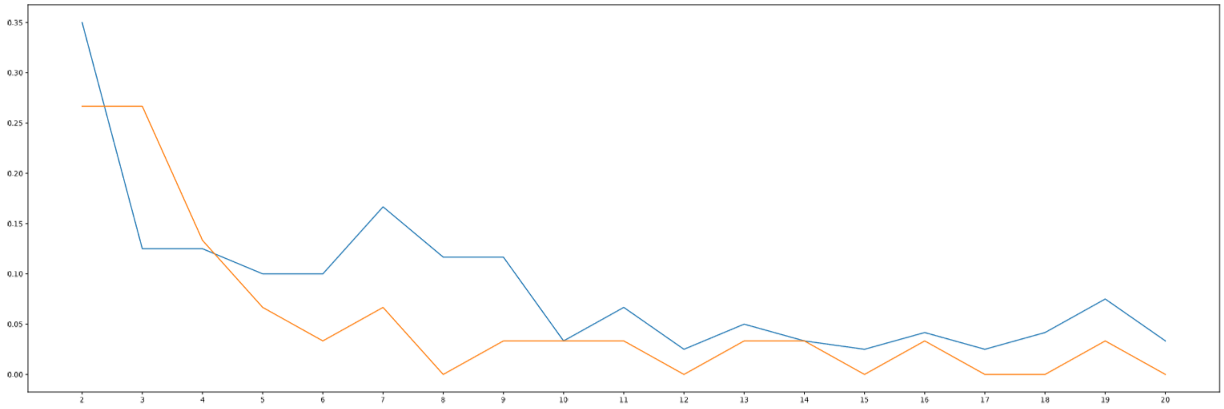
Solution. *(Your solution here.)*

(c)

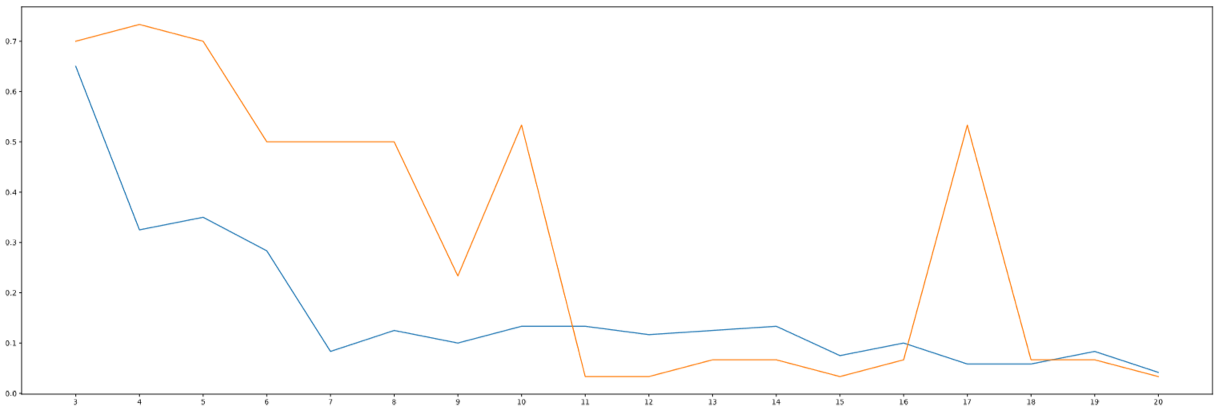


The purity scores fluctuate since the random initialization in the k-means algorithm. Overall, the purity scores show a upward trend as k increases. And at no point the purity reach 1 but the purity scores of some points are really closed to 1.

(f)



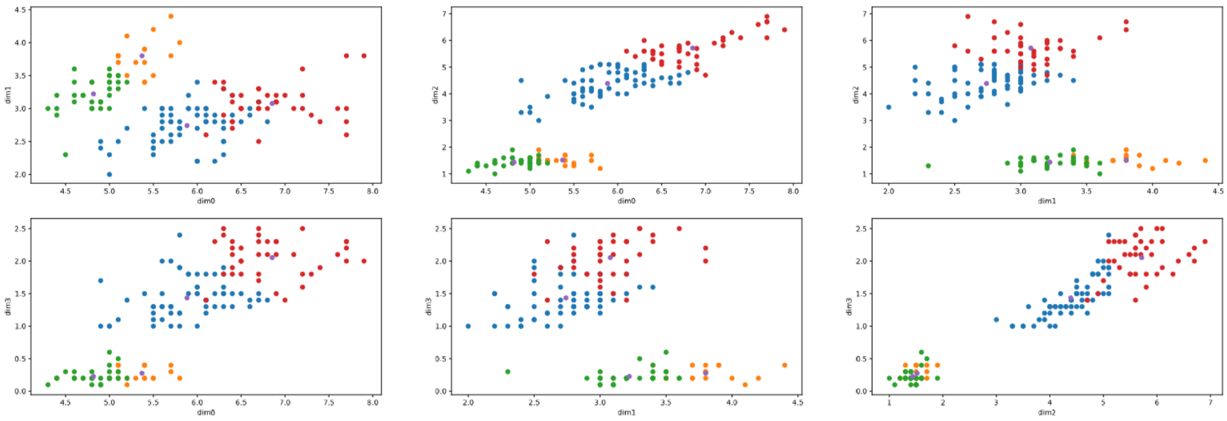
This is the curve of training error and test error of the classier against k for l = 1. The blue line is training error, the orange line is test error.



This is the curve of training error and test error of the classier against k for l = 3. The blue line is training error, the orange line is test error.

All curves show a downward trend as k increases and also fluctuate since the random initialization in the k-means algorithm. And this trend is really like the trend of train error in the 1-NN algorithm as the number of points increase. As the k increases, there are more centers and these centers divide the area into more parts, so the train error and test error will drop.

(g)



3. **E-M and GMMs.**

In this problem you will implement the E-M algorithm to learn a mixture of k Gaussians with diagonal covariances, as detailed in lecture. You will be using the same data as in Problem 2.

(a) Implement the `gmm(X, k)` method. In this method `X` is a data matrix of shape (n, d). The positive integer `k` indicates the number of Gaussian components in the mixture. Your output should be a matrix μ of shape (k, d) where each row corresponds to the mean of one of the Gaussians, a matrix Σ of shape (k, d) matrices where the i-th row corresponds to the diagonal of the covariance matrix of the i-th Gaussians and a k-vector of probability distribution over the k Gaussian components. For initialization, set the weights π to be uniform, the covariances to be diagonal, and the means to be the result of your k-means solution from the previous part after 10 iterations.

(b) Compute and plot the log-likelihood of the model when trained on the data trained by your algorithm with a uniform `pi` against the value of `k` ranging from 2 to 10. You can use the method `line_plot(data1, ..., min_k=2, output_file='plot.pdf')` to plot the data.

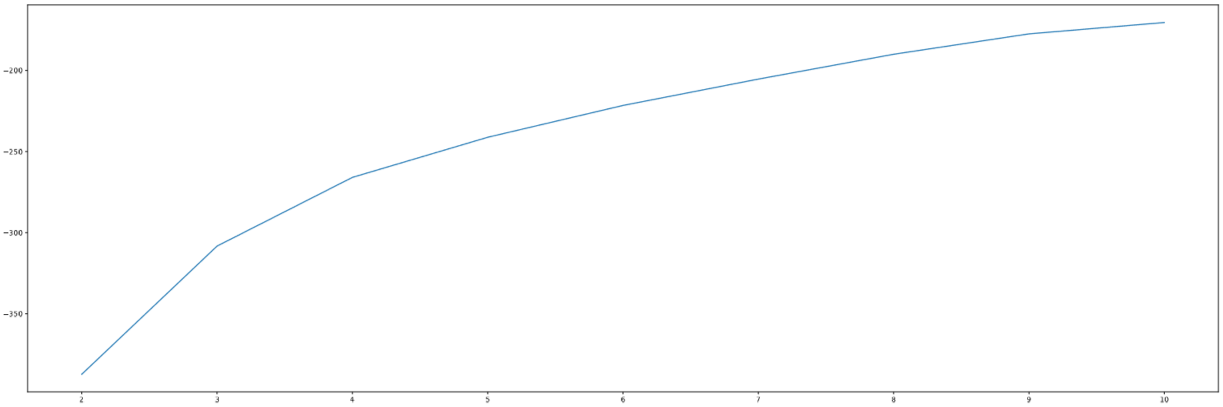
(c) Implement the `gmm_predict(x, mu, covars, weights)` method. In this method, `x` is a $d \times 1$ vector, `mu` is a matrix of shape (k, d), the list `covars` is of length k and consists of $d \times d$ covariance matrices and `weights` is a $k \times 1$ vector that is probability distribution indicating the weights of the k Gaussian components. Your method should return a k-vector that is the probability distribution of the datapoint $\boldsymbol{x}$ having been generated by each of the Gaussian components.

(d) Implement the method `classify_using_gmm(X, Y, k)`. This method takes a data matrix `X` and label vector Y and positive integer `k`, and fits a Gaussian mixture model of k components on this data. Let R be the responsibility matrix of this model. Train a logistic model on the data matrix R and labels Y. You can use the method `logistic_regression` from `hw5_utils.py`.

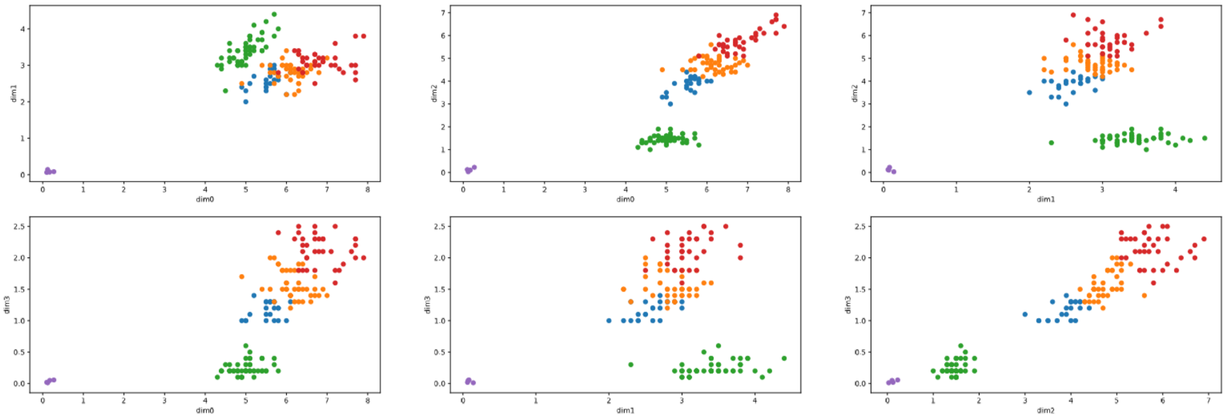
(e) Train a Gaussian mixture model with 4 components. Plot the data and group them by the closest center. To draw the plot, you can use the method `gaussian_plot_2d_project(X1, ..., output_file='output.pdf', ncol=3)`. This method takes multiple matricies `X1, ...` of size $n_i \times d$ each corresponding to one cluster. It generates $\binom{d}{2}$ plots, one for every pair of dimensions. The output is then saved to the file indicated by `output_file`.

Solution. *(Your solution here.)*

(b)



(e)


dim0
dim1
dim2
dim3
0 1 2 3 4 5 6 7 8
0 1 2 3 4
0 1 2 3 4 5 6 7
0.0 0.5 1.0 1.5 2.0 2.5