Homework 4: SVM, Clustering, and Ethics

Introduction

This homework assignment will have you work with SVMs, clustering, and engage with the ethics lecture. We encourage you to read Chapters 5 and 6 of the course textbook.

Please submit the **writeup PDF to the Gradescope assignment 'HW4'**. Remember to assign pages for each question.

Please submit your **LaTeX file and code files to the Gradescope assignment 'HW4 - Supplemental'**.

Problem 1 (Fitting an SVM by hand, 10pts)

For this problem you will solve an SVM by hand, relying on principled rules and SVM properties. For making plots, however, you are allowed to use a computer or other graphical tools.

Consider a dataset with the following 7 data points each with $x \in \mathbb{R}$ and $y \in \{-1, +1\}$:

$$\{(x_i, y_i)\}_{i=1}^{7} = \{(-3, +1), (-2, +1), (-1, -1), (0, +1), (1, -1), (2, +1), (3, +1)\}$$

Consider mapping these points to 2 dimensions using the feature vector $\phi(x) = (x, -\frac{8}{3}x^2 + \frac{2}{3}x^4)$. The hard margin classifier training problem is:

$$\min_{\mathbf{w}, w_0} \frac{1}{2}\|\mathbf{w}\|_2^2$$
$$\text{s.t.} \quad y_i(\mathbf{w}^\top \phi(x_i) + w_0) \geq 1, \ \forall i \in \{1, \ldots, n\}$$

Make sure to follow the logical structure of the questions below when composing your answers, and to justify each step.

1. Plot the transformed training data in $\mathbb{R}^2$ and draw the optimal decision boundary of the max margin classifier. You can determine this by inspection (i.e. by hand, without actually doing any calculations).

2. What is the value of the margin achieved by the optimal decision boundary found in Part 1?

3. Identify a unit vector that is orthogonal to the decision boundary.

4. Considering the discriminant $h(\phi(x); \mathbf{w}, w_0) = \mathbf{w}^\top \phi(x) + w_0$, give an expression for *all possible* $(\mathbf{w}, w_0)$ that define the optimal decision boundary from 1.1. Justify your answer.

   Hint: The boundary is where the discriminant is equal to 0. Use what you know from 1.1 and 1.3 to solve for $\mathbf{w}$ in terms of w_0. (If you solve this problem in this way, then w_0 corresponds to your free parameter to describe the set of all possible $(\mathbf{w}, w_0)$.)

5. Consider now the training problem for this dataset. Using your answers so far, what particular solution to $\mathbf{w}$ will be optimal for the optimization problem?

6. What is the corresponding optimal value of w_0 for the $\mathbf{w}$ found in Part 5 (use your result from Part 4 as guidance)? Substitute in these optimal values and write out the discriminant function $h(\phi(x); \mathbf{w}, w_0)$ in terms of the variable x .

7. Which points could possibly be support vectors of the classifier? Confirm that your solution in Part 6 makes the constraints above tight—that is, met with equality—for these candidate points.

8. Suppose that we had decided to use a different feature mapping $\phi'(x) = (x, -\frac{31}{12}x^2 + \frac{7}{12}x^4)$. Does this feature mapping still admit a separable solution? How does its margin compare to the margin in the previous parts? Based on this, which set of features might you prefer and why?

Solution

1. I drew the points i with $y_i = 1$ in green and points i with $y_i = -1$ in red. The max margin classification boundary is in blue; it is the line $\phi(x) = -1$.
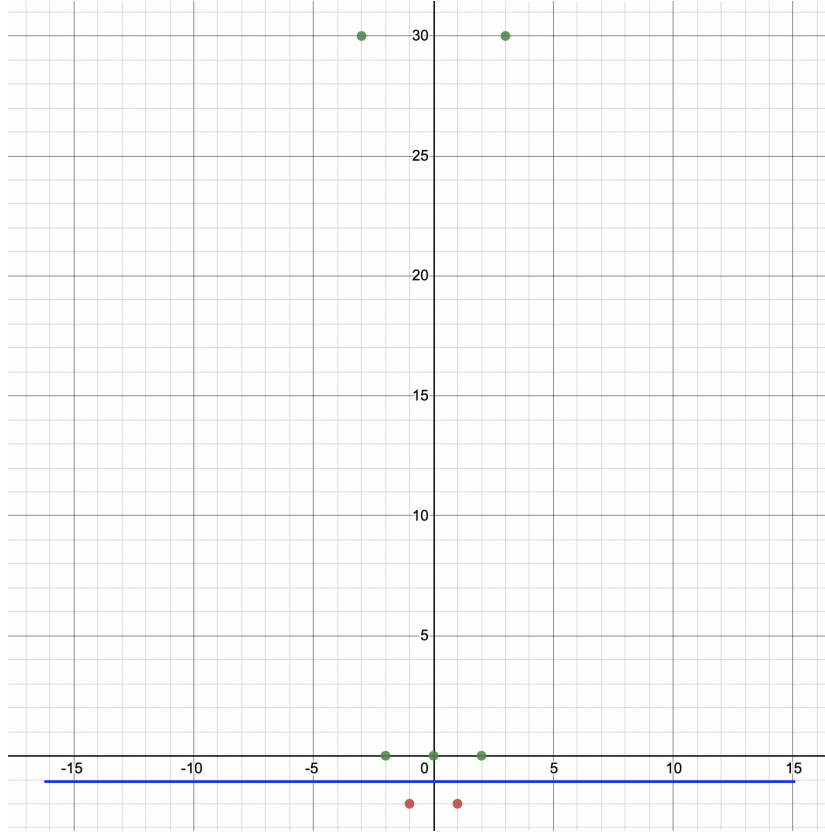
2. As we can see from the plot, the boundary is distance 1 away from the points $(-2, \phi(-2)), (-1, \phi(-1)), (0, \phi(0)), (1, \phi(1))$, and $(2, \phi(2))$, so the value of the margin is 1.

3. The boundary is a horizontal line, so one orthogonal unit vector is $\mathbf{w} = (0, 1)^T$.

4. We need the discriminant to equal zero at the decision boundary. Writing $\mathbf{w} = (w_1, w_2)^T$ Therefore:

$$h((x, -1), \mathbf{w}, w_0) = w_1 x - w_2 + w_0 = 0 \ \ \forall x$$

This can only happen if $w_1 = 0$ since the function has to be constant. Then we get:

$$w_2 = w_0$$

So $\mathbf{w} = (0, w_0)^T$ as a function of w_0.

5. Writing out the constraint for each data point, we need;

$$+1(w_0(-\frac{8}{3}3^2 + \frac{2}{3}3^4) + w_0) \geq 1$$

$$+1(w_0(-\frac{8}{3}2^2 + \frac{2}{3}2^4) + w_0) \geq 1$$

$$-1(w_0(-\frac{8}{3}1^2 + \frac{2}{3}1^4) + w_0) \geq 1$$

$$1(w_0(-\frac{8}{3}0^2 + \frac{2}{3}0^4) + w_0) \geq 1$$

This suffices since $-\frac{8}{3}x^2 + \frac{2}{3}x^4)$ is an even function so I don't need to consider the negative points. Note that at $x = 1$ and $x = 0$, this simplifies to:

$$w_0 \geq 1$$

Then, since we have $||\mathbf{w}||_2^2 = w_1^2 + w_2^2 = w_2^2 = w_0^2$, we can minimize it by choosing:

$$\mathbf{w} = (0, 1)^T$$

6. From the correspondence $w_0 = w_2$, we get $w_0 = 1$, so then:

$$h(\phi(x); \mathbf{w}, w_0) = w_0(-\frac{8}{3}x^2 + \frac{2}{3}x^4) + w_0 = -\frac{8}{3}x^2 + \frac{2}{3}x^4 + 1$$

7. Only the points with $x = 0, x = \pm 1, x = \pm 2$ can be support vectors, since their distance from the margin is 1. To verify that the constraint binds:

$$h(\phi(0); \mathbf{w}, w_0) = -\frac{8}{3}0^2 + \frac{2}{3}0^4 + 1 = 0 + 1 = 1$$

$$-h(\phi(\pm 1); \mathbf{w}, w_0) = -\frac{8}{3}(\pm 1)^2 + \frac{2}{3}(\pm 1)^4 + 1 = -(-\frac{6}{3} + 1) = 1$$

$$h(\phi(\pm 2); \mathbf{w}, w_0) = -\frac{8}{3}(\pm 2)^2 + \frac{2}{3}0^4 + 1 = 0 + 1 = 1$$

Indeed, the constraint binds at all of the potential support vector points.

8. The plot of points looks like this:
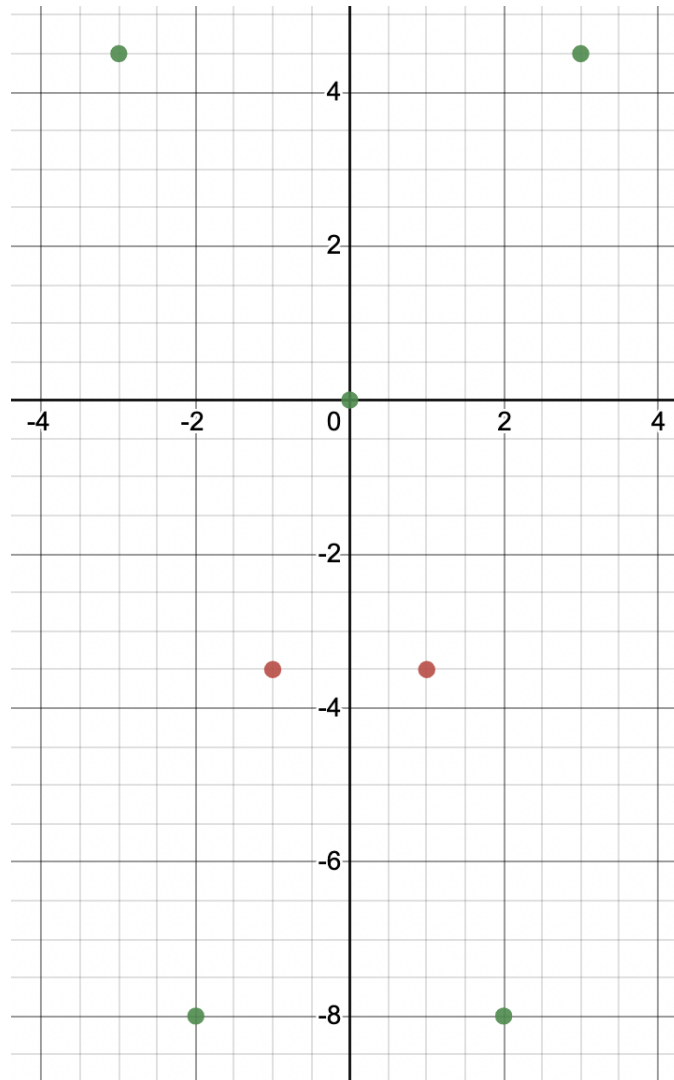
It is apparent that the data is no longer linearly separable under this basis function, since any straight-line boundary will get at least point wrong. The first margin (from part 1) is therefore preferable, since it admits multiple boundaries, including the max-margin classification boundary.

Problem 2 (K-Means and HAC, 20pts)

For this problem you will implement K-Means and HAC from scratch to cluster image data. You may use `numpy` but no third-party ML implementations (eg. `scikit-learn`).

We've provided you with a subset of the MNIST dataset, a collection of handwritten digits used as a benchmark for image recognition (learn more at http://yann.lecun.com/exdb/mnist/). MNIST is widely used in supervised learning, and modern algorithms do very well.

You have been given representations of MNIST images, each of which is a 784×1 greyscale handwritten digit from 0-9. Your job is to implement K-means and HAC on MNIST, and to test whether these relatively simple algorithms can cluster similar-looking images together.

The code in `T4_P2.py` loads the images into your environment into two arrays – `large_dataset`, a 5000x784 array, will be used for K-means, while `small_dataset`, a 300x784 array, will be used for HAC. In your code, you should use the ℓ_2 norm (i.e. Euclidean distance) as your distance metric.

Important: Remember to include all of your plots in your PDF submission!

Checking your algorithms: Instead of an Autograder file, we have provided a similar dataset, `P2_Autograder_Data`, and some visualizations, `HAC_visual` and `KMeans_visual`, for how K-means and HAC perform on this data. Run your K-means (with $K = 10$ and `np.random.seed(2)`) and HAC on this second dataset to confirm your answers against the provided visualizations. Do **not** submit the outputs generated from `P2_Autograder_Data`. Load this data with `data = np.load('P2_Autograder_Data.npy')`.

1. Starting at a random initialization and $K = 10$, plot the K-means objective function (the residual sum of squares) as a function of iterations and verify that it never increases.

2. For $K = 10$ and for 3 random restarts, print the mean image (aka the centroid) for each cluster. There should be 30 total images. Code that creates plots for parts 2, 3, and 4 can be found in `T4_P2.py`.

3. Repeat Part 2, but before running K-means, standardize or center the data such that each pixel has mean 0 and variance 1 (for any pixels with zero variance, simply divide by 1). For $K = 10$ and 3 random restarts, show the mean image (centroid) for each cluster. Again, present the 30 total images in a single plot. Compare to Part 2: How do the centroids visually differ? Why?

4. Implement HAC for min, max, and centroid-based linkages. Fit these models to the `small_dataset`. For each of these 3 linkage criteria, find the mean image for each cluster when using 10 clusters. Display these images (30 total) on a single plot.

   How do the "crispness" of the cluster means and the digits represented compare to mean images for k-means? Why do we only ask you to run HAC once?

   Important Note: For this part ONLY, you may use `scipy`'s `cdist` function to calculate Euclidean distances between every pair of points in two arrays.

5. For each of the HAC linkages, as well as one of the runs of your k-means, make a plot of "Number of images in cluster" (y-axis) v. "Cluster index" (x-axis) reflecting the assignments during the phase of the algorithm when there were $K = 10$ clusters.

   Intuitively, what do these plots tell you about the difference between the clusters produced by the max and min linkage criteria?

   Going back to the previous part: How does this help explain the crispness and blurriness of some of the clusters?

Problem 2 (cont.)

6. For your K-means with $K = 10$ model and HAC min/max/centroid models using 10 clusters on the `small_dataset` images, use the `seaborn` module's `heatmap` function to plot a confusion matrix between each pair of clustering methods. This will produce 6 matrices, one per pair of methods. The cell at the ith row, jth column of your confusion matrix is the number of times that an image with the cluster label j of one method has cluster i in the second method. Which HAC is closest to k-means? Why might that be?

7. Suppose instead of comparing the different clustering methods to each other, we had decided to compute confusions of each clustering method to the *true* digit labels (you do *not* have to actually compute this). Do you think how well the clustering match the true digits is reasonable evaluation metric for the clustering? Explain why or why not.
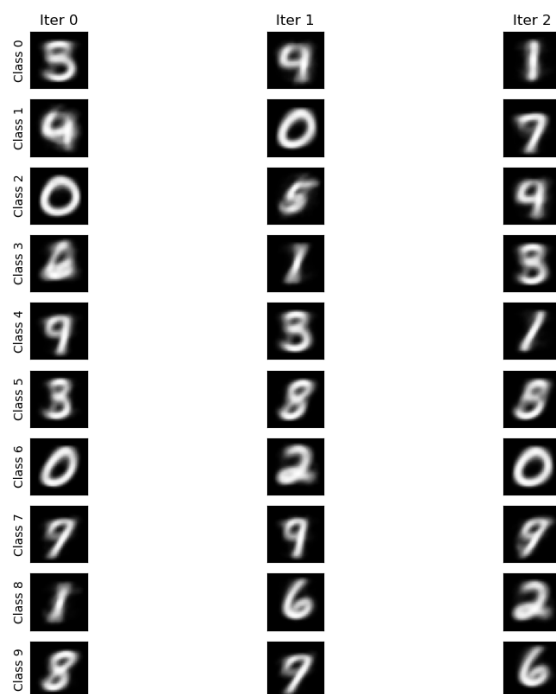
Solution

1. Here is the objective function (the loss) plotted through 10 iterations (close to convergence). It decreases with every iteration!
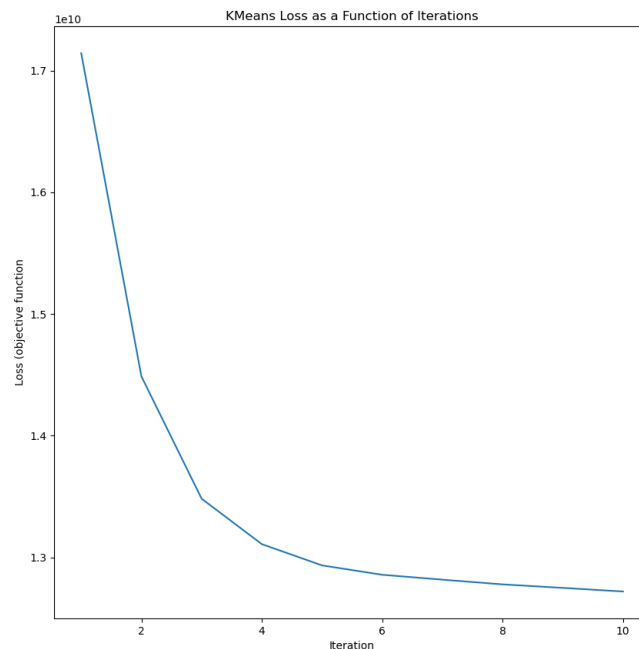
2. Here are the images. Clearly the KMeans algorithm picks up on some real-world features that differentiate the digits but it is far from perfect:

Class mean images across random restarts

3. The algorithm actually does worse on a standardized data set, and the mean images are less similar to actual digits. A part of this may be expected from standardization but still, the centroids are closer to each other than in the unstandardized version.

Class mean images across random restarts (standardized data)

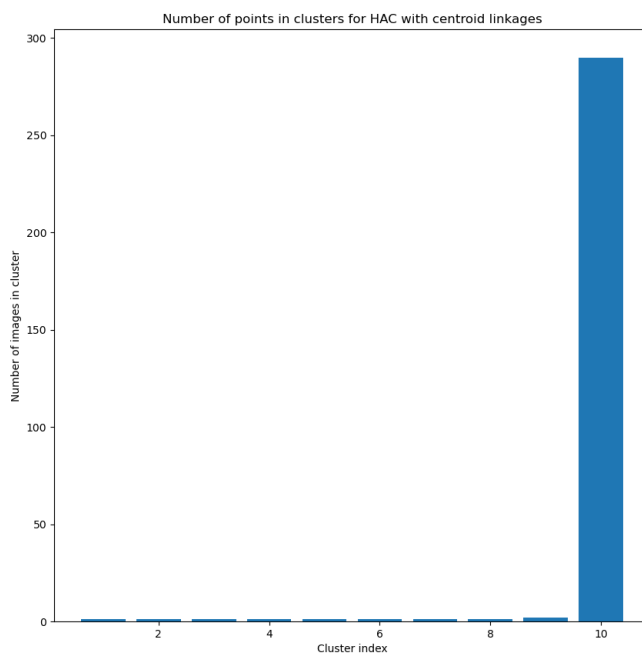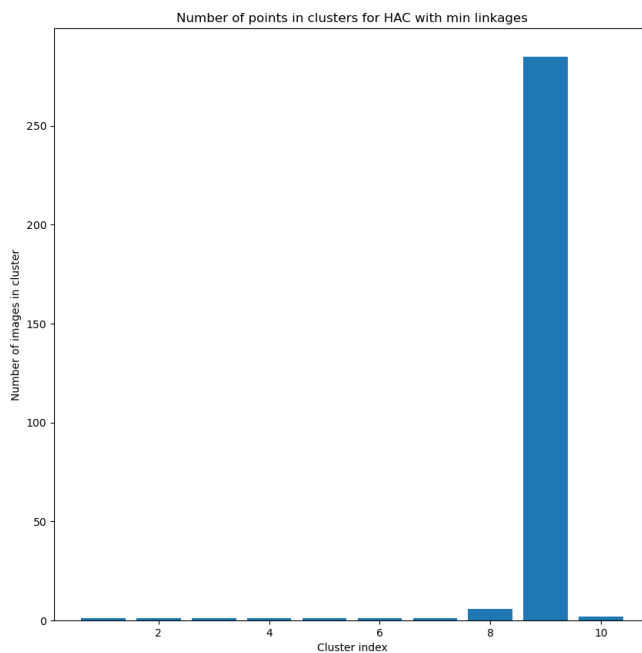4. Here are the HAC means for different linkage choices:

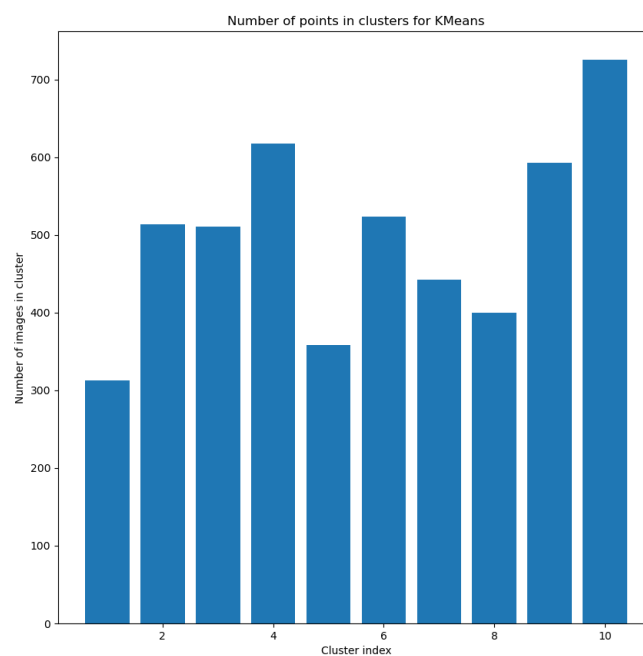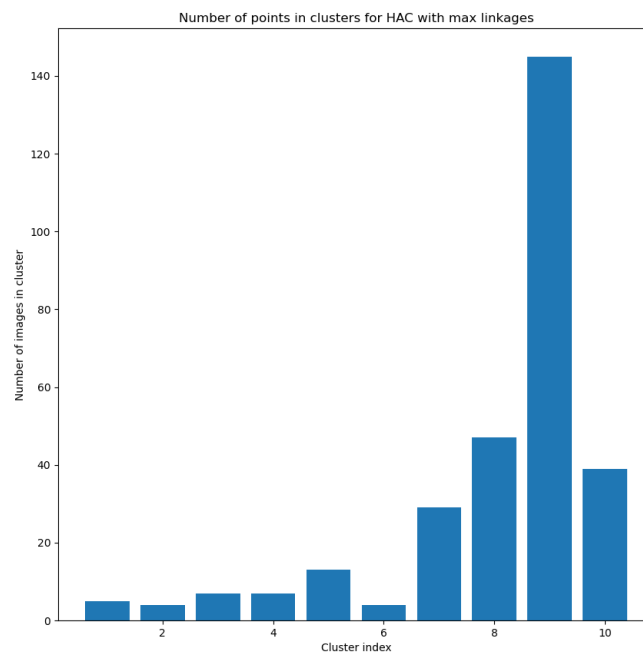HAC mean images with max, min, and centroid linkages
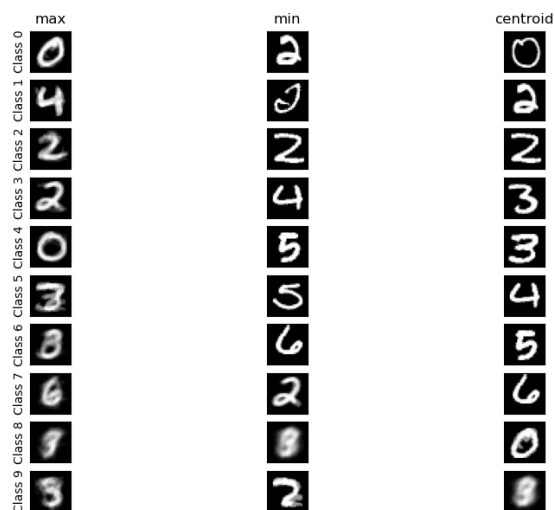
Max-linkage centroids are quite fuzzy, but centroid-linkage improves on this significantly and is the most similar to the runs of KMeans. Min-linkage achieves even greater crispness, though it is not any

better at identifiying the digits.

5. Here are all the plots of the numbers of images in a cluster:

Number of points in clusters for HAC with min linkages

Number of points in clusters for HAC with centroid linkages

Number of points in clusters for HAC with max linkages


Number of points in clusters for KMeans

My runs of minimum- and centroid- linkage HAC ended up puttting almost all points into the same cluster. Max-linkage did slightly better, but there are still large discrepancies between the numbers in

each cluster. In comparison, KMeans assigns images to clustres much more evenly. This explains some of the variation in crispiness: large clusters are likely to be fuuzy since they contain images of several different digits, so their means will be blurry.

6. Here are the 6 heatmaps:
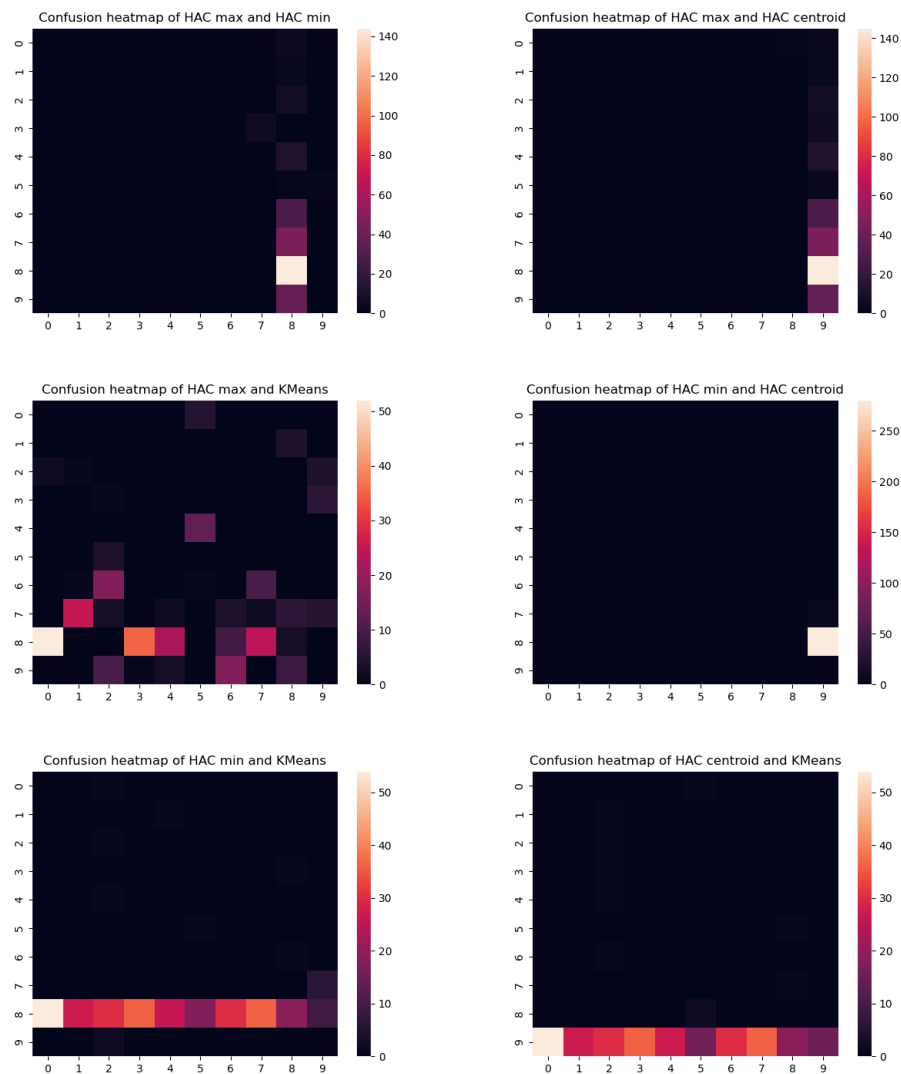
From the heatmaps, HAC with max linkage appears to be the closest to KMeans. This might be because using maximum linkage enforces roughly spherical (high-dimensional spheres) clusters which are also likely to happen under KMeans.

7. Yes, I think the true digit labels are a reasonable metric for a clustering algorithm. This is because other features have been relatively standardized, i.e. all digits are written in white on a black background. Therefore, the 10 digit labels provide a "natural" clustering of the images into categories. We would like a successful clustering algorithm to output clusters that correspond to particular digits.

Problem 3 (Ethics Assignment, 5pts)

Select a real-life outcome in Artificial Intelligence or Machine Learning that you believe is morally wrong. You can select your own outcome from the news or select one of the outcomes in the options below:

- COMPAS, a case management tool predicting recidivism that flagged "blacks are almost twice as likely as whites to be labeled a higher risk but not actually re-offend" (Angwin 2016).

- An NLP algorithm filled in the inference "Man is to ____ as woman is to ____" with "Man is to computer programmer as woman is to homemaker" (Bolukbasi et al, 2016).

- http://www.survivalofthebestfit.com/game: a game that exemplifies algorithmic bias in resume screening

- IBM Diversity in faces: insufficient training data for darker-skinned faces

- Other Unfair Algorithms: Algorithms of Oppression (a really good book with tons of examples), VI-SPDAT, Allegheny Family Screening Tool

Draw a causal chain that resulted in this outcome and circle the choice points that were the largest contributors to the outcome. At each morally relevant choice point, write two alternative decisions that could have prevented the outcome.

Solution

Here is an image of a causal chain: the main choice points are circled in red, and each has an arrow pointing to two suggestions that could have been pursued instead. I chose Survival of The Best Fit, the game about algorithmic bias in hiring.
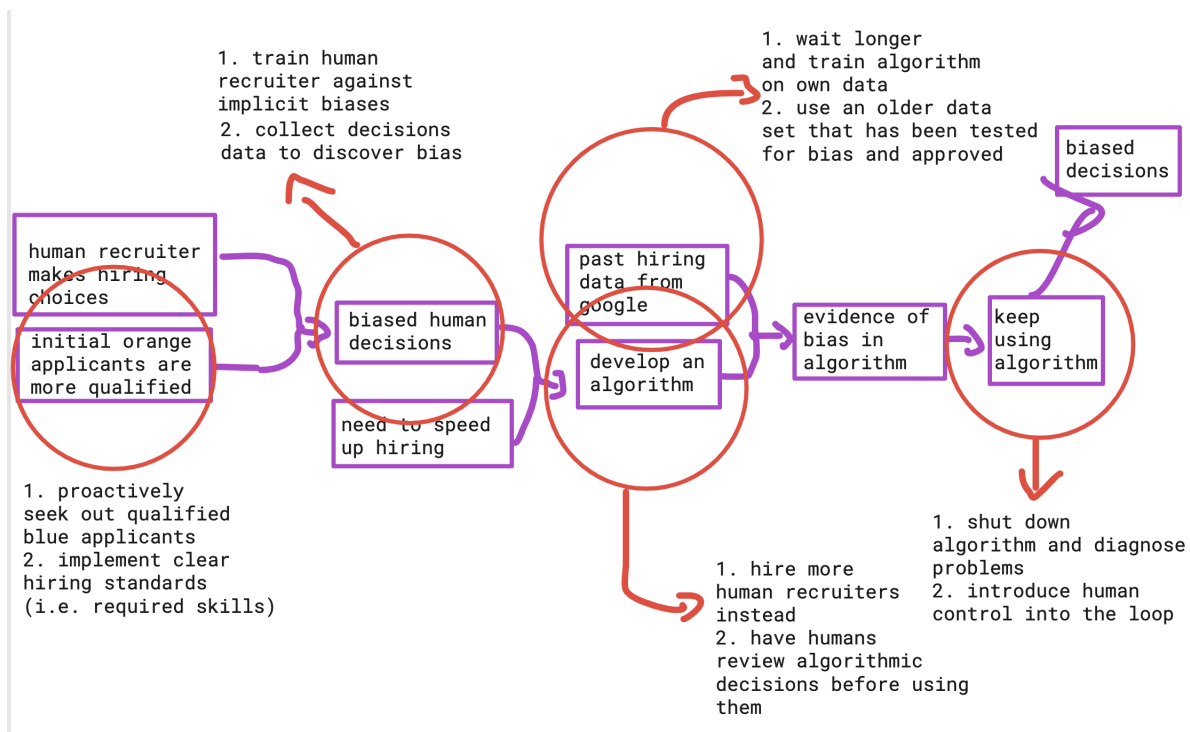
Name

Collaborators and Resources

Whom did you work with, and did you use any resources beyond cs181-textbook and your notes?
No collaborators or outside resources
Did you attend office hours for help with this homework? I did not

Calibration

Approximately how long did this homework take you to complete (in hours)? 20+ hours for this one