CS181-S22

Assignment #4
Due: 11:59pm EST, March 25, 2022

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 $\boldsymbol{\phi}(x) = (x, -\frac{8}{3}x^2 + \frac{2}{3}x^4)$. The hard margin classifier training problem is:

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

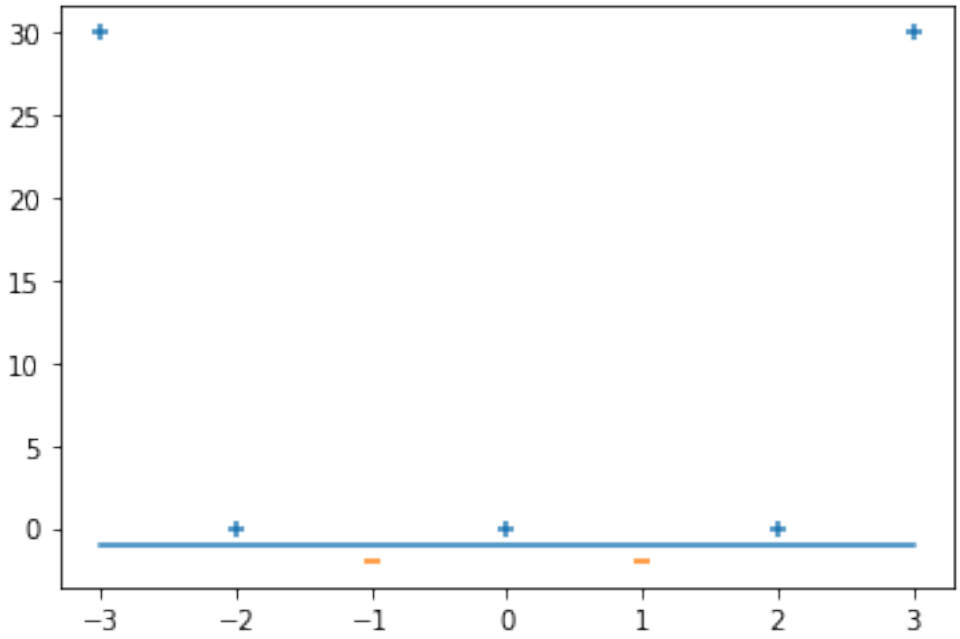
Make sure to follow the logical structure of the questions below when composing your answers, and to justify each step.

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 $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0) = \mathbf{w}^\top \boldsymbol{\phi}(x) + w_0$, 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 $h(\boldsymbol{\phi}(x); \mathbf{w}, w_0)$ 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 $\boldsymbol{\phi}'(x) = (x, -\frac{31}{12}x^2 + \frac{7}{12}x^4)$. 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. The optimal boundary is shown below:



2. The value of the margin is 1, since $x_2 = -2$, $x_4 = 0$, and $x_6 = 2$ are all mapped to points with a second coordinate of 0 when passed through ϕ. Similarly $x_3 = -1$, and $x_5 = 1$ are mapped to points with a second coordinate of -2 when passed through ϕ.

3. The unit vector $[0, 1]$ is orthogonal to the boundary.

4. Since the unit vector $[0, 1]$ is orthogonal to the boundary, any solution $\mathbf{w}$ must be a scalar multiple of this vector. Moreover if $\mathbf{w} = [0, w_2]$, then w_2 must be positive since $\mathbf{w}$ must point toward the positive points which are above the decision boundary.

   Additionally, we'd like any point $\phi(x)$ with a second coordinate of -1 to have a discriminant of 0 based on the boundary. These points can be written as $(v, -1)$ for some value $v \in \mathbb{R}$. Taking the discriminant yields $0 \cdot v + w_2 \cdot -1 + w_0 = -w_2 + w_0$. Setting this expression equal to 0 and rearranging yields the condition $w_2 = w_0$.

   Now we consider the constraint $y_i(\mathbf{w}^\top \boldsymbol{\phi}(x_i) + w_0) \geq 1$. Substituting $\phi(x_i)$ and y_i for the given data yields the constraints $30w_2 + w_0 \geq 1$, $0w_2 + w_0 \geq 1$, and $-(-2w_2 + w_0) \geq 1$. Note that we only have 3 constraints from 7 points because the points $\phi(x_1), \ldots, \phi(x_7)$ only have 3 different second coordinates (30, 0, and -2). Plugging in $w_2 = w_0$ and simplifying yields $31w_0 \geq 1$, $w_0 \geq 1$, and $w_0 \geq 1$. $w_0 \geq 1 \implies 31w_0 \geq 31 \geq 1$, so our sole constraint is $w_0 \geq 1$. Note that this also satisfies the constraint that $w_2 = w_0$ must be positive.

   Thus our set of solutions is $\mathbf{w} = [0, c]$ and $w_0 = c$ for any $c \geq 1$.

5. If $\mathbf{w} = [0, c]$, then $\frac{1}{2}\|\mathbf{w}\|_2^2 = \frac{1}{2}(0^2 + c^2) = \frac{1}{2}c^2$. Minimizing this quantity subject to the constraint $c \geq 1$ gives us $c = 1$. Thus the optimal solution is $\mathbf{w} = [0, 1]$.

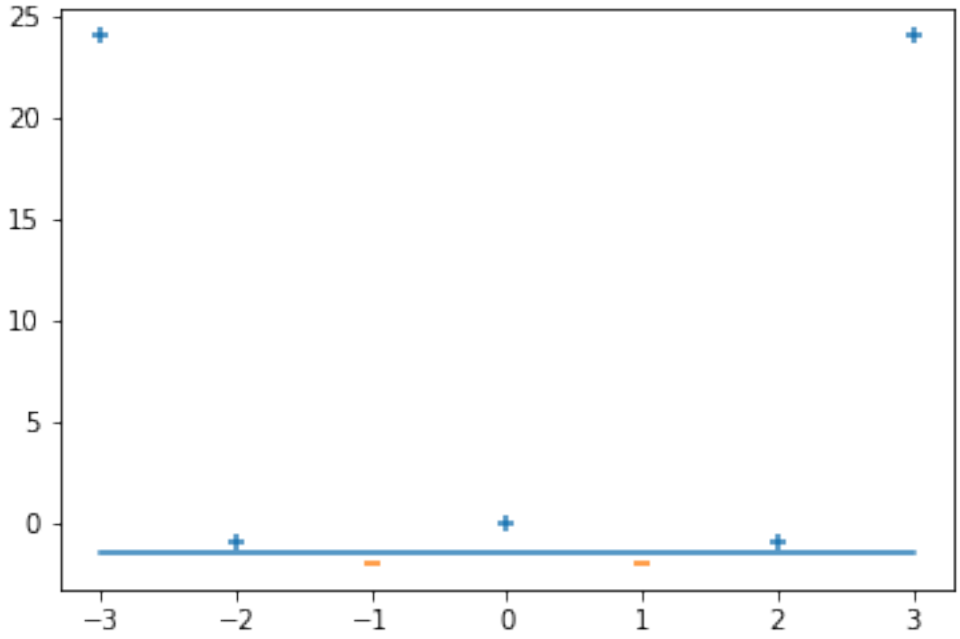
6. As we showed in part 4, $w_2 = w_0$, so the corresponding optimal value of w_0 is 1. Plugging in and expanding the discriminant yields

$$h^*(x) = -\frac{8}{3}x^2 + \frac{2}{3}x^4 + 1.$$

7. The points (-2, +1), (-1, -1), (0, +1), (1, -1), (2, +1) are all support vectors of the classifier. The

constraints are met with equality for these points since $h^*(2) = h^*(0) = h^*(-2) = 1$, and $h^*(-1) = h^*(1) = -1$.

8. The optimal boundary is shown below:



The mapping is still separable, but now the margin is 0.5 which is smaller than the margin from our initial feature mapping. We'd prefer the first set of features because the higher margin means that points near our sample data are more likely to be classified in the same way since the data is farther from the decision 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 (e.g. `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?

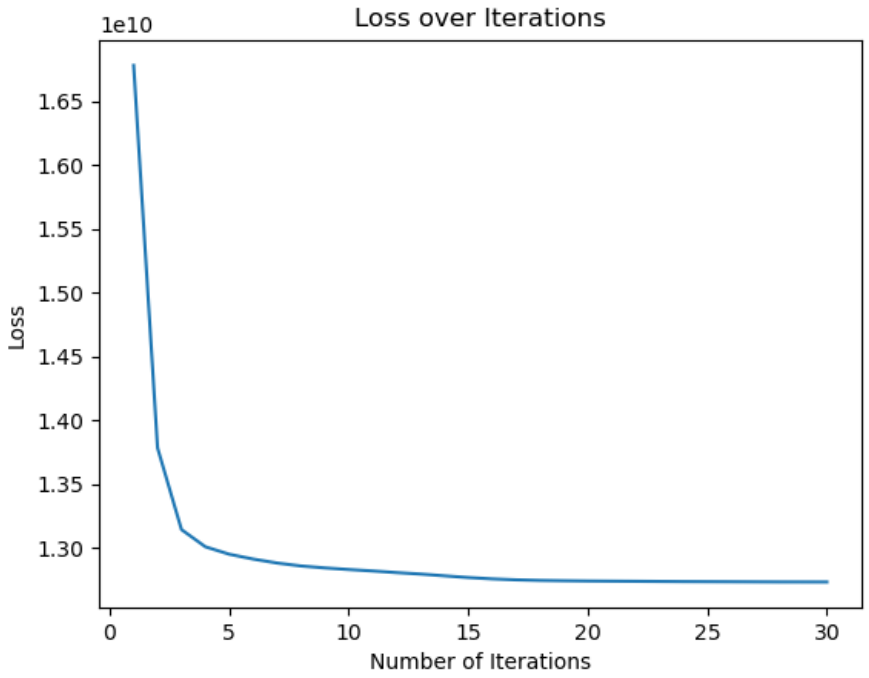
   Going back to the previous part: How does this help explain the crispness and blurriness of some of the clusters?

Problem 2 (cont.)

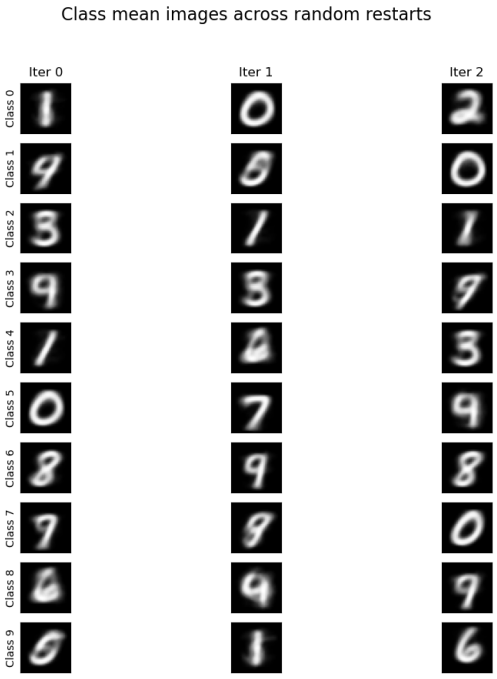
6. For your K-means with $K = 10$ model and HAC min/max/centroid models using 10 clusters on the `small_dataset` images, use the `seaborn` module's `heatmap` function to plot a confusion matrix between each pair of clustering methods. This will produce 6 matrices, one per pair of methods. The cell at 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

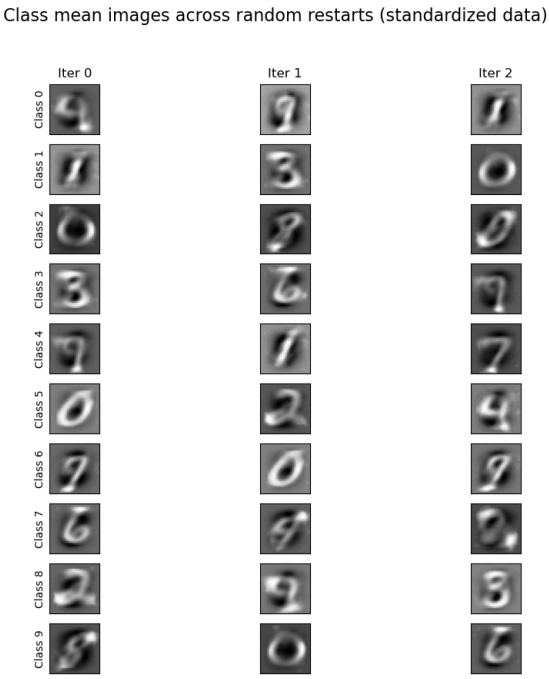
1. The loss function for K-means does not increase:



2. The mean images produced by K-means are:



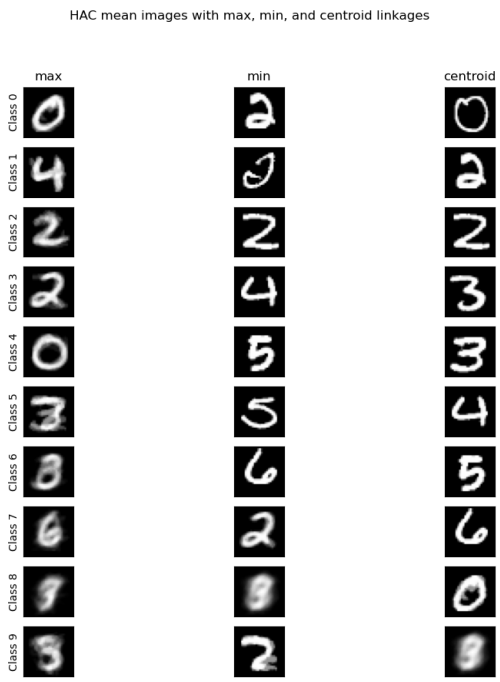
3. The standardized mean images produced by K-means are:



The centroids of the K-means clusters created using standardized data are sort of blotchy. They all seem to have a greyish border all the way around each centroid, and the part in the middle that resembles a digit isn't as clear as it was in part 2. In particular, the number part of each centroid has a very light grey color, and the centers (around the number part) are dark (unlike the rest of the centroid).

I think that this is the result of numerical overflow. In this color scale, black pixels have a value of 0, and white pixels have a value of 255. Over the entire dataset, the outer values are mostly 0 (or low values) since the edges are always dark. However, the mean is greater than zero since a few images have parts of the number reaching into the edges. Thus we're subtracting a small number from 0, and we'll end up with a negative value. Since the color scale goes from 0 to 255, these negative values will correspond to a grey color when we take mod 255. Similarly, the number parts in the middle aren't as crisp or white as in part 2 because the mean of the dataset has medium sized numbers for the middle parts. Thus when we subtract the mean, the white parts become duller like we see in the centroids. Finally, the dark parts in the middle can be explained by both the images for a cluster and the images overall having a similar mean in these areas. For example, consider a cluster corresponding to the number seven. There will be a set of light pixels shared by all images in the cluster but the edges of the seven will vary, so the pixels that are near the middle but not shared by every image in the cluster will have a low mean. The same principle applies to the dataset as a whole, so the dataset's mean roughly cancels out with the cluster's mean, and we're left with low values that correspond to 0.

4. The HAC mean images are:



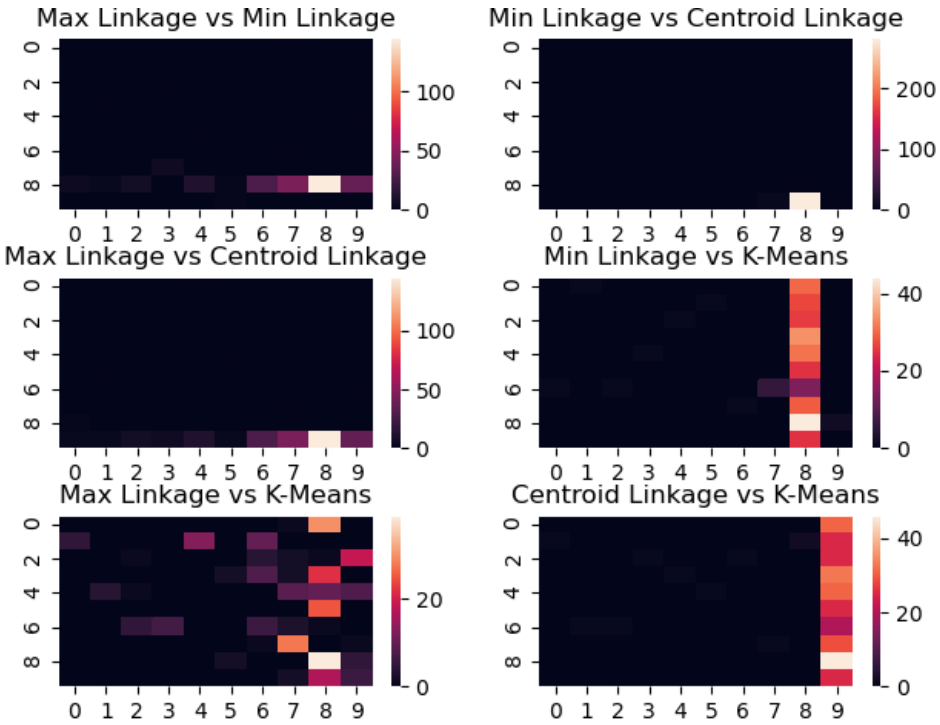
Note that my HAC centroids may line up with the answer key and simply be out of order. The centroids are noticeably crisper (especially for min and centroid linkage) than the k-means centroids. We're only asked to run the HAC model once since it is deterministic and would just output the same clusters/centroids.

5. The cluster distributions are:

The clusters for max linkage are a lot more spread out than they are for min linkage. Thus for each of the sparse clusters for min linkage, the centroid will be the average of fewer (very similar) images and therefore sharper.

More generally, the clusters with more images will tend to be blurrier. In particular, clusterings that are spread out will have several blurry images while the clusterings that are not spread out will have mostly sharp images with a handful being very blurry.

6. The confusion matrices are:



The closest HAC model seems to be the max linkage one. The clusters don't necessarily line up (e.g.

the 3rd cluster of model 1 may represent a 7 while the 5th cluster of model 2 represents a 7), so we can't simply look at the diagonal of the confusion matrix. Instead we're looking for high values in several cells since this suggests that a certain cluster i of model 1 has high correlation with another cluster j of model 2. In the min and centroid linkage models, we see high values in the 8th and 9th columns, respectively, but this just correlates to the 8th and 9th clusters having a lot of images for those models. On the other hand, the higher values are more spread for the max linkage distribution and suggests a closer correlation.

7. I think this is a fine method for checking the performance of a clustering model with one caveat. As we noted in part 6, the ith cluster may not line up with the ith digit, and we don't want to just look for high values along the diagonal. Instead (in the ideal case) we'd like to see one high value per column with no two columns having a high value in the same row. The high value per column suggests that all of the images showing a certain digit are sorted into the same cluster, and no two columns having a high value in the same row suggests that only images of the same digit end up in the same cluster.

Problem 3 (Ethics Assignment, 5pts)

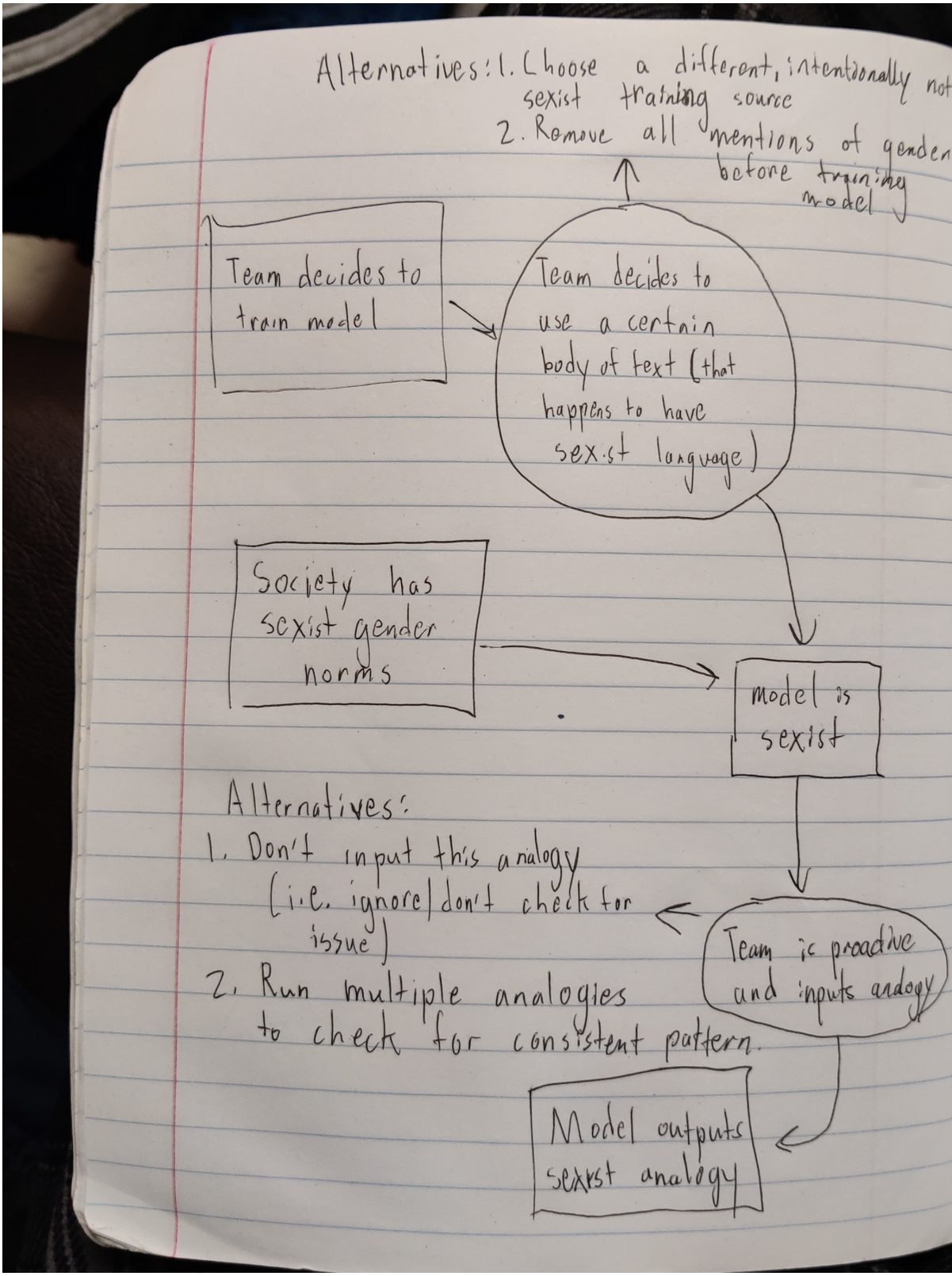
Select a real-life outcome in Artificial Intelligence or Machine Learning that you believe is morally wrong. You can select your own outcome from the news or select one of the outcomes in the options below:

- 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

I drew a chain for number 2: the NLP algorithm's analogy.


Alternatives: 1. Choose a different, intentionally not sexist training source
2. Remove all mentions of gender before training model
Team decides to train model
Team decides to use a certain body of text (that happens to have sexist language)
Society has sexist gender norms
model is sexist
Alternatives:
1. Don't input this analogy (i.e. ignore/don't check for issue)
2. Run multiple analogies to check for consistent pattern.
Team is proactive and inputs analogy
Model outputs sexist analogy

Name

Christy Jestin

Collaborators and Resources

Whom did you work with, and did you use any resources beyond cs181-textbook and your notes?
I worked with David Qian and Emil Liu. I used Google and GeeksforGeeks to check syntax.

Did you attend office hours for help with this homework?
No, I didn't.

Calibration

Approximately how long did this homework take you to complete (in hours)?
The homework took about 16 hours.