6.390: Midterm Exam, Fall 2023

Solutions

- This is a closed book exam. One page (8 1/2 in. by 11 in) of notes, front and back, are permitted. Calculators are not permitted.
- The total exam time is 2 hours.
- The problems are not necessarily in any order of difficulty.
- Record all your answers in the places provided. If you run out of room for an answer, continue on a blank page and mark it clearly.
- If a question seems vague or under-specified to you, make an assumption, write it down, and solve the problem given your assumption.
- If you absolutely *have* to ask a question, come to the front.
- **Write your name on every piece of paper.**

Name: _____ MIT Email: _____

Question	Points	Score
1	15	
2	20	
3	16	
4	17	
5	12	
6	20	
Total:	100	

Name: _____

ML Recipe

1. (15 points) In this question we will compare a basic learning algorithm vs. a learning algorithm that includes cross validation. In both algorithms, you can assume that regularization is available if needed or appropriate.

   (a) Below is pseudo-code for a basic learning algorithm. Here γ might be any hyperparameter of the learning algorithm (i.e., not necessarily related to a regularization loss term in an objective function).

   ```
   1.  divide data D into D_train, D_val, D_test
   2.  for each candidate value for hyperparameter γ
   3.    find lowest loss hypothesis h using D_train
   4.    compute loss using D_val
   5.  select γ_best with smallest loss on D_val
   6.  find h using D_train + D_val and γ_best
   7.  evaluate h on D_test
   ```

   i. Do you use regularization or no regularization in line 3? Explain your reasoning.

   Solution: It is better to regularize, to help find a hypothesis h that will generalize when used on D_{val}.

   ii. Do you use regularization or no regularization in line 4? Explain your reasoning.

   Solution: Now we do not regularize in line 4; instead we want to evaluate based on the loss terms without regularization, in order to assess how well the h generalizes to the validation data.

(b) Next, we will examine a learning algorithm that includes cross-validation. Here is the pseudocode. Again, γ may be any hyperparameter associated with the learning algorithm.

```
1.    divide data D into D_train, D_test
2.    divide D_train into chunks D_1, ..., D_k
3.    for each candidate value for hyperparameter γ
4.      for i = 1 to k
5.        train h_i on D_train - D_i
6.        compute validation loss E_i(h_i) on D_i
7.      compute E_γ := ????
8.    select γ_best with smallest E_γ
9.    find h using D_train and γ_best
10.   evaluate h on D_test
```

i. Complete line 7 in the algorithm (i.e., give a formula for E_γ).

Solution: We calculate the average validation loss across our k splits:

$$E_\gamma = \frac{1}{k} \sum_{i=1}^{k} E_i(h_i) \ ,$$

which also has the advantage of giving us a general sense of typical validation loss on any one split. However, any monotonically increasing function over the $E_i(h_i)$ values (like the sum) would also be sufficient for our goal of picking the best γ_{best}.

ii. Do you use regularization or no regularization in line 9? Explain your reasoning.

Solution: We include regularization. We know hyperparameter γ_{best} is the best for that hyperparameter, and want to use together with regularization to find a final hypothesis h that uses all of the training data to generalize well to the test data.

(c) What is the benefit, if any, of including cross validation in this learning algorithm?

Solution: Both approaches seek to find a "best" value for the hyperparameter γ under consideration. However, any single split of the data is subject to substantial chance as to what data is in each split. The cross-validation approach enable us to average across multiple different splits of our data to reduce the variance of E_γ, and so find a good γ based on this less noisy evaluation of how the resulting hypothesis h generalizes.

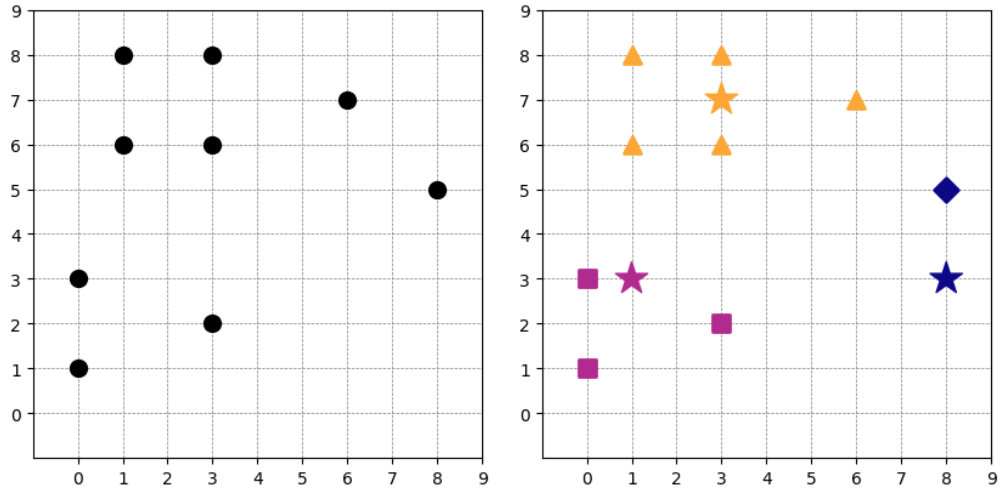
All of the answers here apply whether γ is a regularization hyperparameter like λ in ridge regression, or if γ is some other hyperparameter of the learning algorithm.

Name: _____

k-Means Redistricting

2. (20 points) Redistricting in the United States is the process of drawing boundaries of electoral districts in response to census data that is collected every 10 years. The goal of redistricting is to "accommodate shifts in population and provide equal representation to its citizens." In this problem, we approach redistricting using modified versions of k-means clustering.

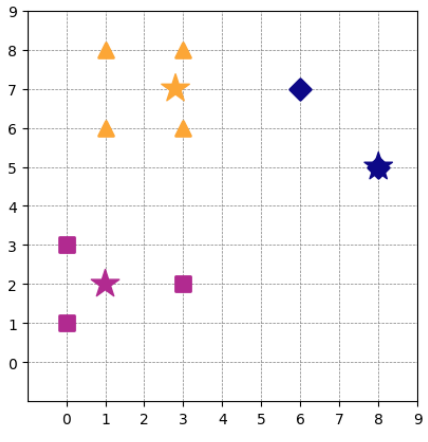
   (a) Chassamusetts is a small state with nine cities (all having equal populations) with spatial coordinates x_1 (horizontal coordinate) and x_2 (vertical coordinate) as shown in the figure below on the left. The cities need to be allocated into three districts. We initialize our standard k-means clustering algorithm with three centroids (stars, as shown in the figure on the right), and then allocate the cities into districts based on these initial cluster centers. Distinct clusters (districts) are differentiated by shape: triangles, squares, or diamonds.



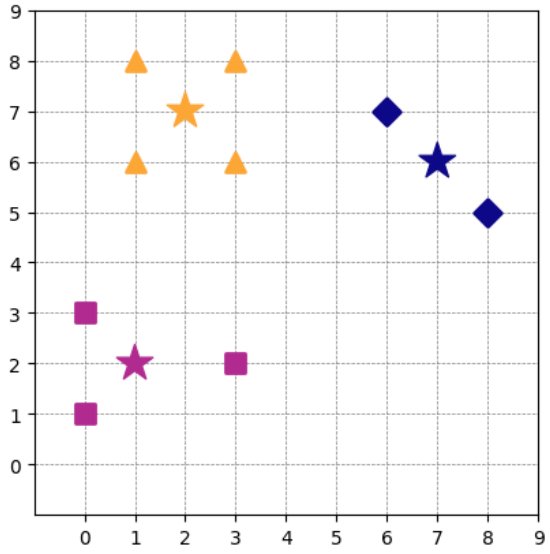
We perform one training update of our model using the standard k-means clustering algorithm to first produce new cluster centroids and then new cluster assignments for the nine cities. On the blank plot below, plot the new centroids and new cluster assignments using the same shapes used in the plot on the right.

Solution:

- Triangle centroid at $[14/5, 7]^T$
- Diamond centroid at $[8, 5]^T$
- Square centroid at $[1, 2]^T$



(b) We train our k-means clustering algorithm with three clusters to convergence, resulting in the centroids and clusters shown in the figure below. Again, distinct clusters are differentiated by shape (triangles, squares, or diamonds), and centroids for all clusters are shown as stars.



Recall the k-means loss:

$$L_{k\text{-means}} = \sum_{j=1}^{k} \sum_{i=1}^{n} \mathbb{1}(y^{(i)} = j) \|x^{(i)} - \mu^{(j)}\|^2$$

where $\mu^{(j)}$ is the position of the centroid for cluster j. The cluster centroids are located at $[1, 2]^T$ for squares, $[2, 7]^T$ for triangles, and $[7, 6]^T$ for diamonds. Calculate the k-means loss for this clustering.

Solution:

$$L_{k\text{-means}} = L_{\square} + L_{\triangle} + L_{\circ} = 20$$

$$L_{\square} = \left\| \begin{pmatrix} 0 \\ 1 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 0 \\ 3 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 3 \\ 2 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right\|^2 = 2 + 2 + 4 = 8$$

$$L_{\triangle} = \left\| \begin{pmatrix} 1 \\ 6 \end{pmatrix} - \begin{pmatrix} 2 \\ 7 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 1 \\ 8 \end{pmatrix} - \begin{pmatrix} 2 \\ 7 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 3 \\ 8 \end{pmatrix} - \begin{pmatrix} 2 \\ 7 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 3 \\ 8 \end{pmatrix} - \begin{pmatrix} 2 \\ 7 \end{pmatrix} \right\|^2 = 4 \cdot 2 = 8$$

$$L_{\circ} = \left\| \begin{pmatrix} 6 \\ 7 \end{pmatrix} - \begin{pmatrix} 7 \\ 6 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 8 \\ 5 \end{pmatrix} - \begin{pmatrix} 7 \\ 6 \end{pmatrix} \right\|^2 = 2 + 2 = 4$$

(c) We want districts to have (approximately) equal population, so we define a loss component L_{unequal} that penalizes districts of unequal size:

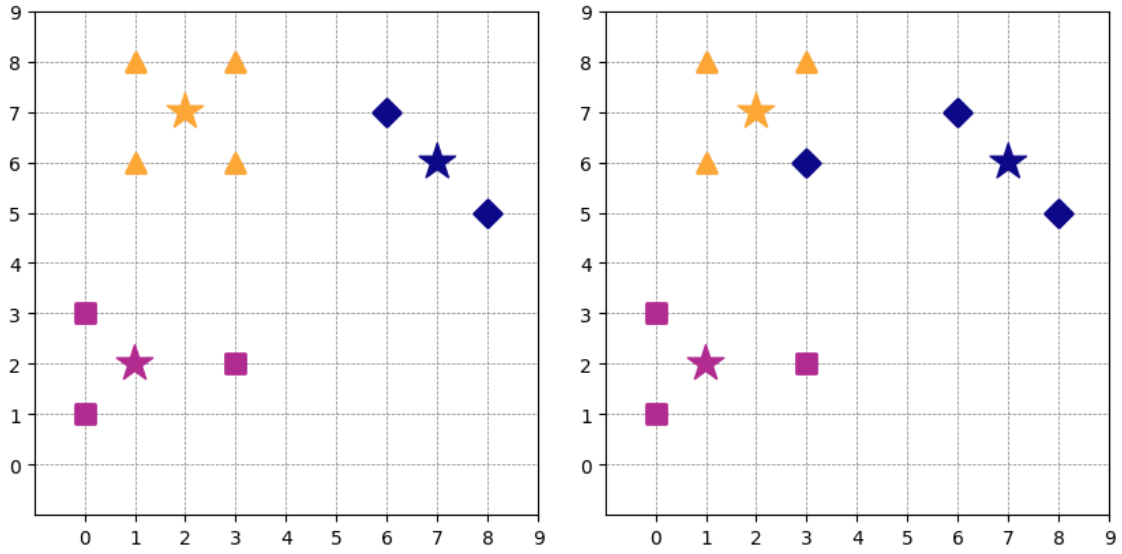
$$L_{\text{unequal}} = \sum_j (N_j - N_{\text{target}})^2$$

where N_j is the number of cities in district j, and N_{target} is the target number of cities per district (equal to the total number of cities, N_{cities}, divided by the number of districts, $N_{\text{districts}}$).

The overall objective for our modified k-means clustering is

$$L_{\text{mod}} = L_{\text{k-means}} + \lambda_{\text{unequal}} \cdot L_{\text{unequal}}$$

with weighting factor λ_{unequal}. In the figures below, we have the original k-means clustering on the left, and on the right we have a proposed "redistricting" that balances these districts based on this modified overall objective L_{mod}.



i. Calculate the L_{unequal} loss for the original k-means clustering on the left.

Solution: $L_{\text{unequal}} = (3-3)^2 + (4-3)^2 + (2-3)^2 = 2$

ii. Calculate the total loss L_{mod} for the original k-means clustering on the left, assuming $\lambda_{\text{unequal}} = 0.1$.

Solution:

$$\begin{aligned} L_{\text{mod}} &= L_{\text{k-means}} + \lambda_{\text{unequal}} \cdot L_{\text{unequal}} \\ &= 20 + 0.1 \cdot 2 = 20.2 \end{aligned}$$

iii. Calculate the L_{mod} loss for the proposed "redistricting"on the right.

Solution: The L_{unequal} loss on the right is zero. We can quickly recalculate the $L_{\text{k-means}}$ for the redistricting by just considering the change due to the one changed point:

$$L_{k\text{-means}}^{(\text{re})} = 20 - 2 + 4^2 = 18 + 16 = 34$$

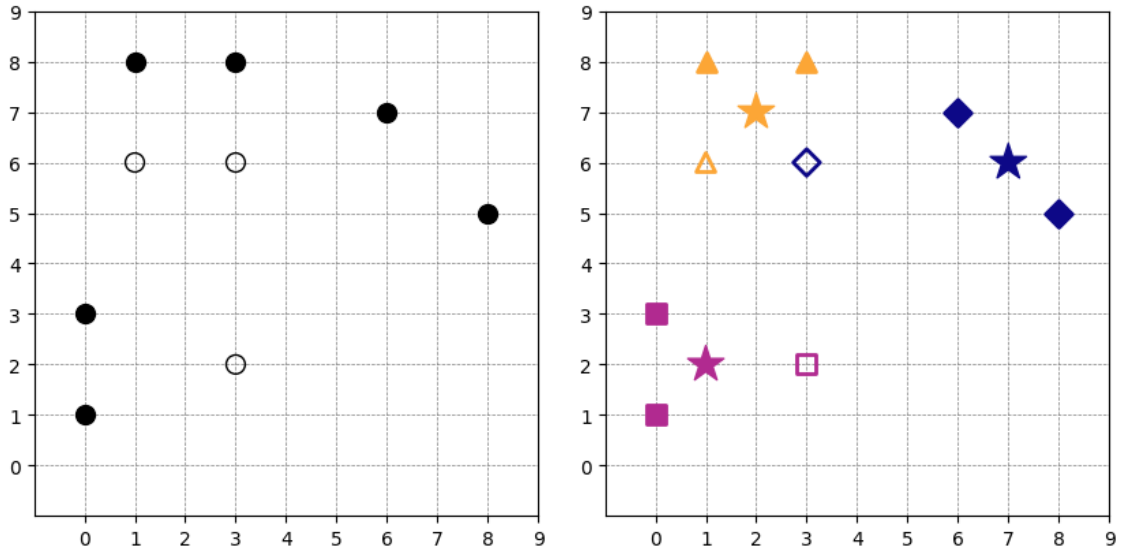
so $L_{\text{mod}} = L_{k\text{-means}} + 0 = 34$.

iv. For this particular set of cities, what range of values for λ_{unequal} do we need in order to ensure that the desired population-balanced district assignments will result (i.e, so that the cluster assignment stage from the left figure above to the right figure above will occur)?

Solution: We need $20 + \lambda_{\text{unequal}} \cdot 2 > 34$, or $\lambda_{\text{unequal}} > 7$.

Name: _____

(d) The cities of Chassamusetts belong to one of two parties, the Depublicans ($p = D$) indicated with open circles or the Remocrats ($p = R$) indicated with filled circles, in the diagram below on the left. On the right, we plot how these demographics are reflected in the redistributed clusters found above (that considered both k-means distance and population imbalance together), again with open or filled symbols to indicate party p.



A goal of redistricting is to provide equal representation to its citizens. Governor Merry Gander proposes a new loss term in order to encourage the demographics at the district level to reflect the demographics of the cities:

$$L_{\text{party}} = \sum_{p \in \{R,D\}} \left(\frac{\#\text{ majority } p \text{ districts}}{N_{\text{districts}}} - \frac{\#\ p \text{ cities}}{N_{\text{cities}}} \right)^2$$

where $N_{\text{districts}} = 3$, $\#\ D$ cities $= 3$, $\#\ R$ cities $= 6$, $N_{\text{cities}} = 9$, and $\#$ majority p districts are determined by the specifics of the clustering. A majority exists if more than 50% of cities in a district are of a certain party.

i. Calculate the value of L_{party} for the clustering shown above on the right.

Solution: $L_{\text{party}} = (\frac{0}{3} - \frac{3}{9})^2 + (\frac{3}{3} - \frac{6}{9})^2 = \frac{2}{9}$

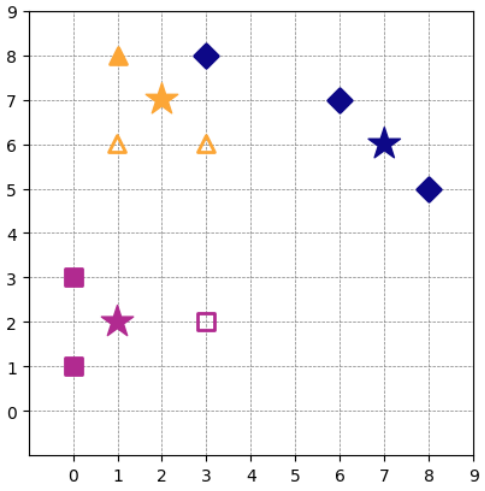
Combining the party affiliation consideration gives a new overall objective

$$L_{\text{combined}} = L_{\text{k-means}} + \lambda_{\text{unequal}} \cdot L_{\text{unequal}} + \lambda_{\text{party}} \cdot L_{\text{party}}$$

where λ_{party} is another weighting factor.

ii. Give the locations of the two cities such that if they swapped districts it would (just after this swap) minimize the L_{party} loss while incurring the smallest increase in the $L_{k\text{-means}}$ loss and would *not* change L_{unequal}.

Solution: If cities at locations $[3, 6]^T$ and $[3, 8]^T$ are swapped, L_{party} loss goes to zero. The L_{unequal} loss does not change, since all districts still have three cities in them. Given the near/similar proximity of both points in the swap to their cluster centers, this will incur the least increase in k-means loss. (Specifically, the $L_{\text{k-means}}$ changes slightly by increasing the squared distance from $[7, 7]^T$ to $[3, 6]^T$ from 4^2 to $4^2 + 2^2$ for $[3, 8]^T$, giving a total k-mean loss increase of 4). This swap results in the following districts:



Name: _____

Descent-Ascent

3. (16 points) To minimize a differentiable function $f: \mathbb{R}^3 \to \mathbb{R}$, Dr. A. I. Ancient uses gradient descent iterations,

$$x^{(t)} = x^{(t-1)} - \eta \nabla_x f(x^{(t-1)}) \qquad (t = 1, 2, \dots), \qquad x^{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

where $x^{(0)}$ is the initial guess, and $x^{(t)}$ is hoped to be an improvement over $x^{(t-1)}$, in the sense that $f(x^{(t)}) < f(x^{(t-1)})$. Dr. Ancient first uses $\eta = 0.1$, which (regretfully) results in

$$x^{(1)} = \begin{bmatrix} 2 \\ -4 \\ 6 \end{bmatrix} = 2\,\hat{e}, \qquad x^{(2)} = \begin{bmatrix} 1 \\ -2 \\ 3 \end{bmatrix} = \hat{e}, \qquad x^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \qquad \text{where} \quad \hat{e} = \begin{bmatrix} 1 \\ -2 \\ 3 \end{bmatrix}.$$

(a) What is the value of $x^{(7)}$ in this instance of gradient descent?

Solution: the first three iterations demonstrate that we are oscillating between three points. Since the step size is fixed, this trend will continue, and $x^{(7)} = x^{(1)} = 2\,\hat{e} = [2, -4, 6]^T$.
NOTE: the conflict exam asked for $x^{(8)}$

(b) What are the vectors $\nabla_x f(a)$ for $a = x^{(0)}$, $a = x^{(1)}$, and $a = x^{(2)}$?

Solution:

$$\begin{aligned}
x^{(1)} &= -0.1 \nabla_x f(x^{(0)}) \quad \Rightarrow \quad \nabla_x f(x^{(0)}) = -20\,\hat{e} = [-20, 40, -60]^T \\
x^{(2)} &= x^{(1)} - 0.1 \nabla_x f(x^{(1)}) \quad \Rightarrow \quad \nabla_x f(x^{(1)}) = -10(x^{(2)} - x^{(1)}) = 10\,\hat{e} = [10, -20, 30]^T \\
x^{(3)} &= x^{(2)} - 0.1 \nabla_x f(x^{(2)}) \quad \Rightarrow \quad \nabla_x f(x^{(2)}) = 10\,\hat{e} = [10, -20, 30]^T
\end{aligned}$$

(c) Could it be true that $f(x^{(t)}) < f(x^{(t-1)})$ for all $t = 1, 2, 3$ in this original instance of running gradient descent? Explain your reasoning.

Solution: no; given the oscillating behavior, we have already seen in part (a) that $x^{(7)} \not< x^{(1)}$.

(d) Disappointed with the results obtained with $\eta = 0.1$, Dr. Ancient decides to use $\eta = 0.05$, with the same initial guess $\bar{x}^{(0)} = x^{(0)}$. Find vectors $\bar{x}^{(1)}$ and $\bar{x}^{(2)}$ for this new instance of running gradient descent (where $\bar{x}^{(i)}$ indicates these are for the new instance of running gradient descent).

Solution:

$$\bar{x}^{(1)} = -0.05\nabla_x f(x^{(0)}) = -0.05(-10x^{(1)}) = \hat{e} = [1, -2, 3]^T$$
$$\bar{x}^{(2)} = \bar{x}^{(1)} - 0.05\nabla_x f(\bar{x}^{(1)}) = x^{(2)} - 0.05 \cdot 10x^{(2)} = 0.5\,\hat{e} = [0.5, -1, 1.5]^T$$

(e) Dr. Ancient engages Dr. M. A. Th to do a theoretical analysis of the minimization problem. Dr. Th discovers that the function $f : \mathbb{R}^3 \to \mathbb{R}$ is convex, and claims that $f(x)$ achieves its minimum at a point $x = t\,\hat{e}$, where t is a real scalar, and $\hat{e}$ is as defined earlier. Hearing this news, Dr. Ancient concentrates on minimizing the function $g : \mathbb{R} \to \mathbb{R}$ defined by $g(t) = f(t\,\hat{e})$. Compute the gradient of g with respect to t at $t = 0$, $t = 1$, and $t = 2$.

Solution: since $\nabla_t g(t) = \hat{e}^T \nabla_x f(t\,\hat{e})$,

$$\nabla_t g(0) = \hat{e}^T(-20\,\hat{e}) = -20\,\hat{e}^T\hat{e} = -20 \cdot 14 = -280,$$
$$\nabla_t g(1) = \hat{e}^T(10\,\hat{e}) = 10\,\hat{e}^T\hat{e} = 140,$$
$$\nabla_t g(2) = \hat{e}^T(10\,\hat{e}) = 10\,\hat{e}^T\hat{e} = 140.$$

(f) Assuming Dr. Th is right, give the range of values of t that could possibly give minimum $g(t)$ for function g as defined in (e).

Solution: since g is convex, its gradient is monotonically non-increasing. Hence $\nabla_t g(t) \geq \nabla_t g(1) = 140$ for $t \geq 1$, and $\nabla_t g(t) \leq \nabla_t g(0) = -280$ for $t \leq 0$. Therefore, any argument of minimum of g must be in the interval $0 < t < 1$.

Name: _____

DeBugging with ML

4. (17 points) Dr. Dee Scent is hot on the trail of a new technique to eradicate mosquitoes carrying the Zika virus. In preliminary exciting research, she has discovered that the *Aedes* species of mosquitoes beat their wings at around 300 Hz, and this species carries the virus. You are hired as an MIT graduate who can turn this new insight into a classifier that can identify which mosquitoes are positive for Zika, and which are not.

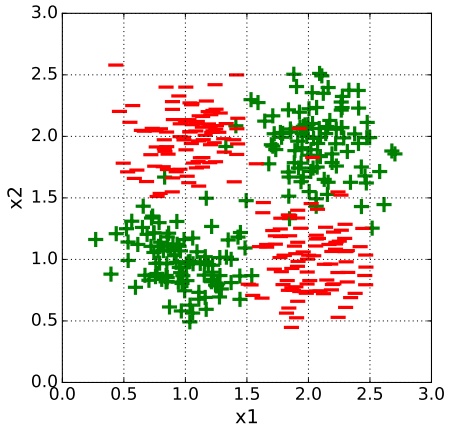
   (a) Dr. Dee Scent improves her apparatus, and in addition to measuring wing beat speed (variable x_1, in hundreds of Hz, with some shift or offset), she now also measures their size (variable x_2, in her custom standard units, again with some shift or offset), for five different collections of mosquitoes. But the data are no longer so clear cut! Or are they?

   For each of the five sample data plots below (where the symbol '+' indicates a mosquito with Zika, and '−' one without), a logistic regression classifier $h(x;\theta,\theta_0) = \sigma(\theta^T\phi(x)+\theta_0)$ is desired, where $h(x;\theta,\theta_0) > 0.5$ indicates class '+', or class '−' otherwise, and ϕ is a feature transformation. The classifier is trained with the data shown using an objective function with negative log likelihood loss $\mathcal{L}_{\text{nll}}$, and regularization with modest non-zero positive λ:

$$\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_{\text{nll}}(\sigma(\theta^T\phi(x^{(i)})+\theta_0), y^{(i)})\right) + \lambda\|\theta\|^2 \ .$$

   For each of the data sets below, help Dr. Dee Scent by choosing the *smallest* dimension feature transformation that can be used for training by the classifier to achieve approximately 80% accuracy or better classification of the data. If none of the feature transformations are able to achieve this, then say so. Explain your choice.

   i. First dataset:



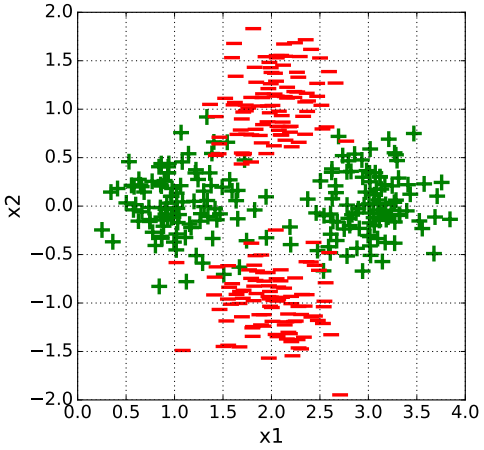
Solution: ◯ $[x_1]$ ◯ $[x_2]$ ◯ $[x_1, x_2]$ ◯ $[x_1^2]$ ◯ $[x_2^2]$

✓ $[x_1,\ x_2,\ x_1x_2,\ x_1^2,\ x_2^2]$ ◯ None of these

The best choice is $[x_1,\ x_2,\ x_1x_2,\ x_1^2,\ x_2^2]$ with decision boundaries around $(x_1 + x_2 - 3)^2 > 0.25$. More generally, this transformation can define an ellipse or hyperbola for the classifier boundary. These can have slanted axes, so can mostly cover the '−' points, for good accuracy separation from the '+' points. The other

smaller dimension options do not achieve good accuracy classifiers. Single vertical, horizontal, or slanted lines (with or without offsets) give poor separation. Squaring the individual axes just stretches the (already all positive) axes, but doesn't help with separation.

ii. Second dataset:

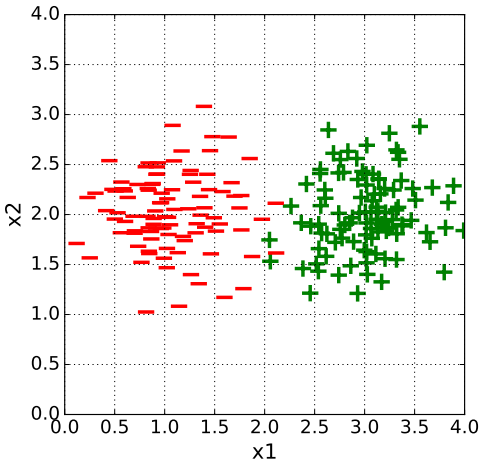


Solution: ◯ $[x_1]$ ◯ $[x_2]$ ◯ $[x_1, x_2]$ ◯ $[x_1^2]$ √ $[x_2^2]$

◯ $[x_1,\ x_2,\ x_1x_2,\ x_1^2,\ x_2^2]$ ◯ None of these

Squaring the x_2 values will map the '−' points in the bottom half into the upper region with the other '−' points, where a horizontal classifier boundary at $x_2^2 = 0.5$ will work to divide most of these from the mapped '+' points. Note that x_1^2 alone does not work, as we would also need an x_1 term to establish classifier boundaries like $(x_1 - 2)^2 > 0.25$.

iii. Third dataset:



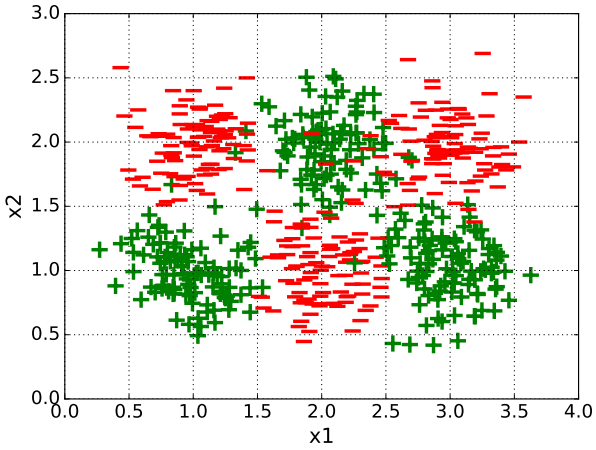
Name: _____

Solution: √ $[x_1]$ ◯ $[x_2]$ ◯ $[x_1, x_2]$ ◯ $[x_1^2]$ ◯ $[x_2^2]$

◯ $[x_1,\ x_2,\ x_1x_2,\ x_1^2,\ x_2^2]$ ◯ None of these

A vertical classifier boundary at $x_1 = 2$ is sufficient!

iv. Fourth dataset:



Solution: ◯ $[x_1]$ ◯ $[x_2]$ ◯ $[x_1, x_2]$ ◯ $[x_1^2]$ ◯ $[x_2^2]$

◯ $[x_1,\ x_2,\ x_1x_2,\ x_1^2,\ x_2^2]$ √ **None of these**

Very hard for a linear classifier here. Again, all x_1^2 and x_2^2 will do is stretch things out. Even if stretched differently along x_1 and x_2 (or defining an ellipse), we don't get something with good separation.

(b) After further improving her apparatus to measure in outdoor environments, Dr. Dee Scent discovers that mosquito wing beat frequency depends slightly (but noticeably) on ambient air pressure. She gives you these air pressure values as x_3, in units of millibars (mb). They all are largely within the range 1013 ± 100 mb. Assuming x_1 and x_2 are typically within the range given in part (a) above, what is the best way to include these new x_3 values as a new feature in your model? Explain.

Solution: We should normalize to comparable ranges that we have for x_1 and x_2.

(c) Your success enables Dr. Dee Scent to establish 24 new mosquito research stations around the world, each named with a letter from the Greek alphabet, i.e., Alpha, Beta, Gamma, $\cdots$, Omega. Mosquito species responses are found to be shifted at different stations. As a test of your classifier, Dr. Dee Scent gives you data (including research station names) from a random half of the stations, and asks you for predictions for data from the other half. How should you encode the names of the research stations so you can do a good job at the predictions? Explain.

Solution: One-hot: we know that the shifts are different at each station (with no hint toward any trends), so one-hot will enable us to account for step differences (offsets) between stations in our training set, but without those propagating into the unknown offsets of the test set.

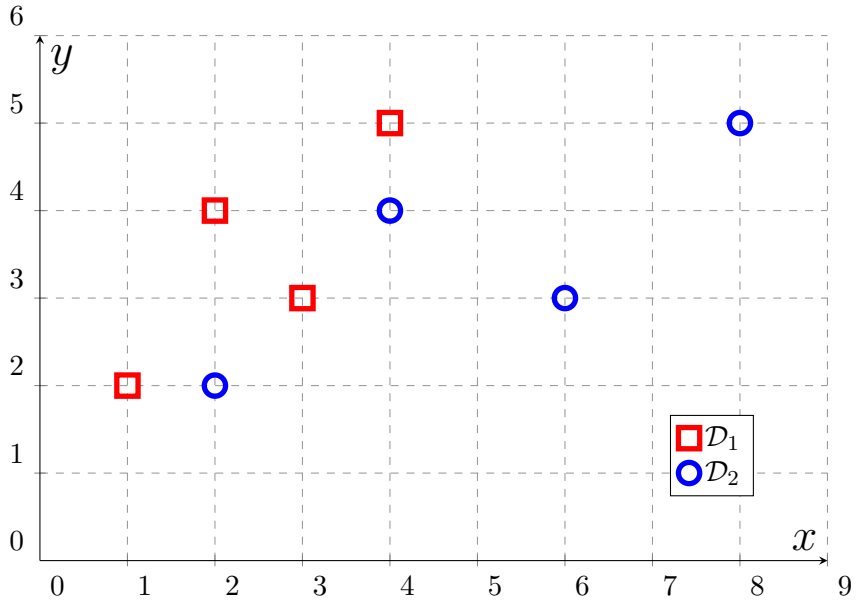
Name: _____

Regression

5. (12 points) We are given two different training datasets,

$$\mathcal{D}_1 = \{(x_1^{(i)}, y_1^{(i)})\}_{i=1}^4, \ \mathcal{D}_2 = \{(x_2^{(i)}, y_2^{(i)})\}_{i=1}^4.$$

The two datasets are plotted together in the figure below; $\mathcal{D}_1$ is plotted with squares while $\mathcal{D}_2$ is plotted with circles.



(a) We have two hypotheses and want to match each hypothesis to the data that it models best according to mean-squared error:

$$h_1(x) = x + 1,$$
$$h_2(x) = \frac{1}{2}x + 1.$$

Calculate mean squared error (MSE) of each hypothesis with respect to each dataset. Based on your MSE calculations, indicate which hypothesis is the best fit for each dataset.

Solution:
NOTE: conflict switched h_1 and h_2:
MSE of h_1 on $\mathcal{D}_1$: $\frac{1}{4}(0+1+1+0) = \frac{2}{4} = 0.5$
MSE of h_1 on $\mathcal{D}_2$: $\frac{1}{4}(1+1+16+16) = \frac{34}{4} = 8.5$
MSE of h_2 on $\mathcal{D}_1$: $\frac{1}{4}(0.25+4+0.25+4) = \frac{8.5}{4} = 2.125$
MSE of h_2 on $\mathcal{D}_2$: $\frac{1}{4}(0+1+1+0) = \frac{2}{4} = 0.5$
h_1 on $\mathcal{D}_1$; h_2 on $\mathcal{D}_2$

(b) Comparing the two datasets, we notice that the labels are identical for four pairs of data points (e.g, at (1, 2) and (2, 2)). We decide to see whether these datasets can be combined

in order to perform a "joint" linear prediction. To do so, consider a new dataset $\mathcal{D}_3$:

$$\mathcal{D}_3 = \left\{ \left(\begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}, y_1^{(i)} \right) \right\}_{i=1}^{4},$$

where $x_1^{(i)}$ correspond to the square ($\mathcal{D}_1$) data points, and $x_2^{(i)}$ correspond to the circle ($\mathcal{D}_2$) data points in the figure. Now, our goal is to learn the ordinary least squares (OLS) estimate of our parameters, including a θ_0 constant offset term as appeared in h_1 and h_2:

$$\theta^{\text{OLS}} = (\tilde{X}^T \tilde{X})^{-1} \tilde{X}^T \tilde{Y} .$$

Identify correct matrices $\tilde{X}, \tilde{Y}$ such that θ^{OLS} above expresses the solution of the joint regression problem.

Solution: $\tilde{X} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 1 \\ 3 & 6 & 1 \\ 4 & 8 & 1 \end{bmatrix}$, $\tilde{Y} = \begin{bmatrix} 2 \\ 4 \\ 3 \\ 5 \end{bmatrix}$

The third column in $\tilde{X}$ corresponds to the constant offset component of θ^{OLS}.

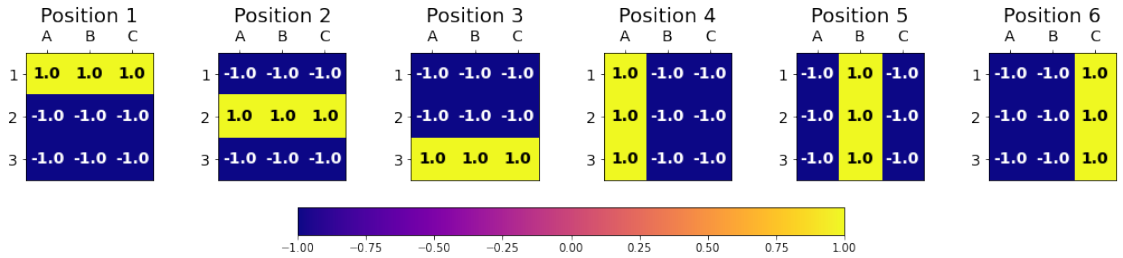
(c) Out of the vector identities listed below, identify which would imply that we will not be able to compute the closed-form solution to the joint regression problem using θ^{OLS} as defined above. Explain your reasoning for your choice(s).

Solution:
NOTE: conflict just switched the choices around.
$[1, 2, 3, 4] = 1/2 * [2, 4, 6, 8]$; the two features are co-linear and so $\tilde{X}^T \tilde{X}$ is not invertible.

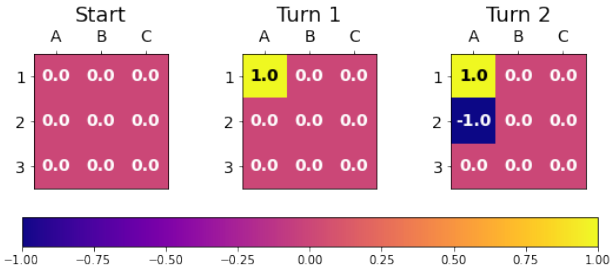
Name: _____

Battleblocks

6. (20 points) Battleblocks is a strategy game where each player picks a position to play their battleblock on a 3×3 game board, which is hidden to the other player. One battleblock takes up either a complete column or a complete row in the game board. The object of the game is to "hit" all three squares occupied by your opponent's hidden battleblock before they hit yours. There are six possible locations for an opponent's battleblock:



Players keep track of hits and misses also on a 3×3 board. For example, suppose an opponent's battleblock is in Position 1 shown above. At the start of the game, the player's game board is that on the left in the figure below: all zeros. If the player guesses A1 on their first turn, then the player's game state is updated to be the state in the middle with a +1 in the A1 position to indicate the hit. If on the second turn, the player then targets A2, their game state is updated to be the state on the right with a -1 at A2 to indicate the miss.



In this problem, we are going to use one-vs-all (OvA) multiclass classification which takes as input the current game board and outputs a probability that the current game board is one of the six positions.

(a) While our game board is a 3×3 square, our classifier operates on feature vectors of shape $[x_{A1}, x_{B1}, \ldots, x_{C3}]^T$. Write the game board for the Turn 2 (A1 - hit, A2 - miss) in this form.

Solution: Intended solution (looking ahead to next parts):
$x = [1, 0, 0, -1, 0, 0, 0, 0, 0]^T$.

(b) Suppose that we are given a dataset $\mathcal{D}_1 = \{(x^{(i)}, y^{(i)})\}_{i=1}^n$ where each feature vector $x^{(i)}$ is a game board as specified in (a) and each label $y^{(i)}$ is +1 if the opponent's battleblock was in Position 1 and 0 otherwise. We use this dataset to learn a binary, linear logistic classifier of the form $h(x^{(i)}; \theta, \theta_0) = \sigma(\theta^T x + \theta_0)$ by minimizing the linear logistic regression objective function,

$$J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{\text{nll}}(h(x; \theta, \theta_0), y^{(i)}) + \lambda \|\theta\|^2,$$

using some small, positive value for λ. The resulting learned parameters of our hypothesis $h^{(1)}(x^{(i)}; \theta^{(1)}, \theta_0^{(1)})$ for identifying Position 1 are:

$$\theta^{(1)} = [2, 2, 2, -1, -1, -1, -1, -1, -1]^T, \ \theta_0^{(1)} = -9.$$

Next, we want a linear logistic classifier that positively classifies Position 2 and negatively classifies Positions 1 and 3-6. However, we do not want to run the optimization process again for Position 2. Given what we know about this Position 1 classifier, give a combination of values $\theta^{(2)}$ and $\theta_0^{(2)}$ to define a binary, linear logistic classifier that labels Position 2 as +1 and 0 otherwise.

Solution: $\theta_0^{(1)} = [-1, -1, -1, 2, 2, 2, -1, -1, -1]^T, \ \theta_0^{(1)} = -9.$

(c) Suppose we re-train the linear logistic classifier *with different regularization* that positively classifies Position 1 to get the following:

$$\bar{\theta}^{(1)} = [5, 5, 5, 0, 0, 0, 0, 0, 0]^T, \ \bar{\theta}_0^{(1)} = -5$$

Was this classifier trained with stronger or weaker regularization than the one given in (b)? Will this model be more or less confident in its predictions? Explain your reasoning.

Solution: As $\|\theta^{(1)}\|^2 < \|\bar{\theta}^{(1)}\|^2$, we can conclude that this new classifier used weaker regularization; it will be more confident in its predictions.

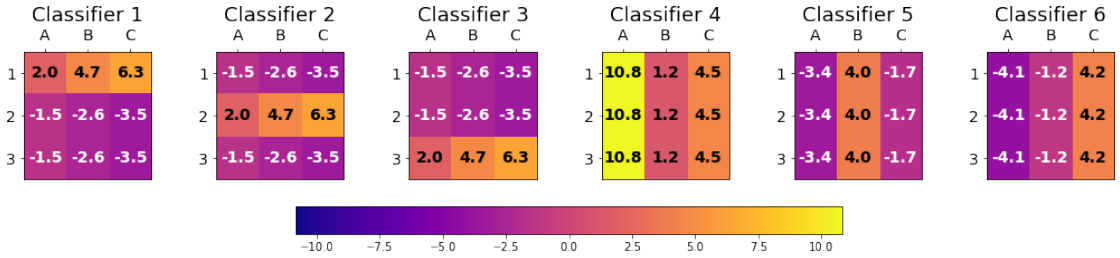
(d) Now, we will combine the OvA classifiers defined by the set of learned parameters $\{\theta^{(i)}, \theta_0^{(i)}\}_{i=1}^6$ into a multiclass hypothesis defined by parameters θ^*, θ_0^* which takes as input a game board and outputs the probabilities that the gameboard should map to each of the six positions:

$$h(x; \theta^*, \theta_0^*) = \text{softmax}(\theta^{*T} x + \theta_0^*)$$

What are the dimensions of θ^* and θ_0^*? What are the dimensions of the output?

Solution: θ^* is a 9×6 matrix and θ_0^* is a 6×1 vector. Thus, the output is also a 6×1 vector.

(e) We obtain a new dataset by playing multiple games with Commander Consistent and train a model on the possible final positions, where all training is done with the same λ regularization parameter. We arrive at new $\{\theta^{(i)}, \theta_0^{(i)}\}_{i=1}^6$ for our six binary linear logistic classifiers. The $\theta^{(i)}$ values for each classifier are plotted below; unfortunately, the corresponding $\theta_0^{(i)}$ values are missing. From the available parameter values, can you determine what Commander Consistent's most frequent placement of his battleblock is? If so, state what that placement is, and explain your reasoning. If not possible, explain why not.



Solution: Commander Consistent seems to prefer placing their battleblock in Position 4, according to the θ parameter values of Classifier 4 having relatively large magnitude in comparison to the other five classifiers. This does not depend on the corresponding θ_0 parameters. Since the same regularization was done in all cases, the NLL term in the overall J_{lr} objective will be relatively more important compared to the regularization term as the number of training examples increases. So the larger θ coefficients (for the same λ) corresponds to where more training examples n are present, telling us what position/strategy Commander Consistent uses more often. (Another way to say this, is that the linear logistic classifier model becomes more confident with more training samples, corresponding to larger θ coefficients.)