6.390: Midterm, Fall 2022

Solutions

- This is a closed book exam. One page (8 1/2 in. by 11 in. or A4) of notes, front and back, is 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	20	
2	20	
3	20	
4	20	
5	20	
Total:	100	

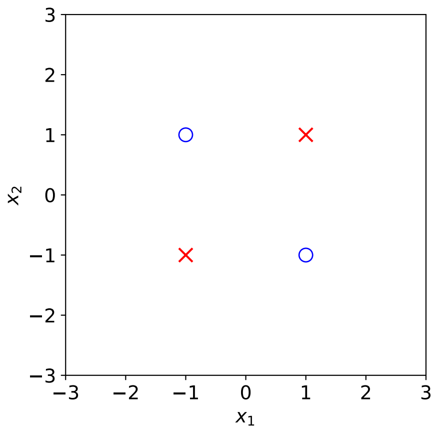
Name: ______________________

Validation Vacillation

1. (20 points) You have the data set shown below, where two samples belong to one class (marked with X's), and two samples belong to a different class (marked with circles). You are interested in finding the best linear classifier for this data, but also want to estimate the accuracy of the resulting classifiers using leave-one-out (or four-fold) cross-validation. Here, accuracy A is measured as

$$A(h; \mathcal{D}) = 1 - \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{01}(g^{(i)}, y^{(i)}),$$

where n is the number of data points in whichever dataset $\mathcal{D}$ is used, $\mathcal{D}$ is $\mathcal{D}_{train}$ for training accuracy, $\mathcal{D}$ is $\mathcal{D}_{test}$ for testing accuracy.



(a) You start by finding a linear classifier with the best training accuracy using the whole data set (training using all four samples). What is the best training accuracy that can be achieved?

Solution: The best classifier (among many) will misclassify one of the four training data points, for a training accuracy $A_{train} = 0.75$.

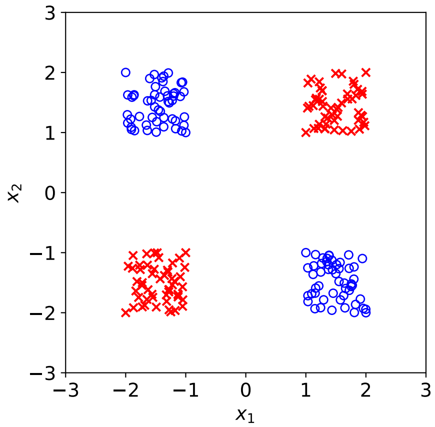
(b) You now conduct four-fold (leave-one-out) cross-validation. For each fold, suppose you find a linear classifier with the highest possible training accuracy. You decide to look at the *training* accuracy for each of these linear classifiers, and report an average of these **training** accuracies across the four folds. What is the final value that you report?

Solution: Leaving any one point out, we're always able to train a linear classifier that linearly separates the remaining three points. So the best average training accuracy is $A_{train} = 1$.

(c) You still consider the same set of four linear classifiers found in (b). But now you look at the **testing** accuracy for each of these linear classifiers on the corresponding fold's testing set, and report an average of these testing accuracies across the four folds (i.e., what we traditionally report when we use cross-validation). What is the final value that you report?

Solution: The "perfect" linear classifiers found from training on three points will always misclassify the held-out test data point. So the average testing accuracy is $A_{test} = 0$.

(d) Concerned that four data points may not be a large enough data set, you obtain a much larger number of data points as shown below, with 25% of the data in each of the four quadrants. In each quadrant, the data is bounded as shown within a square region of width 1 and height 1 from the corner points at $(\pm 1, \pm 1)$.



This much larger data set is randomly shuffled to ensure that the order of data points in the overall data set is random. Now you perform four-fold cross-validation on this data set, and in each fold you take the linear classifier with highest training accuracy on the training data. Approximately what is the average of training accuracies across the folds?

Solution: Now we're back to the situation in (a), where the best linear classifier will misclassify all or most of the points in one quadrant of the data set (which will contain approximately 25% of the training data), resulting in an average training accuracy $A_{train} \approx 0.75$.

(e) You still consider the same set of four linear classifiers that you found in (d). But now you look at the **testing** accuracy for each of these linear classifiers on the corresponding fold's testing set and report an average of these testing accuracies across the four folds. Approximately what value do you report?

Solution: With our data random shuffled, the best trained classifier will misclassify about 25% of the test samples, so the average test accuracy will also be $A_{test} = 0.75$. Unlike the cases with very small number of sample points, here the training and test accuracies are comparable, giving us some confidence that the best we can do with a linear classifier (without feature transformations) is about 75% accurate for this data.

(f) You desire a feature transformation that makes your data in (a) and/or (d) linearly separable, and consider various options. Each feature transformation supplements the original two components, x_1 and x_2, with a third component, to provide three features in total as shown below:

A: $[x_1, x_2, x_1^2]$

B: $[x_1, x_2, x_1^2 - x_2^2]$

C: $[x_1, x_2, x_1 x_2]$

D: $[x_1, x_2, x_1^2 + x_2^2]$

E: $[x_1, x_2, \frac{(x_1+x_2)^2}{2} + \frac{(x_1-x_2)^2}{1}]$

i. Mark all of the feature transformations defined above that will make the four data points in part (a) linearly separable:

○ A. ○ B. √ **C.** ○ D. √ **E.**

ii. Mark all of the feature transformations defined above that will make the many data points in part (d) linearly separable:

○ A. ○ B. √ **C.** ○ D. ○ E.

Solution: Choices A and B do not help; one can consider the four points at $(\pm 1, \pm 1)$ and see that those map to the same values for our additional feature component, even for different class samples.

Choice C makes both data sets linearly separable; all class “X” samples are mapped to positive $x_1 x_2$ and all “circle” samples to negative $x_1 x_2$.

Choice D gives the squared distance of the point from the origin; this does not help distinguish the two classes since many of both classes sit at the same distance from the origin.

Choice E is an ellipse, and is interesting in that it can separate the two classes in part (a) data but not in part (d) data. For the part (a) data, one can draw the ellipse for some value of $\frac{(x_1+x_2)^2}{2} + \frac{(x_1-x_2)^2}{1}$ that encompasses the upper right and lower left points (“X” points) in data from part (a), but not the “circle” class samples. However, expanding the ellipse so that it encompasses all of the upper right and lower left “X” points from part (d) data will cause some of the upper left and lower right data “circle” points in part (d) to be included, making this data not linearly separable with this added feature.

Separability can be seen more easily by considering mappings for all $(\pm 1, \pm 1)$ and $(\pm 2, \pm 2)$ points, to see if the added feature has a threshold that enables separation into the two classes. In the table below, the value of the third feature component is shown under A-E. Values enabled linear separability are bolded.

x_1	x_2	A	B	C	D	E	Class
-1	-1	1	0	**1**	2	**2**	X
-1	1	1	0	**-1**	2	**4**	○
1	1	1	0	**1**	2	**2**	X
1	-1	1	0	**-1**	2	**4**	○
-2	-2	4	0	**4**	8	8	X
-2	2	4	0	**-4**	8	16	○
2	2	4	0	**4**	8	8	X
2	-2	4	0	**-4**	8	16	○

Name: ______________________________

Store Cluster

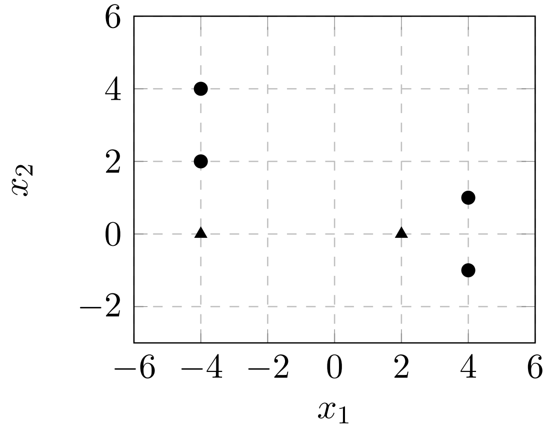
2. (20 points) Amazing, Inc., is going to open their first k stores in the greater Megacity region. They know that the ith customer (out of n total customers) lives at location $x^{(i)}$. They will place the jth store (for $j = 1, \ldots, k$) at location $\mu^{(j)}$. Amazing, Inc., would like to minimize the squared distance between customer locations and stores, and they decide to use k-means clustering to choose their store locations.

   Amazing, Inc., wants to focus on where to place their stores, and decides to formulate the problem as finding the μ that minimizes a loss function

$$L(\mu) = \sum_{i=1}^{n} \min_{j \in 1,\ldots,k} \left\| x^{(i)} - \mu^{(j)} \right\|^2 \quad .$$

   Here $L(\mu)$ is the value of the k-means objective after we have picked the *optimal* assignments of the data points to cluster means (that's what the $\min_j$ does), for the given set of cluster means μ.

   (a) Amazing, Inc., has four customers in Megacity, located at positions (x_1, x_2) as shown by the circles in the plot below, and plans to build two stores. They have initial guesses for these two store locations, as marked by the triangles below. What is the starting loss $L(\mu)$?

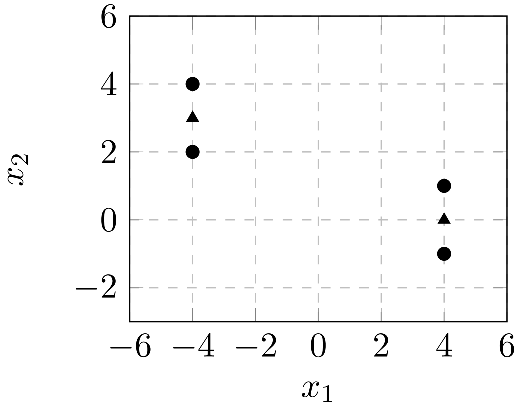


Solution:

$$\begin{aligned} L(\mu) &= 2^2 + 4^2 + (2^2 + 1^2) + (2^2 + 1^2) \\ &= 4 + 16 + 5 + 5 \\ &= 30 \end{aligned}$$

   (b) Amazing, Inc., runs the k-means clustering algorithm starting from the initial guess shown above, until convergence (no further improvements in loss $L(\mu)$ can be made). Mark the final locations of the stores (with triangles) on the plot below. What is the final loss $L(\mu)$?

Solution:



The final loss has improved to:

$$L(\mu) = 4 \cdot 1^2 = 4$$

(c) After several years, Amazing Inc., has grown tremendously. Now rather than tracking individual customers, they instead have a database consisting of r records: the ith record, $i = 1, \ldots, r$, provides both $c^{(i)}$ and $x^{(i)}$, where $c^{(i)}$ is the number of customers that are located at position $x^{(i)}$. They still want to put each store (indexed by $j = 1, \ldots, k$) at a position $\mu^{(j)}$ that minimizes the squared distances between customers and store locations, summed up over all customers. They just don't have individual customer data anymore.

Define a new objective $L_C(\mu)$ to account for the fact that now Amazing, Inc., has access only to record data instead of individual customer data. Please define any new expressions that you use within your formulation.

Solution: We need to include the $c^{(i)}$ term in the objective, as:

$$L_C(\mu) = \sum_{i=1}^{r} (\min_j \left\| x^{(i)} - \mu^{(j)} \right\|^2) \cdot c^{(i)} \quad .$$

(d) Amazing, Inc., test-runs their approach in the One-di City, which has a small number of customers. Their data set $\mathcal{D}$ consists of one-dimensional locations x and customer counts c as pairs, $(x, c) : \mathcal{D} = ((-1, 10), (1, 4))$. Amazing, Inc., is only going to build one store: where should it be located? Show your work.

Solution: For this case, our $L_C(\mu)$ can be expressed as:

$$\begin{aligned} L_C(\mu) &= c^{(1)} \cdot (\mu - x^{(1)})^2 + c^{(2)} \cdot (\mu - x^{(2)})^2 \\ &= 10 \cdot (\mu + 1)^2 + 4 \cdot (\mu - 1)^2 \\ &= 14\mu^2 + 12\mu + 14 \end{aligned}$$

The optimum occurs where $dL_C(\mu)/d\mu = 0$, or:

$$\begin{aligned} 28\mu + 12 &= 0 \\ \mu &= -12/28 = -3/7 \ . \end{aligned}$$

As a sanity check, this makes sense – the store should be located to the left of $x = 0$, since there are more customers to that side.

(e) Amazing, Inc., also has a single distribution center (DC) in Megacity, located at x_{DC}, that supplies all of their stores in Megacity. There is a cost associated with transporting goods from the DC to each of the stores, that grows with both squared distance and with the number of customers served by the store. Specifically, for each store, that cost is equal to the number of customers assigned to cluster j times the square of the distance from $\mu^{(j)}$ to x_{DC}. Amazing, Inc., would like to minimize both its own tranportation costs and the transportation cost of customers visiting their stores, the latter of which is the loss from (c). To that end, they decide to minimize the sum of these two costs.

Define the loss function $L_S(\mu)$ that expresses the overall loss that Amazing Inc. is seeking to minimize: the sum of the Amazing, Inc., transportation costs and the customer-to-store costs. Please define any new expressions that you use within your formulation. All of your definitions should be in equations that ultimately depend only on quantities defined in this problem, not just definitions in words. You may find it useful to define $y^{(i)} = \arg\min_j \left\| x^{(i)} - \mu^{(j)} \right\|^2$ for record i.

Solution: In addition to the $c^{(i)}$ term in the objective, we add the store-to-DC costs as:

$$L_S(\mu) = L_C(\mu) + \sum_{j=1}^{k} N_C^{(j)} \left\| \mu^{(j)} - x_{DC} \right\|^2 \ ,$$

where L_C is defined as in (a):

$$L_C(\mu) = \sum_{i=1}^{r} (\min_j \left\| x^{(i)} - \mu^{(j)} \right\|^2) \cdot c^{(i)} \ .$$

In L_S as defined above, we account for the total number of customers (not just number of points) in cluster j by defining

$$N_C^{(j)} = \sum_{i=1}^{r} c^{(i)} \cdot \mathbb{1}(y^{(i)} = j) \ ,$$

where

$$y^{(i)} = \arg\min_j \left\| x^{(i)} - \mu^{(j)} \right\|^2$$

is which cluster (store) the ith record is assigned to.

Some alternative expressions or notations for L_S are shown below. Let y^i denote the assignment of i^{th} record to its nearest cluster.

$$\begin{aligned}
L_S(\mu) &= \sum_{i=1}^{r} c^{(i)} \sum_{j=1:k} \mathbb{1}[y^i = j](||x^{(i)} - \mu^{(j)}||^2 + ||\mu^{(j)} - x_{DC}||^2) \\
&= \sum_{i=1}^{r} c^{(i)} \min_{j=1:k} ||x^{(i)} - \mu^{(j)}||^2 + \sum_{i=1}^{r} \sum_{j=1:k} \mathbb{1}[y^i = j] c^{(i)} ||\mu^{(j)} - x_{DC}||^2 \\
&= \sum_{i=1}^{r} c^{(i)} \min_{j=1:k} ||x^{(i)} - \mu^{(j)}||^2 + \sum_{j=1:k} ||\mu^{(j)} - x_{DC}||^2 \sum_{i=1}^{r} \mathbb{1}[y^i = j] c^{(i)} \\
&= \sum_{i=1}^{r} c^{(i)} \min_{j=1:k} ||x^{(i)} - \mu^{(j)}||^2 + \sum_{j=1:k} ||\mu^{(j)} - x_{DC}||^2 N_C^j \ .
\end{aligned}$$

(f) Amazing, Inc., returns to the One-di City to see where they should put their single store. The records are the same as in (d), but now we also know that their distribution center is located at $x_{DC} = 10$. If we aim to minimize the objective function desired in (e), at what location μ should their store be located?

Solution: Now our combined $L_S(\mu)$ can be expressed as:

$$\begin{aligned}
L_S(\mu) &= L_C(\mu) + (c^{(1)} + c^{(2)})(\mu - x_{DC})^2 \\
&= (14\mu^2 + 12\mu + 14) + (10 + 4)(\mu - 10)^2 \\
&= (14\mu^2 + 12\mu + 14) + (14\mu^2 - 280\mu + 1400) \\
&= 28\mu^2 - 268\mu + 1414.
\end{aligned}$$

The optimum occurs where $dL(\mu)/d\mu = 0$, or:

$$\begin{aligned}
56\mu - 268 = 0 \\
\mu = 268/56 = 67/14 = 4.79 \ .
\end{aligned}$$

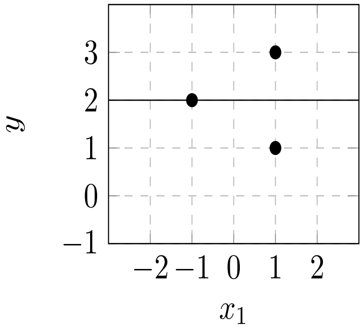
Again as a sanity check, this value looks consistent with putting the store much further right than before, to be closer to the DC.

Not All Regressions are Equal

3. (20 points) The questions below explore linear regressions to find a linear hypothesis $h(x_1) = \theta_1 x_1 + \theta_0$ that best matches a set of data, where $y^{(i)}$ is the output and $x_1^{(i)}$ is the input for each data point $i = 1, \ldots, n$ as shown. We will consider different loss functions. For each feature value $x^{(i)}$, our guess $g^{(i)} = h(x^{(i)})$, and our loss is $\mathcal{L}(g^{(i)}, y^{(i)})$. The objective function we seek to minimize is $J(\theta_1, \theta_0) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(g^{(i)}, y^{(i)}) + \lambda R(\theta_1)$, with $R(\theta_1) = \theta_1^2$ as a ridge regularization term. Various $\mathcal{L}$ and λ will be considered. Solutions to these questions do not require detailed calculations.

(a) Our first data set consists of three points as shown below. We use a squared error loss function, $\mathcal{L} = \mathcal{L}_{SE}(g^{(i)}, y^{(i)}) = (g^{(i)} - y^{(i)})^2$, and $\lambda = 0$. Write the equation for the best hypothesis, and sketch this equation on the plot:

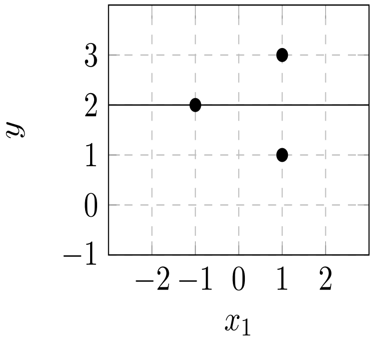
Solution:



Just the straight line $h(x_1) = 0 \cdot x_1 + 2$.

(b) Using the same data as in (a), we now use $\lambda = 1000$, and continue to use $\mathcal{L} = \mathcal{L}_{SE}$. Does the best hypothesis change? Explain why or why not. Whether it has changed or not, provide and justify an (approximate) hypothesis found by linear regression with this regularized objective, and sketch that on the plot below.

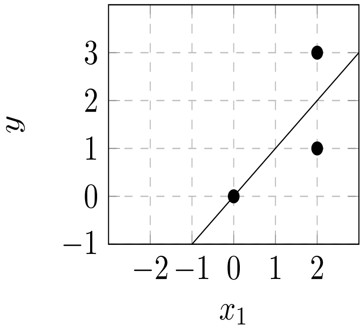
Solution:



The best hypothesis does not change – it remains the straight line $h(x_1) = 0 \cdot x_1 + 2$, because this also minimizes the regularization term (by setting the norm of θ_1 to zero) in addition to minimizing the sum of losses term.

(c) We now consider a second data set as shown below, and use a squared error loss function $\mathcal{L} = \mathcal{L}_{SE}$. We set $\lambda = 0$. Write the equation for the best hypothesis, and sketch that hypothesis on the plot:

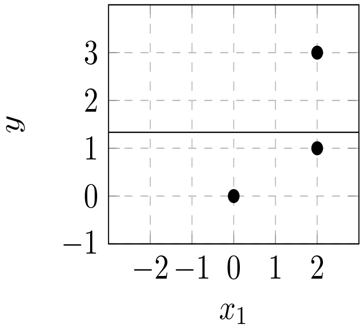
Solution:



Now, the best hypothesis is the line $h(x_1) = 1 \cdot x_1 + 0$, which splits the interval between points at (2,1) and (2, 3).

(d) Using the same data and loss function $\mathcal{L}_{SE}$ as in (c), we change the regularization parameter to $\lambda = 1000$. Does the best hypothesis change? Explain why or why not. Whether it has changed or not, provide and justify an (approximate) hypothesis found by linear regression with this objective and regularization, and sketch that on the plot below.

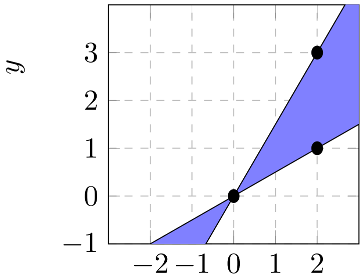
Solution:



The best hypothesis does indeed change. Now the penalty on non-zero θ_1 is quite large, driving that to be near to zero. The θ_0 term will change to reduce squared error between θ_0 and the y values, becoming close to the mean of the y values; thus the hypothesis will be $h(x_1) \approx 0 \cdot x_1 + \frac{4}{3}$.

(e) Continuing with this same data, we now change our loss function to instead be absolute error, $\mathcal{L} = \mathcal{L}_{AE}(g^{(i)}, y^{(i)}) = |g^{(i)} - y^{(i)}|$. We continue with the objective function J (but now using our $\mathcal{L}_{AE}$), and initially with $\lambda = 0$. What is the minimum J that can be achieved? Write the equation for a hypothesis that minimizes J, and sketch that below.

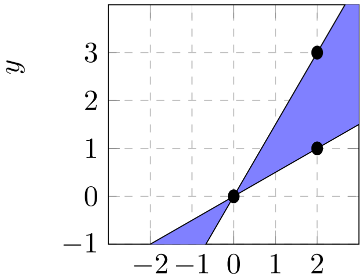
Solution:



The minimum objective value achievable is $J = 2/3$. Any line defined by $h(x_1) = \theta_1 x_1$ for $\theta_1 \in [0.5, 1.5]$ results in this minimum objective value for J.

(f) Is the hypothesis giving minimum J in (e) unique? If it is unique, justify why. If not unique, sketch and shade the region in which all of the minimum-J hypotheses must reside.

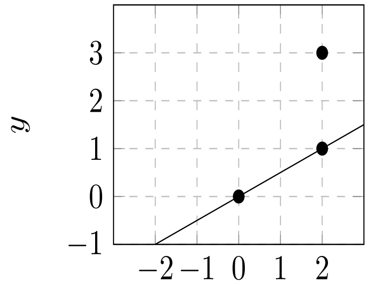
Solution:



The "best" hypothesis is not unique. All lines of the form $h(x) = \theta_1 x_1$ with $\theta_1 \in [0.5, 1.5]$ achieve the minimum $J = 2/3$.

(g) Finally, using the same data and $\mathcal{L}_{AE}$ loss function as in (e), we change the regularization parameter to $\lambda = 0.001$. Now is there a unique best hypothesis? Explain why or why not. Provide and justify an (approximate) best hypothesis found by linear regression with this objective and regularization, and sketch that on the plot below.

Solution:



The best hypothesis changes – now the penalty on non-zero θ_1 is enough to split the ties on J from the previous part, picking the unique one that still minimizes sum of absolute errors, but now that also uses the θ_1 with smallest norm. So now the best hypothesis is $h(x_1) = 0.5 \cdot x_1 + 0$.

Gradient Portfolio

4. (20 points) Joe Smallbucks has a limited budget (\$1000, though he can borrow money with some penalty, or not use all of his budget, also with some lost opportunity penalty). He wants to build a stock portfolio. Joe is considering a set of n stocks with known prices. His task is to decide how much of each stock, $x = \begin{bmatrix} x_1, & x_2, & \ldots, & x_n \end{bmatrix}^\top$, to hold. Let p_i be the price of one unit of stock i, and p be the column vector that collects the p_i. If Joe holds x_i units of stock i, we say that the value of that stock in Joe's portfolio is $x_i p_i$. The value of Joe's whole portfolio is the sum of the values of all individual stocks. And we note that x_i can be positive or negative (also known as "long" or "short" positions, respectively).

   Even though he has a small budget, Joe hopes that by using gradient descent ideas, he can find an optimal portfolio.

   (a) Joe first seeks to formulate his objective, as a function of his decision variable x. He decides to impose a "budget penalty" for being over or under his budget that is 0.1 times the square of how much his total portfolio value differs from his budget (\$1000). Joe also defines a "share holding penalty" that squares the x_i for each stock in his portfolio, then takes the total sum of these, and weights this sum by 0.5. His final objective is the sum of his budget penalty and his share holding penalty.

   Using the variables defined above, write an expression for the objective function that Joe is seeking to minimize, in terms of the vectors x and p. Note: write your solution in term of the **vectors** x and p, **not** in terms of their individual components x_i and p_i.

   Solution: $J(x) = 0.1 \cdot (p^\top x - 1000)^2 + 0.5 \cdot ||x||^2$. The first term corresponds to his "budget penalty" and the second term is the "share holding penalty".

   (b) To perform gradient descent, Joe needs the gradient of J with respect to his decision variable, x. Derive an expression for $\nabla_x J(x)$: Again, make sure to write your final solution in terms of the vectors x and p, and not their individual components x_i and p_i.

   Solution: $\nabla_x J(x) = 0.2 \cdot (p^\top x - 1000) \cdot p + x$.

   (c) Next, Joe needs an update rule. Write the update rule for gradient descent calculation of x_{new} related to the previous value x_{old}, in terms of step-size (learning rate) parameter η and other expressions derived above.

   Solution: $x_{new} = x_{old} - \eta \nabla_x J(x)$.

   (d) Let's try out gradient descent, with a learning rate of $\eta = 0.1$. We consider a particular case where $p = [1, 3]^\top$, and our initial guess for holdings is $x = [0, 0]^\top$. After one iteration of gradient descent, what is x? Show your work.

Solution:

$$\begin{aligned} x_{new} &= x_{old} - \eta \nabla_x J(x) \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.1 \left(0.2 \cdot \left(\begin{bmatrix} 1 & 3 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 1000 \right) \begin{bmatrix} 1 \\ 3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right) \\ &= 20 \cdot \begin{bmatrix} 1 \\ 3 \end{bmatrix} \\ &= \begin{bmatrix} 20 \\ 60 \end{bmatrix} \end{aligned}$$

(e) Now suppose we take one more step of gradient descent. What is x after this second iteration? Show your work.

Solution:

$$\begin{aligned} x_{new} &= \begin{bmatrix} 20 \\ 60 \end{bmatrix} - 0.1 \left(0.2 \cdot \left(\begin{bmatrix} 1 & 3 \end{bmatrix} \begin{bmatrix} 20 \\ 60 \end{bmatrix} - 1000 \right) \begin{bmatrix} 1 \\ 3 \end{bmatrix} + \begin{bmatrix} 20 \\ 60 \end{bmatrix} \right) \\ &= \begin{bmatrix} 20 \\ 60 \end{bmatrix} + \begin{bmatrix} 14 \\ 42 \end{bmatrix} \\ &= \begin{bmatrix} 34 \\ 102 \end{bmatrix} \end{aligned}$$

Name: ________________________________

Corrupted Classification

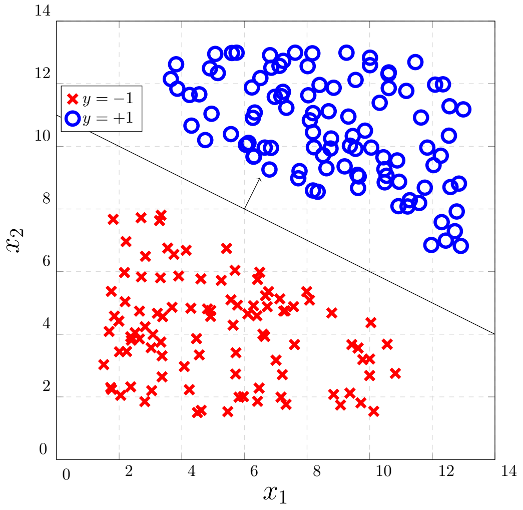
5. (20 points) Consider the data set $\mathcal{D}_n = \left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{n}$ comprised of $n = 200$ data points. For each pair $(x^{(i)}, y^{(i)}) \in \mathcal{D}_n$, each feature vector $x^{(i)} = [x_1^{(i)}, x_2^{(i)}]^\top$ has two components. The labels $y^{(i)} \in \{-1, +1\}$ encode two different classes. There are 100 data points in each class. In the plot below, the -1 class is represented by an X and the $+1$ class by a circle.

   (a) Let the hypothesis class of linear classifiers be defined as,

$$h(x; \theta, \theta_0) = \text{sign}(\theta^\top x + \theta_0) = \begin{cases} +1, & \theta^\top x + \theta_0 > 0, \\ -1, & \text{otherwise.} \end{cases}$$

   Specify θ, θ_0 that define a linear classifier that has 100% accuracy on the data set as shown below. Draw the linear separator and its normal vector for your classifier on the plot below.

Solution:



Many viable choices. One option is plotted, which corresponds to $\theta = [1, 2]^\top$, $\theta_0 = -22$ (the positive class appears above and to the right of the line $x_2 = -0.5x_1 + 11$).

(b) You decide to learn a linear logistic classifier trained on the data set above, by minimizing the following objective function:

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda R(\theta) \ .$$

Here, $\mathcal{L}_{\text{nll}}(g^{(i)}, y^{(i)}) = -(y^{(i)} \log(g^{(i)}) + (1 - y^{(i)}) \log(1 - g^{(i)}))$ is the negative log-likelihood loss function, $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function, and $R(\theta) = \|\theta\|^2$ is the ridge regression regularizer. The value for λ is set to be very small.

You split the data into a training and testing set. For each of the following, determine whether or not the linear classifier trained on the chosen training data set would reliably obtain low training error and/or low testing error across multiple runs of the indicated approach. Circle A if the proposal would reliably obtain low training error, and circle B if the proposal would reliably obtain low testing error. You can circle both, one, or neither. **In every case, explain your response.**

i. Randomly select, without replacement, 50% of the data set to be the training data set. The remaining data is used as the testing data set.
A. low training error **B. low testing error**

Solution: This random sampling should create training and testing subsamples that both reasonably approximate the distribution of available data, with plenty of positive and negative samples, with relatively good separation between the positive and negative samples.

ii. Take all 100 of the data points labeled +1 and a single data point at $x = [-2, 2]^\top$ labeled $y = -1$ to be the training data set. The remaining data is the testing data set.
A. low training error B. low testing error

Solution: The trained classifier will be able to well-separate the positive group from the lone negative data point, with low training error; it can form a perfect separator. But the separator will be very near the lone data point to maximize the probability of all the +1 data points. So many of the other -1 points in the test set will get classified as +1 and therefore not have a low testing error.

iii. Take exactly 60 data points (uniformly at random without replacement) from each class to be the training data set. The remaining data is the testing data set.
A. low training error **B. low testing error**

Solution: This random sampling should also create training and testing subsamples that both reasonably approximate the distribution of available data, similar to (i) above, with low training and testing error.

(c) Now, consider a *corrupted* version of the data set, where 20 out of 200 data points (selected uniformly at random, without replacement) have had their class label *switched*; data points that should belong to class +1 are now labeled −1, and vice versa. The feature vectors,

$x^{(i)}$, remain the same for all 200 data points. At the time of training, you are not told which labels have been switched.

Calculate the value of the following objective function,

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_{01}(\text{sign}(\theta^\top x^{(i)} + \theta_0), y^{(i)}),$$

for the linear classifier that you drew in part (a), now on the corrupted data set.

Solution: None of the data points are moved (i.e., $x^{(i)} = [x_1^{(i)}, x_2^{(i)}]^\top$ remains the same), so this is essentially asking for (1 - accuracy) of the classifier from (a) when 20 random data points have had their label (deterministically) flipped; their classifier will misclassify exactly 20 data points, so $J(\theta, \theta_0) = 1/10$.

(d) You decide to learn a linear logistic classifier trained on the corrupted data set, but now using a *weighted* objective function,

$$J_w(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} w^{(i)} \mathcal{L}_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda R(\theta) \ .$$

Here, $\mathcal{L}_{\text{nll}}(g^{(i)}, y^{(i)})$ is the negative log-likelihood loss function, and $\sigma(z)$ is the sigmoid function, both defined in part (b). The value for λ is set to be very small.

Each data point is assigned a real-valued, non-negative weight $w^{(i)}$ that appears in the objective J_w. Describe the intuition for what these weights do (e.g., which points get a high weight and which points get a low weight?), for each of the following choices:

A. Set $w^{(i)} = 1$ for all data points.

B. For each data point $x^{(i)}$, identify the five closest data points. Compute the mean $\bar{y}^{(i)}$ of the labels for these five data points. Set $w^{(i)} = 1 - \frac{|\bar{y}^{(i)} - y^{(i)}|}{2}$.

C. For each data point $x^{(i)}$, set $w^{(i)} = (x_1 - 7)^2 + (x_2 - 7)^2$, i.e., the squared-distance of the data point to the point $(7, 7)$.

Solution: In Choice A, the weights will have no impact or benefit to the training. See further explanation in solution to part (e) below.

Choice B will consider the data points surrounding any given data point; the weighting will be higher for data points that are surrounded by those with the same label, and we will have a lower weighting for data points that are labeled differently than those surrounding it. This should reduce the influence of corrupted data, improving training and giving us a classifier that better reflects the uncorrupted data.

Finally, Choice C places larger weights on points that are further from the center of the plot, which is independent of the location of the corrupted data points – it is unlikely that this weighting will do anything helpful for this classification problem.

(e) Consider using the entire corrupted data set as the training data set for minimizing the objective function in part (d), with $w^{(i)} = 1$ for all $i = 1, ..., n$. You use gradient descent, where you initialize using the θ, θ_0 from the linear classifier that you drew in part (a). Assume that you choose good values for the learning rate and the number of iterations to run. Would you expect the separator and direction of normal vector for the newly learned linear classifier to be similar to the separator and direction of normal vector that you drew in part (a)? Why or why not?

Solution: A good initialization is key here. The separator and normal vector from part (a) should be a very good starting point, with the corrupted data points randomly distributed. During further training, the corrupted data points will have only minor impact (and are likely to balance out in terms of moving the separator in different directions). Thus we're likely to move the separator (or direction of normal vector) relatively little from what we had in part (a).