HOMEWORK 3
DECISION TREES, K-NN, PERCEPTRON, REGRESSION[1]

10-301 / 10-601 INTRODUCTION TO MACHINE LEARNING (FALL 2021)
http://mlcourse.org

OUT: Sept. 20, 2021
DUE: Sept. 26, 2021
TAs: Weyxin, Kevin, Joseph, Mukund, Brendon

START HERE: Instructions

- **Collaboration policy:** SOLO questions must be solved individually, and GROUP questions may be solved with your assigned groups. Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 2.1"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the Academic Integrity Section on the course site for more information: http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html#7-academic-integrity-policies
- **Late Submission Policy:** You may use up to 2 late days on this assignment. See the late submission policy here: http://www.cs.cmu.edu/~mgormley/courses/10601/syllabus.html#late-homework-policy
- **Submitting your work:**
  - **Written:** For written problems such as short answer, multiple choice, derivations, proofs, or plots, we will be using Gradescope (https://gradescope.com/). Please use the provided template. Submissions can be handwritten onto the template, but should be labeled and clearly legible. If your writing is not legible, you will not be awarded marks. Alternatively, submissions can be written in LaTeX. Regrade requests can be made, however this gives the TA the opportunity to regrade your entire paper, meaning if additional mistakes are found then points will be deducted. Each derivation/proof should be completed on a separate page. For short answer questions you **should not** include your work in your solution. If you include your work in your solutions, your assignment may not be graded correctly by our AI assisted grader.

[1] Compiled on Sunday 26th September, 2021 at 21:19

Instructions for Specific Problem Types

For "Select One" questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

- ● Matt Gormley / Henry Chai
- ○ Marie Curie
- ○ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

- ● Matt Gormley / Henry Chai
- ○ Marie Curie
- ~~●~~ Noam Chomsky

For "Select all that apply" questions, please fill in all appropriate squares completely:

Select all that apply: Which are scientists?

- □ Stephen Hawking
- ■ Albert Einstein
- □ Isaac Newton
- □ None of the above

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are scientists?

- ■ Stephen Hawking
- ■ Albert Einstein
- ■ Isaac Newton
- ~~■~~ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

$\boxed{\text{10-601}}$ $\boxed{\text{10-}\cancel{7}\text{601}}$

1 Decision Tree (Revisited)

1. (4 points) Consider the following 4×4 checkerboard pattern. Suppose our goal is to perfectly classify the range shown such that all black regions are labeled as $+1$ and all white regions are labeled as -1.

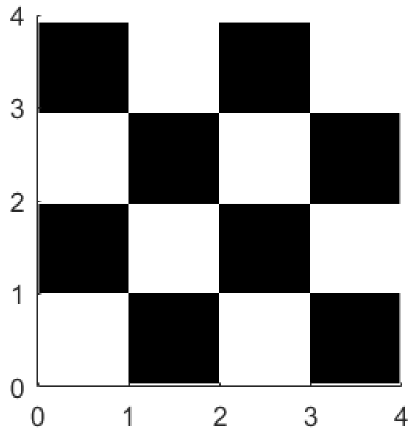


Figure 1: Checkerboard Pattern

(a) (2 points) **[SOLO]** What is the minimum depth of decision tree that perfectly classifies the 4×4 colored regions in Figure 1, using features that only inspect either x or y but not both x and y? (How you use each of x and y to construct a feature is up to you.)

Your Answer

2

(b) (2 points) **[SOLO]** What is the minimum depth of decision trees to perfectly classify the colored regions in Figure 1, using ANY features of x and y?

Your Answer

1

Consider the graph below analyzing the size of tree vs. accuracy for a decision tree which has been pruned back to the red line.

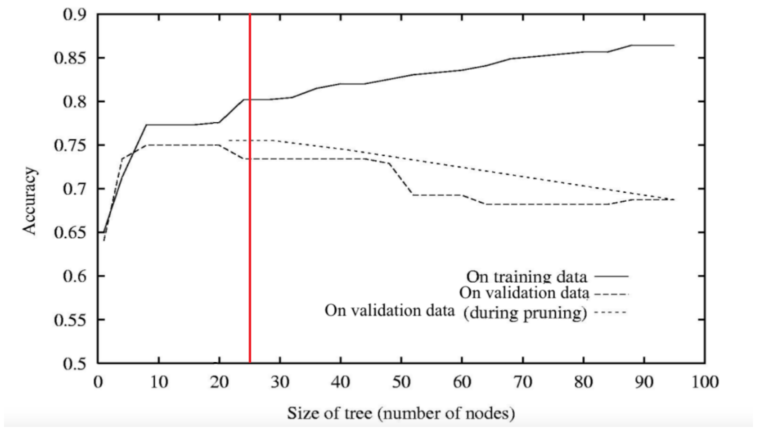


Figure 2: Pruned decision tree

2. (1 point) **[SOLO]** Refer to Figure 2. Let's say that we have a third dataset D_{new} (from the same data distribution), which is not used for training or pruning.

   If we evaluate this new dataset, approximately what is the accuracy when the size of the tree is at 25 nodes, and why? Select one.

   Select one:

   ○ Around 0.76 (slightly higher than the accuracy for validation data at 25 nodes)

   ○ Around 0.73 (the same as the accuracy for validation data at 25 nodes)

   [✓ Around 0.70 (slightly lower than the accuracy for validation data at 25 nodes)]

   ○ None of the above

3. (1 point) **[SOLO]** Which of the following gives us the best approximation of the true error?

   ○ Line corresponding to training data

   ○ Line corresponding to validation data

   [✓ Line corresponding to new dataset D_{new}]

4. (2 points) **[SOLO]** Which of the following are valid ways to avoid overfitting? Select all that apply.

   Select all that apply:

   □ Decrease the training set size.

   [✓ Set a threshold for a minimum number of samples required to split at an internal node.]

   [✓ Prune the tree so that cross-validation error is minimal.]

   □ Maximize the tree depth.

   □ None of the above.

5. (3 points) **Ensemble of Decision Tree.** Say we have a dataset shown below. There are 12 data points, with 6 in label ”-” and 6 in label ”+”. We would like to use Decision Trees to solve this binary classification problem. However, in our problem setting, each Decision Tree has access to only ONE line. That is to say, our Decision Tree would have access to only one attribute, and so has max-depth of 1.

   By accessing this line, the Decision Tree could know (and only know) whether the data point is on the right side of this line or the left side. (Unofficial definition: let’s assume the right side of a line shares the same direction with the **green** normal vector of that line.)

   Finally, please use majority vote strategy to make classification decision at each leaf.

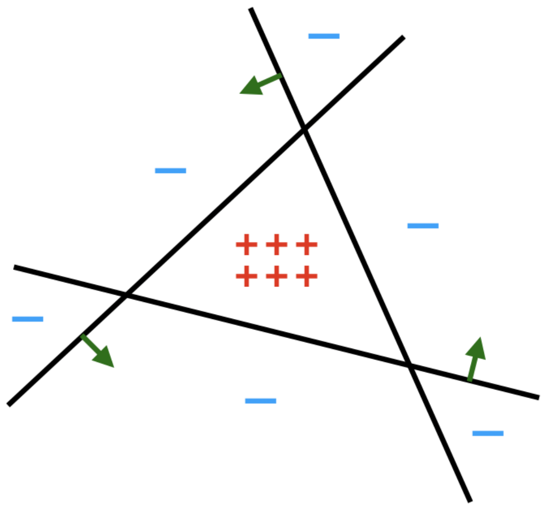


Figure 3: Decision Tree Ensemble

   (a) (1 point) **[SOLO]** In Figure 3, if we train only 1 Decision Tree, what is the best/lowest error rate? Note that we have in total 12 data points. Round to 4 decimal places after the decimal point.

Best/Lowest Error Rate
0.25

   (b) (2 points) **[SOLO]** If we could use 2 Decision Trees in Figure 3, what is the best/lowest error rate? If we have two Decision Trees, then each would predict each data point with ’+’ or ’-’. Then, we would like to combine these predictions as the final result. If both trees predict ’+’, then the result is ’+’. The same with ’-’. However, if one predicts ’+’ while one predicts ’-’, then we always choose ’-’ as the final result to break ties. Round to 4 decimal places after the decimal point.

Best/Lowest Error Rate
0.0833

(c) (2 points) **[SOLO]** Now let's train 3 Decision Trees as a forest in Figure 3. What is the best/lowest error rate? The ensemble strategy is now unanimous voting. That is, if every Decision Tree agrees, then the model predicts a positive label. However, if one of them has a different answer from the other two, then we predict negative. That means, we train each Decision Tree individually, and each Decision Tree choose one unique line as its decision boundary such that it would try its best to achieve maximum accuracy. And, next, if all the Decision Trees agree, then we assign the point a positive label. Round to 4 decimal places after the decimal point.

Best/Lowest Error Rate

0

6. (6 points) Consider a binary classification problem using 1-nearest neighbors with the Euclidean distance metric. We have N 1-dimensional training points $x^{(1)}, x^{(2)}, \ldots x^{(N)}$ and corresponding labels $y^{(1)}, y^{(2)}, \ldots y^{(N)}$ with $x^{(i)} \in \mathbb{R}$ and $y^{(i)} \in \{0, 1\}$. Assume the points $x^{(1)}, x^{(2)}, \ldots x^{(N)}$ are in ascending order by value. If there are ties during the 1-NN algorithm, we break ties by choosing the label corresponding to the $x^{(i)}$ with lower value.

(a) (2 points) **[GROUP]** Is it possible to build a decision tree that behaves exactly the same as the 1-nearest neighbor classifier? Assume that the decision at each node takes the form of "$x \leq t$ or $x > t$," where $t \in \mathbb{R}$.

[✓ Yes]

No

If your answer is yes, please explain how you will construct the decision tree. If your answer is no, explain why it's not possible.

Your answer:

I think it is possible to build a decision tree that behaves like the 1-nearest-neighbor.
In order to determine the t to split the node at ($x \leq t or x > t$), we would iterate from the lowest value of x all the way till the last one. Since these values are in ascending order, we could just take the mean of the difference between two consecutive values of x. Using this value as t would satisfy the requirement plus, the tie-breaking condition as well. However, we need to figure out the best possible t as there will be a total of $N - 1$ possibilities for t at the first node. In order to choose the best splitting t, we could check the information gain for using each of these as the splitting value. The one with the highest information gain would be the best t.
The same can be repeated at every following node till we have 0 entropy (allowing overfitting to match the 1-NN behavior).

(b) (2 points) **[GROUP]** Let's add a dimension! Now assume the training points are 2-dimensional where $\mathbf{x}^{(i)} = (x_1^{(i)}, x_2^{(i)}) \in \mathbb{R}^2$ and the decision at each node takes the form of "$x_j \leq t$ or $x_j > t$," where $t \in \mathbb{R}$ and $j \in \{1, 2\}$. Give an example with at most 3 training points for which it isn't possible to build a decision tree that behaves exactly the same as a 1-nearest neighbor classifier.

Your answer:

Consider the training set consisting of 2 points - $(x^{(1)}, y_{(1)}) = ((1,1),1)$, $(x^{(2)}, y_{(2)}) = ((3,2),0)$.
Since the splitting decision can be made using only one of x_1 and x_2 at a time, the resulting decision boundary at a node would be either a horizontal or vertical line. In this example, choosing any line woul lead to a split of 1 1 and 1 0. So, say, the decision boundary calculated at the first node is $x_1 = 2$. Therefore, $x_1 \leq 2$ would be marked with $y = 1$ and for $x_1 > 2$, $y = 0$. So, consider a test point - $x^{(3)} = (2.1, -1)$. As per the decision tree, $y^{(3)} = 0$. However, as per 1-NN, the point closest to $x^{(3)}$ would be $x^{(1)}$. Thus, according to 1-NN, $y^{(3)} = 1$.
Hence, it's not possible to build a decision tree in this manner.

(c) (2 points) **[GROUP]** Assuming we have 2-dimensional training points $\mathbf{x}^{(i)} = (x_1^{(i)}, x_2^{(i)}) \in \mathbb{R}^2$ and the decision at each node takes the form of "$x_j \leq t$ or $x_j > t$," where $t \in \mathbb{R}$ and $j \in \{1, 2\}$, under what conditions **is** it possible to build a decision tree that behaves exactly the same as a 1-nearest neighbor classifier? Explain why it is possible to build the decision tree as stated in part (a) but, in general, not possible in part (b).

Your answer:

One of the conditions under which the decision tree would act the same as the 1-NN classifier would be if all the points lie on the same horizontal or vertical line - essentially converting this back to a 1-D problem. Another similar condition would be if the data is linearly separable by either a horizontal or vertical line. In either of the above condition, the decision tree would be able to classify the data similar to the 1-NN.
It is possible to build a decision tree similar to a 1-NN in case of (a) as the decision tree can only look at one dimension at a time and since all the data points were 1-D, the data was linearly separable in a single dimension. However, that wasn't the case with (b) or with higher any other higher dimension. To separate such data, the decision tree would have to be able to use a combination of dimensions to make a decision (essentially a line with any slope). However, this is not possible and so, constructing a decision tree similar to 1-NN is not possible

2 k-Nearest Neighbors

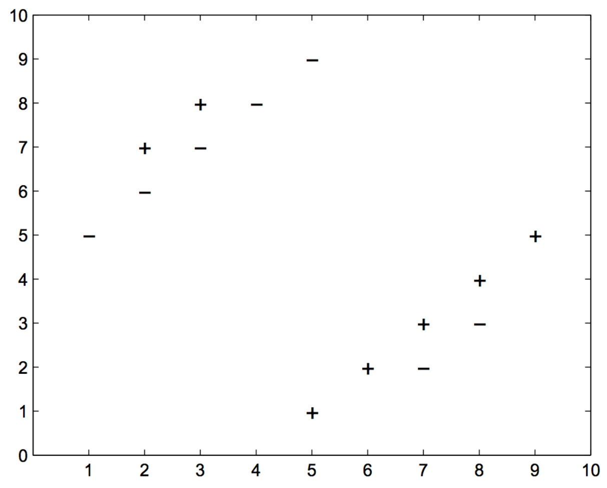


Figure 4: k-NN Dataset

1. (2 points) **[SOLO]** Consider a k-nearest neighbors (k-NN) binary classifier which assigns the class of a test point to be the class of the majority of the k-nearest neighbors, according to a Euclidean distance metric. Using Figure 4 shown above to train the classifier and choosing $k = 6$, what is the classification error on the training set?

   Answer as a decimal with precision 4, e.g. (6.051, 0.1230, 1.234e+7)

   Your answer:

   0.2857

2. (1 point) **[SOLO]** Again using Figure 4, what is the value of k that minimizes the training error? Let's assume we break ties by selecting one of the labels uniformly at random.

   Your answer:

   1

3. (2 points) **[SOLO]** Let's say that we have a new test point (not present in our training data) $\mathbf{x}^{\text{new}} = [3, 9]^T$ that we would like to apply our k-NN classifier to, as seen in figure 5.

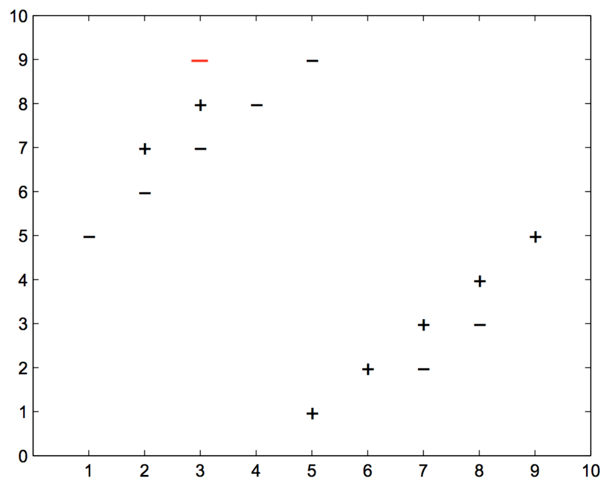


Figure 5: k-NN Dataset with Test Point

For which values of k is this test point correctly classified by the k-NN algorithm?

Select all that apply:

- ☐ $k = 1$
- ■ $k = 5$
- ■ $k = 9$
- ☐ $k = 12$
- ☐ None of the above

4. (5 points) Assume we have a training set and a test set drawn from the same distribution, and we would like to classify points in the test set using a k-NN classifier.

   (a) (1 point) **[SOLO]** In order to minimize the classification error on this test set, we should always choose the value of k which minimizes the training set error.

   Select one:

   - ○ True
   - ● False

(b) (2 points) **[SOLO]** Instead of choosing the hyper-parameters by merely minimizing the training set error, we instead consider splitting the training-all data set into a training and a validation data set, and choose the hyper-parameters that lead to lower validation error. Is choosing hyper-parameters based on validation error better than choosing hyper-parameters based on training error? Justify your opinion with no more than 3 sentences.

Select one:

[✓ Yes]

○ No

Your answer:

Yes, I believe choosing a hyperparameter based on validation error is better than that based on training error.
As the training error decreases, the model starts overfitting to the training data and so, even though the training error is low, the performance of the model is not good. Measuring performance using validation data i.e. data the model hasn't seen during training, is better as it actually indicates whether the model is generalizing well.

(c) (2 points) **[SOLO]** Your friend Sally suggests that instead of splitting the training set into separate training and validation sets, we should instead use the test set as the validation data for choosing hyper-parameters. Is this a good idea? Justify your opinion with no more than 3 sentences.

Select one:

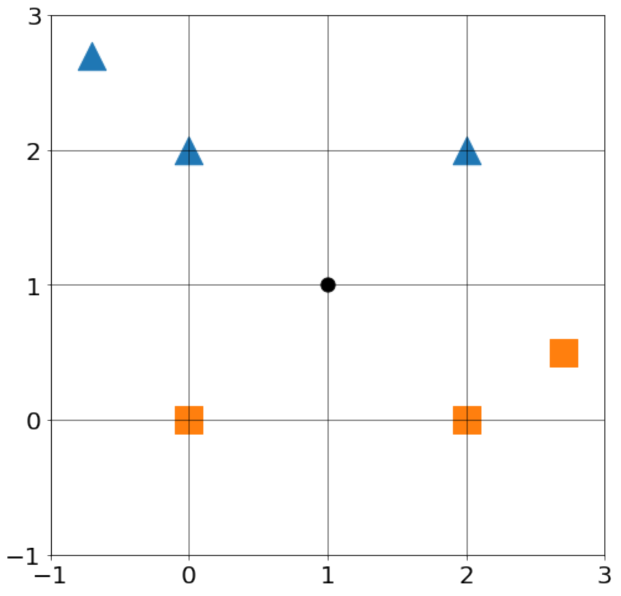
○ Yes

[✓ No]

Your answer:

No, test data shouldn't be used for choosing hyperparameters.
While validation data is a good measure of generalization, by choosing hyperparameters based on low validation error, we're inducing bias. Thus, using test data as validation data, would induce bias. The point of test data to have completely independent data just for testing the final, fully-trained model's generalization capability.

5. (3 points) **[SOLO]** Consider a binary k-NN classifier where $k = 4$ and the two labels are "triangle" and "square". Consider classifying a new point $\mathbf{x} = (1, 1)$, where two of the $\mathbf{x}$'s nearest neighbors are labeled "triangle" and two are labeled "square" as shown below.



Which of the following methods can be used to break ties or avoid ties on this dataset?

1. Assign x the label of its nearest neighbor
2. Flip a coin to randomly assign a label to $\mathbf{x}$ (from the labels of its 4 closest points)
3. Use $k = 3$ instead
4. Use $k = 5$ instead

Select one:

- ○ 1 only
- ○ 2 only
- ○ 2, 3, 4
- [✓ 2, 4]
- ○ 4 only
- ○ 1, 2, 3, 4
- ○ None of the above

6. (2 points) **[SOLO]** In this question, we would like to compare the differences among k-NN, the perceptron algorithm, and linear regression.

   Select all that apply:

   - ■ For classification tasks, both k-NN and the perceptron algorithm can have linear decision boundaries.
   - □ For classification tasks, both k-NN and the perceptron algorithm always have linear decision boundaries.
   - ■ All three models can be susceptible to overfitting.
   - □ In all three models, after the training is completed, we must store the training data to make predictions on the test data.
   - □ None of the above.

7. (3 points) **[SOLO]** Please select all that apply about k-NN in the following options.
   Select all that apply:

   - ■ A larger k gives a smoother decision boundary.
   - ■ To reduce the impact of noise or outliers in our data, we should increase the value k.
   - □ If we make k too large, we could end up overfitting the data.
   - ■ We can use cross-validation to help us select the value of k.
   - □ We should never select the k that minimizes the error on the validation dataset.
   - □ None of the above.

8. (8 points) Consider the following data concerning the relationship between academic performance and salary after graduation. High school GPA and university GPA are two numerical predictors and salary is the numerical target. Note that salary is measured in thousands of dollars per year.

Student ID	**High School GPA**	**University GPA**	**Salary**
1	2.5	3.8	45
2	3.6	2.4	55
3	3.3	3.5	91
4	4.0	4.0	142
5	2.2	3.2	88
6	3.8	3.0	2600
7	3.0	2.0	163
8	3.3	2.8	67
9	3.9	3.8	unknown

(a) (2 points) **[GROUP]** Among Students 1 to 8, who is the nearest neighbor to Student 9, using Euclidean distance? Answer the Student ID only.

Your answer:

4

(b) (3 points) **[GROUP]** Now, our task is to predict the salary Student 9 earns after graduation. We apply k-NN to this regression problem: the prediction for the numerical target (salary in this example) is equal to the average of salaries for the top k nearest neighbors. If $k = 4$, what is our prediction for Student 9's salary? Round your answer to the nearest integer. Be sure to use the same unit of measure (thousands of dollars per year) as the table above.

Your answer:

725

(c) (2 points) **[GROUP]** Suppose that the first 8 students shown above are only a subset of your full training data set, which consists of 10,000 students. We apply k-NN regression using Euclidean distance to this problem and we define training loss on this full data set to be the mean squared error (MSE) of salary. Now consider the possible consequences of modifying the data in various ways. Which of the following changes **could** have an effect on training loss on the full data set as measured by mean squared error (MSE) of salary?

Select all that apply:

■ Rescaling only "High School GPA" to be a percentage of 4.0

■ Rescaling only "University GPA" to be a percentage of 4.0

☐ Rescaling both "High School GPA" and "University GPA", so that each is a percentage of 4.0

☐ None of the above.

9. (3 points) **[GROUP]** Suppose we have N training examples, and each one has M features. Derive the prediction time complexity of running a naive k-NN binary classifier with Euclidean distance on one test point (for a fixed k) and use pseudo-code to justify your explanation.

Your answer:

```
For i from 1 -> N:
    For j from 1 -> M:
        distance[i] -> distance[i] +
         square(testPoint[j] - trainData[i][j])
    distance[i] = sqrt(distance[i])
sortedArgs = argsort(distance)
For i from 1 -> k:
    if trainLabel[sortedArgs[i]] == 0:
        count0 -> count0 + 1
    else:
        count1 -> count1 + 1
if count0 >= count1:
    return 0
return 1
```

From the above pseudocode, the total number of operations is $(NM + N\log(N) + K)$. The NM comes from the nested loop over every training example and features. The $N\log(N)$ comes from the sorting the N distances. The K comes from class voting. The K can be ignored for the complexity as K¡N and it's linear. Also, the $N\log(N)$ can be improved upon by using a priority queue of size K. However, for this naive K-NN, the complexity would be $O(NM + N\log(N))$.

10. (3 points) **[GROUP]** We have been dealing with k-NN with features that are all continuous values. Can we use k-NN on a data set that has categorical features that take three or more values? If yes, what is a distance metric that could be used to compute distance between categorical data points. Explain your reasoning.

Your answer:

Yes, KNN can work with categorical data. One of the simplest distance metric would be a simple binary loss - 0 if attribute category's same else 1. This would be really great in situations wherein all categories are equally punishable as you only care for an exact match. In a case where the categories are not equally punishable (eg. following categories - Strongly Disagree (0), Disagree(1), Nothing(2), Agree(3), Strongly Agree(4)), an absolute difference would make an ideal distance metric (Strongly Agree and Agree would have a "distance" of 1 while Strongly Agree and Strongly Disagree would have 4). Thus, the distance metric would have to be tailored according to what the categories actually mean. In fact, in this manner, we could evaluate categorical and continuous data together.

3 Perceptron

1. (1 point) **[SOLO]** Consider running the online perceptron algorithm on some sequence of examples S (an example is a data point and its label). Let S' be the same set of examples as S, but presented in a different order.

   True or False: the online perceptron algorithm is guaranteed to make the same number of mistakes on S as it does on S'.

   Select one:

   ○ True

   [✓ ● False]

2. (3 points) **[SOLO]** Suppose we have a perceptron whose inputs are 2-dimensional vectors and each feature vector component is either 0 or 1, i.e., $x_i \in \{0, 1\}$. The prediction function $y = \text{sign}(w_1x_1 + w_2x_2 + b)$, and

$$\text{sign}(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

   Which of the following functions can be implemented with the above perceptron? That is, for which of the following functions does there exist a set of parameters w, b that correctly define the function. **Select all that apply:**

   [✓ ■ AND function, i.e., the function that evaluates to 1 if and only if all inputs are 1, and 0 otherwise.]

   [✓ ■ OR function, i.e., the function that evaluates to 1 if and only if at least one of the inputs are 1, and 0 otherwise.]

   □ XOR function, i.e., the function that evaluates to 1 if and only if the inputs are not all the same. For example

$$\text{XOR}(1, 0) = 1, \text{ but } \text{XOR}(1, 1) = 0.$$

   □ None of the above.

3. (2 points) **[SOLO]** Suppose we have a dataset $\left\{\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(N)}, y^{(N)}\right)\right\}$, where $\mathbf{x}^{(i)} \in \mathbb{R}^M$, $y^{(i)} \in \{+1, -1\}$. We would like to apply the perceptron algorithm on this dataset. Assume there is no intercept term. How many parameter values is the perceptron algorithm learning?

   Select one:

   ○ N

   ○ $N \times M$

   [✓ ● M]

4. (2 points) **[SOLO]** Which of the following are true about the perceptron algorithm?

   Select all that apply:

   - ☐ The number of mistakes the perceptron algorithm makes is proportional to the number of points in the dataset.
   - ☐ The perceptron algorithm converges on any dataset.
   - ■ The perceptron algorithm can be used in the context of online learning.
   - ☐ For linearly separable data, the perceptron algorithm always finds the separating hyperplane with the largest margin.
   - ☐ None of the above.

5. (2 points) **[SOLO]** The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm. What is the separating plane w found by the algorithm, i.e. $w = [b, w_1, w_2, w_3]$? Assume that the initial weights are all zero.

x_1	x_2	x_3	y	Times Misclassified
2	1	5	1	10
5	3	3	1	5
4	1	2	1	8
8	4	8	-1	2
3	2	6	-1	3

   Select one:

   - ○ $[3, 22, 11, 24]$
   - ○ $[0, -5, 20, -10]$
   - ○ $[16, 56, 18, 47]$
   - ● $[18, 52, 19, 47]$

6. (2 points) **[SOLO]** Please select the correct statement(s) about the mistake bound of the perceptron algorithm.

   Select all that apply:

   - ☐ If the minimum distance from any data point to the separating hyperplane is increased, without any other change to the data points, the mistake bound will also increase.
   - ■ If the whole dataset is shifted away from origin, then the mistake bound will also increase.
   - ☐ If the size of the data set (i.e., the maximum pair-wise distance between data points) is increased, then the mistake bound will also increase.
   - ☐ The mistake bound is linearly inverse-proportional to the minimum distance of any data point to the separating hyperplane of the data.
   - ☐ None of the above.

7. (2 points) **[SOLO]** Suppose we have data whose elements are of the form $[x_1, x_2]$, where $x_1 - x_2 = 0$. We do not know the label for each element. Suppose the perceptron algorithm starts with $\boldsymbol{\theta} = [3, 5]$, which of the following values will $\boldsymbol{\theta}$ never take on in the process of running the perceptron algorithm on the data?

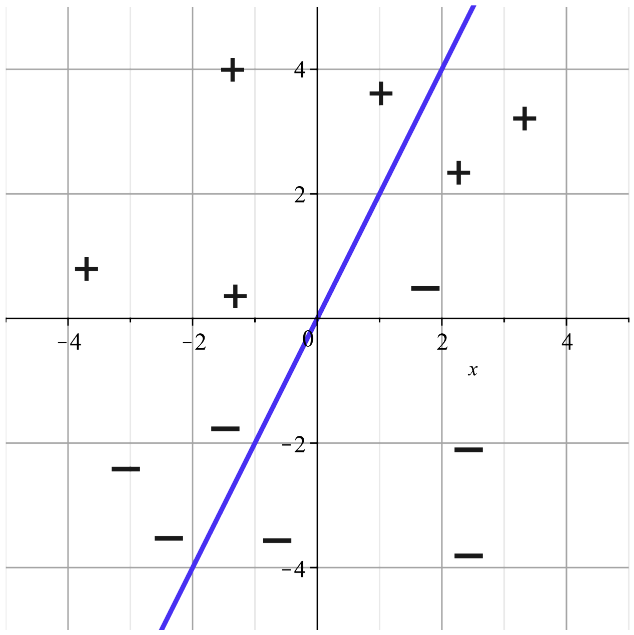
   Select one:

   - ○ $[-1, 1]$
   - ○ $[4, 6]$
   - [✓ $[-3, 0]$]
   - ○ $[-6, -4]$

8. (2 points) **[SOLO]** Consider the linear decision boundary below and the training dataset shown. Which of the following are valid weights $\boldsymbol{\theta}$ and its corresponding training error? (Note: Assume the decision boundary is fixed and does not change while evaluating training error.)

   Select all that apply:

   - □ $\boldsymbol{\theta} = [2, 1]$, error = 5/13
   - [✓ $\boldsymbol{\theta} = [-2, 1]$, error = 5/13]
   - [✓ $\boldsymbol{\theta} = [2, -1]$, error = 8/13]
   - □ $\boldsymbol{\theta} = [2, -1]$, error = 5/13
   - □ $\boldsymbol{\theta} = [-2, 1]$, error = 8/13
   - □ $\boldsymbol{\theta} = [-2, -1]$, error = 8/13
   - □ None of the above.



9. (9 points) The following problem will walk you through an application of the Perceptron Mistake Bound. The following table shows a linearly separable dataset. Your task will be to determine the mistake bound for the dataset below. Then, you will run the training of the perceptron.

x_1	x_2	y
-1.939	2.704	1
-0.928	-3.054	-1
-2.181	-3.353	-1
-0.142	1.440	1
2.605	-0.651	1

(a) (2 points) **[SOLO]** Compute the radius R of the "circle" which bounds the data points. Round your answer to 3 decimal places after the decimal point.

Radius:

4

(b) (2 points) **[SOLO]** Assume that the linear separator with the largest margin is given by

$$\boldsymbol{\theta}^{*T}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0, \text{, where } \boldsymbol{\theta}^{*} = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

Now, compute the margin of the dataset. Round your answer to 3 decimal places after the decimal point.

Margin:

2

(c) (1 point) **[SOLO]** Based on the above values, what is the theoretical perceptron mistake bound for this dataset, given this linear separator? Round your answer to 3 decimal places after the decimal point.

Mistake Bound:

4

(d) (3 points) **[SOLO]** Finally, train the perceptron using the dataset. Train using the datapoints **in the given order, top to bottom** repeatedly until convergence. Give the final weights in a comma-separated list (w_1, w_2, b).

Weights:

$(1.594, 5.107, 1]$

(e) (1 point) **[SOLO]** How many mistakes did the perceptron make?

Mistakes:

3

4 Linear Regression

1. (3 points) We would like to fit a linear regression estimate to the dataset

$$D = \left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \cdots, \left(\mathbf{x}^{(N)}, y^{(N)}\right) \right\}$$

with $\mathbf{x}^{(i)} \in \mathbb{R}^M$ by minimizing the ordinary least square (OLS) objective function:

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N} \left(y^{(i)} - \sum_{j=1}^{M} w_j x_j^{(i)} \right)^2$$

(a) (2 points) **[SOLO]** Specifically, we solve for each coefficient w_k $(1 \le k \le M)$ by deriving an expression of w_k from the critical point $\frac{\partial J(\mathbf{w})}{\partial w_k} = 0$. What is the expression for each w_k in terms of the dataset $(\mathbf{x}^{(1)}, y^{(1)}), \cdots, (\mathbf{x}^{(N)}, y^{(N)})$ and $w_1, \cdots, w_{k-1}, w_{k+1}, \cdots, w_M$?

Select one:

- ● $w_k = \frac{\sum_{i=1}^N x_k^{(i)}(y^{(i)} - \sum_{j=1, j\neq k}^M w_j x_j^{(i)})}{\sum_{i=1}^N (x_k^{(i)})^2}$
- ○ $w_k = \frac{\sum_{i=1}^N x_k^{(i)}(y^{(i)} - \sum_{j=1, j\neq k}^M w_j x_j^{(i)})}{\sum_{i=1}^N (y^{(i)})^2}$
- ○ $w_k = \sum_{i=1}^N x_k^{(i)}(y^{(i)} - \sum_{j=1}^M w_j x_j^{(i)})$
- ○ $w_k = \frac{\sum_{i=1}^N x_k^{(i)}(y^{(i)} - \sum_{j=1, j\neq k}^M w_j x_j^{(i)})}{\sum_{i=1}^N (x_k^{(i)} y^{(i)})^2}$

(b) (1 point) **[SOLO]** How many coefficients do you need to estimate? When solving for these coefficients, how many equations do you have?

Select one:

- ○ N coefficients, M equations
- ○ M coefficients, N equations
- ● M coefficients, M equations
- ○ N coefficients, N equations

2. (3 points) **[SOLO]** Consider the following 3 data points for linear regression: $\mathbf{x}^{(1)} = [0, 1, 2]^T$, $\mathbf{x}^{(2)} = [1, 0, 2]^T$ and $\mathbf{x}^{(3)} = [2, 1, 0]^T$. The corresponding y values are $y^{(1)} = 3$, $y^{(2)} = 6$, $y^{(3)} = 9$.

   Assume the intercept to be 0. Find the weights $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]^T \in \mathbb{R}^3$ such that the mean squared error $J(\boldsymbol{\theta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$ is minimized on this training set. $\mathbf{X}$ is the design matrix where $\mathbf{X}_{ij} = \mathbf{x}_j^{(i)}$.

θ_1:	θ_2:	θ_3:
4	1	1

3. (1 point) **[SOLO]** Assume that a data set has M data points and N variables, where $M > N$. Different loss functions would return the same sets of solutions as long as they are convex.

   Select one:

   ○ True

   [✓ False]

4. (1 point) **[SOLO]** Suppose we are working with datasets where the number of features is 3. The optimal solution for linear regression is always unique regardless of the number of data points that are in this dataset.

   Select one:

   ○ True

   [✓ False]

5. (1 point) **[SOLO]** Consider the following dataset:

x_1	1.5	2.5	3.5	4.5	5.5
x_2	3.0	5.0	7.0	9.0	11.0
y	4.0	7.0	8.0	11.0	17.0

   We want to carry out a multiple-linear regression between y (dependent variable) and x_1 and x_2 (independent variables). The closed-form solution given by $\mathbf{w} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{Y}$ will return the unique solution.

   Note: The i^{th} row of $\mathbf{X}$ contains the i^{th} data point $(x_1^{(i)}, x_2^{(i)})$ while the i^{th} row of $\mathbf{Y}$ contains the i^{th} data point $y^{(i)}$.

   Select one:

   ○ True

   [✓ False]

6. (3 points) **[SOLO]** Identifying whether a function is a convex function is useful because a convex function's local minimum has the nice property that it has to be the global minimum. Please select all functions below that are convex functions. Note $dom(f)$ denotes the domain of the function f.
   Select all that apply:
   - ☐ $f(x) = x^3 + 3x + 7, dom(f) = \mathbb{R}$
   - ■ $f(x) = -\log x, dom(f) = \mathbb{R}_{++}$ (the set of positive real numbers)
   - ☐ $f(x) = -|x|, dom(f) = \mathbb{R}$
   - ☐ $f(x) = -2x^2, dom(f) = \mathbb{R}$
   - ■ $f(x) = ||\mathbf{x}||_2^2,\ dom(f) = \mathbb{R}^n$
   - ☐ None of the above.

7. (2 points) **[GROUP]** Typically we can solve linear regression problems in two ways. One is through direct methods, e.g. solving the closed form solution, and the other is through iterative methods (gradient descent). Consider a linear regression on data $(\mathbf{X}, \mathbf{y})$. We assume each row in $\mathbf{X}$ denotes one input in the dataset. Please select all correct options.

   Select all that apply:
   - ☐ If the matrix $\mathbf{X}^T\mathbf{X}$ is invertible, the exact solution is always preferred for solving the solution to linear regression as computing matrix inversions and multiplications are fast regardless of the size of the dataset.
   - ☐ Assume N is the number of examples and M is the number of features. The computational complexity of N iterations of batch gradient descent is $O(MN)$.
   - ☐ The computational complexity of the closed form solution is linear in number of parameters/features.
   - ■ None of the above.

8. (8 points) Consider the following dataset:

x	9.0	2.0	6.0	1.0	8.0
y	1.0	0.0	3.0	0.0	1.0

Let x be the vector of datapoints and y be the label vector. Here, we are fitting the data using gradient descent. Note that our objective function is $\frac{1}{N}\sum_{i=1}^{N}(wx_i + b - y_i)^2$ where N is the number of data points, w is the weight, and b is the intercept.

(a) (2 points) **[GROUP]** If we initialize the weight as 3.0 and intercept as 0.0, what is the gradient of the loss function with respect to the weight w, calculated over all the data points, in the first step of the gradient descent update? Round to 2 decimal places after the decimal point.

Gradient:

209.2

(b) (4 points) **[GROUP]** Compute the direct solution of the weight and the intercept for the objective function defined in the previous question. Round to 2 decimal places after the decimal point.

Weight:

0.18
0.08

(c) (2 points) **[GROUP]** Let the learning rate be 0.001. Perform five steps of batch gradient descent on the data. Fill in the blank with the value of the weight after five steps of batch gradient descent. Round to 2 decimal places after the decimal point.

Weight:

2.10,

9. (6 points) Consider a dataset $D = \big((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\big)$ and the solution to linear regression on D is $y = w_1 x + b_1$.

   (a) (2 points) **[GROUP]** Now, suppose we have the dataset $(x^{(1)} + \alpha, y^{(1)} + \beta), \ldots, (x^{(n)} + \alpha, y^{(n)} + \beta)$ where $\alpha > 0, \beta > 0$ and $w_1\alpha \neq \beta$. The solution to the linear regression on this dataset is $y = w_2 x + b_2$. Select the correct statement about w_1, w_2, b_1, b_2 below. Note that the statement should hold no matter what values α, β take on within the specified constraints.

   Select one:

   ○ $w_1 = w_2, b_1 = b_2$

   ○ $w_1 \neq w_2, b_1 = b_2$

   [✓ $w_1 = w_2, b_1 \neq b_2$]

   ○ $w_1 \neq w_2, b_1 \neq b_2$

   (b) (2 points) **[GROUP]** Let $\bar{x}$ and $\bar{y}$ be the mean of the x and y coordinates, respectively. After mean centering the dataset to create $D_{new} = \big((x^{(1)} - \bar{x}, y^{(1)} - \bar{y}), \ldots, (x^{(n)} - \bar{x}, y^{(n)} - \bar{y})\big)$, let the solution to linear regression on D_{new} be $y = w_3 x + b_3$. Explain how w_3 compares to w_1 and justify.

   Your answer:

   w_3 is the same as w_1.
   By centering the dataset around the means - $\bar{x}$ and $\bar{y}$, we are shifting the data points to the origin. However, the distribution of the data points relative to the origin now is the same as that relative to $(\bar{x}, \bar{y})$ before centering. So, the line fitting the points now is the same as the line fitting the points before centering, but shifted. Thus, the overall slope of the new line would be the same and hence, w_3 is the same as w_1.

   (c) (2 points) **[GROUP]** We decide to ask a friend to analyze D; however, he makes a mistake by duplicating a subset of the rows in D. Explain why his solution to linear regression on the duplicated data may differ from the solution to linear regression on D.

   Your answer:

   The errors of the duplicated rows/data points would be emphasized more. This would be because the same error values would repeat while calculating the objective function. Due to this, the algorithm would try to minimize the error corresponding to these points more than others and so, the hypothesis line would move closer to these points. Thus, the resultant hypothesis wouldn't have learnt to generalize over the entire data and would have overfitted on these repeated points.

5 Collaboration Questions

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found here.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details.
2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details.
3. Did you find or come across code that implements any part of this assignment ? If so, include full details.

Your Answer

1 - No
2 - No
3 - No