DSC 40A - Homework 1

Due: Tuesday, April 11 at 11:59pm

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Homeworks are due to Gradescope by 11:59pm on the due date. You can use a slip day to extend the deadline by 24 hours.

Homework will be evaluated not only on the correctness of your answers, but on your ability to present your ideas clearly and logically. You should **always explain and justify** your conclusions, using sound reasoning. Your goal should be to convince the reader of your assertions. If a question does not require explanation, it will be explicitly stated.

Homeworks should be written up and turned in by each student individually. You may talk to other students in the class about the problems and discuss solution strategies, but you should not share any written communication and you should not check answers with classmates. You can tell someone how to do a homework problem, but you cannot show them how to do it.

For each problem you submit, you should **cite your sources** by including a list of names of other students with whom you discussed the problem. Instructors do not need to be cited.

This homework will be graded out of 50 points. The point value of each problem or sub-problem is indicated by the number of avocados shown.

Note: For Problem 5, parts (a), (c), (d), and (e), you'll need to code your answers in Python. We've provided some starter code, linked here. You'll need to turn in your completed Python file to Gradescope separately from the rest of this homework, in a file called `hw1code.py` . We'll grade your code using an autograder, so explanations are not necessary for Problem 5, parts (a), (c), (d), and (e),

Problem 0. Welcome Survey

🥑🥑 Make sure to fill out the Welcome Survey, linked here for two points on this homework!

Problem 1. Means

Which of the following statements *must* be true? Remember to justify all answers.

a) 🥑🥑 At least half of the numbers in a data set must be smaller than the mean.

b) 🥑🥑 Some of the numbers in the data set must be smaller than the mean.

c) 🥑🥑 Exactly half of the numbers in a data set must be smaller than the mean.

d) 🥑🥑 Not all of the numbers in the data set can be smaller than the mean.

Problem 2. Linear Functions

Consider the linear function $f(x) = 2x - 5$.

a) 🥑 If $a \leq b$, show that $f(a) \leq f(b)$.

b) 🥑🥑 Both of the statements below are true, but only one is a consequence of the property you proved in part (a). Which is it? Show that this statement is true, using the result of part (a).

1. $\text{Mean}(f(x_1), \dots, f(x_n)) = f(\text{Mean}(x_1, \dots, x_n))$
2. $\text{Median}(f(x_1), \dots, f(x_n)) = f(\text{Median}(x_1, \dots, x_n))$

c) 🥑🥑 Now, prove the other statement.

d) Suppose we consider a different linear function $g(x) = -3x + 2$. Prove or find a counterexample to disprove each of the following:

1. 🥑 If $a \le b$, then $g(a) \le g(b)$.
2. 🥑🥑 $\text{Mean}(g(x_1), \dots, g(x_n)) = g(\text{Mean}(x_1, \dots, x_n))$
3. 🥑🥑 $\text{Median}(g(x_1), \dots, g(x_n)) = g(\text{Median}(x_1, \dots, x_n))$

Problem 3. Max's Idea

In the first lecture, we argued that one way to make a good prediction h was to minimize the mean absolute error:

$$R(h) = \frac{1}{n}\sum_{i=1}^{n} |h - y_i|.$$

We saw that the median of $y_1, \dots, y_n$ is the prediction with the smallest mean error. Your friend Max has many ideas for other ways to make predictions.

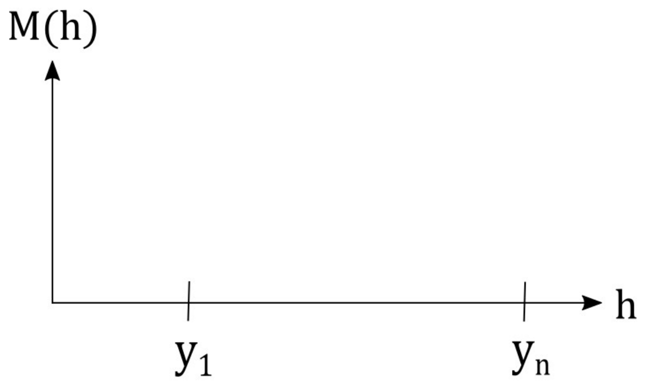
Max suggests that instead of minimizing the mean error, we could minimize the *maximum error*:

$$M(h) = \max_{i=1,\dots,n} |y_i - h|$$

In this problem, we'll see if Max has a good idea.

a) 🥑🥑🥑🥑 Suppose that the data set is arranged in increasing order, so $y_1 \le y_2 \le \dots \le y_n$. Argue that $M(h) = \max(|y_1 - h|, |y_n - h|)$.

b) 🥑🥑 On the axes below, draw the graph of $M(h) = \max(|y_1 - h|, |y_n - h|)$. Label key points with their coordinates.



c) 🥑🥑🥑🥑 Show that $M(h)$ is minimized at $h^* = \frac{y_1 + y_n}{2}$, which is sometimes called the *midrange* of the data. Then discuss whether Max had a reasonable idea.

Problem 4. Max's Other Idea

You friend Max from problem 2 has another idea. He suggests that instead of minimizing mean absolute error, we try maximizing the following quantity:

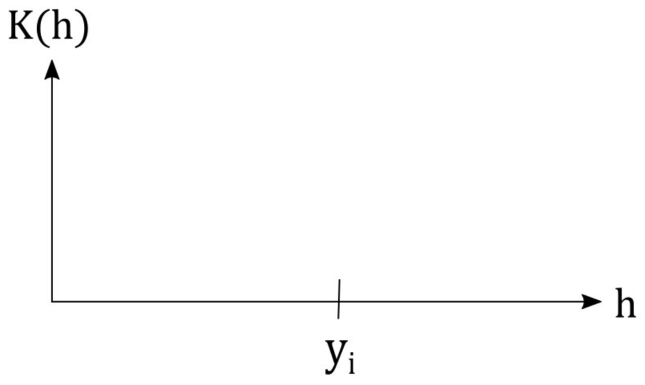
$$P(h) = \prod_{i=1}^{n} e^{-|h-y_i|}.$$

The above formula is written using product notation, which is similar to summation notation, except terms are multiplied and not added. For example,

$$\prod_{i=1}^{n} a_i = a_1 \cdot a_2 \cdot \ldots \cdot a_n.$$

Max's reasoning is that for some models, $K(h) = e^{-|h-y_i|}$ is used to compute how likely prediction h will appear given the observation y_i – hence it is called "likelihood." Then, we should attempt to maximize the chance of getting the prediction h, given the set of observations. In this problem, we'll see if Max has a good idea.

a) 🥑🥑 On the axes below, sketch a graph of the basic shape of the likelihood function $K(h) = e^{-|h-y_i|}$. Label key points with their coordinates. Explain, based on the graph, why larger values of $K(h)$ correspond to better predictions h.



b) 🥑🥑🥑🥑 Recall from Groupwork 1 that for a function of one variable $f(x)$, a value x^* is said to be a **minimizer** of $f(x)$ if

$$f(x^*) \leq f(x) \quad \text{for all } x.$$

Similarly, x^* is said to be a **maximizer** of $f(x)$ if

$$f(x^*) \geq f(x) \quad \text{for all } x.$$

Suppose that $f(x)$ is a function that is minimized at x^* and c is a positive constant. Show that the function $g(x) = e^{-c \cdot f(x)}$ is maximized at x^*.

c) 🥑🥑 At what value h^* is $P(h)$ maximized? Did Max have a reasonable idea?

Hint: Try writing out $P(h)$ by expanding the product notation, similarly to how we expanded summation notation in Groupwork 1.

Problem 5. Side Hustle

Note: For parts (a), (c), (d), and (e) of this problem, you'll need to code your answers in Python. We've provided some starter code, linked here. You'll need to turn in your completed Python file to Gradescope separately from the rest of this homework, in a file called `hw1code.py` . We'll grade your code using an autograder, so explanations are not necessary for parts (a), (c), (d), and (e) of this question.

Suppose your backyard avocado tree produces more avocados than you can eat, so you decide to sell some to help offset the rising costs of UCSD tuition. Instead of selling each avocado for a fixed price, like most grocery stores do, you prefer to sell them per pound.

a) 🥑🥑🥑 Suppose you want to keep track of the mean avocado weight of all your sales, but your scale does not record or remember the weights of previous measurements. Instead of writing down the weight of each sale and re-calculating the mean with each transaction, you want to come up with a way to only keep track of the mean weight.

Complete the function `avo_mean` which should calculate the mean weight of all avocado sales based on three parameters:

- `prev_mean`, the mean weight of all avocado sales so far,
- `n`, the number of avocado sales so far, and
- `weight`, the weight of the very next avocado sale.

The function should return the mean weight of all `n+1` avocado sales.

b) 🥑 Suppose that so far, you've sold avocados to 12 customers, and the mean avocado weight for these 12 sales is 2.37 pounds. How much would your next sale have to weigh in order to bring the mean weight up to 2.5 pounds?

c) 🥑🥑 We've shown that it's possible to update the mean after each sale without keeping track of all the individual weights of each sale. Show that we cannot do the same for the median, by giving an example of two different lists of avocado weights such that:

- both lists are of the same length,
- both lists have the same last number,
- both lists have the same median value when we exclude the last number, and
- both lists have different median values when we include the last number.

This shows that the new median weight cannot be determined by the value of the previous median weight, the number of transactions, and the new weight alone. Save your lists as `list1c` and `list2c`.

d) 🥑🥑 Suppose now that instead of determining the mean weight of all purchases, you want to find the **midrange** weight of all sales, where the midrange is how we defined it in Problem 2(a). Again, we don't want to have to keep track of every sale's weight in order to do this.

Unfortunately, knowledge of the previous midrange, number of transactions, and the weight of the next sale are insufficient to determine the midrange. Show that this is the case by providing an example of two lists of avocado weights such that:

- both lists are of the same length,
- both lists have the same last number,
- both lists have the same midrange when we exclude the last number, and
- both lists have different midranges when we include the last number.

Save your lists as `list1d` and `list2d`.

e) 🥑🥑🥑🥑 It turns out that if we keep track of just one additional piece of information, the maximum weight of all sales, then we can calculate the midrange without recording all individual sale weights!

Complete the function `avo_midrange` which should calculate the midrange of all avocado sale weights based on four parameters:

- `prev_midrange`, the midrange of all avocado sales weights so far,
- `prev_max`, the largest weight of all avocado sales so far,
- `n`, the number of avocado sales so far, and
- `weight`, the weight of the very next avocado sale.

The function should return a **list** of two elements, the first being the midrange of all `n+1` sale weights, the second being the maximum of all `n+1` sale weights.