

---

## DSC 40A - Homework 4

Due: Tuesday, May 2 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.

### Notes:

- Make sure to **assign pages to questions** when you upload your submission to Gradescope. This really helps our graders. Starting with this homework, assigning pages will be required and we'll deduct points if you don't.
- For Problem 4, parts (b), (c), and Problem 5, you'll need to code your answers in Python. We've provided a [supplementary Jupyter notebook \(linked\)](#). You'll need to turn in your completed Python file to Gradescope separately from the rest of this homework, in a file called `hw4code.py`. We'll grade these problems using an autograder, so explanations are not necessary.

### Problem 0. Reflection and Feedback Form

 Make sure to fill out this [Reflection and Feedback Form, linked here](#) for three points on this homework! This form is primarily for your benefit; research shows that reflecting and summarizing knowledge helps you understand and remember it.

### Problem 1. Vector Calculus Involving Matrices

Let  $X$  be a fixed matrix of dimension  $m \times n$ , and let  $\vec{w} \in \mathbb{R}^n$ . In this problem, you will show that the gradient of  $\vec{w}^T X^T X \vec{w}$  with respect to  $\vec{w}$  is given by

$$\frac{d}{d\vec{w}}(\vec{w}^T X^T X \vec{w}) = 2X^T X \vec{w}.$$

Let  $\vec{r}_1, \vec{r}_2, \dots, \vec{r}_m$  be the column vectors in  $\mathbb{R}^n$  that come from transposing the rows of  $X$ . For example, if

$$X = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 3 & 1 \end{bmatrix}, \text{ then } \vec{r}_1 = \begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix} \text{ and } \vec{r}_2 = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}.$$

a) 🥑🥑🥑🥑 Show that, for arbitrary  $X$  and  $\vec{w}$ , we can write

$$\vec{w}^T X^T X \vec{w} = \sum_{i=1}^m (\vec{r}_i^T \vec{w})^2.$$

*Hint:* First, show that we can write  $\vec{w}^T X^T X \vec{w}$  as a dot product of two vectors. Then, try and re-write those vectors in terms of  $\vec{r}_1, \vec{r}_2, \dots, \vec{r}_m$  and  $\vec{w}$ .

Now that we have written

$$\vec{w}^T X^T X \vec{w} = \sum_{i=1}^m (\vec{r}_i^T \vec{w})^2$$

we can apply the chain rule, along with the result of part (a) above, to conclude that

$$\begin{aligned} \frac{d}{d\vec{w}} (\vec{w}^T X^T X \vec{w}) &= \sum_{i=1}^m 2(\vec{r}_i^T \vec{w}) \frac{d}{d\vec{w}} (\vec{r}_i^T \vec{w}) \\ &= \sum_{i=1}^m 2(\vec{r}_i^T \vec{w}) \vec{r}_i \end{aligned}$$

b) 🥑🥑🥑🥑 Next, show that, for arbitrary  $X$  and  $\vec{w}$ , we can write

$$2X^T X \vec{w} = \sum_{i=1}^m 2(\vec{r}_i^T \vec{w}) \vec{r}_i$$

*Hint 1:* Use the column-mixing interpretation of matrix-vector multiplication from Lecture 10.

*Hint 2:* It is likely that you'll need to use one of your intermediate results from part (a).

Since you've shown that  $\frac{d}{d\vec{w}} (\vec{w}^T X^T X \vec{w})$  and  $2X^T X \vec{w}$  are both equal to the same expression,  $\sum_{i=1}^m 2(\vec{r}_i^T \vec{w}) \vec{r}_i$ , you have proven that they are equal to one another, i.e. that

$$\frac{d}{d\vec{w}} (\vec{w}^T X^T X \vec{w}) = 2X^T X \vec{w}$$

as desired.

## Problem 2. Sums of Residuals

Let's start by recalling the idea of orthogonality from linear algebra. This will allow us to prove a powerful result regarding linear regression.

Two vectors are **orthogonal** if their dot product is 0, i.e. for  $\vec{a}, \vec{b} \in \mathbb{R}^n$ :

$$\vec{a}^T \vec{b} = 0 \implies \vec{a}, \vec{b} \text{ are orthogonal}$$

Orthogonality is a generalization of perpendicularity to multiple dimensions. (Two orthogonal vectors in 2D meet at a right angle.)

Suppose we want to represent the fact that some vector  $\vec{b}$  is orthogonal to many vectors  $\vec{a}_1, \vec{a}_2, \dots, \vec{a}_d$  all at once. It turns out that we can do this by creating a new  $n \times d$  matrix  $A$  whose columns are the vectors  $\vec{a}_1, \vec{a}_2, \dots, \vec{a}_d$ , and writing  $A^T \vec{b} = 0$ .

For instance, suppose  $\vec{a}_1 = \begin{bmatrix} 8 \\ 4 \\ -2 \end{bmatrix}$ ,  $\vec{a}_2 = \begin{bmatrix} 3 \\ 5 \\ 1 \end{bmatrix}$ , and  $\vec{b} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}$ . Then,

$$A = \begin{bmatrix} 8 & 3 \\ 4 & 5 \\ -2 & 1 \end{bmatrix} \implies A^T = \begin{bmatrix} 8 & 4 & -2 \\ 3 & 5 & 1 \end{bmatrix}$$

Note that the product  $A^T \vec{b}$  involves taking the dot product of each row in  $A^T$  with  $\vec{b}$ .

$$A^T \vec{b} = \begin{bmatrix} 8 & 4 & -2 \\ 3 & 5 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 8(1) + 4(-1) + (-2)(2) \\ 3(1) + 5(-1) + 2(1) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Since  $A^T \vec{b} = \vec{0}$ , then it is the case that  $\vec{b}$  is orthogonal to each row of  $A^T$ , and hence orthogonal to each column of  $A$ .

(We will not use this fact in this class, but if  $A^T \vec{b} = 0$ , it also means that  $\vec{b}$  is orthogonal to the **column space** of  $A$ , which is the space of all linear combinations of the columns of  $A$ . As a good exercise in linear algebra, see if you can prove this result!)

- a) 🥑🥑 Suppose  $\vec{1}$  is a vector in  $\mathbb{R}^n$  containing the value 1 for each element, i.e.  $\vec{1} = \begin{bmatrix} 1 \\ 1 \\ \dots \\ 1 \end{bmatrix}$ .

For any other vector  $\vec{b} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$ , what is the value of  $\vec{1}^T \vec{b}$ , i.e. what is the dot product of  $\vec{1}$  and  $\vec{b}$ ?

- b) 🥑🥑 Now, consider the typical multiple regression scenario where our prediction rule has an intercept term ( $w_0$ ):

$$H(\vec{x}) = w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \dots + w_d x^{(d)}.$$

For this scenario,  $X$  is a  $n \times (d+1)$  design matrix,  $\vec{y} \in \mathbb{R}^n$  is an observation vector, and  $\vec{w} \in \mathbb{R}^{(d+1)}$  is the parameter vector. We'll use  $\vec{w}^*$  to denote the optimal parameter vector, or the one that satisfies the normal equations.

Show that the error vector,  $\vec{e} = \vec{y} - X\vec{w}^*$ , is orthogonal to the columns of  $X$ .

*Hint:* Use the normal equations and the definition of orthogonality to the columns of a matrix given in the problem description.

- c) 🥑🥑🥑🥑 We define the  $i$ th **residual** to be the difference between the actual and predicted values for individual  $i$  in our data set. In other words, the  $i$ th residual  $e_i$  is

$$e_i = (\vec{y} - X\vec{w}^*)_i$$

Here,  $(\vec{y} - X\vec{w}^*)_i$  is referring to element  $i$  of the vector  $\vec{y} - X\vec{w}^*$ . We use the letter  $e$  for residuals because residuals are also known as errors.

Using what you learned in parts (a) and (b), show that for multiple linear regression with an intercept term, the sum of the residuals is zero, that is

$$\sum_{i=1}^n e_i = 0.$$

- d) 🥑🥑🥑🥑 Now suppose our multiple linear regression prediction rule does not have an intercept term:

$$H(\vec{x}) = w_1 x^{(1)} + w_2 x^{(2)} + \dots + w_d x^{(d)}.$$

1. Is it still guaranteed that  $\sum_{i=1}^n e_i = 0$ ? Why or why not?
2. Is it still possible that  $\sum_{i=1}^n e_i = 0$ ? If you believe the answer is yes, come up with a simple example where a prediction rule without an intercept has residuals that sum to 0. If you believe the answer is no, state why not.




### Problem 3. Real Estate

You are given a data set containing information on recently sold houses in San Diego, including

- square footage
- number of bedrooms
- number of bathrooms
- year the house was built
- asking price, or how much the house was originally listed for, before negotiations
- sale price, or how much the house actually sold for, after negotiations

The table below shows the first few rows of the data set. Note that since you don't have the full data set, you cannot answer the questions that follow based on calculations; you must answer conceptually.

House	Square Feet	Bedrooms	Bathrooms	Year	Asking Price	Sale Price
1	1247	3	3	2005	500,000	494,000
2	1670	3	2	1927	1,000,000	985,000
3	716	1	1	1993	335,000	333,850
4	1600	4	2	1962	830,000	815,000
5	2635	4	3	1993	1,250,000	1,250,000
⋮	⋮	⋮	⋮	⋮	⋮	⋮

-  Suppose you standardize all six variables and fit a linear prediction rule to predict the sale price of the house based on all five of the other variables. Which feature would you expect to have the largest magnitude weight? Without standardizing, which feature would you expect to have the largest magnitude weight? Explain why.
-  Suppose you use multiple linear regression on the original (unstandardized) data and the weight associated with Year is  $\alpha$ . Suppose you replace Year with a new predictor variable, Age, which is 0 if the house was built in 2023, 1 if the house was built in 2022, 2 if the house was built in 2021, etc. If we do multiple linear regression again using Age instead of Year, what will be the weight associated with Age in terms of  $\alpha$ ?
-  Suppose you add a new feature called Rooms, which is the total number of bedrooms and bathrooms in the house. Would multiple linear regression with this extra feature enable you to make better predictions?

#### Problem 4. Least Absolute Deviation Regression for Multiple Variables

In this week's lectures, we explored least squares regression and derived the general solution for least squares regression with multiple parameters, also known as the normal equations:

$$X^T X \vec{w} = X^T \vec{y},$$

where  $X$  is the design matrix,  $\vec{y}$  is the observation vector, and  $\vec{w}$  is the parameter vector.

In this problem, we are going to try and expand our concept of least absolute deviation (LAD) regression from Homework 3 to accommodate linear prediction rules with two features.

This time, instead of data in  $\mathbb{R}^2$ , you will be given data in  $\mathbb{R}^3$ . Now that we have added an extra dimension to our data, we are no longer solving for a regression line, but rather a regression plane of the form  $H(\vec{x}) = w_0 + w_1 x^{(1)} + w_2 x^{(2)}$ . In order to use notation that is more convenient and more similar to what you've seen in calculus, let's suppose we're trying to find a regression plane of the form

$$z = H(x, y) = ax + by + c$$

When performing LAD regression, our loss function is absolute loss. That means that we're trying to find the  $a$ ,  $b$ , and  $c$  that together minimize mean absolute error,

$$R_{abs}(a, b, c) = \frac{1}{n} \sum_{i=1}^n |z_i - (ax_i + by_i + c)|$$

We will adopt a strategy similar to Homework 3 to solve for the LAD regression plane. Recall the theorem from last week: If you have a data set with  $n$  data points in  $\mathbb{R}^k$ , where  $k \leq n$ , then one of the optimal LAD regression prediction rules must pass through  $k$  data points.

Since our data will be in  $\mathbb{R}^3$ , we will generate all possible unique **triplets** of points and calculate the coefficients  $a$ ,  $b$ , and  $c$  that define a plane of the form  $z = ax + by + c$  that goes through the three points in our triplet. Then we'll just select which  $a, b, c$  triplet among these finite options has the smallest value of  $R_{abs}(a, b, c)$ . This is guaranteed by the theorem to be an optimal LAD regression plane.

a) 🥑🥑🥑 Before implementing the above strategy in code, let's get familiar with the process by doing

an example by hand. Let's say we have three points,  $A = \begin{bmatrix} 1 \\ 4 \\ 5 \end{bmatrix}$ ,  $B = \begin{bmatrix} 2 \\ 3 \\ -1 \end{bmatrix}$ , and  $C = \begin{bmatrix} 4 \\ -2 \\ 0 \end{bmatrix}$ .

Given these three points in  $\mathbb{R}^3$ , compute the equation of the plane going through them, giving your answer in the form  $z = ax + by + c$ .

*Note:* You may need to review some concepts from multivariable calculus (i.e. Math 20C) to do this problem. [Here is a nice summary and some examples](#), obtained from MIT OpenCourseWare under a Creative Commons License.

b) 🥑🥑 Just like last time, we'll start by finding the regular least squares regression plane, which is the plane  $z = ax + by + c$  that minimizes the mean squared error

$$R_{sq}(a, b, c) = \frac{1}{n} \sum_{i=1}^n (z_i - (ax_i + by_i + c))^2.$$

We already know from class that we can minimize  $R_{sq}$  by solving the normal equations.

In the [supplementary Jupyter notebook \(linked\)](#), fill in the body of the `least_squares_regression` function. This function should take in the design matrix  $X$  and observation vector  $\vec{y}$ , solve the normal equations, and return a tuple  $(a, b, c)$  of parameters that define the least squares regression plane  $z = ax + by + c$ .

c) 🥝🥝🥝🥝🥝 Now, we'll find the LAD regression plane.

Recall from the problem description the procedure outlined to generate an optimal LAD regression plane. In the supplemental notebook, we've already defined several functions for you:

- `generate_all_combinations`, which takes in a dataset and a combination size (in our case, 3) and returns all subsets of the dataset of that size.
- `plane_mae`, which takes in values of  $a$ ,  $b$ ,  $c$  and a dataset and returns the mean absolute error of the plane  $z = ax + by + c$  on that dataset.
- `find_best_plane`, which takes in a list of planes (i.e. a list of  $(a, b, c)$  triplets) and a dataset and finds the plane with the lowest mean absolute error.

Your job is to complete the implementation of a function, `generate_all_planes`, that generates all planes given a list of the triplets of points and return a list of these planes; the list of planes should be a list of  $(a, b, c)$  triplets.

### Problem 5. Prediction Competition

🥝🥝🥝🥝🥝 We are hosting a class-wide competition to see who can make the best predictions! Top predictions can earn extra credit on Midterm 1!

In the [supplementary Jupyter notebook \(linked\)](#), you are given access to a CSV containing training data with information about some of the top Android games in different game categories. We read this in as a DataFrame whose first five rows look like this:

rank	title	total ratings	installs	average rating	growth (30 days)	growth (60 days)	price	category	5 star ratings	4 star ratings	3 star ratings	2 star ratings	1 star ratings	paid
72	Lost in Harmony	55936	1.0 M	4.61	0.2	0.4	0.0	GAME MUSIC	45276	5129	2099	1069	2359	False
9	Farm Heroes Saga	9014495	100.0 M	4.61	0.5	1.1	0.0	GAME CASUAL	7016344	1171316	426099	123431	277303	False
63	Эврика! Логические Задачи, Игры и Головоломки	76755	1.0 M	4.48	0.1	0.2	0.0	GAME TRIVIA	55311	12973	2806	1018	4644	False
69	BLOCK STORY	392609	10.0 M	4.39	0.1	0.2	0.0	GAME ROLE PLAYING	284184	46862	22881	9182	29496	False
70	Immortal Taoists-Idle Game of Immortal Cultiva...	176947	1.0 M	4.45	1.4	3.4	0.0	GAME WORD	125043	28786	9948	3559	9608	False

The columns are:

- **rank:** Rank of game in its category.
- **title:** Title of game.
- **total ratings:** Total number of consumer ratings.
- **installs:** Approximate number of installs.
- **average rating:** Average rating out of 5.
- **growth (30 days):** Percent growth in the past 30 days.
- **growth (60 days):** Percent growth in the past 60 days.
- **price:** Price in dollars.
- **category:** Name of game category.
- **5 star ratings:** Number of 5 star ratings.
- **4 star ratings:** Number of 4 star ratings.

- **3 star ratings:** Number of 3 star ratings.
- **2 star ratings:** Number of 2 star ratings.
- **1 star ratings:** Number of 1 star ratings.
- **paid:** Whether the game is paid (T/F values).

Your task for this problem is to find the best prediction rule using regression to estimate the rank of a game within its category. The requirements are as follows:

1. You must use regression.
2. The function used for regression is your choice (linear, polynomial, exponential, ...)
3. You may use **up to three variables**. You decide which ones.
4. Your design matrix may have **up to five columns**. You decide what the design matrix looks like.
5. While the rank of a game within its category will always be an integer, your prediction function does not need to always make an integer prediction.

We've provided you with a function `calculate_MSE` to calculate the mean squared error of your predictions on each game in the training data. Your job is to fill in the body of the `predict` function. This function should take as input one row of the DataFrame (corresponding to one particular game) and return the predicted rank corresponding to this game. How you make this prediction is up to you, subject to the rules above. Feel free to add more cells and functions, and to change the provided `predict` function, but do not change the provided `calculate_MSE` function.

When we grade this question, we will run your prediction function on a hidden test dataset as well, so be mindful of *overfitting* the training data. You'll earn full credit on this homework problem by finding a prediction function whose MSE on the hidden test data is below a certain threshold, which we hope most students will achieve. Additionally, the ten best prediction functions (as determined by the MSE on the hidden test data) will earn some extra credit on the upcoming midterm exam according to the following scheme: for  $n \leq 10$ , the  $n^{\text{th}}$  ranked prediction function in the class earns  $11 - n$  percentage points as extra credit on Midterm 1.