

Foundations

stanford CS221 Spring 2021-2022

Owner CA: Ishaan Gulrajani



Version: 03/24/2022

Ed Release Post

General Instructions

This (and every) assignment has a written part and a programming part.

The full assignment with our supporting code and scripts can be downloaded as [foundations.zip](#).

-  This icon means you should write responses in [foundations.pdf](#).
-  This icon means you should write code in [submission.py](#).

All written answers must be **typeset (preferably in LaTeX)**. We strongly recommend using Overleaf. A link to a LaTeX template with prompts can be found on Ed, or further down this page.

Also note that your answers should be **in order** and **clearly and correctly labeled** to receive credit. Be sure to submit your final answers as a PDF and tag all pages correctly when submitting to Gradescope.

You should modify the code in [submission.py](#) between

```
# BEGIN_YOUR_CODE
```

and

```
# END_YOUR_CODE
```

but you can add other helper functions outside this block if you want. Do not make changes to files other than [submission.py](#).

Your code will be evaluated on two types of test cases, **basic** and **hidden**, which you can see in [grader.py](#). Basic tests, which are fully provided to you, do not stress your code with large inputs or tricky corner cases. Hidden tests are more complex and do stress your code. The inputs of hidden tests are provided in [grader.py](#), but the correct outputs are not. To run the tests, you will need to have [graderUtil.py](#) in the same directory as your code and [grader.py](#). Then, you can run all the tests by typing

```
python grader.py
```

This will tell you only whether you passed the basic tests. On the hidden tests, the script will alert you if your code takes too long or crashes, but does not say whether you got the correct output. You can also run a single test (e.g., [3a-0-basic](#)) by typing

```
python grader.py 3a-0-basic
```

We strongly encourage you to read and understand the test cases, create your own test cases, and not just blindly run [grader.py](#).


Welcome to your first CS221 assignment! The goal of this assignment is to sharpen your math, programming, and ethical analysis skills needed for this class. If you meet the prerequisites, you should find these problems relatively innocuous. Some of these problems will occur again as subproblems of later homeworks, so make sure you know how to do them. If you're unsure about them or need a refresher, we recommend going through our prerequisites module or other resources on the Internet, or coming to office hours.

Before you get started, please read the Homeworks section on the course website thoroughly.

We've created a LaTeX template [here](#) for you to use that contains the prompts for each question.

Problem 1: Optimization and probability

In this class, we will cast a lot of AI problems as optimization problems, that is, finding the best solution in a rigorous mathematical sense. At the same time, we must be adroit at coping with uncertainty in the world, and for that, we appeal to tools from probability.

- a.  [2 points] Let x_1, \dots, x_n be real numbers representing positions on a number line. Let w_1, \dots, w_n be positive real numbers representing the importance of each of these positions. Consider the quadratic function:


$f(\theta) = \sum_{i=1}^n w_i(\theta - x_i)^2$. Note that θ here is a scalar. What value of θ minimizes $f(\theta)$? Show that the optimum you find is indeed a minimum. What problematic issues could arise if some of the w_i 's are negative?

Note: You can think about this problem as trying to find the point θ that's not too far away from the x_i 's. Over time, hopefully you'll appreciate how nice quadratic functions are to minimize.

What we expect: An expression for the value of θ that minimizes $f(\theta)$ and how you got it. A short calculation/argument to show that it is a minimum. 1-2 sentences describing a problem that could arise if some of the w_i 's are negative.

A necessary condition for θ to be a minimizer of a differentiable function $f(\theta)$ is that its derivative $\frac{d}{d\theta}f(\theta) = 0$. Thus, let's take the derivative and set it to zero: $\frac{d}{d\theta}f(\theta) = 2(\sum_{i=1}^n w_i)\theta - 2\sum_{i=1}^n w_i x_i = 0$. We can now solve for θ . Doing some basic algebra yields $\theta = \frac{\sum_{i=1}^n w_i x_i}{\sum_{i=1}^n w_i}$. To double check that θ is a true minimizer, we check that

the second derivative is positive. The second derivative is $\frac{d^2}{d\theta^2}f(\theta) = 2\sum_{i=1}^n w_i$, which is positive because of the assumption that the w_i 's are positive real numbers. Note that θ is just the weighted average of the x_i 's, which is nice and interpretable. If some w_i 's are negative, then $f(\theta)$ may not be a convex function (based on the second derivative), so a minimizer for this function may not exist.

- b.  [3 points] In this class, there will be a lot of sums and maxes. Let's see what happens if we switch the order. Let

$f(\mathbf{x}) = \max_{s \in [-1, 1]} \sum_{i=1}^d s x_i$ and $g(\mathbf{x}) = \sum_{i=1}^d \max_{s_i \in [-1, 1]} s_i x_i$, where $\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d$ is a real vector and $[-1, 1]$ means the closed interval from -1 to 1 . Which of $f(\mathbf{x}) \leq g(\mathbf{x})$, $f(\mathbf{x}) = g(\mathbf{x})$, or $f(\mathbf{x}) \geq g(\mathbf{x})$ is true for all \mathbf{x} ? Prove it.


Hint: You may find it helpful to refactor the expressions so that they are maximizing the same quantity over different sized sets.

What we expect: A short (3-5) line/sentence proof. You should use mathematical notation in your proof, but can also make your argument in words.

Intuitively, for $g(\mathbf{x})$, we can select a different s for each i , whereas for $f(\mathbf{x})$, we have to select a single s for all i . More formally, to make $f(\mathbf{x})$ and $g(\mathbf{x})$ look more similar for comparison, let's rewrite them:

$f(\mathbf{x}) = \max_{s_1, \dots, s_d} \sum_{i=1}^d s_i x_i$ and $g(\mathbf{x}) = \max_{s_1, \dots, s_d} \sum_{i=1}^d s_i x_i$. Now it's obvious that both functions are maximizing over the same quantity, except that $g(\mathbf{x})$ is maximizing over a larger set. Therefore, we have that

$f(\mathbf{x}) \leq g(\mathbf{x})$ for all \mathbf{x} . Alternatively, we can reach the same answer by realizing that in $g(\mathbf{x})$, we have $\sum_{i=1}^d |x_i|$, while in $f(\mathbf{x})$, we are taking the max of $\sum_{i=1}^d -x_i$ and $\sum_{i=1}^d x_i$.

- c.  [3 points] Suppose you repeatedly roll a fair six-sided die until you roll a 1 or a 2 (and then you stop). Every time you roll a 3, you lose a points, and every time you roll a 6, you win b points. You do not win or lose any points if you roll a 4 or a 5. What is the expected number of points (as a function of a and b) you will have when you stop?


Hint: You will find it helpful to define a recurrence. If you define V as the expected number of points you get from playing the game, what happens if you roll a 3? You lose a points and then get to play again. What about the other cases? Can you write this as a recurrence?

What we expect: A recurrence to represent the problem and the resulting expression from solving the recurrence (no more than 1-2 lines).

Let V be the expected number of points you will earn (the expected value of the game). Consider the different possible cases. If you roll a 3, you lose a points, and then get to re-roll/play the same game again, with an expected value of V . If you roll a 6, the same happens but you gain b points instead. If you roll a 4 or 5, you immediately play the same game again with expected value V . If you roll a 1 or 2, the game ends immediately and you gain no points and get no chance to play again.

We can define the recurrence by considering the probability of each outcome and the value of each outcome:

$V = \frac{1}{6}(V - a) + \frac{1}{6}(V + b) + \frac{2}{6}V + \frac{2}{6}(0) = \frac{1}{6}(-a) + \frac{1}{6}(b) + \frac{4}{6}V$. Solving for V yields $\frac{1}{2}b - \frac{1}{2}a$. These types of calculations will show up in Markov decision processes, where we consider iterated games of chance.

- d.  [3 points] Suppose the probability of a coin turning up heads is p (where $0 < p < 1$), and we flip it 6 times and get {T, H, H, H, T, H}. We know the probability (likelihood) of obtaining this sequence is


$L(p) = (1 - p)pppp(1 - p)p = p^4(1 - p)^2$. What value of p maximizes $L(p)$? Prove/Show that this value of p maximizes $L(p)$. What is an intuitive interpretation of this value of p ?

Hint: Consider taking the derivative of $\log L(p)$. You can also directly take the derivative of $L(p)$, but it is cleaner and more natural to differentiate $\log L(p)$. You can verify for yourself that the value of p which maximizes $\log L(p)$ must also maximize $L(p)$ (you are not required to prove this in your solution).

What we expect: The value of p that maximizes $L(p)$ and the work/calculation used to solve for it. Note that you must prove/show that it is a maximum. A 1-sentence intuitive interpretation of the value of p .

We have $\log L(p) = 4 \log p + 2 \log(1 - p)$. Taking the derivative yields $\nabla \log L(p) = \frac{4}{p} - \frac{2}{1-p}$. Setting this to zero and solving yields $p = \frac{2}{3}$. Checking that the second derivative is negative for $0 < p < 1$:

$\nabla \log L(p) = -\frac{4}{p^2} - \frac{2}{(1-p)^2} < 0$. The optimal p has a very intuitive interpretation: it's just the fraction of heads. This is a classic derivation of the maximum likelihood estimator in statistics.

- e.  [3 points] Now for a little bit of practice manipulating conditional probabilities. Suppose that A and B are two events such that $P(A|B) = P(B|A)$. We also know that $P(A \cup B) = \frac{1}{2}$ and $P(A \cap B) > 0$. Prove that $P(A) > \frac{1}{4}$.


Hint: Note that A and B are not necessarily mutually exclusive. Consider how we can relate $P(A \cup B)$ and $P(A \cap B)$.

What we expect: A short (~5 line) proof/derivation.

We start with Bayes' rule, which tells us that $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. We can divide both sides by $P(A|B)$ to yield $1 = \frac{P(B|A)P(A)}{P(A)P(B)}$. Using the fact that $P(A|B) = P(B|A)$, we cancel and rearrange to find $P(A) = P(B)$.

Next, we consider that $P(A \cup B) = P(A) + P(B) - P(A \cap B)$. Given that $P(A \cup B) = \frac{1}{2}$, we can rearrange this as $P(A) + P(B) = \frac{1}{2} + P(A \cap B)$.

Now, using the fact that $P(A \cap B) > 0$, it holds that $P(A) + P(B) > \frac{1}{2}$. Using our earlier finding that $P(A) = P(B)$ allows us to write $P(A) + P(A) > \frac{1}{2}$, and therefore $P(A) > \frac{1}{4}$, as desired.

- f.  [4 points] Let's practice taking gradients, which is a key operation for being able to optimize continuous functions. For $\mathbf{w} \in \mathbb{R}^d$ (represented as a column vector), and constants $\mathbf{a}_i, \mathbf{b}_j \in \mathbb{R}^d$ (also represented as column vectors), $\lambda \in \mathbb{R}$, and a positive integer n , define the scalar-valued function

$$f(\mathbf{w}) = \left(\sum_{i=1}^n \sum_{j=1}^n (\mathbf{a}_i^\top \mathbf{w} - \mathbf{b}_j^\top \mathbf{w})^2 \right) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2,$$

where the vector is $\mathbf{w} = (w_1, \dots, w_d)^\top$ and $\|\mathbf{w}\|_2 = \sqrt{\sum_{k=1}^d w_k^2} = \sqrt{\mathbf{w}^\top \mathbf{w}}$ is known as the L_2 norm. Compute the gradient $\nabla f(\mathbf{w})$.

Recall: the gradient is a d -dimensional vector of the partial derivatives with respect to each w_i :

$$\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w_1}, \dots, \frac{\partial f(\mathbf{w})}{\partial w_d} \right)^\top.$$

If you're not comfortable with vector calculus, first warm up by working out this problem using scalars in place of vectors and derivatives in place of gradients. Not everything for scalars goes through for vectors, but the two should at least be consistent with each other (when $d = 1$). Do not write out summations over dimensions, because that gets tedious.

What we expect: An expression for the gradient and the work used to derive it. (~5 lines). No need to expand out terms unnecessarily; try to write the final answer compactly.

$\nabla f(\mathbf{w}) = \nabla \left(\sum_{i=1}^n \sum_{j=1}^n (\mathbf{a}_i^\top \mathbf{w} - \mathbf{b}_j^\top \mathbf{w})^2 \right) + \nabla \left(\frac{\lambda}{2} \|\mathbf{w}\|_2^2 \right)$.

We know $\nabla \|\mathbf{w}\|_2^2 = \nabla (\mathbf{w}^\top \mathbf{w}) = 2\mathbf{w}$.


Hence as λ is a constant, $\nabla f(\mathbf{w}) = \nabla \left(\sum_{i=1}^n \sum_{j=1}^n (\mathbf{a}_i^\top \mathbf{w} - \mathbf{b}_j^\top \mathbf{w})^2 \right) + \lambda \mathbf{w}$.

$\mathbf{a}_i, \mathbf{b}_j \in \mathbb{R}^d$ are constants. $\nabla f(\mathbf{w}) = \sum_{i=1}^n \sum_{j=1}^n \nabla (\mathbf{a}_i^\top \mathbf{w} - \mathbf{b}_j^\top \mathbf{w})^2 + \lambda \mathbf{w}$.

Applying the chain rule, $\nabla f(\mathbf{w}) = 2 \sum_{i=1}^n \sum_{j=1}^n (\mathbf{a}_i^\top \mathbf{w} - \mathbf{b}_j^\top \mathbf{w})(\mathbf{a}_i - \mathbf{b}_j) + \lambda \mathbf{w}$.

Problem 2: Complexity


When designing algorithms, it's useful to be able to do quick back-of-the-envelope calculations to see how much time or space an algorithm needs. Hopefully, you'll start to get more intuition for this by being exposed to different types of problems.

- a.  [2 points] Suppose we have an $n \times n$ grid of points, where we'd like to place 4 arbitrary axis-aligned rectangles (i.e., the sides of the rectangle are parallel to the axes). Each corner of each rectangle must be one of the points in the grid, but otherwise there are no constraints on the location or size of the rectangles. For example, it is possible for all four corners of a single rectangle to be the same point (resulting in a rectangle of size 0) or for all 4 rectangles to be on top of each other. How many possible ways are there to place 4 rectangles on the grid? In general, we only care about asymptotic complexity, so give your answer in the form of $O(n^c)$ or $O(c^n)$ for some integer c .

Note: It is unnecessary to consider whether order matters in this problem, since we are asking for asymptotic complexity. You are free to assume either in your solution, as it doesn't matter change the final answer.

What we expect: A big-O bound for the number of possible ways to place 4 rectangles and some simple explanation/reasoning for the answer (~2 sentences).

There are $O(n)$ coordinates, $O(n^2)$ points (each defined by two coordinates), and $O(n^4)$ possible rectangles in the grid (each defined by two corner points). Since we need to select 4 rectangles, the total number of ways is $O(n^4) = O(n^{16})$.

- b.  [3 points] Suppose we have an $n \times 3n$ grid of points. We start at the point in the upper-left corner (the point at position $(1, 1)$), and we would like to reach the point at the lower-right corner (the point at position $(n, 3n)$) by taking single steps down or to the right. Suppose we are provided with a function $c(i, j)$ that outputs the cost associated with position (i, j) , and assume it takes constant time to compute for each position. Note that $c(i, j)$ can be negative. Define the cost of a path as the sum of $c(i, j)$ for all points (i, j) along the path, including both endpoints. Give an algorithm for computing the cost of the minimum-cost path from $(1, 1)$ to $(n, 3n)$ in the most efficient way (with the smallest big-O time complexity). What is the runtime (just give the big-O)?

What we expect: A description of the algorithm for computing the cost of the minimum-cost path as efficiently as possible (~5 sentences). The big-O runtime and a short explanation of how it arises from the algorithm.

We can compute the cost of the minimum-cost path by defining the following recurrence:

$f(i, j) = c(i, j) + \min(f(i - 1, j), f(i, j - 1))$ for $i, j = 1, \dots, n$. The initial state is $f(1, 1) = c(1, 1)$, and our solution will be found as $f(n, 3n)$. To handle the top and left edges, we can simply define $f(i, j) = \infty$ whenever $i = 0$ or $j = 0$. For each $i = 1, \dots, n; j = 1, \dots, 3n$, we need to compute $f(i, j)$. If we loop through i and j in increasing order, we will have already computed $f(i - 1, j)$ and $f(i, j - 1)$, so we can compute $f(i, j)$ in $O(1)$ time. Thus, the total running time is $O(n^2)$. Note that this is a basic case where we are memoizing the result of $f(i, j)$; if we didn't do that, then it would take time exponential in n , which is too slow.

Problem 3: Ethical Issue Spotting

One of the goals of this course is to teach you how to tackle real-world problems with tools from AI. But real-world problems have real-world consequences. Along with technical skills, an important skill every practitioner of AI needs to develop is an awareness of the ethical issues associated with AI. The purpose of this exercise is to practice spotting potential ethical concerns in applications of AI - even seemingly innocuous ones.


In this question, you will explore the ethics of four different real-world scenarios using the ethics guidelines produced by a machine learning research venue, the NeurIPS conference. The [NeurIPS Ethical Guidelines](#) list sixteen non-exhaustive concerns under Potential Negative Social Impacts and General Ethical Conduct (the numbered lists). For each scenario, you will write a potential negative impacts statement. To do so, you will first determine if the algorithm / dataset / technique could have a potential negative social impact or violate general ethical conduct (again, the sixteen numbered items taken from the [NeurIPS Ethical Guidelines](#) page). If the scenario does violate ethical conduct or has potential negative social impacts, list one concern it violates and justify why you think that concern applies to the scenario. If you do **not** think the scenario has an ethical concern, explain how you came to that decision. Unlike earlier problems in the homework there are many possible good answers. If you can justify your answer, then you should feel confident that you have answered the question well.

Each of the scenarios is drawn from a real AI research paper. The ethics of AI research closely mirror the potential real-world consequences of deploying AI, and the lessons you'll draw from this exercise will certainly be applicable to deploying AI at scale. As a note, you are **not** required to read the original papers, but we have linked to them in case they might be useful. Furthermore, you are welcome to respond to anything in the linked article that's not mentioned in the written scenario, but the scenarios as described here should provide enough detail to find at least one concern.


What we expect: A 2-5 sentence paragraph for each of the scenarios where you either A. identify at least one ethical concern from the [NeurIPS Ethical Guidelines](#) and justify why you think it applies, or B. state that you don't think a concern exists and justify why that's the case. Chosen scenarios may have anywhere from zero to multiple concerns that match, but you are only required to pick one concern (if it exists) and justify your decision accordingly. Furthermore, copy out and underline the ethical checklist item to which you are referring as part of your answer (i.e., [Severely damage the environment](#)). We have also included a citation in the example solution below, but you are not required to add citations to your response.

Example Scenario: You work for a U.S. hospital that has recently implemented a new intervention program that enrolls at-risk patients in programs to help address their chronic medical issues proactively before the patients end up in the hospital. The intervention program automatically identifies at-risk patients by predicting patients' risk scores, which are measured in terms of healthcare costs. However, you notice that for a given risk score tier, the Black patients are considerably sicker when enrolled than white patients, even though their assigned illness risk score is identical. You manually re-assign patients' risk scores based on their current symptoms and notice that the percentage of Black patients who would be enrolled has increased from 17% to over 45% [1].


Example Solution: This algorithm has likely encoded, contains, or potentially exacerbates bias against people of a certain race or ethnicity since the algorithm predicts healthcare costs. Because access to medical care in the U.S. is unequal, Black patients tend to have lower healthcare costs than their white counterparts [2]. Thus the algorithm will incorrectly predict that they are at lower risk.

- a.  [2 points] An investment firm develops a simple machine learning model to predict whether an individual is likely to default on a loan from a variety of factors, including location, age, credit score, and public record. After looking through their results, you find that the model predicts mainly based on location and that the model mainly accepts loans from urban centers and denies loans from rural applicants [3]. Furthermore, looking at the gender and ethnicity of the applicants, you find that the model has a significantly higher false positive rate for Black and male applicants than for other groups. In a false positive prediction, a model misclassifies someone who does not default as likely to default.


Example solution: This scenario could potentially have a detrimental effect on people's livelihood or economic security. It appears the algorithm has learned a spurious correlation where it thinks rural dwellers are more likely to default than urban dwellers and thus rejects rural dwellers' loan applications more frequently. If that's the case, using the algorithm would reduce rural dwellers' access to credit and negatively impact their businesses and livelihoods. Separately, the application will also likely encode, contain, or potentially exacerbate bias against people of a certain race or ethnicity, specifically Black male applicants. If the model is predicting that Black men are more likely to default than they actually are, then likely the model has also learned a spurious correlation between race and loan acceptance, and using it would likely exacerbate economic bias towards an already marginalized community.

- b.  [2 points] Stylometry is a way of predicting the author of contested or anonymous text by analyzing the writing patterns in the anonymous text and other texts written by the potential authors. Recently, highly accurate machine learning algorithms have been developed for this task. While these models are typically used to analyze historical documents and literature, they could be used for de-anonymizing a wide range of texts, including code [4].

Example solution: This scenario could potentially develop or extend harmful forms of surveillance. For example, stylometry could be used to de-anonymize online texts for surveillance purposes."

- c.  [2 points] A research group scraped millions of faces of celebrities off of Google images to develop facial recognition technology [5]. The celebrities did not give permission for their images to be used in the dataset and many of the images are copyrighted. For copyrighted photos, the dataset provides URL links to the original image along with bounding boxes for the face.

Example Solution: "This dataset violates the ethical guideline of consent to use or share the data, since members of the dataset did not consent to be used in the dataset. The dataset also contains any personally identifiable information or sensitive personally identifiable information which could be potentially degrading to the members of the dataset, since some of the example photos are of the celebrities in personal scenarios."

- d.  [2 points] Researchers have recently created a machine learning model that can predict plant species automatically directly from a single photo [6]. The model was trained using photos uploaded to the iNaturalist app by users who consented to use of their photos for research purposes, and the model is only used within the app to help users identify plants they might come across in the wild.

Example Solution: "This application does not violate any of the ethical guidelines. The users have consented to have their photos used for research purposes, which the trained model is being used for (namely, to assist in identifying species in the field)."

Problem 4: Programming

In this bonus, you will implement a bunch of short functions. The main purpose of this exercise is to familiarize yourself with Python, but as a bonus, the functions that you will implement may come in handy in subsequent homeworks.

Do not import any outside libraries (e.g. numpy). Only standard Python libraries and/or the libraries imported in the starter code are allowed.

If you're new to Python, the following provide pointers to various tutorials and examples for the language:

- [Python for Programmers](#)
- [Example programs of increasing complexity](#)

What we expect: Python code implementing the functions provided in [submission.py](#). Try to make your code as clean and simple as possible and be sure to write your answers between the begin answer and end answer comments.

- a.  [3 points] Implement `find_alphabetically_first_word` in [submission.py](#).

- b.  [3 points] Implement `euclidean_distance` in [submission.py](#).

- c.  [6 points] Implement `mutate_sentences` in [submission.py](#).

- d.  [4 points] Implement `sparse_vector_dot_product` in [submission.py](#).

- e.  [4 points] Implement `increment_sparse_vector` in [submission.py](#).

- f.  [4 points] Implement `find_nonsingleton_words` in [submission.py](#).

Submission

Submission is done on Gradescope.

Written: When submitting the written parts, make sure to select **all** the pages that contain part of your answer for that problem, or else you will not get credit. To double check after submission, you can click on each problem link on the right side, and it should show the pages that are selected for that problem.

Programming: After you submit, the autograder will take a few minutes to run. Check back after it runs to make sure that your submission succeeded. If your autograder crashes, you will receive a 0 on the programming part of the assignment. Note: the only file to be submitted to Gradescope is [submission.py](#).

More details can be found in the Submission section on the course website.

[1] Obermeyer et al. Dissecting racial bias in an algorithm used to manage the health of populations. 2019.

[2] Institute of Medicine of the National Academies. Unequal Treatment: Confronting Racial and Ethnic Disparities in Health Care. 2003.

[3] Imperial College London. Loan Default Prediction Dataset. 2014.

[4] Caliskan-Islam et. al. De-anonymizing programmers via code stylometry. 2015.

[5] Parkhi et al. VGG Face Dataset. 2015.

[6] iNaturalist. A new vision model. 2020.