

Assignment 1. Foundations

Hwanjo Yu
CSED342 - Artificial Intelligence

Contact: TA Seongje Lee (lsj720@postech.ac.kr), Jaehyun Lee (jminy8@postech.ac.kr)

General Instructions


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


You should write both types of answers in `submission.py` between

```
# BEGIN_YOUR_ANSWER
```

and

```
# END_YOUR_ANSWER
```

 This icon means a written answer is expected. Some of these problems are multiple choice questions that impose negative scores if the answers are incorrect. So, don't write answers unless you are confident.

 This icon means you should write code. you can add other helper functions outside the answer 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 all the tests, type

```
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`.

Problems

Problem 1. Optimization and probability

In this class, we will cast 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.

Read the following problems and write your answers in `submission.py`.

Problem 1a [3 points]

Suppose the probability of a coin turning up heads is $0 < p < 1$, and that we flip it 5 times and get $\{H, T, H, T, T\}$. We know the probability (likelihood) of obtaining this sequence is $L(p) = p(1-p)p(1-p)(1-p) = p^2(1-p)^3$. Now let's go backwards and ask the question: what is the value of p that maximizes $L(p)$?

Hint: Consider taking the derivative of $\log L(p)$. Taking the derivative of $L(p)$ works too, but it is cleaner and more natural to differentiate $\log L(p)$. You can verify for yourself that the value of p which minimizes $\log L(p)$ must also minimize $L(p)$.

Problem 1b [3 points]

Let's practice taking gradients, which is a key operation for being able to optimize continuous functions. For $\mathbf{w} \in \mathbb{R}^d$ and constants $a_i, b_j \in \mathbb{R}^d$ and $\lambda \in \mathbb{R}$, define the scalar-valued function

$$f(\mathbf{w}) = \sum_{i=1}^n \sum_{j=1}^n (a_i^\top \mathbf{w} - b_j^\top \mathbf{w})^2 + \lambda \|\mathbf{w}\|_2^2,$$

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

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

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

What's the formula for the gradient? Choose one of the following expressions:

- (1) $\nabla_{\mathbf{w}} f(\mathbf{w}) = 2 \left(\sum_{i=1}^n \sum_{j=1}^n a_i^\top b_j \right) \mathbf{w} + 2\lambda \mathbf{w}$
- (2) $\nabla_{\mathbf{w}} f(\mathbf{w}) = -2 \left(\sum_{i=1}^n \sum_{j=1}^n a_i^\top b_j \right) \mathbf{w} + 2\lambda \mathbf{w}$
- (3) $\nabla_{\mathbf{w}} f(\mathbf{w}) = 2 \left(\sum_{i=1}^n \sum_{j=1}^n (a_i - b_j)^\top (a_i - b_j) \right) \mathbf{w} + 2\lambda \mathbf{w}$

$$(4) \nabla_{\mathbf{w}} f(\mathbf{w}) = 2 \left(\sum_{i=1}^n \sum_{j=1}^n (a_i - b_j)(a_i - b_j)^\top \right) \mathbf{w} + 2\lambda \mathbf{w}$$

Hint: you may need the following formulas for matrix differentiation:

- $\frac{\partial \mathbf{u}^\top \mathbf{w}}{\partial \mathbf{w}} = \frac{\partial \mathbf{w}^\top \mathbf{u}}{\partial \mathbf{w}} = \mathbf{u}$
- $\frac{\partial \mathbf{w}^\top \mathbf{w}}{\partial \mathbf{w}} = 2\mathbf{w}$
- $\frac{\partial \mathbf{w}^\top \mathbf{M} \mathbf{w}}{\partial \mathbf{w}} = (\mathbf{M} + \mathbf{M}^\top) \mathbf{w}$

where \mathbf{w} and \mathbf{u} are vectors and \mathbf{M} is a matrix.

Problem 2: Programming

In this problem, 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 will come in handy in subsequent homeworks.

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

- Python for Programmers:
<https://wiki.python.org/moin/BeginnersGuide/Programmers>
- Example programs of increasing complexity:
<https://wiki.python.org/moin/SimplePrograms>

Problem 2a [4 points]

Implement *getWordKey* for *computeMaxWordLength* in `submission.py`.

Problem 2b [3 points]

Implement *manhattanDistance* in `submission.py`.

Problem 2c [7 points]

Implement *countMutatedSentences* in `submission.py`.

Problem 2d [2 points]

Implement *dotProduct* in `submission.py`.

Problem 2e [2 points]

Implement *incrementSparseVector* in `submission.py`.

Problem 2f [3 points]

Implement *computeMostFrequentWord* in `submission.py`.