DSC 140A - Homework 03

Due: Wednesday, April 24

Instructions: Write your solutions to the following problems either by typing them or handwriting them on another piece of paper. Show your work or provide justification unless otherwise noted. If you write code to solve a problem, include the code by copy/pasting or as a screenshot. You may use `numpy`, `matplotlib` (or another plotting library), and any standard library module, but no other third-party libraries unless specified. Submit homeworks via Gradescope by 11:59 PM.

Problem 1.

Suppose that $f : \mathbb{R}^d \to \mathbb{R}$ is convex and $g : \mathbb{R} \to \mathbb{R}$ is convex and non-decreasing. That is, if $a > b$, then $g(a) \geq g(b)$.

Show that the composition of these functions, $h(\vec{x}) = g(f(\vec{x}))$, is also convex.

Hint: you'll want to go back to the definition to show this is true. A similar problem was solved in discussion.

Solution:

We need to show that, for any $\vec{a}, \vec{b} \in \mathbb{R}^d$ and $t \in [0, 1]$, the following holds:

$$h(t\vec{a} + (1-t)\vec{b}) \leq th(\vec{a}) + (1-t)h(\vec{b}).$$

Starting with the left-hand side and using the definition of h, we have:

$$h(t\vec{a} + (1-t)\vec{b}) = g(f(t\vec{a} + (1-t)\vec{b}))$$

Because f is convex, we know that $f(t\vec{a} + (1-t)\vec{b}) \leq tf(\vec{a}) + (1-t)f(\vec{b})$. Combining this with the fact that g is non-decreasing, we have:

$$\leq g(tf(\vec{a}) + (1-t)f(\vec{b}))$$

Now we use the fact that g is convex to write:

$$\leq tg(f(\vec{a})) + (1-t)g(f(\vec{b}))$$

Which we recognize as the right-hand side of the inequality we want to show:

$$= th(\vec{a}) + (1-t)h(\vec{b}).$$

Problem 2.

The *hinge loss* is defined to be

$$L_{\text{hinge}}(\vec{w}, \vec{x}, y) = \max\{0, 1 - y\,\vec{w} \cdot \text{Aug}(\vec{x})\}$$

Consider the below quantity, which is called the *regularized* empirical risk:

$$R(\vec{w}) = \frac{C}{n} \sum_{i=1}^{n} L_{\text{hinge}}(\vec{w}, \vec{x}^{(i)}, y_i) + \|\vec{w}\|^2$$

Here, C is a positive constant, n is the number of data points, $\vec{x}^{(i)}$ is the ith data point, y_i is the label of the ith data point, and $\vec{w}$ is a vector of weights.

Show that $R(\vec{w})$ is a convex function of $\vec{w}$.

(We will discuss regularization and the hinge loss in future lectures; for this problem it's not necessary to know anything about them apart from the definitions given above.)

Hint: you will probably *not* want to use the formal definition of convexity here. Instead, you'll want to show that R is composed of simpler functions which themselves are convex.

Solution: We will break the function into pieces and show that each piece is convex, and so the sum of the pieces is also convex.

First, $\|\vec{w}\|^2$ is a convex function of $\vec{w}$. This can be shown in several ways; the simplest might be the second derivative test. Remember that $\|\vec{w}\|^2 = \vec{w} \cdot \vec{w} = w_1^2 + w_2^2 + \ldots + w_d^2$. The first derivative $\partial/\partial w_i$ is $2w_i$, and the second derivative $\partial/\partial w_i \partial w_j$ is 2 if $i = j$ and 0 otherwise. Therefore, the Hessian matrix is:

$$\begin{pmatrix} 2 & 0 & \cdots & 0 \\ 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 2 \end{pmatrix} = 2I,$$

where I is the identity matrix. This matrix is positive semi-definite, so $\|\vec{w}\|^2$ is convex.

To show that $C/n \sum_{i=1}^{n} L_{\text{hinge}}(\vec{w}, \vec{x}^{(i)}, y_i)$ is convex, we will show that the hinge loss is convex and then use the fact that the sum of convex functions is convex.

The hinge loss can be written as the maximum of two functions:

$$L_{\text{hinge}}(\vec{w}, \vec{x}, y) = \max\{0, 1 - y\,\vec{w} \cdot \text{Aug}(\vec{x})\}.$$

From lecture, we know that the maximum of two convex functions is convex, so we only need to show that each of the two functions is convex. The first function, 0, is a constant function and is therefore convex. The second function, $1 - y\,\vec{w} \cdot \text{Aug}(\vec{x})$, is also convex. We can most easily show this using the result from Discussion, which says that functions of the form $a\vec{x} \cdot \vec{w} - b$ are convex functions of $\vec{w}$.

To show that $1 - y\,\vec{w} \cdot \text{Aug}(\vec{x})$ is of this form, we set $a = -y$ and $b = -1$. Therefore, the function is convex, the hinge loss is convex, and the sum of hinge losses is convex.

Lastly, since $C/n \geq 0$, the first term is convex, and so the whole thing is convex.

Problem 3.

Consider a linear prediction function H used for binary classification, and assume that when the output of H is positive we predict for class +1, and when it's negative we predict for class -1. This means that the **decision boundary** is where $H(\vec{x}) = 0$.

In lecture, we saw that a linear prediction function has the form:

$$\begin{aligned} H(\vec{x}) &= w_0 + w_1 x_1 + \ldots + w_d x_d \\ &= \vec{w} \cdot \text{Aug}(\vec{x}) \end{aligned}$$

where $\vec{w} = (w_0, w_1, \ldots, w_d)^T$. In this problem, it will also be useful to define the vector $\vec{w}' = (w_1, \ldots, w_d)^T$, which is the same as $\vec{w}$ except that it does not include w_0. Note that $\vec{x} \cdot \vec{w}' = w_1 x_1 + \ldots w_d x_d$. With this definition, we can write $H(\vec{x})$ in a slightly different way:

$$H(\vec{x}) = w_0 + \vec{w}' \cdot \vec{x}$$

Remember this formula, as it will be useful several times below!

Over the course of this problem, we'll answer the question: how is the magnitude of $H(\vec{x})$ related to the distance between $\vec{x}$ and the decision boundary?

Note: for this problem it may be useful to review the properties of the dot product and vector algebra. In particular, remember that $\|u\|$ denotes the norm (length) of a vector, and that $\vec{u} \cdot \vec{u} = \|u\|^2$. By dividing a vector by its norm, as in $\vec{u}/\|\vec{u}\|$, we obtain a *unit vector* with unit length – a unit vector is useful for specifying a direction. To find the component of a vector $\vec{u}$ that points in the same direction as another vector $\vec{v}$, we write $\vec{u} \cdot \vec{v}/\|\vec{v}\|$. Also remember that $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$, and that $\vec{u} \cdot (\vec{v} + \vec{w}) = \vec{u} \cdot \vec{v} + \vec{u} \cdot \vec{w}$.

a) Show that for any point $\vec{z}$ on the decision boundary, $\vec{w}' \cdot \vec{z} = -w_0$.

Hint: use that fact that $H(\vec{z}) = 0$.

Solution: For any point $\vec{z}$, we have:

$$H(\vec{z}) = \vec{w}' \cdot \vec{z} + w_0$$

Since $\vec{z}$ is on the decision boundary, though, $H(\vec{z}) = 0$, so the above is equal to zero, and $\vec{w}' \cdot \vec{z} = -w_0$.

b) Argue that $\vec{w}'$ is orthogonal to the decision boundary.

Hint: take two arbitrary points $\vec{x}^{(1)}$ and $\vec{x}^{(2)}$ that are assumed to be on the decision boundary. Then, since we know that the boundary is linear (it is a line, plane, etc.), the difference of these vectors, $\vec{\delta} = \vec{x}^{(1)} - \vec{x}^{(2)}$ is parallel to the boundary. To show that $\vec{w}'$ is orthogonal to the boundary, it suffices to show that $\vec{w}' \cdot \vec{\delta} = 0$. Make sure you somehow use the fact that $\vec{x}^{(1)}$ and $\vec{x}^{(2)}$ are on the decision boundary.

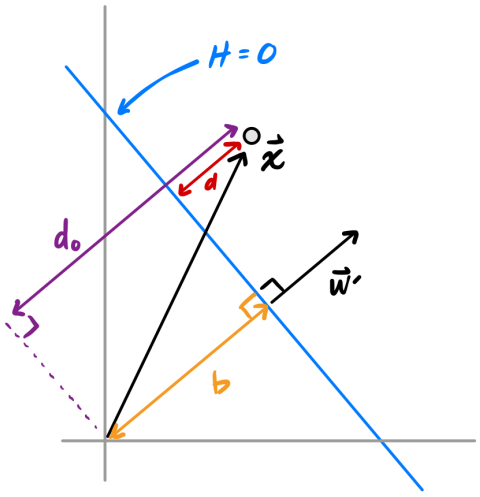
Solution:

$$\begin{aligned} \vec{w}' \cdot \vec{\delta} &= \vec{w}' \cdot (\vec{x}^{(1)} - \vec{x}^{(2)}) \\ &= \vec{w}' \cdot \vec{x}^{(1)} - \vec{w}' \cdot \vec{x}^{(2)} \end{aligned}$$

Because $\vec{x}^{(1)}$ and $\vec{x}^{(2)}$ are on the boundary, we know from the previous part that $\vec{w}' \cdot \vec{x}^{(1)} = -w_0$ and $\vec{w}' \cdot \vec{x}^{(2)} = -w_0$. Therefore:

$$\begin{aligned} &= -w_0 - (-w_0) \\ &= 0 \end{aligned}$$

c) Now that we know that $\vec{w}'$ is orthogonal to the decision boundary, we can draw a better picture of the situation:



The blue line is the decision boundary – it is where $H = 0$. We have drawn an arbitary point $\vec{x}$, along with several distances:

- d: the (signed) distance from the decision boundary to $\vec{x}$
- b: the distance from the the origin to the decision boundary
- d_0: the length of the component of $\vec{x}$ that is orthogonal to the decision boundary.

We're most interested in knowing d. First, though, we need to find b.

Consider the vector $b\vec{w}'/\|\vec{w}'\|$; this is a vector from the origin to the decision boundary that is orthogonal to the boundary and with length b.

Since this vector is on the decision boundary, $H(b\vec{w}'/\|\vec{w}'\|) = 0$. Using this fact, show that $b = -\dfrac{w_0}{\|\vec{w}'\|}$.

Hint: first show that $H(b\vec{w}'/\|\vec{w}'\|) = b\|\vec{w}'\| + w_0$, then set this to zero and solve for b.

Solution: First, we compute:

$$\begin{aligned} H(\vec{z}) &= H\left(\frac{b\vec{w}'}{\|\vec{w}'\|}\right) \\ &= \frac{b\vec{w}'}{\|\vec{w}'\|} \cdot \vec{w}' + w_0 \\ &= \frac{b\|\vec{w}'\|^2}{\|\vec{w}'\|} + w_0 \\ &= b\|\vec{w}'\| + w_0 \end{aligned}$$

Since $\vec{z}$ is on the decision boundary, $H(\vec{z}) = 0$, and so $b\|\vec{w}'\| + w_0 = 0$, meaning that $b = -w_0/\|\vec{w}'\|$.

d) Recall that d_0 is the component of $\vec{x}$ that is orthogonal to the decision boundary; this is simply $\vec{x} \cdot \vec{w}'/\|\vec{w}'\|$. From the picture, $d_0 = d + b$. We know d_0 and b, and can therefore solve for d.

Use this to show that $|d| = |H(\vec{x})|/\|\vec{w}'\|$.

Solution:

$$\begin{aligned} d &= d_0 - b \\ &= \frac{\vec{x} \cdot \vec{w}'}{\|\vec{w}'\|} + \frac{w_0}{\|\vec{w}'\|} \\ &= \frac{\vec{x} \cdot \vec{w}' + w_0}{\|\vec{w}'\|} \\ &= \frac{H(\vec{x})}{\|\vec{w}'\|} \end{aligned}$$

We have shown that the distance between $\vec{x}$ and the decision boundary is proportional to the output of the prediction function, $H(\vec{x})$. This gives us a very useful interpretation of $|H(\vec{x})|$! For example, this means if $H(\vec{x}^{(1)}) > H(\vec{x}^{(2)}) > 0$, then $\vec{x}^{(1)}$ is further from the decision boundary than $\vec{x}^{(2)}$.

Problem 4.

The file below contains data suitable for a binary classification problem. `https://f000.backblazeb2.com/file/jeldridge-data/003-two_clusters/data.csv`

The file contains three columns: x_1, x_2, and y. The first two columns are the features, and the third column is the label.

a) Write a function to compute the empirical risk for the perceptron loss, $R_{\text{tron}}(\vec{w})$, with respect to the data and the parameters $\vec{w}$.

Use your function to compute the empirical risk for the vector $(1, 1, 1)^T$, and report the result. Include your code.

Solution:

```
import numpy as np
import matplotlib.pyplot as plt


data = np.loadtxt("data.csv", delimiter=",")

X = data[:, :-1]
X = np.column_stack((np.ones(X.shape[0]), X))
y = data[:, -1]


def perceptron_loss(w, x, y):
    return np.maximum(0, -y * x @ w)


def perceptron_risk(w):
    return np.mean([perceptron_loss(w, x, y) for (x, y) in zip(X, y)])
```

Using the above function, we can compute the empirical risk for the vector $(1, 1, 1)^T$ as follows:

```
print(empirical_risk(np.array([1, 1, 1])))
```

We get a value of approximately 1.1249.

b) Write a function to compute the subgradient of the empirical risk for the perceptron loss, and use

your code to compute the subgradient at the point $(1, 1, 1)^T$. Report the result.

Solution:

```
def subgradient_of_loss(w, x, y):
    h = x @ w
    if y * h <= 0:
        return -y * x
    else:
        return np.zeros_like(w)


def subgradient_of_risk(w):
    return np.mean([subgradient_of_loss(w, x, y) for (x, y) in zip(X, y)], axis=0)
```

Using this function to compute the subgradient at the point $(1, 1, 1)^T$, we get the result $[0.24, 0.304, 0.5809]$.

c) Run subgradient descent to train a perceptron on the data. Use the initial vector $(1, 1, 1)^T$. Report the learned parameter vector.

Solution: We can use the usual code for gradient descent, but with the subgradient of the empirical risk for the perceptron loss. The code is as follows:

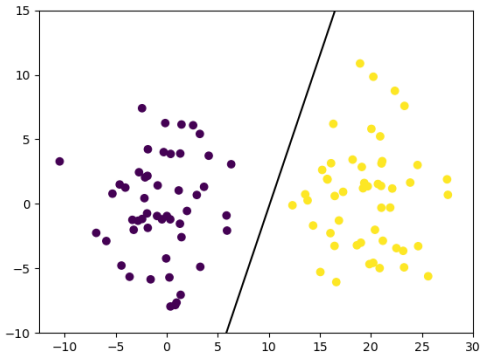
```
def gradient_descent(gradient, z_0, learning_rate, stop_threshold):
    z = z_0
    while True:
        z_new = z - learning_rate * gradient(z)
        if np.linalg.norm(z_new - z) < stop_threshold:
            break
        z = z_new
    return z_new


w_opt = gradient_descent(subgradient_of_risk, np.array([1, 1, 1]), 0.1, 1e-6)
```

We converge to a parameter vector of approximately $\vec{w}^* = [-0.211, 0.02089, -0.0089]$. However, note that when the data are linearly separable there are infinitely many parameter vectors that achieve zero empirical risk. That is, the weight vector above is not unique, and you might have found a different one.

d) Plot the data and the decision boundary of the perceptron you learned above. Each point should be colored according to its label.

Hint: There are two ways to plot the decision boundary. One way is to use the fact that $\text{Aug}(\vec{x}) \cdot \vec{w} = 0$ for points on the decision boundary to solve for x_2 in terms of x_1, giving you the equation of a line. Another approach is to use `plt.contour` to plot the contours of the prediction function, $H(\vec{x})$; in particular, we plot the contour where $H(\vec{x}) = 0$. The second approach is more general and can be used for any prediction function, but you'll want to read the docs to understand how to use it.

Solution:



This figure was generated using the following code:

```
plt.figure()

# plot the data; we can use c=y to color points by their class
plt.scatter(X[:, 1], X[:, 2], c=y)

# plot the decision boundary
X1 = np.linspace(-10, 30, 100)
X2 = np.linspace(-10, 15, 100)
X1, X2 = np.meshgrid(X1, X2)
H = w_opt[0] + w_opt[1] * X1 + w_opt[2] * X2
plt.contour(X1, X2, H, levels=[0], colors="k")
```