CIS 520, Machine Learning, Fall 2019
Homework 5
Due: Monday, November 2nd, 11:59pm
Submit to Gradescope

Instructions. Please write up your responses to the following problems clearly and concisely. We require you to write up your responses using LaTeX; we have provided a LaTeX template, available on Canvas, to make this easier. **Submit your answers in PDF form to Gradescope. We will not accept paper copies of the homework.**

Collaboration. You are allowed and encouraged to work together. You may discuss the **written homework** to understand the problem and reach a solution in groups. However, **it is recommended that each student also write down the solution independently and without referring to written notes from the joint session.** You must understand the solution well enough to reconstruct it by yourself. (This is for your own benefit: you have to take the exams alone.)

Learning Objectives

After completing this assignment, you will be able to:

- Understand what perceptrons attempt to optimize and when they succeed
- Understand the relationship between singular values/vectors and eigenvectors/values
- Understand the relationship between PCA reconstruction error and eigenvalues

Deliverables

This homework can be completed individually or in groups of 2. You need to make one submission per group. Make sure to add your team member's name on Gradescope when submitting the homework's written and coding part. Please view the following link if you are not familiar with adding group member's name for Gradescope submission - https://www.gradescope.com/courses/145384

1. **A PDF compilation of `hw5.tex`**
2. **A `.ipynb` file with the functions implemented**

1 Perceptron vs. Winnow [25 points]

For online binary classification problems, you saw the perceptron algorithm in class. In the linearly separable case, we can bound the number of mistakes the perceptron algorithm makes:

Theorem (Perceptron mistake bound). *Suppose that the examples seen in T trials are linearly separable by a non-negative weight vector, i.e. that there exists a weight vector $\mathbf{u} \in \mathbb{R}^d$ and $\gamma > 0$ such that*

$$y_t(\mathbf{u}^\top \mathbf{x}_t) \geq \gamma \quad \text{for all } t \in \{1, \ldots, T\}.$$

Also let $\|\mathbf{x}_t\|_2 \leq R_2$ for all t. If $\|\mathbf{u}\|_2$, γ, and R_2 are known, then the number of mistakes made by the perceptron algorithm in the T trials is at most

$$\left(\frac{R_2^2 \|\mathbf{u}\|_2^2}{\gamma^2}\right)$$

Another algorithm that is used for such problems is the *Winnow* algorithm, which also maintains a linear classification model $\mathbf{u}_t$, but makes *multiplicative updates* to $\mathbf{u}_t$ rather than additive ones (such multiplicative updates now play an important role in many modern optimization algorithms). In this case, the weight vectors $\mathbf{u}_t$ always have positive entries that add up to 1:

Algorithm **Winnow**

Learning rate parameter $\eta > 0$
Initial weight vector $\mathbf{u}_1 = (\frac{1}{d}, \ldots, \frac{1}{d}) \in \mathbb{R}^d$
For $t = 1, \ldots, T$:
- Receive instance $\mathbf{x}_t \in \mathbb{R}^d$
- Predict $\widehat{y}_t = \text{sign}(\mathbf{u}_t^\top \mathbf{x}_t)$
- Receive true label $y_t \in \{\pm 1\}$
- Update: If $\widehat{y}_t \neq y_t$ then

$$\text{For each } i \in \{1, \ldots, d\}: \quad u_{t+1,i} \leftarrow \frac{u_{t,i} \exp(\eta\, y_t x_{t,i})}{Z_t}$$
$$\text{where } Z_t = \sum_{j=1}^n u_{t,j} \exp(\eta\, y_t x_{t,j})$$

  else
    $\mathbf{u}^{t+1} \leftarrow \mathbf{u}^t$

For examples that are linearly separable by a non-negative weight vector, the Winnow algorithm is known to have the following mistake bound:

Theorem (Winnow mistake bound). *Suppose that the examples seen in T trials are linearly separable by a non-negative weight vector, i.e. that there exists a weight vector $\mathbf{u} \in \mathbb{R}_+^d$ and $\gamma > 0$ such that*

$$y_t(\mathbf{u}^\top \mathbf{x}_t) \geq \gamma \quad \text{for all } t \in \{1, \ldots, T\}.$$

Also suppose $\|\mathbf{x}_t\|_\infty \leq R_\infty$ for all t. If $\|\mathbf{u}\|_1$, γ, and R_∞ are known, then one can select the learning rate parameter η in a way that the number of mistakes in the T trials is at most

$$2\left(\frac{R_\infty^2 \|\mathbf{u}\|_1^2}{\gamma^2}\right) \ln(d)\,.$$

(a) Sparse target vector u, dense feature vectors $\mathbf{x}_t$. [10 points]
Suppose you are in a setting with high-dimensional features (large d), and that all features are of roughly constant magnitude; for simplicity, suppose $\mathbf{x}_t \in \{\pm 1\}^d$ for all t. Suppose you are told that the examples in T trials are linearly separable by a sparse weight vector $\mathbf{u} \in \{0, 1\}^d$ which has only $k \ll d$ non-zero entries, and that you are given $\gamma > 0$ such that $y_t(\mathbf{u}^\top \mathbf{x}_t) > \gamma$ for all t. Calculate upper bounds on the numbers of mistakes that would be made by both Perceptron and Winnow in terms of k, γ and d. Which algorithm would be a better choice here?

(b) Dense target vector u, sparse feature vectors $\mathbf{x}_t$. [10 points]
Suppose you are in a setting with high-dimensional features (large d), and that the feature vectors are sparse; for simplicity, suppose $\mathbf{x}_t \in \{0, -1, +1\}^d$ for all t and that each $\mathbf{x}_t$ has $k \ll d$ non-zero entries. Suppose you are told the examples in T trials are linearly separable by a dense weight vector $\mathbf{u} \in \mathbb{R}^d_+$ with $\|\mathbf{u}\|_1 = d$ and $\|\mathbf{u}\|_2 \leq 2\sqrt{d}$, and that you are given $\gamma > 0$ such that $y_t(\mathbf{u}^\top \mathbf{x}_t) > \gamma$ for all t. Calculate upper bounds on the numbers of mistakes that would be made by both Perceptron and Winnow in terms of k, γ and d. Which algorithm would be a better choice here?

(c) If your problem has non-negative feature vectors $\mathbf{x}_t \in \mathbb{R}^d_+$, is the Winnow algorithm a meaningful choice? Why or why not? [5 points]

2 Singular Value Decomposition [25 points]

1. **[10 points]** Let $\mathbf{X}$ be a n by p matrix. Show that if $\mathbf{X}$ has a rank p (all its columns are linearly independent), and $n > p$, then using the p-dimensional pseudo-inverse $\mathbf{X}^+ = \mathbf{V}_k \Lambda_k^{-1} \mathbf{U}_k^T$ in $\widehat{\mathbf{w}} = \mathbf{X}^+ \mathbf{y}$ with $k = p$ solves the least squares problem $\widehat{\mathbf{w}} = \arg\min_w (\mathbf{y} - X\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$.
   Hint: A useful matrix derivative identity is: $\frac{\partial}{\partial \mathbf{s}} (\mathbf{x} - \mathbf{A}\mathbf{s})^T (\mathbf{x} - \mathbf{A}\mathbf{s}) = -2\mathbf{A}^T (\mathbf{x} - \mathbf{A}\mathbf{s})$
2. **[10 points]** Given the eigenvectors of $\mathbf{X}\mathbf{X}^T$ as $(\mathbf{u}_1, ..., \mathbf{u}_k)$ and corresponding eigenvalues as $(\lambda_1, ..., \lambda_k)$, give an expression for computing an eigenvector $\mathbf{v}_i$ of $\mathbf{X}^T\mathbf{X}$ in terms of $\mathbf{X}$, $\mathbf{u}_i$, and λ_i.
3. **[5 points]** Let $\mathbf{X}$ be a n by p matrix. Under what conditions (in terms of the relationship between n and p) would the above calculation be an efficient way to find the largest eigenvectors of $\mathbf{X}^T\mathbf{X}$? Please explain your reasoning.

3 Principal Component Analysis [33 points]

In this assignment, we will implement Principal Component Analysis and perform it on a simple 2-dimensional dataset. First fill out the helper functions provided to retrieve covariance matrices, eigenvectors/values, and component projections. Use these helper functions to implement a PCA function.

3.1 Part 1: Comparing Principal Components

1. **[6 points]** Report the eigenvectors and eigenvalues here.

2. **[4 points]** Express mathematically, explain why the first PC is the eigenvector associated with the largest eigenvalue?

3. **[2 points]** What can you say about the relationship between the first principal component and the second?

3.2 Part 2: Plotting Principal Components in Original Space

1. **[3 points]** Please describe how the principal components relate to the points.

2. **[2 points]** Paste the graph here of the plot of the given points (with both axis in same scale) as well as the lines representing the principal components in original space, with x1 in the x axis and x2 in the y axis.

3.3 Part 3: Plotting Data Projected onto Component Space

1. **[3 points]** Explain how the graph of points on principal component space relates to the graph of points on original space above.

2. **[3 points]** Explain the difference in distribution of points projected on the first component vs. projected on the second.

3. **[2 points]** Paste the plot of the given points (with both axis in same scale) in principal component space.

3.4 Part 4: PCA and Reconstruction Error

1. **[4 points]** Using the digit data set from the PCA worksheet (used in PCA Maximize Variance), what is the reconstruction error using the first and second principal components

2. **[4 points]** Mathematically express how you come up with the answer, and explain how PCA is minimizing the reconstruction error. (Hint: write out the expression in terms of eigenvectors, and think about the smallest eigenvalues.)

4 Principal Component Analysis on Faces [17 points]

Now we will perform PCA on images of faces and see how reducing the latent dimensions of our images affects the images reconstructions. First, start by uncommenting the code below to retrieve the faces dataset.

4.1 Part 1: PCA with SVD and Resulting Eigenfaces

1. **[2 points]** To check the outputs of your PCA functions, report the singular values here.

2. **[3 points]** Please describe what the eigenfaces look like. What do you expect to observe with the eigenfaces associated with larger eigenvalues?

3. **[2 points]** Please insert your eigenfaces output here.

4.2 Part 2: Reconstructing Faces

1. **[2 points]** Paste in the reconstructed faces plot. Compare the reconstructed images to the original images. How are they similar and how are they different? Shortly explain why they are different?

2. **[2 points]** What do you expect to see from the reconstructed images as the number of principal components chosen for PCA increases? Please explain why.

4.3 Part 3: Variance Explanation

1. **[2 points]** How do you expect (based on theory; please be precise!) the plot of variance explained as the number of components to relate to the eigenvalues of the corresponding components?

2. **[2 points]** What is the relation between reconstruction error and the variance explained?

3. **[2 points]** Insert the three line plots of explanation vs. number of components, descending eigenvalues vs. number of components, and reconstruction error vs. number of components here.