# Introduction to Machine Learning

**ECE 580**
**Spring 2025**

**HW #5, Due 04/06/25 11:59pm**

*Submission Instructions*

Submit your work to the corresponding assignment in Gradescope. Although Gradescope accepts multiple file formats, they strongly recommend submitting your assignment as a single PDF file.

It is your responsibility to ensure the uploaded file is: 1) the correct file, 2) complete (includes all pages), 3) legible, and 4) submitted on-time as determined by the Gradescope server system clock.

It is your responsibility to submit a multi-page PDF file and tag the pages that correspond to each question. Pages may be tagged after submission, even if the submission deadline has passed. If you are submitting close to the submission deadline, submit your assignment first then immediately return to tag pages.

When code is requested, submit a PDF print-out of your code. Submitting a URL for a cloud-based repository is insufficient.

*Late Submissions*

Late submissions will be accepted up to 5 days after the submission deadline, with the following point penalty applied if its late submission is not excused: [1]

1 day ($0^+$ to 24 hours) late: 2 point deduction ($\frac{1}{5}$ letter grade)
2 days ($24^+$ to 48 hours) late: 5 point deduction ($\frac{1}{2}$ letter grade)
3 days late: 10 point deduction (1 letter grade)
4 days late: 20 point deduction (2 letter grades)
5 days late: 30 point deduction (3 letter grades)
6 or more days late: score = 0 (not accepted for credit)

The late policy is designed to be minimally punitive for submissions up to 3 days late, yet encourages staying current with the coursework for our course by not allowing one assignment's late submission to overlap with the next assignment's submission.

A homework score will not drop below 0 as a result of applying the late penalty point deduction.

[1] One day = one 24-hour period or fraction thereof.

## The Power Method

1. In this problem, we will explore a simple algorithm that enables us to efficiently compute the top eigenvector of a general symmetric positive definite (PD) matrix $M$. This algorithm thus also enables us to find the top-k principal components.

   Let $M$ denote some symmetric matrix with eigenvalues $\lambda_1 > \lambda_2 > ...\lambda_d > 0$, and corresponding eigenvalues $u_1, u_2...u_d \in \mathbb{R}^d$. In other words. $M = \sum_{i=1}^{d} \lambda_i u_i u_i^\top = UDU^\top$, where $D$ is a diagonal matrix of $\lambda_i$'s, and $U$ is an orthogonal matrix, where the $i^{th}$ column of $U$ is $u_i$.

(5) (a) Show that for any $v \in \mathbb{R}^d$, we can write $v$ as

$$v = \sum_{i=1}^{d} \alpha[i] u_i,$$

where $\alpha \in \mathbb{R}^d$ is a vector of coefficients, and $\alpha[i]$ is its $i^{th}$ entry. Give explicit expression for $\alpha[i]$ in terms of $v$ and $u_i$.

(5) (b) For fixed $u_1...u_n, v$, is the choice of $\alpha$ in part (a) unique? Is it guaranteed to exist? Why or why not?

(15) (c) The Power Method (for matrix $M$) refers to the following iterative algorithm:

$$\begin{aligned} x_0 &= \text{some initial vector with } \|x_0\|_2 = 1 \\ \hat{x}_{t+1} &= M x_t \\ x_{t+1} &= \frac{\hat{x}_t}{\|\hat{x}_t\|_2}. \end{aligned}$$

Let $\alpha_t \in \mathbb{R}^d$ denote the vector of coefficients for $x_t$, i.e.

$$x_t = \sum_{i=1}^{d} \alpha_t[i] u_i.$$

Show that

$$\alpha_{t+1}[i] = \frac{\lambda_i \alpha_t[i]}{C},$$

where $C$ is some constant with respect to $i$. State the explicit expression for $C$ as a function of $\alpha_t$ and $\lambda_i$'s.

(5) (d) Assume that $\alpha_0[i] > 0$ for all $i = 1...d$. Show that

$$\alpha_t[i] = \frac{\lambda_i^t \alpha_t[i]}{C_t}.$$

Where $C_t$ is some constant wrt $i$. State the explicit expression for $C_t$ as a function of $t$, $\alpha_0$, and $\lambda_i$'s.

(10) (e) Show that

$$\sum_{i=2}^{d} \alpha_t[i]^2 \leq \left(\frac{\lambda_2}{\lambda_1}\right)^{2t} \left(\frac{\sum_{i=2}^{d} \alpha_0[i]^2}{\alpha_0[1]^2}\right)$$

(5) (f) Show that we can guarantee $\|X_T - u_1\|_2^2 \leq \varepsilon$ for

$$T \geq \frac{1}{2\log(\lambda_1/\lambda_2)} \log\left(\frac{\sum_{i=2}^{d} \alpha_0[i]^2}{\varepsilon \alpha_0[1]^2}\right)$$

## Optimality of PCA for Minimizing Reconstruction Error

2. In this question, we will verify the optimality of PCA for minimizing the reconstruction error. We are given $x_1 \ldots x_n \in \mathbb{R}^d$. We want to find $k$ orthonormal basis vectors $v_1 \ldots v_k$ to minimize the reconstruction error

$$\begin{aligned} \min_{v_1 \ldots v_d \in \mathbb{R}^d} \quad & \sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{k} (x_i^\top v_j) v_j \right\|_2^2 \\ \text{subject to} \quad & \|v_i\|_2 = 1 \text{ for all } i = 1 \ldots k \\ \text{and} \quad & v_i^\top v_j = 0 \text{ for all } i \neq j \end{aligned} \tag{1}$$

(10) (a) Let $X \in \mathbb{R}^{n \times d}$ denote the matrix whose $i^{th}$ row is $x_i^\top$. Let $V \in \mathbb{R}^{k \times d}$ denote the matrix whose $i^{th}$ row is $v_i$. Then

$$\sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{k} (x_i^\top v_j) v_j \right\|_2^2 = \left\| X - X V^\top V \right\|_F^2,$$

where $\|M\|_F = \sqrt{\text{tr}(M^\top M)} = \sqrt{\text{tr}(M M^\top)}$ is the Frobenius norm.

(5) (b) For $V$ as defined in (a), show that the conditions

1. $\|v_i\|_2 = 1$ for all $i = 1 \ldots k$
2. $v_i^\top v_j = 0$ for all $i \neq j$

is equivalent to the condition that

$$V V^\top = I_{k \times k}.$$

(5) (c) Combining (a) and (b), we verify that the optimization problem in (1) is equivalent to

$$\begin{aligned} \min_{V \in \mathbb{R}^{k \times d}} \quad & \left\| X - X V^\top V \right\|_F^2 \\ \text{subject to} \quad & V V^\top = I_{k \times k} \end{aligned} \tag{2}$$

Show that for any $X$, the optimal value of (2) is **lower bounded by** the optimal value of (3) below:

$$\begin{aligned} \min_{M \in \mathbb{R}^{n \times d}} \quad & \|X - M\|_F^2 \\ \text{subject to} \quad & \text{rank}(M) \leq k. \end{aligned} \tag{3}$$

**Hint:** observe that if $V$ satisfies the constraints of (2), then $M = X V^\top V$ satisfies the constraints of (3).

(5) (d) Consider the Singular Value Decomposition of $X$, defined as

$$X = \sum_{i=1}^{d} \sigma_i w_i z_i^\top,$$

where $w_i \in \mathbb{R}^n$ are the left singular vectors, and $z_i \in \mathbb{R}^d$ are the right singular vectors. Furthermore, let $\sigma_1 \geq \sigma_2, , , \geq \sigma_d \geq 0$. (such a decomposition always exists).

Show that

1. The top eigenvalues of $X^\top X$ are $\sigma_1^2, \sigma_2^2 \ldots \sigma_k^2$
2. The corresponding eigenvectors of $X^\top X$ are $z_1, z_2 \ldots z_k$.

(10) (e) The Eckart–Young–Mirsky theorem states that the minimizer of (3) is given by

$$M^* = \sum_{i=1}^{k} \sigma_i w_i z_i^\top.$$

Show that $M^* = XV^{*\top}V^*$, where $V^* \in \mathbb{R}^{k\times d}$ is the matrix whose $i^{th}$ row is the $i^{th}$ principal component ($i^{th}$ eigenvector of $X^\top X$).

(5) (f) In one sentence, explain why $V^*$ in (e) above must be the optimizer of (2).

(5) (g) In another sentence, explain why this implies that the optimal choice of $v_1 \ldots v_d$ in (1) is exactly the top-$k$ eigenvectors.

## Implementing Eigen-Faces

In this problem, we will explore the application of PCA to image classification. We will be using the Olivetti faces dataset. `https://scikit-learn.org/0.19/datasets/olivetti_faces.html`

This dataset contains 400 $64 \times 64$ gray-scale images. There are 40 distinct people, each with 10 images. Each image is represented as a vector $x_i \in \mathbb{R}^{4096}$, and the labels are given by $y_i \in \{0 \ldots 39\}$, indicating the ID of the person.

The starter code (including code for loading the data) has been provided for you in `hw5_starter_code.ipynb`. The training data (280 $(x,y)$ pairs) and test data (120 $(x,y)$ pairs) are provided in `olivetti_faces_data.pth`.

(10) 3. (a) Let $v_k$ denote the $k^{th}$ principle component. Compute $v_k$ for $k = 1 \ldots 120$, plot the percentage of variance explained by the top-$k$ eigenvectors (use training set only).

(5) (b) For each of $v_1 \ldots v_6$, reshape it into a 64x64 image and visualize the $v_i$'s in a row using `imshow`.

(5) (c) Create a 4x6 grid of subplots as follows: in the first row, visualize the first 6 $x_i$'s from the training set. In rows {1,2,3}, visualize the reconstructed $x_i$'s using the first $k = \{10, 50, 100\}$ principal components respectively.

(10) (d) Implement logistic regression on the full set of 4096 features. Print the test accuracy.

(20) (e) For any integer $k$, let $V_k \in \mathbb{R}^{k\times d}$ denote the matrix containing the top-k eigenvectors as rows. Let $z_{k,i} := V_k x_i$ denote the $k$-dimensional representation of $x_i$. For each of $k \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$, perform logistic regression over the training set $\{(z_{k,i}^{train}, y_{k,i}^{train})\}$ (you should have a separate set of logistic regression parameters for each $k$).

Evaluate your logistic regression models on the test set, and plot the test accuracy against $k$. On the same plot, include a horizontal line for the accuracy of the full-dimension logistic classifier from part (d).

(5) (f) How many principal components $k$ are needed to achieve over 90% test accuracy? What is this $k$ as a fraction of $d$?

(5) (g) Print your code.