LUMS
A Not-for-Profit University

Department of Electrical Engineering
School of Science and Engineering

EE514/CS535 Machine Learning

ASSIGNMENT 1

Due Date: 4:00 pm, Thursday, March 2, 2023.
Format: 9 problems, for a total of 100 marks
Instructions:

- You are allowed to collaborate with your peers but copying your colleague's solution is strictly prohibited. This is not a group assignment. Each student must submit his/her own assignment.
- Solve the assignment on blank A4 sheets and staple them before submitting.
- Submit in-class or in the dropbox labeled EE-514 outside the instructor's office.
- Write your name and roll no. on the first page.
- Feel free to contact the instructor or the teaching assistants if you have any concerns.

- You represent the most competent individuals in the country, do not let plagiarism come in between your learning. In case any instance of plagiarism is detected, the disciplinary case will be dealt with according to the university's rules and regulations.

Problem 1 (10 marks)

For a linear model $\mathbf{y} = \mathbf{X}\mathbf{w}$, we can find an estimate for the parameter vector $\hat{\mathbf{w}}$ given the data matrix $\mathbf{X}$ and the output vector $\mathbf{y}$ iteratively using gradient descent.
Given the matrices,

$$\mathbf{X} = \begin{bmatrix} ,\dots & \mathbf{x}_1^T & \dots \\ \dots & \mathbf{x}_2^T & \dots \\ & \vdots & \\ \dots & \mathbf{x}_n^T & \dots \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$$

where, $\mathbf{x}_i \in R^d$.
The objective function for Ridge Regression is given by,

$$\text{minimize} \quad f(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda ||\mathbf{w}||_2^2$$

where λ is the regularization parameter.

Prove that the update step of steepest gradient descent for Ridge Regression is,

$$\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{w}}_k(1 - 2\alpha\lambda) - 2\alpha X^T(X\hat{\mathbf{w}}_k - \mathbf{y}).$$

Problem 2 (10 marks)

For a linear model $\mathbf{y} = \mathbf{X}\mathbf{w}$, we can find the parameter vector $\mathbf{w}$ given the data matrix $\mathbf{X}$ and the output vector $\mathbf{y}$ by formulating the following optimization problem

$$\text{minimize} \quad f(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \,,$$

which minimizes the objective function $f(\mathbf{w})$. The solution can be obtained using ordinary least squares as

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Now we use a modified loss function, $g(\mathbf{w}) = f(\mathbf{w}) + f_r(\mathbf{w})$, where $f_r(\mathbf{w})$ is the regularizing term given by

$$f_r(\mathbf{w}) = \lambda\|\mathbf{D}\mathbf{w}\|_2^2$$

for any positive definite matrix $\mathbf{D}$. Show that the argument $\mathbf{w}$ that minimizes $f(\mathbf{w})$ is given by

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D}^T\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}.$$

Similar to the ridge regression that we discussed in class, this is referred to as weighted ridge regression. Discuss its interpretation.

Problem 3 (12 marks)

Given the data matrix formulated as

$$\mathbf{X} = \begin{bmatrix} \dots & \mathbf{x}_1^T & \dots \\ \dots & \mathbf{x}_2^T & \dots \\ & \vdots & \\ \dots & \mathbf{x}_n^T & \dots \end{bmatrix}$$

Where $\mathbf{x}_i \in R^d$. We assume that the data is **zero-mean** i.e., the mean of each feature is zero.

Low-Rank Approximation refers to approximating a data matrix with a matrix that is similar to the original but with the constraint that it has a lower rank. The mathematical formulation is given as,

$$\begin{aligned} \text{minimize} \quad & ||\mathbf{X} - \mathbf{Y}||_F \\ \text{such that} \quad & \text{Rank}(\mathbf{Y}) = r \end{aligned} \tag{1}$$

where $\mathbf{X}$ is the original data matrix and we are approximating it with the matrix $\mathbf{Y}$ that has a rank of r such that r is less than the rank of $\mathbf{X}$.
$||.||_F$ is the Frobenius norm and here it is used to quantify the dissimilarity between the matrices $\mathbf{X}$ and $\mathbf{Y}$.

In the class, we studied that Principal Component Analysis (PCA) gives an optimal low-rank approximation of data. In this question, we will prove this.

$\mathbf{X}$ can be decomposed using singular value decomposition (SVD) as $\mathbf{USV}^T$.

Using the SVD of $\mathbf{X}$, prove that Principal Component Analysis (PCA) gives the best low-rank approximation of the matrix. (Hint: Eckart-Young Theorem)

Problem 4 (10 marks)

Given the following matrix $\mathbf{X}$,

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 4 & 5 \\ 6 & 7 \\ 5 & 5 \\ 8 & 4 \\ 9 & 9 \\ 10 & 8 \end{bmatrix}$$

for $n = 7$ and $d = 2$.

We will make use of Principal Component Analysis (PCA) to reduce the dimensions of the matrix $\mathbf{X}$ from $d = 2$ to $d = 1$ by carrying out the following steps:

(a) Plot the data points on a 2-dimensional plane.

(b) Compute the principal components using the procedure taught in class (refer to the slides) and plot them as well.

(c) Now, project the original data matrix X onto its first principal component and plot on a 1-dimensional number line.

Problem 5 (20 marks)

In class, we discussed Principal Component Analysis (PCA) in detail for dimensionality reduction. Here we will discuss another method for dimensionality reduction, Linear Discriminant Analysis (LDA). First, we will find the solution to the LDA algorithm and then apply it to a 2D data set.

LDA tries to find a linear combination of features that achieves maximum separation for samples between classes and the minimum separation of samples within each class. Here we will assume only 2 classes, but this can easily be generalized to more classes. We will use LDA to project our data onto a line.

LDA achieves this by

1. Maximizing the distance between the mean of the two classes.
2. Minimizing the scatter (variation) within each class.

Mathematically, We want to find a projection vector $\mathbf{w}$ which we can use to obtain the one-dimensional approximation (projection) of each data-point $\mathbf{x}_i$ as $z_i = \mathbf{w}^T\mathbf{x}_i$, such that the following objective function is maximized

$$J(\mathbf{w}) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2},$$

where the numerator is the difference between the **projected class means**, and the denominator is the within class scatter of the **projected samples** defined as

$$\tilde{s_i}^2 = \sum_{z\in \text{Class}_i} (z - \tilde{\mu}_i)^2$$

Here $z = \mathbf{w}^T\mathbf{x}$ is the projected sample, and $\tilde{\mu}_i$ is the projected class mean for i-th class. In *simple words*, we want a projection such that samples of the same class are projected close to each other and the class means of the projected samples are far from each other.

(a) First, we will prove that the objective function formulated above can be expressed in terms of projection vector $\mathbf{w} \in \mathbf{R}^d$ as

$$J(W) = \frac{\mathbf{W}^T\mathbf{S_B}\mathbf{W}}{\mathbf{W}^T\mathbf{S_W}\mathbf{W}},$$

where

- $\mathbf{S_B}$ is the between-class scatter matrix of the samples in the original space
$$\mathbf{S_B} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$$
- $\mathbf{S_W} = \mathbf{S}_1 + \mathbf{S}_2$ is the within-class scatter matrix, where $\mathbf{S_i}$ is the covariance matrix of class i given by
$$\mathbf{S}_i = \sum_{\mathbf{x}\in \text{Class}_i} (\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T$$
- $\boldsymbol{\mu}_i$ denotes the mean of samples for i-th class.

(b) Show that $\mathbf{S_W}$ and $\mathbf{S_B}$ are symmetric and positive semi-definite.

(c) In part (a), we have the formulation of the objective function in terms of the projection vector $\mathbf{w}$. We want to determine $\mathbf{w}$ as a solution to the following optimization problem:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\text{maximize}} \quad J(\mathbf{w}),$$

Assuming that $\mathbf{S_W}$ is non-singular, show that the solution is the eigenvector of $\mathbf{S_W}^{-1}\mathbf{S_B}$ corresponding to the largest eigenvalue.

(d) Now we have a closed-form solution of LDA, we will implement it on a simple data set for a binary classification problem.

X_1	X_2	Label
1	1	0
2	2	0
3	4	0
3	6	0
8	8	1
7	10	1
10	6	1
8	7	1

i. Visualize the data.

ii. Project the data onto a line using LDA and visualize it again.
You can do the visualizations, the matrix multiplications and the eigenvalue decomposition using Matlab/Python. But you must implement LDA using the closed-form solution derived above, you can not use any libraries for it.

Problem 6 (10 marks)

Consider the following function

$$f(\mathbf{x}) = 0.5x_1^2 + x_2^2 + 2x_1 + x_2$$

(a) Compute Hessian of the matrix and show that the function is convex.

(b) Use analytical solution to find the value of $\mathbf{x}^*$ for which $f(\mathbf{x})$ is minimized.

(c) It is easy to analytically minimize a simple function like this, but it gets tougher as we go to higher dimensions and the function becomes complex, hence we use iterative optimization methods. Gradient descent is one of the most common such methods and its variants are widely used. In this part, you will run a few iterations of gradient descent and show that it converges to the solution you found in part b.

Start from the point $\mathbf{x}^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, and select a suitable learning rate.

Problem 7 (8 marks)

In this problem, we will use gradient descent algorithm for linear regression. Given a data matrix $\mathbf{X}$, a parameter vector $\mathbf{w}$, a bias term b, and the output vector $\mathbf{y}$. Following is a linear model:

$$\tilde{y} = \mathbf{w}^{\mathbf{T}}\mathbf{x} + b$$

Now given the following data matrix X, the output vector $\mathbf{y}$, and an initial estimates of the parameters $\mathbf{w_0}$.

$$\mathbf{X} = \begin{bmatrix} 1 & 3 \\ 1 & 4 \\ 1 & 28 \\ 1 & 19 \\ 1 & 12 \end{bmatrix}, \mathbf{w_0} = \begin{bmatrix} -0.8 \\ 0.6 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} 2 \\ 0.5 \\ 3 \\ 2 \\ 0.25 \end{bmatrix}$$

The column of 1 in the data matrix incorporates the bias term. Please note that you must use the least square loss function to compute the loss after each iteration. Take the value for the learning rate $\alpha = 0.001$.

(a) Run 2 iterations of gradient descent and find out the updated weight vector after each iteration. Also plot the data points and the line of best fit after each iteration.

(b) Why is it feasible to use a smaller value of learning rate / step size rather than using a larger value of it. Please give your reasoning to support the statement above.

Problem 8 (8 marks)

A software engineers research team has developed two classifiers, A and B, to predict whether developers in their firm use Open AI's newest invention of **Chat GPT**. They have collected a data set of 600 individuals, with a mix of positive and negative cases. They want to evaluate the performance of these classifiers on this data set and compare their results to determine which classifier is more effective in identifying whether a developer uses Chat GPT or not. The following table depicts the results they have obtained:

	Positive	**Negative**
Classifier A, predicted positive	202	100
Classifier A, predicted negative	104	194
Classifier B, predicted positive	199	97
Classifier B, predicted negative	114	190

(a) Which classifier is preferable in terms of **True Positive Rate (TPR)**?

(b) Which classifier is preferable in terms of **Accuracy**?

(c) Which classifier is preferable in terms of **F1 score**?

Show your working for all of the parts above.

Problem 9 (12 marks)

In class, we proved the upper bound on the performance of the 1 Nearest Neighbour (1-NN) Classifier for binary classification as $N \to \infty$. Here you will prove the general bound on performance for M classes.

$$R^* \leq R_{NN} \leq R^*(2 - \tfrac{MR^*}{M-1}),$$

where R^* is the Bayes error rate, and R_{NN} is the 1-NN error rate.

— End of Assignment —