ShanghaiTech University

CS182 Introduction to Machine Learning

Spring 2025

Final Exam

Instructors: Yujiao Shi

Time: June 17th 13:30-15:30

INSTRUCTIONS

Please read and follow the following instructions:

- You have 120 minutes to answer the questions.
- You are not allowed to bring any books or electronic devices, including regular calculators. The only allowed is two A4 pages of cheatsheets.
- You are not allowed to discuss or share anything with others during the exam.
- You should write the answer to every problem in the dedicated box **clearly**.
- You should write **your name and student ID** as indicated at the top of **each page** of the exam sheet.
- If you need more space, write "Continued on Page #" and continue your solution on the referenced scratch page at the end of the exam sheet.

Name	
Student ID	
Exam Classroom Number	
Seat Number	
(please copy this and sign)	All the work on this exam is my own.

THIS PAGE INTENTIONALLY LEFT BLANK.

DO NOT WRITE ANY ANSWER IN THIS PAGE!

1. (21 points) MISC: Choices

Each question **may have more than one** correct answer. Fill your answers **in the box below**.

Notice: Fulfill your answer or we may take it unspecified.

(a)	◯ A	◯ B	◯ C	◯ D	(b)	◯ A	◯ B	◯ C	◯ D	◯ E
(c)	◯ A	◯ B	◯ C	◯ D	(d)	◯ A	◯ B	◯ C	◯ D	◯ E
(e)	◯ A	◯ B	◯ C	◯ D	(f)	◯ A	◯ B	◯ C	◯ D	◯ E
(g)	◯ A	◯ B	◯ C	◯ D						

(a) (3') Based on what you have learned in class, **select all statement(s) which is(are) true**

A. k-NN algorithm does more computation on test time rather than train time.

B. With proper initialization, k-means can always find the global optimal clustering results.

C. k-NN works well when the dimensions of the instance features are relatively small, but struggles when the dimensions of the instance features are relatively large.

D. We can use the Manhattan distance in k-NN and k-means for continuous data.

Solution:

1. The training phase of the k-NN algorithm consists only of storing the feature vectors and class labels of the training samples. In the testing phase, a test point is classified by assigning the label which are most frequent among the k training samples nearest to that query point-hence higher computation.
2. k-means is not a globally optimal algorithm.
3. Curse of Dimensionality.
4. Obviously.

(b) (3') Based on what you have learned in class, **select all statement(s) which is(are) true**:

A. k-NN is widely used in tasks with ground truth.

B. k-NN is widely used in tasks without ground truth.

C. k-means is widely used in tasks with ground truth.

D. k-means is widely used in tasks without ground truth.

E. k-NN can be used in regression problems, while k-means can't be used in regression problems.

Solution:

1. k-NN is a classification model (Supervise), while k-means is a clustering model (Unsupervise).

2. We can also use k-NN for regression problems. In this case, the prediction can be based on the mean or the median of the k-most similar instances.

(c) (3') Based on what you have learned in class, **select all statement(s) which is(are) true**:

A. GMM offers probabilities for data points belonging to each cluser.

B. The EM algorithm optimizes a lower bound on its objective function, which is the marginal likelihood $\prod_i P(x_i)$ of the observed data points $x_1, x_2, \ldots x_N$.

C. When we use the EM algorithm to optimize a GMM, since each Gaussian distribution's parameters have a closed-form solution, we can obtain the final GMM without iteration, which significantly improves computational efficiency.

D. GMM is computationally efficient and interpretable, especially for moderate-dimensional data, compared to complex models like deep neural networks.

Solution: A. It is the advantage of GMM.
C. Obviously, we need to iterate many times until convergence, but we don't need to iteratively update parameters in M-step.
D. It is the advantage of GMM.

Choose the best answer for each of the following 2 questions based on the concepts of Support Vector Machines (SVM) and Maximum Margin Classifiers discussed in the materials.

(d) (3') The primary goal of a Maximum Margin Classifier for linearly separable data is to:

A. Maximize the number of correctly classified training points.

B. Minimize the training error by finding a separating hyperplane.

C. Maximize the distance between the decision boundary and the closest training points from both classes.

D. Minimize the sum of distances from all training points to the decision boundary.

E. None of the above.

Solution: (C) Maximize the distance between the decision boundary and the closest training points from both classes.

(e) (3') In the simplified SVM optimization problem: minimizing $\|\mathbf{w}\|_2^2$ subject to constraints, the margin of the classifier is:

A. Directly proportional to $\|\mathbf{w}\|_2$.

B. Inversely proportional to $\|\mathbf{w}\|_2$.

C. Equal to $\|\mathbf{w}\|_2^2$.

D. Unrelated to $\|\mathbf{w}\|_2$.

Solution: (B) Inversely proportional to $\|\mathbf{w}\|_2$. (Specifically, it's $\frac{1}{\|\mathbf{w}\|_2}$)

(f) (3') The probability density function of a continuous random scalar variable, x, distributed according to a distribution with parameter $\theta > 0$ is: $p(x \mid \theta) = 2\theta e^{-2\theta x}$, for $x \geq 0$. Assume that we observed $x_1, x_2, \ldots, x_n$ i.i.d. draws from a uniform distribution with unknown parameter θ. Letting $\bar{x}$ be the mean, and $\bar{\sigma}^2$ be the variance of the observed data, then the maximum likelihood estimator for θ is given by:

A. $2\bar{x}$

B. $\frac{1}{2\bar{x}}$

C. $\frac{\bar{\sigma}^2}{2\bar{x}}$

D. $2\bar{\sigma}^2\bar{x}$

E. None of the above

Solution: The log-likelihood function for the data is:

$$\begin{aligned} f(\theta) &= p(x_1, \ldots, x_n \mid \theta) \\ &= n\log(2\theta) - 2n\theta\bar{x} \\ f'(\theta) &= \frac{n}{\theta} - 2n\bar{x} \\ f''(\theta) &= -\frac{n}{\theta^2} < 0 \\ f'(\theta) = 0 \quad &\Rightarrow \quad \theta = \frac{1}{2\bar{x}} \end{aligned}$$

(g) (3') Which of the following statements correctly contrasts the ensemble learning methods?

A. Bagging aggregates predictions from models trained on bootstrap samples, thereby cutting variance while leaving the base-learner bias essentially unchanged.

B. Boosting is similar to Bagging, trains each base learner on a fresh bootstrap sample, and keeps the observation weights fixed from one iteration to the next.

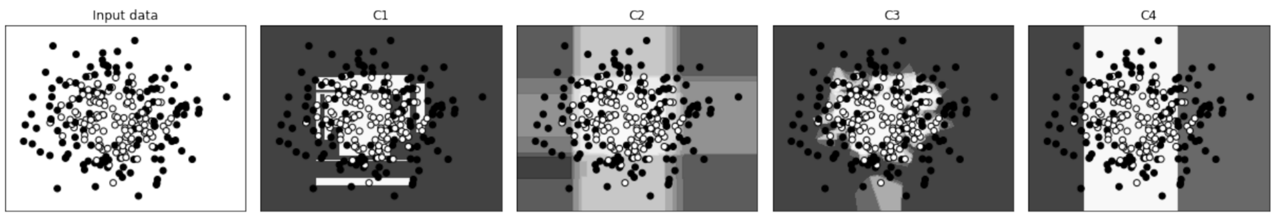
C. Random Forest trains its trees sequentially and re-weights the observations after each tree, so it is a Boosting algorithm.

D. Boosting can still achieve high accuracy when every individual learner is only marginally better than random guessing, whereas Bagging needs noticeably stronger base learners to deliver a substantial accuracy boost.

Solution: B is wrong because Boosting (e.g., AdaBoost) re-uses the same data but adjusts sample weights instead of resampling fresh bootstraps.
C is wrong because Random Forest grows its trees in parallel on bootstrap samples and does not re-weight observations—hence it is a Bagging, not a Boosting, method.

2. (12 points) Ensemble Methods

(a) (4') The following figure depicts a dataset for binary classification and four ensembles trained on the same training set. Match the classifier name to its corresponding plot (C_1, C_2, C_3, C_4)



AdaBoost with deep decision trees	◯ C_1	◯ C_2	◯ C_3	◯ C_4
Bagging shallow decision trees	◯ C_1	◯ C_2	◯ C_3	◯ C_4
Shallow decision tree	◯ C_1	◯ C_2	◯ C_3	◯ C_4
Bagging 1NN classifiers	◯ C_1	◯ C_2	◯ C_3	◯ C_4

Solution: AdaBoost with deep decision trees: C_1
Bagging shallow decision trees: C_2
Shallow decision tree: C_4
Bagging 1NN classifiers: C_3

(b) We will use the dataset below to learn a decision tree which predicts if people pass the course Introduction to Machine Learning (Yes(T) or No(F)), based on their previous GPA (High(H), Medium(M), or Low(L)) and whether or not they studied(Studied(T), not studied(F)).

GPA	Studied	Passed
L	F	F
L	T	T
M	F	F
M	T	T
H	F	T
H	T	T

For this problem, you can write your answers using $\log_2$, but it may be helpful to note that $\log_2 3 \approx 1.6$.

i. (2') What is the entropy $H(\text{Passed} \mid \text{GPA})$?

Solution:

$$\begin{aligned} H(\text{Passed} \mid \text{GPA}) &= \sum_x P(\text{GPA}=x) H(\text{Passed} \mid \text{GPA}=x) \\ &= -\frac{1}{3}\left(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}\right) - \frac{1}{3}\left(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}\right) - \frac{1}{3}(1\log_2 1) \\ &= \frac{2}{3} \approx 0.66 \text{ bit} \end{aligned}$$

ii. (2') What is the entropy $H(\text{Passed} \mid \text{Studied})$?

Solution:

$$\begin{aligned} H(\text{Passed} \mid \text{Studied}) &= \sum_x P(\text{Studied}=x) H(\text{Passed} \mid \text{Studied}=x) \\ &= -\frac{1}{2}\left(\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}\right) - \frac{1}{2}(1\log_2 1) \\ &= \frac{1}{2}\log_2 3 - \frac{1}{3} \approx 0.46 \text{ bit} \end{aligned}$$

iii. (4') Draw the full decision tree that would be learned for this dataset, and show the split criterion.

Solution:
Since

$$I(X;Y) = H(X) - H(X|Y)$$

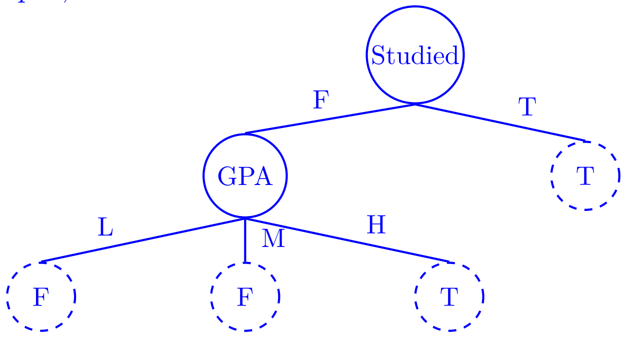
So

$$\begin{aligned} I(\text{Passed};\text{GPA}) &= H(\text{Passed}) - H(\text{Passed} \mid \text{GPA}) \\ I(\text{Passed};\text{Studied}) &= H(\text{Passed}) - H(\text{Passed} \mid \text{Studied}) \end{aligned}$$

Since $H(\text{Passed} \mid \text{GPA}) > H(\text{Passed} \mid \text{Studied})$, so

$$I(\text{Passed};\text{GPA}) < I(\text{Passed};\text{Studied})$$

So for the first split, we choose the 'Studied' to be the criterion, and for the second split, we have to use 'GPA' to be the criterion. And the decision tree is as follows:



3. (15 points) Kernel Perceptron Calculation

Kernel method is a specific important tool in machine learning. Now we want to use the *kernel perceptron* with the second-degree polynomial kernel

$$K(\mathbf{x}, \mathbf{z}) = \left(1 + \mathbf{x}^\top \mathbf{z}\right)^2, \quad \mathbf{x}, \mathbf{z} \in \mathbb{R}^2$$

on the dataset:

data number	$\mathbf{X_1}$	$\mathbf{X_2}$	$\mathbf{Y}$
1	-1	2	-1
2	-2	-2	$+1$
3	1	-1	-1
4	-3	1	$+1$

Hint: The kernel method used in Perceptron is similar to the inner product, where the data $\mathbf{x}$ is regarded as mapping into a higher-dimensional space $\phi(\mathbf{x})$.

In the original perceptron algorithm, for the initialization, we assume that the initial weights $\mathbf{w} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, with bias $b = 0$.

And use each data in ascending order of their indices $1 \to 4$ recycly until it convergence. Thus, we can define an n-dimensional vector $\boldsymbol{\alpha} = \mathbf{0}$ initially, where n is the number of data points, then for each time, when reaching the j-th data point, the predicted class is

$$\hat{y}_j = \mathbf{sign}\left(\mathbf{w}^\top \mathbf{x}_j + b\right), \text{ where } \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

Specifically, we define $\mathbf{sign}(0) = 1$. And the update rule is when misclassified, where α_j is the j-th component of $\boldsymbol{\alpha}$

$$\alpha_j \leftarrow \alpha_j + 1, \qquad b \leftarrow b + y_j$$

(a) (2') Show the update rule for the kernel perceptron.

Solution:

In the original perceptron, the predicted class is

$$\hat{y}_j = \mathbf{sign}\left(\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)^\top \mathbf{x}_j + b\right) = \mathbf{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^\top \mathbf{x}_j + b\right)$$

Thus, for the kernel perceptron, we have that the predicted class is

$$\hat{y}_j = \mathbf{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b\right)$$

And the update rule is when misclassified,

$$\alpha_j \leftarrow \alpha_j + 1, \qquad b \leftarrow b + y_j$$

(b) (2') After processing data 1, what is the coefficient vector $\boldsymbol{\alpha}$ and the bias b?

Solution:

$$\hat{y}_i = \textbf{sign}\,(0+0) = 1$$
$$\boldsymbol{\alpha} = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}^\top, b = -1$$

(c) (2') After processing data 2, what is the coefficient vector $\boldsymbol{\alpha}$ and the bias b?

Solution:

$$\hat{y}_i = \textbf{sign}\left(-1\cdot(1+(-2))^2 - 1\right) = -1$$
$$\boldsymbol{\alpha} = \begin{pmatrix}1 & 1 & 0 & 0\end{pmatrix}^\top, b = 0$$

(d) (2') After processing data 3, what is the coefficient vector $\boldsymbol{\alpha}$ and the bias b?

Solution:

$$\hat{y}_i = \textbf{sign}\left(-1\cdot(1+(-3))^2 + 1\cdot(1+0)^2 + 0\right) = -1$$
$$\boldsymbol{\alpha} = \begin{pmatrix}1 & 1 & 0 & 0\end{pmatrix}^\top, b = 0$$

(e) (1') Your friend claims that, for every case, once all n data are reached, neither $\boldsymbol{\alpha}$ nor b will change any further. Is the claim correct(Directly answer True or False)?

Solution: False

(f) (3') In the kernel perceptron, the weight vector in feature space can always be written as $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y^{(i)} \phi(\mathbf{x}^{(i)})$. Is this representation correct? Answer True or False, and provide proof for your selection.

Solution: True.

$$\begin{aligned}
\hat{y}_j &= \textbf{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b\right) \\
&= \textbf{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) + b\right) \\
&= \textbf{sign}\left(\left(\sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x}_i)\right)^\top \phi(\mathbf{x}_j) + b\right) \\
&= \textbf{sign}\left(\left(\sum_{i=1}^{n} \alpha_i y_i \phi(\mathbf{x}_i)\right)^\top \phi(\mathbf{x}_j) + b\right) \\
&= \textbf{sign}\left(\mathbf{w}^\top \phi(\mathbf{x}_j) + b\right)
\end{aligned}$$

(g) (3') Show that the given polynomial kernel is a valid kernel. You only need to consider the situation that $\mathbf{x}, \mathbf{z} \in \mathbb{R}^2$. Hint: Construct a mapping $\phi(\mathbf{x}) : \mathbb{R}^2 \mapsto \mathbb{R}^d$, where $d > 2$.

Solution:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \qquad \mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$$

$$K(\mathbf{x}, \mathbf{z}) = \left(1 + \mathbf{x}^\top \mathbf{z}\right)^2 = (1 + x_1 z_1 + x_2 z_2)^2$$

Define

$$\phi(\mathbf{x}) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ x_1^2 \\ \sqrt{2}x_1 x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^6$$

$$\begin{aligned} \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle &= 1 \cdot 1 + (\sqrt{2}x_1)(\sqrt{2}z_1) + (\sqrt{2}x_2)(\sqrt{2}z_2) \\ &\quad + (x_1^2)(z_1^2) + (\sqrt{2}x_1 x_2)(\sqrt{2}z_1 z_2) + (x_2^2)(z_2^2) \\ &= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + 2x_1 x_2 z_1 z_2 + x_2^2 z_2^2 \\ &= (1 + x_1 z_1 + x_2 z_2)^2 \\ &= K(\mathbf{x}, \mathbf{z}) \end{aligned}$$

So above all, we have proved that

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$$

Which means that the kernel is a valid kernel.

4. (8 points) KNN, Kmeans

(a) (4') We are given the location of each object and the type of object (triangle, circle, or square). The data is tabulated below. Try to find the object located in $(2, -1)$ that belongs to which type using a k-nearest neighbors estimator with $k = 3$, where the distance measure is computed by Euclidean distance.

Shape	Location		
Triangle	$(4, 5)$	$(4, 6)$	$(5, 4)$
Circle	$(-3, 2)$	$(-2, -1)$	
Square	$(8, 5)$	$(3, 4)$	$(0, 2)$

Solution:

$$\begin{aligned}
T : (4,5) :&\quad \sqrt{4+36} = \sqrt{40} \\
(4,6) :&\quad \sqrt{4+49} = \sqrt{53} \\
(5,4) :&\quad \sqrt{9+25} = \sqrt{34} \\
C : (-3,2) :&\quad \sqrt{25+9} = \sqrt{34} \\
(-2,-1) :&\quad \sqrt{16+0} = 4 \\
S : (8,5) :&\quad \sqrt{36+36} = \sqrt{72} \\
(3,4) :&\quad \sqrt{1+25} = \sqrt{26} \\
(0,2) :&\quad \sqrt{4+9} = \sqrt{13}
\end{aligned}$$

$k = 3$, Select 3 points that are closest to each other:

$$S : (3,4), (0,2)$$

$$C : (-2, 1)$$

So the object S is square.

(b) (4') Consider a Dataset $\mathbf{D}$ with 4 points as shown below. Perform a k-means method on this dataset with k as 2 using the Euclidean distance as the distance function. Initially, the 2 cluster centers are chosen as $\mathbf{c}_1 = (-4, -3), \mathbf{c}_2 = (1, 2)$. What are the final results after performing the k-means algorithm? Please provide the calculation process and the final cluster centers.

$$\mathbf{D} = \begin{bmatrix} -3 & -2 \\ 2 & 5 \\ 0 & 3 \\ 1 & -4 \end{bmatrix}$$

Solution:

First iteration:

$$(-3,-2) \text{ to } c_1 : d_1 = \sqrt{(-3+1)^2+(-2+3)^2} = \sqrt{4+1} = \sqrt{5}$$
$$\Rightarrow (-3,-2) \text{ belongs to cluster } C_1.$$
$$\text{to } c_2 : d_2 = \sqrt{(-3-1)^2+(-2-2)^2} = \sqrt{16+16} = \sqrt{32}.$$
$$(2,5) \text{ to } c_1 : d_1 = \sqrt{(2+1)^2+(5+3)^2} = \sqrt{9+64} = \sqrt{73}.$$
$$\text{to } c_2 : d_2 = \sqrt{(2-1)^2+(5-4)^2} = \sqrt{1+1} = \sqrt{2}$$
$$\Rightarrow (2,5) \text{ belongs to cluster } C_2.$$
$$(0,3) \text{ to } c_1 : d_1 = \sqrt{(0+1)^2+(3+3)^2} = \sqrt{1+36} = \sqrt{37}.$$
$$\text{to } c_2 : d_2 = \sqrt{(0-1)^2+(3-4)^2} = \sqrt{1+1} = \sqrt{2}$$
$$\Rightarrow (0,3) \text{ belongs to cluster } C_2.$$
$$(1,-4) \text{ to } c_1 : d_1 = \sqrt{(1+1)^2+(-4+3)^2} = \sqrt{4+1} = \sqrt{5}$$
$$\Rightarrow (1,-4) \text{ belongs to cluster } C_1.$$
$$\text{to } c_2 : d_2 = \sqrt{(1-1)^2+(-4-4)^2} = \sqrt{0+64} = 8.$$
$$\text{Update } c_1 = \left(\frac{-3+1}{2}, \frac{-2-4}{2}\right) = (-1,-3).$$
$$c_2 = \left(\frac{2+0}{2}, \frac{5+3}{2}\right) = (1,4).$$

Second iteration:

$$(-3,-2) \text{ to } c_1 : d_1 = \sqrt{(-3+1)^2+(-2+3)^2} = \sqrt{4+1} = \sqrt{5}$$
$$\Rightarrow (-3,-2) \text{ belongs to } C_1.$$
$$\text{to } c_2 : d_2 = \sqrt{(-3-1)^2+(-2-4)^2} = \sqrt{16+36} = \sqrt{52}.$$
$$(2,5) \text{ to } c_1 : d_1 = \sqrt{(2+1)^2+(5+3)^2} = \sqrt{9+64} = \sqrt{73}.$$
$$\text{to } c_2 : d_2 = \sqrt{(2-1)^2+(5-4)^2} = \sqrt{1+1} = \sqrt{2}$$
$$\Rightarrow (2,5) \text{ belongs to } C_2.$$
$$(0,3) \text{ to } c_1 : d_1 = \sqrt{(0+1)^2+(3+3)^2} = \sqrt{1+36} = \sqrt{37}.$$
$$\text{to } c_2 : d_2 = \sqrt{(0-1)^2+(3-4)^2} = \sqrt{1+1} = \sqrt{2}$$
$$\Rightarrow (0,3) \text{ belongs to } C_2.$$
$$(1,-4) \text{ to } c_1 : d_1 = \sqrt{(1+1)^2+(-4+3)^2} = \sqrt{4+1} = \sqrt{5}$$
$$\Rightarrow (1,-4) \text{ belongs to } C_1.$$
$$\text{to } c_2 : d_2 = \sqrt{(1-1)^2+(-4-4)^2} = \sqrt{0+64} = 8.$$

Since at iteration 2, the cluster centers have not changed. Then, k-means has converged. Cluster centers: $c_1 = (-1,-3), c_2 = (1,4)$.

5. (6 points) Maximum Margin Classifier Analysis

Consider a 2D binary classification problem with three training data points:

$$\mathbf{x}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, y_1 = +1, \quad \mathbf{x}_2 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}, y_2 = -1, \quad \mathbf{x}_3 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, y_3 = -1$$

A linear decision boundary is specified by the hyperplane

$$\mathbf{w} \cdot \mathbf{x} + \alpha = 0 \quad \text{with} \quad \mathbf{w} = \begin{pmatrix} -1 \\ 3 \end{pmatrix}, \alpha = -1$$

The simplified maximum-margin formulation minimises $\|\mathbf{w}\|^2$ subject to the constraints $y_i(\mathbf{w} \cdot \mathbf{x}_i + \alpha) \geq 1$ for all training examples.

(a) (2') For each training pair $(\mathbf{x}_i, y_i)$, compute the value $y_i(\mathbf{w} \cdot \mathbf{x}_i + \alpha)$ and verify whether the constraint is satisfied.

Solution:

$$\begin{aligned} y_1(\mathbf{w} \cdot \mathbf{x}_1 + \alpha) &= (+1)\big((-1) \cdot 1 + 3 \cdot 1 - 1\big) = 1, \\ y_2(\mathbf{w} \cdot \mathbf{x}_2 + \alpha) &= (-1)\big((-1) \cdot 2 + 3 \cdot 0 - 1\big) = 3, \\ y_3(\mathbf{w} \cdot \mathbf{x}_3 + \alpha) &= (-1)\big((-1) \cdot 0 + 3 \cdot 0 - 1\big) = 1 \end{aligned}$$

All three values are ≥ 1, so every training point satisfies the margin constraint.

(b) (2') Calculate the given classifier's margin.

Solution:

$$\|\mathbf{w}\| = \sqrt{(-1)^2 + 3^2} = \sqrt{10}, \qquad \gamma = \frac{1}{\sqrt{10}}.$$

(c) (2') Identify the Support Vectors of the given classifier in the given dataset.

Solution: Equality $(= 1)$ occurs for points $\mathbf{x}_1$ and $\mathbf{x}_3$. Hence, these two points are the Support Vectors; $\mathbf{x}_2$ lies strictly outside the margin.

6. (10 points) Logistic Regression

Assume that we have n i.i.d. data points $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, where each y_i is a binary label in $\{0, 1\}$. We model the posterior probability as a Bernoulli distribution and the probability for each class is the sigmoid function, i.e., $p(y \mid \mathbf{x}; \mathbf{w}) = q^y(1-q)^{1-y}$, where $q = s\left(\mathbf{w}^\top \mathbf{x}\right)$ and $s(\zeta) = \dfrac{1}{1+e^{-\zeta}}$ is the sigmoid function.

(a) (3') Show that for a given data point $\mathbf{x}$, the log ratio of the conditional probabilities, or log odds, is linear in $\mathbf{x}$. More specifically, show that

$$\log \frac{p(y=1 \mid \mathbf{x}; \mathbf{w})}{p(y=0 \mid \mathbf{x}; \mathbf{w})} = \mathbf{w}^\top \mathbf{x}$$

Solution:

$$\begin{aligned}
\log \frac{p(y=1 \mid \mathbf{x}; \mathbf{w})}{p(y=0 \mid \mathbf{x}; \mathbf{w})} &= \log \frac{q}{1-q} \\
&= \log \frac{\frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}}}}{\frac{e^{-\mathbf{w}^\top}}{1+e^{-\mathbf{w}^\top \mathbf{x}}}} \\
&= \log \frac{1}{e^{-\mathbf{w}^\top \mathbf{x}}} \\
&= \mathbf{w}^\top \mathbf{x}
\end{aligned}$$

(b) (3') For n data points, the likelihood is:

$$L(\mathbf{w}) = \prod_{i=1}^{n} p\left(y = y_i \mid \mathbf{x}_i\right) = \prod_{i=1}^{n} q_i^{y_i}\left(1-q_i\right)^{1-y_i}$$

Show that finding the maximum likelihood estimate of $\mathbf{w}$ is equivalent to the following optimization problem:

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left[\sum_{i=1}^{n}\left(\left(1-y_i\right)\mathbf{w}^\top \mathbf{x}_i + \log\left(1+\exp\left\{-\mathbf{w}^\top \mathbf{x}_i\right\}\right)\right)\right]$$

Hint: The next page has some empty space, you use it if you need.

Solution: Now, we step through minimizing the negative log likelihood of the training data as a function of the parameters $\mathbf{w}$:

$$\begin{aligned}
\hat{\mathbf{w}} &= \left[\underset{\mathbf{w}}{\operatorname{argmin}} - \sum_{i=1}^{n}\left(y_i\left(\log\left(q_i\right) + \left(1-y_i\right)\log\left(1-q_i\right)\right)\right)\right] \\
&= \left[\underset{\mathbf{w}}{\operatorname{argmin}} - \sum_{i=1}^{n} y_i\left(\log\left(\frac{q_i}{1-q_i}\right) + \log\left(1-q_i\right)\right)\right]
\end{aligned}$$

From the first question, we get that $y_i \log\left(\dfrac{q_i}{1-q_i}\right) = y_i \mathbf{w}^\top \mathbf{x}_i$. The second term of the sum can be simplified as follows:

$$\begin{aligned}\log(1-q_i) &= \log\left(\frac{1+\exp\left(-\mathbf{w}^\top\mathbf{x}_i\right)-1}{1+\exp\left(-\mathbf{w}^\top\mathbf{x}_i\right)}\right)\\ &= \log\left(\frac{\exp\left(-\mathbf{w}^\top\mathbf{x}_i\right)}{1+\exp\left(-\mathbf{w}^\top\mathbf{x}_i\right)}\right)\\ &= -\mathbf{w}^\top\mathbf{x}_i - \log\left(1+\exp\left(-\mathbf{w}^\top\mathbf{x}_i\right)\right)\end{aligned}$$

Plugging in, we get that

$$\begin{aligned}\hat{\mathbf{w}} &= \operatorname*{argmin}_{\mathbf{w}}\left[-\sum_{i=1}^{n} y_i\mathbf{w}^\top\mathbf{x}_i - \mathbf{w}^\top\mathbf{x}_i - \log\left(1+\exp\left\{-\mathbf{w}^\top\mathbf{x}_i\right\}\right)\right]\\ &= \operatorname*{argmin}_{\mathbf{w}}\left[\sum_{i=1}^{n}(1-y_i)\,\mathbf{w}^\top\mathbf{x}_i + \log\left(1+\exp\left\{-\mathbf{w}^\top\mathbf{x}_i\right\}\right)\right]\end{aligned}$$

(c) (4’) Comment on whether it is possible to find a closed-form maximum likelihood estimate of $\mathbf{w}$, if so, write down its closed form, otherwise, describe how to get the solution. And show that the solution is the optimal solution.

Solution: Let us denote $J(\mathbf{w}) = \sum_{i=1}^{n}(1-y_i)\,\mathbf{w}^\top\mathbf{x}_i + \log\left(1+\exp\left\{-\mathbf{w}^\top\mathbf{x}_i\right\}\right)$. Notice that $J(\mathbf{w})$ is convex in $\mathbf{w}$, so a global minimum can be found. Note that $s'(\zeta) = s(\zeta)(1-s(\zeta))$. Now let us take the gradient of $J(\mathbf{w})$ w.r.t $\mathbf{w}$:

$$\begin{aligned}\nabla_{\mathbf{w}}J(\mathbf{w}) &= \sum_{i=1}^{n}(1-y_i)\,\mathbf{x}_i - \frac{\exp\left\{-\mathbf{w}^\top\mathbf{x}_i\right\}}{1+\exp\left\{-\mathbf{w}^\top\mathbf{x}_i\right\}}\mathbf{x}_i\\ &= \sum_{i=1}^{n}\left(-1+s\left(\mathbf{w}^\top\mathbf{x}_i\right)-y_i+1\right)\mathbf{x}_i\\ &= \sum_{i=1}^{n}(s_i-y_i)\,\mathbf{x}_i\\ \nabla^2_{\mathbf{w}}J(\mathbf{w}) &= \sum_{i=1}^{n}s_i(1-s_i)\mathbf{x}_i\mathbf{x}_i^\top \succeq 0\end{aligned}$$

where, $s_i = s\left(\mathbf{w}^\top\mathbf{x}_i\right)$. Unfortunately, we can’t get a closed-form estimate for $\mathbf{w}$ by setting the derivative to zero, given that the term $\mathbf{s}$ still contains $\mathbf{w}$, and further-order derivatives will continue to carry expressions over $\mathbf{w}$. However, the convexity of this problem allows for first-order optimization algorithms, such as gradient descent, to converge to a global minimum:

$$\mathbf{w}_{k+1} \leftarrow \mathbf{w}_{k+1} - \alpha_k\nabla_{\mathbf{w}}J(\mathbf{w}_k)$$

For some suitable non-negative stepsize α_k.

7. (10 points) Neural Networks and Backpropagation

Consider a 2-layer neural network with input variables x_1 and x_2. The hidden layer computes:

$$z_1 = w_{11}^{(1)} x_1 + w_{12}^{(1)} x_2 + b_1^{(1)}$$
$$z_2 = w_{21}^{(1)} x_1 + w_{22}^{(1)} x_2 + b_2^{(1)}$$

followed by activation $h(\cdot)$. The output layer produces:

$$y_1 = w_{11}^{(2)} h(z_1) + w_{12}^{(2)} h(z_2) + b_1^{(2)}$$
$$y_2 = w_{21}^{(2)} h(z_1) + w_{22}^{(2)} h(z_2) + b_2^{(2)}$$

with loss function $\mathcal{L} = t_1 y_1 + t_2 y_2$.

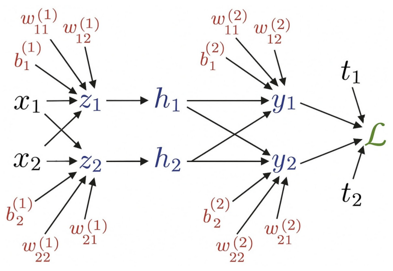


Figure 1: The Framework of Network.

To train this model, we first need to follow the forward process to compute its loss. Then, based on the loss, we compute the gradient of the loss to optimize the model parameters.

(a) (2') Compute the loss: Express $\mathcal{L}$ in terms of x_1, x_2 and the network parameters. (Hint: First express $\mathcal{L}$ using $h(z_i)$ and the parameters, group the coefficients of $h(z_i)$ separately, then substitute the expressions for z_i.)

Solution: According to the expressions of z_i, y_i and $\mathcal{L}$:

$$\begin{aligned}
\mathcal{L} &= t_1 y_1 + t_2 y_2 \\
&= t_1(w_{11}^{(2)} h(z_1) + w_{12}^{(2)} h(z_2) + b_1^{(2)}) + t_2(w_{21}^{(2)} h(z_1) + w_{22}^{(2)} h(z_2) + b_2^{(2)}) \\
\textbf{[+1]} &= (t_1 w_{11}^{(2)} + t_2 w_{21}^{(2)}) h(z_1) + (t_1 w_{12}^{(2)} + t_2 w_{22}^{(2)}) h(z_2) + t_1 b_1^{(2)} + t_2 b_2^{(2)} \\
\textbf{[+1]} &= (t_1 w_{11}^{(2)} + t_2 w_{21}^{(2)}) h\left[w_{11}^{(1)} x_1 + w_{12}^{(1)} x_2 + b_1^{(1)}\right] + (t_1 w_{12}^{(2)} + t_2 w_{22}^{(2)}) h\left[w_{21}^{(1)} x_1 + w_{22}^{(1)} x_2 + b_2^{(1)}\right] \\
&\quad + t_1 b_1^{(2)} + t_2 b_2^{(2)}
\end{aligned}$$

(b) Compute the gradient: After performing the forward pass, we obtain the hidden layer activations $h_1 = h(z_1)$ and $h_2 = h(z_2)$, as well as the output values y_1 and y_2. Given sigmoid activation $h(z) = \dfrac{1}{1+e^{-z}}$, your tasks are:

i. (2') Express $\dfrac{\partial h_1}{\partial z_1}$ using only h_1.

Solution: Given sigmoid activation $h(z) = \dfrac{1}{1+e^{-z}}$ and forward pass values $h_1 = h(z_1)$, $h_2 = h(z_2)$:

$$\begin{aligned}\frac{\partial h_1}{\partial z_1} &= \frac{e^{-z_1}}{(1+e^{-z_1})^2} \\ &= \frac{1+e^{-z_1}-1}{(1+e^{-z_1})^2} \\ &= \frac{1}{1+e^{-z_1}} - \frac{1}{(1+e^{-z_1})^2} \\ \textbf{[+2]} &= h_1(1-h_1)\end{aligned}$$

ii. (3') Compute $\dfrac{\partial \mathcal{L}}{\partial w_{11}^{(2)}}$ using h_1, t_1 and parameters.

Solution: Using the chain rule:

$$\textbf{[+2]}\frac{\partial \mathcal{L}}{\partial w_{11}^{(2)}} = \frac{\partial \mathcal{L}}{\partial y_1} \cdot \frac{\partial y_1}{\partial w_{11}^{(2)}} = t_1 \cdot h_1$$

iii. (3') Compute $\dfrac{\partial \mathcal{L}}{\partial w_{11}^{(1)}}$ using x_1, h_1, t_1 and parameters.

Solution: Using the chain rule:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial w_{11}^{(1)}} &= \frac{\partial \mathcal{L}}{\partial h_1} \cdot \frac{\partial h_1}{\partial z_1} \cdot \frac{\partial z_1}{\partial w_{11}^{(2)}} \\ \textbf{[+4]} &= (t_1 w_{11}^{(2)} + t_2 w_{21}^{(2)}) \cdot h_1(1-h_1) \cdot x_1\end{aligned}$$

8. (10 points) Performing PCA by Hand

Given the following dataset in 3D space $\mathbb{R}^3$:

$$\left\{ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 4 \\ 1 \end{bmatrix} \right\}$$

(a) (2') Compute the mean vector μ of the dataset. Then, obtain the centered data matrix $\dot{X}$. (Hint: $\dot{X}$ should be a 4×3 matrix.)

Solution: [+1] The sample mean is that:

$$\boldsymbol{\mu} = \frac{1}{4}\left(\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} + \begin{bmatrix} 3 \\ 4 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} 2 \\ 3 \\ 2 \end{bmatrix}$$

[+1] After subtracting the mean from each sample, we form the centered design matrix:

$$\dot{X} = X - \mu = \begin{bmatrix} 1-2 & 2-3 & 3-2 \\ 3-2 & 2-3 & 1-2 \\ 1-2 & 4-3 & 3-2 \\ 3-2 & 4-3 & 1-2 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & 1 & -1 \end{bmatrix}$$

(b) (5') Find all principal components of this sample. Write them as unit vectors. (Hint: Use SVD. The characteristic polynomial of this symmetric matrix is $\det\left(\lambda I - \dot{X}^\top \dot{X}\right)$.)

Solution: [+1] We can calculate that:

$$S = \dot{X}^\top \dot{X} = \begin{bmatrix} -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 4 & 0 & -4 \\ 0 & 4 & 0 \\ -4 & 0 & 4 \end{bmatrix}$$

The characteristic polynomial of this symmetric matrix is:

$$\det(S - \lambda I) = \det \begin{bmatrix} 4-\lambda & 0 & -4 \\ 0 & 4-\lambda & 0 \\ -4 & 0 & 4-\lambda \end{bmatrix} = (\lambda - 8)(\lambda - 4)\lambda = 0$$

[+1] So the eigenvalues of $\dot{X}^\top \dot{X}$ are $\lambda_1 = 8, \lambda_2 = 4, \lambda_3 = 0$

[+1] For $\lambda = 8$, we have the first corresponding eigenvector is that $\mathbf{v}_1 = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

[+1] For $\lambda = 4$, we have the second corresponding eigenvector is that $\mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

[+1] For $\lambda = 0$, we have the third corresponding eigenvector is that $\mathbf{v}_3 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

(c) (1') Suppose we add a new point $\mathbf{x}_{\text{new}} = \begin{bmatrix} 2 \\ 3 \\ 2 \end{bmatrix}$ Does this change principal components? Please explain briefly.

Solution: [+1] Does not change the centered data (since $2-2=0$, $3-3=0$, $2-2=0$). Therefore, **covariance matrix S remains unchanged**. Thus, there will be **no change** to principal components.

(d) (2') Suppose instead we add a point along the direction of the first principal component (PC1)(e.g., $\mathbf{x}_{\text{new}} = \begin{bmatrix} 102 \\ 3 \\ -98 \end{bmatrix}$), how does this affect PC1? Make a guess and explain briefly.

Solution: [+1] Adding a point in the direction of the first principal component (PC1) will **not change PC1**.

If we add a new data point of the form: $\mathbf{x}_{\text{new}} = \mu + \lambda PC1$ (where μ is the mean of the original data, and PC1 is the first principal component), then this new point lies exactly along the direction of the current PC1.

[+1] Adding a point in the direction of PC1 will increase the variance along that direction the most. PCA finds the direction of maximum variance through the origin (after mean-centering). Therefore, the direction of PC1 does not change—it is only reinforced. The eigenvalue (variance) along PC1 increases, but the eigenvector (direction) stays the same.

9. (8 points) Matrix Factorization

In Recommendation Systems, Matrix Factorization is a common technique within Latent Factor Methods to approximate a partially observed User-Item rating matrix $R \in \mathbb{R}^{m \times n}$ by the product of two low-rank matrices $U \in \mathbb{R}^{m \times k}$ (user factors) and $V \in \mathbb{R}^{n \times k}$ (item factors), such that $R \approx UV^\top$. The objective is to learn the matrices U and V by minimizing an objective function, typically the sum of squared errors on known ratings with regularization. The objective function for a set of known ratings Z with L2 regularization is given by:

$$J(U,V) = \frac{1}{2} \sum_{(i,j) \in Z} \left(R_{i,j} - \left(UV^\top \right)_{i,j} \right)^2 + \frac{\lambda}{2} \left(\|U\|_F^2 + \|V\|_F^2 \right)$$

where the predicted rating $\hat{R}_{i,j} = \left(UV^\top\right)_{i,j} = \sum_{l=1}^{k} U_{i,l} V_{j,l}$, $\|U\|_F^2 = \sum_{i=1}^{m} \sum_{l=1}^{k} U_{i,l}^2$, $\|V\|_F^2 = \sum_{j=1}^{n} \sum_{l=1}^{k} V_{j,l}^2$, and $\lambda > 0$ is the regularization parameter. Stochastic Gradient Descent (SGD) is a common method to optimize this objective.

Consider a scenario with $m = 2$ users, $n = 3$ items, and $k = 2$ latent factors. The known ratings are $R_{1,1} = 4$ and $R_{2,2} = 5$. The initial latent factor matrices U and V are given as:

$$U = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix}, \quad V = \begin{pmatrix} 0 & 1 \\ 2 & 0 \\ 1 & 1 \end{pmatrix}$$

The learning rate is $\eta = 0.01$ and the regularization parameter is $\lambda = 0.1$.

Perform one step of Stochastic Gradient Descent using the known rating $R_{1,1} = 4$ (User 1, Item 1). Let $U_{1,\cdot}$ be the first row of U and $V_{1,\cdot}$ be the first row of V.

(a) (1') Calculate the predicted rating $\hat{R}_{1,1}$ based on the current U and V.

Solution:

$$\hat{R}_{1,1} = U_{1,\cdot} \cdot V_{1,\cdot}^\top = \begin{pmatrix} 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = (1)(0) + (2)(1) = 0 + 2 = 2$$

(b) (1') Calculate the error $E_{1,1} = R_{1,1} - \hat{R}_{1,1}$.

Solution:

$$E_{1,1} = R_{1,1} - \hat{R}_{1,1} = 4 - 2 = 2$$

(c) (3') Calculate the gradients of the loss function for this single rating $(1,1)$ (including regularization) with respect to $U_{1,\cdot}$ and $V_{1,\cdot}$.

Solution:

$$\nabla_{U_{1,\cdot}} J_{1,1} = -E_{1,1} V_{1,\cdot} + \lambda U_{1,\cdot} = -2 \begin{pmatrix} 0 & 1 \end{pmatrix} + 0.1 \begin{pmatrix} 1 & 2 \end{pmatrix} = \begin{pmatrix} 0.1 & -1.8 \end{pmatrix}$$

$$\nabla_{V_{1,\cdot}} J_{1,1} = -E_{1,1} U_{1,\cdot} + \lambda V_{1,\cdot} = -2 \begin{pmatrix} 1 & 2 \end{pmatrix} + 0.1 \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} -2 & -3.9 \end{pmatrix}$$

(d) (2') Update the user vector $U_{1,\cdot}$ and item vector $V_{1,\cdot}$ using the calculated gradients and the learning rate η.

Solution:

$$U_{1,\cdot}^{\text{new}} = U_{1,\cdot}^{\text{old}} - \eta \nabla_{U_{1,\cdot}} J_{1,1} = \begin{pmatrix} 1 & 2 \end{pmatrix} - 0.01 \begin{pmatrix} 0.1 & -1.8 \end{pmatrix} = \begin{pmatrix} 0.999 & 2.018 \end{pmatrix}$$
$$V_{1,\cdot}^{\text{new}} = V_{1,\cdot}^{\text{old}} - \eta \nabla_{V_{1,\cdot}} J_{1,1} = \begin{pmatrix} 0 & 1 \end{pmatrix} - 0.01 \begin{pmatrix} -2 & -3.9 \end{pmatrix} = \begin{pmatrix} 0.02 & 1.039 \end{pmatrix}$$

(e) (1') Write down the updated matrices U and V after this one SGD step.

Solution: Only the first row of U and the first row of V are updated. The other rows remain unchanged in this SGD step, focusing only on rating (1,1). Updated user matrix U:

$$U^{\text{new}} = \begin{pmatrix} 0.999 & 2.018 \\ 3 & 1 \end{pmatrix}$$

Updated item matrix V:

$$V^{\text{new}} = \begin{pmatrix} 0.02 & 1.039 \\ 2 & 0 \\ 1 & 1 \end{pmatrix}$$

THIS PAGE INTENTIONALLY LEFT BLANK.

Label it clearly when you need to continue your solution on the scratch page.

THIS PAGE INTENTIONALLY LEFT BLANK.

THIS PAGE INTENTIONALLY LEFT BLANK.

THIS PAGE INTENTIONALLY LEFT BLANK.

THIS PAGE INTENTIONALLY LEFT BLANK.

THIS PAGE INTENTIONALLY LEFT BLANK.