Indian Statistical Institute

Mid Semestral Examination: Machine Learning I

Maximum Marks: 60, Time: 2 Hrs. 15 Mins.

February 27, 2025

Answer ALL questions. The maximum score you can obtain is 60.

1. Consider a linear machine with a discriminant function $g(\mathbf{x}) = \mathbf{w^T x} + w_0$ for points from two classes.

   (a) Show that the perpendicular (Euclidean) distance from the point $\mathbf{x}_a$ to the hyperplane is given by:
   $$\frac{|g(\mathbf{x}_a)|}{\|\mathbf{w}\|}.$$

   (b) Show that the projection of $\mathbf{x}_a$ onto the hyperplane is given by:
   $$\mathbf{x}_p = \mathbf{x}_a - \frac{g(\mathbf{x}_a)}{\|\mathbf{w}\|^2}\mathbf{w}.$$

   $[7 + 3 = 10]$

2. We will perform Principal Component Analysis (PCA) on the following six sample points in $\mathbb{R}^2$, given as column vectors:
   $$X_1 = \begin{bmatrix}0\\0\end{bmatrix}, \quad X_2 = \begin{bmatrix}0\\1\end{bmatrix}, \quad X_3 = \begin{bmatrix}1\\0\end{bmatrix}, \quad X_4 = \begin{bmatrix}1\\2\end{bmatrix}, \quad X_5 = \begin{bmatrix}2\\1\end{bmatrix}, \quad X_6 = \begin{bmatrix}2\\2\end{bmatrix}$$

   Answer the following:

   (a) Compute the mean of the sample points and construct the centered design matrix $\dot{X}$.

   (b) Find the principal components of the sample by computing the eigenvectors of the covariance matrix and express them as unit vectors.

   (c) Project each of the original sample points (not the centered points) onto the first principal component and give the projected points in $\mathbb{R}^2$.

   $[4 + 7 + 4 = 15]$

3. Consider the ridge regression problem, which seeks to minimize the regularized least squares objective:
   $$J(w) = \frac{1}{2}\sum_{i=1}^{n}(y_i - x_i^T w)^2 + \frac{\lambda}{2}\|w\|^2$$

   where:

- $X \in \mathbb{R}^{n \times d}$ is the design matrix containing feature vectors $x_i \in \mathbb{R}^d$,
- $y \in \mathbb{R}^n$ is the vector of responses,
- $w \in \mathbb{R}^d$ is the weight vector to be estimated,
- $\lambda > 0$ is the regularization parameter.

(a) Derive the gradient $\nabla J(w)$ and Hessian $H = \nabla^2 J(w)$ of the ridge regression objective function.

(b) Derive the explicit update rule of Newton's method in terms of X, y, and λ.

(c) Prove that Newton's method converges in a single iteration if the initial guess $w^{(0)}$ is chosen optimally. Explain why this happens in the case of ridge regression.

[3 + 3 + 4 = 10]

4. Consider m training examples $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, where $x \in \mathcal{X}$ and $y \in \{-1, 1\}$. Suppose we have a weak learning algorithm A which produces a hypothesis $h : \mathcal{X} \to \{-1, 1\}$ given any distribution D of examples. AdaBoost works as follows:

- Begin with a uniform distribution $D_1(i) = \frac{1}{m}$, for $i = 1, \ldots, m$.
- At each round $t = 1, \ldots, T$:
  - Run A on D_t and obtain h_t.
  - Update the distribution:

$$D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

where Z_t is the normalizer, and $i = 1, \ldots, m$.

Since A is a weak learning algorithm, the produced hypothesis h_t at round t is only slightly better than random guessing, say by a margin γ_t:

$$\epsilon_t = \mathrm{err}_{D_t}(h_t) = \Pr_{x \sim D_t}[y \neq h_t(x)] = \frac{1}{2} - \gamma_t.$$

In the end, AdaBoost outputs the final hypothesis:

$$H = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t h_t\right).$$

(a) Imagine there is an adversary who wants to fool h_t in the next round by adjusting the distribution. More formally, given h_t, the adversary wants to set D_{t+1} such that:

$$\mathrm{err}_{D_{t+1}}(h_t) = \frac{1}{2}.$$

Show that the particular choice of

$$\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$$

achieves this goal.

(b) Show that the updated weight distribution follows:

$$D_{T+1}(i) = \left(m \cdot \prod_{t=1}^{T} Z_t \right)^{-1} e^{-y_i f(x_i)},$$

where:

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x).$$

(c) Prove the following bound on the training error of the final hypothesis:

$$\mathrm{err}_S(H) \leq \prod_{t=1}^{T} Z_t.$$

(d) Show that:

$$\prod_{t=1}^{T} Z_t \leq e^{-2\sum_{t=1}^{T} \gamma_t^2}.$$

[4 + 4 + 5 + 3 = 16]

5. Recall the setup of logistic regression: we assume that the posterior probability is of the form

$$p(Y = 1 \mid X) = \frac{1}{1 + e^{-x^T \beta}}.$$

This makes Y a Bernoulli random variable.

We now turn to the case where Y is a multinomial random variable over K outcomes. This is called *softmax regression*, because the posterior probability is given by the softmax function:

$$p(Y = k \mid X) = \mu_k(X) = \frac{e^{x^T \beta_k}}{\sum_{j=1}^{K} e^{x^T \beta_j}}.$$

Assume we have observed data

$$D = \{(x_i, y_i)\}_{i=1}^{N}.$$

Our goal is to learn the weight vectors $\beta_1, \ldots, \beta_K$.

(a) Find the negative log-likelihood of the data $\{(x_i, y_i)\}$, which we denote by

$$l(\beta_1, \ldots, \beta_K).$$

(b) We want to minimize this negative log-likelihood. To combat overfitting, we put a regularizer on the objective function. Specifically, consider

$$l(\beta_1, \ldots, \beta_K) + \lambda \sum_{k=1}^{K} \|\beta_k\|^2.$$

Find the gradient of this regularized objective with respect to each β_k.

(c) State the gradient updates for batch gradient descent.

[3 + 4 + 2 = 9]

6. Consider a classification problem where the impurity of a node S in a decision tree is measured using the Gini index, defined as

$$\text{Gini}(S) = 1 - \sum_{k=1}^{K} p_k^2,$$

where p_k is the proportion of class k in S and there are K classes.

Suppose the dataset S is split into two subsets S_L and S_R by a binary split. Let

$$\alpha = \frac{|S_L|}{|S|} \quad \text{and} \quad 1 - \alpha = \frac{|S_R|}{|S|}$$

denote the fractions of samples in the left and right nodes, respectively. The Gini indices for the subsets are given by

$$\text{Gini}(S_L) = 1 - \sum_{k=1}^{K} p_{k,L}^2, \quad \text{and} \quad \text{Gini}(S_R) = 1 - \sum_{k=1}^{K} p_{k,R}^2,$$

where $p_{k,L}$ and $p_{k,R}$ are the class proportions in S_L and S_R, respectively.

(a) Reduction in impurity (Gini gain) due to the split is measured as:
$\Delta\text{Gini} = \text{Gini}(S) - [\alpha\,\text{Gini}(S_L) + (1-\alpha)\,\text{Gini}(S_R)]$. Prove that the reduction in impurity (Gini gain) due to the split:

$$\Delta\text{Gini} = \alpha \sum_{k=1}^{K} p_{k,L}^2 + (1-\alpha) \sum_{k=1}^{K} p_{k,R}^2 - \sum_{k=1}^{K} p_k^2.$$

(b) Prove that $\Delta\text{Gini} \geq 0$ for any valid split. Under what conditions does $\Delta\text{Gini} = 0$?

(c) For a binary classification problem ($K = 2$) with class proportions p and $1 - p$ in node S, show that

$$\text{Gini}(S) = 2p(1-p).$$

Determine the value of p that maximizes the impurity $\text{Gini}(S)$ and compute the maximum impurity.

$[4 + 4 + 4 = 12]$

End Semestral Examination: Machine Learning 1

M. Tech. (CS) 2024 - 26

Indian Statistical Institute

Maximum Marks: 100, Time: 3 Hrs. 15 Mins.

May 09, 2025

You may attempt ALL the questions. The maximum score you can obtain is 100.

1. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, be a dataset that is linearly separable. Consider the classical perceptron algorithm, which processes one example at a time in a sequential pass over the dataset. The weight vector $\mathbf{w}$ is updated using the following rule:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + y_i \mathbf{x}_i \quad \text{if } y_i(\mathbf{w}^{(t)} \cdot \mathbf{x}_i) \leq 0,$$

where i is the index of the current data point being examined and t denotes the update step (not necessarily the epoch). Now suppose that we modify the update condition to enforce a margin threshold $\gamma > 0$, such that the perceptron updates only if:

$$y_i(\mathbf{w}^{(t)} \cdot \mathbf{x}_i) \leq \gamma.$$

   (a) Show that if the dataset is linearly separable with margin $\gamma^* > \gamma$, then the modified algorithm converges in a finite number of steps. Derive an upper bound on the number of updates in terms of γ, γ^*, and the maximum norm of the inputs.

   (b) Compare the convergence behavior of this margin-based perceptron to the conventional perceptron algorithm. Under what conditions could the modified perceptron converge faster or slower?

   (c) Now consider a variation of the update rule where the step size decays with time:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta_t y_i \mathbf{x}_i, \quad \text{where } \eta_t = \frac{1}{t}.$$

   Does this version of the algorithm still converge for linearly separable data? Provide a justification or a counterexample.

$[8 + 6 + 6 = 20]$

2. Consider a convolutional neural network (CNN) layer where an input image of size $H \times W$ is passed through a convolutional layer with the following parameters:

   Filter (kernel) size: $K \times K$, Stride: S, Padding: P, Number of filters: F. Assume the input image has C channels (depth), and the convolution is applied across the spatial dimensions (height and width) with padding applied symmetrically on all sides.

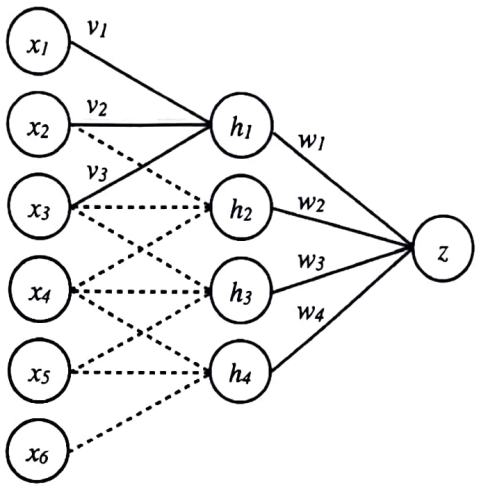
   (a) Given an input of size $32 \times 32 \times 3$, a convolutional layer with $F = 16$ filters of size 5×5, stride $S = 1$, and padding $P = 2$, compute the size of the output feature map.

(b) Suppose you want to downsample the feature map by a factor of 2 without using pooling. What stride should you use to achieve this, keeping the filter size the same?

(c) If you stack two such convolutional layers, each with padding $P = 2$, stride $S = 1$, and filter size $K = 5$, what is the overall spatial size of the output relative to the input?

[3 + 3 + 4 = 10]

3. Consider the following simple 1D convolutional neural network architecture:



- The input is a one-dimensional vector $x = (x_1, x_2, \ldots, x_6) \in \mathbb{R}^6$.
- The first layer is a one-dimensional convolutional layer with a **single filter of size 3**, with weights v_1, v_2, v_3, and **no bias**. The convolution produces 4 outputs:

$$h_i = s\left(\sum_{j=1}^{3} v_j x_{i+j-1}\right), \quad \text{for } i = 1, 2, 3, 4,$$

  where $s(x) = \frac{1}{1+e^{-x}}$ is the sigmoid activation function.
- The second layer is a **fully connected layer** with weights w_1, w_2, w_3, w_4, producing a scalar output:

$$z = \sum_{i=1}^{4} w_i h_i.$$

- The loss function is the squared error:

$$R = (y - z)^2,$$

  where y is the scalar label corresponding to input x.

Note that there are no bias terms in either layer. The output unit is linear (i.e., no activation function is applied after the second layer).

(a) How many trainable parameters does the network have in total?

(b) Derive an expression for $\frac{\partial R}{\partial w_i}$ for a given $i \in \{1, 2, 3, 4\}$.

(c) Express the gradient $\frac{\partial R}{\partial w}$ in vectorized form.

(d) Derive an expression for $\frac{\partial R}{\partial v_j}$ for $j \in \{1, 2, 3\}$. That is, compute how the loss depends on the convolutional filter weights via the backpropagation chain.

$[2 + 5 + 2 + 6 = 15]$

4. Given the gradient calculated at a point, the Adam optimizer has three distinct steps. First, update the moving averages. Second, apply the bias correction. Third, update the parameters. Consider the moving average of the square of the gradients. It is given by the recursive formula:

$$S_t = \beta_2 S_{t-1} + (1 - \beta_2) g_t^2.$$

(a) Derive the expression for S_t only in terms of the gradients $g_0, g_1, \ldots\ldots, g_t$

(b) Given your expression in part (a), what is $E[S_t]$ in terms of $E[g_t^2]$ and β_2? You may assume that g_i's are independent and identically distributed.

$[4 + 6 = 10]$

5. Recall the loss function for k-means clustering with k clusters, sample points $x_1, \ldots, x_n$, and centers $\mu_1, \ldots, \mu_k$:

$$L = \sum_{j=1}^{k} \sum_{i \in S_j} \|x_i - \mu_j\|^2,$$

where S_j refers to the set of data points that are closer to μ_j than to any other cluster mean.

(a) Instead of updating μ_j by computing the mean, let's minimize L with **batch gradient descent** while holding the sets S_j fixed. Derive the update formula for μ_1 with learning rate (step size) ϵ.

(b) Derive the update formula for μ_1 with **stochastic gradient descent** on a single sample point x_i. Use learning rate ϵ.

(c) Recall that in the update step of the standard algorithm, we assign each cluster center to be the mean (centroid) of the data points closest to that center. It turns out that a particular choice of the learning rate ϵ (which may be different for each cluster) makes the two algorithms (batch gradient descent and the standard k-means algorithm) have identical update steps. Let's focus on the update for the first cluster, with center μ_1. Calculate the value of ϵ so that both algorithms perform the same update for μ_1.

$[3 + 3 + 4 = 10]$

6. You are given a dataset $\{(x_i, y_i)\}_{i=1}^4$, where $x_i \in \mathbb{R}$, and your goal is to fit a model to predict y from x using **gradient boosting** with squared error loss. You are given the following training data:

$$(x_1, y_1) = (1, 4), \quad (x_2, y_2) = (2, 5), \quad (x_3, y_3) = (3, 8), \quad (x_4, y_4) = (4, 10)$$

and the learning rate is $\eta = 0.5$. At iteration $m = 1$, the base learner $h_1(x)$ is a decision stump that splits the input at $x = 2.5$, assigning a constant prediction to each side of the split.

(a) Compute the initial prediction $F_0(x)$.

(b) Compute the residuals at iteration 1, and fit the decision stump $h_1(x)$ based on the split at $x = 2.5$.

(c) Write the expression for $F_1(x) = F_0(x) + \eta h_1(x)$, and compute $F_1(x_i)$ for all $i = 1, 2, 3, 4$.

(d) Compute the updated residuals at iteration 2.

(e) Draw the approximate regression function realized by $F_1(x)$ and also place the training points in the same figure.

$[2+3+4+3+3=15]$

7. Consider the data $\{\binom{1}{1}, \binom{3}{3}, \binom{2}{*}\}$, sampled from a two dimensional(seperable) distribution $p(x_1, x_2) = p(x_1)p(x_2)$, with

$$p(x_1) = \begin{cases} \frac{1}{\theta_1} \exp\left(-\frac{x_1}{\theta_1}\right) & \text{if } x_1 \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

and

$$p(x_2) \sim U(0, \theta_2) = \begin{cases} \frac{1}{\theta_2} & \text{if } 0 \leq x_2 \leq \theta_2 \\ 0 & \text{otherwise.} \end{cases}$$

As usual, * represents a missing feature value.

(a) Start with an initial estimate $\theta^0 = \binom{2}{4}$ and analytically calculate $Q(\theta|\theta^0)-$ the **E step** in the EM algorithm. Be sure to consider the normalization of your distribution.

(b) Find the θ that maximizes your $\theta|\theta^0-$ the **M step.**

$[7+8=15]$

8. You are given the predicted scores and true binary labels for a binary classification task. The classifier outputs a score for each instance (a higher score indicates a higher likelihood of class 1), and the true labels are either 0 or 1.

Instance	Predicted Score	True Label
A	0.95	1
B	0.85	0
C	0.75	1
D	0.65	0
E	0.55	1
F	0.45	0

(a) Use the thresholds $t = \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ to classify each instance as positive (1) if its score $\geq t$, otherwise negative (0). For each threshold, compute *True Positive Rate* (TPR) and *False Positive Rate* (FPR).

(b) Approximately plot the ROC curve using the FPR and TPR values obtained for all thresholds.

(c) Calculate the Area Under the ROC Curve (AUROC).

(d) Briefly explain what the ROC curve and AUROC value suggest about the classifier's performance.

$[3+3+2+2=10]$

9. A drone is attempting to localize itself along a 1D path using a GPS sensor. At each time step t, the GPS reports a noisy observation $x_t \in \mathbb{R}$ of the true position $\theta \in \mathbb{R}$. The GPS noise is Gaussian: $x_t \mid \theta \sim \mathcal{N}(\theta, \sigma^2)$. The drone has a prior belief about its location based on an earlier map or prediction, modeled as: $\theta \sim \mathcal{N}(\mu_0, \tau^2)$.You are given a sequence of n GPS readings $\{x_1, x_2, \ldots, x_n\}$.

   (a) Derive the MAP estimate of the true position θ using the observations and the prior.

   (b) Interpret the MAP estimate as a weighted average of the prior mean μ_0 and the sample mean of the GPS observations. What do the weights depend on?

[6 + 4 = 10]