Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 1

QUESTION 1

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

To find the optimal values of w_c and M_c for the given objective/loss function, we'll minimize the loss function with respect to these parameters. We can do this by taking derivatives with respect to w_c and M_c and setting them equal to zero.

The objective function to minimize is:

$$\min_{w_c, M_c} \frac{1}{N_c} \sum_{n:y_n=c} \left((x_n - w_c)^T M_c (x_n - w_c) - \log |M_c| \right)$$

Derivative with respect to w_c:

Differentiate the objective function with respect to w_c and set the result to zero:

$$\frac{\partial}{\partial w_c} \left(\frac{1}{N_c} \sum_{n:y_n=c} \left((x_n - w_c)^T M_c (x_n - w_c) - \log |M_c| \right) \right) = 0$$

Now, let's simplify this expression:

$$\frac{1}{N_c} \sum_{n:y_n=c} -2M_c(x_n - w_c) = 0$$

Rearranging:

$$\sum_{n:y_n=c} M_c(x_n - w_c) = 0$$

Now, we can isolate w_c:
M_c is considered constant for the given class and while differentiating with respect to w_c
Thus after rearranging final value of w_c is as follows:

$$w_c = \frac{1}{N_c} \sum_{n:y_n=c} x_n$$

Now, let's find the optimal value of M_c. We need to differentiate the objective function with respect to M_c:

Derivative with respect to M_c:

$$\frac{\partial}{\partial M_c}\left[\frac{1}{N_c}\sum_{n=1}^{N_c}(x_n - w_c)^T M_c(x_n - w_c) - \log|M_c|\right]$$

First, let's find the derivative of the first term:

$$\frac{\partial}{\partial M_c}\left[\frac{1}{N_c}\sum_{n=1}^{N_c}(x_n - w_c)^T M_c(x_n - w_c)\right]$$

Using the fact that M_c is symmetric, we can rewrite this as:

$$\frac{1}{N_c}\sum_{n=1}^{N_c}(x_n - w_c)(x_n - w_c)^T$$

Now, let's find the derivative of the second term:

$$\frac{\partial}{\partial M_c}\left[-\log|M_c|\right] = -M_c^{-1}$$

Now, set the derivative of the entire expression with respect to M_c equal to zero:

$$\sum_{n=1}^{N_c}(x_n - w_c)(x_n - w_c)^T - N_c M_c^{-1} = 0$$

$$M_c^{-1} = \frac{1}{N_c}\sum_{n=1}^{N_c}(x_n - w_c)(x_n - w_c)^T$$

Taking the inverse of both sides:

$$M_c = \frac{1}{N_c}\left(\sum_{n=1}^{N_c}(x_n - w_c)(x_n - w_c)^T\right)^{-1}$$

So, the optimal value of M_c is as above.

In the special case when M_c is an identity matrix ($M_c = I$), the optimization problem simplifies to:

$$(w_c, I) = \arg\min_{w_c, I}\frac{1}{N_c}\sum_{n:y_n=c}\left((x_n - w_c)^T I(x_n - w_c) - \log|I|\right)$$

Since the determinant of the identity matrix is 1, the $\log|I|$ term becomes 0, and we are left with:

$$(w_c, I) = \arg\min_{w_c, I}\frac{1}{N_c}\sum_{n:y_n=c}(x_n - w_c)^T(x_n - w_c)$$

This is equivalent to minimizing the mean squared distance from the class mean w_c to the data points x_n for class c. In other words, it reduces to a standard least-squares classification problem, where we are trying to find the class mean w_c that minimizes the sum of squared distances to the data points in that class.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

QUESTION 2

Yes, the one-nearest-neighbor (1-NN) algorithm is consistent in the noise-free setting. In this scenario, where every training input is labeled correctly, 1-NN will always classify any new point by finding the nearest neighbor from the training data, which is guaranteed to be the correct class. As the amount of training data approaches infinity, the error rate of 1-NN approaches zero, making it consistent in this idealized scenario.
But it is theoretical. In reality when we have large data set which almost approach to infinity, finding distance of test data point from each and every point will be almost too expensive and theoretically there are such infinite points.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

QUESTION 3

In the context of regression, where the goal is to predict real-valued labels, we use different criteria to choose a feature to split on compared to classification. Two commonly used criteria are Mean Squared Error (MSE) and Variance. These criteria quantify the homogeneity or diversity of the set of real-valued labels at each node. Let's discuss them in more detail:

1. **Mean Squared Error (MSE):**

   The MSE measures the average squared difference between the predicted values and the actual values at a given node. Lower MSE values indicate that the predicted values are closer to the actual values, which signifies greater homogeneity.

   Mathematically, for a node with n data points and their corresponding labels $\{y_1, y_2, \ldots, y_n\}$, the MSE is calculated as:

   $$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

   Where: - y_i is the label of the i-th data point. - $\bar{y}$ is the mean (average) of the labels in the node, calculated as $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

   When constructing a decision tree for regression, we aim to minimize the MSE when selecting features for splitting. We choose the feature that results in the lowest MSE after the split, indicating that it leads to a more homogeneous group of data points in terms of their real-valued labels.

2. **Variance:**

   Variance is another metric used for assessing the homogeneity of labels at a node. It quantifies the spread or dispersion of the labels in a node. Lower variance indicates that the labels are closer to each other, which implies greater homogeneity.

   Mathematically, for a node with n data points and their corresponding labels $\{y_1, y_2, \ldots, y_n\}$, the variance is calculated as:

   $$Variance = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

   Where $\bar{y}$ is the mean of the labels, as defined above.

   Similar to MSE, when constructing a decision tree for regression, we aim to minimize the variance when selecting features for splitting. The feature that results in the lowest variance after the split is chosen because it leads to a more homogeneous group of data points with respect to their real-valued labels.

In summary, for regression tasks, we prefer to split on features that minimize the Mean Squared Error (MSE) or Variance because these criteria quantify the homogeneity of the real-valued labels at each node in the decision tree. The feature that results in the lowest MSE or variance after the split is selected as the splitting criterion.

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

QUESTION

4

$f(x*) = \sum_{i=1}^{N} y_n w_n$

But f(x*) is basically prediction on x*. So it should also obey the equation: $f(x*) = w'^T x*$

Equate the above two equations

$w'^T x* = \sum_{i=1}^{N} y_n w_n$

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

QUESTION 5

The given loss function is:

$$L(w) = \sum_{n=1}^{N}(y_n - w^T x_n)^2$$

When we apply feature masking to the features x_m using $\hat{x}_n = x_n \odot m_n$, the modified loss function becomes:

$$L'(w) = \sum_{n=1}^{N}(y_n - w^T \tilde{x}_n)^2$$

Here, $\tilde{x}_n = x_n \odot m_n$, and m_n is a binary mask vector drawn from a Bernoulli distribution with parameter p.

Let's examine the estimation of this modified loss function:

$$\begin{aligned} E[L'(w)] &= E\left[\sum_{n=1}^{N}(y_n^2 - 2y_n w^T \tilde{x}_n + (w^T \tilde{x}_n)^2)\right] \\ &= \sum_{n=1}^{N}(E[y_n^2] - 2E[y_n \sum_{d=1}^{D} w_d x_{nd} m_{nd}] + E[(w^T \tilde{x}_n)^2]) \end{aligned}$$

Since $E[m_{nd}] = p$, we can express $E[\sum_{d=1}^{D} w_d x_{nd} m_{nd}]$ as $pw^T x_n$. This simplifies the equation to:

$$\begin{aligned} E[L'(w)] &= \sum_{n=1}^{N}((y_n - pw^T x_n)^2 - (pw^T x_n)^2 + E[(w^T \tilde{x}_n)^2]) \\ &= \sum_{n=1}^{N}(y_n - pw^T x_n)^2 - \sum_{n=1}^{N} p^2 \left(\sum_{d=1}^{D} w_d^2 x_{nd}^2 + \sum_{\substack{d \neq e \\ d,e \in [1,D]}} w_d x_{nd} w_e x_{ne}\right) + \sum_{n=1}^{N} E\left[\sum_{d=1}^{D} w_d^2 x_{nd}^2 m_{nd}^2\right] + E\left[\sum_{\substack{d \\ d,e \in}}\right. \end{aligned}$$

Further simplifying:

$$\begin{aligned} E[L'(w)] &= \sum_{n=1}^{N}((y_n - pw^T x_n)^2 - p^2 \sum_{d=1}^{D} w_d^2 x_{nd}^2 - p^2 \sum_{\substack{d \neq e \\ d,e \in [1,D]}} w_d x_{nd} w_e x_{ne}) + \sum_{d=1}^{D} p w_d^2 x_{nd}^2 + \sum_{\substack{d \neq e \\ d,e \in [1,D]}} p^2 w_d x_{nd} w_e x_{ne} \\ &= \sum_{n=1}^{N}(y_n - pw^T x_n)^2 + p(1-p)\sum_{d=1}^{D} w_d^2 x_{nd}^2 \end{aligned}$$

The same principle applies when the initial loss function is minimized with an additional regularization term. In that case, the loss function becomes:

$$L''(w) = \sum_{n=1}^{N}(y_n - pw^T x_n)^2 + \lambda \sum_{d=1}^{D} w_d^2 x_{nd}^2$$

Here, λ represents the product of p and $(1 - p)$. Feature masking and L2 regularization are closely related concepts, both encouraging smaller weight values, simplifying the model, and reducing the risk of overfitting.

Introduction to ML (CS771), Autumn 2023
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Hemang M Khatri
Roll Number: 231110016
Date: September 15, 2023

QUESTION 6

Output:

Method 1:

```
C:\Users\khatr\PycharmProjects\ML_Assignment\venv\Scripts\python.exe C:\Users\khatr\PycharmProjects\ML_Assignment\m1.py
Accuracy: 46.89320388349515%

Process finished with exit code 0
```

Method 2:

```
C:\Users\khatr\PycharmProjects\ML_Assignment\venv\Scripts\python.exe C:\Users\khatr\PycharmProjects\ML_Assignment\m2.py
Accuracy: 58.090614886731395% for Lambda 0.01
Accuracy: 59.54692556634305% for Lambda 0.1
Accuracy: 67.39482200647248% for Lambda 1
Accuracy: 73.28478964401295% for Lambda 10
Accuracy: 71.68284789644012% for Lambda 20
Accuracy: 65.08090614886731% for Lambda 50
Accuracy: 56.47249190938511% for Lambda 100

From the above lambda values, 10 i.e. 73.28478964401295%

Process finished with exit code 0
```