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

Student Name: Siddhant Jakhotiya
Roll Number: 211030
Date: September 15, 2023

QUESTION 1

The given loss function to be optimized is

$$L\left(\mathbf{w}_c, \mathbf{M}_c\right) = \underset{\mathbf{w}_c, \mathbf{M}_c}{\arg\min} \sum_{\mathbf{x}_n : y_n = c} \frac{1}{N_c} (\mathbf{x}_n - \mathbf{w}_c)^T \mathbf{M}_c (\mathbf{x}_n - \mathbf{w}_c) - \log |\mathbf{M}|$$

We can use first-order optimization to find the optimal values of $\mathbf{w}_c$ and $\mathbf{M}_c$.
(it is assumed that L is convex)
1. Taking the partial derivative wrt $\mathbf{w}_c$, we get

$$\frac{\partial L\left(\mathbf{w}_c, \mathbf{M}_c\right)}{\partial \mathbf{w}_c} = - \sum_{\mathbf{x}_n : y_n = c} \frac{1}{N_c} (\mathbf{M}_c + \mathbf{M}_c^T)(\mathbf{x}_n - \mathbf{w}_c) \quad \because \frac{\partial \mathbf{a}^T \mathbf{X} \mathbf{a}}{\partial \mathbf{a}} = (\mathbf{X} + \mathbf{X}^T)\mathbf{a} \text{ for a vector } \mathbf{a} \text{ and a matrix } \mathbf{X}$$

Proof. $\mathbf{a}^T \mathbf{X} \mathbf{a}$ is a scalar. We can express it as

$$\mathbf{a}^T \mathbf{X} \mathbf{a} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_i X_{ij} a_j$$

We can find the partial derivative of $\mathbf{a}^T \mathbf{X} \mathbf{a}$ with respect to a_k (the k-th component of $\mathbf{a}$):

$$\frac{\partial (\mathbf{a}^T \mathbf{X} \mathbf{a})}{\partial a_k} = \frac{\partial}{\partial a_k} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} a_i X_{ij} a_j \right) = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\partial (a_i X_{ij} a_j)}{\partial a_k}$$

Since coefficients of $\mathbf{a}$ in different dimensions are independent, the above expression will simplify to (in matrix form):

$$\frac{\partial (\mathbf{a}^T \mathbf{X} \mathbf{a})}{\partial \mathbf{a}} = (\mathbf{X} + \mathbf{X}^T)\mathbf{a}$$

(Note: we can use the result $\frac{\partial \mathbf{b}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{b}$ and the chain rule (replacing $(\mathbf{x}_n - \mathbf{w}_c$ by $\mathbf{a}$) to obtain the derivative wrt $\mathbf{w}_c$ written above ($\mathbf{b}$ would be a vector with all entries -1).

For an optima, the gradient should be 0. Thus we get

$$(\mathbf{M}_c + \mathbf{M}_c^T) \sum_{\mathbf{x}_n : y_n = c} \frac{1}{N_c} (\mathbf{x}_n - \mathbf{w}_c) = \mathbf{0}$$

Denote the entire vector summation as some vector $\mathbf{a}$. Also, since all positive-definite matrices are symmetric, so we have

$$2\mathbf{M}_c \mathbf{a} = \mathbf{0}$$

Premultiplying by $\mathbf{a}^T$ on both sides

$$\mathbf{a}^T \mathbf{M}_c \mathbf{a} = \mathbf{0}$$

Since $\mathbf{M}_c$ is positive definite, this can only imply that the vector $\mathbf{a}$ is a null vector. So we get

$$\sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}\mathbf{w}_c = \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n)$$
$$\therefore \hat{\mathbf{w}}_c = \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n) = \mu_c$$

Where μ_c is the mean vector of the class.
2. Taking the partial derivative wrt $\mathbf{M}_c$, we get

$$\frac{\delta L\left(\mathbf{w}_c, \mathbf{M}_c\right)}{\delta \mathbf{M}_c} = \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n - \mathbf{w}_c)(\mathbf{x}_n - \mathbf{w}_c)^T - \left((\mathbf{M}_c)^{-1}\right)^T$$

Since

$$\frac{\partial \mathbf{a}^T \mathbf{X} \mathbf{a}}{\partial X} = \mathbf{a}\mathbf{a}^T$$

And

$$\frac{\partial \ln |\mathbf{X}|}{\partial \mathbf{X}} = \left((\mathbf{X})^{-1}\right)^T$$

Simplifying the equation we get

$$((\mathbf{M}_c)^{-1})^T = \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n - \mathbf{w}_c)(\mathbf{x}_n - \mathbf{w}_c)^T$$

Taking the transpose

$$(\mathbf{M}_c)^{-1} = \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n - \mathbf{w}_c)(\mathbf{x}_n - \mathbf{w}_c)^T$$

Taking the inverse

$$\mathbf{M}_c = \left(\sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n - \mathbf{w}_c)(\mathbf{x}_n - \mathbf{w}_c)^T \right)^{-1}$$

In the special case when $\mathbf{M}_c = \mathbf{I}$, the model reduces to

$$L\left(\hat{\mathbf{w}}\right) = \arg\min_{\mathbf{w}_c} \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c}(\mathbf{x}_n - \mathbf{w}_c)^T(\mathbf{x}_n - \mathbf{w}_c) = \arg\min_{\mathbf{w}_c} \sum_{\mathbf{x}_n:y_n=c} \frac{1}{N_c} \left\|\mathbf{x}_n - \mathbf{w}_c\right\|$$

This is equivalent to the LwP model.

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

Student Name: Siddhant Jakhotiya
Roll Number: 211030
Date: September 15, 2023

QUESTION 2

With an infinite number of noise-free training points, it would be like having a continuous spectrum of points defining each class thus leading to well defined boundaries between classes as compared to models with finite data where the boundary is estimated (using a training algorithm).

Formally,

Claim 1.1. Given an *infinite* amount of *noise-free* data, each test point in $\mathbb{R}^D$ is labelled correctly and the label is the same as that of the training point that is closest to it in the $\mathbb{R}^D$ space (here, D is the number of features in the training data)

Proof. Denote $\mathbf{x}_*$ as the test point. For any point in $\mathbf{x}_* \in \mathbb{R}^D$, we have 2 cases:

Case 1. The test point coincides with one of the training points: then the vector $\mathbf{x}_*$ and the vector for one of the points in the training set (say, $\mathbf{a}$) would be the same i.e. $\mathbf{x}_* = \mathbf{a} \implies \|x - a\| = 0$ so the closest point to $\mathbf{x}_*$ is $\mathbf{a}$ and it is assigned this label. This is correct trivially since the data is *noise-free* so the label for $\mathbf{a}$ is correct thus the label for $\mathbf{x}_*$ is correct as well.

Case 2. The test point doesn't coincide with any of the training points:
Suppose claim 1.1 doesn't hold
$\implies \exists$ points $\mathbf{a}$ and $\mathbf{b}$ in the training set s.t. the correct label for $\mathbf{x}_*$ is the label of $\mathbf{a}$ however $\|\mathbf{x}_* - \mathbf{a}\| > \|\mathbf{x}_* - \mathbf{b}\|$ i.e. it is assigned the *incorrect* label with the 1-NN strategy. Let the label of $\mathbf{a}$ be blue and that of $\mathbf{b}$ be red.

Claim 1.2. We can add $\mathbf{x}_*$ and remove $\mathbf{a}$ from the training set

Proof. Since we have an infinite number of noise-free training points and by our assumption, $\mathbf{x}_*$ is assigned the correct label, we can add it to the training set to get a training set which also has infinite points and is noise-free. We can also remove $\mathbf{a}$ from the training set because of the same reasons.

Claim 1.3. $\exists \mathbf{c}$ belonging to blue class s.t.

$$\|\mathbf{c} - \mathbf{a}\| < \|\mathbf{x}_* - \mathbf{a}\|$$

Proof. Suppose the claim doesn't hold. By claim 1.2, remove the point $\mathbf{a}$ from the training set and add the point $\mathbf{x}_*$. Now consider a test example $\mathbf{x}'_* = \mathbf{a}$. The closest point to it either be $\mathbf{x}_*$ or another point not belonging to the blue class $\implies \mathbf{x}'_*$ gets assigned to a class that isn't blue but that is incorrect since the training data provided to us was noise-free - A contradiction. Thus, claim 1.3 holds.

Denote $\|\mathbf{x}_* - \mathbf{a}\|$ as d_1 and $\|\mathbf{c} - \mathbf{a}\|$ as d_2. We know that $d_2 = fd_1$ where $f \in (0, 1)$. Now, we add $\mathbf{x}_*$ to the training set and consider a test point $\mathbf{x}_1$ lying on the line segment joining $\mathbf{x}_*$ and $\mathbf{a}$ dividing it internally in the ratio 1:2 (i.e. $\|\mathbf{x}_1 - \mathbf{a}\| = \frac{2d_1}{3}$). This point doesn't lie in the original training set (if it were, it would have been the closest point to $\mathbf{x}_*$ and not $\mathbf{a}$). We prove

that the closest point to $\mathbf{x}_1$ in the training set is $\mathbf{x}_*$. Suppose it isn't. Consider a D-dimensional circle centred at $\mathbf{x}_1$ with a radius of $\frac{d_1}{3}$. The point on this circle farthest from $\mathbf{x}_*$ is at a distance of $\frac{2d_1}{3}$ thus if there was any other training examples inside this circle, it would also have been the closest training exaple to $\mathbf{x}_*$- A contradiction since the closest point to $\mathbf{x}_*$ in the training set is $\mathbf{a}$ at a distance of d_1.

So, the point closest to $\mathbf{x}_1$ is $\mathbf{x}_*$, which is at a distance of $\frac{2d_1}{3}$ from $\mathbf{a}$. Assign red label to $\mathbf{x}_1$ and add it to the training set (valid by claim 1.2).
Consider another point $\mathbf{x}_2$ dividing the segment joining $\mathbf{x}_1$ and $\mathbf{a}$ internally in the ratio 1:2. By the same argument, $\mathbf{x}_2$ is assigned red. Repeat this process i times s.t. $\left(\frac{2}{3}\right)^i < f$. So, $\mathbf{x}_i$ is at a distance of $(\frac{2d_1}{3})^i$ from $\mathbf{a}$ and $\|\mathbf{c}-\mathbf{a}\| = fd_1 > \|\mathbf{x}_i - \mathbf{a}\|$. Now, remove the point $\mathbf{a}$ from the training set and consider the test point $\mathbf{x}_{i+1}$ s.t. $\mathbf{x}_{i+1} = \mathbf{a}$. The closest point to it is $\mathbf{x}_i \implies$ it is marked red. However, this is a contradiction since the original training dataset was noise-free. Thus our assumption was wrong - either the correct label for $\mathbf{x}_*$ was not blue or $\mathbf{a}$ is not the point closest to it. We have arrived at a contradiction thus claim 1.1 holds.

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

Student Name: Siddhant Jakhotiya
Roll Number: 211030
Date: September 15, 2023

QUESTION 3

A good and viable criterion for measuring homogeneity/purity in samples for a regression task can be the variance of the labels at a given node. At a given node, let $\mathbf{S}$ be the set of labels and let $\overline{y}$ be their mean. Then the variance at a given node is

$$Var = \frac{1}{|S|} \sum_{y:y \in S} (y - \overline{y})^2$$

Now, we create a split based on a parameter and the branches now possess the smaller sets $\mathbf{S}_1$ and $\mathbf{S}_2$. The equivalent of information gain (call it IG') for regression thus is

$$IG' = \frac{1}{|S|} \sum_{y:y \in S} (y - \overline{y})^2 - \frac{1}{|S_1|} \sum_{y:y \in S_1} (y - \overline{y_1})^2 - \frac{1}{|S_2|} \sum_{y:y \in S_2} (y - \overline{y_2})^2$$

Where $\overline{y_1}$ and $\overline{y_2}$ denote the mean of the elements in $\mathbf{S}_1$ and $\mathbf{S}_2$ respectively.

As with the decision trees for classification, we would choose the parameter that maximizes the IG' at each node with thresholds for when to stop splitting.

Qualitatively speaking, for regression tasks, a leaf would predict a value. We wish to build our tree such that all the training labels at each leaf are as close to each other as possible (and quite obviously if they are numerically close to each other, they are close to their mean as well). This is measured quantitatively using variance, as described above.

QUESTION 4

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

Now, note that since the training data is fixed and already available to us, $(\mathbf{X}^T\ \mathbf{X})^{-1}\mathbf{X}^T$ is a constant matrix. So, let us denote it by $\mathbf{C}$. $\mathbf{w}$ now is

$$\mathbf{w} = \mathbf{C}\mathbf{y}$$

For a given test point $\mathbf{x}_*$, the prediction is

$$\begin{aligned} f(\mathbf{x}_*) &= \mathbf{x}_*^T\mathbf{w} \\ &= \mathbf{x}_*^T\mathbf{C}\mathbf{y} \end{aligned}$$

Note that $\mathbf{x}_*^T\mathbf{C}$ can be written as

$$\begin{bmatrix} \sum_{i=1}^D x_i c_{i1} & \sum_{i=1}^D x_i ci2 & \dots & \sum_{i=1}^D x_i c_{iN} \end{bmatrix}$$

where c_{ij} is the entry in the ith row and jth column of $\mathbf{C}$ and x_i is the ith feature of $\mathbf{x}_*$
Therefore, the product $\mathbf{x}_*^T\mathbf{C}\mathbf{y}$ is

$$f(\mathbf{x}_*) = \sum_{n=1}^N \left(\sum_{i=1}^D x_i c_{in} \right) y_n = \sum_{n=1}^N w_n y_n$$

Here, $w_n = \mathbf{x}_*^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_n$ where $\mathbf{x}_*$ is the test input of dimensions Dx1. Thus $\mathbf{w}$ depends upon the combined effect of **all** training inputs (since we are taking a dot product with $(\mathbf{X}^T\mathbf{X})^{-1}$) and doesn't depend upon the proximity of $\mathbf{x}_*$ to any training example. In *weighted KNN*, even though we assign weights, we only look at the K closest neighbours which is a localized approach. Here, all inputs contribute to the "weight".

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

Student Name: Siddhant Jakhotiya
Roll Number: 211030
Date: September 15, 2023

QUESTION 5

The loss function is

$$\begin{aligned} L(\mathbf{w}) &= \sum_{n=1}^{N} (y_n - \mathbf{w}^T \tilde{\mathbf{x}}_n)^2 \\ &= \sum_{n=1}^{N} \left(y_n - \sum_{d=1}^{D} (w_d \tilde{x}_{nd}) \right)^2 \end{aligned}$$

where $\tilde{\mathbf{x}}_n = \mathbf{x}_n \circ \mathbf{m}_n$. Here, $m_{nd} \sim$ Bernoulli(p) $\forall d \in [D]$ When we expand the square, we get terms of the forms:

- y_n^2
- $w_d^2 \tilde{x}_{nd}^2$
- $y_n w_d \tilde{x}_{nd}$
- $w_d \tilde{x}_{nd} w_{d'} \tilde{x}_{nd'}$ where $d \neq d'$

We can ignore term 1 since it is a constant being added and doesn't have any randomness. Terms of type 2:

$$\mathbb{E}\left[w_d^2 \tilde{x}_{nd}^2\right] = w_d^2 \mathbb{E}\left[x_{nd}^2 m_{nd}^2\right]$$

(Since w_d is constant wrt the probability distribution and E[cx] = cE[x] for a random variable x and constant c). Now, m_{nd} takes value 1 with probability p and 0 with probability 1-p i.e. m_{nd} is drawn only once so $\mathbb{E}[x_{nd}^2 m_{nd}^2] = p x_{nd}^2$ and not p^2

$$\mathbb{E}[x_{nd}^2 m_{nd}^2] = p x_{nd}^2 \implies w_d^2 x_{nd}^2 \mathbb{E}\left[m_{nd}^2\right] = p w_d^2 x_{nd}^2$$

$\because m_{nd}$ is 1 with Similarly, for terms of type 3 and 4,

$$\begin{aligned} \mathbb{E}\left[y_n w_d \tilde{x}_{nd}\right] &= y_n w_d x_{nd} p \\ \mathbb{E}\left[w_d \tilde{x}_{nd} w_{d'} \tilde{x}_{nd'}\right] &= w_d w_{d'} x_d x_{d'} p^2 \qquad (1) \end{aligned}$$

(1) follows as the features are dropped *independently*

Thus, the expected value of the loss function becomes

$$\begin{aligned}
\mathbb{E}\left[L(w)\right] &= \sum_{n=1}^{N}\left[y_n^2 + p\sum_{d=1}^{D}(w_d^2 x_{nd}^2 - 2y_n w_d \tilde{x}_{nd}) + p^2\sum_{d\neq d'} w_d w_{d'} x_{nd} x_{nd'}\right] \\
&= \sum_{n=1}^{N}\left(y_n - p\mathbf{w}^T\mathbf{x}_n\right)^2 + (p-p^2)\sum_{n=1}^{N}\sum_{d=1}^{D} w_d^2 x_{nd}^2 \quad \text{(add and subtract } p^2 w_d^2 x_{nd}^2 \text{ in the equation)} \\
&= \sum_{n=1}^{N}\left(y_n - p\mathbf{w}^T\mathbf{x}_n\right)^2 + (p-p^2)\sum_{d=1}^{D}\left((\sum_{n=1}^{N} x_{nd}^2) w_d^2\right) \\
&= L(\mathbf{w}) + (p-p^2)\sum_{d=1}^{D}\left(\lambda_d' w_d^2\right)
\end{aligned}$$

The first term is the MSE loss (with a scaling factor p that can be viewed as normalization for the inputs) and in the second term,
$\because p \in (0,1), p > p^2$ so the second term is positive (λ' is also positive since it is a sum of squares).
So, the second term is a positive constant times R(w) which is equivalent to regularization.

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

Student Name: Siddhant Jakhotiya
Roll Number: 211030
Date: September 15, 2023

QUESTION 6

Method 1

Using method 1, we get a test-set accuracy of **46.89%**

Method 2

Using method 2, when we vary the value of λ, we get different test-set accuracies. They are listed in the table below:

Value of λ	**Test-set Accuracy**
0.01	58.09061527252197
0.1	59.54692363739014
1	67.39482283592224
10	**73.28478693962097**
20	71.6828465461731
50	65.08090496063232
100	56.47249221801758

As can be seen, the maximum accuracy was obtained using the value of $\lambda = 10$.