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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 1

To find optimal values for w_c and M_c for the given objective/loss function.We aim to do this by leveraging first-order optimality conditions, setting the partial derivatives with respect to w_c and M_c are zero.

Objective function:

$$(w_c, M_c) = \arg\min \left[\frac{1}{N_c} \sum_{x_n : y_n = c} \Big((x_n - w_c)^\top M_c (x_n - w_c) - \log |M_c| \Big) \right]$$

1. Derivative with respect to w_c:

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

Simplifying:

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

Resulting in:

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

The optimal value of w_c for class c is, therefore, the mean of all the training examples in class c.

2. Derivative with respect to M_c: ϵ

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

So

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

Derivative of the log determinant term:

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

Thus,

$$\frac{1}{N_c} \sum_{x_n : y_n = c} \Big((x_n - w_c)(x_n - w_c)^\top \Big) - M_c^{-\top} = 0$$

M_c:

$$M_c = \left(\frac{1}{N_c} \sum_{x_n : y_n = c} \left((x_n - w_c)(x_n - w_c)^\top \right) \right)^{-\top}$$

This gives the optimal value of M_c for class c.

Consider the special case, M_c is an identity matrix, i.e., $M_c = I$ $(D \times D)$.

Optimization problem reduces to:

$$(w_c, M_c) = \arg\min \left[\frac{1}{N_c} \sum_{x_n : y_n = c} \left((x_n - w_c)^\top I (x_n - w_c) - \log |I| \right) \right]$$

Or

$$w_c = \arg\min \left[\frac{1}{N_c} \sum_{x_n : y_n = c} \left((x_n - w_c)^\top (x_n - w_c) \right) \right]$$

This is the same as learning with prototypes using Euclidean distance as w_c is still mean of class c.

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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 2

Yes, the one-nearest-neighbor (1-NN) method works well when there is no noise and all of the training data is correctly labeled. This consistency comes from the fact that if there are an endless number of correctly labeled training examples, there will always be a training point that is very close to (almost on top of) a test data point. As the number of training data points gets closer to infinity, the chance that a test point is a close match gets closer to 1. So, 1-NN will always identify test data correctly, making no mistakes, because it treats each test point as if it was already in the large training set.

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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 3

We start by dividing the feature space into J distinct and non-overlapping regions $R_1, R_2, ..R_J$. We split the predictor space so that the sum of squares of the residuals is as small as possible. So we can move forward with our greedy plan. The process starts at the top of the tree (the root node) and divides the prediction space into smaller and smaller pieces. Each split makes two stems, or "binary." At each step of the process, it chooses the split that will make the best tree at that step. It doesn't look ahead and choose a split that will make a better tree overall (hence the name "greedy").

$$R_1(j,s) = \{X \,|\, X_j < s\} \quad \text{and} \quad R_2(j,s) = \{X \,|\, X_j \geq s\}$$

Now, for each combination of (j,s), we look for the pair (j,s) that reduces the following sum of residual errors.

$$\Sigma_{i:x_i \in R_1(j,s)}(y_i - \hat{y}R1)^2 + \Sigma_{i:x_i \in R_2(j,s)}(y_i - \hat{y}R2)^2$$

where $\hat{y}R_1$ is the mean of all samples in $R1$ and $\hat{y}R_2$ is the mean of samples in $R2$.

For each splitting variable, it is easy to find the split point (s), so it is possible to find the best pair (j, s) by going through all of the inputs. Once we've found the best way to split the data, we divide it into the two parts that result and split each of those again. Then, this process is done again for each of the areas that were made.So, we can find a feature to split for a regression task in this way.

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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 4

Given: $\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. Therefore, a prediction at test input $\mathbf{x}_*$ can be written as

$$y_* = \hat{\mathbf{w}}^T\mathbf{x}_* = \mathbf{x}_*^T\hat{\mathbf{w}}$$

Therefore,

$$y_* = \mathbf{x}_*^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

$$y_* = \mathbf{W}\mathbf{y}$$

where $\mathbf{W} = \mathbf{x}_*^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$. Therefore, $\mathbf{W} = (w_1, w_2, \ldots, w_N)$ comes out to be a $1 \times N$ matrix. We can also write $\mathbf{y} = (y_1, y_2, \ldots, y_N)^T$.

Therefore,

$$y_* = \mathbf{W}\mathbf{y} = \sum_{n=1}^{N} w_n y_n$$

Therefore, w_n are the nth index of the $1 \times N$ matrix $\mathbf{W}$. Knowing that $\mathbf{X}$ is a matrix whose rows are the N training vectors x_n, w_n can be written as:

$$w_n = \mathbf{x}_*(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_n$$

So, w_n depends on the input $\mathbf{x}_*$ and all the training data from $\mathbf{x}_1$ to $\mathbf{x}_n$. Since there exists $\mathbf{X}^T\mathbf{X}$ term in the expression of w_n, which is not the case with weighted **kNN**, where individual weights depend only on $\mathbf{x}_*$ and $\mathbf{x}_n$. Also, $\mathbf{x}_*$ comes in the numerator in the case of this, while for **kNN** it comes in the denominator. Another difference is that w_n are expressed as products of $\mathbf{x}_*$ while in **kNN** they are expressed as sums in the denominator.

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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 5

Let the given Loss function using masked input be

$$\mathbf{L}(\mathbf{M}) = \sum_{n=1}^{N} \left(y_n - \mathbf{w}^T \tilde{\mathbf{x}}_n\right)^2$$

Therefore,

$$\mathbf{L}(\mathbf{M}) = \|\mathbf{y} - \tilde{\mathbf{X}}\mathbf{w}\|^2$$

and

$$\mathbf{L(M)} = \|\mathbf{y} - (\mathbf{R} * \mathbf{X})\mathbf{w}\|^2$$

When the input $\mathbf{X}$ is dropped out so that any input dimension is kept with probability $\mathbf{p}$, then the expected value of $L(M)$, i.e.,

$$E_{R\sim\text{Bernoulli}(n;p)}[\ \mathbf{L(M)}]$$

This needs to be minimized with respect to $\mathbf{w}$. So, let the new objective function be:

$$L(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ E_{R\sim\text{Bernoulli}(n;p)}[(\mathbf{y} - (\mathbf{R} * \mathbf{X})\mathbf{w})^T (\mathbf{y} - (\mathbf{R} * \mathbf{X})\mathbf{w})] \right\}$$

Let $\mathbf{k} = \mathbf{y} - (\mathbf{R} * c\mathbf{X})\mathbf{w}$ and $\mu = E_{R\sim\text{Bernoulli}(n;p)}[\mathbf{k}] = \mathbf{y} - p\mathbf{X}\mathbf{w}$. Then, we get

$$L(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ [E_{R\sim\text{Bernoulli}(n;p)}[\mathbf{k}^T\mathbf{k}] \right\}$$

Therefore,

$$L(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ \mu^T\mu + \text{TRACE}\left[(\mathbf{k} - \mu)(\mathbf{k} - \mu)^T\right] \right\}$$

$$L(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ (\mathbf{y} - p\mathbf{X}\mathbf{w})^T(\mathbf{y} - p\mathbf{X}\mathbf{w}) + p(1-p)\text{TRACE}\left[(\mathbf{X}\mathbf{w})(\mathbf{X}\mathbf{w})^{\mathbf{T}}\right] \right\}$$

$$L(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ \|\mathbf{y} - p\mathbf{X}\mathbf{w}\|^2 + p(1-p)\left\| \sqrt[2]{\text{diag}(\mathbf{X}^T\mathbf{X})}\mathbf{w} \right\|^2 \right\}$$

Comparing $L(\mathbf{w})$ with ridge regression objective function,

$$L_{\text{ridge}}(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ (\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda\mathbf{w}^T\mathbf{w} \right\}$$

$$L_{\text{ridge}}(\mathbf{w}) = \arg\min_{\mathbf{w}} \left\{ \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|^2 \right\}$$

Clearly, the objective $L(\mathbf{w})$ bears a resemblance to the objective function in ridge regression. Consequently, it can be seen as the minimization of a regularized loss function. In this context, the term $p(1-p)\left| \sqrt[2]{\text{diag}(\mathbf{X}^T\mathbf{X})}\mathbf{w} \right|^2$ serves as the regularizer, while $|\mathbf{y} - p\mathbf{X}\mathbf{w}|^2$ represents the squared loss.

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

Student Name: Anshu Kumar Rajkumar
Roll Number: 210155
Date: September 15, 2023

QUESTION 6

My solution to problem 6

METHOD 2:

For the lambda = 0.01 the accuracy is 58.090614886731395
For the lambda = 0.1 the accuracy is 59.54692556634305
For the lambda = 1 the accuracy is 67.39482200647248
For the lambda = 10 the accuracy is 73.28478964401295
For the lambda = 20 the accuracy is 71.68284789644012
For the lambda = 50 the accuracy is 65.08090614886731
For the lambda = 100 the accuracy is 56.47249190938511

Clearly Lambda = 10 gives best accuracy (On closer inspection I found Lambda=5 gave the best accuracy among all integral Lambdas