

FoML Assign 4

Anurag Sarva

cs24mtech14003

1 VC-Dimension -

IN this i have to determine the value of the **VC-dimension** in the hypothesis space H and H is defined or written as:

$$H = \{h : h(x) = 1 \text{ if } p < x < q, \text{ and } h(x) = 0 \text{ otherwise}\},$$

The defination if the hypothesis space H is defined by the interval on the real line R^1 .

p and q are the parameters which will helps to define the interval (p, q) . x is a point which will be classified as 1, it is only when $p < x < q$.

VC-dimension in the hypothesis space will be the maximum number of points that will be **shattered** by the hypotheses in the space of hypothesis. And a set of the points is going to shattered it is only when, for the all possible labeling of the points, there will be exists a hypothesis in H that will correctly classifying the points, it is according to the labeling.

For the any single points x_1 , we can be classify it as either
this 1: Choose $p < x_1 < q$,
or this 0: Choose $x_1 \notin (p, q)$.

Among them, one point will be gets shattered.

Let the two point x_1 and x_2 are in the relation $(x_1 < x_2)$ be the two points. Now i am consider all the possible labelings here -

(0,0): Choose $p \geq x_2$,

(0,1): Choose $p < x_2$ and $q = x_2$,

(1,0): Choose $p = x_1$ and $q \leq x_1$,

(1,1): Choose $p < x_1$ and $q > x_2$.

Since all the labelings are going to be achievable, and here 2 points can be shattered.

Let us take threee different points which is $x_1, x_2, \text{ and } x_3$ and all have a relation like this $(x_1 < x_2 < x_3)$. And also i am considering the labels (1,0,1) -

So, there is no even a single interval (p, q) that will be including x_1 and x_3 , but it will excludes x_2 . Thus, all the 3 points **cannot** be shattered.

So, here is the **Conclusion** of my observation from all the above analysis -

When, $d = 2$ - We can only be shattered 2 points.

When, $d = 3$ - We cannot be able to shattered even 3 points.

Finally, the **VC-dimension** of H is equals to 2.

2 Regularizer -

Here is the Problem Setup -

We are given with the linear model in the question -

$$y(x, w) = w_0 + \sum_{k=1}^D w_k x_k$$

For the N data samples is like (x_i, t_i) , and sum of the square error function can be written as -

$$E(w) = \frac{1}{2} \sum_{i=1}^N \left(y(x_i, w) - t_i \right)^2$$

Let us assume the Gaussian noise is $\epsilon_k \sim \mathcal{N}(0, \sigma^2)$ and then it is adding independently in to the each inputs variables x_k , giving the noisy data as input like $x'_k = x_k + \epsilon_k$.

Here my aim is to relate minimize a errors which is averaged over the noisy data so that it minimize the error on the noisy free data with in a ℓ_2 -regularization term.

Errors with the Noisy Data's -

Noisy input with the $x'_i = [x_{i1} + \epsilon_1, \dots, x_{iD} + \epsilon_D]$. Here is the error function with the noisy inputs are shown below -

$$E_{noisy}(w) = \frac{1}{2} E_{\epsilon} \left[\sum_{i=1}^N \left(y(x'_i, w) - t_i \right)^2 \right]$$

Now expanding this $y(x'_i, w)$ -

$$y(x'_i, w) = w_0 + \sum_{k=1}^D w_k (x_{ik} + \epsilon_k) = w_0 + \sum_{k=1}^D w_k x_{ik} + \sum_{k=1}^D w_k \epsilon_k$$

Then the error equation will becomes -

$$E_{noisy}(w) = \frac{1}{2} \sum_{i=1}^N E_{\epsilon} \left[\left(w_0 + \sum_{k=1}^D w_k x_{ik} + \sum_{k=1}^D w_k \epsilon_k - t_i \right)^2 \right]$$

Now simplify the equation by using the Noise Properties -

Here i am expanding the square in the equation -

$$\left(w_0 + \sum_{k=1}^D w_k x_{ik} + \sum_{k=1}^D w_k \epsilon_k - t_i\right)^2 = \left(w_0 + \sum_{k=1}^D w_k x_{ik} - t_i\right)^2 + 2\left(w_0 + \sum_{k=1}^D w_k x_{ik} - t_i\right) \sum_{k=1}^D w_k \epsilon_k + \left(\sum_{k=1}^D w_k \epsilon_k\right)^2$$

Here i am taking the expectation over the ϵ_k is -

1. $E[\epsilon_k] = 0$, so the cross term is going to vanishes here -

$$E\left[2\left(w_0 + \sum_{k=1}^D w_k x_{ik} - t_i\right) \sum_{k=1}^D w_k \epsilon_k\right] = 0$$

2. This is for the noisy variance term -

$$E\left[\left(\sum_{k=1}^D w_k \epsilon_k\right)^2\right] = \sum_{k=1}^D w_k^2 E[\epsilon_k^2] = \sigma^2 \sum_{k=1}^D w_k^2$$

Therefore the the expected error will becomes -

$$E_{noisy}(w) = \frac{1}{2} \sum_{i=1}^N \left(w_0 + \sum_{k=1}^D w_k x_{ik} - t_i\right)^2 + \frac{1}{2} \sigma^2 \sum_{k=1}^D w_k^2$$

Here is the relation for the Regularization is -

The first term is the standard sum of squares error $E_{clean}(w)$, computed over the noise free data -

$$E_{clean}(w) = \frac{1}{2} \sum_{i=1}^N \left(w_0 + \sum_{k=1}^D w_k x_{ik} - t_i\right)^2$$

Here is the second term which is ℓ_2 regularization with the regularization strength is written as $\lambda = \frac{\sigma^2}{2}$. And then omitting the biasness w_0 from the regularization, and then the objective function will be written as -

$$E_{noisy}(w) = E_{clean}(w) + \frac{\lambda}{2} \sum_{k=1}^D w_k^2$$

3 Hierarchical Clustering -

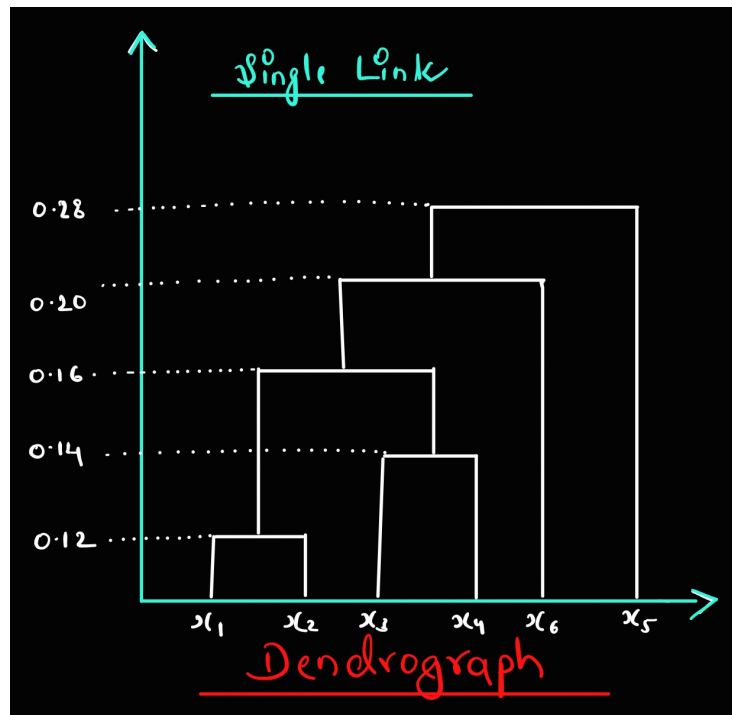
Here is the distance matrix which is given in the question for the 6 data points are -

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0					
x_2	0.12	0				
x_3	0.51	0.25	0			
x_4	0.84	0.16	0.14	0		
x_5	0.28	0.77	0.70	0.45	0	
x_6	0.34	0.61	0.93	0.20	0.67	0

(a) Dendrogram for Single Link -

In the Hierarchical clustering single link is one of its type of clustering which merge the clusters totally based on the minimum distance between two points from the distance matrix.

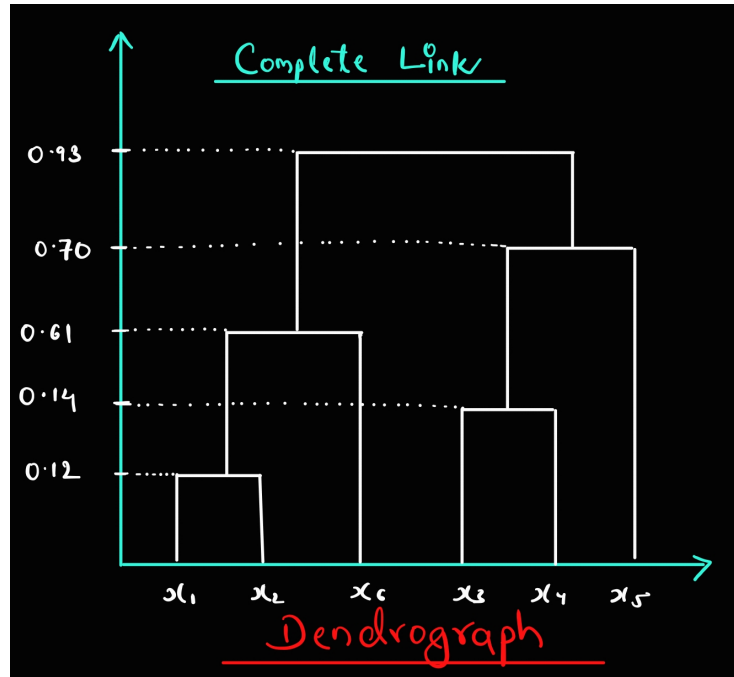
Below is the **dendrogram** for the single link clustering -



(b) Dendrogram for Complete Link -

In the Hierarchical clustering complete link is one of its type of clustering which merge the clusters totally based on the maximum distance between two points from the distance matrix.

Below is the **dendrogram** for the complete link clustering -



(c) Modification of a Distance Matrix -

As per the question i, have to make some changes in the distance matrix so that the answers of first two part of this question may not vary means remain same for part (a) and (b), we have to change the distances between two pair of points such that the minimum and the maximum distances between the two points are equal, for the same pair of points and making sure that final result for both the part remain same.

Here i am changing the values between -

(x_3, x_4) to 0.16 and (x_5, x_6) to 0.45

Here is the modification in the distance matrix provided in the question -

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	0.12	0.51	0.84	0.28	0.34
x_2	0.12	0	0.25	0.16	0.77	0.61
x_3	0.51	0.25	0	0.16	0.70	0.93
x_4	0.84	0.16	0.16	0	0.45	0.20
x_5	0.28	0.77	0.70	0.45	0	0.45
x_6	0.34	0.61	0.93	0.20	0.45	0

From the above modified distance matrix, the dendrogram of single link and the dendrograms complete link will be remain same.

4 Principal Component Analysis -

Let's suppose each of the data points \mathbf{x} is a M dimensional vectors are of the form -

$$\mathbf{x} = a\delta_k = (0, \dots, 0, a, 0, \dots)^T,$$

here a is in the k -th slot, and the k, a are the random variables. and k is the uniformly distribution over the $1, \dots, M$, and $P(a)$ is the arbitrary.

(a) Covariance Matrix -

The expectation exercise of the \mathbf{x} is can be written as -

$$E[\mathbf{x}] = E[a] \cdot \frac{1}{M}(1, 1, \dots, 1)^T.$$

And the covariance matrix can be written as following -

$$\mathbf{C} = E[(\mathbf{x} - E[\mathbf{x}])(\mathbf{x} - E[\mathbf{x}])^T]$$

Then after calculation, it can be written as the -

$$C_{ij} = \lambda + \mu\delta_{i,j},$$

You can replace this in the above equation $\lambda = \frac{Var(a)}{M}$ and $\mu = Var(a) - \frac{Var(a)}{M}$

(b) Eigenvalues and Eigenvectors -

Here i have to find the eigen value and eigen vector -

The covariance matrix has only one eigen vector in the form of -

$$\mathbf{v}_1 = \frac{1}{\sqrt{M}}(1, 1, \dots, 1)^T,$$

$$\lambda_1 = Var(a).$$

And the number of the remaining $M - 1$ eigen vectors is the orthogonal to the \mathbf{v}_1 and correspond to the eigenv value can be written as -

$$\lambda_2 = \frac{Var(a)}{M}.$$

(c) PCA as a Feature Selection Method -

May be the PCA is not a good or effective feature selection process or method for the this problem because of the following -

1. Here is the data point that are constrained to one of the dimension which is defined by the a , so the PCA for the this variation are effectively.

2. And the uniformity for the k and then the structure of the eigen value suggest to that the PCA may not differentiate between the meaning full feature , which is beyond to the first principal component (PC).

Therefore, the PCA gets reduces the dimensionality to one of the effective features, but may this not provides the additional insight into the entire structure of the original data.

5 Logistic Regression -

(a) Implementation of the Logistic Regression Classifier -

The logistic regression classifier is implemented by the following component:

- **Sigmoid Function** - ($\sigma(z)$): This will map the input z (it is the linear combination are the features and the weights) to the probability b/w 0 and 1. Formula -

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- **Cross-Entropy Error Function** - It helps to measures the error for the binary classification task. Formula -

$$E = -\frac{1}{m} \sum [y \cdot \log(P) + (1 - y) \cdot \log(1 - P)]$$

Here, $P = \sigma(f_{\theta}(x_1, x_2))$.

- **Gradient Descent** - It updates the weights of $\theta_0, \theta_1, \theta_2$ iteratively so that it minimize the cross-entropy error value.

(b) Results -

(i) Logistic Model and Cross-Entropy Error Function -

The Logistic regression model can be written as -

$$P(\hat{y} = 1|x_1, x_2) = \frac{1}{1 + e^{-f_{\theta}(x_1, x_2)}}$$

where $f_{\theta}(x_1, x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$.

The cross-entropy error function is written as -

$$E = -\frac{1}{m} \sum [y \cdot \log(P) + (1 - y) \cdot \log(1 - P)]$$

(ii) Gradient Descent Update for One Iteration -

By using the training dataset and the initial weights which is -

$$\theta_0 = -1, \theta_1 = 1.5, \theta_2 = 0.5$$

and the learning rate is 0.1, and the weight are updating using the gradient descent model.

Updating the weights after one iteration -

$$\theta_0 = -1.0032, \theta_1 = 1.5054, \theta_2 = 0.5020$$

Then updating the logistic regression model is written below -

$$f_{\theta}(x_1, x_2) = -1.0032 + 1.5054 \cdot x_1 + 0.5020 \cdot x_2$$

(iii) Model Evaluation at Convergence

- After running gradient descent model of ML for the 100 iterations, the model will converged with the loss value of the approximately 0.5212 in the last iteration of the model. By using this model, then the predictions were made for the test dataset.

Predictions for Test Dataset -

$$Prediction : - [1, 1, 0, 1, 1, 1]$$

Evaluation Metrics -

• Accuracy -

$$Accuracy = \frac{NumberOfCorrectPredictions}{TotalPredictions} = \frac{4}{6} \times 100 = 66.67\%$$

• Precision -

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} = \frac{3}{5} = 0.60$$

• Recall -

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} = \frac{3}{3} = 1.00$$

Deliverables

-

Code

- The implementation part of the logistic regression classifier, by including the gradient descent and the model evaluation, has been provided in Python code in ipynb notebook.

```
Question 5.)

*** PART A ***
Iteration number is 1 and the loss value is 0.5569500797547652
Iteration number is 11 and the loss value is 0.5527434547813901
Iteration number is 21 and the loss value is 0.5486156177889855
Iteration number is 31 and the loss value is 0.5445502635976432
Iteration number is 41 and the loss value is 0.5405404093716722
Iteration number is 51 and the loss value is 0.5365827998335017
Iteration number is 61 and the loss value is 0.5326756772349874
Iteration number is 71 and the loss value is 0.5288178901925459
Iteration number is 81 and the loss value is 0.5250085368196484
Iteration number is 91 and the loss value is 0.5212468215749928

*** PART B (i) ***
Writing the Logistic Model as  $P(\hat{y}=1|x_1, x_2) = \frac{1}{1 + \exp(-f_\theta(x_1, x_2))}$ 
 $f_\theta(x_1, x_2) = \theta_0 + \theta_1*x_1 + \theta_2*x_2$ 
Cross Entropy Error Function can be written as  $E = -(1/m) * \sum [y*\log(P) + (1-y)*\log(1-P)]$ 

*** PART B (ii) ***
here is the updated weight  $\theta$  is [-1.00316626  1.50535086  0.50196867]
Logistic Regression Model After One Iteration -
 $f_\theta(x_1, x_2) = -1.0032 + 1.5054*x_1 + 0.5020*x_2$ 

*** PART B (iii) ***
Model Evaluation at Convergence:
accuracy is 66.67%
precision is 0.60
recall is 1.00
prediction is [1 1 0 1 1 1]
```

6 House Price Prediction -

Objective -

The objective of this project is to predict the house price by using appropriate machine learning models. The models of the ML is going to be evaluated is totally based on the **Root Mean Squared Logarithmic Error (RMSLE)** metric, that is particularly suitable for the regression task with the skewed target variables like a house prices.

Data Preprocessing -

- **Numerical Data** - All the missing values are handled by the median of that particular columns, and followed by the standardization with `StandardScaler`.
- **Categorical Data** - Missing value are filled by using the most frequently category's, and the data is encoded by using the `OneHotEncoder` to convert into categorical variables into the numeric format.

- **Pipeline** - The unified preprocessing pipeline is implemented by using the `ColumnTransformer` to making ensure consistent transformation across the both the training and the testing datasets.

Models used and their Results -

1. Gradient Boosting -

- **Train RMSLE** value is 0.0829
- **Test RMSLE** value is 0.1369
- **Model Description** - The Gradient-Boosting is follows the ensemble method that helps to builds the decision tree sequentially. And each tree corrects the error of the previous one by the optimizing a specific loss function. And it is also very highly effective into the capturing of complex patterns and generalizes well to the unseen data.

2. Random Forest -

- **Train RMSLE** value is 0.0605
- **Test RMSLE** value is 0.1534
- **Model Description** - The Random Forest is the another ensemble ML techniques that construct a number of decision tree and then averaging their outputs. Which is robust to the over-fitting, and it may not capturing the nuanced feature interactions as effectively as boosting algorithms in ML.

```
Random Forest - Train RMSLE: 0.0605, Test RMSLE: 0.1534
Gradient Boosting - Train RMSLE: 0.0829, Test RMSLE: 0.1369
Best Model: Gradient Boosting, Test RMSLE: 0.1369
```

Best Model -

- **Gradient Boosting Regressor** - By using this model i get the best performance with the test RMSLE value of **0.1369**, it indicates the stronger generalizations for the unseen datas.
- **Kaggle Leader Board** - In this i achiveve **2385 rank out of 6122** submissions, and placing this model in the **top 39%** of the participants.

Analysis of the Model Performance -



- **Gradient Boosting's Strength** -
 - The sequential approach of this model is minimise the prediction error in incrementally, and giving to the high accuracy.

- It captures the complex features interaction and also learns the pattern that will help to improve the generalization.

- **Random Forest's Limitation -**

- In this model averaging the prediction can intensively miss the subtle relationships in the data.
- The model is shown a slightly overfitting, as evidenced by the very large gap between the train value is 0.0605 and the test value is 0.1534 RMSLE scores.

The **Gradient Boosting Model in ML** beats the Random Forest due to its important features such as robust learning approach and ability to generalize better. With the Kaggle rank is **2385 out of 6122**, the performance of the model demonstrates us a very strong performance and also the competitiveness. Then further hyperparameter tuning and the feature engineering helps me to improve the result for the deployment and the future competitions.

2386	Anurag Sarva IIT H		0.13903	2	22s
 Your Best Entry! Your most recent submission scored 0.13903, which is the same as your previous score. Keep trying!					