1.>Implementing KNN on IRIS data

KNN algorithm:

In KNN the algorithm found k nearest neighbour of the test data from the training set.Then it returns the nearest of the neighbour class from k nearest neigbours whose frequency is maximum.

Iris Data:No. of data=150

No.of class:3(iris-versicolr,iris-verginica,iris-setosa)

No. Of features:4

Computing the Accuracy by cross validating the given data.

|  |  |
| --- | --- |
| K-value | Accuracy |
| 5 | 96.0 |
| 7 | 96.666 |
| 11 | 97.33 |
| 13 | 98.0 |
| 17 | 94.66 |
| 19 | 96.0 |
| 23 | 94.66 |

From table k for maximum accuracy is=13

**Confusion matrix**: A **confusion matrix** is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known

Mean (confusion) accuracy=96.88%

Misclassification rate=3.11%

Precision:

For Iris-setosa:100%

For Iris-Virginica=90.74%

For Iris-Versicolor=95.4%

2.>Implementation of multi label KNN classification:

For a given test instance, its K nearest neighbours in the training set are firstly identified. After that, based on statistical information gained from the label sets of these neighboring instances, i.e. the number of neighboring instances belonging to each possible class maximum a posteriori (MAP) principle is utilized to determine the label set for the unseen instance

Scene data:

No of class=6

No of features in each data=294

Scene-data(no.of class=6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| K-value | HAMMING LOSS | ONE ERROR | COVERAGE EROR | RANK LOSS | AVERAGE PRECISION |
| 7 | 0.0499 | 0.0499 | 1.60 | 0.074 | 0.929 |
| 9 | 0.05 | 0.058 | 1.64 | 0.084 | 0.916 |
| 11 | 0.066 | 0.066 | 1.7 | 0.095 | 0.904 |
| 13 | 0.033 | 0.034 | 1.5 | 0.055 | 0.955 |
| 17 | 0.091 | 0.091 | 1.85 | 0.125 | 0.866 |

Yeast-data(No. of class=14)

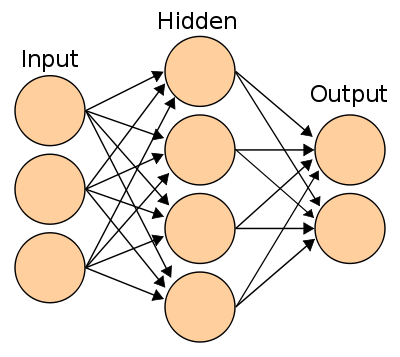
No of class=14

No of feature in each data=103

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| K-value | HAMMING LOSS | ONE ERROR | COVERAGE EROR | RANK LOSS | AVERAGE PRECISION |
| 15 | 0.249 | 0.24 | 11.69 | 0.48 | 0.62 |
| 17 | 0.217 | 0.21 | 12.5 | 0.48 | 0.64 |
| 19 | 0.23 | 0.22 | 13.5 | 0.52 | 0.60 |
| 21 | 0.22 | 0.22 | 13.0 | 0.54 | 0.63 |
| 23 | 0.21 | 0.21 | 12.05 | 0.52 | 0.64 |

Ref: ML-KNN:A lazy learning approach to multi-label learning

3.> ELM(Extreme learning machine)



W B

W=Weights between input and hidden layer.

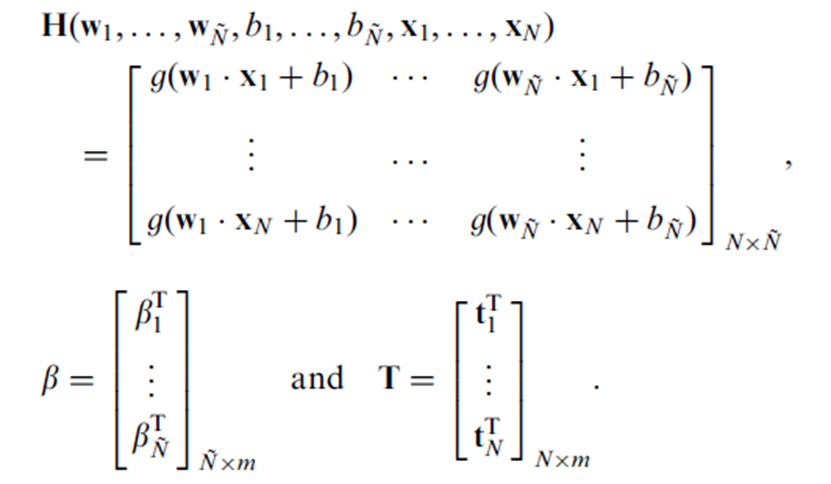
T=Target class

~N=No. of nodes in hidden layer.

N=No. of training examaple.

n=no. of nodes in input.

Beta=weights between hidden layer and output layer.



The ELM training algorithm can be summarized as follows

* Randomly assign the hidden node parameters
  + e.g., the input weights **wi** and biases ***bi***for additive hidden nodes, *i* = 1*, . . . , L*.
* Calculate the hidden layer output matrix **H**.
* Obtain the output weight vector using Moore Penrose inverse.



Accuracy measure on iris-data

|  |  |
| --- | --- |
| L | Accuray |
| 5 | .86 |
| 8 | .90 |
| 10 | .92 |
| 15 | .94 |
| 20 | .96 |
| 25 | .96 |
| 35 | .965 |

ELM With autoencoder.

Nof=no. of nodes of hidden layer of auto encoder.

L=35(no .of nodes of hidden layer of classifier)

|  |  |
| --- | --- |
| NOF | Accuracy |
| 3 | .93 |
| 5 | .94 |
| 10 | .96 |
| 15 | .97 |
| 20 | .98 |
| 25 | .95 |
| 30 | .93 |

**Multilabel Classification using ELM.**

The ElM model can be converted into multilabel classifier. The output of the ELM is fed into a MLP for thresholding.

Without autoencoder

|  |  |
| --- | --- |
| Data name | Hamming Loss |
| Scene data | 0.11 |
| Yeast Data | 0.12 |

With 2-layer autoencoder

|  |  |
| --- | --- |
| Data Name | Hamming Loss |
| Scene Data | Ranges from 0.20-0.50 |
| Yeast Data | 0.17 |

Multi label classification using MLP

2 layers(10,15 nodes)

Without autoencoder

|  |  |
| --- | --- |
| Data Name | Hamming Loss |
| Scene-Data | 0.12 |
| Yeast-Data | 0.13 |

With 2 layer elm autoencoder

|  |  |
| --- | --- |
| Data Name | Hamming Loss |
| Scene Data | Ranges from 0.20-0.50 |
| Yeast Data | 0.22 |

**The Elm model takes much less time to compute than the MLP and the losses are almost comparable.**

**Clustering on Multilabel Data.**

**By Kmeans** :

The data is clustered into k clusters.

Observation: The data with same label set is clustered into one cluster approximately.

**By DBSCAN:**

The DBSCAN algorithm does not make enough cluster for the data and the clustering is not good.