

Analysis of Distance Measures Using K-Nearest Neighbor Algorithm on KDD Dataset

Punam Mulak¹, Nitin Talhar²

¹Computer Department, AISSMS, Collage of Engineering, Kennedy Road, Pune, India

²Professor, Computer Department, AISSMS, Collage of Engineering, Kennedy Road, Pune, India

Abstract: Classification is the process of analyzing the input data and building a model that describes data classes. K-Nearest Neighbor is a classification algorithm that is used to find class label of unknown tuples. Distance measure functions are very important for calculating distance between test and training tuples. Main aim of this paper is to analyze and compare Euclidian distance, Chebychev distance and Manhattan distance function using K-Nearest Neighbor. These distance measures are compared in terms of accuracy, specificity, sensitivity, false positive rate and false negative rate on KDD dataset. Manhattan distance gives high performance.

Keywords: K-nearest neighbor, lazy learner, eager learner, knowledge discovery and data mining, intrusion detection system

1. Introduction

Classification is a technique used to build a classification model from training examples. The class label of unknown samples can be predicted by using a classification model. Classification model is like a black box that assigns a class label to unknown records. Two types of learner are available in classification model [1]. First is eager learner. Before receiving a test tuple, eager learner builds a model from given training tuples. Second is lazy learner. Lazy learner stores all the training tuples and waits for test tuples. Whenever test tuple is available, then classification is done based on similarity between stored tuples and new unknown tuple. K-nearest neighbor is lazy learner used for classification of data.

In K-nearest neighbor algorithm, tuples are predicted based on the class of its nearest neighbors [2]. Following figure 1 shows the 1-, 2- and 3- nearest neighbors of data point which is placed at the center of circle. In figure 1(a), nearest neighbor of data point is negative so negative class label is assigned to data point. If there is tie between the two classes, then random class is chosen for data point. As shown in figure 1(c), three nearest neighbor are present. One is negative and other two is positive. So in this case, majority voting is used to assign class label to data point.

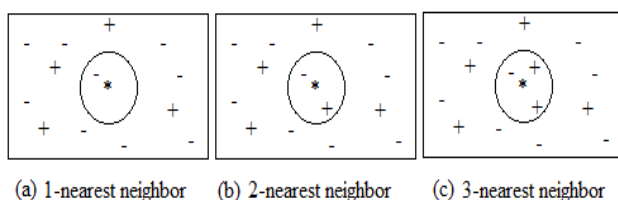


Figure 1: The 1-, 2- and 3- nearest neighbors

K Nearest Neighbor is very simple, highly efficient and effective technique. It is used in many applications like data mining, text categorization, object recognition, etc [3]. Lots of work has been done on KNN algorithm. T. M. Cover et Al. [4] has proposed k-nearest neighbor approach. Nearest neighbor is found based on the value of k. This approach is effective in case of large training data. This approach has some drawbacks. T. Bailey has presented kNN which is

based on weights [5]. This approach overcomes the disadvantages of previous technique. It helps to make algorithm global. In paper, K-nearest neighbor algorithm is introduced along with three distance function. Comparative analysis is done on KDD dataset. Classifier is evaluated in terms of different performance measure that is accuracy, sensitivity, specificity, etc.

The paper is arranged as follows: K-Nearest Neighbor algorithm is described in Section 2. Distance measures are given in section 3. Dataset is described in section 4. Implementation is given in section 5. Classifier evaluation techniques are presented in section 6. Results that compare the distances are described in section 7. And section 8 describes conclusion of paper.

2. K-Nearest Neighbor Algorithm

Cover and Hart proposed the K Nearest Neighbor algorithm, in 1968. K-Nearest neighbor is instance based lazy learner. Because it stores all the training instances and delays the process of model building until test is given for classification. It is used for prediction.

For describing training tuples, n-numeric attributes are used. N-dimensional space is used to store all training examples. When a test sample is available, K nearest neighbor algorithm finds the k training samples that are closest to that test sample. Different distance functions are used to search difference between training and testing samples which is describe in next section. K nearest neighbor algorithm is given below [6]:

Algorithm

1. Determine k nearest neighbors and D set of training example.
2. for each test example x_i do
Calculate $d(x_i, y_j)$ based on distance measure.
3. Select the k closest training examples y_i to test example x_i .
4. Use majority voting to classify the test examples
5. End for.

K-nearest neighbor algorithm is simple to implement and outcome of KNN is easy to justify. This algorithm is robust with small error ratio.

3. Proximity Distance Measures

Distance measures are very essential to find the similarity and dissimilarity between data points. Similarity is the measure of two objects is alike. And dissimilarity is two objects are different. The main aim of distance metric calculation is to find appropriate or similar distance. In this paper, following distance measures are used in KNN and evaluation is done on KDD dataset.

3.1 Euclidian Distance

K-Nearest Neighbor can be calculated by using Euclidian distance. It gives efficiency and productivity [7]. It is a distance between two points in Euclidian space. It computes the root of square differences between co-ordinates of pair of data points [8]. Formula for Euclidian distance is given below.

$$Dist_{xy} = \sqrt{\sum_{k=1}^m (x_{ik} - x_{jk})^2} \quad (1)$$

3.2 Chebychev Distance

It is also called as maximum value distance. It is calculated as the absolute magnitude of the differences between coordinate of a pair of data points. This distance is given below.

$$Dist_{xy} = \max_k |x_{ik} - x_{jk}| \quad (2)$$

3.3 Manhattan Distance

It calculates the differences between coordinates of pair of data points.

$$Dist_{xy} = |X_{ik} - X_{jk}| \quad (3)$$

4. KDD Dataset

KDD is a knowledge discovery and data mining dataset. These dataset is mainly used in intrusion detection system. KDD dataset contains 41 features and class label that is normal or attack [9]. Total 23 attacks are available in KDD dataset along with its class label that is denial of service, Remote to Local Attack (R2L), probing Attack, User-to-Root (U2R) and normal. KDD dataset is given in table 1[10].

Table 1: KDD Dataset

Feature No	Feature Name	Feature Type
1	Duration	Continuous
2	protocol type	Discrete
3	Service	Discrete
4	Flag	Discrete
5	source bytes	Continuous
6	destination bytes	Continuous
7	Land	Discrete
8	wrong fragment	Continuous
9	Urgent	Continuous
10	Hot	Continuous
11	failed logins	Continuous
12	logged in	Discrete
13	Compromised	Continuous

14	root shell	Continuous
15	su_attempted	Continuous
16	Root	Continuous
17	file creations	Continuous
18	Shells	Continuous
19	access files	Continuous
20	Outbound cmds	Continuous
21	is hot login	Discrete
22	is guest login	Discrete
23	Count	Continuous
24	srv count	Continuous
25	error rate	Continuous
26	srv error rate	Continuous
27	error rate	Continuous
28	srv error rate	Continuous
29	same srv rate	Continuous
30	diff srv rate	Continuous
31	srv diff host rate	Continuous
32	dst host count	Continuous
33	dst host same srv rate	Continuous
34	dst host same srv rate	Continuous
35	dst host diff srv rate	Continuous
36	dst host same src port rate	Continuous
37	dst host srv diff host rate	Continuous
38	dst host error rate	Continuous
39	dst host srv error rate	Continuous
40	dst host error rate	Continuous
41	dst host srv error rate	Continuous

5. Implementation

K-nearest neighbor algorithm is implemented using Euclidian distance, Manhattan distance and Chebychev Distance. KDD dataset is applied in system. Preprocessing is required on KDD. In data mining, preprocessing is important step. It mainly includes normalization, data cleaning, and data integration. Range of the KDD dataset is varying widely. So normalization is done to fit the values in specific range. In kNN algorithm, all training dataset is stored. And k value is determined.

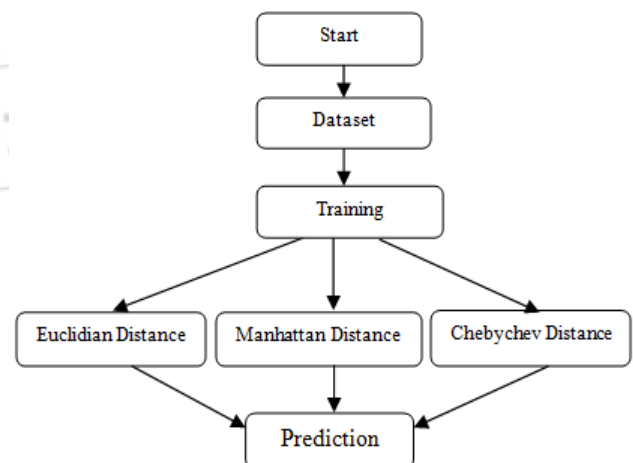


Figure 2: Flow of system

Figure 2 shows overall flow of the system. After normalization, distances between training and test examples are computed using Manhattan distance, Euclidian distance and Chebychev Distance. Unknown examples are predicted based on its nearest neighbor.

In Manhattan distance, absolute distance between stored

tuple and test tuples are calculated by using equation 3. In case of Euclidian distance, root square differences of training and test tuple are calculated. By using equation 2, Chebychev distances between tuples are calculated. And based on these distances, k closest training samples are found. Based on majority voting criteria, unknown tuples are classified and class labels are assigned. This kNN classifier is evaluated using different measures as given below.

6. Classifier Evaluation

It is very important that how accurately classifier will predict the class label of test examples. Different measures are available for evaluating the performance of classifiers.

- True positive vs. True Negative: True positive means positive tuples that are correctly classified as positive. True negative is negative tuples that are correctly predicted as negative. In case of KDD dataset, true positive is normal classes are classified as normal. And attack tuples are predicted as attack class.
- False positive rate (FPR): It is the fraction of negative tuples that are classified as positive. FPR is calculated using following equations.

$$FPR = \frac{\text{False Positive}}{(\text{True Negative} + \text{False Positive})} \quad (4)$$

- False Negative Rate (FNR): It is the fraction of positive tuples that are classified as negative. It calculated using:

$$FNR = \frac{\text{False Negative}}{(\text{True Positive} + \text{False Negative})} \quad (5)$$

- Accuracy: It is the percentage of test tuples that are correctly classified by classifier. It is defined as:

$$\text{Accuracy} = \frac{\text{True positive} + \text{true negative}}{\text{positive} + \text{negative}} \quad (6)$$

- Sensitivity: It is proportion of positive tuples that are correctly identified. Sensitivity is calculated using:

$$\text{sensitivity} = \frac{\text{true positive}}{\text{Positive}} \quad (7)$$

- Specificity: It is a proportion of negative tuples that are correctly identified by classifier. It is calculated using:

$$\text{Sensitivity} = \frac{\text{true negative}}{\text{negative}} \quad (8)$$

7. Experimental Results

Result is evaluated after implementation of K nearest neighbor algorithm using Euclidian distance, Manhattan distance and Chebychev Distance in terms of accuracy, sensitivity and specificity. Experiment is performed on KDD dataset.

Comparative graphs of three distance functions are given below. Manhattan distance gives better accuracy than Chebychev Distance and Euclidian distance as shown in figure 3.

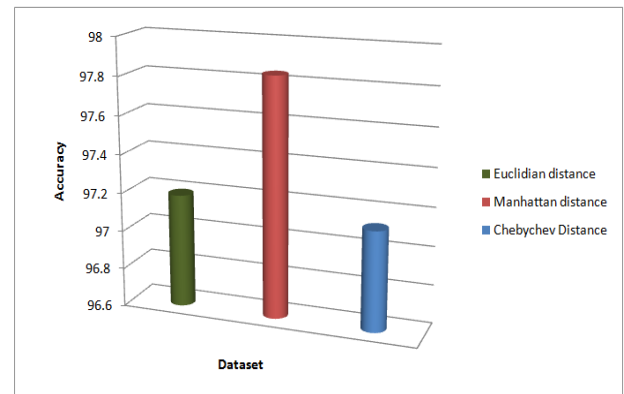


Figure 3: Accuracy Graph

As shown in figure 4, Manhattan distance gives sensitivity rate up to 96.76 %. Chebychev and Euclidian distance gives 95.68% and 95.79% sensitivity rate respectively.

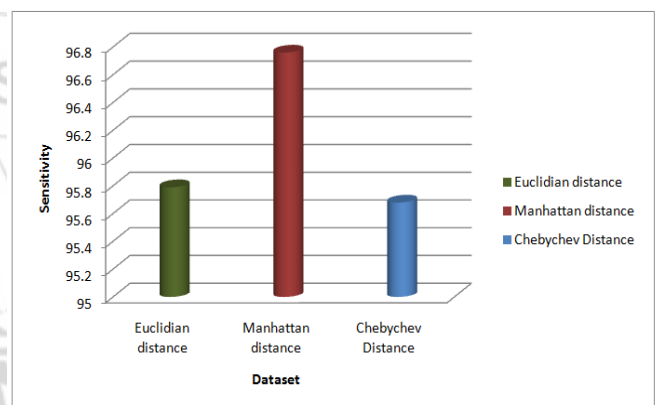


Figure 4: Sensitivity Graph

Following graph describes specificity rate of three distance function. Percentage of negative tuples that are correctly identified by Manhattan distance based K-NN classifier is more as compare to other two distances.

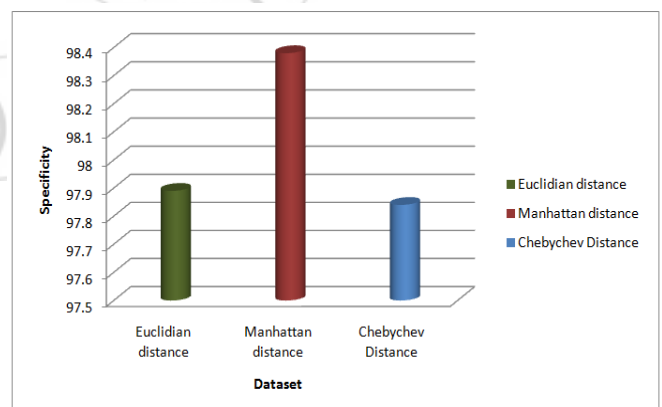


Figure 5: Specificity Graph

Following table shows the false positive rate calculated using distance measures. In case of KDD dataset, false positive rate is attack class that is incorrectly classified as normal class. Manhattan distance based KNN gives lower false positive rate as compare to Chebychev Distance and Euclidian Distance.

Table 2: False positive rate

Chebychev Distance	Euclidian Distance	Manhattan Distance
0.0215	0.0210	0.016

Number of normal class tuples are predicted as attack class is given in following table 3. FNR is given using these three distance measure on KDD dataset. In case of Manhattan distance, normal classes are classified as attack classes is less as compare to other distance function.

Table 3: False Negative Rate

Euclidian distance	Chebychev Distance	Manhattan distance
0.042	0.043	0.032

8. Conclusion

K-nearest neighbor is implemented using Euclidian distance, Manhattan distance and Chebychev distance on KDD dataset. KNN classifier is evaluated in terms accuracy, specificity, sensitivity and false positive rate. These distance measures are compared. It is observed that Manhattan distance gives better results than other distance measures. Performance of Euclidian distance is low as compare to Chebychev distance.

References

- [1] Jiawei Han, Micheline Kamber, Jian Pei, "Data Mining concepts and Technologies", Third Edition Elsevier.
- [2] Pang-Nang Tan, Michael Steinbach, Vipin Kumar, "Data Mining".
- [3] Nitin Bhatia , Vandana, "Survey of Nearest Neighbor Techniques" (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8, No. 2, 2010
- [4] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification", IEEE Trans. Inform. Theory, Vol. IT-13, pp 21-27, Jan 1967.
- [5] T. Bailey and A. K. Jain, "A note on Distance weighted k-nearest neighbor rules", IEEE Trans. Systems, Man Cybernetics, Vol.8, pp 311-313, 1978.
- [6] Asha Gowda Karegowda , M.A. Jayaram, A.S. Manjunath "Cascading K-means Clustering and K-Nearest Neighbor Classifier for Categorization of Diabetic Patients", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-3, February 2012.
- [7] Dasarathy, B. V., "Nearest Neighbor (NN) Norms, NN Pattern Classification Techniques". IEEE Computer Society Press, 1990.
- [8] "K-means with Three different Distance Metrics", International Journal of Computer Applications (0975 – 8887) Volume 67– No.10, April 2013.
- [9] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set", Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009).
- [10] M.Vidhya, "Efficient Classification of Portscan Attacks using Support Vector Machine", Proceedings of 2013 International Conference on Green High Performance Computing, March 14-15, 2013, India