Statistical Data Classification Using Instance Based Learning Algorithm

S. Krishna Srivatsava, Devavarapu Sreenivasarao, Shaik Khasim Saheb

Abstract: In the past two decades, with an exponential growth in data storage and huge data accumulated, an intelligent analysis of data is very important of the current scenario. K-NN Algorithm and Instance based learning is greatly helps in classification of data on grounds of similarity. The classification technique in K-Nearest Neighbor is evolved on the necessity to distinct analysis when authentic variable are not known and difficult to find out. The data is classified into two categories. That is, Trained Data Set and Test Data Set and the KNN algorithm is implemented by finding k-Nearest Neighbor between trained and test dataset by similarity distance measure.

Index Terms: KNN, Memory based learning, distant measure, word cloud

1 INTRODUCTION

AS day by day the data was increasing tremendously, it is required to use the data effectively. The data is very important and need to classify. There are many methods available to classify the data. But as the data was large we need new techniques and advanced classification approach to handle the data. Machine Learning was the advanced approach to classify the data. The Instance based learning (K-NN) is most prominent classification method in Machine learning.

1.1. MACHINE LEARNING

Machine Learning, an advanced approach in computer science is an emerged technology in artificial intelligence for visualizing pattern recognition and study of statistical models for data classification. It is the development of Algorithms to conduct predictions on data sets. Mathematical model is built by Machine Learning algorithms, with the sample data. That is, training data, to evolve predictions without explicit programming to conduct a particular task.

> *"A Computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."*

Machine Learning adopts supervised learning approach for classification and the Computer program understands the data input submitted to classify new observation. The data set to be taken as bi-class or multiclass. The classification problems are in wide range of fields i.e Document Classification, Speech and Hand Writing Recognition, Identification of Biometric etc. The different types algorithm of classification in Machine Learning [1].

- K-Nearest Neighbor
- Decision Trees
- Linear Classifiers:
- Logistic Regression,
- Naive Baye's Classifier
- Support Vector Machines
- Neural Networks
- Boosted Trees
- Random Forest

1.2. K-NEAREST NEIGHBOR

Many current industries implementing KNN in the problems related to data classification. In Machine Learning, three aspects are looked into

- Elucidate Output in Easiest
- Minimum Estimated Time
- Finest Prediction.

Aspects	Logistic regression	CART	Random Forest	KNN
Euclidean Output in Easiest	2	3	1	3
Minimum Estimate Time	3	2	1	3
Finest Prediction	2	2	3	2

KNN algorithm performs well with all required parametric values. It is invariably used for easier interpretation and minimum estimated time. KNN is an efficient algorithm for storing the existing parameters and the new data is classified on grounds of similarity measure. KNN is non parametric and comes under supervised learning [6]. Supervised learning is itself a data mining activity hypothesize function from marked training data. The training example in supervised learning consider vector as input object and required output. KNN does not adopts any assumptions on the fundamental data hence defined as non parametric algorithm. KNN algorithm strive to find out the data points belongs to a group comparing the other data points around it. A modal of the data set is does not generate before is " lazy learner" and an example of KNN. It only predicts neighbor data points when it asked and this

- ***S. Krishna Srivatsava*** *is currently pursuing B.Tech Degree program in Computer Science & Engineering, Sreenidhi Institute of Science and Technology, Affiliated to Jawaharlal Nehru Technical University Hyderabad, Telangana, India, PH-9381861509. E-mail: sksrivatsava@gmail.com*
- ***Devavarapu Sreenivasarao*** *is currently working an Assistant Professor in Computer Science & Engineering Department in Sreenidhi Institute of Science and Technology and Pursuing Ph. D (CSE) from Annamalai University, Chidambaram. Chennai. PH-9866014581. E-mail: sreenivasaraodevavarapu@gmail.com*
- ***Shaik Khasim Saheb*** *is currently working as Assistant Professor in Computer Science & Engineering Department in Sreenidhi Institute of Science and Technology, and and Pursuing Ph. D (CSE) from Annamalai University, Chidambaram. Chennai. PH-9642097865. E-mail: shaikkhasims@sreenidhi.edu.in*

makes KNN to be an efficient in data mining. KNN is well categorized based on working method of supervised classification algorithm. KNN easily stores everything in the training set and differentiate test documents. KNN is purely classified as memory based learning itself as instance based learning [14]. In Machine Learning it is required to have a more training data and named as 'Lazy Learner [15] as it not holds distinguish function from the training data and training data set is memorized. The training data points is divided into different classes for example the training data set points is divided into Class A, Class B, Class C and so on. We will take one query point and set the all nearest data points on the basis k-factor .where k will be 1,2. ..and so on. The most data points in a particular set of any class will be the resultant class for query point. For example If the most data points in a particular set of class A, the query will be defined as a class A.

2. CLASSIFICATION APROACH

An object is categorized with more number of options based on neighbor classes. An object is allocated to most usual class in KNN and implemented by distant measure [2].

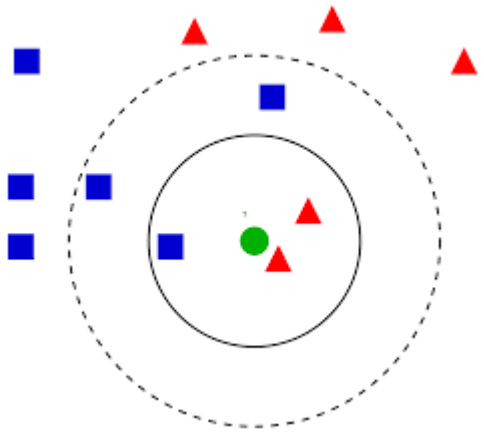


Fig.1: Classification approach of K-NN by the data points

In the above figure, different data points have different sizes like it has class Red points and class Blue points. The Green point is the query point in which it is to be classified as the fig shows the Green point is closer to the two Red points and one Blue point so the Green point is similar to Red point because it's nearer to the Reddest points so that the Green point is classified as Red class. By this the data will be classified to the respective neighbor classes so that the data is similar to one of the classes. We will get the most nearer neighbors by the distant measures there are three types of distant measures [8].

- Euclidean Distances
- Manhattan Distances

2.1. EUCLIDEAN DISTANCE

Euclidean distance function is more commonly used to compute the distance between two points A and B in a feature space. Let A = $(x_1,x_2,...x_m)$ and B=$(y_q,y_w,.....y_m)$, in which m is feature space dimension. To compute the distance between A and B, the Euclidean normalized metric is

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

2.2. MANHATTAN DISTANCE

The points on axes at right angles are measured in a plane i.e $A_1(x_1,y_1)$ and $A_2(x_2,y_2)$ is any two points in between distance which are at right angles on the axes is called Manhattan distance .the distance between A_1 and A_2 is $X_1 - X_2$ [11].

$$\sum_{i=1}^{k} | x_i - y_i |$$

Most of KNN classifications we will use Euclidean distances. Multi dimensional feature space with its points related to most of the occurrences. Each occurrence is entitled with a set of arithmetical properties. Each point in n-dimensional featured space corresponds to instances which are constituted with a numerical set. A set of vectors are assigned to training data each vector is labeled with class. Featured vectors are compared with different K nearest neighbours for classification.

Customer	Age	Income	No. credit cards	Class	Distance from John
George	35	35K	3	No	sqrt [(35-37)²+(35-50)² +(3-2)²]=15.16
Rachel	22	50K	2	Yes	sqrt [(22-37)²+(50-50)² +(2-2)²]=15
Steve	63	200K	1	No	sqrt [(63-37)²+(200-50)² +(1-2)²]=152.23
Tom	59	170K	1	No	sqrt [(59-37)²+(170-50)² +(1-2)²]=122
Anne	25	40K	4	Yes	sqrt [(25-37)²+(40-50)² +(4-2)²]=15.74
John	37	50K	2	YES	

Fig 2: Example for KNN classification

In the above figure, we have the data set by the name of customer and his income , age, credits. and each of the customer was classified into class yes and no and we take the query of john and we want to estimate the class of john. By the Euclidean distance of age, income, credits Anne, Rachel, George are the top three neighbors or top three least distance neighbors in which Anne and Rachel are in class YES, therefore john will be classified as class YES [8].

3. PARAMETER SELECTION

Parameter selection is nothing but a selection of k value. Let us take k value is n, then it is nothing but a we want get the top n nearest neighbors of the given query data point or top n similar neighbor points to the given data point. the query data point is classified to the most top n similar neighbor class. k Value is found to be small and it is subtle to data points and k value if larger is seems good, but from other classes it covers greater extent points. The thumb rule adopted here is k < sort(n), here n is number of models.

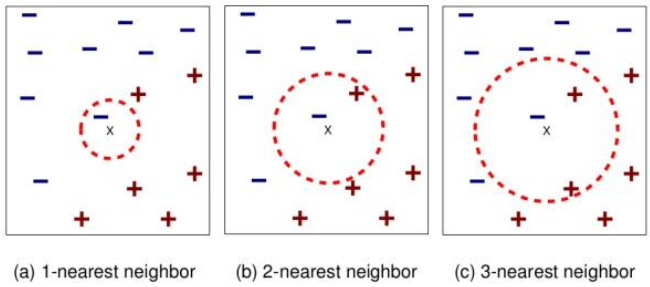


Fig 3: Classification of x point by the basis of k-Factor *In the above figure*

(a). **1-Nearest Neighbour:** in this k=1 value so the first one near neighbour for x point is was '-' class therefore x point was classified to '-' class
(b). **2-Nearst Neighbour:** in this k=2 value so the first two near neighbour of x point are '-','+' classes but the x point is nearer to '-' than '+' therefore x is belonged to '-' class
(c). **3-Nearest Neighbour:** in this k=3 value so the first three near neighbours of x point are '-' ,'+','+'classes ,so that '+'class is in more number the x is classified to '+' class

By the above example by the varying the k value the classification of data point may change. So the k will be the crucial to select [2].

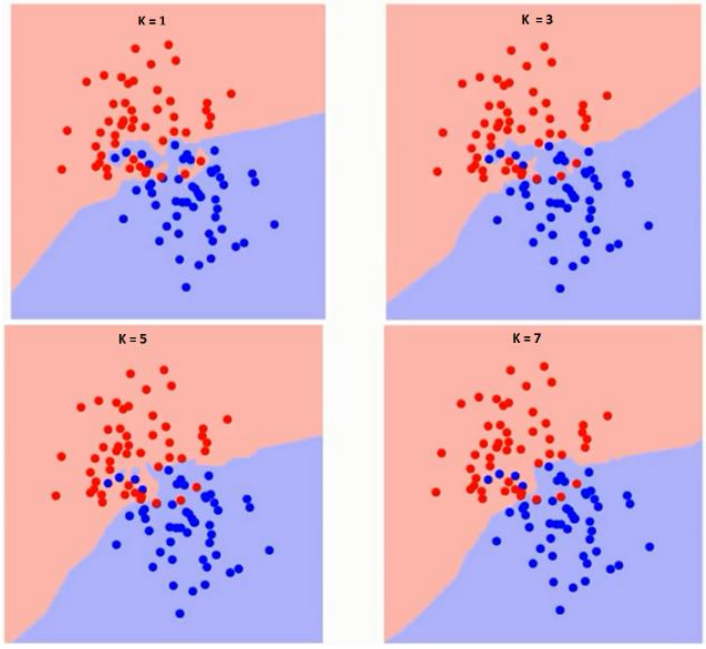


Fig 4:The distinct boundaries dividing the two classes with variable K values.

In the above Fig, it is observed that with increase in K value the boundary turns flat. On reaching the k value to infinity , it becomes to one color. Different k–Values deduces different error rates i.e training error and validation error.

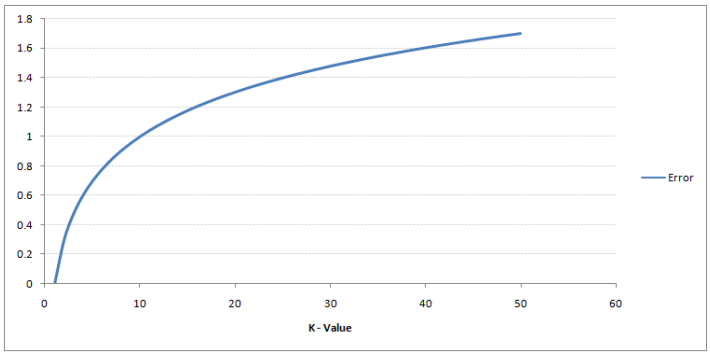


Fig 5:Training Error Rate Curve with changing K value.

In the above figure, in the training sample the error rate is zero corresponding to k value as 1. It is due to training data point is itself is closest point. The point nearer to any training data will have error rate zero at k=1, so the predictions will be accurate. And error increases when k-value increases as the points will be longer to the training data

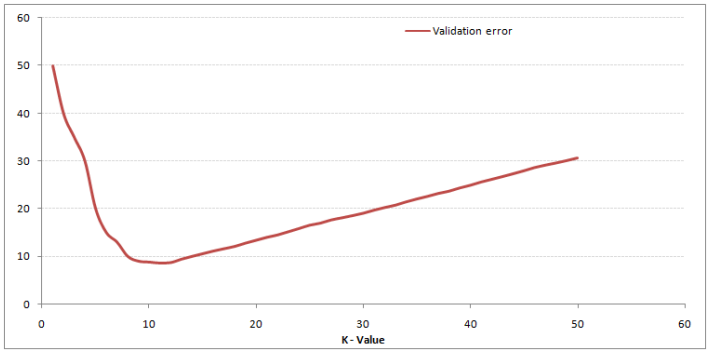


Fig 6: K-Value vs validation curve

In the above figure, the boundaries are appearing to be highest fit at K value is 1. So, error rate Reduces at beginning and reach to minimum required. It then increases after this minimum point as K increases to get the optimum K value. All predictions are based on this K value.

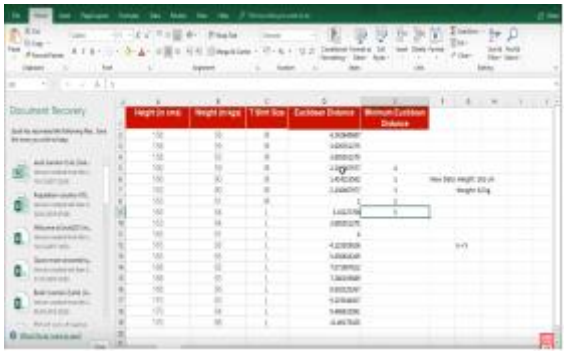


Figure 7: Example for k-factor

In the above figure, we have found the top five nearest or similar neighbours to the query data has we have taken the k value is equal to five .this was obtained by the top 5 minimum Euclidean distances between the query value and training data. We have just ranked the 5 minimum Euclidean distances so that 1 rank data will be the most similar to the data point and 2 will be the second, and so on.[9] In the top 5 ranks 4 ranks are in M-

T –shirt size(class) so the query point was classified to M-T-shirt size(class)[9]

4. A SIMPLE ALGORITHM OF K-NN

Step 1: Load the data set into train and test data set
Step 2: Find the Euclidian distances between train and test data and append into the list and sort it in ascending order
Step 3: Take the k-value, and for test case collect first k near train values in the list
Step 4: By the most accordance class in the list will be the class for the every test data
Step 5: End[5]

5. IMPLEMENTATION OF KNN

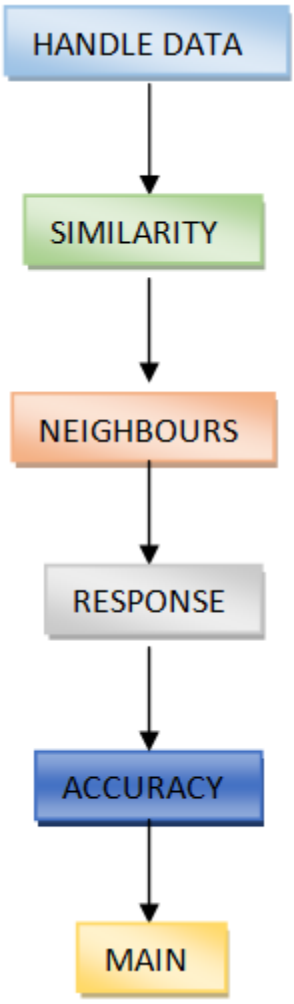
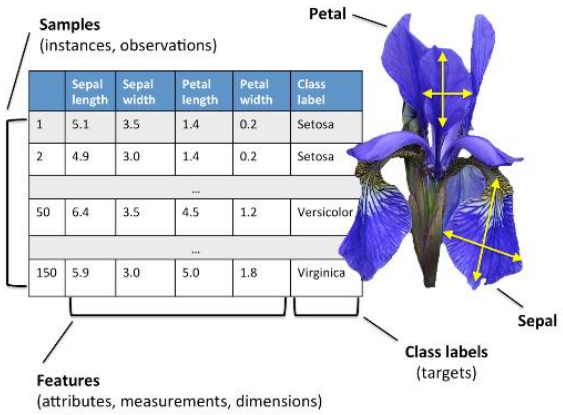


Figure 8: KNN – Steps

In the implementation of KNN there are six step to perform. We have to take the iris data set and implement in the python language. In this problem three varied species are considered and its iris flowers observations are made. Same unit of measurement is adopted for the flowers i,.e petal width, petal length, sepal width and sepal length. Setosa, Versicolor, virginica are the predicted classification of species. All occurrences termed as standard data sets. In this context, the data can be divided into training and test data set. Accuracy percentage above 95 is considered as good and 98% is treated as better.



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Fig 9: Iris Data Set

The main thing in assessing data mining model is to divide data into training and test sets. Majority of the data used is in training compared to test on dividing the data set into training and test data set. It is ensured the similarity in training and testing data to Reduce the data variance and better interpret model characteristics. The predictions are made on test data set after model is developed with training data set. Already known values for the characteristics of test data set are known prior, it is easier to found the correctness of the model. In Fig 8 Steps to execute KNN in Python.

Data Handling: The data set in CSV is opened and divided into train/test datasets.
Equivalence: Two occurrences between distances should be determined.
Neighbours: The K Most equivalent data occurrences are located.
Reaction: Evaluate a reaction form a set of data occurrences.
Accuracy: From the predictions accuracy should be calculated and summarize it.
Main: Combine all steps together.[3]

5.1 DATA HANDLING

It is required to load data file to handle the data. The data file is required to be in CSV format and should not contain header line or quotes. In the CSV module the reader function and open function are there. by the open function file is opened by the reader function data lines can be read. [3] We can import spreadsheet data and database in the python interpreter using the CSV module[13]. Import or export spreadsheets and database in Python by using the CSV module.

```
1.  import csv
2.  with open('iris.data','rb')as csvfile:
3.      lines = csv.reader(csvfile)
4.      for row in lines:
5.          print', '.join(row)
```

An accurate model can be materialized by dividing the data into training and test dataset by which predictions can be made by

KNN with test data set. It is required to convert the flower measures which are represented as strings into numbers to work further. The arrived data set is divided to training and test dataset. The dataset so arrived is divided into train and test dataset. A common proportion of trained/test dataset like 67/33. [3] To achieve this, a function can be defined as loadDataset and loads the CSV file. The random module is imported with import random which performs random number generation. In this module, random() function generated numbers between 0 and 1. The syntax random.random() is used to doing the import. [13]. the iris flowers dataset CSV file is downloaded into the local directory. The function is tested with iris dataset as follows:[3]

```
import csv
def loadDataset(filename, split, trainingSet=[] , testSet=[]):
    with open(filename, 'r') as csvfile:
        lines = csv.reader(csvfile)
        dataset = list(lines)
        for x in range(len(dataset)-1):
            for y in range(4):
                dataset[x][y] = float(dataset[x][y])
            if random.random() < split:
                trainingSet.append(dataset[x])
            else:
                testSet.append(dataset[x])
```

5.2. Similarity

Predictions are made by computing the similarity between two data occurrences. The most close K data occurrences in training data set can be found for given test dataset. The Euclidean distance measure can be used directly such that measurements of all four flowers are with similar units i.e numeric. This is derived as squared difference between the two arrays of numbers and its square root. The distance is calculated duly including the fields and in this case four attributes are added. The Euclidian distance is limited to fixed length by ignoring the final dimension. [3]

```
data1 = [2,2,2,'a']
data2 = [4,4,4,'b']
distance = euclideanDistance(data1,data2,3)
print'Distance: ' + repr(distance)
```

The following code is used to define Euclidean Distance function. Python contains the standard math module to define Euclidean Distance.

```
import math

def euclideanDistance(instance1, instance2, length):
    distance = 0
    for x in range(length):
        distance += pow((instance1[x] - instance2[x]), 2)
    return math.sqrt(distance)
```

By the following example the function can be tested with some example data

5.3. Neighbours

For similarity measure the K most similar instances are used. K most similar instances can be collected with similarity measure for a given hidden instance. For all instances this is the simple way to determine distances. The training set containing the k most similar neighbours are returned defined in the getNeighbours function. [3] An efficient function related to intrinsic operators of Python can be exported by operator module. The equivalent expression for x+y is operator.add(x,y). In special class methods and variants without leading and trailing, function names are used.

```
import operator
def getNeighbors(trainingSet, testInstance, k):
    distances = []
    length = len(testInstance)-1
    for x in range(len(trainingSet)):
        dist = euclideanDistance(testInstance, trainingSet[x], length)
        distances.append((trainingSet[x], dist))

    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(k):
        neighbors.append(distances[x][0])
    return neighbors
```

Function can be tested as follows:

```
trainSet = [[2,2,2,'a'],[4,4,4,'b']]
testInstance = [5,5,5]
k = 1
neighbors = getNeighbors(trainSet,testInstance,1)

print(neighbors)
```

5.4. Response

Majority of the identical neighbours are identified for a test instance is once completed then the next activity is to formulate for anticipated response depending on neighbours. This is done by permitting each neighbour to vote for class attribute. The function below illustrates for obtaining most voted response from neighbours[3]. Function can be tested by using some test neighbours

```
import operator
def getResponse(neighbors):
    classVotes = {}
    for x in range(len(neighbors)):
        response = neighbors[x][-1]
        if response in classVotes:
            classVotes[response] += 1
        else:
            classVotes[response] = 1
    sortedVotes = sorted(classVotes.items(), key=operator.itemgetter(1), reverse=True)
    return sortedVotes[0][0]
```

In case of a draw, one response is returned in this approach. Adopting no response or selecting and neutral random response can be handled in specific way.

5.5. ACCURACY

All the code used in KNN are available and it is required to compute the accuracy of predictions. A proportion of total correct predictions to all predictions is evaluated for arriving accuracy. The following code defines the getAccuracy function to give the accuracy as percentage duly summing the total correct predictions.[3]

```
1.  def getAccuracy(testSet, predictions):
2.      correct = 0
3.      for x in range(len(testSet)):
4.          if testSet[x][-1] == predictions[x]:
5.              correct += 1
6.      return (correct/float(len(testSet))) * 100.0
```

We can test this function with a test dataset and predictions, as follows:

```
1.  testSet = [[1,1,1,'a'],[2,2,2,'a'],[3,3,3,'b']]
2.  predictions = ['a','a','a']
3.  accuracy = getAccuracy(testSet,predictions)
4.  print(accuracy)
```

```
import csv
import random
import math
import operator

def loadDataset(filename, split, trainingSet=[] , testSet=[]):
        with open(filename, 'r') as csvfile:
            lines = csv.reader(csvfile)
            dataset = list(lines)
            for x in range(len(dataset)-1):
                for y in range(4):
                    dataset[x][y] = float(dataset[x][y])
                if random.random() < split:
                    trainingSet.append(dataset[x])
                else:
                    testSet.append(dataset[x])


def euclideanDistance(instance1, instance2, length):
        distance = 0
        for x in range(length):
                distance += pow((instance1[x] - instance2[x]), 2)
        return math.sqrt(distance)

def getNeighbors(trainingSet, testInstance, k):
        distances = []
        length = len(testInstance)-1
        for x in range(len(trainingSet)):
                dist = euclideanDistance(testInstance, trainingSet[x], length)
                distances.append((trainingSet[x], dist))
        distances.sort(key=operator.itemgetter(1))
        neighbors = []
        for x in range(k):
                neighbors.append(distances[x][0])
        return neighbors
```

5.6. MAIN

The above functions which are discussed in the algorithm are combined together and put in the main function. The main function is discussed in the below following figures

```
def getResponse(neighbors):
        classVotes = {}
        for x in range(len(neighbors)):
                response = neighbors[x][-1]
                if response in classVotes:
                        classVotes[response] += 1
                else:
                        classVotes[response] = 1
        sortedVotes = sorted(classVotes.items(), key=operator.itemgetter(1), reverse=True)
        return sortedVotes[0][0]

def getAccuracy(testSet, predictions):
        correct = 0
        for x in range(len(testSet)):
                if testSet[x][-1] == predictions[x]:
                        correct += 1
        return (correct/float(len(testSet))) * 100.0
def main():
        # prepare data
        trainingSet=[]
        testSet=[]
        split = 0.67
        loadDataset('iris.data', split, trainingSet, testSet)
        print( 'Train set: ' + repr(len(trainingSet)))
        print( 'Test set: ' + repr(len(testSet)))
        # generate predictions
        predictions=[]
        k = 3
        for x in range(len(testSet)):
                neighbors = getNeighbors(trainingSet, testSet[x], k)
                result = getResponse(neighbors)
                predictions.append(result)
                print('> predicted=' + repr(result) + ', actual=' + repr(testSet[x][-1]))
        accuracy = getAccuracy(testSet, predictions)
        print('Accuracy: ' + repr(accuracy) + '%')
main()
```

6. RESULTS

By the discussed example .the predictions was compared to the actual class by using test set after interpreting the accuracy will be printed. in this case the accuracy was 98 percent.[3]...

```
>predicted='Iris-virginica',actual='Iris-virginica'
>predicted='Iris-virginica',actual='Iris-virginica'
>predicted='Iris-virginica',actual='Iris-virginica'
>predicted='Iris-virginica',actual='Iris-virginica'
>predicted='Iris-virginica',actual='Iris-virginica'
Accuracy: 98.0392156862745%
```

As the k value is three we get the top three nearest class labels . predicted classes come from the getResponse function and actual was come from test set. By this algorithm we can classify the test data set by using similarity measures. KNN give competitive results even with the simple classification program.

7. INDUSTRIAL APPLICATION OF KNN

Industrial application of KNN algorithm are vast and most significant one is classification of textual documents. The huge amount of documents generated in the current time needs the attention for automated textual document classification into predefined categories. The KNN algorithm is best useful in this scenario. [4] The KNN algorithm can be implemented for analysis of text and the frequent terms in textual documents. The terms are tabulated in the matrix form with occurrence in different documents.

Term	Doc1	Doc2	Doc3	Doc4	Doc5
Competition	0	1	0	0	0
Data	0	1	1	1	0
eGov	1	0	0	0	0
Fund	1	0	0	0	0
Government	0	0	0	1	1
Guarantee	1	0	0	0	0
Information	0	0	1	1	0
Infrastructure	0	0	0	1	0
Portal	0	0	1	0	1
India	0	0	0	1	1

Table 1: Term-Document Matrix from the document conversion

The above table represents a term as a 'Term-Document Matrix'. A Term-Document Matrix is represented by a point or a vector, and documents will be sort out, and converted Into a term document matrix[3].

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

The documents can be arrived which are relevant to the query request. For example, the documents in the portal to give the information of draft regulations on policy matters launched in India. The query can be represented as a request vector. Now presume that we need documents to be discover that are pertinent to the request "The portal for open discussion of draft regulation launched in India" this should be constructed as a term document matrix by constitute by a vector:[9]

$$q = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \in R^{10}.$$

The query is treated as a document. The classification of text can be modeled mathematically as a matrix to find the columns of document matrix which are closed to the request vector matrix.

7.1 APPLICATIONS OF KNN

Major shopping sites implement KNN algorithm to give feature similarity of their products. The shopping sites guide the customer to choose the best product modal of their intended features. In many e-commerce site like Amazon, the products are displayed based on the choice of features of customer. The site recommends the similar featured products which are bought by other customers previously with similar shopping behavior [10].

Figure 10: An application of KNN in e-commerce

The above figure shows an example for implementing KNN in Amazon, a prominent site of e-commerce. Amazon tracks data of every purchase made, and learns from the instances obtained. When a customer searches for models with similar features, suggestions from the learnt instances are displayed to the customer. The KNN Algorithm is best implemented in banking system to decide technically for approval of the loan to customer. The banking system processes loans to huge number of applicants. The dataset contains the data of the applicant i.e personal data and banking data. The data can be used to approve the loan of the applicant. This can be solved by using

KNN algorithm by dividing into two category classes. Those are

1. Loan Approved
2. Loan Disapproved.

The dataset is to be imported and it is required to analyze the data set structure. Some variable are very important in the structure of dataset which helps to decide the approval of the loan. The complexity of the dataset can be Reduced by eliminating the irrelevant variables from the dataset. The significant modal is arrived and normalized for a optimized output. The data normalized is stored into a subset variable. Thus the normalized dataset converted into training and test dataset. The trained dataset is observed with test dataset and the number of observations to find the optimized value i.e K of KNN Modal.

8. CONCLUSION

In the past two decades, with an exponential growth in data storage and huge data accumulated, an intelligent analysis of data is very important of the current scenario. KNN Algorithm and Instance based learning is greatly helps in classification of data on grounds of similarity. Most of the present industries are using the KNN for classification algorithm. In this paper an overview of KNN algorithm is presented with instance based learning system. The classification technique in K-Nearest Neighbour is evolved on the necessity to distinct analysis when authentic variable are not known and difficult to find out. The data is classified into two categories i.e trained data set and test data set and the KNN algorithm is implemented by finding K nearest neighbour between trained and test dataset by similarity distance measure. Regression and classification setting can be achieved by KNN algorithm.

REFERENCES

[1] Types of classification algorithms in Machine Learning – Medium https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14

[2] Introduction to KNN, K-Nearest Neighbors : SimplifiedAnalytiCSVidyahttps://www.analytiCSVidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

[3] Tutorial To Implement k-Nearest Neighbors in Python From Scratchhttps://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/

[4] Aimanmoldagulavo,Rosnafisah Sulaiman"Using KNN algorithm for classification of textual documents", international journal on information technology ,2017

[5] K Nearest Neighbours—Introduction to Machine Learning Algorithmshttps://towardsdatascience.com/k-nearest neighbours-introduction-to-machine-learning-algorithms-18e7ce3d802a

[6] Machine Learning basics with knn algorithmhttps://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

[7] Classification of data into classes https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

[8] K Nearest Neighbors – Classificationhttps://www.saedsayad.com/k_nearest_neighbors.htm

[9] Example for k-factorhttps://www.edureka.co/blog/k-nearest-neighbors-algorithm/

[10] Applicationshttps://www.slideserve.com/EdurekaIN/knn-algorithm-using-python-how-knn-algorithm-works-python-data-science-training-edureka-powerpoint-ppt-resentation

[11] Distant methods https://dataaspirant.com/2015/04/11/five-most-popular-similarity-measures-implementation-in-python/[12] train and test data sethttps://docs.microsoft.com/en-us/analysis-services/data-mining/training-and-testing-data-sets

[12] Module in pythonhttps://docs.python.org/3/py-modindex.html

[13] instance based learninghttps://en.wikipedia.org/wiki/Instance-based_learning

[14] lazy learninghttps://en.wikipedia.org/wiki/Lazy_learning