


GeeksforGeeks

A computer science portal for geeks



COURSES

Login

HIRE WITH US

Shortest Superstring Problem | Set 2 (Using Set Cover)

Implementation of K Nearest Neighbors

Best Python libraries for Machine Learning

Neural Networks | A beginners guide

ML | Stochastic Gradient Descent (SGD)

DBMS | OLAP Operations

Generative Adversarial Network (GAN)

ML | Classification vs Regression


Game Playing in Artificial Intelligence

ML | Mini-Batch Gradient Descent with Python

Q-Learning in Python

Implement your own word2vec(skip-gram) model in Python

Python | Tokenize



text using TextBlob

Deep Learning |
Introduction to
Long Short Term
Memory

ML | Using SVM to
perform
classification on a
non-linear dataset

ML | Locally
weighted Linear
Regression

Applying
Multinomial Naive
Bayes to NLP
Problems

ML | Monte Carlo
Tree Search
(MCTS)

Data Structure to
Design a special
social network

ML | Natural
Language
Processing using
Deep Learning

MATLAB | Display
histogram of a
grayscale Image

Audio processing
using Pydub and
Google
speechRecognition
API

LSB based Image
steganography
using MATLAB

Elasticsearch
Search Engine | An
introduction

Python | Speech
recognition on
large audio files

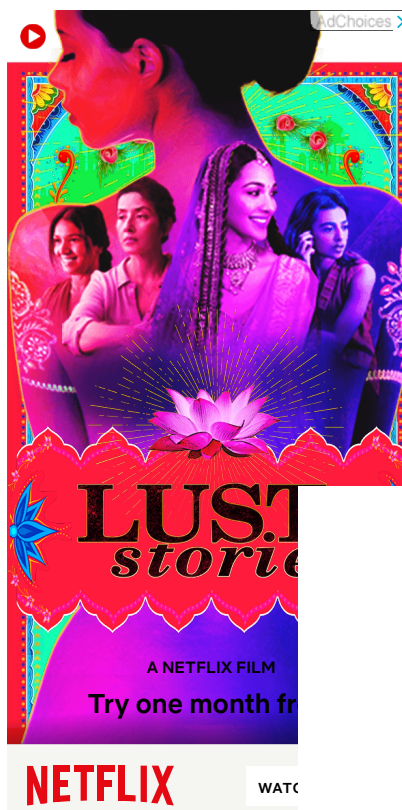
Draw Austria flag
using Matlab

Introduction to
Signals and
Systems:
Properties of
systems

Draw Bangladesh
Flag Using Matlab

Text extraction
from image using
LSB based
steganography

MATLAB | Convert
video into slow
motion



K-Nearest Neighbours

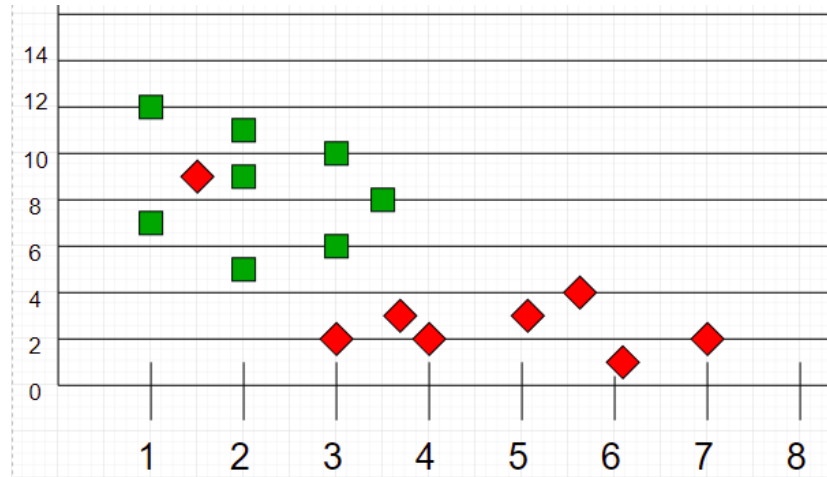
K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detec-

tion.

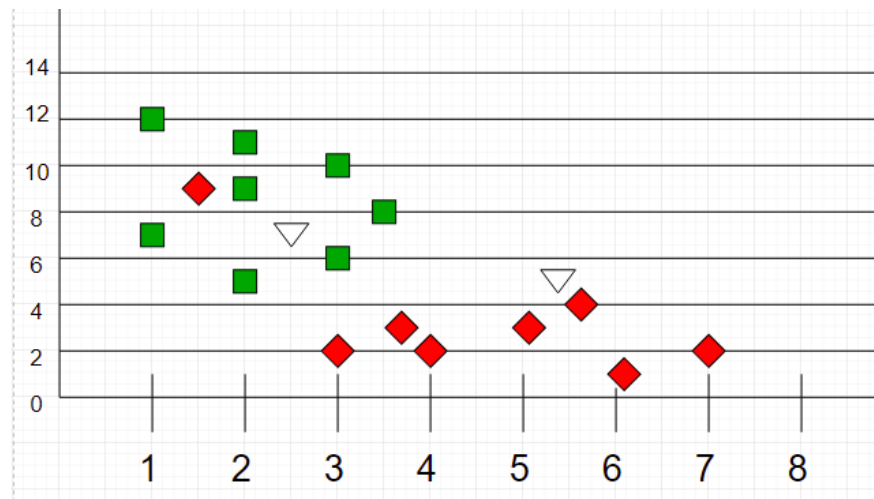
It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as **GMM**, which assume a Gaussian distribution of the given data).

We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute.

As an example, consider the following table of data points containing two features:



Now, given another set of data points (also called testing data), allocate these points a group by analyzing the training set. Note that the unclassified points are marked as 'White'.



Intuition

If we plot these points on a graph, we may be able to locate some clusters, or groups. Now, given an unclassified point, we can assign it to a group by observing what group its nearest neighbours belong to. This means, a point close to a cluster of points classified as 'Red' has a higher probability of getting classified as 'Red'.

Intuitively, we can see that the first point (2.5, 7) should be classified as 'Green' and the second point (5.5, 4.5) should be classified as 'Red'.

Algorithm


Let m be the number of training data samples. Let p be an unknown point.

1. Store the training samples in an array of data points `arr[]`. This means each element of this array represents a tuple (x, y) .
2. for $i=0$ to m :

Calculate Euclidean distance $d(arr[i], p)$.

3. Make set S of K smallest distances obtained. Each of these distances correspond to an already classified data point.
4. Return the majority label among S .

Recommended: Please try your approach on {IDE} first, before moving on to the solution.

 **APPLIED COURSE**

GATECS 2020
Blended Course

> Learn from IISC/IIT's Alumni





> 2 LIVE sessions with 2 practice tests per week

K can be kept as an odd number so that we can calculate a clear majority in the case where only two groups are possible (e.g. Red/Blue). With increasing K , we get smoother, more defined boundaries across different classifications. Also, the accuracy of the above classifier increases as we increase the number of data points in the training set.

Example Program

Assume 0 and 1 as the two classifiers (groups).

C/C++



```
// C++ program to find groups of unknown
// Points using K nearest neighbour algorithm.
#include <bits/stdc++.h>
using namespace std;

struct Point
{
    int val;        // Group of point
    double x, y;    // Co-ordinate of point
    double distance; // Distance from test point
};

// Used to sort an array of points by increasing
// order of distance
bool comparison(Point a, Point b)
{
    return (a.distance < b.distance);
}

// This function finds classification of point p using
// k nearest neighbour algorithm. It assumes only two
// groups and returns 0 if p belongs to group 0, else
// 1 (belongs to group 1).
int classifyAPoint(Point arr[], int n, int k, Point p)
{
    // Fill distances of all points from p
    for (int i = 0; i < n; i++)
        arr[i].distance =
            sqrt((arr[i].x - p.x) * (arr[i].x - p.x) +
                (arr[i].y - p.y) * (arr[i].y - p.y));

    // Sort the Points by distance from p
    sort(arr, arr+n, comparison);

    // Now consider the first k elements and only
    // two groups
    int freq1 = 0;    // Frequency of group 0
    int freq2 = 0;    // Frequency of group 1
    for (int i = 0; i < k; i++)
    {
        if (arr[i].val == 0)
            freq1++;
        else if (arr[i].val == 1)
            freq2++;
    }

    return (freq1 > freq2 ? 0 : 1);
}

// Driver code
int main()
{
    int n = 17; // Number of data points
    Point arr[n];

    arr[0].x = 1;
    arr[0].y = 12;
    arr[0].val = 0;

    arr[1].x = 2;
    arr[1].y = 5;
    arr[1].val = 0;

    arr[2].x = 5;
    arr[2].y = 3;
    arr[2].val = 1;

    arr[3].x = 3;
    arr[3].y = 2;
    arr[3].val = 1;

    arr[4].x = 3;
    arr[4].y = 6;
    arr[4].val = 0;

    arr[5].x = 1.5;
```

Python

```

# Python3 program to find groups of unknown
# Points using K nearest neighbour algorithm.

import math

def classifyAPoint(points,p,k=3):
    '''
    This function finds classification of p using
    k nearest neighbour algorithm. It assumes only two
    groups and returns 0 if p belongs to group 0, else
    1 (belongs to group 1).

    Parameters -
        points : Dictionary of training points having two keys - 0 and 1
                  Each key have a list of training data points belong to that

        p : A tuple ,test data point of form (x,y)

        k : number of nearest neighbour to consider, default is 3
    '''

    distance=[]
    for group in points:
        for feature in points[group]:

            #calculate the euclidean distance of p from training points
            euclidean_distance = math.sqrt((feature[0]-p[0])**2 +(feature[1]-p[1])**2)

            # Add a tuple of form (distance,group) in the distance list
            distance.append((euclidean_distance,group))

    # sort the distance list in ascending order
    # and select first k distances
    distance = sorted(distance)[:k]

    freq1 = 0 #frequency of group 0
    freq2 = 0 #frequency of group 1

    for d in distance:
        if d[1] == 0:
            freq1 += 1
        elif d[1] == 1:
            freq2 += 1

    return 0 if freq1>freq2 else 1

# driver function
def main():

    # Dictionary of training points having two keys - 0 and 1
    # key 0 have points belong to class 0
    # key 1 have points belong to class 1

    points = {0:[(1,12),(2,5),(3,6),(3,10),(3.5,8),(2,11),(2,9),(1,7)],
              1:[(5,3),(3,2),(1.5,9),(7,2),(6,1),(3.8,1),(5.6,4),(4,2),(2,5)]}

    # testing point p(x,y)
    p = (2.5,7)

    # Number of neighbours
    k = 3

    print("The value classified to unknown point is: {}".format(classifyAPoint(points,p,k)))

if __name__ == '__main__':
    main()

# This code is contributed by Atul Kumar (www.fb.com/atul.kr.007)

```

Output:

The value classified to unknown point is 0.



This article is contributed by **Anannya Uberoi**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

[Maximum previous and next element product](#)

[ML | Auto-Encoders](#)

[Explanation of Fundamental Functions involved in A3C algorithm](#)

[Data with Hadoop](#)

[Building an Auto-Encoder using Keras](#)

[Crossover in Genetic Algorithm](#)

[ML | Log Loss and Mean Squared Error](#)

[Maximum number of nodes which can be reached from each node in a graph.](#)

[Rearrange characters in a string such that no two adjacent are same using hashing](#)

[Cyber Security in Cloud computing](#)

[Pandas Built-in Data Visualization | ML](#)

[Python | Named Entity Recognition \(NER\) using spaCy](#)

[Encoding Methods in Genetic Algorithm](#)

[Implementing ANDNOT Gate using Adaline Network](#)

Improved By : NIRBHAYKUMAR1

Article Tags : [Advanced Computer Subject](#) [Algorithms](#) [Machine Learning](#) [Directi](#)**Practice Tags :** [Directi](#) [Algorithms](#) [Machine Learning](#)

4

☐ To-do ☐ Done**2.8**Based on **13** vote(s)[Feedback/ Suggest Improvement](#)[Add Notes](#)[Improve Article](#)Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.[Load Comments](#)

GeeksforGeeks
A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

[About Us](#)
[Careers](#)
[Privacy Policy](#)
[Contact Us](#)

LEARN

[Algorithms](#)
[Data Structures](#)
[Languages](#)
[CS Subjects](#)
[Video Tutorials](#)

PRACTICE

[Courses](#)
[Company-wise](#)
[Topic-wise](#)
[How to begin?](#)

CONTRIBUTE

[Write an Article](#)
[Write Interview Experience](#)
[Internships](#)
[Videos](#)

@geeksforgeeks, Some rights reserved