ESKISEHIR OSMANGAZI UNIVERSITY
ELECTRICAL AND ELECTRONICS ENGINEERING
2021-2022 OBJECT ORIENTED PROGRAMMING

K-Nearest Neighbours

RAMAZAN AKKULAK

TABLE OF CONTENTS

LIST OF FIGURES 2
LIST OF TABLES 3
ABSTRACT 4
INTRODUCTION 4
1.1 K-Means Algorithm 5
1.1.1 How does the K-Means algorithm work? 5
2.1 The Coding Hierarchy and Running 5
3.1 Result 6
4.1 References 7

LIST OF FIGURES

Figure 1. KNN algorithm result 4
Figure 2. KNN Sample 1 example 6
Figure 3. KNN Sample 2 example 7

LIST OF TABLES

Table 1. MatPlotLib setups steps. (1)Github link (2) Python lib setup, (3) Example run 6
Table 2. Terminal complier CMake 6
Table 3. Output command 6

ABSTRACT

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on the Supervised Learning technique. The K-NN algorithm assumes the similarity between the new state/data and the existing states and places the new state in the category most similar to the existing categories. The K-NN algorithm stores all available data and classifies a new data point based on similarity. This means that when new data appears, it can easily be classified into a well pack category using the K-NN algorithm. K-NN algorithm can be used for Regression as well as Classification, but it is mostly used for Classification problems. K-NN is a non-parametric algorithm, meaning it doesn't make any assumptions on the underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately, but instead stores the dataset and performs an operation on the dataset at the time of classification. The KNN algorithm only stores the data set during the training phase and when it receives new data, it classifies this data in a category that is very similar to the new data. Let's say we have an image of a creature that looks like a cat and a dog, but we want to know if it's a cat or a dog. For this identification, we can use the KNN algorithm as it works on the similarity measure. Our KNN model will find features similar to cat and dog images in the new dataset and place it in the cat or dog category based on the most similar features.

INTRODUCTION

The KNN algorithm assumes that similar things exist nearby. In other words, similar things are close together.

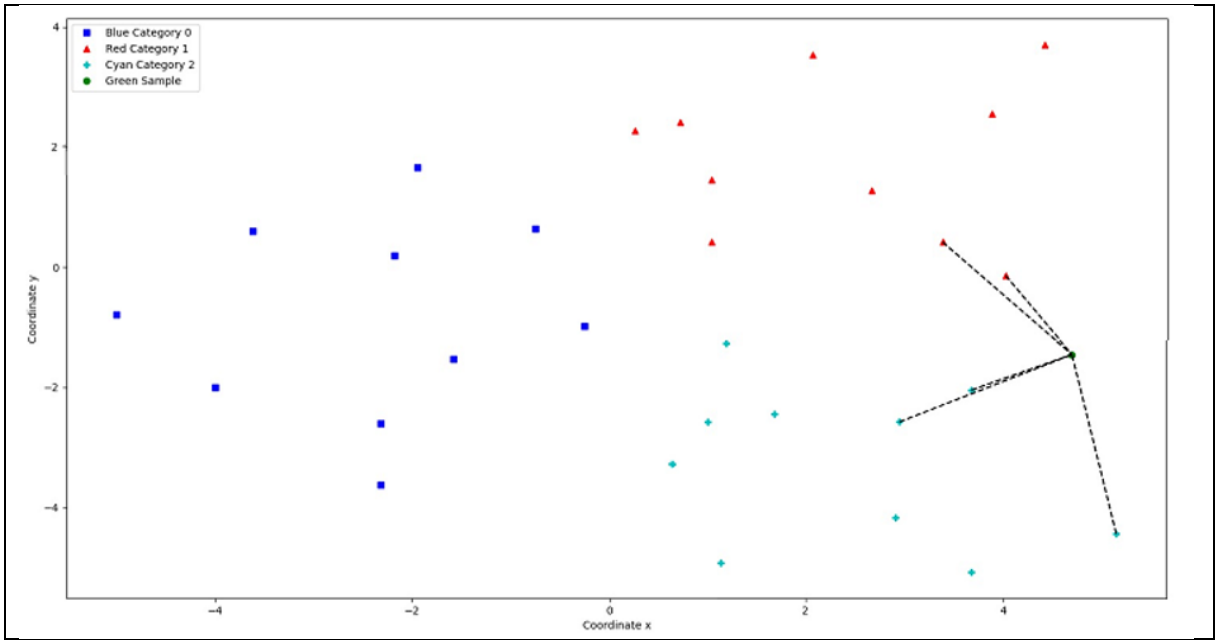

Figure 1. *KNN algorithm result*

Note that in the image above, similar data points are often close together. The KNN algorithm relies on this assumption being accurate enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or proximity) with some of the math we may have learned in our childhood - calculating the distance between points on a graph.

There are other ways to calculate distance and one way may be preferred depending on the problem we are solving. However, straight line distance (also called Euclidean distance) is a popular and familiar choice.

1.1K-Means Algorithm

The k-nearest neighbors algorithm (k-NN) was first developed by Evelyn Fix and Joseph Hodges in 1951[1] and later by Thomas Cover[2]. It is used for classification and regression. In both cases, the input consists of the k closest training examples in a data set. The output depends on whether the k-NN is used for classification or regression.

In k-NN classification, the output is a class membership. An object is classified by the plural of its neighbors, and the object is assigned the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, the object is simply expected to be assigned to that nearest neighbor's class.

1.1.1 How does the K-Means algorithm work?

K-NN study can be explained on the basis of the following algorithm:

Step-1: Choose the K number of neighbors

Step-2: Calculate the Euclidean distance of K neighbors

Step-3: According to the calculated Euclidean distance, get the K nearest neighbors.

Step-4: Count the number of data points in each category among these k neighbors.

Step-5: Assign the new data points to the category with the maximum number of neighbors.

Step-6: Our model is ready.

2.1 The Coding Hierarchy and Running

The KNN algorithm I designed consists of 2 classes and one inherit class. These classes are as follows:

1. Data Class (base)
2. KNN Class (inherit)
3. Graph Class (base)
4. MatPlotLib.cpp

The study was developed on VS code ide. There is also the matplotlib.cpp function developed for plotting over the matplotlib library developed for python. The installation of this file is simple. This file can be accessed via github specified in table 2 step 1[3]. This should be copied to the part where the project files are.The following guidelines should be followed to run this run.

(1)	https://github.com/lava/matplotlib-cpp
(2)	sudo apt-get install python-matplotlib python**X**-numpy python**X**.**X**-dev
(3)	g++ example.cpp -I/usr/include/python**X**.**X** -lpython**X**.**X**

Table 1. *MatPlotLib setups steps. (1)Github link (2) Python lib setup, (3) Example run*

The above 2 steps are performed. Before this process is performed, the python version installed on the Linux (ubuntu) operating system is important in places marked as **X** in bold. The work was run on python3.8. Then run one of the examples in the file installed from github to test the accuracy of this process. After doing this process, you can go to the directory where you are working via the terminal on the visual code or via the ubuntu terminal with the help of the "cd" command. After navigating to the required directory:

```
g++ main.cpp Data.cpp KNN.cpp Graph.cpp Prim.cpp -std=c++11 -I/usr/include/python3.8 -lpython3.8
```

Table 2. Terminal complier CMake

This command is typed into the terminal to view the results. The video of the study is given in [4].

```
./a.out
```

Table 3. *Output command*

3.1 Result

1. Sample 1:

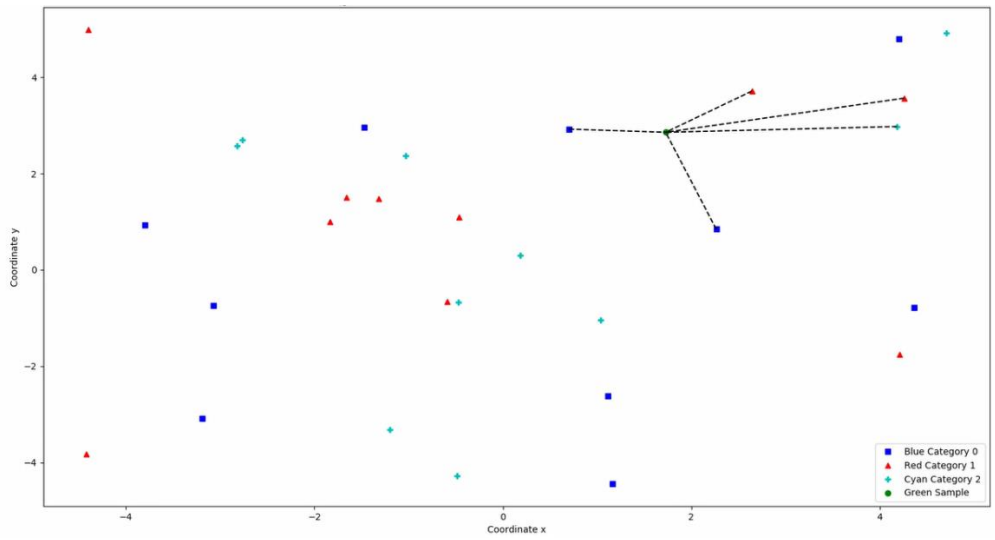

Figure 2. *KNN Sample 1 example*

The category result of sample first is 1.

2. Sample 2:

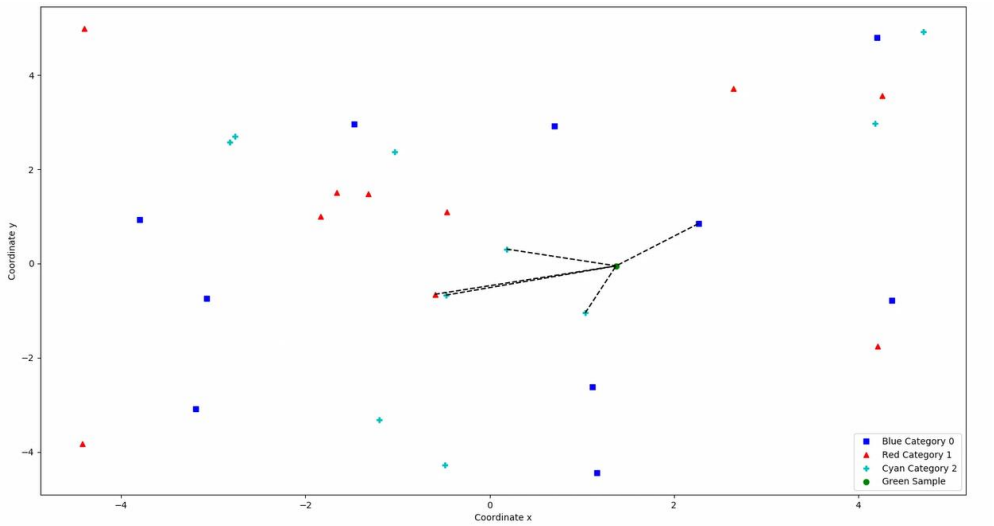

Figure 3. KNN Sample 2 example

The category result of sample first is 2.

The number of samples may change according to the user's request. But the point data does not change.

4.1 References

[1] Fix, E., & Hodges, J. L. (1989). Discriminatory analysis. Nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, *57*(3), 238-247.

[2] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175-185.

[3] https://github.com/lava/matplotlib-cpp

[4] https://youtu.be/fES9V6NLvkc

[5] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm