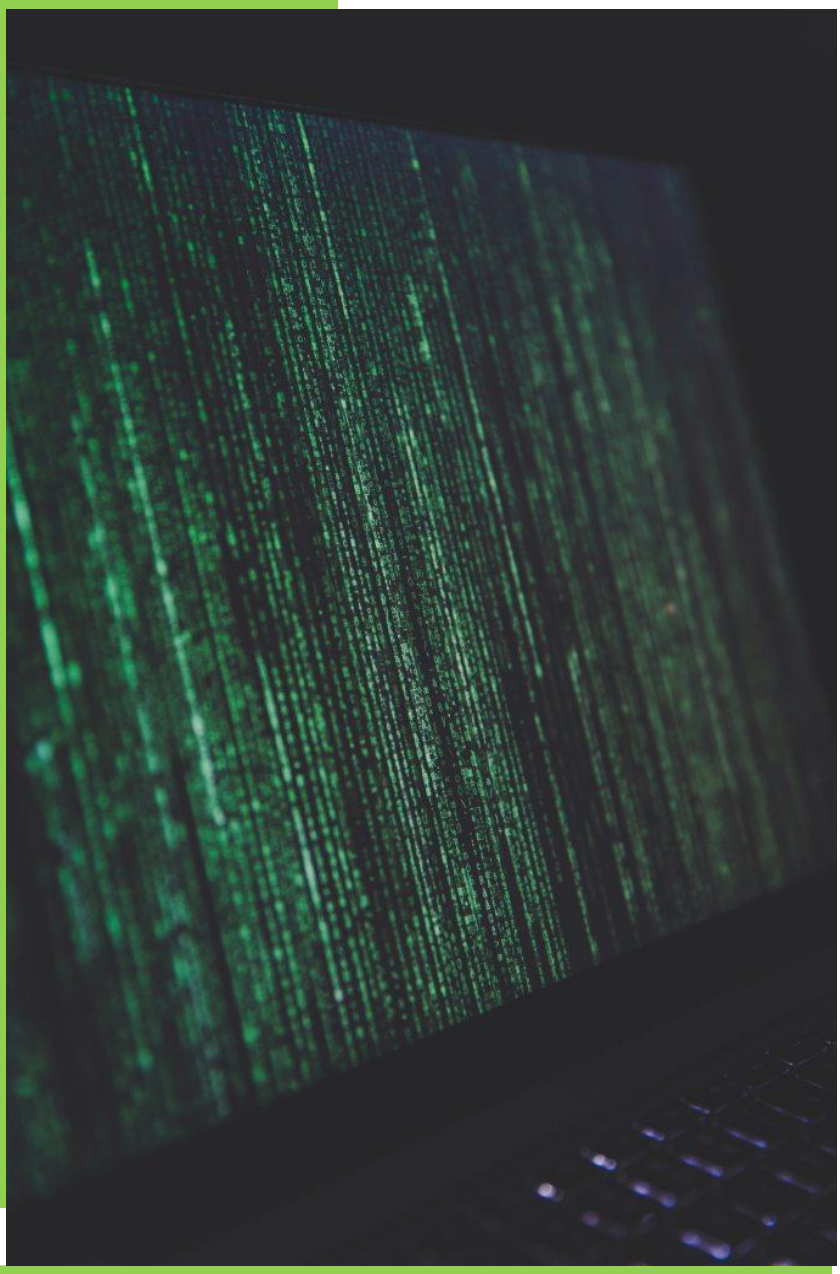
Smart Systems and computational Intelligence.

Project Report



Name : Amany Fathy Mohamed Yaseen

ID : 2022565968

Department : AI

K-Nearest Neighbors (KNN) on Iris Dataset

CONTENTS

ABOUT THE ALGORITHM

K-Nearest Neighbors

The k-Nearest Neighbor algorithm is one of the simplest algorithms based on the supervised learning technique.

Stores all the available data and classifies a new data point based on the similarity.

Mostly used for Classification Problems.

Work Steps

1. Select number (K) of the neighbors.
2. Calculate the Euclidean distance of (K) number of neighbors
3. Take the K nearest neighbors as per the calculated Euclidean distance.
4. Among these k neighbors, count the number of data points in each category.
5. Assign the new data points to the category for which the number of neighbors is maximum.

The K value

Using odd numbers, fit a KNN classifier for each number.

The optimal K value usually found is the square root of N, where N denotes the total number of samples in the training dataset.

Also, domain knowledge is very useful in choosing the K value.

ABOUT THE DATASET

Iris Dataset

The IRIS dataset is a collection of 150 records (rows) of Iris flowers. Each record (row) includes four attributes / features (columns) : the petal length and width, and the sepal length and width.

The goal of this dataset is to predict the type of Iris flower based on the given features.

There are three types (classes) of Iris flowers in the dataset represented by 50 records each: Iris setosa, Iris virginica, and Iris versicolor.

Target is 0 , 1 or 2.

How the algorithm works ?

The algorithm finds the Euclidean distance between the input points and the dataset points and makes predictions as to which class will the input value (flower) belongs to.

Categorizes the input parameters (Sepal length, width || Petal length, width) into the corresponding classes (Setosa, Versicolor, and Virginica)

CODE EXPLANATION

Import Libraries

The Iris dataset and k-nearest neighbor classifier have been imported from the Scikit-learn library.

- Import matplotlib for plotting the data

Load the Dataset

- We load the iris dataset from the library sklearn.datasets

Details about the Data

- I printed the four features' names of the data (Sepal length(cm), sepal width(cm), petal length(cm), petal width(cm)), which are the columns of the dataset that the prediction is based on them.
- Print the target which is a value of 0,1 or 2, and each target has a name, that is the target names (0: Setosa, 1: Versicolor, 2: Virginica)

Data Plotting

I plotted the data using a scatter plot and plotted four plots for the three iris flowers (features).

- The first plot was a comparison of the sepal width and petal length for the three iris flowers (features).
- The second plot was a comparison of the sepal length and petal width for the three iris flowers (features).
- The third plot was a comparison of the sepal length and sepal width for the three iris flowers (features).
- The fourth plot was a comparison of the petal length and petal width for the three iris flowers (features).

DATA SPLITTING

I Assigned the data -records of all four features- that is an array of a shape (150,4) to a variable f, where 150 is the number of rows and 4 is the number of columns (features).
I Assigned the data target to a variable (targ), which is a 1D array of 150 rows. The target is of values 0,1, or 2.

Split to train and test

We split the data to train and test data so we can then apply the algorithms on the training data and then predict using the test data.

First, I tried to import the function train_test_split to split the data from the library sklearn.cross_validation, but I faced a problem concerning the module.
So I got to substitute it with model_selection.

The function took the parameters of the data and target, splitting each of them into a size of 80% for training and the rest 20% for testing.

- The training data became of shape (120,4) and the test data is (30,4).
- The training target data became of shape(120,) and the test is (30,).

APPLYING THE ALGORITHM

First, I applied the algorithm with k neighbors equals 3 and computed the accuracy score.

But, as we don't know which k is optimal, putting a range helps us try different values of k and choose the best one that returns the best accuracy.

I put a range for the number of neighbors so I can compute the accuracy for each neighbor number. K neighbors' range is (1,30).

Then I Declared a dictionary and list that are empty to store the output scores of each iteration.

- The Dictionary is to store the accuracy score in a pair of keys and values.
- The key is the value of the k neighbor, and the value is the accuracy score's value of that key (k value).

For Loop

Loop iterates in the range of (1,30), for each value in that range, the KNN classifier is applied with a number of neighbors equal to that value (k value in the range).

For each k value in the range :

- Fit the model from the training datasets.
- Predict the class labels for the provided test data.
- Compute accuracy classification score using metrics.accuracy_score imported from sklearn, given the predicted labels and labels of test data.
- Assign the accuracy score to the dictionary paired with its key value.
- Append the accuracy score to the scores list.

Relationship between k value and testing accuracy

Visualize how the accuracy score is for each different value of k neighbors.

The plot shows the relationship between each k neighbor value and its corresponding accuracy score.

I observed from the plot :

- That the accuracy score starts with a low value for k's values 1 and 2.
- Then it gets high and stays in this state for a range of k values until k equals 19, then it gets low again and goes for high and low until k of value 23.
- At k = 23, the score went high and remained in this state until k = 27, it went down again and stayed in this state for the rest values.

KNN ON THE DATA

I applied the KNN classifier for a value of neighbors equals 2.
Fit the model from the dataset (data, target).

Predict on a new data

Declare a classes dictionary with key-value pairs where :

- Keys are the targets of the dataset of values 0,1 or 2.
- Values are the corresponding target names of the target values (Setosa, Versicolor, Virginica).

I declared new data (new values for the four features) of a shape (3,4).
Predict the class label for this new data, where the predicted output which is a value of 0,1 or 2, goes as an input to the dictionary of classes to predict the output class label, whether it is Setosa, Versicolor, or virginica.

REFERENCES

sklearn.datasets.load_iris
https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html

Python - Creating Scatter Plot with IRIS Dataset
Follow meAjitesh KumarI have been recently working in the area of Data analytics including Data Science and Machine Learning / Deep Learning. I am also passionate about different technologies including programming languages such as Java/JEE
https://vitalflux.com/python-creating-scatter-plot-with-iris-dataset/

matplotlib.pyplot.scatter#
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html#

matplotlib.pyplot.legend#
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.legend.html#matplotlib.pyplot.legend

How to Add Legend to Scatterplot Colored by a Variable with Matplotlib in Python
Datavizpyr
https://datavizpyr.com/add-legend-to-scatterplot-colored-by-a-variable-with-matplotlib-in-python/

ImportError: No module named sklearn.cross_validation
arthurcklarthurckl 5 et al.
https://stackoverflow.com/questions/30667525/importerror-no-module-named-sklearn-cross-validation

sklearn.model_selection.train_test_split
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

sklearn.neighbors.KNeighborsClassifier
https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

sklearn.metrics.accuracy_score
https://scikit-

learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html#sklearn.metrics.accuracy_s

core

Creating multiple subplots using plt.subplots#
https://matplotlib.org/stable/gallery/subplots_axes_and_figures/subplots_demo.html

Applying-Knn-on-IRIS-Dataset/iris_KNN.ipynb at main · zayn-1/Applying-Knn-on-IRIS-Dataset
zayn-1
https://github.com/zayn-1/Applying-Knn-on-IRIS-Dataset/blob/main/iris_KNN.ipynb

Iris_dataset-kNN/Iris_Dataset__kNN.ipynb at master · amyy28/Iris_dataset-kNN
amyy28
https://github.com/amyy28/Iris_dataset-kNN/blob/master/Iris_Dataset__kNN.ipynb

Scientific Visualization Using Python
https://education.molssi.org/python-visualization/matplotlib/subplots.html