University of Tehran

Faculty of Modern Sciences and Technologies

KNN

name and name family	Ramtin Kabiri

Question report list

Definitions and preliminaries

Algorithm classificationK-based nearest neighbor The examined data interval is done with the labeled data in the training dataset.

In this problem, the Euclidean distance is used as a measure of distance. Euclidean distance formula for

DataThe specification is as follows: I with the number of b and a

$$d=(a,b)=sqrt(sum(a_k-(0-I)=k)_2)-b_k)_2$$

In this algorithmup to the nearest neighbors of the data k We find the examined item, then we count their tags and add any tag that was more to the examined item data.

Check the code

The code sections are as follows:

1. Euclidean distance calculation function: This distance function is two DataCalculates b and a

```
                                                    ### distance calculator: {
                                                        :)b,a(distancedef
                                                            ) a(len=l
                                                            0=sum_out
                                                    :)l,0(rangeincfor
                                    ) 2,)]c[b- ]c[a((pow.math=+sum_out

                                                    ) sum_out(sqrt.math=dist
                                                            distreturn

                                                                    # #check : {
                                    # a_test = np.array([1, 3, 6, 2] )
                                    # b_test = np.array([5, 2, 0, 1] )
                                    # print(distance(a_test, b_test ) )
                                                                        } #
                                                                    } ###
```

2. k-nearest neighbor algorithm: this function is based on A training dataset and the number of desired neighborhoods

It labels the data of the test dataset and calculates its labeling error value.

```
                                                    ### k-nearest neighbor : {
                                                      :)k,test,train(knndef
                                                    ) test(shape.n.p=q_test ]0[
                                                        q_test=i_test
                                                              ]1[q_test=J
                                                    ) train(shape.n.p=q_train ]0[
                                                        q_train=i_train
                                                                        0=t
                                                                        0=f

                                                :)i_test,0(rangeinrowfor
                                                   ])1-J(:0,row[test=a ]1- ,row[
                                                   test=result
                                                            ][ =dist_list
                                          :)i_train,0(rangeinindexfor
                                             ])1-J(:0,index[train=b
                                                  ) b,a(distance=dist
                                  ) ) index,))dist(float(((append.dist_list

                                                             ][ =knn_list
                                                :)k,0(rangeink_neighborfor
                                             ) ) dist_list(min(append.knn_list
                                          ) ) dist_list(min(remove.dist_list

                                                             ][ =neighbors
                                                   :)k,0(rangeincolumnfor
                                                      ) 0(pop.knn_list=w
                                                ) ) ]1[w(float(append.neighbors


                                                          ][ =neighbor_result
                                                   :)k,0(rangeincolumnfor
                                                     ) 0(pop.neighbors=e
                                                ]1- ,)e(int[train=nbresult
                                  ) ) nbresult(float(append.neighbor_result

                                             ) ) 0(float(count.neighbor_result=r0
                                             ) ) 1(float(count.neighbor_result=r1
                                             ) ) 2(float(count.neighbor_result=r2
                                             ) ) 3(float(count.neighbor_result=r3
                                                                  ][ =vote
                                                           ) ) 0,r0((append.vote
                                                           ) ) 1,r1((append.vote
                                                           ) ) 2,r2((append.vote
                                                           ) ) 3,r3((append.vote
```

```
                                                    ) vote(max=maximum
                                                ]1[maximum=knn_result
                                    :)result(float==)knn_result(floatif
                                                                1=+t
                                    :)result(float=! )knn_result(floatelif
                                                                1=+f

                                                    ) ) t+f(/(f)(=error
                                                        # print("true: ", t )
                                                    # print("false: ", f )
                                                # print("error: ", error)
                                                            errorreturn

                                                                        } ###
```

3. Reading problem data and converting them into forms
   Usable: In this section, data from the file
   read and then converted into a matrix Thyroid.txt
   And so on to dointo 5 equal parts of cross validation It is divided and inCross validation 5 times each time One part is used as a test and the remaining four parts are considered for training.

   In the following, the data is once again normalized and the necessary transformations are made to create the required data This time, cross validation was performed on the normalized data has taken. The following formula is used to normalize the data (except binary and label value):

   $$x\_new = |(x\_old – avg(x))| /variance(x)$$

4. Receiving the output: the data for cross validation
   We made it once in a simple form and another time in a normalized function formWe give the nearest neighborhood and -k
   Do this forWe do 1 to 10 and k for each. We calculate the average value k
5. Drawing a graph: on a graph, the average error Algorithm nearest neighbor for both k-states We drew the original and normalized.

```
                                                    ### error average diagram: {
                                                        ) (subplots.plt=ax,fig
                                            ]10,9,8,7,6,5,4,3,2,1[=k_list
```

```
                                                    ) 2.0=linewidth,'b',avrage,k_list(plot.ax
                                                    ) 2.0=linewidth,'g',navrage,k_list(plot.ax
                                                                                        ) (show.plt

                                                                                        } ###
```

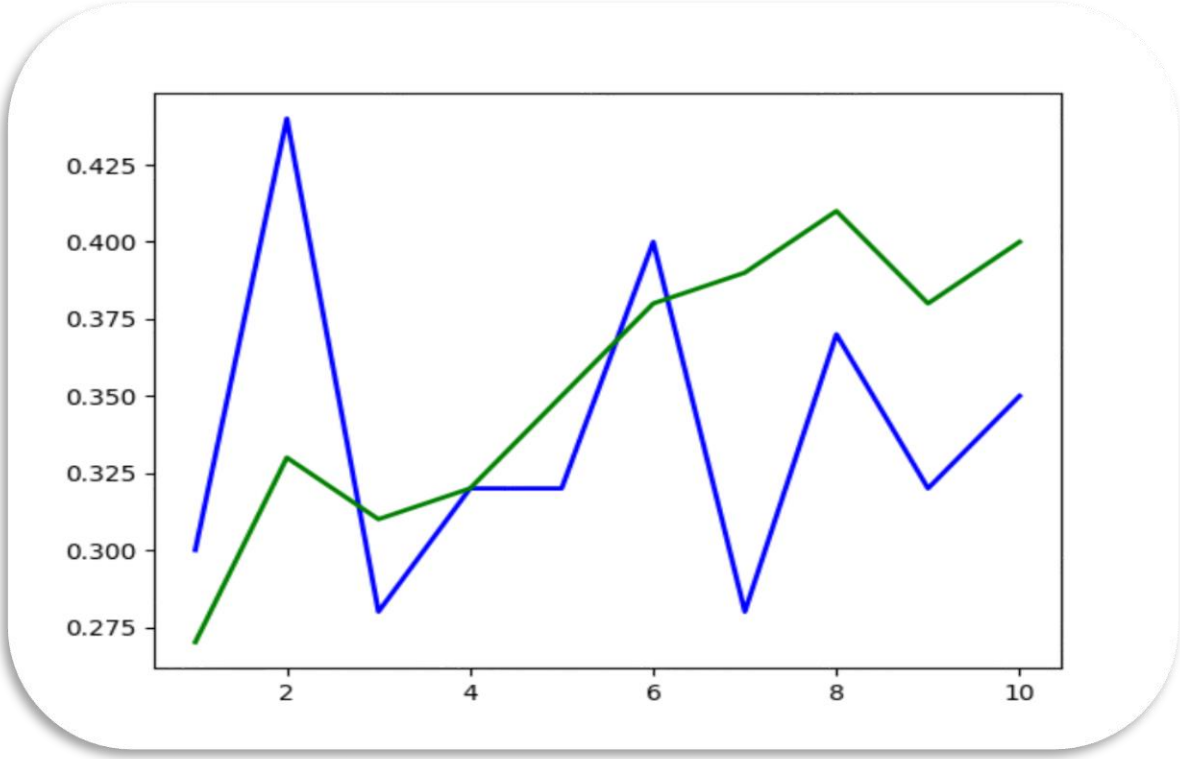
result discussion

The output values of the program are as described in the following table:

Normalized data	Original data	k
0.27	0.3	1
0.32999999999999996	0.44000000000000006	2
0.31	0.28	3
0.31999999999999995	0.32	4
0.35	0.32	5
0.38	0.4	6
0.39	0.27999999999999997	7
0.41	0.37	8
0.38	0.31999999999999995	9
0.4	0.35	10

The lowest amount of error for the original mode withHappened k = 7 is normalized with the value. k = 1

In the diagram drawn for this table, the way the error changes with increaseYou can see k



Green: Normalized–Blue: Original