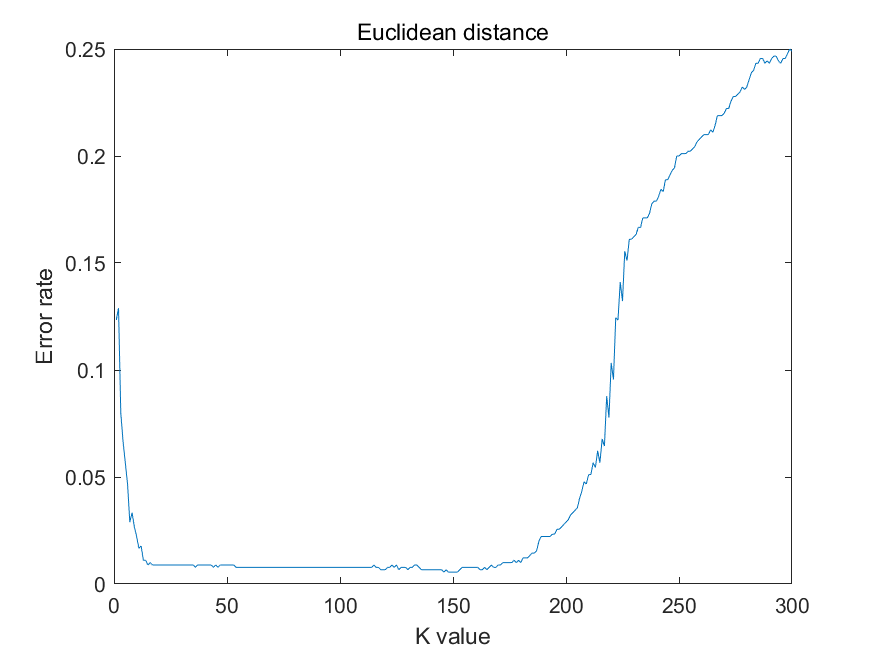
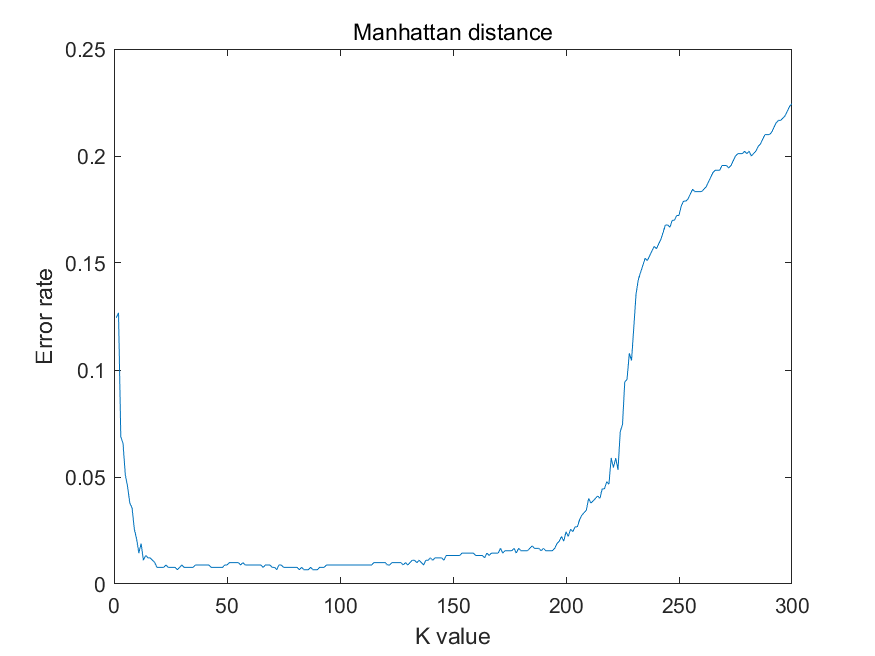
PA1 Report

themis12@kaist.ac.kr / 010 2830 3651  
20184448 Jo MinKi

1. **Introduction**This report contains the summary of classic classifier matlab implementation and experiments. There are KNN classifier with Manhattan distance and Euclidean distance in various K value.
2. **Result**

  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
KNN with Manhattan distance has the minimum error rate 0.0067 at k = 90 and KNN with Euclidean distance has **the minimum error rate 0.0056 at k = 150**. Since the number of training data and test data was too small, the error rate has same value at many different k value. For example, the all Error rate was 0.0056 at K = 146:152. In addition, according to this experiment, NN is worse than KNN by comparing the error rate at k = 1.

**3 Implementation**

KNN implementation

function ErrRate = classify(W1\_train, W2\_train, W3\_train, W1\_test, W2\_test, W3\_test)

tr\_data = [W1\_train; W2\_train; W3\_train];

te\_data = [W1\_test; W2\_test; W3\_test];

% K-value

k = 150;

% prediction list, returns the predicted class

predict\_list = [];

data\_size = size(tr\_data);

num\_data = data\_size(1);

for i = 1:num\_data

instance = te\_data(i,:);

% Euclidean distance

diff = abs(tr\_data - instance);

euc\_dist = sqrt(sum(diff.^2,2));

% Manhattan distance

man\_dist = sum(diff,2);

dist = euc\_dist;

% ranking of the close datapoints

[~, ranking] = sort(dist, 'ascend');

num\_class = [0, 0, 0];

for j = 1:k

if ranking(j)<=300

num\_class(1) = num\_class(1) + 1;

elseif ranking(j) <= 600

num\_class(2) = num\_class(2) + 1;

else

num\_class(3) = num\_class(3) + 1;

end

end

% prediction based on KNN

[~, predict] = max(num\_class);

predict\_list = [predict\_list, predict];

end

Error\_W1 = 300 - sum(predict\_list(1:300)==1);

Error\_W2 = 300 - sum(predict\_list(301:600)==2);

Error\_W3 = 300 - sum(predict\_list(601:900)==3);

Error\_total = Error\_W1 + Error\_W2 + Error\_W3;

ErrRate = Error\_total/num\_data;

end

KNN implementation

Calculate the number of closer data than Kth close data, and select the most likelihood class based on that statistic. You can choose the metric Euclidean distance or Manhattan distance by modifying dist = line.

Testing  
  
result = [];

k\_list = 1:300;

for i = k\_list

Err = classify\_k(W1\_train, W2\_train, W3\_train, W1\_test, W2\_test, W3\_test, i);

result = [result; [i, Err]];

end

fig = plot(result(:,1), result(:,2));

title('Euclidean distance');

xlabel('K value');

ylabel('Error rate');

saveas(fig, 'Euclidean\_distance.png');

[val, ind] = min(result(:,2));

There is classify\_k function in the zip file which can be modified k value. This code calculate the error rate at 300 different k value from 1 to 300.