Assignment 1

Noor Pratap Singh

201364176

Datasets

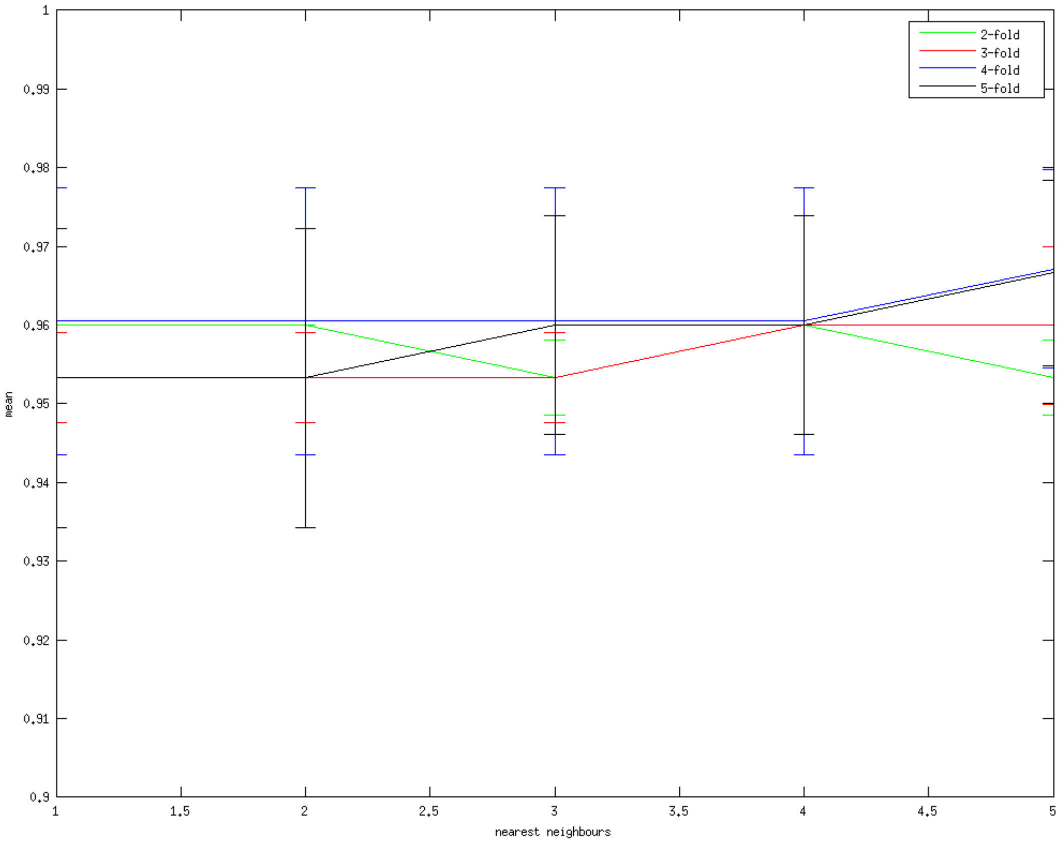
1. Iris Plants Database

A) Number of Instances – 150
Number of features – 4
Number of classes – 3
Missing Attributes – None
Attribute Information

- Sepal width in cm
- Sepal width in cm
- Petal length in cm
- Petal width in cm
- Class
  - Iris Setosa
  - Iris Versicolour
  - Iris Virginica

B) The distance function used was 'Euclidean' though 'cityblock' and 'cosine' were tried but the best mean was coming through Euclidean. Also in case of tiebreaks 'nearest' was used though 'random' was tried but again best result came with 'nearest'.

C)



d)**Mean-**	0.9667	0.9667	0.9667	0.9600	0.9600
	0.9600	0.9600	0.9667	0.9667	0.9600
	0.9539	0.9605	0.9737	0.9737	0.9803
	0.9533	0.9533	0.9667	0.9667	0.9600

Standard Deviation	0.9667	0.9667	0.9667	0.9600	0.9600
	0.9600	0.9600	0.9667	0.9667	0.9600
	0.9539	0.9605	0.9737	0.9737	0.9803
	0.9533	0.9533	0.9667	0.9667	0.9600

As seen from the above graph the mean value increases as the no of neighbours increase across each fold and then decreases but here in it for some folds it remains constant and then increases or vice versa. In general best result comes with 4-5 folds and 5 nearest neighbours.

2) Car Evaluation Database

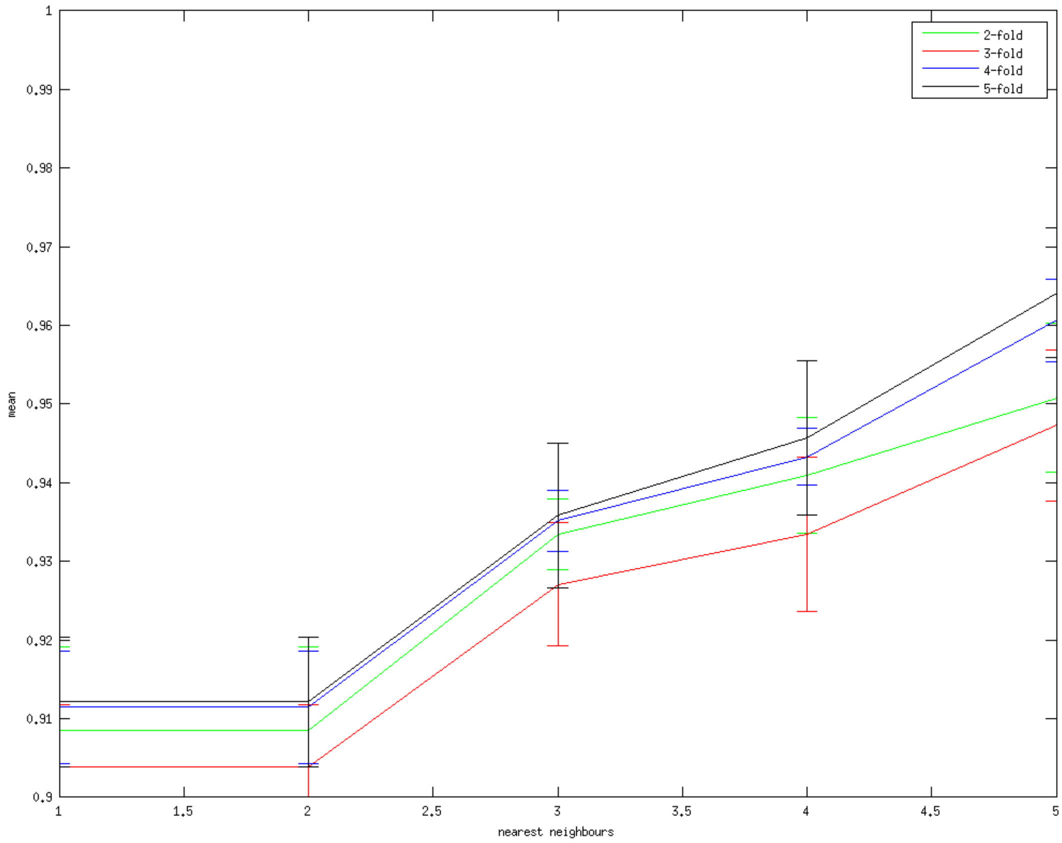
A) Number of Instances: 1728
Number of Features: 6
Number of Classes: 4
Missing Attributes: None
Attribute Values:

buying	v-high, high, med, low
maint	v-high, high, med, low
doors	2, 3, 4, 5-more
persons	2, 4, more
lug_boot	small, med, big
safety	low, med, high

-

B)The distance used was 'cityblock' though 'Euclidean' was also tried but best results came with it. Also 'nearest' was used for tie-breaking again beacuse of best results.

C)



D) Mean	0.9086	0.9086	0.9334	0.9410	0.9508
	0.9039	0.9039	0.9271	0.9334	0.9473
	0.9115	0.9115	0.9352	0.9433	0.9606
	0.9121	0.9121	0.9358	0.9457	0.9642

Standard Deviation	0.0213	0.0213	0.0090	0.0147	0.0188
	0.0157	0.0157	0.0156	0.0195	0.0192
	0.0143	0.0143	0.0078	0.0072	0.0105
	0.0166	0.0166	0.0184	0.0196	0.0164

From the graph we see increasing neigbours increases the accuracy and since it is not following implies that k =6,7 would be the optimum value of k before it starts to fall. Also 'cityblock' has been used which minimises the sum of differences. More neighbours would have given us a better approximation.

Also for the above special feature extraction was done as the inputs were strings and they were converted to corresponding numbers.

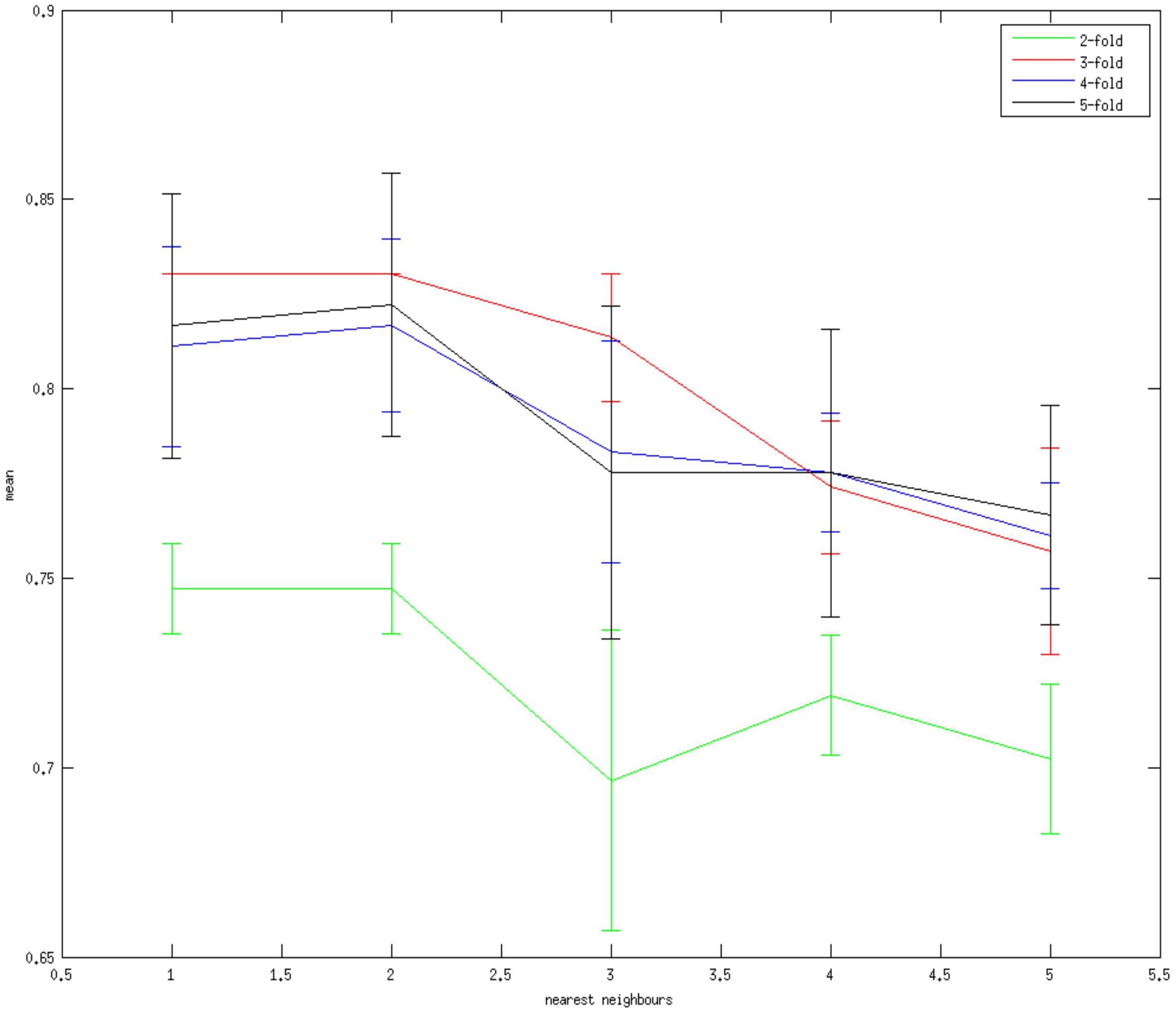
3) Wine Recognition Data

A) Number of Instances: 168
Number of Features: 13
Number of Classes: 3
Missing Attributes: None
Attributes

- Alcohol
- Malic acid
- Ash
- Alcalinity of ash
- Magnesium
- Total phenols
- Flavanoids
- Nonflavanoid phenols
- Proanthocyanins
- Color intensity
- Hue
- OD280/OD315 of diluted wines
- Proline

B)The distance used was 'cityblock' though 'Euclidean' was also tried but best results came with it. There was a vast difference between the two and for t tie-breaking 'nearest' was used.

C)



D) Mean

Mean	0.7472	0.7472	0.6966	0.7191	0.7022
	0.8305	0.8305	0.8136	0.7740	0.7571
	0.8111	0.8167	0.7833	0.7778	0.7611
	0.8167	0.8222	0.7778	0.7778	0.7667
Standard Deviation	0.0238	0.0238	0.0795	0.0318	0.0397
	0	0	0.0339	0.0353	0.0545
	0.0529	0.0458	0.0584	0.0314	0.0280
	0.0697	0.0697	0.0878	0.0761	0.0576

The results for 2 fold are very bad which could say that more training data is required for getting accurate results means that for more generic boundary we need larger training set. Also from graph we can say that the optimum k values is 1,2 and increasing the neighbours affects our inference of data which could mean data is more sparse.

4)Contraceptive Method Choice

A) Number of Instances: 1473

Number of Features: 9

Number of Classes: 3

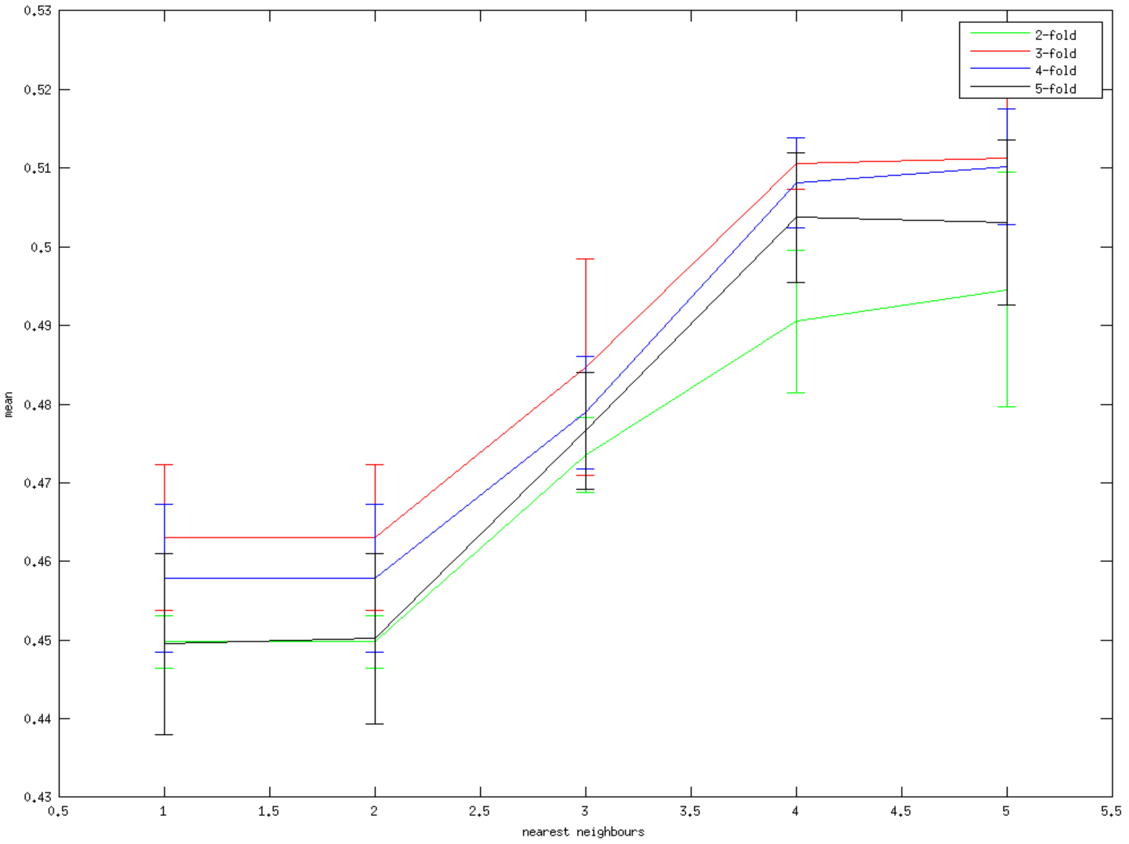
Missing Attributes: None

Attributes

• Wife's age	(numerical)	
• Wife's education	(categorical)	1=low, 2, 3, 4=high
• Husband's education	(categorical)	1=low, 2, 3, 4=high
• Number of children ever born	(numerical)	
• Wife's religion	(binary)	0=Non-Islam, 1=Islam
• Wife's now working?	(binary)	0=Yes, 1=No
• Husband's occupation	(categorical)	1, 2, 3, 4
• Standard-of-living index	(categorical)	1=low, 2, 3, 4=high
• Media exposure	(binary)	0=Good, 1=Not good
• Contraceptive method used	(class attribute)	1=No-use 2=Long-term 3=Short-term

B)'Euclidean' distance was chosen though cityblock was also tried because of better results. “Nearest' was used for tie-breaking.

C)



D)The accuracy of above data is less. I specifically chose this to show that knn could fail for large data sets and for some specific sort of classification. In general the optimum value of k comes out to be 4 from the above figure.

Codes

Extraction.m – Basic Feature extraction and passing inputs and receiving the desired ouput for means,sds

```
Files = {'iris.data', 'car.data',  'wine.data', 'cmc.data'};

f1 = fopen(Files{1});
data = textscan(f1, '%f%f%f%f%s', 'delimiter', ',');
inp_data_iris = data;
fclose(f1);

inp_data_iris = randomize(data);

f2 = fopen(Files{2});
data = textscan(f2, '%s%s%s%s%s%s%s', 'delimiter', ',');
inp_data_car = data;
fclose(f2);

inp_data_car = randomize(data);

f3 = fopen(Files{3});
data = textscan(f2, '%f%f%f%f%f%f%f%f%f%f%f%f%f%f', 'delimiter', ',');
inp_data_wine = data;
fclose(f3);
inp_data_wine = randomize(data);
class_samp_iris = 0;

f4 = fopen(Files{4});
data = textscan(f2, '%f%f%f%f%f%f%f%f%f%f', 'delimiter', ',');
inp_data_cmc = data;
fclose(f4);
inp_data_cmc = randomize(data);



%%%Feature Extraction of Iris
for i = 1:length(inp_data_iris{1})
        if(strcmp(inp_data_iris{5}(i), 'Iris-setosa'))
                class_samp_iris(i) = 1;
        elseif(strcmp(inp_data_iris{5}(i), 'Iris-versicolor'))
                class_samp_iris(i) = 2;
        else
                class_samp_iris(i) = 3;
        end
end
inp_data_iris{5} = class_samp_iris;

%%%Feature Extraction of Car
buy = zeros(length(inp_data_car{1}), 1);
maint = zeros(length(inp_data_car{1}), 1);
doors = zeros(length(inp_data_car{1}), 1);
persons = zeros(length(inp_data_car{1}), 1);
lug = zeros(length(inp_data_car{1}), 1);
safety = zeros(length(inp_data_car{1}), 1);
class_samp_car = zeros(length(inp_data_car{1}), 1);

for i = 1:length(inp_data_car{1})

        if(strcmp(inp_data_car{1}(i), 'vhigh'))
                buy(i) = 6;
        elseif(strcmp(inp_data_car{1}(i), 'high'))
                buy(i) = 4.5;
        elseif(strcmp(inp_data_car{1}(i), 'med'))
                buy(i) = 3;
        else
                buy(i) = 1.5;
        end

        if(strcmp(inp_data_car{2}(i), 'vhigh'))
                maint(i) = 6;
        elseif(strcmp(inp_data_car{2}(i), 'high'))
                maint(i) = 4.5;
        elseif(strcmp(inp_data_car{2}(i), 'med'))
                maint(i) = 3;
        else
                maint(i) = 1.5;
        end

        if(strcmp(inp_data_car{3}(i), '2'))
                doors(i) = 2;
        elseif(strcmp(inp_data_car{3}(i), '3'))
                doors(i) = 3;
        elseif(strcmp(inp_data_car{3}(i), '4'))
                doors(i) = 4;
```

```
    else
        doors(i) = 6;
    end

    if(strcmp(inp_data_car{4}(i), '2'))
        persons(i) = 2;
    elseif(strcmp(inp_data_car{4}(i), '4'))
        persons(i) = 4;
    else
        persons(i) = 6;
    end

    if(strcmp(inp_data_car{5}(i), 'small'))
        lug(i) = 2;
    elseif(strcmp(inp_data_car{5}(i), 'med'))
        lug(i) = 4;
    else
        lug(i) = 6;
    end

    if(strcmp(inp_data_car{6}(i), 'low'))
        safety(i) = 2;
    elseif(strcmp(inp_data_car{6}(i), 'med'))
        safety(i) = 4;
    else
        safety(i) = 6;
    end
    if(strcmp(inp_data_car{7}(i), 'unacc'))
        class_samp_car(i) = 1;
    elseif(strcmp(inp_data_car{7}(i), 'acc'))
        class_samp_car(i) = 2;
    elseif(strcmp(inp_data_car{7}(i), 'good'))
        class_samp_car(i) = 3;
    else
        class_samp_car(i) = 4;
    end

end

inp_data_car = [buy, maint, doors, persons, lug, safety, class_samp_car];

%%For wine
tem = transp([inp_data_iris{1}; inp_data_iris{2}; inp_data_iris{3}; inp_data_iris{4};
class_samp_iris]);
inp_data_iris = tem;
```

```
length_wine = length(inp_data_wine);
for i = 1: length_wine
        te(:,i) = inp_data_wine{mod(i,length_wine) + 1} ;
end

inp_data_wine = te;

%%For contraception
for i = 1: length(inp_data_cmc)
        temp_cmc(:,i) = inp_data_cmc{i};
end
inp_data_cmc = temp_cmc;

inp_data = {inp_data_iris, inp_data_car, inp_data_wine, inp_data_cmc};

me = zeros(4,5,5)
for i = 1:4
        [mean,sd] = k_all(inp_data{i}, 1);
        me(i,:,:) = mean;
        std(i,:,:) = sd;
end
```

Knn_Classifier.m

```
function [ output ] = knn_classifier( test, training, group, k, d)
    dim = size(test);
    output = zeros(dim(1), 1);
    for i = 1:dim(1)
        grp = inf(k,1);
        distances = inf(k, 1);
        abs_distances = inf(k, 1);
        grp2 = inf(k,1);
        for j = 1:size(training, 1)
            dist = sum((training(j,:)- test(i,:)).^2);
            dist2 = sum(abs(training(j,:)- test(i,:)));
            [large1, index1] = max(abs_distances);
            [large, index] = max(distances);
            if(dist < large)
                distances(index) = dist;
                grp(index) = group(j);
            end
            if(dist2 < large1)
                abs_distances(index1) = dist2;
                grp2(index1) = group(j);
            end
        end
        if(d == 1)
            counts = tabulate(grp);

            [m , ind] = max(counts(:, 2));
            output(i) = counts(ind);
        end
        if(d==2)
            counts = tabulate(grp2);

            [m , ind] = max(counts(:, 2));
            output(i) = counts(ind);
        end
    end


end
```

K_fold.m – All folds in it

```
function [ mean, sd ] = k_fold(k, sample, p )

          size_sample = size(sample);
          no_of_rows = size_sample(1);
          len_of_seg = int32(no_of_rows/k);
          start_indexes =  1:len_of_seg:no_of_rows;
          actual = 1:no_of_rows;
          m = 1;
          n = 1;
          mean = 0;
          count = 0;
          obs = zeros(k,1);
          for p = 1:5
                    for i = 1:k
                              for j = 1:no_of_rows
                                        if(j >= start_indexes(i) && j < start_indexes(i) + len_of_seg
)
                                                  test(m) = actual(j);
                                                  m = m + 1;
                                        else
                                                  training(n) = actual(j);
                                                  n = n + 1;
```

```
                end
            end
            output_rows = knn_classifier(sample(test, 1:size_sample(2) - 1),
sample(training, 1:size_sample(2) - 1),...
             sample(training, size_sample(2)), p, 1);

            %output_rows == sample(test, size_sample(2))
            count = 0;
            for j=1:size(output_rows)
                if(output_rows(j) == sample(test(j), size_sample(2)))

                    count = count + 1;
                end
            end

            obs(i) = count/double(len_of_seg);
            count = 0;

            m = 1;
            n = 1;
        end

        mean(p) = sum(obs)/k;
        sd(p) = std(obs);

    end

end
```

k_all.m - Runs the K_fold in loop from 2-5 and plots the graph

```
function[mean_com,sd_com] = k_all(sample, d)
    mean_com = zeros(4, 5);
    sd_com =zeros(4,5);
    for i = 2:5
        [mean_com(i,:), sd_com(i,:)] = k_fold(i, sample, d);
    end
    n = 1:5;
    figure();
    f2 = errorbar(n, mean_com(2,:), sd_com(2,:)/2,'g'); hold on;
    f3 = errorbar(n, mean_com(3,:), sd_com(3,:)/2,'r'); hold on;
    f4 = errorbar(n, mean_com(4,:), sd_com(4,:)/2,'b'); hold on;
    f5 = errorbar(n, mean_com(5,:), sd_com(5,:)/2,'k');
    legend([f2, f3, f4, f5], {'2-fold', '3-fold', '4-fold', '5-fold'});
    xlabel('nearest neighbours' );
    ylabel('mean');

end
```

Randomize.m – Randomizes data

```
function [ out_data ] = randomize( input_data )
% This function randomizes the given cell
rand = randperm(length(input_data{1}));
for i = 1:length(rand)
    for j = 1:length(input_data)
        out_data{j}(i) = input_data{j}(rand(i));
    end
end
```