Homework 1
Applied Machine Learning
Fall 2017
CSCI-P 556/INFO-I 526

Chuck Jia
jiac@indiana.edu

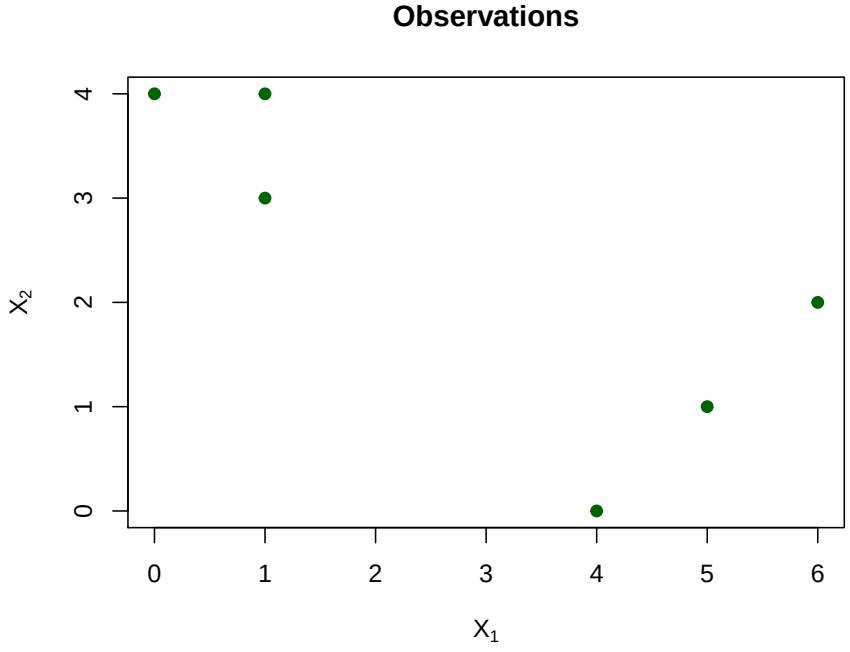
September 18, 2017

"All the work herein is solely mine."

Problem 1 [20 points]

From textbook, Chapter 10 exercise 3 (Page 414).

(a) A plot of the observations is given below.



(b) The cluster labels are randomly assigned to each observations in `R` by using

```
sample.int(2, size = 6, replace = TRUE)
```

One result is given in the table below. This result will be used in later calculations in this problem.

Observation	Cluster Label
1	1
2	2
3	2
4	1
5	2
6	2

(c) In this case, the coordinates of the centroids are calculated as

- centroid 1 $= \left(\dfrac{x_{11} + x_{41}}{2}, \dfrac{x_{12} + x_{42}}{2} \right) = (3.0, 2.5)$
- centroid 2 $= \left(\dfrac{x_{21} + x_{31} + x_{51} + x_{61}}{4}, \dfrac{x_{22} + x_{32} + x_{52} + x_{62}}{4} \right) = (2.75, 2.25)$

(d) Using the coordinates of the centroids derived in (c), the distance between the i^{th} observation (x_{i1}, x_{i2}) and the j^{th} centroid (c_{j1}, c_{j2}) can be calculated by the following formula

$$\text{distance} = \sqrt{(x_{i1} - c_{j1})^2 + (x_{i2} - c_{j2})^2}$$

Applying this formula to all pairs of observation and centroid, I derived the following results.

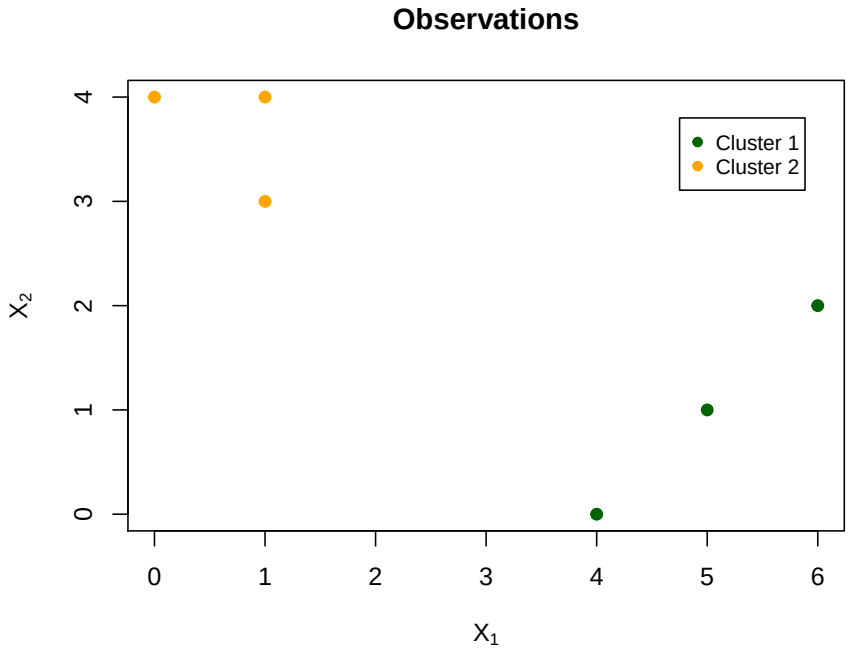
Observation	Distance from centroid 1	Distance from centroid 2	Label
1	2.5	2.474874	2
2	2.061553	1.903943	2
3	3.354102	3.259601	2
4	2.5	2.573908	1
5	3.041381	3.259601	1
6	2.692582	2.573908	2

(e) I repeated the process and generated the following results

Observation	Initial Label	Label After 1 Calculation	Label After 2 Calculations
1	1	2	2
2	2	2	2
3	2	2	2
4	1	1	1
5	2	1	1
6	2	2	1

The labels (or equivalently the assignments to clusters) stopped changing after 2 calculations.

(f) The clustering result is shown in the the plot below. Green points represent observations in cluster 1 and orange points represent observations from cluster 2.


Observations
Cluster 1
Cluster 2
X2
X1
0
1
2
3
4
5
6

Problem 2 [20 points]

The pseudo-code for k-means and a running example of k-means on a small data set are provided above. Answer the following questions

2.1 Does k-means always converge? Given your answer, a bound on the iterate must be included. How is its value determined?

2.2 What is the run-time of this algorithm?

(2.1) k-means algorithm (the one given in the homework description) always converges. But it can converge very slowly. A theoretical (but trivial) upper bound on the number of iterations is k^n, which is the number of all possible clustering results with at most k clusters.

In practice, the bound (usually) should be set to a much smaller number than the exponential upper bound (especially when k and n are large). Its value can be determined by running the algorithm on specific problems with specific machines and observing the run time. Then a reasonable value for the upper bound on iterations can be set according to the performance of the algorithm.

(2.2) The run-time of the given k-means algorithm is $O(nkdi)$, where n is the number of data points, d is the dimension of data, and i is the total number of iterations.

(The run-time is contributed by line 20-23, where in comparison, line 13-16 contributed $O(k)$, line 24-31 contributed $O(k^2 i)$, and line 34 contributed $O(kdi)$.)

An upper bound on the run-time can be given as $O(k^{n+1}nd)$.

Problem 3 [50 points]

Implement Lloyd's algorithm for k-means (see algorithm k-means below) in R and call this program C_k. As you present your code explain your protocol for

3.1 initializing centroids

3.2 maintaining k centroids

3.3 splitting centroids

3.4 deciding ties

3.5 stopping criteria

(3.1) In my implementation, the centroids are initialized by randomly labeling all the data points and then calculate the centroid of each cluster.

Specifically, for each data point, one label is randomly selected from the k labels and assigned to that data point.

To make sure k clusters are generated, after all data points are labeled, k data points are drawn randomly from the n data points and then labeled as `cluster 1`, `cluster 2`, $\cdots$, `cluster k`.

(3.2-3.3) To maintain k centroids, I used the following splitting method.

In any iteration, once empty clusters form, they will be detected when new centroids are calculated (specifically when the means are taken). Then the number of empty clusters will be noted. Suppose there are m empty clusters. The algorithm maintains k clusters by the following steps.

1. Randomly draw m data points from all data points that do NOT belong to clusters with single points.
2. Assign these m data points to the m empty clusters as their new centroids.
3. Then proceed as usual and label all data points by using the newly derived k centroids.

(3.4) In the situation of a tie, I did not add any special procedure. A data point will be assigned smaller label number whenever there is a tie in comparing distances.

This treatment will not add to the complexity of algorithm, and thus will be easy on run-time. Multiple runs with different initial conditions can help with possible bias generated.

(3.5) For the stopping criteria, I observed the performance of my program on my machine and set the maximum number of iterations to 1000.

Also, my program uses the average of the 2-norm in the changes of the centriods (see line 34 in the algorithm)

$$\texttt{change} = \frac{1}{k}\sum_{j=1}^{k} \|c_j^{i-1} - c_j^i\|$$

The tolerance τ is set to be 0.001. This value allows my program to finish in reasonable time and produce a good result.

Problem 4 [50 points]

In this question, you are asked to run your program, C_k, against the Ringnorm and Ionosphere data sets and answer the following question. Click on the links to download the data sets.

- Ringnorm Data Set
- Ionosphere Data Set

Upon stopping, you will calculate the quality of the centroids and of the partition. For each centroid c_i, form two counts (over Ionosphere Data Set) :

$$g_i \leftarrow \sum_{\delta \in c_i.B} [\delta.C == \text{“g”}], \quad \text{good}$$
$$b_i \leftarrow \sum_{\delta \in c_i.B} [\delta.C == \text{“b”}], \quad \text{bad}$$

where $[x = y]$ returns 1 if True, 0 otherwise. For example, $[2 = 3] + [0 = 0] + [34 = 34] = 2$

The centroid c_i is classified as good if $g_i > b_i$ and bad otherwise. We can now calculate a simple error rate. Assume c_i is good. Then the error is:

$$error(c_i) = \frac{b_i}{b_i + g_i}$$

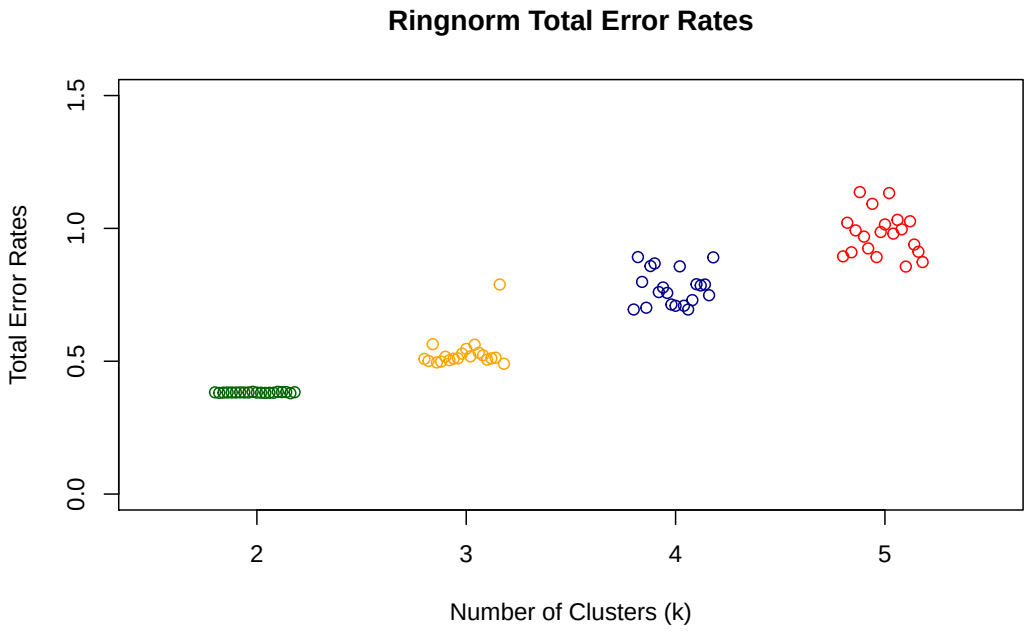
We can find the total error rate easily:

$$Error(\{c_1, c_2, \ldots, c_k\}) = \sum_{i=1}^{k} error(c_i)$$

Report the total error rates for $k = 2, \ldots 5$ for 20 runs each, presenting the results that are easily understandable. Plots are generally a good way to convey complex ideas quickly. Discuss your results.

---

(1) The plot of total error rates for the Ringnorm Data Set is given below. The green, orange, purple, and red dots represents total error rates from $k = 2$, 3, 4, and 5 respectively. Each color has 20 dots, representing results from 20 runs.

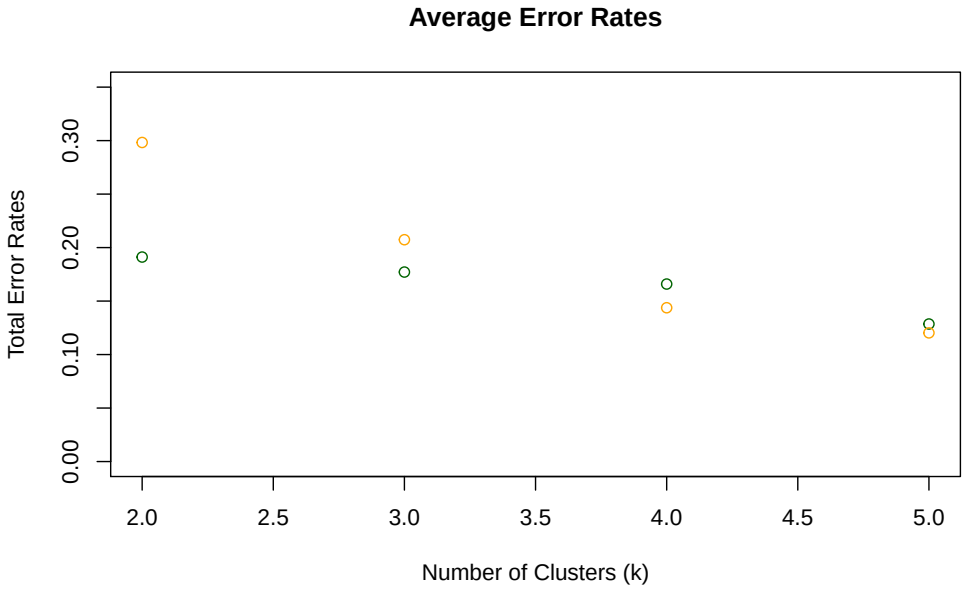


(2) The plot of total error rates for the Ionosphere Data Set is given below. The green, orange, purple, and red dots represents total error rates from $k = 2,\ 3,\ 4$, and 5 respectively. Each color has 20 dots, representing results from 20 runs.

(3) Analysis of the results. From the two plots of total error rates, the following observations can be seen.

1. In general, the error rates are not high, but not small as well. The average error rate in one cluster is around 20% (see the plot for average error rates in one cluster below). This

implies that the (Lloyd) k-means algorithm works for this data set, but does not cluster perfectly.



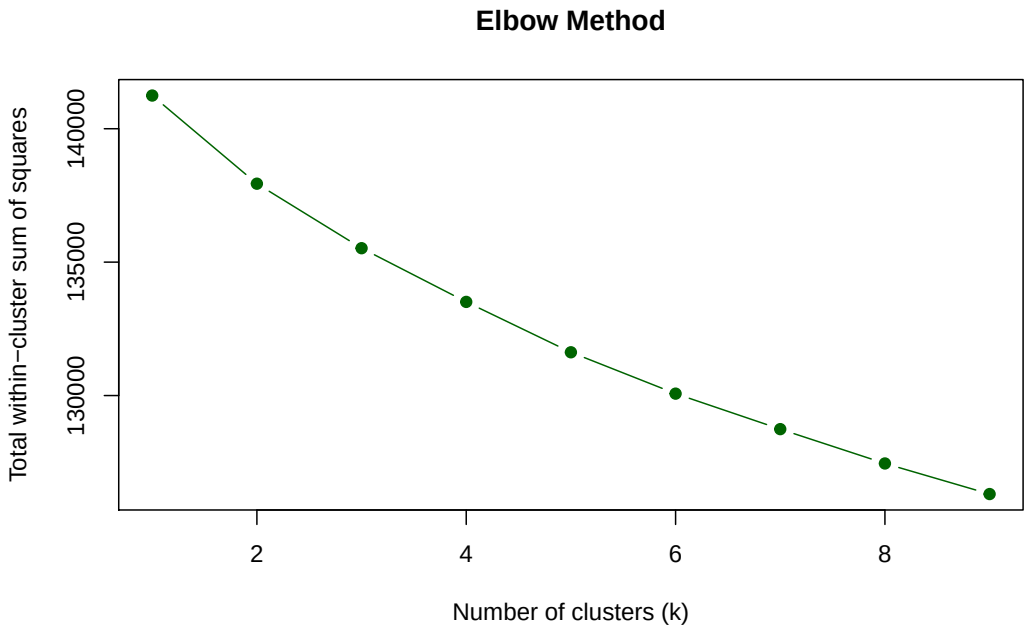
In the plot, each dark green dot represents the average error rate in one cluster from the Ringnorm calculations. The orange dot represents the ones from the Ionosphere calculations.

2. The error rates for $k = 2$ in calculations for both data set are very stable, indicating that there is high probability that only one local optimum exists, which is the global optimum (although another possibility is that the optimum is the only algorithm can converge to, which indicates flaws in the program, specifically in the initialization and tie-breaking methods). For the Ringnorm data, the error rates do not vary too much, whereas for the Ionosphere data, the error rates showed large variance. This indicates that many local solutions exist for Ionosphere Data Set problem and the solution is highly dependent on the initial conditions.

Problem 5 [50 points]

In this question, you are asked to make use of the R package for k-means implementation. Elbow method is one of the techniques to decide the optimal cluster number. Find the optimal number of clusters using elbow method for Ringnorm and Ionosphere data sets. Report your results in a plot as shown here for $2 \leq k \leq 10$. (The link includes an example.)

We plot the total within-clusters sum of squares for $k = 2, 3, \cdots, 10$.



From the plot we can see that at $k = 2$ (or $k = 3$), the (between sum of squares) / (total sum of squares) ratio tends to change slowly and remain less changing as compared to other k's. So $k = 2$ or $k = 3$ should be good choices for the optimal number of clusters.

Problem 6 [20 points]

Let $X \subset \mathbb{R}^n$ ($\mathbb{R}$ is the set of reals) for positive integer $n > 0$. Define a distance $d : X \times X \to \mathbb{R}_{\geq 0}$ as

$$d(x, y) = max\{|x_i - y_i|\}, \forall i \ 1 \geq i \geq n$$

Is d a metric?

---

d is a metric. This can be shown by checking the 4 conditions in the definition of a metric.

1. (*Non-negativity*) Since $|x_i - y_i| \geq 0$ for all i,

$$d(x, y) = \max_{1 \leq i \leq n} \{|x_i - y_i|\} \geq 0$$

2.
$$\begin{aligned} & d(x, y) = 0 \\ \Leftrightarrow & \max_{1 \leq i \leq n} \{|x_i - y_i|\} = 0 \\ \Leftrightarrow & |x_i - y_i| = 0, \quad 1 \leq i \leq n \text{ (Because } |x_i - y_i| \geq 0 \text{ for all } i) \\ \Leftrightarrow & x_i = y_i, \quad 1 \leq i \leq n \\ \Leftrightarrow & x = y \end{aligned}$$

3. (*Symmetry*)
$$\begin{aligned} d(x, y) &= \max_{1 \leq i \leq n} \{|x_i - y_i|\} \\ &= \max_{1 \leq i \leq n} \{|y_i - x_i|\} \\ &= d(y, x) \end{aligned}$$

4. (*Triangle inequality*)
$$\begin{aligned} d(x, y) + d(y, z) &= \max_{1 \leq i \leq n} \{|x_i - y_i|\} + \max_{1 \leq i \leq n} \{|y_i - z_i|\} \\ &\geq \max_{1 \leq i \leq n} \{|x_i - y_i| + |y_i - z_i|\} \\ &\geq \max_{1 \leq i \leq n} \{|x_i - z_i|\} \\ &= d(x, z) \end{aligned} \tag{1}$$

The first inequality follows from the fact that

$$\max_{1 \leq i \leq n} \{|a_i|\} + \max_{1 \leq i \leq n} \{|b_i|\} \geq \max_{1 \leq i \leq n} \{|a_i| + |b_i|\} \tag{2}$$

(A simple proof for this would be to see that $\max_{1 \leq i \leq n}\{|a_i|\} + \max_{1 \leq i \leq n}\{|b_i|\} \geq |a_j| + |b_j|$ for any j, by the definition of max. And then by taking max on j on the right hand side, the inequality (2) is proved.)

The second inequality in (1) follows by applying the triangle inequality on the real line, i.e.

$$|x_i - y_i| + |y_i - z_i| \geq |x_i - z_i|$$

Q.E.D.

Extra credit [90 points]

This part is optional.

1 Answer problem 4 using Breast Cancer Wisconsin Data Set. The data sets given in Problem 4 are clean. There are no missing values on those data sets. However, Breast Cancer Wisconsin Data Set has some missing values that must be removed to use with k-means algorithm. The data set can be found here [30 points]

2 The k-means algorithm provided above stops when centroids become stable (Line 34). In theory, k-means converges once SSE is minimized

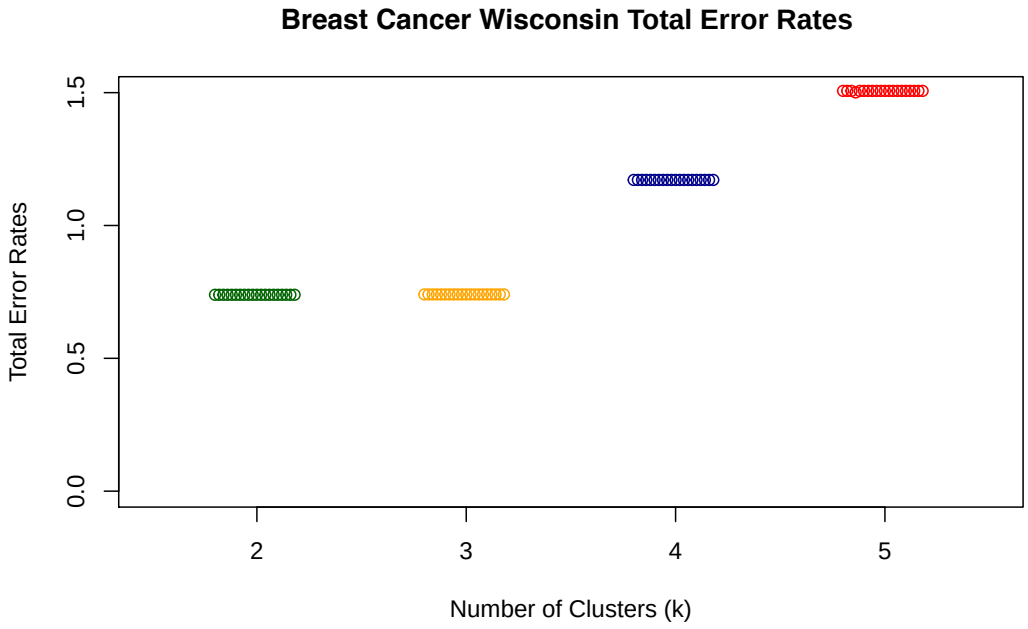
$$SSE = \sum_{j}^{k} \sum_{x \in c_j.B} ||\mathbf{x} - c_j.v||_2^2$$

In this question, you are asked to use SSE as stopping criterion. Run k-means over Breast Cancer Wisconsin Data Set and report the total SSE in a plot for $k = 2, \ldots, 5$ for 20 runs each [30 points].

3 Traditional k-means initialization is based on choosing values from a uniform distribution. In this question, you are asked to improve k-means through initialization. k-means ++ is an extended k-means clustering algorithm and induces non-uniform distributions over the data that serve as the initial centroids. Read the paper and discuss the idea in a paragraph. Implement this idea to improve your k-means program. [30 points]

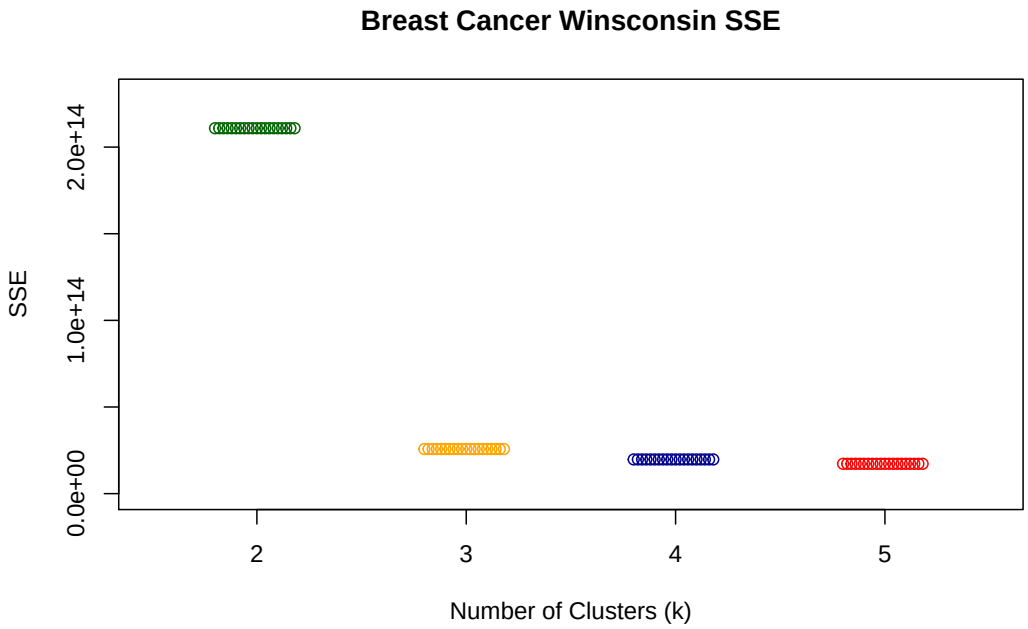
---

(1) A plot on the total error rates is given below.



Analysis of the result. In this data set, my Lloyd k-means program almost always converges to the same local optimum. This can be either because the program finds the true global optimum (the only local optimum), or the initialization method does not work well with this type of data set.

(2) Using SSE in the stopping criterion, a plot of SSE is given below.



(3) The k-means++ algorithm improves the original (Llyod) k-means algorithm on the initialization of centroids. Instead of choosing initial centroids randomly from uniform distribution, the k-means++ algorithm uses a different distribution, which is defined in the following way.

Let $D(x)$ denote the shortest distance from a data point to the closest center we have already chosen. First, one centroid is randomly chosen from a uniform distribution from Δ (the data set). Then we choose another centroid from a distribution where each data point x has probability

$$\Pr[\text{choose point } x] = \frac{D(x)^2}{\sum_{x\in\Delta} D(x)^2}$$

Then update the probabilities and repeat the step until we have k centroids in total.

When choosing each centroid, k-means++ tries to make sure that points further away from the centroids already chosen have better chance of being selected. This in some way helps the algorithm to find better initial conditions to start with.

The algorithm is implemented by the `R` code in the file `ProbEC3.R`.