REPORT
Manish Jaisinghani

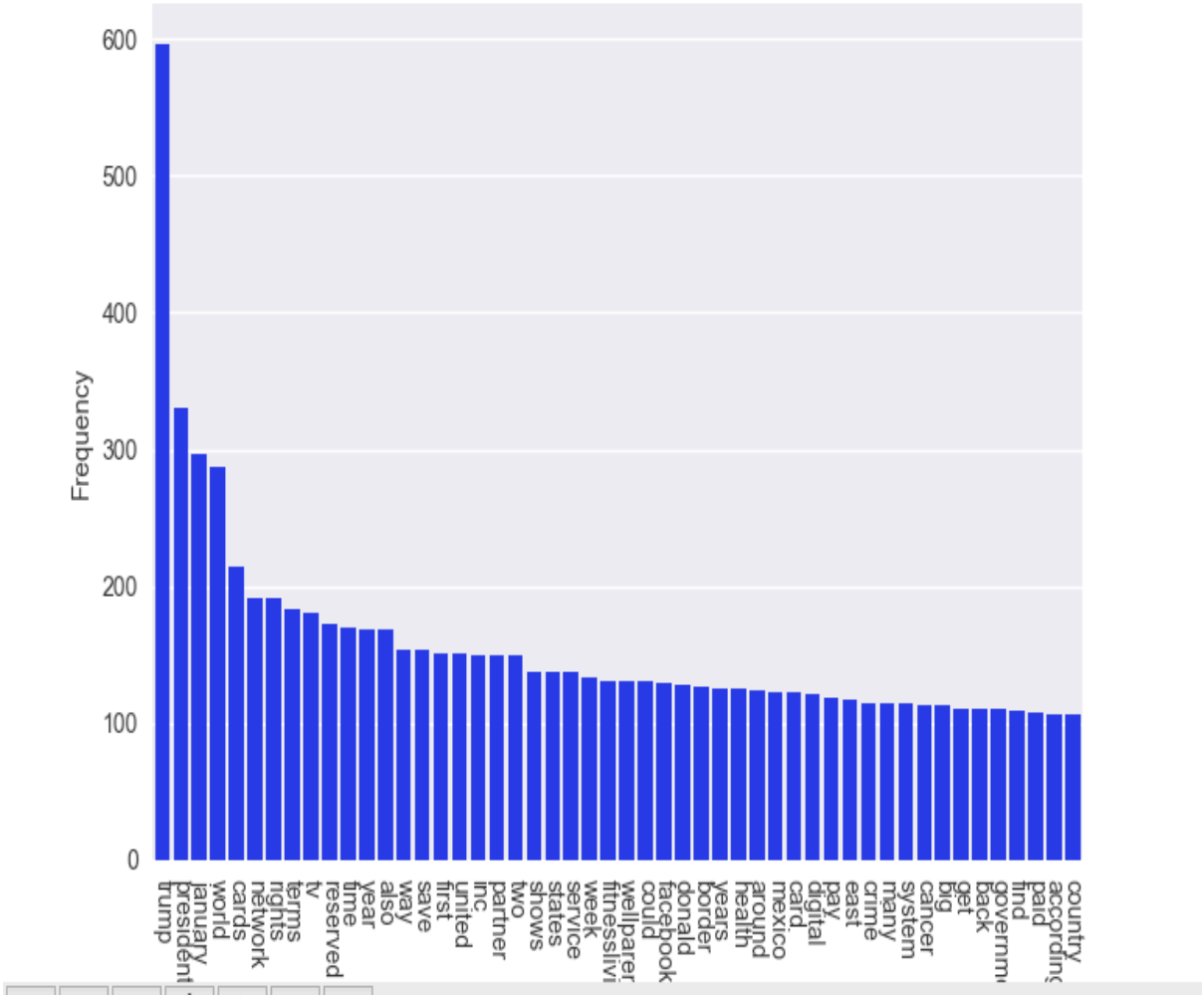
1. Objective: Demonstrate Kmeans algorithm using different distance metrics. The metrics involve Original KMeans, KMeans with Cosine distance, KMeans with Euclidean distance and KMeanse with Jaccard distance.

2. Data Collection: The data collection process is completed in two steps as described below:

   a. Website List: I collected the website list by using online website scrapper http://www.xsitemap.com. The scrapper parses all the sub URLs from a given URL and provides a list of the parsed URLs. The URL list has been divided into 4 categories and 25 URLs have been taken from each category:
      - Politics – 25 articles
      - Health – 25 articles
      - World – 25 articles
      - Technology – 25 articles

   b. Website Scrapper: I have written HTML website scrapper in python using beautiful soup and NLTK as packages to parse the HTML packages and remove words from the collected list that do not add to meaning of the article, for example “a, the, he, she, these etc.” A complete list of the words which have been remove or not taken into consideration has been attached by the file name “stopwords”.

Top 50 Word Frequency



3.Data Processing: The data processing is done in two parts as described below:

a. Website parsing: A list of URLs is provided in the file by name "**website_list**". The python script parses through each website one by one. The script makes use of beautiful soup, parses the website and pulls out text data eliminating all the symbols, punctuations, styles and scripts. Once the data is collected the data is recorded in a list data structure and then stored making sure that there are no duplicate words in the list. The same procedure is repeated for every URL. While parsing the text of current article frequency of words occurring in the article is also recorded. The final step is just writing this data to a CSV file("**data.csv"**). In the csv file each article is represented as a row and

each word is recorded in a column and the final matrix consists of the frequency of words.

b. Data manipulation: For Jaccard distance metric we convert the frequency matrix to similarity matrix which only reports presence or absence of a word in an article. We also normalize the data by feature subset selection.

4.Module specifications:

The algorithms have been implemented using python 3 and below listed libraries have been used:

- **Pandas** – Deals with csv data and helps in manipulation.
- **Numpy** – Deals with data in the form of arrays and assists in manipulation
- **Matplotlib** – Helps to visualize the data.
- **Seaborn** – Another library which helps in data visualization, the only difference here is seaborn is vast than matplotlib.
- Used KMeans algorithm provided in lecture notes and implemented all the distance metrics in the same algorithm.

5. Terminology & Explanation:

- *Euclidean Distance:* The **Euclidean distance** between points **p** and **q** is the distance between two points and is given by:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) \quad = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}.$$

- *Cosine Distance*: Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them.

$$\mathbf{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

$$Cosine\ Distance = 1 - Similarity$$

- *Jaccard Distance:* Given two objects, *A* and *B*, each with *n* binary attributes, the Jaccard coefficient is a useful measure of

the overlap that *A* and *B* share with their attributes. Each attribute of *A* and *B* can either be 0 or 1. The total number of each combination of attributes for both *A* and *B* are specified as follows:

- represents the total number of attributes where *A* and *B* both have a value of 1.
- represents the total number of attributes where the attribute of *A* is 0 and the attribute of *B* is 1.
- represents the total number of attributes where the attribute of *A* is 1 and the attribute of *B* is 0.
- represents the total number of attributes where *A* and *B* both have a value of 0.

The Jaccard distance, d_J, is given as

$$d_J = \frac{M_{01} + M_{10}}{M_{01} + M_{10} + M_{11}} = 1 - J.$$

- *sum of squared errors of prediction (SSE):* is the sum of the squares of residuals (deviations predicted from actual empirical values of data). It is a measure of the discrepancy between the data and an estimation model.

$$\text{SSE} = \sum_{i=1}^{n} (y_i - (\alpha + \beta x_i))^2$$

6. Implementation details:

Below are the details on how the algorithms have been written and pre-requisites for successfully running the same:-

- **kmeans_Cosine.py** - The file contains the implementation of KMeans algorithm using Cosine distance metric.
- **kmeans_Euclidean.py** – The file contains the implementation of KMeans algorithm using Euclidean distance metric.
- **Kmeans_Jaccard.py** – The file contains the implementation of KMeans algorithm using Jaccard distance metric.
- **kmeans_Original.py** – The file contains the implementation of KMeans algorithm.

- **Jacard_Data.csv** – This file contains the data with respect to Jaccard distance. A separate data file is used for jaccard distance because jaccard reports presence or absence rather than frequency of words.

- **Data_KMeans.cvs** – This file contains word frequency metrics which is utilized by all the other distance metrics except jaccard similarity.
- **Constraints** – The system should be installed with all python modules listed in "Module Specifications" section.
- The python script should be executed using python 2.0.

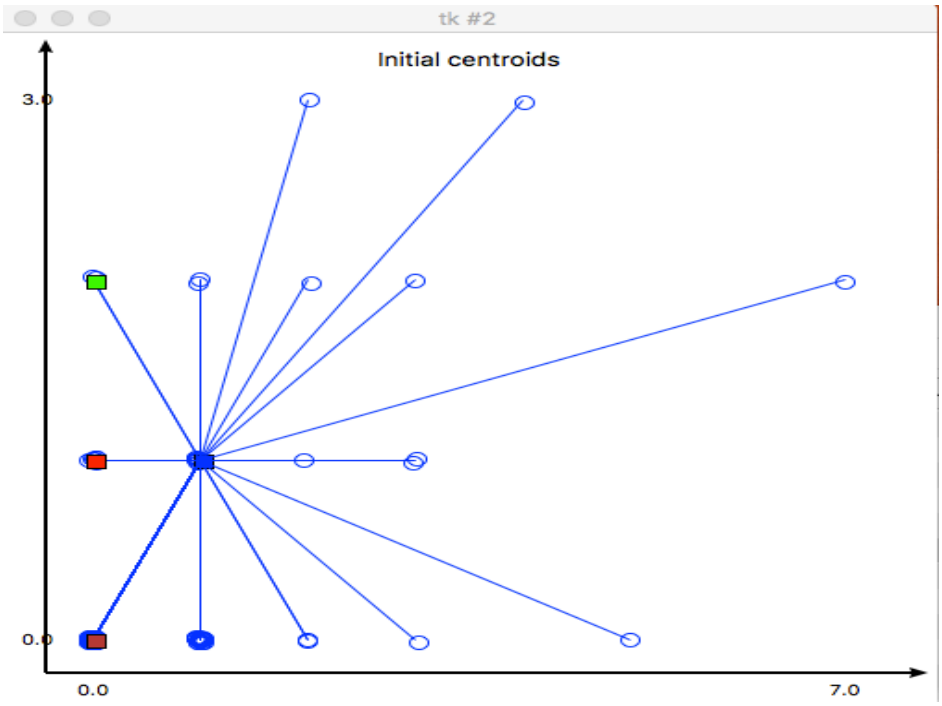
7. Data Analysis: The data analysis section is divided in 5 subsections:

- KMeans algorithm with Eculidean distance.
- KMeans algorithm with Cosine distance.
- KMeans algorithm with Jaccard distance.
- KMeans algorithm original.

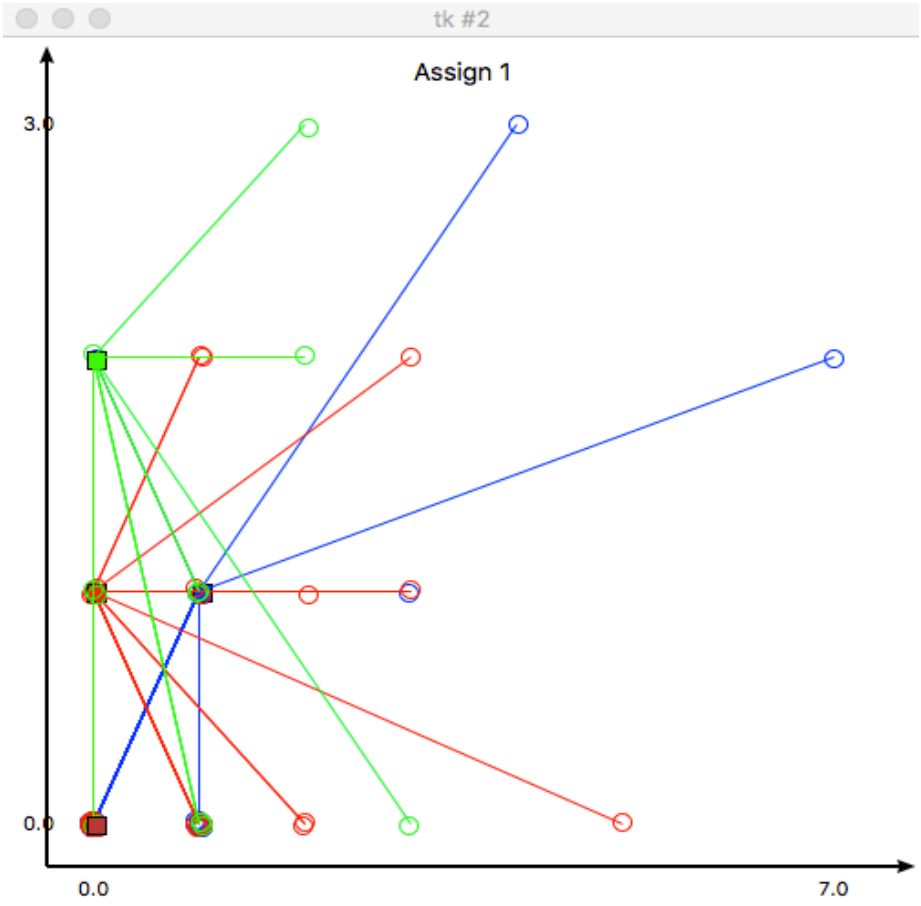
a. *KMeans algorithm*: The original KMeans algorithm converges in 6 iterations.

- Below is the graphical representation of cluster formation:

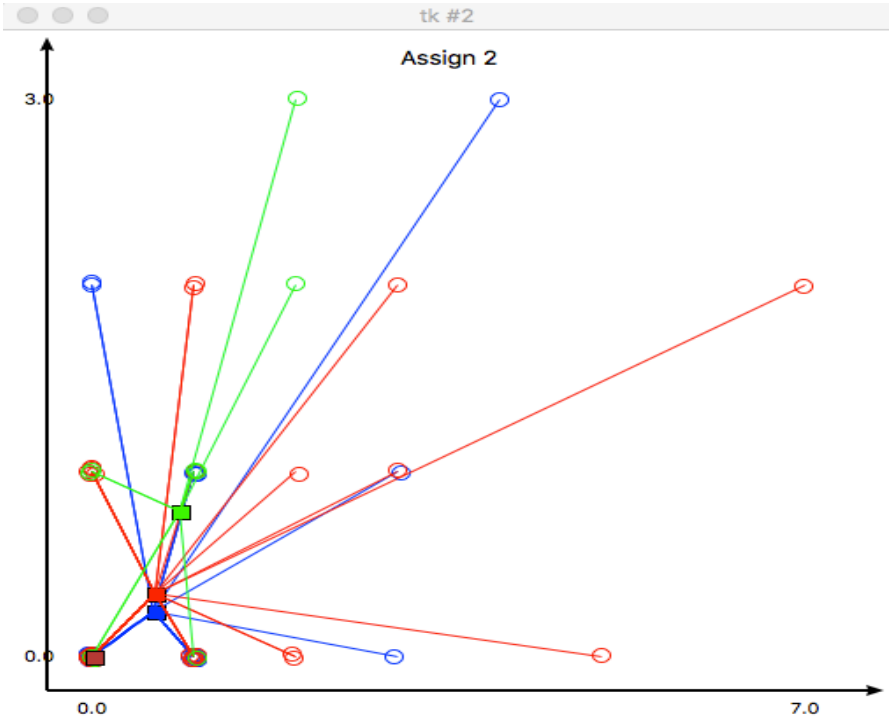
- Initial Centroids:



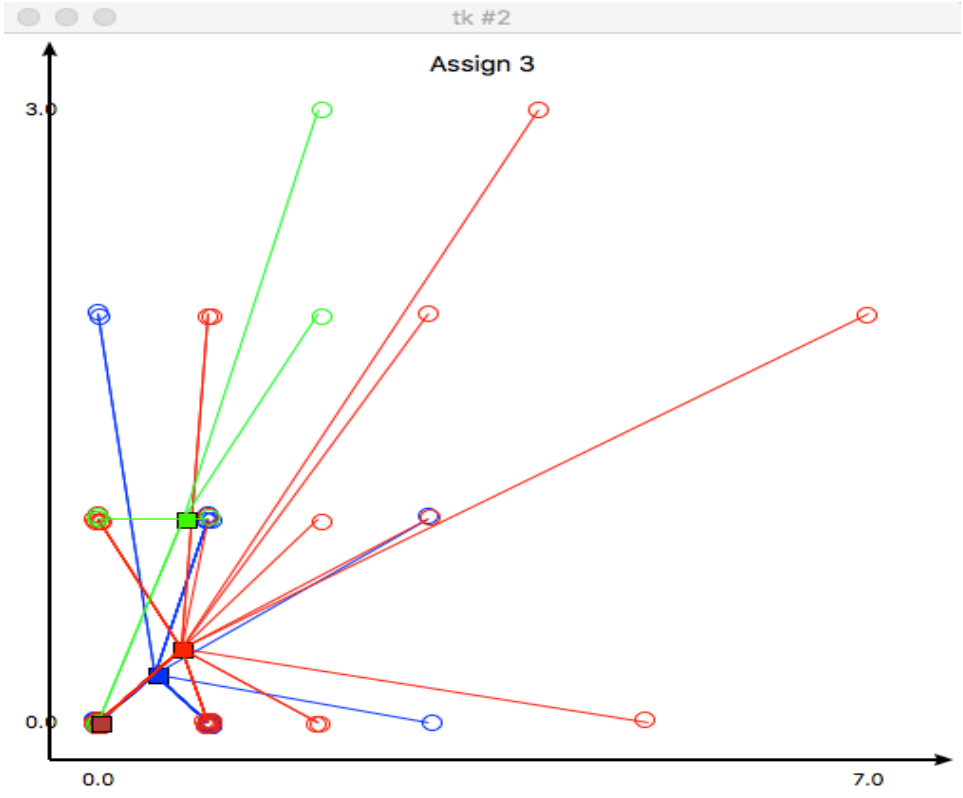
- First iteration depicting assignment of data points to centroids:



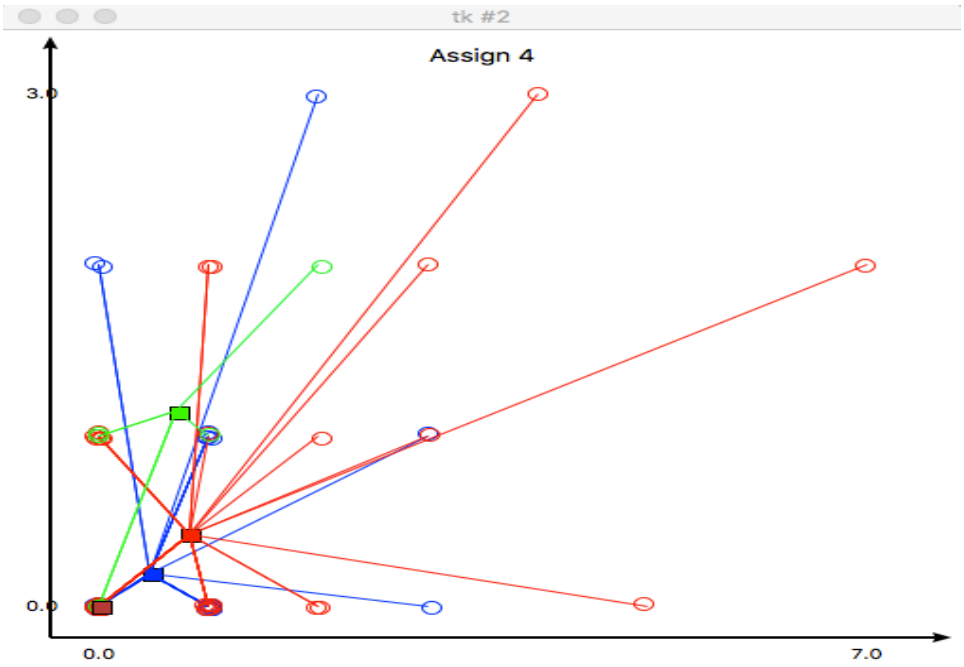
- Second iteration depicting assignment of data points to centroids:



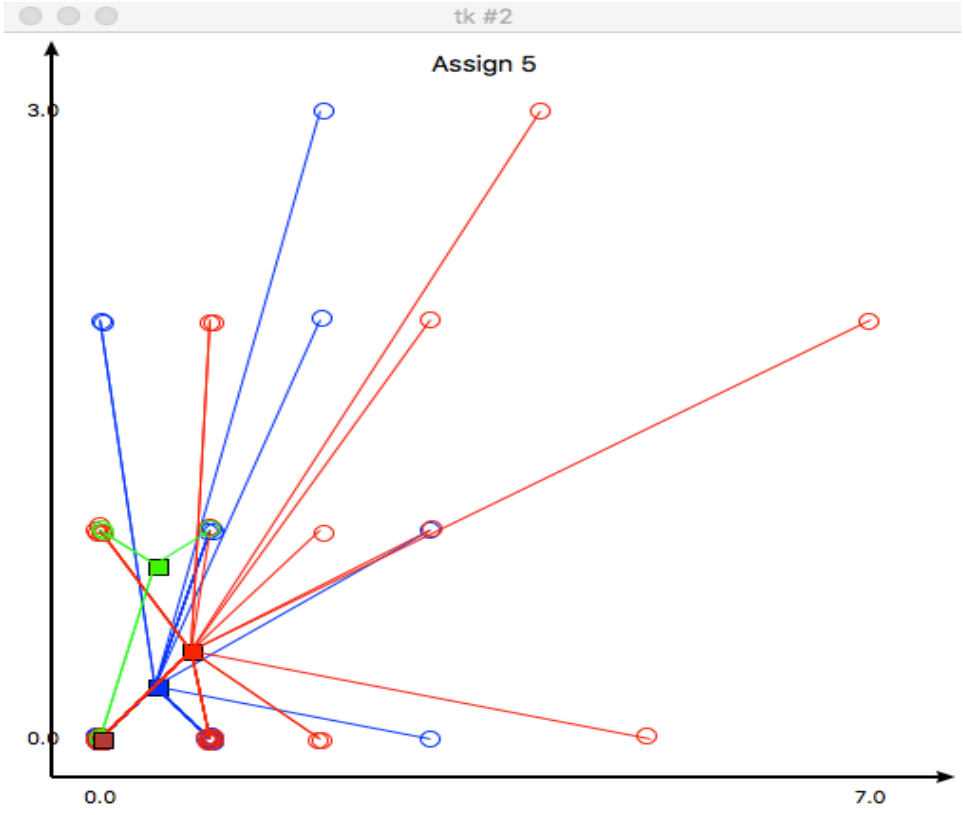
- Third iteration depicting assignment of data points to centroids:



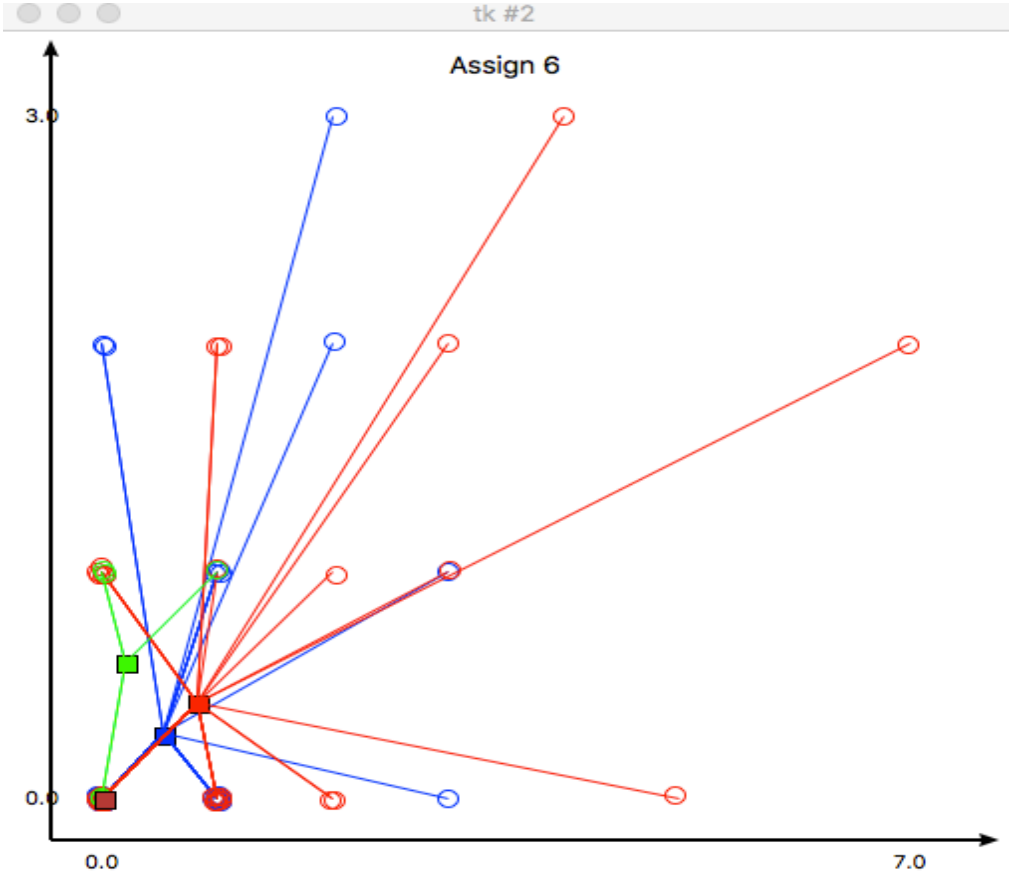
- Fourth iteration depicting assignment of data points to centroids:



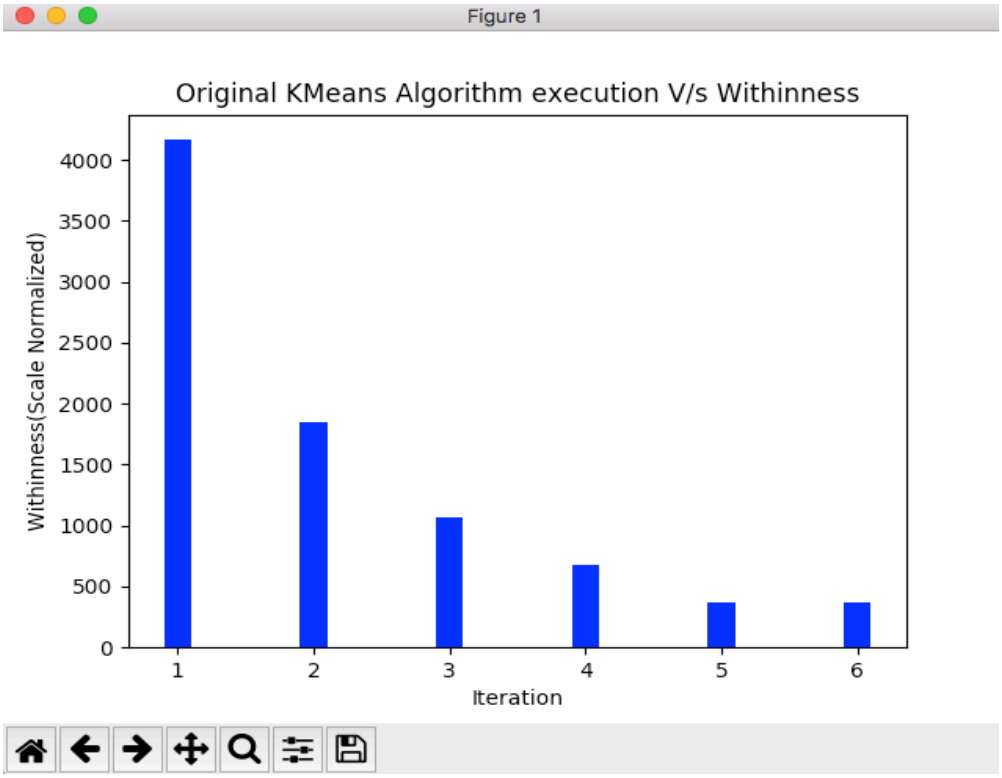
- Fifth iteration depicting assignment of data points to centroids:



- Sixth iteration depicting assignment of data points to centroids:



- The **withinness calculation** for Original KMeans is displayed below:



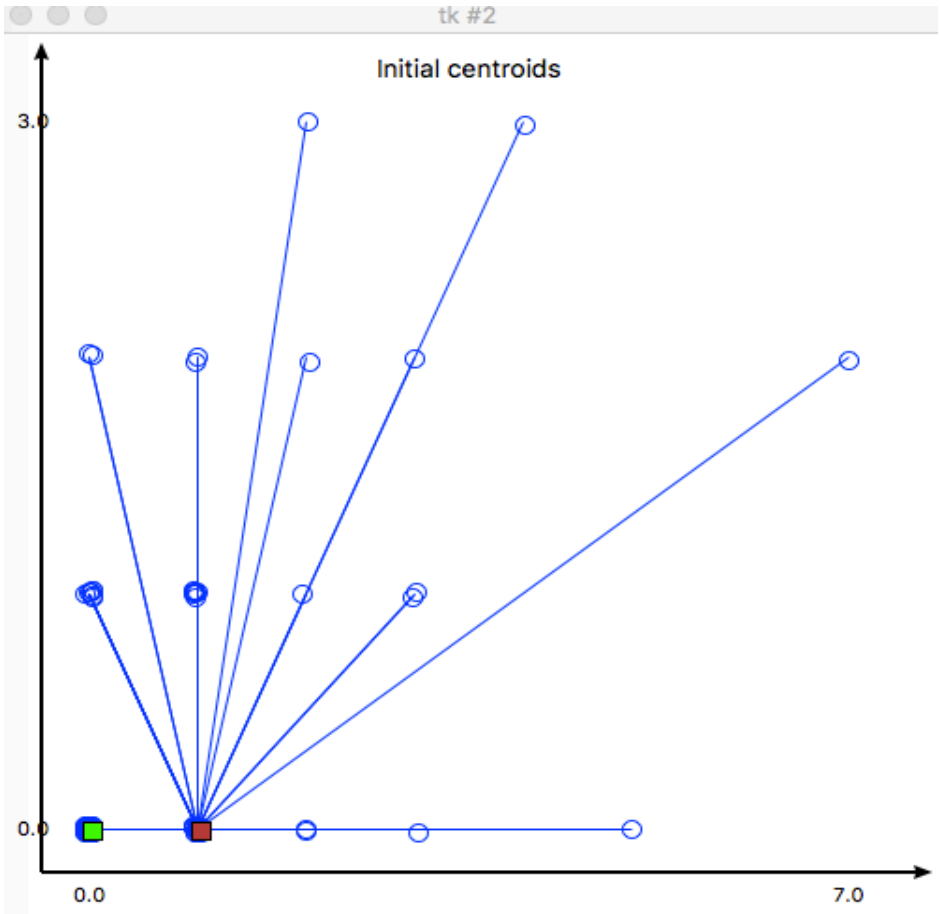
- Clearly visible from the above graph the withinness of cluster formation decreases along the successive assignments.
- Centroid allocation: Below are the centroids allocated once the algorithm converges

```
centroid0       0.53 0.29 3.63 0.02 0.27 0.27 0.43 0.37 0.31 0.43 0.49 0.45 0.02 0.00 0.24 0.63 0.94 0.51 0.69 1.73 0.82 0.67 0.73 0.63 0.63 0.14 0.82 0.08 0.
22 0.08 0.43 0.00 1.51 0.39 0.98 0.35 1.10 0.14 0.14 0.45 0.02 0.10 0.31 0.67 0.82 0.33 0.59 0.37 0.47 0.18 0.61 0.61 0.69 0.35 1.27 1.10 1.80 0.18 0.71 0.69
0.78 0.39 0.39 1.98 0.80 0.41 0.41 0.35 0.12 0.29 0.71 0.20 0.02 0.43 0.90 1.27 0.24 0.78 0.33 0.39 0.24 0.35 0.31 0.27 2.39 0.10 0.02 0.31 0.00 0.20 0.16 0.0
2 0.08 0.06 0.27 0.59 0.71 0.16 0.08 0.12 0.29 0.27 0.12 0.10 0.06 0.47 0.53 0.08 0.10 1.04 0.73 1.45 1.00 0.55 0.49 0.53 1.12 2.00 2.41 0.80 0.73 0.45 0.63 1
.73 0.63 2.69 2.47 2.51 0.02 0.37 7.84 3.59 0.37 0.14 0.08 0.35 0.16 0.12 0.98 0.27 0.37 0.29 0.94 0.22 0.10 0.08 0.16 0.00 0.27 0.00 0.76 0.31 2.29 0.16 0.69
 0.37 0.18 0.90 0.14 0.31 0.55 0.04 0.14 0.14 0.35 0.71 0.08 0.04 0.33 0.43 0.27 0.06 0.00 0.43 0.24 0.04 0.49 0.27 0.16 0.14 0.53 0.55 0.31 0.29 1.06 1.37 0.
37 1.08 1.27 0.51 0.27 0.31 0.04 0.10 0.33 0.47 0.12 0.45 0.16 0.16 0.63 0.37 0.63 0.29 0.33 0.84 0.27 0.80 0.51 0.08 0.80 0.61 1.08 0.82 0.57 0.24 0.20 0.20
0.10 0.33 0.76 0.61 0.08 0.22 0.41 0.10 0.73
centroid1       0.82 0.42 1.96 0.78 2.56 2.56 3.47 1.09 2.04 2.09 2.11 2.53 0.20 0.13 0.64 1.00 1.69 1.00 1.51 4.16 1.44 0.80 0.93 1.13 0.96 0.40 1.33 0.33 0.
80 1.22 0.98 0.33 0.58 0.53 1.09 0.98 2.11 0.04 0.42 0.44 0.38 0.16 0.67 1.78 0.51 0.27 0.80 0.27 0.31 0.87 0.71 0.27 1.44 0.20 1.53 0.82 1.47 0.58 0.36 0.62
2.04 0.33 0.58 1.02 1.22 0.22 0.56 1.18 0.36 0.44 0.42 0.04 0.09 0.60 0.67 0.22 0.44 1.64 0.09 0.33 0.53 0.67 0.27 0.71 0.24 0.71 0.22 0.40 0.27 0.11 0.40 2.4
9 0.49 0.33 0.13 1.02 0.62 0.60 0.53 0.62 0.53 2.22 0.04 0.13 0.04 0.96 0.42 0.09 0.04 1.02 1.16 1.02 2.11 1.22 0.98 1.62 1.07 0.98 1.91 1.76 1.20 1.07 1.40 2
.11 1.69 1.16 1.00 1.18 0.11 0.09 0.60 2.22 0.33 0.56 0.04 0.13 0.02 0.36 0.20 0.02 0.71 0.20 0.09 0.09 0.13 0.47 0.31 0.00 0.33 0.00 0.18 0.47 0.33 0.44 0.13
 0.51 0.07 0.18 0.38 0.02 0.13 0.00 0.38 0.00 0.02 0.09 0.00 0.04 0.00 0.04 0.07 0.31 0.11 0.02 0.02 0.04 0.44 0.71 0.24 0.13 0.07 0.58 0.51 0.38 1.02 1.42 0.
16 0.76 2.11 0.56 0.84 0.16 0.22 0.49 0.09 0.11 0.36 0.27 0.09 0.27 0.73 0.22 0.64 0.09 1.42 1.20 0.53 0.18 0.40 0.00 0.84 1.40 1.80 1.82 0.42 0.51 0.18 0.16
0.38 0.69 0.29 0.18 0.18 0.58 0.44 0.02 0.49
centroid2       0.20 0.60 13.00 0.00 0.00 0.00 0.00 0.00 1.20 0.00 0.00 0.00 0.00 0.00 0.20 1.00 1.00 1.00 1.20 2.40 1.60 3.00 2.20 2.40 2.40 0.20 0.40 0.60 1
.00 1.00 0.80 0.20 0.20 0.00 1.00 1.00 3.20 0.00 0.60 2.20 1.40 0.60 1.40 3.40 3.60 0.20 0.00 4.80 2.40 0.20 0.00 6.80 2.20 0.60 3.80 1.40 3.20 0.60 2.20 0.20
 4.20 0.80 0.20 1.00 2.40 1.40 0.00 0.60 0.00 0.60 1.40 0.00 0.60 1.40 4.00 6.40 0.20 2.40 1.40 0.20 0.00 1.60 0.80 1.00 1.60 0.20 0.00 0.20 0.00 0.00 0.00 0.
00 0.00 0.00 0.60 1.80 0.60 0.40 0.20 0.00 0.20 0.80 0.60 1.00 0.00 1.40 0.60 0.60 0.80 0.80 6.00 0.80 1.60 1.00 1.20 0.80 0.80 1.20 1.60 0.80 2.00 0.80 3.60
2.00 1.40 1.40 1.00 1.20 6.20 4.40 37.00 4.00 18.00 1.60 1.40 20.60 2.20 0.60 14.20 15.20 1.80 2.20 8.80 2.60 1.40 2.60 2.00 0.20 1.20 1.20 2.80 1.20 1.40 1.6
0 0.20 1.20 0.40 1.40 0.60 0.20 2.00 1.00 0.60 0.40 1.00 0.40 0.80 0.40 1.00 2.20 0.80 0.60 0.60 0.80 1.40 1.40 2.60 3.80 0.60 1.40 2.60 3.00 1.80 0.40 2.40 2
.00 0.40 0.80 2.00 1.40 1.00 2.20 0.80 1.80 0.20 0.80 0.20 1.20 0.40 2.20 2.00 1.00 1.00 1.00 0.40 3.00 0.20 0.20 1.60 1.00 2.00 1.40 3.00 2.80 0.40 1.40 1.20
 1.00 0.20 0.60 0.60 0.40 1.20 1.00 0.40 0.20 1.00
centroid3       0.00 0.00 0.00 2.00 3.00 3.00 3.00 1.00 2.00 2.00 2.00 2.00 0.00 0.00 0.00 1.00 2.00 1.00 1.00 3.00 2.00 0.00 2.00 2.00 2.00 0.00 4.00 0.00 0.
00 0.00 1.00 0.00 1.00 1.00 1.00 30.00 3.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00 3.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 4.00 1.00
 3.00 0.00 22.00 1.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00 2.00 1.00 0.00 3.00 2.00 0.00 0.00 1.00 1.00 1.00 0.00 0.00 0.00 0.00 0.00 0
.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 1.00 1.00 8.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 1.00 1.00 2.00 1.00 1.00 1.00 1.00 1.00 2.00 2.00 1.00 1.00 1.00
 2.00 1.00 1.00 1.00 1.00 0.00 0.00 0.00 1.00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 0.
00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 2.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 2.00 0.00 0.00 0.00 0.00 0.00 3.00 0.00 8.00
0.00 1.00 2.00 0.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 2.00 2.00 0.00 2.00 0.00 0.0
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

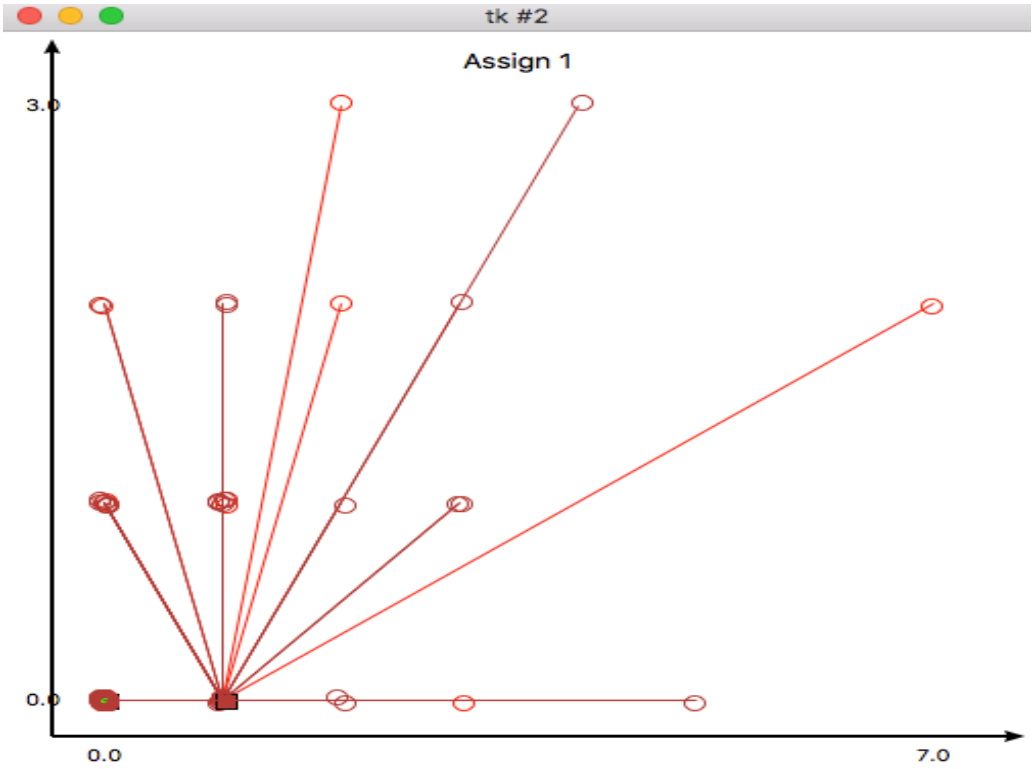
b. *KMeans algorithm with Euclidean:* The Euclidean distance is used as distance metric with KMeans algorithm. Using euclidean as the distance metric KMeans algorithm converges in 8 iterations.

- Below is the graphical representation of cluster formations:

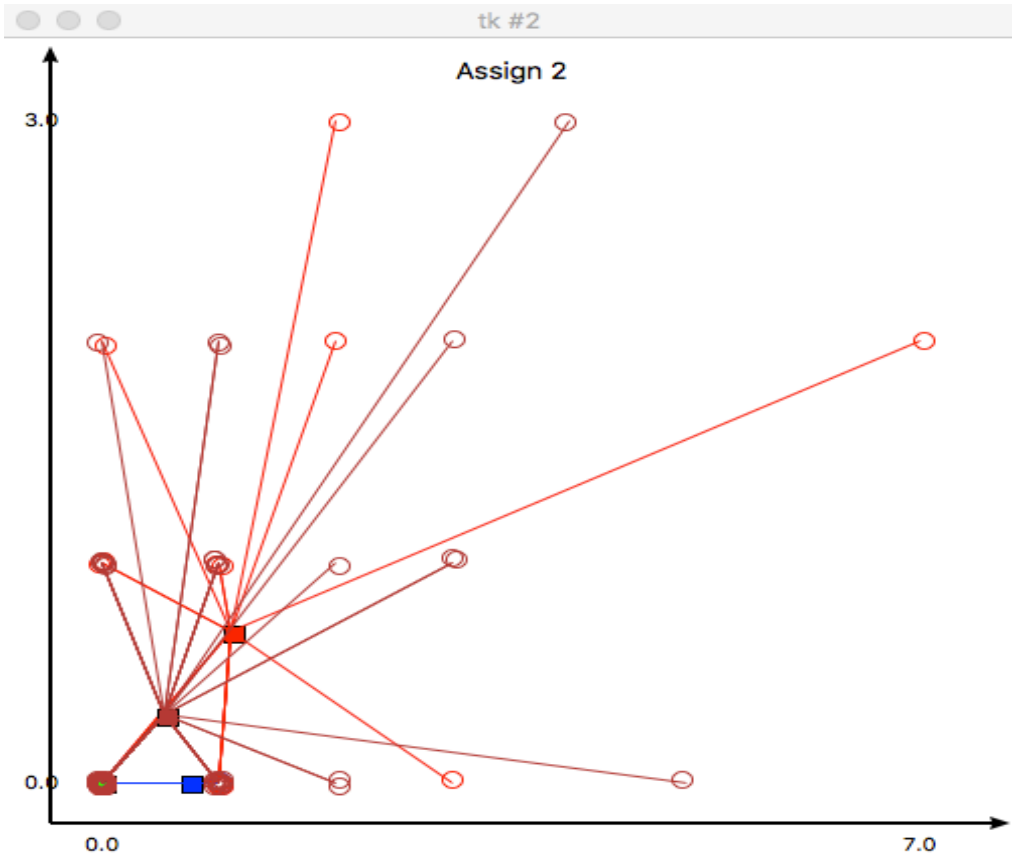
- Initial Centroids:



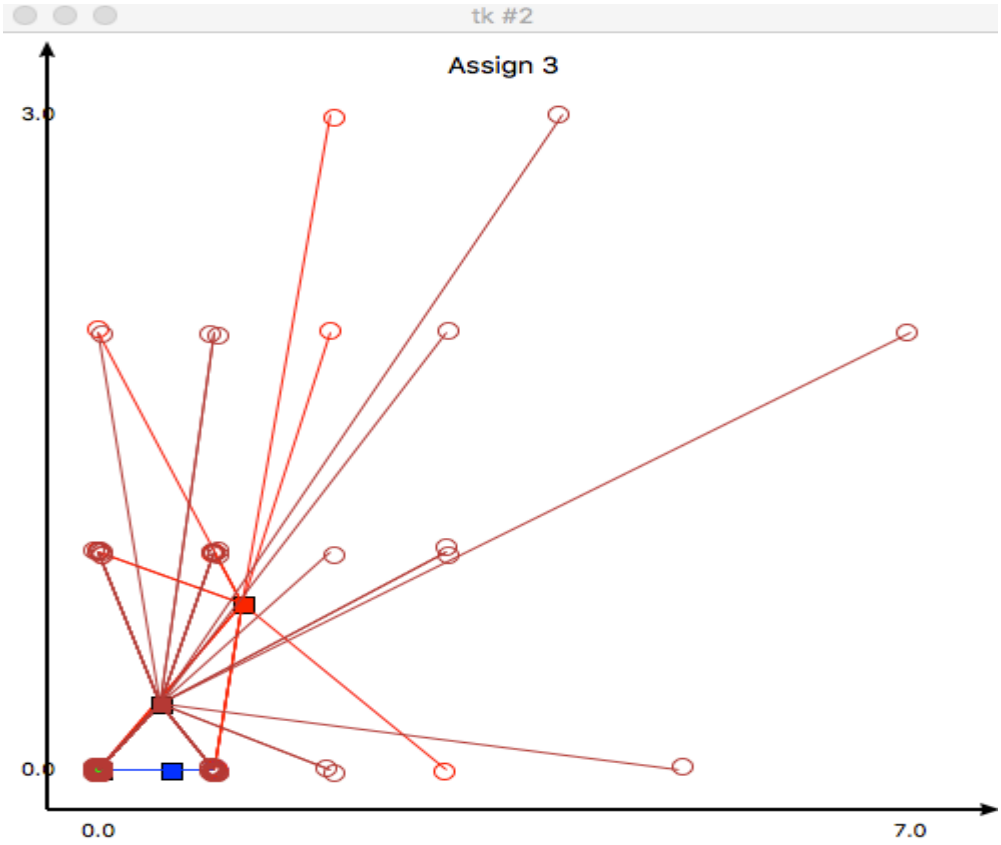
- First iteration depicting assignment of data points to centroids:



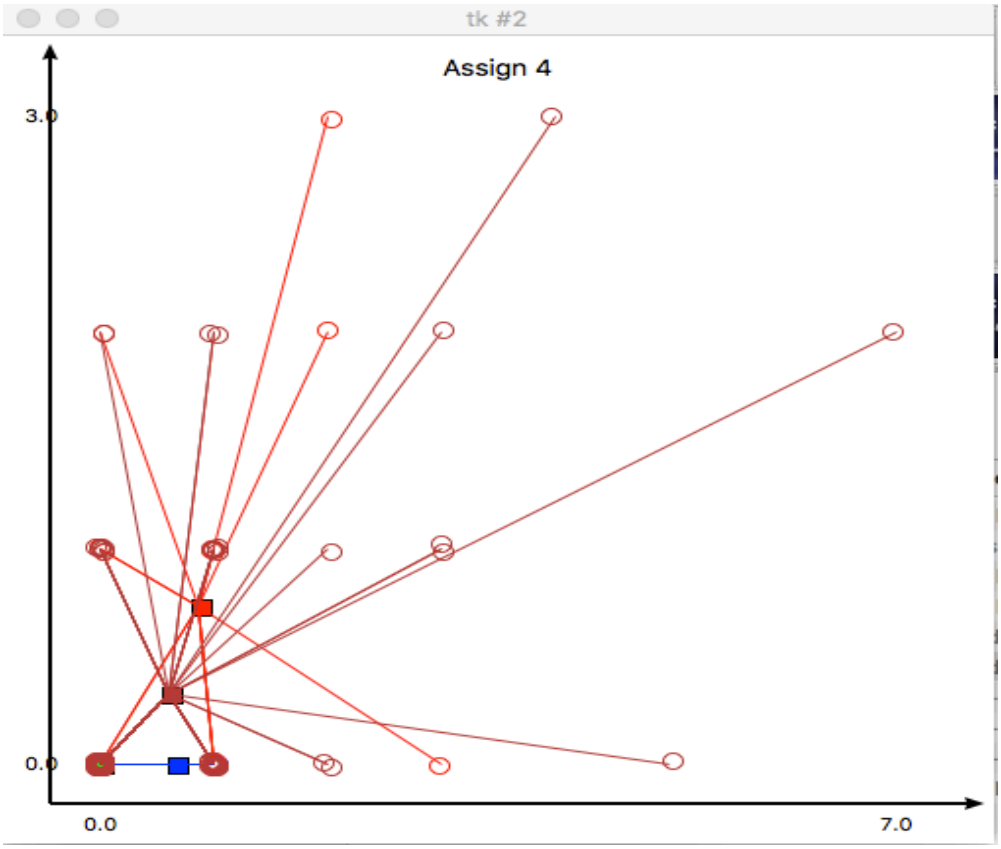
- Second iteration depicting assignment of data points to centroids:



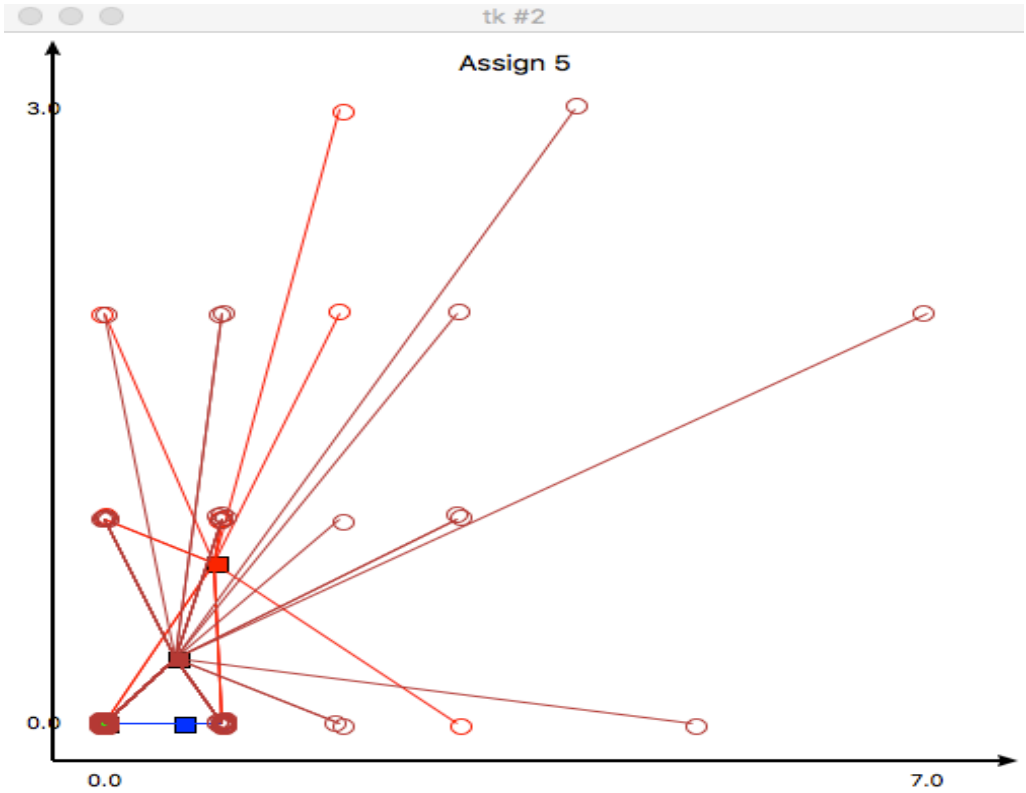
- Third iteration depicting assignment of data points to centroids:



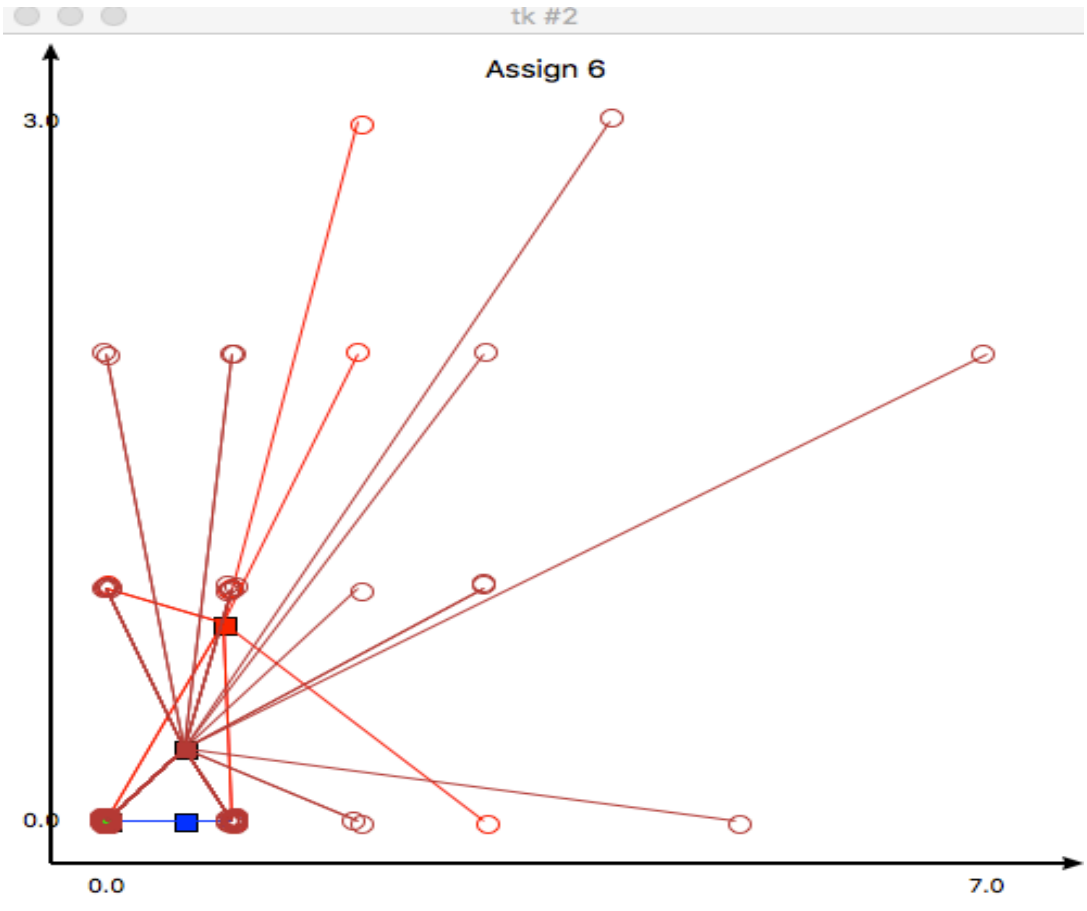
- Fourth iteration depicting assignment of data points to centroids:



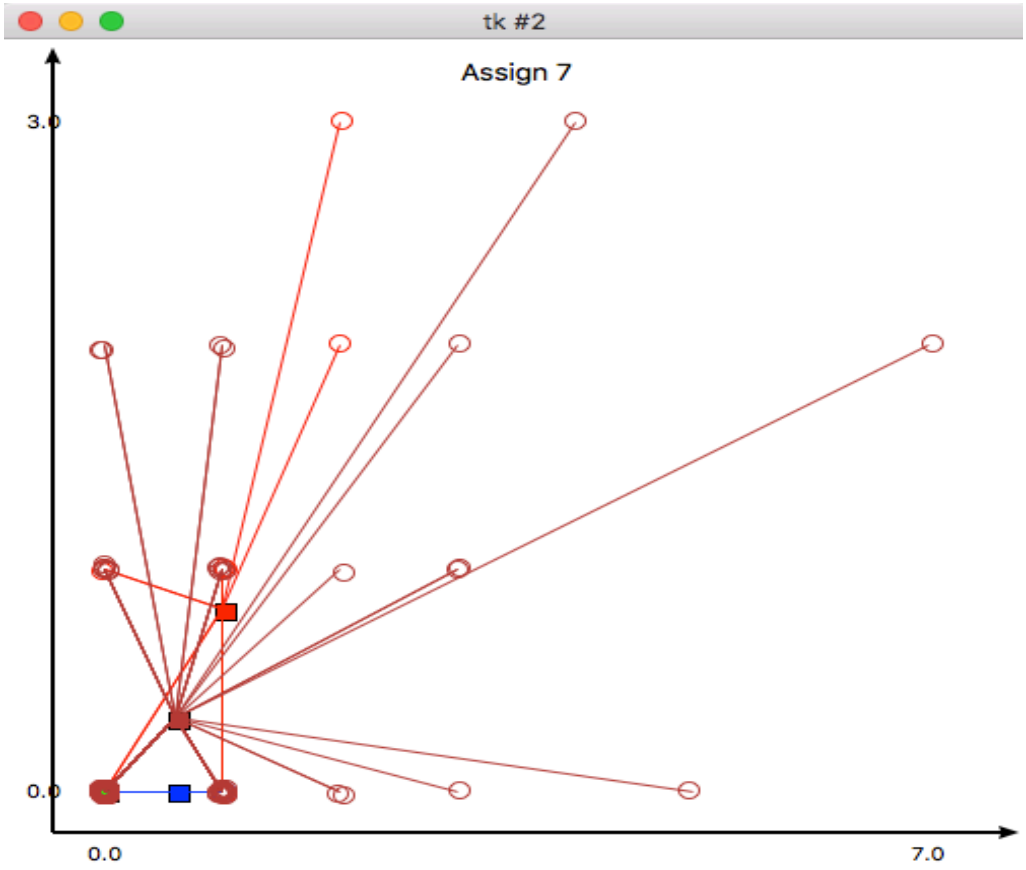
- Fifth iteration depicting assignment of data points to centroids:



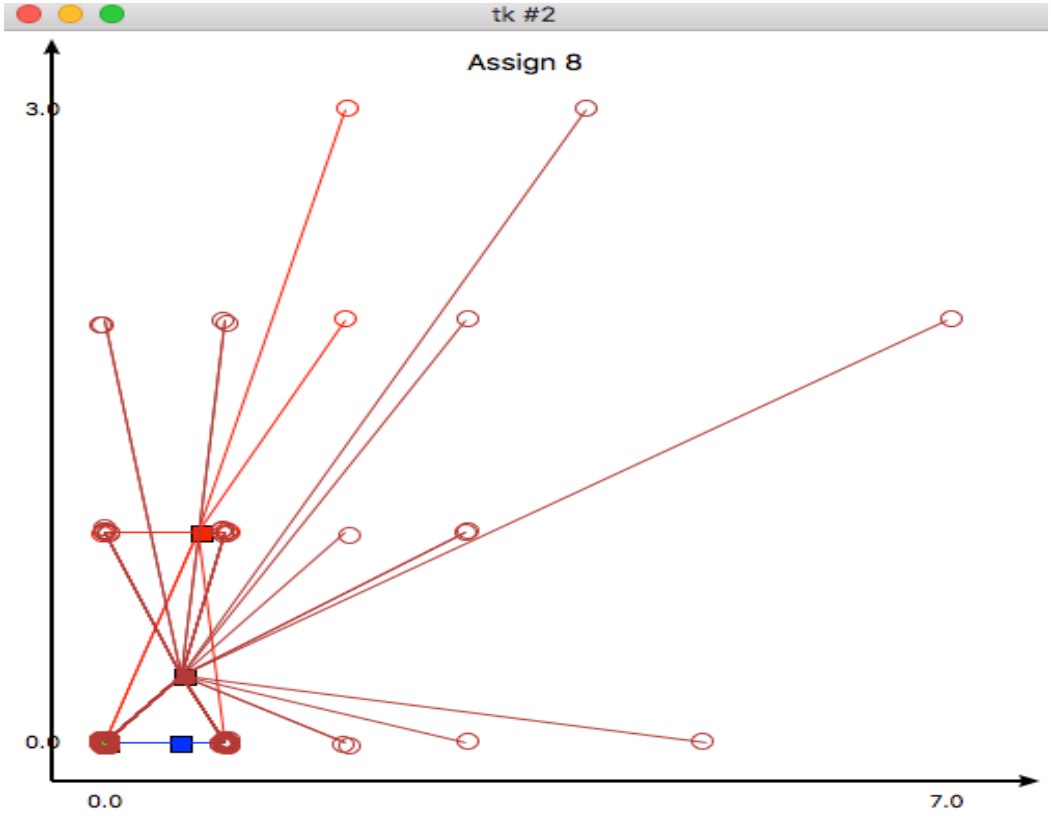
- Sixth iteration depicting assignment of data points to centroids:



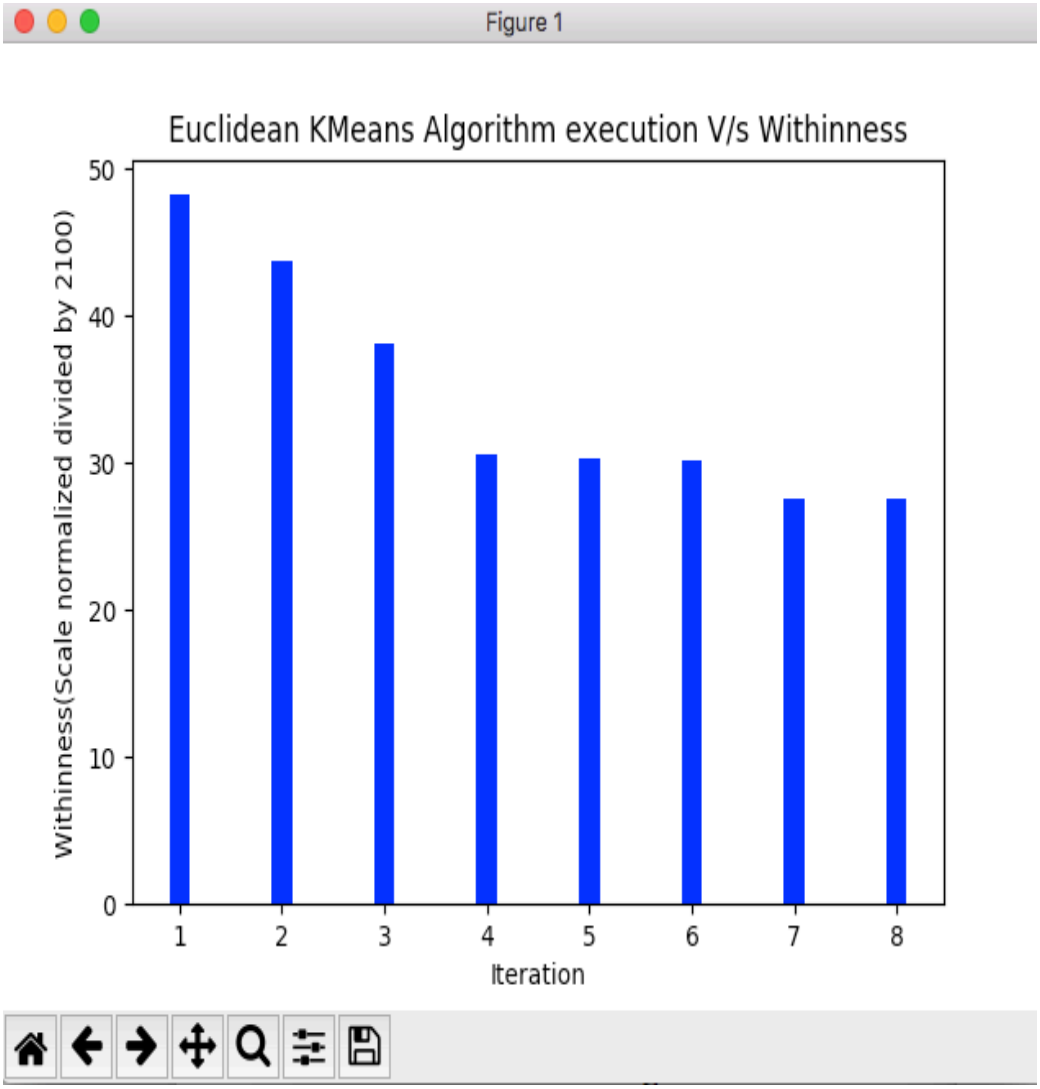
- Seventh iteration depicting assignment of data points to centroids:



- Eighth iteration depicting assignment of data points to centroids:



- The **withinness calculation** for Euclidean KMeans algorithm is displayed below:

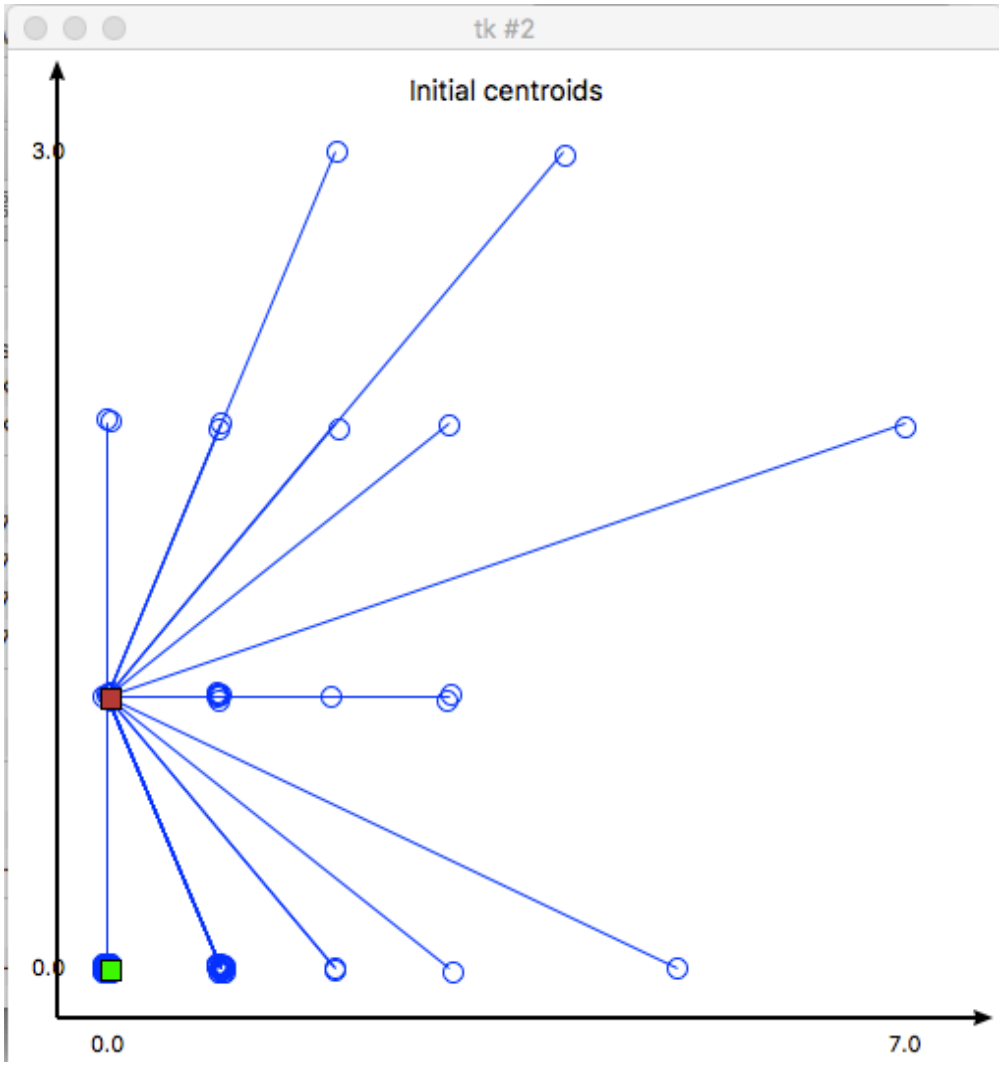


- Clearly visible from the above graph the withinness of cluster formation decreases along the successive assignments for Euclidean distance metric as well.

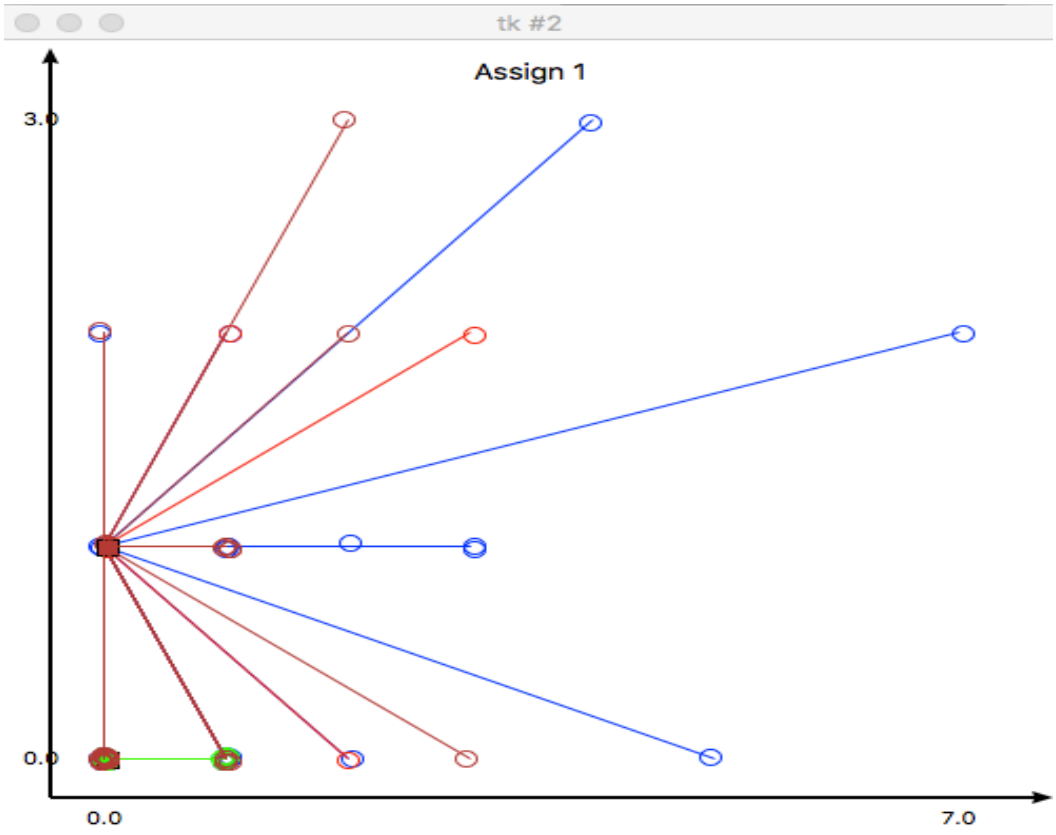
- *Centroid allocation:* Below are the centroids allocated once the algorithm converges (The text file eucdist.txt contains the centroid allocation information):

c. *KMeans algorithm with Cosine distance:* The Cosine distance is used as distance metric with KMeans algorithm. Using Cosine as the distance metric KMeans algorithm converges in 6 iterations.
   - Below is the graphical representation of cluster formations:

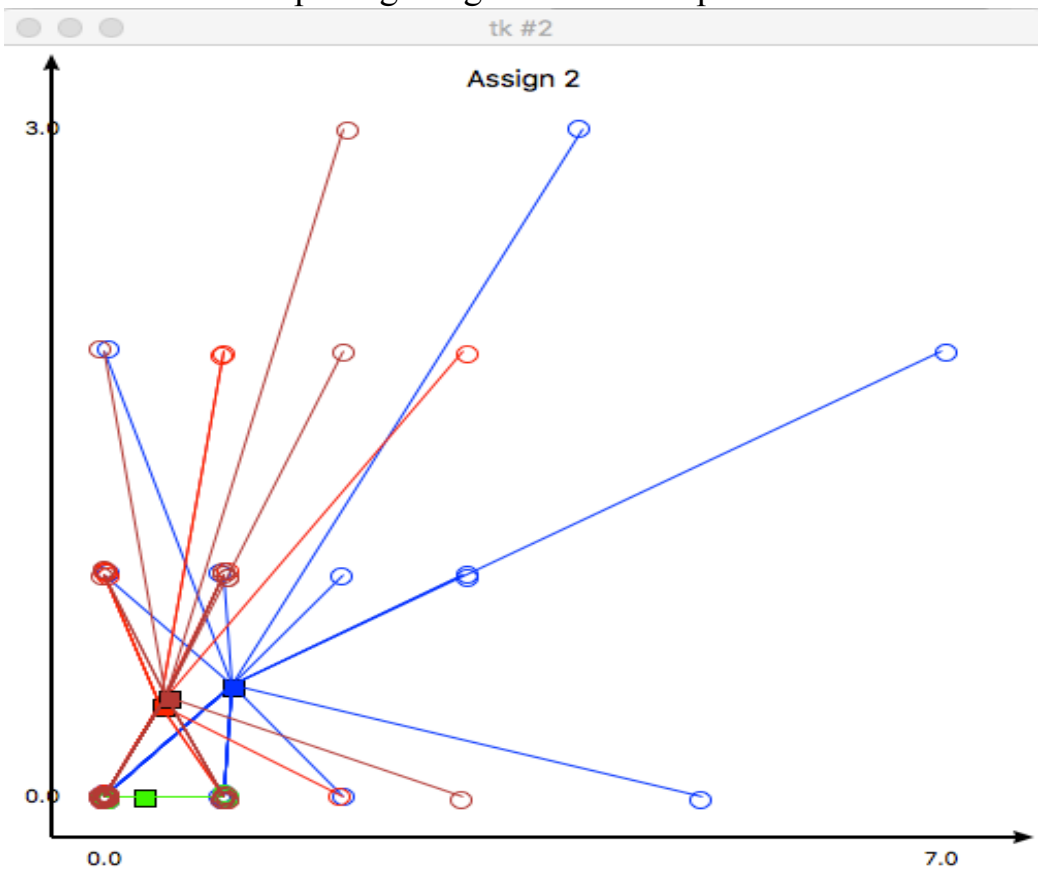
   - Initial Centroids:



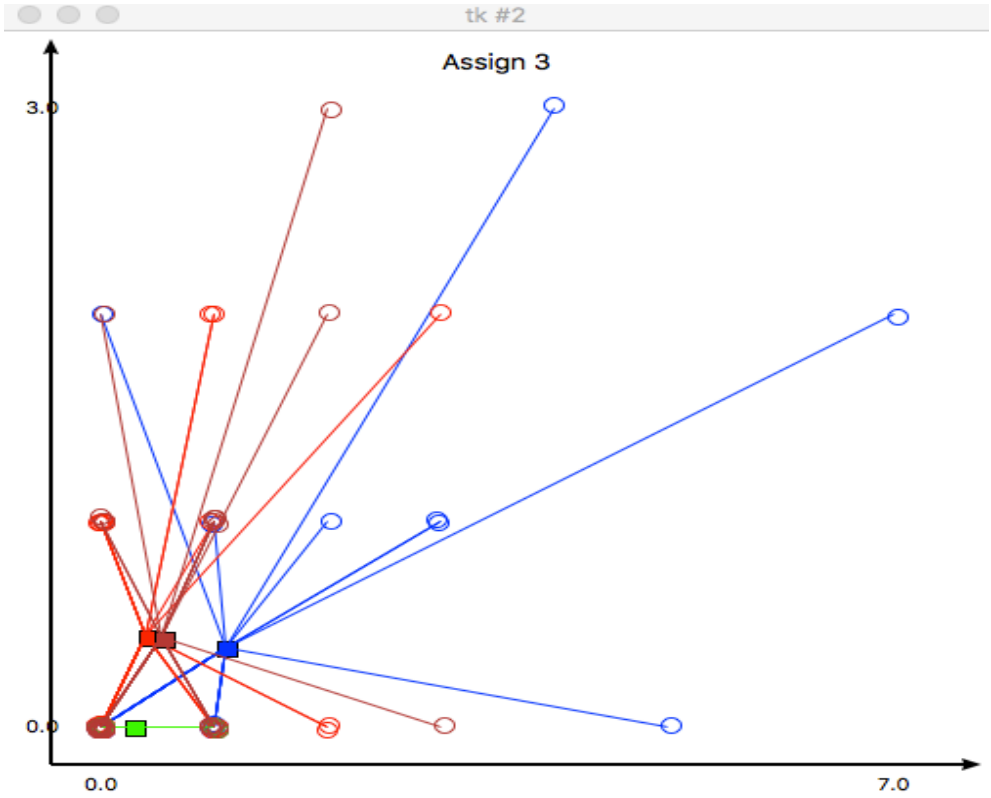
- First iteration depicting assignment of data points to centroids:



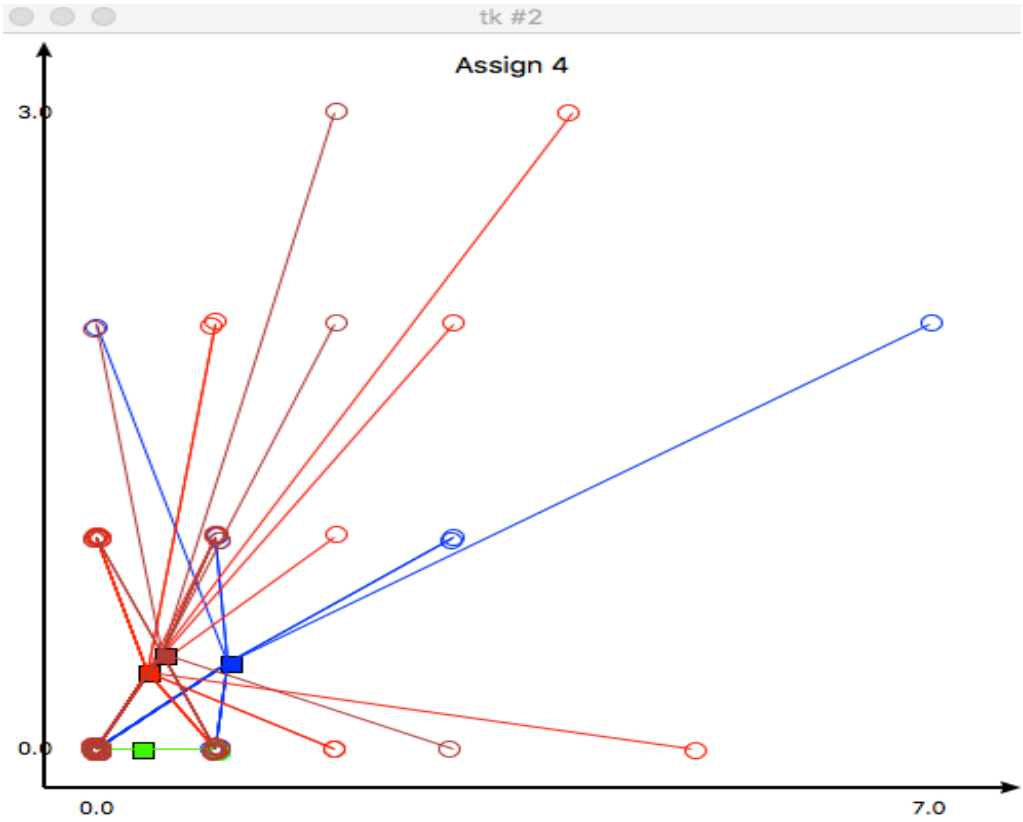
- Second iteration depicting assignment of data points to centroids:



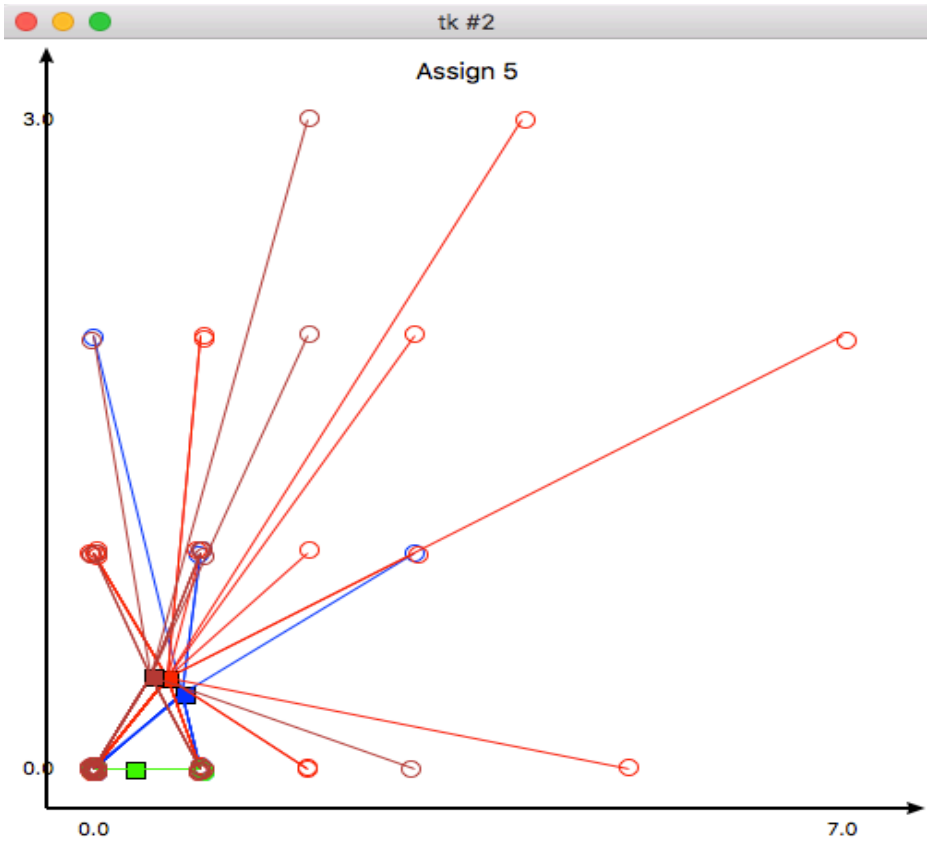
- Third iteration depicting assignment of data points to centroids:



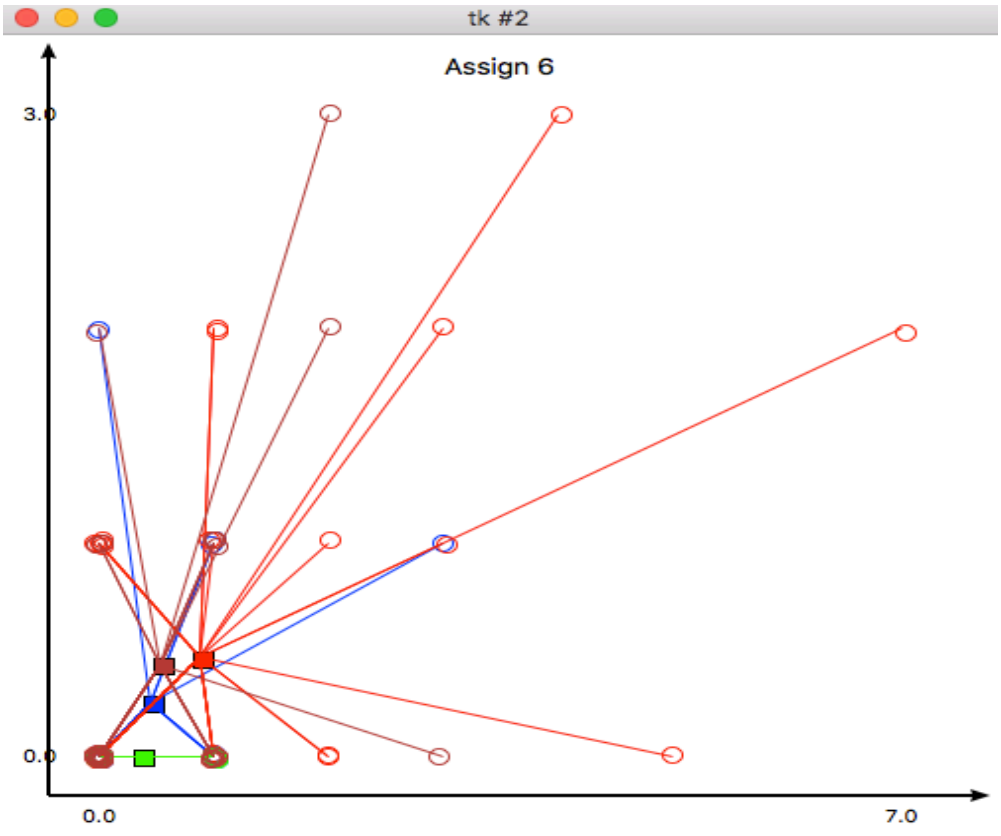
- Fourth iteration depicting assignment of data points to centroids:



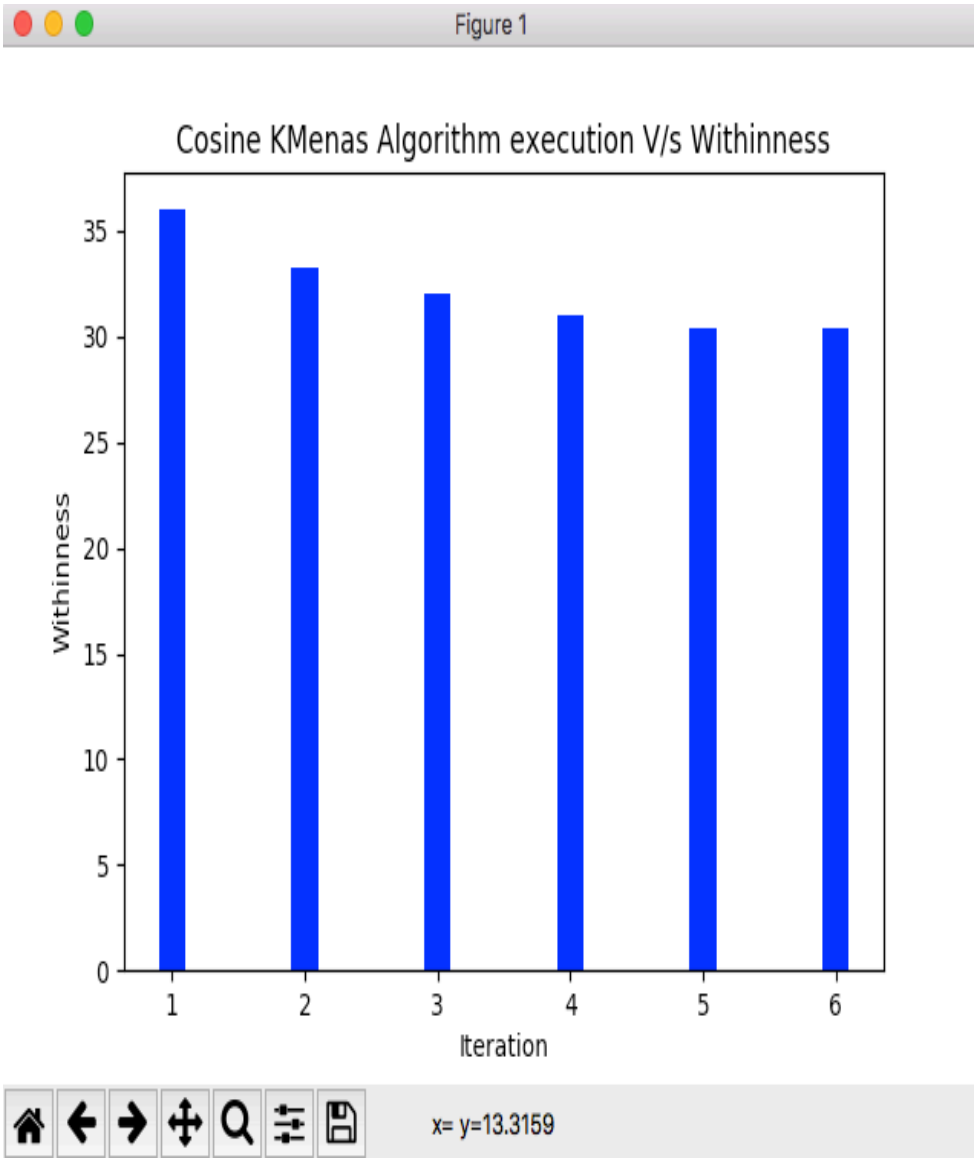
- Fifth iteration depicting assignment of data points to centroids:



- Sixth iteration depicting assignment of data points to centroids:

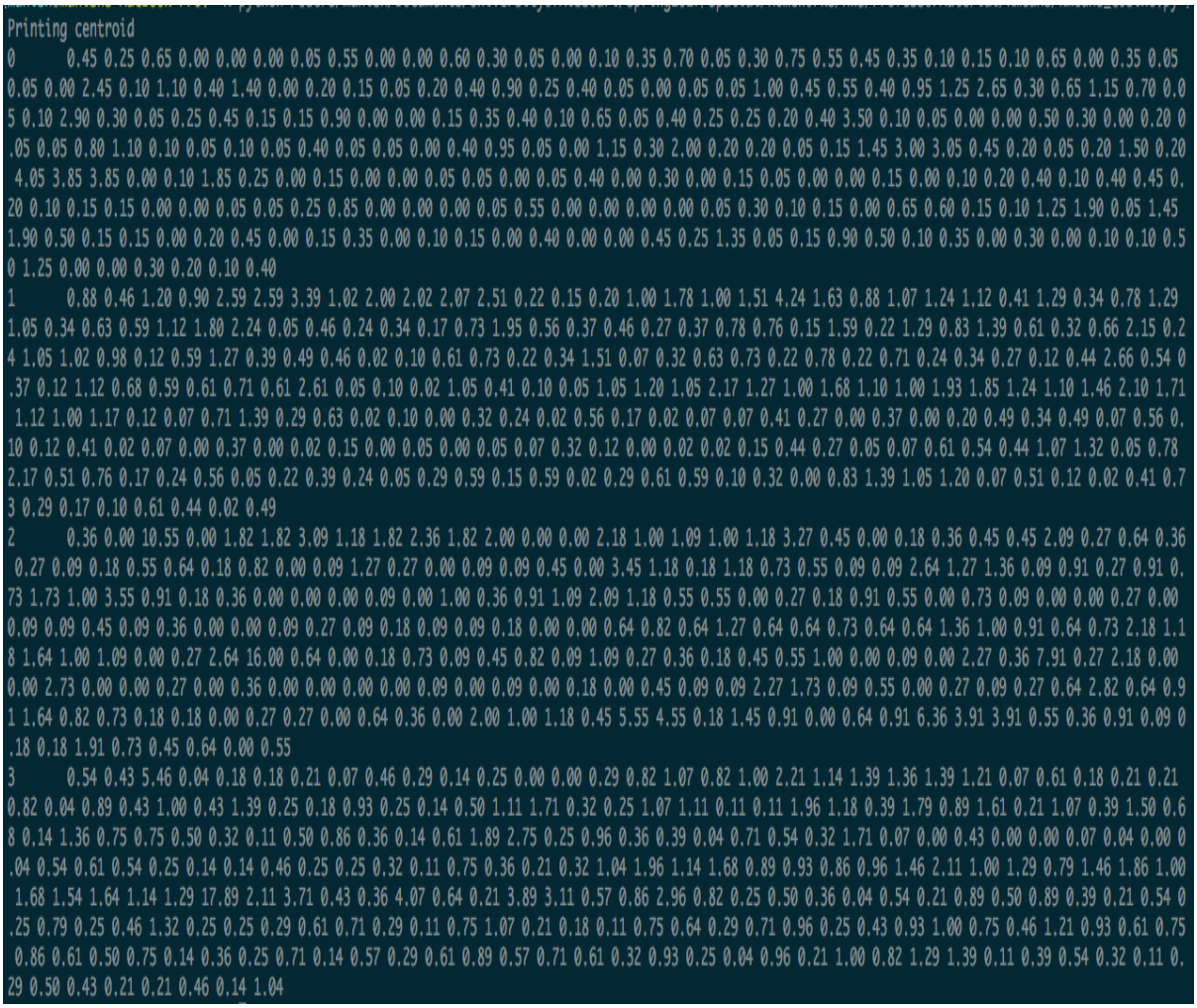


- The **withinness calculation** for Cosine KMeans algorithm is displayed below:



- Clearly visible from the above graph the withinness of cluster formation decreases along the successive assignments for Cosine distance metric as well.

- *Centroid allocation:* Below are the centroids allocated once the algorithm converges (The text file Cosine_Points.txt contains the centroid allocation information):

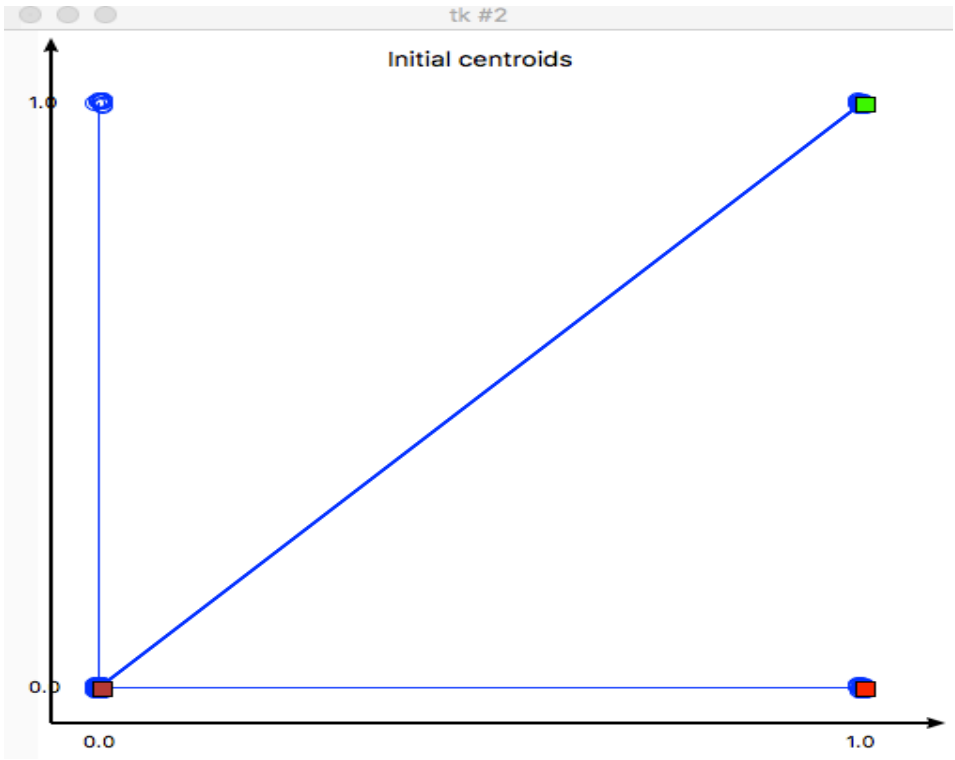


d. *KMeans algorithm with Jaccard distance:* The Jaccard distance is used as distance metric with KMeans algorithm. Using Jaccard as the distance metric KMeans algorithm converges in 3 iterations, however the results are not at par with other algorithms.

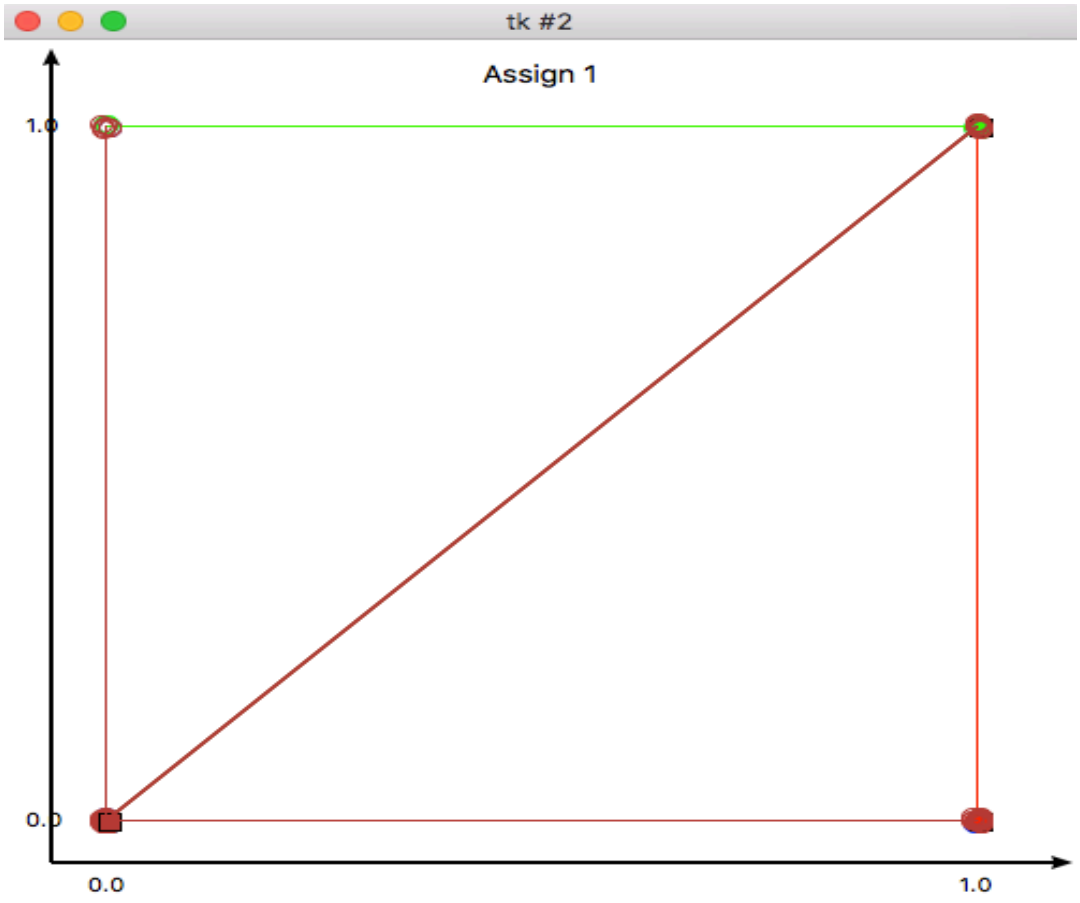
NOTE – For jaccard distance calculation we used similarity matric rather than the distance matric as the jaccard similarity works on presence or absence of data.

- Below is the graphical representation of cluster formations:

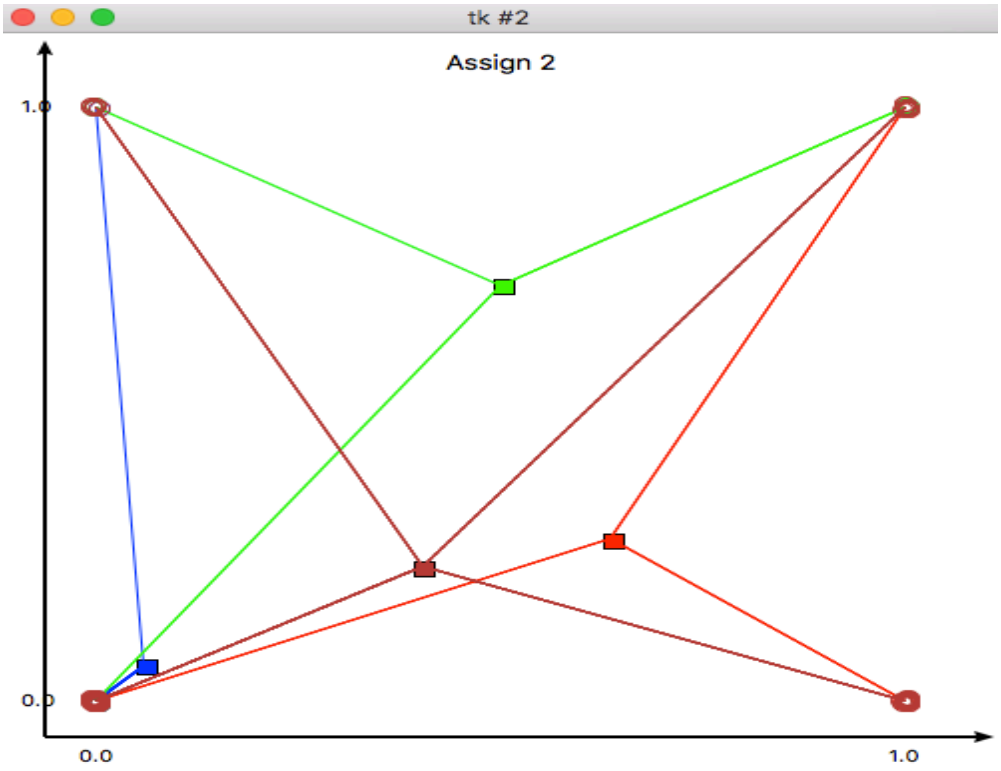
- Initial Centroids:



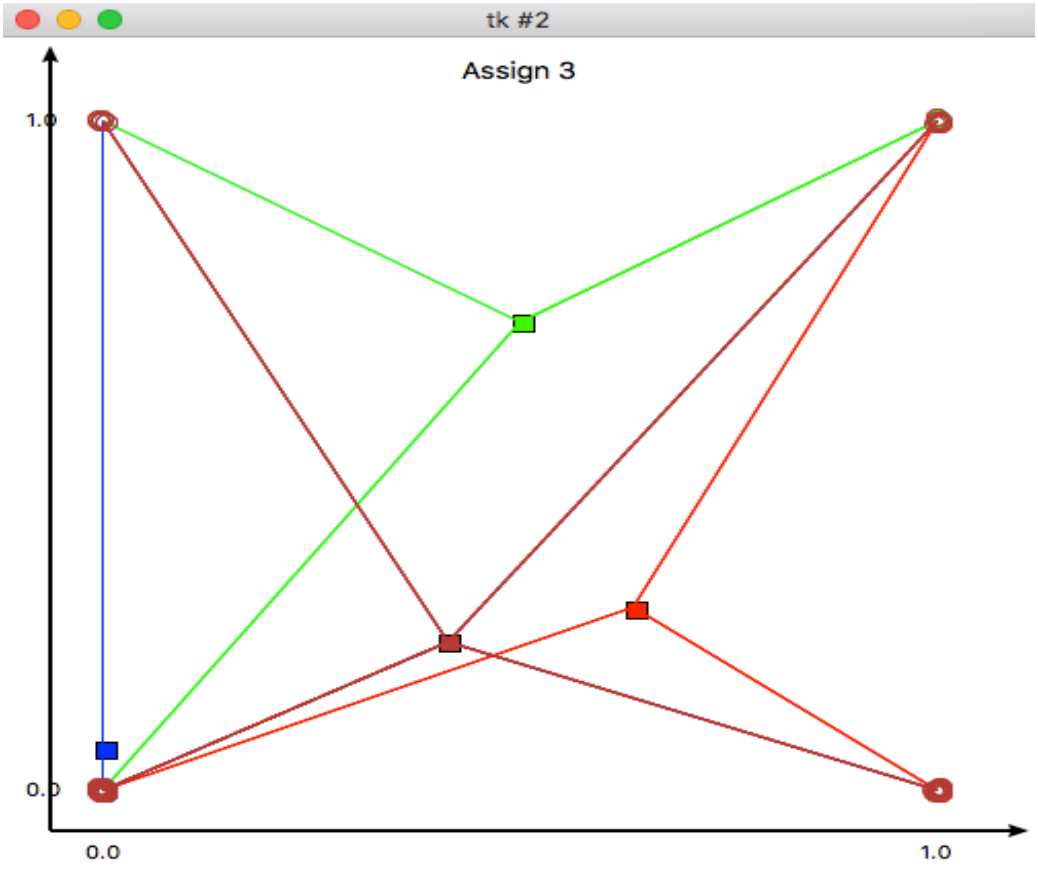
- First iteration depicting assignment of data points to centroids:



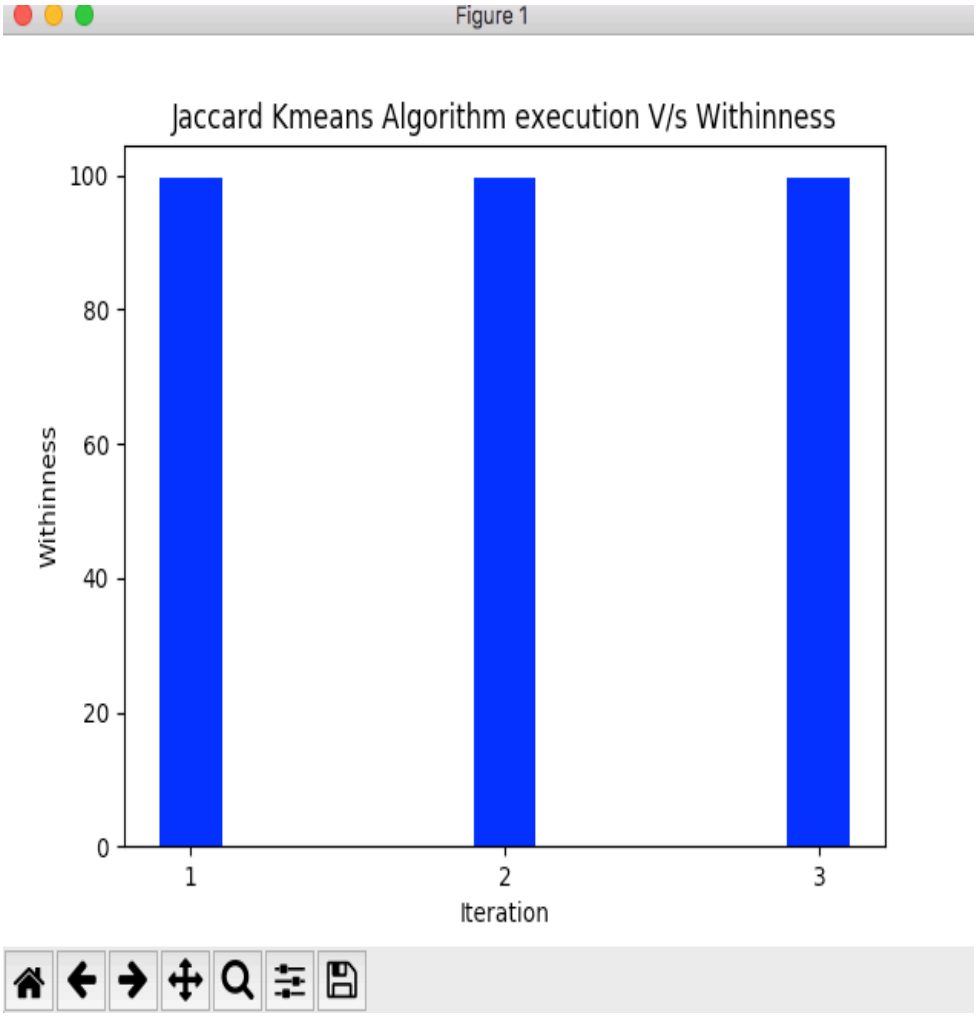
- Second iteration depicting assignment of data points to centroids:



- Third iteration depicting assignment of data points to centroids:



- The **withinness calculation** for Jaccard KMeans algorithm is displayed below:



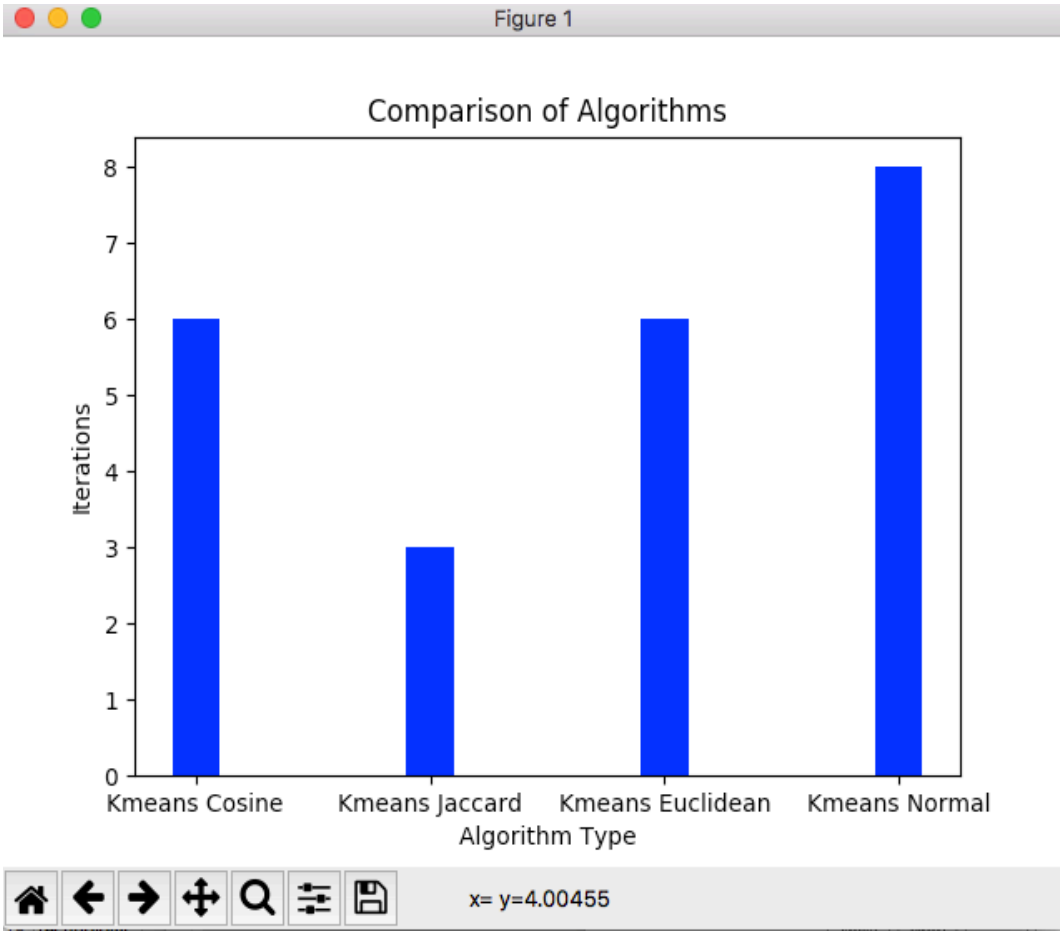
- Clearly visible from the above graph the withinness of cluster formation does not change along the successive assignments for Jaccard distance metric, indicating that jaccard distance is not a very good measure of distance when implemented with KMeans algorithm.
- Intuitively this seems to be right as well. Jaccard is a similarity matrix which calculates similarity on the basis of presence or absence of features which does not fit well within the concept of KMeans algorithm.

- *Centroid allocation:* Below are the centroids allocated once the algorithm converges (The text file jacard.txt contains the centroid allocation information):

8.Conclusion:

- On the basis of number of iterations taken to converge KMeans: The number of iterations taken for converging KMeans algorithm are similar for Cosine distance metric with Kmeans and normal Kmeans algorithm, Jaccard KMeans algorithm converges a lot earlier then all the other metric distances and Euclidean KMeans taking the largest number of iterations.

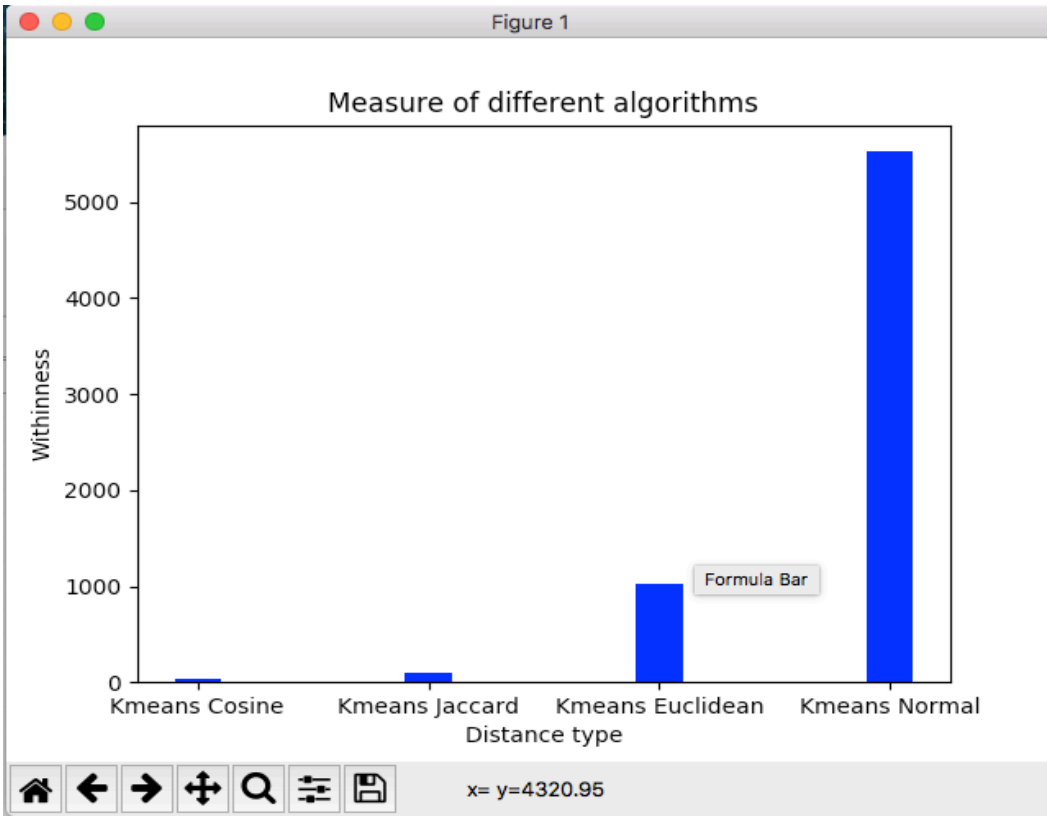
- The graphical presentation of the same is as below:



- We see the lowest withinness for Cosine metric, then for jaccard, then Euclidean and highest for kmeans_original.

- We use Sum of Squared Error formula to calculate the withinness and report it in the form of graph below. Higher SSE means lower accuracy, lowest SSE means maximum accuracy.

- Below is the graphical representation for withinness for different algorithms, the accuracy will be opposite of the below, meaning Cosine KMeans will have highest accuracy whereas Normal KMeans will have lowest accuracy.



- As per the analysis Cosine is the best distance metric to run KMeans algorithm, then comes Jaccard, then Euclidean and final is the kmeans normal algorithm