**Assignment3**

**Q1:**

1. The code is in the file *kmeans.py*
2. The code is in the file *singlelinkage.py*
3. Kmeans:

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster Number | size | Most common label | Percentage of points in cluster with that label |
| 0 | 159 | 1 | 64.1% |
| 1 | 81 | 2 | 96.2% |
| 2 | 141 | 3 | 46.8% |
| 3 | 70 | 6 | 77.1% |
| 4 | 74 | 0 | 83.7% |
| 5 | 104 | 4 | 25.0% |
| 6 | 61 | 6 | 85.2% |
| 7 | 117 | 4 | 34.1% |
| 8 | 130 | 9 | 37.6% |
| 9 | 63 | 0 | 38.0% |

If we would classify each sample based on the most common label in its cluster, then the error of each cluster is the number of samples that were misclassified.  
therefore, the total error is the number of all the samples that differ from the most common label on their cluster.

Single linkage:

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster Number | size | Most common label | Percentage of points in cluster with that label |
| 0 | 289 | 0 | 12.4% |
| 1 | 1 | 4 | 100% |
| 2 | 3 | 3 | 66.7% |
| 3 | 1 | 3 | 100% |
| 4 | 1 | 3 | 100% |
| 5 | 1 | 2 | 100% |
| 6 | 1 | 8 | 100% |
| 7 | 1 | 0 | 100% |
| 8 | 1 | 0 | 100% |
| 9 | 1 | 8 | 100% |

The Kmeans algorithm worked better for this problem.

e. kmeans k=6:

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster Number | size | Most common label | Percentage of points in cluster with that label |
| 0 | 192 | 1 | 61.9% |
| 1 | 87 | 0 | 93.1% |
| 2 | 170 | 6 | 51.7% |
| 3 | 161 | 3 | 42.8% |
| 4 | 243 | 4 | 32.0% |
| 5 | 147 | 8 | 36.0% |

**Chart, line chart

Description automatically generatedQ2:**  
a.

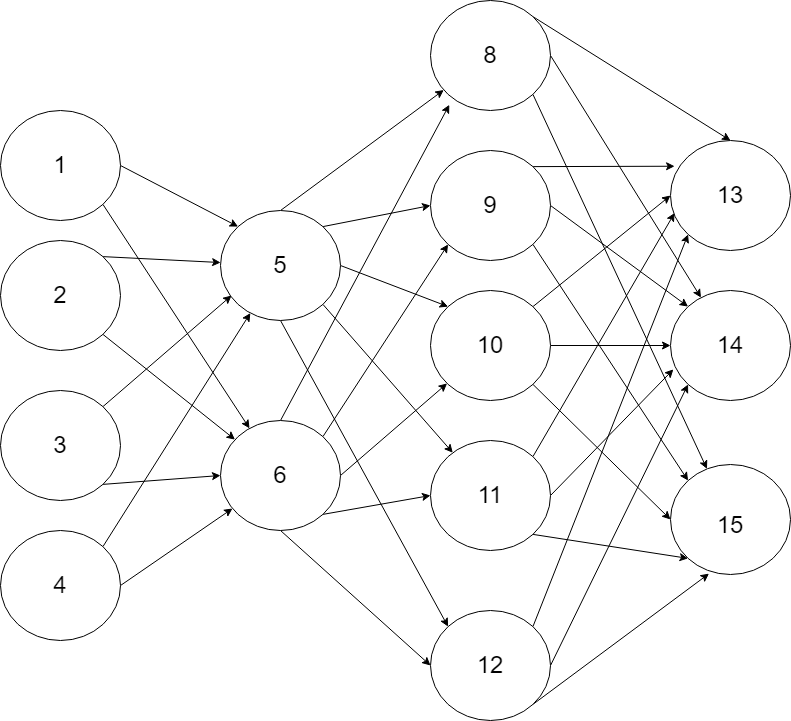
b. Based on what we learned in class, we expect that the value of will decrease as a function of the sample size, because for large the algorithm will try to minimize and ignore the loss over the sample.

1. When we took the average lambda for a sample size based on 10 experiments, we did get the trend we expected. However, for a single experiment we got unexpected increases for some of the sample size. We think that this happened because the random sample was not representative enough for the distribution.
2. Squared loss:

Absolute loss:  
 since   
 =

**Q3:**



**Q4:**

2. The input domain is

,

**Q5:**

1. For all there are coordinates. For each coordinate we can ask questions of the form for at each node of the tree. We can also decide the label at each node. Therefore, there are possibilities at node of the tree. Since the depth of the tree is at most , there are at most nodes in the tree.  
    Therefore, .
2. Danny is wrong. Since ID3 is not ERM algorithm, we cannot guarantee that PAC analysis will hold for .

**Q6:**

,   
  
The Naïve-Base assumption does not hold for the distribution.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 1 | 1 |  |  | 1 |
| -1 | 1 |  |  | 1 |
| 1 | -1 |  |  | -1 |
| -1 | -1 |  |  | 1 |

**Q7:**  
  
a. Let be a matrix that holds the data from the experiment. Since   
 and , the 3rd and 4th columns of are dependent on the 1st and 2nd columns. Therefore, the rank of is at most 2, which means that has at least two eigen values that equals to 0. The distortion is the sum of the smallest eigenvalues, therefore, the distortion is 0.

1. Since the distortion in section a is 0, we need to generate an experiment such that the result matrix has at least 3 eigen values larger than 0 and satisfy the constraints equations. we chose and used *Python* to generate a random experiment that satisfy the equations. for we got:

The eigen values of are: . For this experiment, the smallest eigen values are larger than 0, therefore the distortion is larger than the distortion of section a.

**Q8:**

1. let .

Therefore,

1. (i) let :  
    where

(ii)

let