

CS 4210 – Assignment #5

**Maximum Points: 100 pts.**



Bronco ID: 013 789 022

Last Name: Nash

First Name: Mason

**Note 1:** Your submission header must have the format as shown in the above-enclosed rounded rectangle.

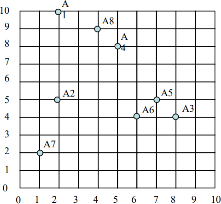
**Note 2:** Homework is to be done individually. You may discuss the homework problems with your fellow students, but you are NOT allowed to copy – either in part or in whole – anyone else’s answers.

**Note 3:** Your deliverable should be a .pdf file submitted through Gradescope until the deadline. Do not forget to assign a page to each of your answers when making a submission. In addition, source code (.py files) should be added to an online repository (e.g., github) to be downloaded and executed later.

**Note 4:** All submitted materials must be legible. Figures/diagrams must have good quality.

**Note 5:** Please use and check the Canvas discussion for further instructions, questions, answers, and hints. The bold words/sentences provide information for a complete or accurate answer.

1. [25 points] By considering the following 8 2D data points below do:
   1. [20 points] Group the points into 3 clusters by using k-means algorithm with Euclidean distance. Show the intermediate clusters (**by drawing ellipses on this 2D space**) and centroids (**by drawing marks like X on this 2D**) in each iteration until convergence. Also, **fill the solution table** below to show the distances from data points to the current centroids and the new centroids found. Consider the initial centroids as: C1 = A1, C2 = A4, and C3 = A7.





|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1st Iteration | | | | | | | | |
| Centroid: (C1 = A1, C2 = A4, C3 = A7) | | | | | | | | |
| Instance | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
| C1 (A1) dist. | 0 | 5 | 8.48528137 | 3.60555128 | 7.07106781 | 7.21110255 | 8.06225775 | 2.23606798 |
| C2 (A4) dist. | 3.60555128 | 4.242640691st | 5 | 0 | 3.60555128 | 4.12310563 | 7.21110255 | 1.41421356 |
| C3 (A7)dist. | 8.06225775 | 3.16227766 | 7.28010989 | 7.21110255 | 6.70820393 | 5.38516481 | 0 | 7.61577311 |
| Cluster Assigned | C1 | C3 | C2 | C2 | C2 | C2 | C3 | C2 |

A picture containing text, screenshot, diagram, number

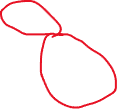
Description automatically generated



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2nd iteration | | | | | | | | |
| Centroid: (C1 = (2, 10), C2 = (6, 6), C3 = (1.5,3.5)) | | | | | | | | |
| Instance | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
| C1 (2, 10) dist. | 0 | 5 | 8.48528137 | 3.60555128 | 7.07106781 | 7.21110255 | 8.06225775 | 2.23606798 |
| C2 (6, 6) dist. | 5.65685425 | 4.12310563 | 2.82842712 | 2.23606798 | 1.41421356 | 2 | 6.40312424 | 3.60555128 |
| C3 (1.5, 3.5)dist. | 6.51920241 | 1.58113883 | 6.51920241 | 5.70087713 | 5.70087713 | 4.52769257 | 1.58113883 | 6.04152299 |
| Cluster Assigned | C1 | C3 | C2 | C2 | C2 | C2 | C3 | C1 |

A picture containing text, screenshot, diagram, number

Description automatically generated



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3rd iteration | | | | | | | | |
| Centroid: (C1 = (3, 9.5), C2 = (5.2, 5.25), C3 = (1.5,3.5)) | | | | | | | | |
| Instance | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
| C1 (2, 10) dist. | 1.11803399 | 4.60977223 | 7.43303437 | 2.5 | 6.02079729 | 6.26498204 | 7.76208735 | 1.11803399 |
| C2 (6.5, 5.25) dist. | 5.72734668 | 3.20975077 | 3.06634962 | 2.75726314 | 1.81727818 | 1.48408221 | 5.3106026 | 3.93732142 |
| C3 (1.5, 3.5)dist. | 6.51920241 | 1.58113883 | 6.51920241 | 5.70087713 | 5.70087713 | 4.52769257 | 1.58113883 | 6.04152299 |
| Cluster Assigned | C1 | C3 | C2 | C1 | C2 | C2 | C3 | C1 |

A picture containing text, screenshot, diagram, number

Description automatically generated



* 1. [5 points] Calculate the SSE (Sum of Square Errors) of the final clustering.

SSE = 1.1180^2 + 1.5811^2 + 3.0663^2 + 2.5^2 + 1.8172^2+ 1.4840^2 + 1.5811^2 + 1.1180^2

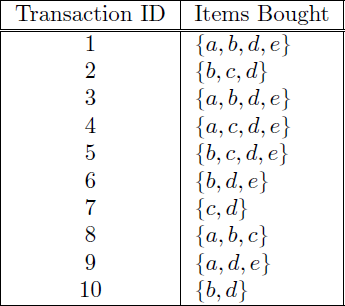
SSE = 28.656

1. [15 points] Complete the Python program (clustering.py) that will read the file training\_data.csv to cluster the data. Your goal is to run k-means multiple times and check which k value maximizes the Silhouette coefficient. You also need to plot the values of k and their corresponding Silhouette coefficients so that we can visualize and confirm the best k value found. Next, you will calculate and print the Homogeneity score (the formula of this evaluation metric is provided in the template) of this best k clustering task by using the testing\_data.csv, which is a file that includes ground truth data (classes).
2. [20 points] The dataset below presents the user ratings on a 1-3 scale for 6 different rock bands.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Bon Jovi | Metallica | Scorpions | AC/DC | Kiss | Guns n’ Roses |
| Fred | 1 | 3 | - | 3 | 1 | 3 |
| Lillian | 3 | - | 2 | 2 | 3 | 1 |
| Cathy | 2 | 2 | 2 | 3 | - | 2 |
| John | 3 | 2 | 2 | 2 | ? | ? |

* 1. [10 points] Apply **user-based** collaborative filtering on the dataset to decide about recommending the bands Kiss and Guns n’ Roses to John. You should make a recommendation when the predicted rating is greater than or equal to 2.0. Use cosine similarity, a neutral value (1.5) for missing values, and the top 2 similar neighbors to build your model.
  2. [10 points] Now, apply **item-based** collaborative filtering to make the same decision. Use the same parameters defined before to build your model.

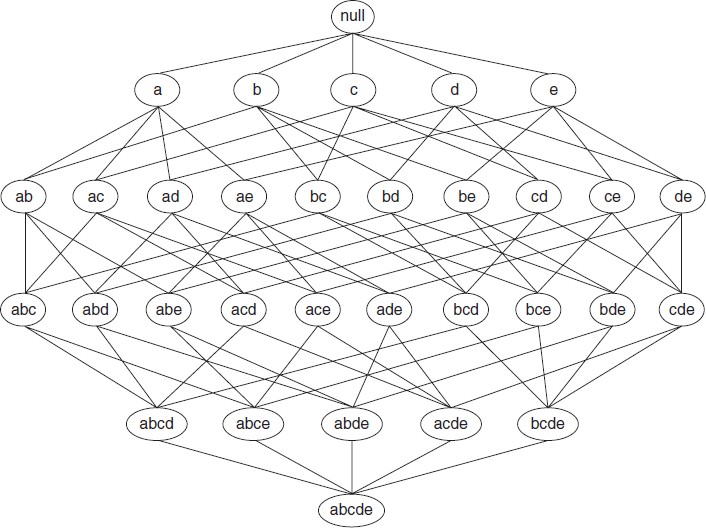
1. [25 points] Consider the following transaction dataset.



Suppose that minimum support is set to 30% (*minsup*) and minimum confidence is set to 60%.

* 1. [5 points] Rank all frequent itemsets according to their support (list their support values).
  2. [5 points] For all frequent 3-itemsets, rank all association rules - according to their confidence values - which satisfy the requirements on minimum support and minimum confidence (list their confidence values).
  3. [5 points] Show how the 3-itemsets candidates can be generated by the 𝐹𝑘−1 X 𝐹𝑘−1 method and if these candidates will be pruned or not.
  4. [10 points] Consider the lattice structure given below. Label each node with the following letter(s):

*F* if it is frequent and *I* if it is infrequent.



1. [15 points] Complete the Python program (association\_rule\_mining.py) that will read the file retail\_dataset.csv to find strong rules related to supermarket products. You will need to install a python library this time. Just use your terminal to type: pip install mlxtend. Your goal is to output the rules that satisfy *minsup* = 0.2 and *minconf* = 0.6, as well as the priors and probability gains of the rule consequents when conditioned to the antecedents. The formulas for this math are given in the template.

**Important Note:** Answers to all questions should be written clearly, concisely, and unmistakably delineated. You may resubmit multiple times until the deadline (the last submission will be considered).

**NO LATE ASSIGNMENTS WILL BE ACCEPTED. ALWAYS SUBMIT WHATEVER YOU HAVE COMPLETED FOR PARTIAL CREDIT BEFORE THE DEADLINE!**