1. **Classification (8 p total).**

After a data mining course the results of the exam was recorded along with some data about the students. The results can be found in in the table below. (GPA is the Grade Point Average.)

ID	Phone number	Language	Passed all assignments	GPA	Passed exam
1	555 - 3452	Java	No	3.1	Yes
2	555 - 6294	Java	No	2.0	No
3	555 - 9385	C++	Yes	3.5	Yes
4	555 - 9387	Python	Yes	2.5	Yes
5	555 - 9284	Java	Yes	3.9	No
6	555 - 0293	C++	No	2.9	No
7	555 - 9237	Java	No	1.9	No
8	555 - 3737	Python	Yes	3.2	Yes

(a) **(6 p)** In no more than one page of text, describe the design of a K-Nearest Neighbor classifier to predict if a student will fail or pass the exam.

Solution

The ID and Phone number are unrelated to a students capacity to pass the exam so they are discarded. The language categoray has three different values C++, Java and Python. Since the language values are nominal rather than ordinal we will consider the distance between two languages to be 1 if they are different and 0 if they are the same.

Passed all assignments is a binary category this means that we can easily use the same idea as for language i.e. if they are different the distance is 1 and if they are the same the distance is 0. The GPA is a quantitative category that ranges from 1.9 to 3.9. For the distance in this dimension we can just take the absolute value of the diffrence in gpa. To make each category have about the same weight we can multiply both the language and assignments distances with 2.

We end up with the following distance measure:

$$dist(X,Y) = 2 \times diff_lang(X,Y) + 2 \times diff_assign(X,Y) + |gpa(X) - gpa(Y)|$$

For the value of K both 1 and 3 are reasonable a larger value for K would be too high since there are so few datapoints. We will select 3 to make the classifier less sensetive.

(b) **(2 p)** Use your K-NN classifier to predict whether the following student (who overslept and missed the original exam) will pass the re-exam.

ID	Phone number	Language	Passed all assignments	GPA	Passed exam
9	555 - 6295	C++	Yes	3.0	?

Solution

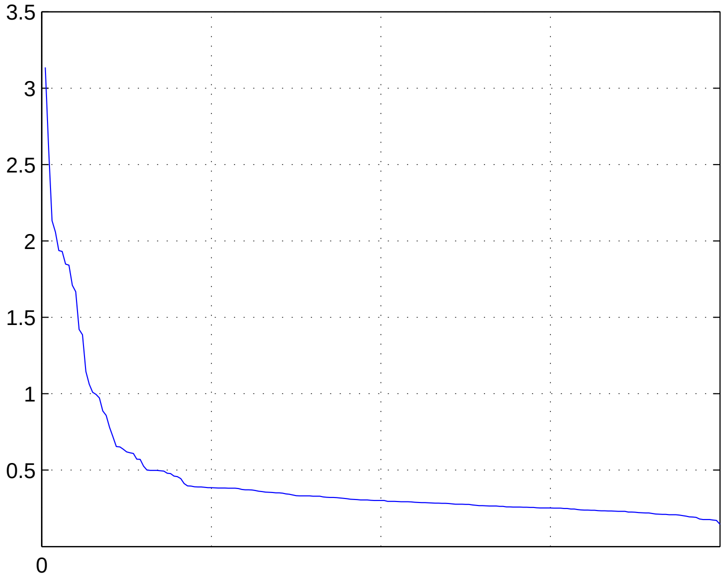
We calculate all the distances using our distance formula:

dist(1,9) = 2 + 2 + 0.1 = 4.1
dist(2,9) = 2 + 2 + 1.0 = 5.0
dist(3,9) = 0 + 0 + 0.5 = 0.5
dist(4,9) = 2 + 0 + 0.5 = 2.5
dist(5,9) = 2 + 0 + 0.9 = 2.9
dist(6,9) = 0 + 2 + 0.1 = 2.1
dist(7,9) = 2 + 2 + 1.1 = 5.1
dist(8,9) = 2 + 0 + 0.2 = 2.2

Students 3, 6 and 8 are the three closest neighbours this gives us two votes for yes and one for no. So our prediction is that the student will pass the assignment.

2. **Clustering (8 p total).**

   (a) The pictures shows a 5-dist graph for a data set.



When we run dbscan on the data with $MinPts = 5$ and $\epsilon = 0.5$

i. **(1 p)** Which is the largest region in the 5-dist graph that contains only core points?
Solution If the distance to the fifth neighbour is less than epsilon a point is a core point so all points to the right of where the graph crosses 0.5 on the Y-axis are core points

ii. **(2 p)** Which is the largest region in the 5-dist graph that contains only noise points?
Solution If the distance to the fifth neighbour is more than two times epsilon than none of the points within epsilon can be a core point which means that

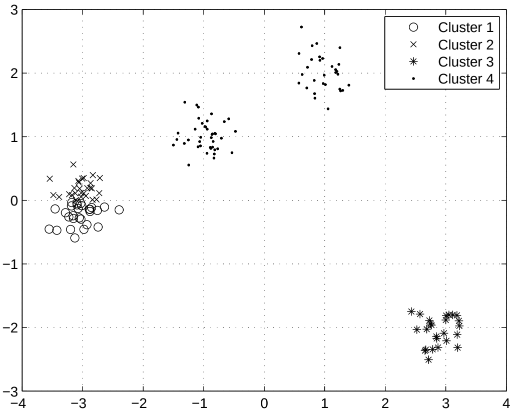
the point must be a noise point. So all points to the left of where the graph crosses 1.0 on the Y-axis are noise points

iii. **(1 p)** Does the regions above cover all points? If not, what points can appear in the remaining region?
Solution The regions do not cover all points. In the remaining area there can be both noise and border points.

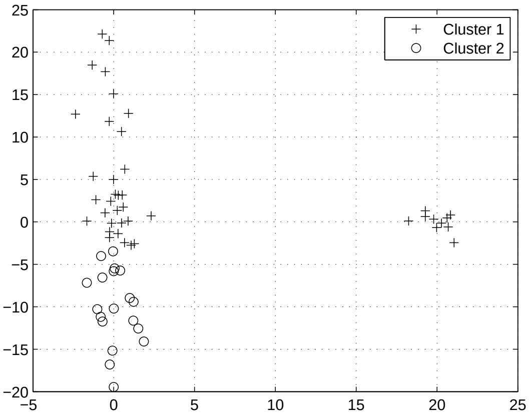
(b) In the figures below two bad clusterings based on K-means is shown. What is the main reason for the bad results, and what can be done to address the problems.

i. **(2 p)** In figure (a)
Solution The problem is caused by bad initial values for the clusters. The easiest solution is to make several runs with different initial values and choose the clustering with the smallest mean distance to the centroids.

ii. **(2 p)** In figure (b)
Solution The problem is caused by the difference in size and density of the two clusters. The problem can be solved by using bisecting k/means and than merging clusters in postprocessing.



(a)



(b)

3. **Association Rules (8 p total).**

(a) **(2 p)** After mining a transaction database for large itemsets (i.e., itemsets with enough support), there is only one large itemset of size 8. Let N be the total number of large itemsets (including the one of size 8). What is the minimal value of N?

Solution There must be at least $2^8 - 1 = 255$ large itemsets, since at least all of the subsets of the 8-itemset must be large.

(b) The $F_{k-1} \times F_{k-1}$ method is a method for generating apriori candidates.

i. **(2 p)** Describe the method.
Solution To generate candiates of size k+1. Take each pair of large itemsets of size k compare the first k-1 items and if they are all the same generate a new candidate consisting of these k-1 items plus the last item from each itemset

ii. **(1 p)** With a short motivation, does this method eliminate the need for candidate pruning?
Solution No if a,b and a,c are large itemsets the candidate a,b,c will be generated even if b,c is not large.

(c) **(3 p)** For each of the following, say if the statement is always true, always false or sometimes true and sometimes false. S is the support and C is the confidence. No motivation is needed. **(0.5 p for correct answer, -0.5 points for wrong answer and 0 points for no answer. You can never get negative result on the whole question.)**

i. $S(a \rightarrow b) \geq S(a \rightarrow b, c)$

ii. $C(a \rightarrow b) \geq C(a \rightarrow b, c)$

iii. $S(a \rightarrow b) = S(b \rightarrow a)$

iv. $C(a \rightarrow b) = C(b \rightarrow a)$

v. $C(a \rightarrow b, c) > C(a, b \rightarrow c)$

vi. $Min(C(a \rightarrow b), C(b \rightarrow c)) > C(a \rightarrow c)$

Solution

i. true

ii. true

iii. true

iv. sometimes true, sometimes false

v. false

vi. sometimes true, sometimes false

4. **Direct hashing and pruning (8 p total).**

For the transaction database below, perform the following steps of the apriori algorithm with direct hashing and transaction pruning.

1. Count the itemsets of size 1.

2. Prune unsupported itemsets.

3. Perform transaction pruning.

4. Perform direct hashing for itemsets of size 2.

Minimum support is 0.5. Use the hash function $\left(\sum_i x_i i^2\right) mod\, 4$, where x_i is one element of an itemset and $x_{i+1} > x_i$.

Which candidate itemsets can be pruned based on the hash table? Meassured in the total count in the hash table, how much work do you save by using the pruned transaction table, compared to the non-pruned version?

Transactions
1 3 6
1 2 3 6
1 4 5
2 3 6
1 3 5
2 3 4 5 6
1 2
1 2 4 6

Solution

1.

Itemset	Count
{1}	6
{2}	5
{3}	5
{4}	3
{5}	3
{6}	5

2. Itemsets {4} and {5} do not have enough support so they can be removed

3.

Pruned Transactions
1 3 6
1 2 3 6
2 3 6
1 3
2 3 6
1 2
1 2 6

4.

Hash Value	Itemsets	Count
0		0
1	{1,2}, {1,3}, {1,6}	9
2	{2,3}, {2,6}	7
3	{3,6}	4

Non of the itemsets can be pruned based on the hashing. We only need to calculate 20 hash values since we use the pruned transaction table. If we had used the original one we would have had to calculate 35 hash values.