CIS 5930 - Assignment 4

Due Date: Wed, Feb 25th, 4pm

The following assignment has two parts, a written and a programming. PLEASE CHOOSE JUST ONE OF THEM.

Submission instructions:

- Written: At the instructor's office hours.
- Programming: Email to instructor and TA by 4pm.

1 WRITTEN

Problem 1 [10 points - 2 points each]

A. What is incremental clustering?

Incremental clustering is a process where data is clustered by processing one point at a time. Typically there are space constraints that prevent the algorithm from storing all the data in main memory.

B. Why is incremental clustering popular?

There is more and more data available for data analysis. Incremental algorithms are often suitable for big data analysis.

C. What does it mean that a data set is clusterable?

If a data set is clusterable, then there exists an underlying cluster structure and the the data can be meaningfully clustered.

D. Are there cases where a poorly clusterable data set should be clustered? Justify your answer.

Yes. Even if the data is poorly clusterable, clustering results may be useful, an example being identifying target markets.

E. What is the purpose of notions of clusterability?

Notions of clusterability are often used for theoretical analysis.

Problem 2 [10 points - 2 points each]

Discuss which algorithm/s or types of algorithms would you try in the following cases. Justify your answers.

A. There are 100 million data points, and at least 1000 new ones arrive every day. A clustering of all data must be available at all times.

   Incremental clustering, since new data arrives every day and eventually there will not be enough space to store it all in main memory.

B. The data consists of less than 500 points and cluster separation is most critical.

   Single, average, or complete linkage. These algorithms are effective at identifying well-separated clusters (when they are sufficiently separated).

C. The data includes many outliers, and you would like the output of the algorithm to be robust to them.

   Randomized Lloyd's method, as it exhibits some robustness to outliers.

D. The data consists of many duplicated elements, and you would like the algorithm to be robust to them.

   Single or complete linkage, since they are weight robust.

E. The clusters should have similar sizes (number of points).

   Randomized Lloyd's method, as it tends to identify relatively balanced clusters.

Problem 3 [15 points - 10+5]

Consider the setting where data arrives uniformly at random.

A. Prove that, in this setting, sequential k-means is weight sensitive.

   Fix some weights on the data. Let C be the output of sequential k-means on that data when it arrives uniformly at random. We show that a different clustering will be output if we change the weights on the data. Pick any pair of points that belong to the same cluster in C. Make their weights sufficiently large so that with arbitrarily high probability these two points will be appear within the first k elements in a randomized ordering of the data. As a result, these points become centers and belong to different clusters.

B. Given an example of an incremental clustering algorithm that is not weight-sensitive in this setting. Justify your answer.

   Sequential nearest-neighbor clustering is weight robust, as the algorithm relies entirely on minimal distances.

Sequential nearest-neighbor clustering:
Set T to the first k data points
Repeat:
Get the next point x and add it to T
Let t, t_0 be the two closest points in T
Replace t, t_0 by either of these two points

Problem 4 [25 points - 10+5+10]

An algorithm is *k-perfect-detecting* if it outputs a perfect clustering whenever it is present in the data. Is sequential k-means k-perfect-detecting in the following settings? Prove your answers.

A. Data is ordered by an adversary (worst case)

Sequential k-means is not perfect detecting in this case.

Consider a perfect k-clustering C, where one of the clusters C_i consists of at least k points all of which have weight W, and the remaining points have weight 1. Note that a data set has at most one perfect k-clustering for any values of k, so C is the unique perfect k-clustering. Then, assume that the algorithm sees all points in C_i before all other data. Then, if W is sufficiently large, the remaining points don't shift the centers by much, and as such the final solution subdivided data in C_i, so it does not equal C, and so it is not perfect.

B. Data arrives uniformly at random (average case)

Sequential k-means is not perfect-detecting in this case.

Consider a perfect k-clustering C, where one of the clusters C_i consists of at least k points all of which have weight W, and the remaining points have weight 1. Then, if W is sufficiently large, then with high probability, the algorithm will see points in C_i before other data, and so it will have multiple centers in C_i. Then, if W is sufficiently large, the centers in C_i will shift by an arbitrarily small amount, and as such the final solution subdivided data in C_i, so it does not equal C, and so it is not perfect.

C. Data is ordered by a helpful teacher (best case)

Sequential k-means is not perfect detecting in this case. See Theorem 4.4 in "Incremental Clustering: The Case for Extra Clusters" by Ackerman and Dasgupta.

Problem 5 [30 points - 15 points each]

A. Prove that no weight-sensitive algorithm is k-perfect-detecting.

On all data sets, the output of weight sensitive algorithms can be altered by changing the weights on the data. Yet, a k-perfect detecting algorithm always outputs the same result on data that has a perfect k-clustering, so it cannot be weight sensitive.

B. Define a clusterability condition so that no weight-robust algorithm can identify such cluster structure on all data sets that exhibit it. Justify your answer.

Notion of clusterability: The sum of weights of clusters don't different by more than ϵ. Weight robust algorithms cannot detect such clusterings since they must output the same result regardless of the setting of the weights.

Problem 6 [10 points (5+5)+ 10 Bonus]

A. Give an example of a cluster structure that can be detected both offline and in the incremental setting. Justify your answer.

A perfect clustering can be detected in both offline and incremental setting.

B. Give an example of a cluster structure that can be detected offline but provably cannot be detected in the incremental setting. Justify your answer.

Nice Clustering can be detected offline but provably cannot be detected in the incremental setting. Since we will not store the data. However there exists a dataset that require all data set to detect a nice clustering.

C. (Bonus) Come up with an intuitive notion of clusterability that is detected by sequential k-means, other than the example given in class. You may assume that the data arrives uniformly at random. Prove your answer.

2 PROGRAMMING

In this programming assignment you'll be applying clustering on real data.

UCI data sets

Go to UCI Data Sets `https://archive.ics.uci.edu/ml/datasets.html`, and select *five* clustering sets. The "default task" column should include "clustering", and the "attribute types" column should be numeric (integer or real). The number of instances should be at least 500. You may use the link on the left to navigate.

You will be clustering each data set you choose using the following algorithms:

- Average linkage
- Single linkage
- Lloyd's method with random initialization
- Lloyd's method with furthest centroids initalization
- Incremental k-means

NOTE: UCI data sets are often high dimensional. Your algorithms should work for high dimensional data (in the previous programming assignment dimension was set to 2).

BONUS (5 points): k-means++

As a bonus, you may also implement k-means++. This algorithm is a different way to initialize Lloyd's method. The first center is chosen uniformly at random. Let $D(x)$ denote the distance from point x to the closest existing center. Choose each subsequent center by selecting a point with probability proportional to $D(x)^2$.

Comparing clusterings

You will be asked to compare the output of the different algorithms to the provided "target clustering". Each UCI data set comes with such a target clustering. It's suppose to be the correct way to cluster that data. Of course, as we learned in class, this is not necessarily the only reasonable way to cluster this data.

Compare the clusterings using Hamming Distance or Misclassification Error by implementing these methods (see notes on "Clustering distance" on the course website). **BONUS:** For an extra 5 points, compare using both.

Report

You will be producing a report, which will compare all the algorithms you implement on the 5 data sets that you select.

For each data set, report the following.

- How long each algorithm took, measuring in minutes, seconds, etc.
- Compare the output of each clustering algorithm to the target clustering.
- For each data set, highlight which algorithm (1) was the fastest, and (2) found the most accurate clustering.

Then summarize what you have learned from this assignment. Did any algorithm stand out in terms of speed or accuracy? Was there a lot of variety regarding which performed best? If you chose to implement both Hamming Distance and Misclassification Error, were the results different using these two distances? Discuss anything else that you learned.

Other details

- You may use any programming language of your choice.
- All clustering algorithms must be implemented from scratch, the only exception is relying on your own code from the previous programming assignment in this course.
- **You may be asked to demo your project to the TA and/or instructor.** Please be prepared to demonstrate your project by showing us its output on the data you chose as well as new data.
- Make sure that it is easy to run your algorithm on new data, and that it is easy to find the distance between two clusterings (using Hamming Distance or Misclassification Error).

Submission

Please email your assignment to the TA and instructor before the deadline.

Your submission should include all of your source code, an executable file, and your report. We may also ask you to do a demo of your project to the TA or instructor.

Evaluation

- *Code (15%):* We will look at your code to ensure that you wrote all algorithms from scratch (other than reusing your own code from the previous assignment in this course), and that the code is well-designed. Please ensure to use objective-oriented programming and comment your code well for ease of navigation.
- *Correctness (50%)*: Whether your algorithms and clustering distance (Hamming or Misclassification Error) are implemented correctly. We may run your algorithms on additional data.
- *Report (35%)*: Quality of your report and discussion section.