School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP5349

Assignment name: Assignment 2

Tutorial time: Thursday **Tutor name:** 4:00 PM - 6:00 PM

DECLARATION
We the undersigned declare that we have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, an, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Academic Dishonesty and Plagiarism in Coursework Policy* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1. Shaowei Zhang	470144491	Yes / No	Yes/No	Zhang Shaowei
2. Binbin Song	450621769	Yes / No	Yes / No	Binbin Song
3.		Yes / No	Yes / No	
4.		Yes / No	Yes / No	
5.		Yes / No	Yes / No	
6.		Yes / No	Yes / No	
7.		Yes / No	Yes / No	
8.		Yes / No	Yes / No	
9.		Yes / No	Yes / No	
10.		Yes / No	Yes / No	

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

Cloud Computing – Assignment 2

The report will include three parts:

1. Describe different transformation functions from task1 to task3
2. Display Execution times of our tasks on small and large dataset (Appendix)
3. Final output files from various executions (Appendix)

Task1: Number of (valid) measurements conducted per researcher

Below flowchart (Fig. 1) explains how we design the framework of task1.

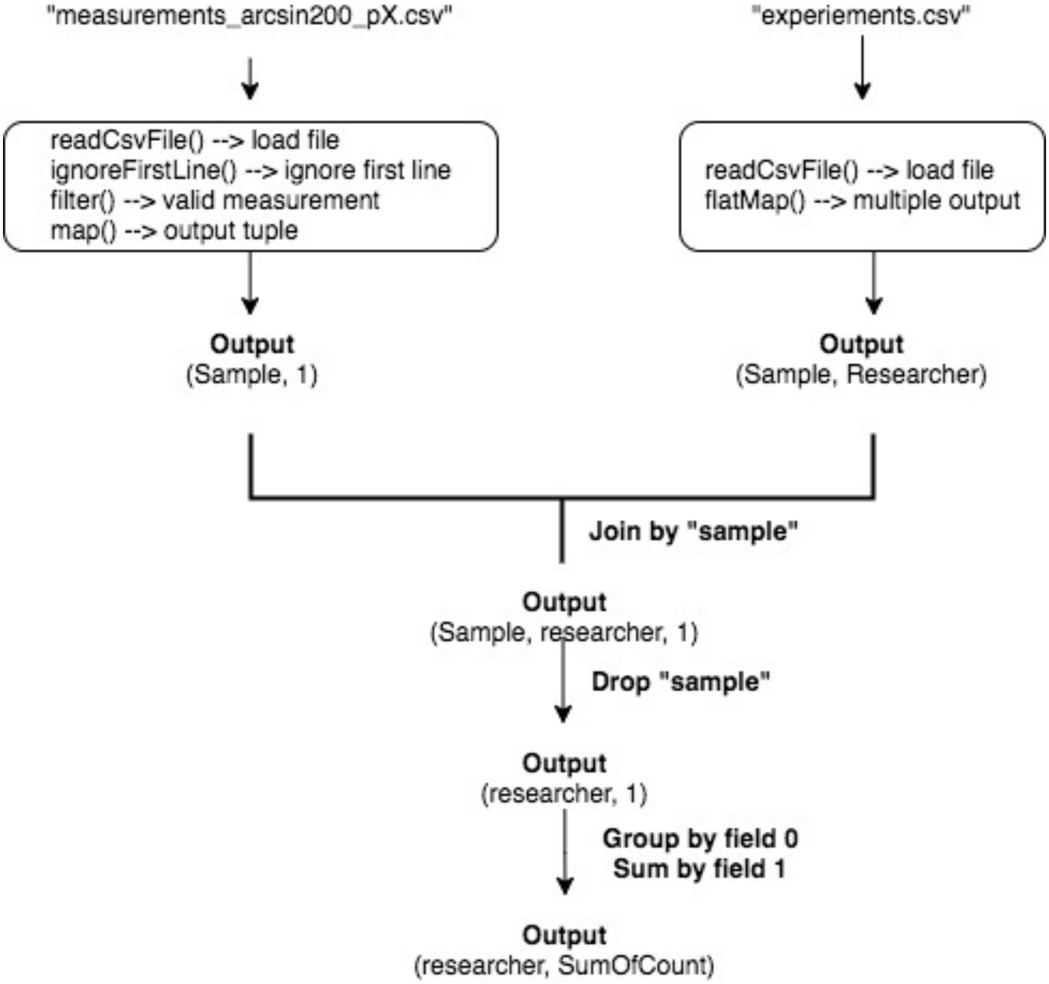

Figure 1 Task1 Job Design

Task2: k-means clustering of the measurements

There are totally 4 steps, the whole framework of task2 is described in the flow chart (Fig. 2) . For k-means, the default k is 4. Besides built-in functions, there are totally 5 classes and UDF used to implement this job (Table 1).

Class/ UDF/ Operator	Step	Functions
getPointDataSet (params, env)	Step 1	Load initial data points
getCentroidDataSet(params,env)	Step 1	Initialize centroids points
SelectNearestCenter()	Step 2	Calculate closet centroid for each point
CountAppender() & CentroidAverager()	Step 2	Update centroid
finalCentroidsplit()	Step4	Spit centroids into required format

Tabel 1

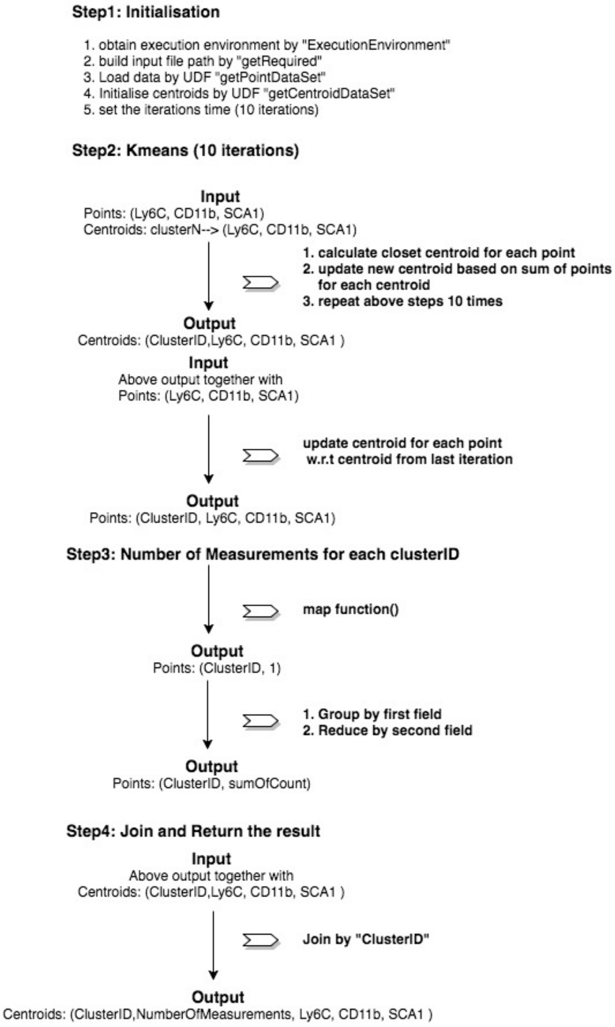
Step1: Initialisation
1. obtain execution environment by "ExecutionEnvironment"
2. build input file path by "getRequired"
3. Load data by UDF "getPointDataSet"
4. Initialise centroids by UDF "getCentroidDataSet"
5. set the iterations time (10 iterations)
Step2: Kmeans (10 iterations)
Input
Points: (Ly6C, CD11b, SCA1)
Centroids: clusterN--> (Ly6C, CD11b, SCA1)
1. calculate closet centroid for each point
2. update new centroid based on sum of points for each centroid
3. repeat above steps 10 times
Output
Centroids: (ClusterID,Ly6C, CD11b, SCA1)
Input
Above output together with
Points: (Ly6C, CD11b, SCA1)
update centroid for each point w.r.t centroid from last iteration
Output
Points: (ClusterID, Ly6C, CD11b, SCA1)
Step3: Number of Measurements for each clusterID
map function()
Output
Points: (ClusterID, 1)
1. Group by first field
2. Reduce by second field
Output
Points: (ClusterID, sumOfCount)
Step4: Join and Return the result
Input
Above output together with
Centroids: (ClusterID,Ly6C, CD11b, SCA1)
Join by "ClusterID"
Output
Centroids: (ClusterID,NumberOfMeasurements, Ly6C, CD11b, SCA1)

Figure 2 Task2 flowchart

Task3: Outlier removal and reclustering

The code and framework of Task3 is similar with task2. The step 1, step 2, and step3 are almost same except that the output of step 2 is in format (clusterID, distance, Ly6c, CD11b, SCA1). Therefore, we do not show previous three steps, and start explaining our job from step 4. The framework from step4 to step 6 is shown in figure3:

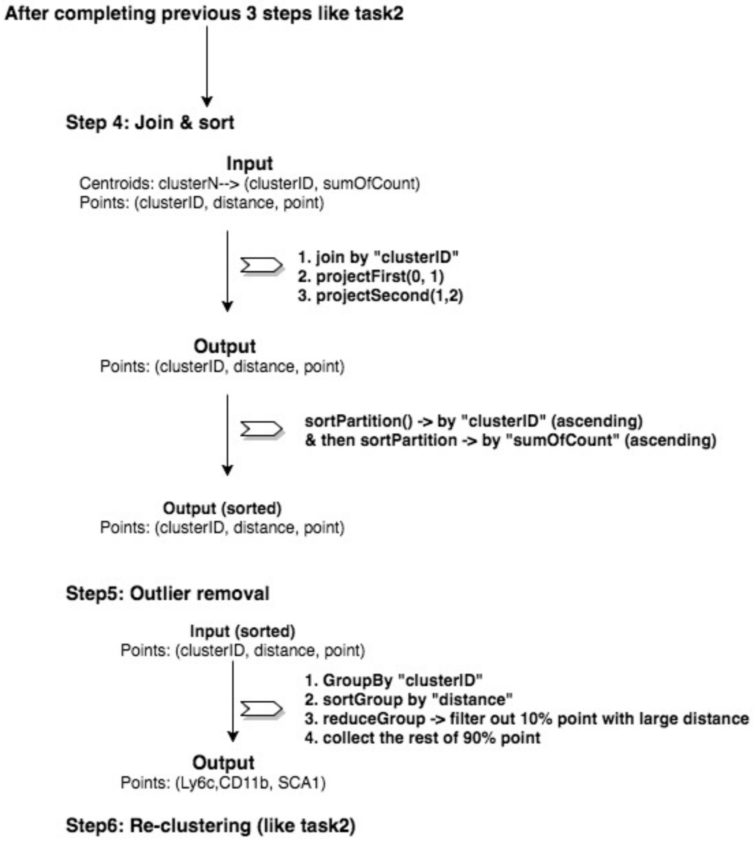
After completing previous 3 steps like task2
Step 4: Join & sort
Input
Centroids: clusterN--> (clusterID, sumOfCount)
Points: (clusterID, distance, point)
1. join by "clusterID"
2. projectFirst(0, 1)
3. projectSecond(1,2)
Output
Points: (clusterID, distance, point)
sortPartition() -> by "clusterID" (ascending)
& then sortPartition -> by "sumOfCount" (ascending)
Output (sorted)
Points: (clusterID, distance, point)
Step5: Outlier removal
Input (sorted)
Points: (clusterID, distance, point)
1. GroupBy "clusterID"
2. sortGroup by "distance"
3. reduceGroup -> filter out 10% point with large distance
4. collect the rest of 90% point
Output
Points: (Ly6c,CD11b, SCA1)
Step6: Re-clustering (like task2)

Fig3. Task3 flowchart

Figure 4 is task2 clustering result, and figure 5 is task 3 clustering result. As we can see, after removing outlier, the clustering result is improved a lot.

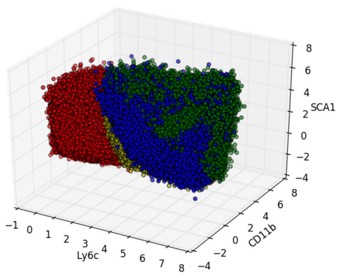

Figure 4 (Task2 – with outlier)

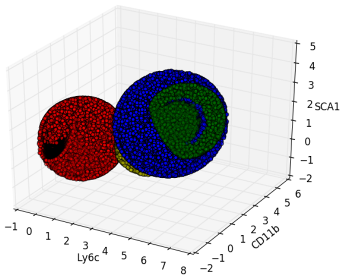

Figure 5 (Task3 – without outlier)

Appendix:

Final output path

Task1 output:

hdfs:///user/szha5691/assignment2TrueFinal

Task2 output:

hdfs:///user/szha5691/assignment2TrueFinal

Task3 output:

hdfs:///user/szha5691/assignment2TrueFinal

Execution times

Task item	Small dataset (ms)	Large dataset (ms)
Task1	2434	6940
Task2	5610	25591
Task3	9099	52988