[1]

Regional Weather Temperature Classification by Employing Machine Learning Algorithms: Clustering

Ashish Ghai, Ramyakrishna Variagyam, Akash Malla, Samarth Mehta, Lakshmi Shankaro, Sabrina Rohatgi, Priyam Bajaj
Santa Clara University Santa Clara, California, USA

The purpose of this paper is to explore machine learning algorithms by predicting temperature by using k-means algorithm across three different platforms; Hadoop/Mahout, Weka and Spark. We will explore the challenges faced and the methods used to overcome them. By exploring these machine learning platforms we compare their performance and set to decide which one is the most efficient.

[1]This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support information, including sponsor and financial support acknowledgment. For example, "This work was supported in part by the U.S. Department of Commerce under Grant BS123456".

The next few paragraphs should contain the authors' current affiliations, including current address and e-mail. For example, F. A. Author is with the National Institute of Standards and Technology, Boulder, CO 80305 USA (e-mail: author@ boulder.nist.gov).

S. B. Author, Jr., was with Rice University, Houston, TX 77005 USA. He is now with the Department of Physics, Colorado State University, Fort Collins, CO 80523 USA (e-mail: author@lamar.colostate.edu).

T. C. Author is with the Electrical Engineering Department, University of Colorado, Boulder, CO 80309 USA, on leave from the National Research Institute for Metals, Tsukuba, Japan (e-mail: author@nrim.go.jp).

I. INTRODUCTION

Each machine learning environment provides its positives and negatives. Mahout is intended to support scalable machine learning and it is particularly strong in recommendations. Mahout produces free implementations of distributed or otherwise scalable machine learning algorithms focused primarily in the areas of collaborative filtering, clustering and classification. Many of the implementations use the Apache Hadoop platform. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.Weka is intended to support a broad range of algorithms, all implemented in an in-memory fashion. Scalability beyond in-memory sizes is explicitly not a goal. Weka also has a graphical interface that controls essentially all functions. Spark provides a general machine learning library that is designed for simplicity, scalability, and easy integration with other tools. With the scalability, language compatibility, and speed of Spark, data scientists can solve and iterate through their data problems faster. By finding a machine learning algorithm common to all three, mahout, spark and weka we are able to use weather data set and run it on all three.

II. ARCHITECHTURAL OVERVIEW

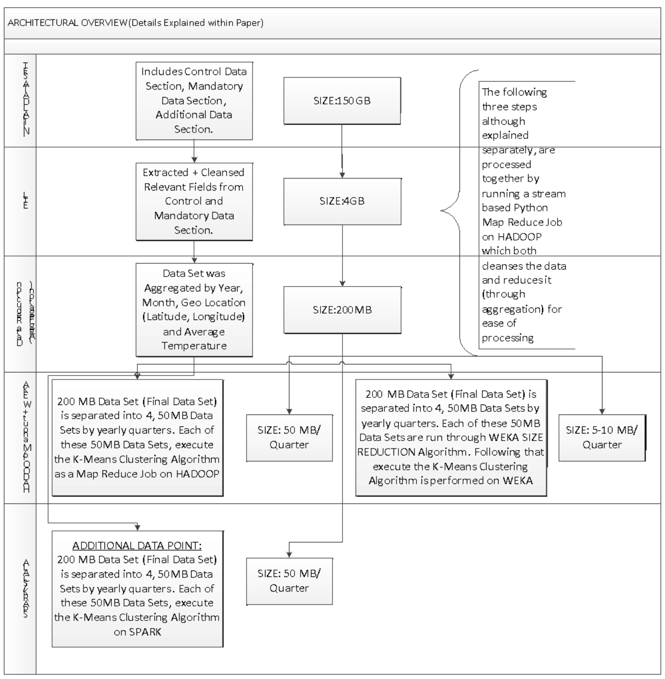

Fig. 1. The above image depicts the Architectural Overview

As depicted from the diagram above, the initial data set comprised of Climatological data from the past 5 Years (2012-2015). The data set was procured from the National Climatic Data Center (NCDC), in conjunction with the Federal Climate Complex (FCC). The Total Size of this initial data set was ~150Gb but due to restrictions faced on the Linux File System with WEKA, this large data set wasn't used as the based for Mahout/Hadoop, WEKA and SPARK Machine Learning Algorithm Evaluations.

The Record Structure of the following data set contained a variable length and comprised of "Control Data Section", "Mandatory Data Section", and "Additional Data Section". Since the 150GB data couldn't be used, the first step explained in the above architecture is to cleanse the data and along with the cleansing of the data, we also extract the relevant fields from the data for the machine learning algorithm.

The relevant data which is extracted includes relevant fields of the "Control Data Section", which includes the Year, Month, Date, Time, Geo Location (Latitude, Longitude). Note that the "Control Data Section" is 60 characters long and is of fixed length. The second section which was considered was the "Mandatory Data Section". This is also of fixed length and contains several attributes pertaining to elements such as wind, visibility and temperature. However, for the purpose of this evaluation, we used two specific parameters which include Air Temperature (defined in degrees Celsius and indicating a scaling factor of 10) and Air Temperature Quality Code. Besides the extraction of the data, we also conduct the other steps of ETL (Extraction, Transformation and Loading) the data. The cleansing of the data within this architecture is performed in two ways. The first indicates looking for missing data. Missing data within the above "Control Data Section" and "Mandatory Data Section" is represented in the original data set by '9'. However, depending on the length of the Field, the missing field could be represented by multiple 9's. The second set of cleansing which is performed on this is data is one in which we check the Air Temperature Quality Code which denotes a quality status of an Air Temperature observation. Hence, within the data which is considered, we only consider observations which pass "All Gross Limit Checks" indicating that the value of Air Temperature is one which can be trusted.

The new cleansed data set is of size 4GB. In the idealistic approach, we would have taken this 4GB Data to run Mahout/Hadoop Map Reduce Algorithm and ran the same 4GB Data Set through WEKA and reduced it. However, the 4GB data set was still not possible to reduce on the WEKA system due to linux file system storage space issues. Having faced this issue yet again, another Python Streaming Based Map Reduce Function was created to help aggregate the same 4GB Data Set. The Data Set was aggregated by Year, Month, Geo Location (Latitude, Longitude), and Average Temperature. It was possible to perform the aggregation on the multiple temperature points calculate during a whole day for each of the Geo Locations provided. The aggregation reduced the data set to 2 Million entries which comprised of the above mentioned fields and let to a total file size of 200MB.

The 200MB Data Set (Final Data Set) was separated into four quarters, each of which included three sequential months of data. The purpose of splitting these files is for Temperature Analysis across different quarter giving a sense of seasonality. Each of these files is run the K-Means Clustering Algorithm as a Map Reduce job on Hadoop. Each for the four 50MB files are further reduced by WEKA into 5-10MB Files and then run through K-Means Clustering Algorithm on WEKA.

Additionally, in order to measure performance, the reduced files from Weka are also run as Map Reduce Job on Hadoop. Each of the four 50MB files contains 500K entries, which were broken into a training data set and test data set to test the robustness of the cluster. However the robustness of each cluster can also be measured by comparing the continental temperature for a quarter with the actual temperature traced back in history.

An additional data point is also collected which contains the execution of the K-Means Clustering Algorithm on Spark. The Spark Algorithm is executed on the 200MB Data Set.

III. Materials and Methods

The following section deep dives into the different sections defined within the Architectural Overview. The deep dive focusses on the steps taken and the methodologies adapted to execute each step.

A. ETL + Data Reduction (Aggregation)

Deep Diving into the Architectural Overview, the first step which is required involves performing ETL + Data Reduction in order to create a Data Set which can be executed both on HADOOP/Mahout and WEKA. In order to do this, a Python Streaming Based Map Reduce Job is executed which begins by taking the 150GB Data and Reducing to a 200 MB File.
The below shown map reduce command sequence is executed on the cluster which helps to reduce the data.

```
hadoop jar /opt/cloudera/parcels/CDH/jars/hadoop-streaming-2.6.0-cdh5.4.3.jar -input WDATA/WDATA_FINAL5
-output WDATA_ATODPT -mapper /home/aghai01/WDATA/MAP/WDATA_MAP_ATOAT.py -reducer
/home/aghai01/WDATA/REDUCE/WDATA_REDUCE_ATOAT.py
```

The above sequence uses the HADOOP streaming jar for reduction. The input file provided is stored onto the HDFS File System and is called WDATA_FINAL5 (This contains the complete 150GB Data Set with Control, Mandatory and Additional Data for the last five years). The Mapper is written in python and is named WDATA_MAP_ATOAT.py and the Reducer which is also created in Python is called WDATA_REDUCE_ATOAT.py
The code below shows a snippet of the Mapper Function. Since we are using a python streaming function, the mapper will extract and cleanse the data and only provide the relevant data to the STDIN. From where, each line can be picked up and reduced accordingly. Within the snippet shown below, we extract specific fields of the Control and Mandatory Data. The two Mandatory Field Data fields (Air Temperature, Air Temperature Quality Code) are checked against missing data vales and also checked against the Air Temperature Quality requirements. Those entries which meet the requirements are then converted into a Key, Value Pair. Key Including YEAR, Month, Latitude, Longitude, and value including the Air Temperature of the Entry.

```
for line in sys.stdin:
	val = line.strip()
	#Control Data Filtering
	(YEARMM, YEAR, MM, DATE, SOURCE, LAT, LONG) =
(val[15:21],val[15:19],val[19:21],val[21:23],val[27:28],val[28:34],val[34:41])
	#Air Temperature
	(ATOAT,ATOQC,ATODPT,ATODPQC) = (val[87:92],val[92:93],val[93:98],val[98:99])

	#(YEARMM,ATOAT,ATOQC) = (val[15:21],val[88:92],val[92:93])
	if(ATOAT != "+9999" and re.match("[01459]",ATOQC) ):
		print '%s\t%s' % (YEARMM + '_' + MM + '_' + LAT + '_' + LONG,ATOAT)
```

The code below shows a snippet of the Reducer Function. The reducer begins by reading the STDIN for the entries which were flushed by the Mapper. It then begins to aggregate (average), based on every unique key by calculating the sum and count based on each key. Dividing the sum by the count provides the average. Note that each of the Key's are associated to a specific reducer.

```
sum_ATOAT = {}
count_ATOAT = {}

for line in sys.stdin:
	(YMLL,ATOAT) = line.strip().split("\t")
	(key,val_ATOAT) = (YMLL,ATOAT)


	# convert count (currently a string) to int
	try:
		val_ATOAT = int(val_ATOAT)
		sum_ATOAT[key] = sum_ATOAT.get(key,0) + val_ATOAT
		count_ATOAT[key] = count_ATOAT.get(key,0) + 1
	except ValueError:
		# count was not a number, so silently
		# ignore/discard this line
		continue

#sort the words lexigraphically;  this step is NOT required, we just do it so that our  final output will look more like
the official Hadoop
sorted_sum_ATOAT = sorted(sum_ATOAT.items(), key=itemgetter(0))
sorted_count_ATOAT = sorted(count_ATOAT.items(), key=itemgetter(0))

for key1, sum in sorted_sum_ATOAT:
	for key2, count in sorted_count_ATOAT:
		if(key1 == key2):
			average = sum/count;
			print '%s\t%s\t%s' % (key1, 'avg_ATOAT',average)
```

B. Execution of K-Means Clustering Algorithm as a Map Reduce Job on HADOOP

As explained in the Architecture Overview, the Final Data Set was segregated into four data sets as per yearly quarters. The code snippet below indicates the preparation and execution of each of these quarterly datasets. But as an example, we only show the commands for Quarter 1. Similar execution can be performed the remaining Quarters. The following Command Sequence was executed on the Hadoop Cluster by using mahout map reduce command sequences.

```
hadoop fs -copyFromLocal wdata-q1/ .
mahout seqdirectory -c UTF-8 -i wdata-q1/ -o wdata-q1-seq
mahout seq2sparse -i wdata-q1-seq -o wdata-q1-vec -ow -chunk 100 -x 90 -seq -ml 50 -n 2 -nv
mahout kmeans -i wdata-q1-vec/tfidf-vectors/ -c wdata-q1-kmeans-centroids -cl -o wdata-q1-kmeans-clusters -k
10 -ow -x 10
mahout clusterdump -d wdata-q1-vec/dictionary.file-0 -dt sequencefile -i
wdata-q1-kmeans-clusters/clusters-1-final -n 20 -b 100 -o cdump_q1.txt -p
wdata-q1-kmeans-clusters/clusteredPoints/
ls -l cdump_q1.txt
mahout clusterdump -d wdata-q1-vec/dictionary.file-0 -dt sequencefile -i
wdata-q1-kmeans-clusters/clusters-2-final -n 20 -b 100 -o cdump_q1.txt -p
wdata-q1-kmeans-clusters/clusteredPoints/
```

Note that the execution of the K-Means Algorithm on Mahout requires the splitting of each of the 50MB Quarter Data Set Files into multiple small files. This aspect of running K-Means on Mahout wasn't clear and hence originally the 50MB file on its own didn't execute. Hence, the files required to be split. The splitting can be simply achieved by doing a split on the data to create smaller files. The above snippet of code indicates the execution of the K-Means Algorithm on fields - Geo Location (Latitude, Longitude), Year, Month and Average Temperature. These are the specific fields which are available within the 50MB file which was segregated and dumped within the "wdata-q1" folder. The number of clusters which are requested within the above analysis is 10 and this is represented within the command sequence "kmeans-clusters –k 10". Once the vectors are created, they can be used to associate each entry to one of the specific clusters.

C. Execution of K-Means Clustering Algorithm on WEKA

As explained within the architecture overview, each of the 50MB files from every quarter are further reduced by the WEKA into 5-10MB files. These 50 MB Files contain 500K Entries, however the 5-10MB Files contain ~200K Entries. In order to further reduce the 50MB Files, the files need to be first converted into 'arff' format which is the file format which WEKA Supports. This can be achieved by taking the 50 MB Files and converting them using a python script. Following, the conversion of the files, the files can be fed into WEAK for reduction.

Need to ADD further Information as to how the files were split on WEKA and if there is any code examples which can be added– Samarth

Unlike Mahout, the 5-10MB doesn't require any further splitting of the files in order to run K-Means Algorithm on WEKA. Fig 2 depicts a GUI is invoked within WEKA to help execute the Simple K-Means Algorithm and similar to Mahout the total of 10 Clusters are requested to be executed on the smaller dataset. Also note that the maximum amount of iterations allowed within the data below is 500 (Essentially this means that every entry within the data needs to be grouped to a cluster within 500 iterations and that entry should have the shortest distance to its associated cluster.

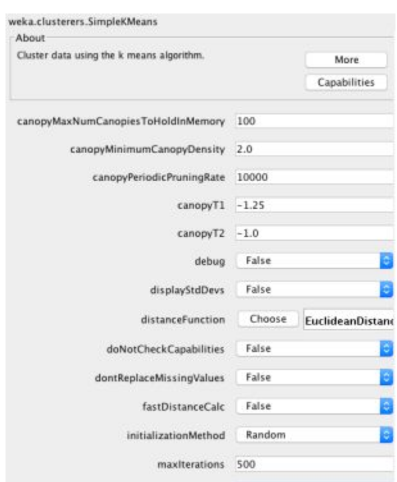

Fig 2. WEKA Simple K-Means Clustering GUI

The command sequence shown below is executed on WEKA:

```
weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 10 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10
```

D. Execution of K-Means Clustering Algorithm on Spark

As depicted within the Architecture Overview, besides the execution on HADOOP/Mahout and WEKA, the Final Data Set (200 MB Files) was also executed on SPARK. The below snippet is written in SCALA and executed the KMeans algorithm.

```
import org.apache.spark.mllib.clustering.{KMeans, KMeansModel}
import org.apache.spark.mllib.linalg.Vectors

//Start timer, it will print time in nano seconds.
val t0 = System.nanoTime()

// load file and remove header
val data = sc.textFile("/Users/akashmalla/Documents/COEN 242/spark dataset/WDATA_Q4.csv")
val rows = data.filter(l => l != data.first)

// define case class
case class CC2(year_month_lat_long: String, avg_temp: Integer, c_year: String, c_month: String, c_lat: String,
c_long: String, year: Integer, month: Integer, lat: Integer, long: Integer)

// comma separator split
val allSplit = rows.map(line => line.split(","))
// map parts to case class
val allData = allSplit.map( p => CC2( p(0).toString, p(1).trim.toInt, p(2).toString, p(3).toString, p(4).toString,
p(5).toString, p(6).trim.toInt, p(7).trim.toInt, p(8).trim.toInt, p(9).trim.toInt))
// convert RDD to DATA FRAME(DF)
val allDF = allData.toDF()
// Cache the DF in order to access it faster and it will stay in memory
allDF.cache()
// Register table in order to perform SQL query on dataframe
allDF.registerTempTable("weather")
// Get avg_temp,year,month,lat,long columns and create new dataframe weather
val weather = sqlContext.sql("SELECT avg_temp,year,month,lat,long FROM weather")
//val weather = df.select(df("avg_temp"), df("year"), df("month"), df("lat"), df("long"))

// convert data to RDD which will be passed to KMeans and cache the data. We are passing in avg_temp,
year, month, lat, long columns to KMeans. These are the attributes we want to use to assign the instance to
a cluster
val train_data = weather.rdd.map(r => Vectors.dense( r.getInt(0), r.getInt(1), r.getInt(2), r.getInt(3), r.getInt(4)))

train_data.cache()

//KMeans model with 10 clusters and 500 iterations
val kMeansModel = KMeans.train(train_data, 10, 500)

// Print cluster centers
kMeansModel.clusterCenters.foreach(println)
```

```
// Get the prediction from the model and create a dataframe with Cluster column which we would later have to add to base data set
val predictions = kMeansModel.predict(train_data)
val favg = predictions.toDF("CLUSTER")

//Save Cluster dataframe
predDF.write.format("com.databricks.spark.csv").option("header", "true").save("/Users/akashmalla/Documents/COEN 242/spark dataset/wdata_results_q4")
//clusterNumberDF.save("results.csv", "com.databricks.spark.csv")

//Print the end time of this program and calculate total time for this program to run
val t1 = System.nanoTime()
println("Elapsed time: " + (t1 - t0) + "ns")

// Evaluate clustering by computing Within Set Sum of Squared Errors
val WSSSE = kMeansModel.computeCost(train_data)
println("Within Set Sum of Squared Errors = " + WSSSE)
```

IV. RESULTS

Results from each step of the Architectural Overviews are explained within this section of the paper. The Analysis focuses on HADOOP/Mahout, WEKA and Spark implementations and executions of K-Means Algorithm and provides the findings pertaining to regional weather temperature classification by employing K-Means Clustering Algorithm. Hence it aims to determine as to which regions across the Earth have the same average temperatures during different time frames of the year. And by doing so, it is possible to understand similar temperature conditions for situations such as vacationing and relocation. We also aim to determine if regions clustered together during a certain time frame of the year would continue to be clustered together during different parts of the year, and if they do change clusters then is that justified based on the recorded temperatures within those regions.

Additional the results focus on various different aspects such as clustering based on learning data set and test data set and clustering with limited attributes.

A. ETL + Data Reduction (Aggregation): Ashish to add the results from HUE

B. Execution of the K-Means Clustering Algorithm as a Map Reduce Job on Hadoop

The Execution of the Code Sequence defined within the Materials and Method Section will result in the K-Means Clustering. The Cluster Dump File defined above will help provide both the Mean (Centroid) along with the Standard Deviation of Each Cluster. Note that clustering is performed on each quarter separately. Once each entry is associated with a cluster, an overall world map can be constructed mapping each Geo Location to a Cluster. Each cluster is associated with a Median value of the average temperature

The below analysis is conducted on the Q1 Data Set. The results indicated 143 iterations to define a cluster. The below analysis reported by mahout dump logs indicates 490K Entries divided roughly equally across each Cluster. To recall, the Means (Centroids) of the Average Temperature shown below depict values which are scaled by a factor of 10. Hence if a value indicates 236.78 degree C, it would convert to 23.6 degree C.

Cluster Summary			
Cluster	Count	Step	Criterion
1	79743	143	0
2	51742		
3	14818		
4	50863		
5	36903		
6	35059		
7	53990		
8	55267		
9	61486		
10	50755		

Cluster Means					
Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	46.3255333	2012.22837	2.57478399	48162.1385	-41225.692
2	236.787465	2012.39825	2.61195934	-1157.5982	105539.038
3	70.4717236	2012.27662	1.91031178	-53255.281	7550.94318
4	130.811081	2012.21035	1	34725.8802	-61748.686
5	238.285993	2012.56239	1	-5894.5862	79965.8135
6	-127.31909	2012.81029	1.45993896	57781.6939	42122.1004
7	229.020819	2012.33999	2.53696981	13876.0966	-74038.78
8	33.30767	2015.20075	2.67170282	47087.4118	3886.32794
9	255.562681	2015.24443	2.02182611	-4392.8063	66267.5444
10	-15.128007	2015.14432	1.34103044	45491.1375	-63021.16

Cluster Standard Deviations					
Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	86.6932864	0.48035708	0.49437572	10462.2209	65608.5302
2	60.6503344	0.72658268	0.48730392	22685.0472	44654.5727
3	117.817567	0.81295366	0.71455022	15697.7273	94056.6291
4	92.4820703	0.52804992	0	16751.1378	54245.9071
5	52.7834404	0.85593562	0	21120.2854	65377.8966
6	117.399103	1.06108864	0.56086476	14281.9455	93018.0122
7	44.0610205	0.74317496	0.49863136	17767.1272	42309.6444
8	86.5470453	0.7936734	0.46959359	11050.5855	70519.3041
9	42.8171104	0.68636029	0.75368646	17762.6381	75631.3352
10	100.776218	0.82558166	0.47413868	10663.9027	62657.0594

Table 1: Q1 Mahout K-Means Clustering Summary

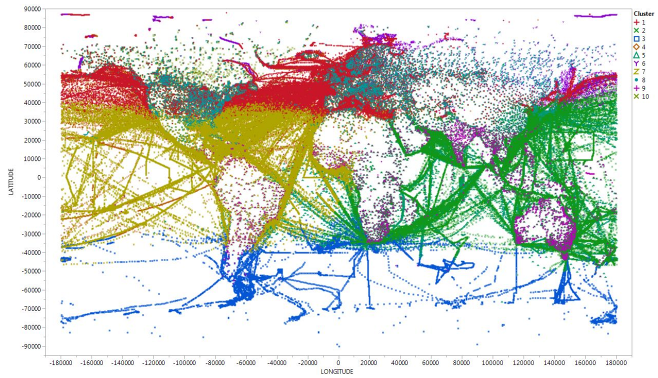

Fig 3: Q1 HADOOP/Mahout K-Means Clustering Results – All Clusters Highlighted

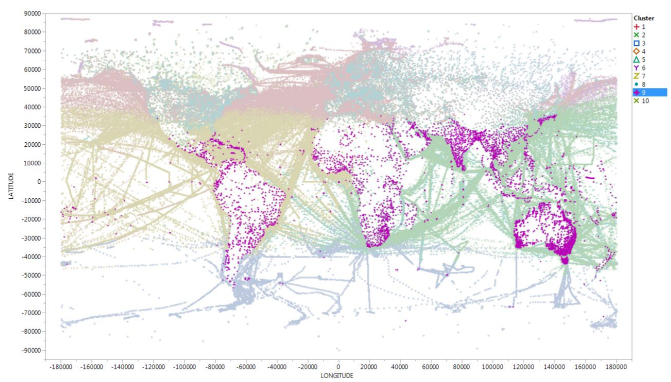

Fig 4: Q1 HADOOP/Mahout K-Means Clustering Results – Cluster 9 Highlighted

In order to validate the cluster, Fig 4 highlights cluster 9 with and Centroid of the AVG_TEMP Value of 255. Factoring in scaling factor of 10, the value is 25.5 °C with a Standard Deviation of the AVG_TEMP is 4.2 °C. When analyzing "recorded" Average Temperatures for Q1 across the regions for the last 5 years, the Temperature below can be clearly seen to be within the Standard Deviation of the centroid of 25.5 °C. All Temperature values are within 21.3 °C and 29.7°C. Hence, it successfully clusters those regions across the Earth which have the same average temperatures during Q1 time frame. By doing so, it is possible to understand similar temperature conditions for situations such as vacationing. For example, if an individual wants to vacation during Q1, and is looking for regions with Average Temperatures ~20 °C to 25 °C, then regions falling under Cluster 2,5,7,9 can be considered.

Year	Quarter	MONTH	REGION/COUNTRY/CONTINENT	RECORDED AVG_TEMP
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	PERTH, AUSTRALIA	24.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	MELBOURNE, AUSTRALIA	21.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	KOCHI, INDIA (SOUTH INDIA)	28 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	COLOMBO, SRILANKA	27.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	EAST LONDON, SOUTH AFRICA	22.67 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	LAGOS, NIGERIA	28.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	DUBAI, U.A.E	21.67 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	MANTA, EUCADOR	26.66 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	MENDOZA, ARGENTIA	24.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	PANAMA CANAL, PANAMA	27.6 °C

Table2: Q1 Recorded Regional Temperatures for areas highlighted by Cluster 9

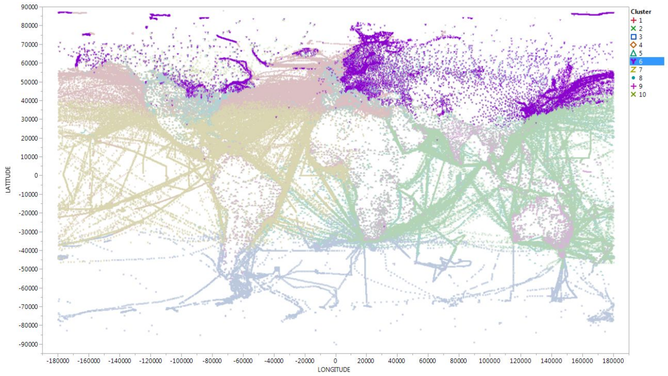

Fig 5: Q1 HADOOP/Mahout K-Means Clustering Results – Cluster 6 Highlighted

Fig 5 highlights cluster 6, with an average temperature during Q1 is -1.2 °C with a Standard Deviation of 11.7 °C. Analyzing the Average Temperatures for Q1 across a sample regions within the above highlighted regions, indicates that all the average temperature below falls under the limits of -12.9 °C to 10.5 °C

Year	Quarter	MONTH	REGION/COUNTRY/CONTINENT	RECORDED AVG_TEMP
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	MUNICH, GERMANY	1 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	INTERLAKEN, SWITZERLAND	1.6 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	DIJON, FRANCE	4.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	CHICAGO, USA	-2.3 °C

Table3: Q1 Recorded Regional Temperature for areas highlighted by Cluster 6

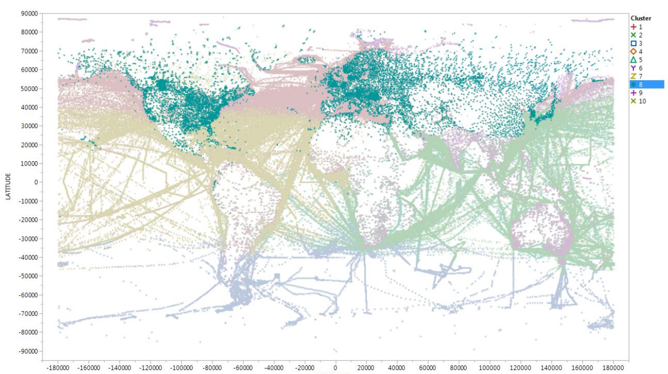

Fig 6: Q1 HADOOP/Mahout K-Means Clustering Results – Cluster 6 Highlighted

Fig. 6 Highlights cluster 8. The Centroid of the average temperature cluster, indicates an average temperature of 3.3 °C with a standard deviation of 8.6°C. Analyzing the Average Temperatures for Q1 across a sample regions within the above highlighted regions, indicates that all the average temperature below falls under the limits of -5.3 °C to 11.9 °C

Year	Quarter	MONTH	REGION/COUNTRY/CONTINENT	RECORDED AVG_TEMP
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	SAN FRANCISCO, USA	10.6 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	ATLANTA, USA	10.0 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	MILAN, ITALY	6.3 °C
2012-2016	Q1	JANUARY/FEBRUARY/MARCH	WARSAW, POLAND	-0.6 °C

Table 4: Q1 Recorded Regional Temperatures for areas highlighted by Cluster 8

C. Learning Data vs. Test Data Evaluation

The machine learning algorithm deployed on Hadoop is clustering which falls under the unsupervised learning class and hence it doesn't have any class label for classification. However the Q1 Data Set was broken into "Learning Data Set" of 71% and "Testing Data Set" 29%.

The Learning Data Set generates a cluster formula which was applied on the Training Dataset to determine the cluster for which each data entry belongs to. Table 2 and Figure 5 represent the clustering analysis performed on the Learning Data Set. As shown above 10 clusters were generated post 70 Clustering Iterations.

Cluster Summary

Cluster	Count	Step	Criterion
1	23568	70	0
2	34000		
3	39012		
4	58897		
5	39566		
6	37230		
7	20718		
8	44556		
9	19228		
10	33225		

Cluster Means

Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	248.769603	2015.03704	1.50063646	-2932.5047	-32625.456
2	253.745294	2014.90788	1.57617647	-3803.3692	117278.713
3	227.939455	2012.09597	1.47572542	-10110.789	115423.901
4	189.308963	2012.10467	1.44640644	27377.8426	-58625.532
5	-2.6184856	2015.18137	2.10177931	47033.9395	-23104.349
6	-26.724953	2015.1177	1	47054.3612	-23296.635
7	188.396756	2012.12535	1.5653538	-30799.168	-40073.138
8	2.34127839	2012.32005	1.56870006	49918.237	51205.7102
9	175.442012	2012.51597	3	23039.627	-4138.0754
10	-59.971919	2012.32918	1.52710309	53841.2539	-10161.81

Cluster Standard Deviations

Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	40.0037098	0.80431474	0.51685731	17558.7491	44424.5199
2	50.5132045	0.80160661	0.56200018	18490.8369	34441.1075
3	75.1675972	0.29818337	0.5055828	26252.0826	40366.9034
4	59.6772082	0.35219552	0.49711943	12355.5262	47098.0309
5	98.1254279	0.79149207	0.30235786	11232.1771	75067.2772
6	103.662311	0.80846706	0	10627.0272	76266.0066
7	100.056707	0.42050598	0.51434725	18755.1203	60687.2171
8	102.43738	0.56513395	0.49828481	12204.5687	60250.7458
9	95.2725286	0.9795628	0	23728.5477	86406.5393
10	117.478914	0.57692681	0.52385799	13208.2534	36012.2956

Table 5: Q1 Mahout K-Means Clustering Summary for Learning Data Set.

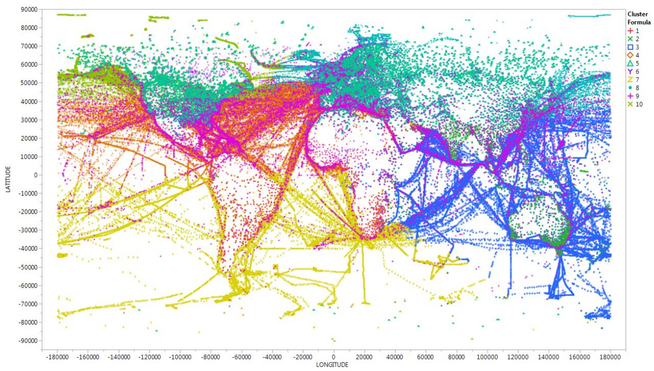

Fig 7 Q1 HADOOP/Mahout K-Means Clustering Results – All Clusters Highlighted for Learning Data Set

Note that analysis shown in Fig 7 from the Learning Data indicates different colors than before in Fig 3 (**The colors are independent and randomly generated. Cluster colors need to me associated with Cluster Means and as long as the Clusters Means are associated with the same regions, the data is identical)**

The above analysis from the learning dataset is similar to that of running clustering on the complete data set. For example, regions in Australia and India associate themselves to Cluster 2 and parts of South America and Africa which should have belonged to the same cluster, now associate themselves to Cluster 1. However, note that the Average Temperature of Cluster 1 is 24.8 °C and that of cluster 2 is 25.8 °C. The reason for different cluster associations is because the K-Mean Clustering Algorithm is also clustering by Year, Month and Geo Location. The following is depicted in Fig 8.

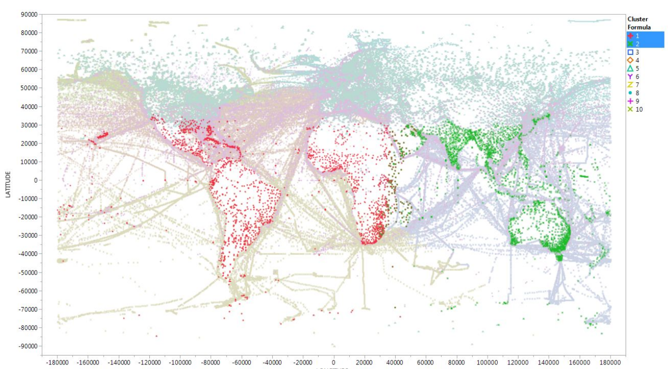

Fig 8 Q1 HADOOP/Mahout K-Means Clustering Results – Clustering 1,2 Highlighted for Learning Data Set

Testing Data Set comprises of 29% of the data and deploys the clustering formula which is derived from the Learning Data Set. The formula aims to classify each entry to either Cluster 1 or Cluster 10. Formula Snippet for Cluster 5 and 6 are shown below. Similar formulae can be derived for the remaining clusters.

$$\mathrm{IfMin}\left|\begin{array}{l}\left(\frac{(AVG\_TEMP-(-2.6184855684173))}{147.325644727447}\right)^2 \\ +\left(\frac{(YEAR-2015.18136784108)}{1.53213622077525}\right)^2 \\ +\left(\frac{(MONTH-2.10177930546429)}{0.61958122158489}\right)^2 \\ +\left(\frac{(LATITUDE-47033.9394935045)}{31463.2778153995}\right)^2 \\ +\left(\frac{(LONGITUDE-(-23104.349188697))}{91434.0932368421}\right)^2\end{array}\right. \Rightarrow 5 \quad \left|\ \mathrm{IfMin}\left|\begin{array}{l}\left(\frac{(AVG\_TEMP-(-26.724952994897))}{147.325644727447}\right)^2 \\ +\left(\frac{(YEAR-2015.11770077894)}{1.53213622077525}\right)^2 \\ +\left(\frac{(MONTH-1)}{0.61958122158489}\right)^2 \\ +\left(\frac{(LATITUDE-47054.3611603546)}{31463.2778153995}\right)^2 \\ +\left(\frac{(LONGITUDE-(-23296.634756916))}{91434.0932368421}\right)^2\end{array}\right. \Rightarrow 6\ \right|$$

Fig 9 indicates the Testing data once the cluster formula is applied to the dataset. It is important to note since this data set is only 29% of the original Q1 weather dataset, all clusters are not present within this data set. The strength of the cluster prediction is seen within Fig 9 where only Clusters 1 and 2 are highlighted to indicate clustering formula successfully predicted the entries in question. It was able to successfully classify the same regions as the learning data set. Along with that, the clustering indicates successfully the regions which have the same temperatures. Hence if any data entry is provided with the 5 attributes shown above, it is now trivial to judge as to which cluster it would belong to and for which regions are the average temperatures the same during the Q1 Time Frame.

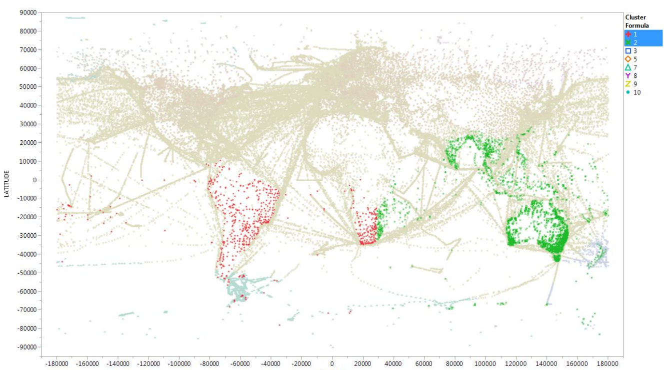

Fig 9 Q1 HADOOP/Mahout K-Means Clustering Results – Clustering 1,2 Highlighted for Testing Data Set

D. K-Means Clustering as Map Reduce Jon on HADOOP/Mahout (with limited attributes)

One might argue that the above clustering is possibly biased since Geo Location (Latitude and Longitude) are part of the clustering algorithm. Although this is not a biased approach, since proximity is an important aspect of temperature clustering, it is also possible to perform clustering by considering Year, Month, and Average Temperature.

Cluster Summary				Cluster Means				Cluster Standard Deviations			
Cluster	Count	Step	Criterion	Cluster	AVG_TEMP	YEAR	MONTH	Cluster	AVG_TEMP	YEAR	MONTH
1	53349	28	0	1	220.45789	2012.07303	1	1	49.9030597	0.26018336	0
2	13642			2	-245.27591	2012.70789	1.49802082	2	77.5873965	0.94322526	0.53237418
3	31036			3	238.306902	2014.91996	1	3	54.9751422	0.81294761	0
4	40793			4	26.6887946	2012.20003	2.00808962	4	65.2508038	0.40002574	0.08957779
5	34480			5	-31.82964	2015.1241	1	5	87.5290285	0.81010104	0
6	35225			6	241.219134	2014.98811	2.08516678	6	52.7259507	0.80276343	0.27912973
7	36996			7	-14.698319	2015.15202	2.09244243	7	83.7479101	0.79507926	0.28964948
8	36681			8	21.9588888	2012.21523	1	8	62.6551171	0.41098461	0
9	49300			9	226.397444	2012.08043	2	9	49.0409682	0.27195152	0
10	18498			10	167.570386	2012.33685	3	10	97.1381598	0.68879267	0

Table 6: Q1 Mahout K-Means Clustering Summary with Limited Attributes

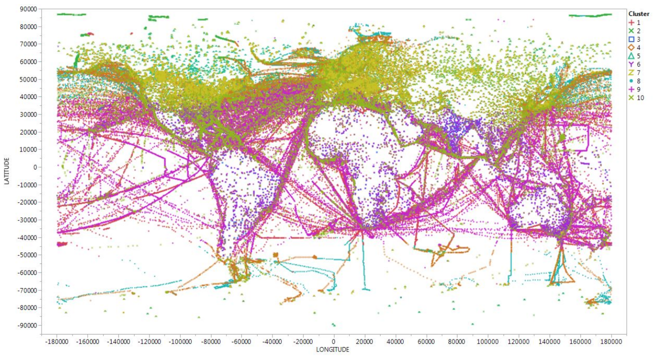

Fig 10 Q1 HADOOP/Mahout K-Means Clustering Results w/ Limited Parameters. All Clusters Highlighted.

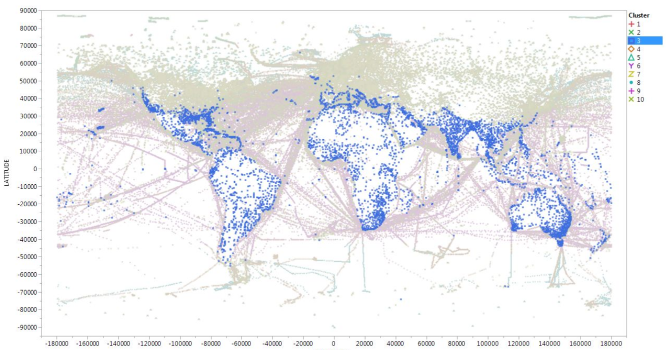

Fig 11 Q1 HADOOP/Mahout K-Means Clustering Results w/ Limited Parameter. Custer 3 Highlighted.

Fig 10 is an overlay of the world map with the cluster shading. Since Geo Location, is not used for the clustering, it is important to begin by assessing the accuracy of the clusters in question. Fig 10 where have highlighted only cluster 3, indicates that we get similar clustering as shown in Fig 4. Cluster 3 in Fig 11 indicates an Average Temperature of 23.8°C. As the analysis above has dictated, this is an accurate average temperature of the highlighted regions. Hence the data proves that limited attribute clustering provides the same results as that which is provided by clustering which includes geo location

E. K-Means Clustering Algorithm as a Map Reduce Job on Hadoop (Q2 → Q4 Analysis)

The analysis shown in the previous sub sections is focused on Q1 which comprised of January, February and March Time Frame. Similar Analysis can be conducted for the remaining quarters and clusters can be validated in a similar fashion and regions with similar temperature conditions can be highlighted. As mentioned before, depending on an individual's temperature preference, similar regions can be explored for vacationing during those Quarters or perhaps even for relocation purposes. This analysis was conducted on the remaining Quarters, with the final testing data sets, but instead of focusing on the world-map, the result of Australia and India were focused on. This helps to answer the question if Australia and India are indeed clustered under the same cluster across all Quarters or do their average temperatures change across Quarters.

Cluster Summary				Cluster Means						Cluster Standard Deviations						Cluster
Cluster	Count	Step	Criterion	Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE	Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE	
1	42906	86	0	1	11.6389083	2012.2157	4.19549713	58612.302	-42643.458	1	87.1587549	0.46347652	0.39675918	13837.0479	80391.0022	1
2	46136			2	82.4984611	2014.98916	4.23211809	48822.424	-23976.02	2	66.9756276	0.77971427	0.42249187	10538.3395	78620.5334	2
3	51528			3	238.060977	2012.2141	4.55323319	-293.18854	10211.943	3	59.2653833	0.41019457	0.49715815	23901.6661	46476.3964	3
4	62657			4	241.982683	2012.19605	5.54380197	18563.1369	-64020.894	4	38.1440801	0.41316153	0.49807769	19811.0617	49199.6733	4
5	45116			5	252.603444	2014.77039	4.37831368	-2766.1437	72641.5112	5	55.3250264	0.72369019	0.48496643	19260.6019	72885.3608	5
6	44202			6	195.62108	2014.50785	5.68139451	37189.9546	-30291.948	6	59.4056018	0.53631052	0.46593565	18598.002	69252.1483	6
7	40214			7	239.006068	2012.89305	6	1261.33742	103692.357	7	65.6946497	1.04795016	0	23189.4396	46883.6313	7
8	85799			8	109.191634	2012.19371	5.5151109	53123.9335	-37309.058	8	52.301277	0.42501535	0.49977161	11375.5179	72985.8718	8
9	41059			9	210.689276	2012.20843	4	21082.3447	-58775.225	9	52.6736574	0.51097316	0	19456.331	51223.2015	9
10	5788			10	-72.218556	2012.86265	4.86990325	-60463.796	-66191.482	10	163.792986	0.99877294	0.81611249	14369.1914	83613.2722	10

Table 7: Q2 Mahout K-Means Clustering Summary

Cluster Summary				Cluster Means						Cluster Standard Deviations						Cluster
Cluster	Count	Step	Criterion	Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE	Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE	
1	33917	90	0	1	14.8289059	2012.08365	8.54217649	73740.8561	-48982.869	1	42.5758501	0.35737188	0.50210855	9094.27736	93418.8749	1
2	62080			2	253.353882	2012.19546	8.40534794	20569.3179	-71984.98	2	33.9579178	0.44595691	0.51538546	19699.6341	48902.8256	2
3	29776			3	136.459699	2012.81069	7.94676921	-32608.767	77360.163	3	47.9542848	1.04258209	0.78264158	8419.50563	84118.6668	3
4	58327			4	174.517805	2014.53116	8.17626485	47392.2817	-38627.147	4	57.451559	0.49902808	0.78296911	11663.0231	70573.7528	4
5	62338			5	160.085117	2012.20753	8.63548718	49872.6955	-18208.066	5	40.0068974	0.40553822	0.48129328	8120.15435	63666.1326	5
6	55295			6	122.64315	2012.18618	7.17913012	56836.0919	-75842.316	6	59.724583	0.45422023	0.38346124	13011.4912	66928.3002	6
7	1001			7	-270.07792	2013.28871	8.04495504	-70177.531	33623.1808	7	185.638827	1.08737301	0.8138272	14379.7327	96500.2407	7
8	58905			8	255.977082	2012.24271	7.00022069	19744.9467	21548.0931	8	40.9060551	0.42872289	0.01485415	20577.0007	84606.0943	8
9	50873			9	259.554675	2014.55525	7.69457276	12145.0304	44345.8749	9	40.7073999	0.49693857	0.73971063	20877.1474	75089.2157	9
10	49466			10	266.401953	2012.48207	8.57134193	11272.026	101733.323	10	34.2732157	0.69119774	0.49488416	18862.2098	41688.0587	10

Table 8: Q3 Mahout K-Means Clustering Summary

Cluster Summary

Cluster	Count
1	58965
2	81595
3	47336
4	52197
5	25634
6	53303
7	50991
8	37554
9	30459
10	28973

Step	Criterion
54	0

Cluster Means

Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	47.0395489	2014.6635	11.2263207	46833.9302	-34269.463
2	87.0356885	2012.17769	11.459685	45263.8617	-43393.556
3	241.433877	2014.54487	10.6973551	-477.61235	55718.7728
4	227.998448	2012.23475	11.5267736	1747.4296	-65882.435
5	-34.547164	2012.73988	11.2718265	51779.277	109177.91
6	240.512767	2012.44292	11.5605876	-5696.7096	106516.578
7	78.3522975	2012.52184	10	53951.5891	-50963.059
8	230.960377	2012.18467	10.0004527	19459.7001	-57236.325
9	-207.97649	2012.10122	11.0500016	79509.0618	-119568.78
10	228.406482	2012.25462	10	-1437.3651	102079.928

Cluster Standard Deviations

Cluster	AVG_TEMP	YEAR	MONTH	LATITUDE	LONGITUDE
1	83.4991588	0.4725138	0.74695175	11210.9374	68359.3995
2	70.3163118	0.38225554	0.49837205	9093.41847	54224.3144
3	58.9528695	0.49798255	0.72166064	21838.7727	75598.1886
4	63.4113104	0.54526098	0.49951284	24705.5431	44779.5158
5	106.753575	0.86001342	0.7139946	15048.8136	46953.2789
6	61.0177312	0.6720182	0.49631557	22299.3488	43595.4596
7	60.6996995	0.77224909	0	11275.1879	64502.2335
8	46.6921403	0.47927522	0.0212715	21392.0778	51485.014
9	86.1180147	0.35640062	0.78448227	9099.08503	37681.7935
10	68.9337023	0.43683214	0	24912.4042	47649.6127

Cluster: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Table 9: Q4 Mahout K-Means Clustering Summary

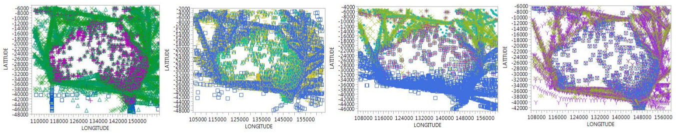

Fig 12 Q1→Q4 Average Temperature Clustering for Australia

Year	Quarter	Month	Region	Cluster Number: Temp Range	Recorded Average Temperature
2012-2016	Q1	Jan-Feb-March	BRISBANE, AUS	9:25.5 +/- 4.2 °C	25°C
2012-2016	Q2	April-May-June	BRISBANE, AUS	3: 23.8 +/- 5.9°C	19°C
2012-2016	Q3	July-August-Sep	BRISBANE, AUS	3: 13.6 +/-4.7°C	16°C
2012-2016	Q4	Oct-Nov-Dec	BRISBANE, AUS	3: 24.1 +/- 5.8 °C	23°C

Table 10: Brisbane, Australia Q1→Q4 Average Temperature Comparison between Actual Recorded Average Temperature vs HADOOP/Mahout Cluster Average Temperature Range

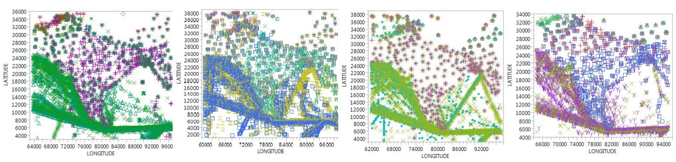

Fig 13 Q1→Q4 Average Temperature Clustering for India

Year	Quarter	Month	Region	Cluster Number: Temp Range	Recorded Average Temperature
2012-2016	Q1	Jan-Feb-March	KOCHI, INDIA	9:25.5 +/- 4.2 °C	28°C
2012-2016	Q2	April-May-June	KOCHI, INDIA	3: 23.8 +/- 5.9°C	28.3°C
2012-2016	Q3	July-August-Sep	KOCHI, INDIA	10: 22.8 +/-6.8°C	25.6°C
2012-2016	Q4	Oct-Nov-Dec	KOCHI, INDIA	3: 24.1 +/- 5.8 °C	27°C

Tabel 11: Kochi, India Q1→ Q4 Average Temperature Comparison between Actual Recorded Average Temperature vs HADOOP/Mahout Cluster Average Temperature Range

As seen within the analysis shown above, both regions tracked within Australia and India don't necessarily follow the same average temperature trends across all quarters. Therefore, they are not clustered by the same cluster number. When considering Q3, it can be seen that Kochi, India is represented by cluster 10, whereas Brisbane, Australia, is represented by cluster 3. Hence, this indicates that depending on the Month/Quarter in question, the algorithm doesn't continue to cluster regions within the same average temperature profiles. This analysis however is accurate since when checking the recorded average temperatures, both the regions in Australia and India don't follow the same average temperature profiles. Hence the algorithm needs to cluster these regions separately which is exactly what the algorithm tends to do. Also note that the Average Temperature is depicted as a Range, since it represents the Centroid Average Temperature across the standard deviation range.

F. Executing the K-Means Clustering Algorithm on WEKA

Similar to the results on Hadoop, WEKA generates an output log file to provide the Mean (Centroids) of the Cluster. Note that the clustering is performed on each quarter separately. Once each entry is associated within a cluster, an overall world map can be constructed by mapping each Geo Location to a Cluster. Each cluster is associated with a median value of average temperature.

The below evaluation is performed on the Q1 WEKA reduced Data Set. The results indicated 25 iterations to define a cluster. The below analysis reported by WEAK logs indicates that all entries divided across each cluster ranging from 6% to 13% entries associated with a cluster and at an average of 10% date entries within each entry. To recall, the Means (Centroids) of the Average Temperature shown below depict values which are scaled by a factor of 10. Hence if a value indicates 236.78 degree C, it would convert to 23.6 degree C. Note that similar to the Mahout/Hadoop Analysis, the WEKA Data Set can be broken into Learning and Test Data Set and similar analysis as before can be performed on the Data Set.

WEKA indicates the Starting Cluster and eventually provides the Final Cluster Centroids. The following information is extracted from the WEKA data log and added into a table for ease of readability. The information provided is for the Final Cluster.

Quarter 1		Cluster Number									
Attribute	Full Data	0	1	2	3	4	5	6	7	8	9
AVG_TEMP	115.8424	87.7561	97.345	168.1304	207.133	199.5391	178.0472	51.5289	107.965	94.9522	80.0485
YEAR	2013.3392	2015.0315	2012.1402	2012.2959	2015.4712	2012.5707	2012.5131	2015.0546	2012.1341	2012.1434	2015.0636
MONTH	2.0089	3	2	1	2.5528	3	2	2	3	1	1
LATITUDE	24444.9297	36160.9079	33643.1112	-882.2334	1657.19	3246.6559	-439.8338	35901.9371	34483.4193	34563.4788	27565.2836
LONGITUDE	1411.0888	-34672.3037	-55295.9148	108472.993	117062.762	104188.916	106813.9528	-33181.0666	-55200.7482	-58070.8538	1716.4951

Table 12 Q1 WEKA K-Means Clustering Summary

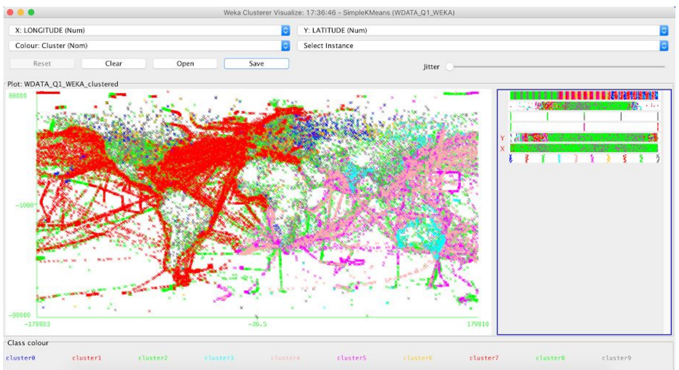

Fig 14 Q1 WEKA K-Means Clustering Results – All Clusters Highlighted

WEKA reduction algorithm should maintain the distribution of the "50MB Original Data Set" when reduced to the WEKA Data Set of 5-10MB. Hence, the analysis on the Test Data Set on WEKA should provide similar clustering results as that performed on the 50MB Original Data Set with HADOOP/Mahout. If the results of the clustering from WEKA after applying the cluster formula from the learning data set to the test data set, match that of HADOOP/Mahout in terms of regions clustered and the Mean (Centroid) of the Average Temperature then it is an excellent indicator that the WEKA Algorithm reduced the data while maintaining the distribution. It also proves that WEKA also provides similar final results as HADOOP/Mahout on a smaller data set which can be used to determine regions with similar temperatures during similar quarters.

When analyzing the results on Q1 on the WEKA Reduced Data Set (Test Data Set), we begin by focusing on the same regions of Australia, India, Africa and South America. From Fig 14, these regions are represented by Cluster 3 (Turquoise Cluster Color), which indicates and AVG_TEMP Value of 207.133. Factoring in scaling factor of 10, the value is 20.7°C. When comparing the results of this analysis to that of HADOOP/Mahout on the complete data set, from the earlier analysis, we notice similar results. Since we have validated the actual recorded average temperatures of the regions during the past 5 years and during Q1, we already know that that the temperatures of the regions match that of the centroid of the cluster (+/- Standard Deviation). Hence this data clearly proves that WEKA reduced data set can provide similar final results as HADOOP/Mahout on a smaller data set and is successful at clustering regions with similar temperatures for Q1 and it successfully clusters those regions across the Earth which have the same average temperatures during Q1 time frame.

Hence, WEKA also successfully clusters the regions across the Earth which have the same average temperatures during Q1 time frame. By doing so, it is possible to understand similar temperature conditions for situations such as vacationing or relocation. For example, if an individual wants to vacation during Q1, and is looking for regions with Average Temperatures ~20 °C to 25 °C, then regions falling under Cluster 2,5,7,9 can be considered.

G. K-Means Clustering Algorithm on WEKA (Q2 → Q4 Analysis)

The Q1 Analysis shown above indicates the effectiveness running the clustering K-Means Algorithm on WEKA. This effectiveness of WEKA can also been seen by conducting the same analysis as performed on HADOOP/Mahout for Q2 to Q4. However, as explained above, this analysis which is done on WEKA is done on the WEKA reduced Q2, Q3, Q4 Data Sets which is 5-10MB in size. Similar to the Hadoop analysis, depending on an individual's temperature preference, similar regions can be explored for vacationing during those Quarters or perhaps even for relocation purposes. This analysis was conducted on the remaining Quarters, with the final testing data sets, but instead of focusing on the world-map, the result of Australia and India were focused on. This helps to answer the question if Australia and India are indeed clustered under the same cluster across all Quarters or do their average temperatures change across Quarters.

Quarter 2		Cluster Number									
Attribute	Full Data	0	1	2	3	4	5	6	7	8	9
AVG_TEMP	169.4763	225.105	199.2035	165.5079	146.2561	141.7533	209.1776	210.7457	124.3544	167.2747	221.6332
YEAR	2013.0143	2012.2902	2014.4945	2012.1708	2015.1114	2012.1729	2014.5117	2012.5376	2012.1382	2014.4717	2012.2946
MONTH	4.9666	6	5	6	4	5	6	4	4	5.4972	5
LATITUDE	27028.7059	7959.7065	18255.0584	38777.7559	27386.338	37792.9052	19444.6668	4636.6208	33977.2778	41186.9253	6166.0266
LONGITUDE	-3530.5597	102294.3681	56376.9759	-58465.1522	135.8513	-56918.8561	55363.5098	105180.1602	-57321.995	-101240.8973	102923.4388

Quarter 3		Cluster Number									
Attribute	Full Data	0	1	2	3	4	5	6	7	8	9
AVG_TEMP	191.5972	222.2585	157.4159	214.8551	157.3294	177.0765	189.92	230.0816	213.7991	192.9567	212.6473
YEAR	2012.8222	2012.2641	2012.137	2014.6588	2012.1236	2012.0859	2012.1831	2012.2295	2012.2781	2014.5029	2013.7083
MONTH	7.9994	8	9	7.6713	8	7	8	7	9	9	7
LATITUDE	30513.3066	8121.4279	42538.13	26222.8426	53021.9215	42583.9221	33970.3015	4283.1422	9864.591	25502.9075	26836.4805
LONGITUDE	-5991.8713	108207.127	-62182.9104	7425.1998	-4461.2302	-60557.0391	-102048.4099	105563.1389	99745.8503	8173.3861	7184.1051

Quarter 4		Cluster Number									
Attribute	Full Data	0	1	2	3	4	5	6	7	8	9
AVG_TEMP	124.4817	110.6866	203.065	177.3681	67.95	190.3448	162.4484	73.1147	154.3649	149.2077	-101.8097
YEAR	2012.8355	2012.1323	2012.2696	2012.2878	2012.166	2012.2746	2014.5065	2014.5315	2012.1466	2014.475	2012.1515
MONTH	10.966	10	10	12	12	11	10	11.305	11	11.6763	11
LATITUDE	28753.7125	44174.4753	7813.4961	3417.763	37918.9471	5967.0482	25700.2225	37874.2784	29686.7694	15949.0757	67878.8928
LONGITUDE	-7276.7572	-66786.2582	93802.7377	96514.079	-63826.4558	109009.647	8104.9339	-57322.4816	-40562.3538	70108.4291	-119373.5738

Table 13: Q2→ Q4 WEKA K-Means Clustering Summary

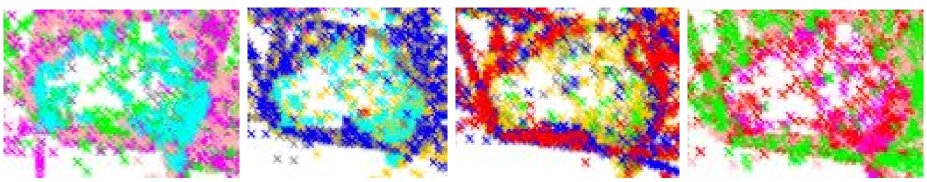

Fig 15 Q1→ Q4 WEKA Average Temperature Clustering for Australia

Year	Quarter	Month	Region	Cluster Number: Average Temp	Recorded Average Temperature
2012-2016	Q1	Jan-Feb-March	BRISBANE, AUS	3: 20.7°C	25°C
2012-2016	Q2	April-May-June	BRISBANE, AUS	3: 14.6°C	19°C
2012-2016	Q3	July-August-Sep	BRISBANE, AUS	4: 17.7°C	16°C
2012-2016	Q4	Oct-Nov-Dec	BRISBANE, AUS	3: 20.1 °C	23°C

Table 14: Brisbane, Australia Q1→Q4 Average Temperature Comparison between Actual Recorded Average Temperature vs WEKA K-Means Cluster Average Temperature Range

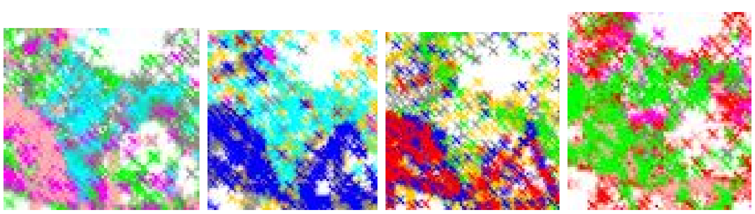

Fig 16: Q1→Q4 WEKA Average Temperature Clustering for Australia

Year	Quarter	Month	Region	Cluster Number: Average Temp	Recorded Average Temperature
2012-2016	Q1	Jan-Feb-March	KOCHI, INDIA	3: 20.7 °C	28°C
2012-2016	Q2	April-May-June	KOCHI, INDIA	0: 22.5 °C	28.3°C
2012-2016	Q3	July-August-Sep	KOCHI, INDIA	0: 22.2 °C	25.6°C
2012-2016	Q4	Oct-Nov-Dec	KOCHI, INDIA	1: 20.1 °C	27°C

Table 15: Kochi, India Q1→Q4 Average Temperature Comparison between Actual Recorded Average Temperature vs WEKA Cluster Average Temperature Range

Similar to the analysis conducted on HADOOP/Mahout, as seen within the analysis shown above, both regions tracked within Australia and India do not necessarily follow the same average temperature trends across all quarters. Therefore, they are not clustered by the same cluster number. When considering Q2, Q3, Q4, it can be seen that Kochi, India is represented by a different cluster when compared to Brisbane, Australia. Hence, this indicates that depending on the Month/Quarter in question, the algorithm could provide different clusters, and although Kochi, India and Brisbane, Australia might be clustered the same during the first quarter, that might not be the case in the second quarter. This analysis however is accurate since when checking the recorded average temperatures, both the regions in Australia and India don't follow the same average temperature profiles. Hence the algorithm needs to cluster these regions separately which is exactly what the algorithm tends to do. Also note that compared to the HADOOP/Mahout K-Means Clustering which was performed on the completed data set, whereas the above analysis on Q2 to Q4 has clustering performed on WEKA reduced data set. Hence the Average Temperature range depicted by the cluster is different from the results from Mahout, although they fall within the standard deviation.

H. SPARK/SCALA K-Means Clustering Results

As explained within the Methods and Materials Section, K-Means Clustering within SPARK is performed on the complete 200MB Data Set which was divided into 4 Quarters. The results shown are only for Q1, however similar to Mahout/WEKA results for Q2→ Q4 can be achieved in the same fashion.

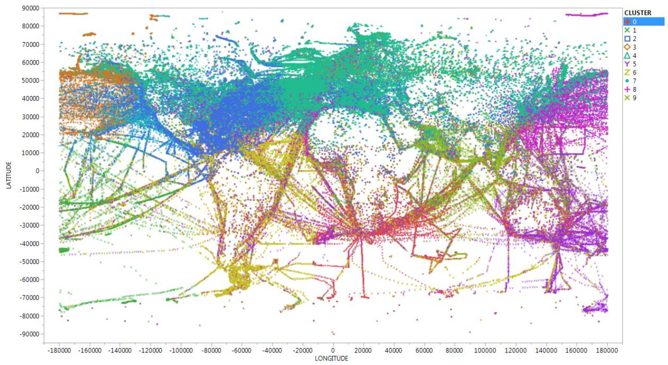

Fig 17: Q1 SPARK Average Temperature Clustering – All Clusters Highlighted

When analyzing the results on Q1 on the SPARK Data Set (Test Data Set), we begin by focusing on the same regions of Australia, India, Africa and South America. From Fig 17, these regions are represented by Cluster 3 (Turquoise Cluster Color), which indicates and AVG_TEMP Value of 255. Factoring in scaling factor of 10, the value is 25.5°C. When comparing the results of this analysis to that of HADOOP/Mahout on the complete data set or WEKA on the reduced Data Set, from the earlier analysis, we notice similar results. Since we have validated the actual recorded average temperatures of the regions during the past 5 years and during Q1, we already know that that the temperatures of the regions match that of the centroid of the cluster (+/- Standard Deviation). Hence this data clearly proves that the analysis on SPARK has resulted in provide similar final results as HADOOP/Mahout or WEKA, and it successfully clusters those regions across the Earth which have the same average temperatures during Q1 time frame.

I. Performance Analysis

The performance analysis is broken down as per the Architectural Overview Steps

Data Preparation:
- Initial Data Set (150Gb) → ETL
- ETL → DATA REDUCTION (Aggregation): Hadoop Map Reduce Job to reduce 150GB to 200MB Time: 40 Mins

HADOOP/Mahout: Preparation of Vectors + Executing of the K-Means Algorithm as a Map Reduce Job
- Q1 Execution: 8.36 Mins
- Q2 Execution: 4.95 Mins
- Q3 Execution: 5.71 Mins
- Q4 Execution: 9.03 Mins

WEKA: Includes Data Reduction + Executing of K-Means on WEKA.
- Reduction of a 50 MB File to 5-10MB: 2 Seconds

- Q1 Execution: 3.63 Seconds
- Q2 Execution: 3.66 Seconds
- Q3 Execution: 3.26 Seconds
- Q4 Execution: 3.55 Seconds

SPARK: Preparation of RDD/Data Frames + Executing of the K-Means Algorithm

- Q1 Execution: 2.86 Mins
- Q2 Execution: 1.91 Mins
- Q3 Execution: 2.03 Mins
- Q4 Execution: 2.93 Mins

Note that due to size limitations faced on WEKA, the overall files size had to be significantly reduced from 150GB to 4GB and since the 4GB data set also couldn't be loaded into WEKA for WEKA Reduction to be performed, additional aggregation was performed to reduce the data set to 200MB so that the base dataset of 200MB could be used on WEKA, HADOOP/Mahout, and SPARK.

Due to this limitation which is faced, all performance analysis conclusions can only be drawn based on the 200MB Data Set. And hence for a smaller file size, WEKA indicates a better performance than HADOOP.

For example when just considering a single Quarter for Analysis, for example Q1, the amount of time taken for HADOOP/Mahout is 8.36 Mins, whereas WEKA executes in matter of seconds (3.63 to be exact). The reasoning for this is high performance on WEKA is that it operates In-Memory. Last but not the least, the amount of time SPARK took to run for Q1 results is 2.86 seconds which is less than half the time taken for HADOOP. Also, a great benefit of SPARK other than the fact that it does in memory processing is it does "lazy evaluation," which means, any transformation on the data set is not computed right away until and unless there is a requirement to return result to the driver program. Once the data set was loaded into RDD, it was cached which enabled fast data access, data was available in the memory. Therefore, we can see that SPARK performs better than WEKA.

However, WEKA although a popular and comprehensive Data Mining Workbench with a well-known and intuitive interface, nonetheless it supports only sequential single-node execution. Hence the size of the datasets and the processing tasks that WEKA can handle with the existing environment is limited bit by the amount of memory in a single node and by sequential execution. Hence for larger dataset WEKA would crash for larger dataset and due to its sequential processing it would be slower for larger data sets, since it's not running a distributed model.

V. CONCLUSION

This project did not come without its challenges. To run the k-means cluster on Mahout, Weka, & Spark was tough, however the part which caused the most amount of struggle is getting a data set that could be used across all three. Given the restriction on space of the Linux system, we had to be creative in reducing the data and used a combination of map-reduce and aggregation to get the initial data set of 150GB to a 200mb file size. After getting a workable size, the k-means cluster algorithm was run on Mahout, Weka and Spark. After execution we were left with clusters centered on the mean of temperature values. Looking at execution time of Mahout, Weka and Spark we were able to conclude that Weka. Comparing Q1 across all three we see that Weka was the fastest executing in 3.63sec, followed by Spark in 2.86min and then Mahout in 8.36min. From these results a conclusion can be made that Weka would be best to use when trying to perform machine learning algorithms on large sets of data given there is no limitation on space.

References

United States. National Oceanic and Atmospheric Administration. US Department of Commerce. N.P: N.P.,N.D. Web < ftp://ftp.ncei.noaa.gov/pub/data/noaa/>.

"WeatherSpark Beta." *Beautiful Weather Graphs and Maps.* N.p., Web