Improvements in K-Means Algorithm: A case-study on previously used techniques

Sinjoni Mukhopadhyay
University of California, Santa Cruz
simukhop@ucsc.edu

Abstract

Clustering problems arise in many different applications: machine learning data mining and knowledge discovery, data compression and vector quantization, pattern recognition and pattern classification. The main aim of clustering is to partition a set of objects which have associated multi-dimensional attribute vectors into homogeneous groups such that the patterns within each group are similar. The k-means algorithm is considered as the best-known squared error based clustering algorithm. It is simple and can be easily implemented in solving many practical problems. Although reasonably efficient in execution, the algorithm does encounter quite a lot of bottlenecks like – Sensitivity to initial conditions, Convergence to local optimum than global optimum, Complexity of K-means, Sensitivity to outliers and noise, Application limited only to numerical variables. There has been a lot of previous research that has been done to come up with newer methods to take care of these shortcomings in the k-means algorithm. The main aim of this paper is to compare the results of an analysis of the representative works related to the research lines of k-means algorithm devoted to overcome its shortcomings.

1 RELATED WORK

A lot of research has already been done to improve the k-means algorithm and they have come up with numerous methods to get rid of each shortcoming. The sensitivity to initial conditions has been tackled using strategies and algorithms like the sub-merger strategy, the particle swarm optimization algorithm, defining three new algorithms: Weighted k-means (WK), Inverse Weighted K-means (IWK) and Inverse Exponential k-means (IEK), using kd-trees, patterning selection of value of K and comparing initialization methods. The convergence of algorithm to a local optimum rather than a global optimum is taken care of using a new convergence condition method, estimation of lower limit for local optimum, using Genetic K-means algorithm etc. Improvement in complexity of algorithm was done using partial distance algorithm, Rival algorithm, using something called GAKREM which eliminates the need to specify a priori the number of clusters and combines genetic algorithms and logarithmic regression Maxima expectation, implementation using core-sets and entropy weighted measures etc. Sensitivity to outliers and noise was improved using the Geodesic K-means clustering and processes effect of outliers on clustering by performing dynamic estimation of K. Restriction to only numerical variables was tested by using K-means on categorical attributes and by using a genetic algorithm that finds similarities conceptually.

2 BACKGROUND

2.1. INTRODUCTION TO CLUSTERING

Clustering aims to classify data from the whole data space. The difference between each data object in the same class is similar. However, the difference between each data objects in different classes is large. Clustering belongs to the unsupervised learning method and it can automatically sort data sets. Basically, the result of clustering algorithm is to find the same classification of different data in the whole data sets. A clustering algorithm groups all the same kind of data into one single class. The computer will recognize the specific features of all data so that it can separate data to the proper classes.

2.2. K-MEANS ALGORITHM

The K-means algorithm is based on the distance from each data to the initial cluster centers. The distance is the evaluation standard for the similarity of the data. This means that if the distance between two objects is small, then the similar level is high. In the K-means algorithm, an initial set of k as the clustering point is chosen. The computer will find all the data which is close to the initial set of k. After that, by using the method of iteration, the computer will update the value of k to get the new cluster for the rest of data. Then, the computer will retrieve the best result after running it again and again.

The formula for K-Means algorithm is:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

All the results will be related to the initial set of k, therefore the random value of k is very important to the whole algorithm system.

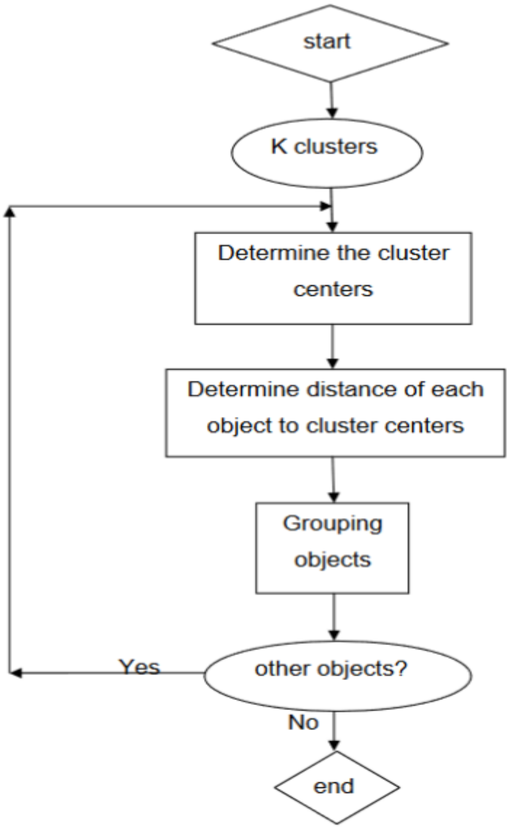

start
K clusters
Determine the cluster centers
Determine distance of each object to cluster centers
Grouping objects
other objects?
Yes
No
end

FIGURE 1: K-MEANS FLOWCHART

2.3. ADVANTAGES OF K-MEANS

MacQeen J., who was one of the main authors of the initial versions of the K-means algorithm, also the most frequently cited, states.

The process, which is called "k-means", appears to give partitions which are reasonably efficient in the sense of within-class variance, corroborated to some extend by mathematical analysis and practical experience. Also, the k-means procedure is easily programmed and is computationally economical, so that it is feasible to process very large samples on a digital computer.

MacQeen also summarizes the benefits of K-means, while introducing his work as:

K-means algorithm is one of first which a data analyst will use to investigate a new data set because it is algorithmically simple, relatively robust and gives "good enough" answers over a wide variety of data sets.

2.4. SHORTCOMINGS IN K-MEANS

Research on the k-means algorithm, has been developed in two major directions, from theory and field of applications. From theory, a set of advantages and shortcomings has been identified.

2.4.1. Sensitivity to Initial Conditions

The k-means algorithm is an algorithm to cluster n objects based on attributes into k partitions, where k < n. It is similar to the expectation-maximization algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data. It assumes that the object attributes form a vector space. The objective it tries to achieve is to minimize total intra-cluster variance, or, the squared error function. The k-means algorithm is sensitive to the number of partitions and the initial centroids. There have been several methods implemented to take care of this shortcoming – the sub-merger algorithm, the particle swarm optimization algorithm, the weighted and inverse weighted k-means algorithm, the k-means algorithm using kd-trees etc.

2.4.2. Convergence of algorithm

The iterative procedure of the k-means algorithm does not guarantee convergence to global optimum values, but instead converges to local optimum values. K-means algorithm with a new convergence condition was implemented to tackle this shortcoming. Other approaches include Inverse exponential k-means algorithm, weighted and inverse weighted k-mean algorithm, estimation of lower limit for local optimum, the genetic k-means algorithm etc.

2.4.3. Efficiency of algorithm

The running time of the k-means algorithm is given as O(nkdi), where *n* is the number of *d*-dimensional vectors, *k* the number of clusters and *i* the number of iterations needed until convergence. On data that does have a clustering structure, the number of iterations until convergence is often small, and results only improve slightly after the first dozen iterations. In practice the k-means algorithm is therefore considered to be of linear complexity. Solutions to this shortcoming have been under constant scrutiny- the partial distance algorithm, the Rival penalizing competitive learning algorithm, the GAKREM algorithm, k-means implementations using core-sets or entropy weighting measures, unique distance measure approach etc.

2.4.4. Sensitivity to outliers and noise

The k-means algorithm is sensitive to outliers and noise. Even if an object is quite far away from the cluster centroid, it is still forced into a cluster and, thus, it distorts the cluster shapes. There were few extensions made to the k-means algorithm to resist the effect of outliers- the geodesic k-means clustering, a fast and robust general purpose clustering algorithm that eliminates the effect of outliers through a process that considers real points as centroids, ISODATA that performs dynamic estimation of K considering the effect of outliers in the process of clustering.

2.5. IMPROVED VERSIONS

As we read before in the paper there have been many methods applied to the original k mean algorithm to reduce the effect of its shortcomings on the final results. In this paper we are going to study four main algorithms which are most popularly used and fix most of the shortcomings.

2.5.1. The Genetic K-Means algorithm

The genetic algorithm is a randomized search and optimization technique guided by the principles of evolution and natural genetics, having a large amount of implicit parallelism. It performs search in complex, large and multimodal landscapes, and provide near-optimal solutions for objective or fitness function of an optimization problem. In the genetic algorithm, the parameters of the search space are encoded in the form of strings. A collection of such strings is called a population. Initially, a random population is created, which represents different points in the search space. An objective and fitness function is associated with each string that represents the degree of goodness of the string. Based on the principle of survival of the fittest, a few of the strings are selected and each is assigned a number of copies that go into the mating pool. The process of selection, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied.

The cluster obtained from a kernel k-means clustering is considered as input to our refinement algorithm. Initially a random point is selected from each cluster; with this a chromosome is build. Like this an initial population with 10 chromosomes is build. For each chromosome the entropy is calculated as fitness value and the global minimum is extracted. With this initial population, the genetic operators such as reproduction, crossover and mutation are applied to produce a new population. While applying crossover operator, the cluster points will get shuffled means that a point can move from one cluster to another. From this new population, the local minimum fitness value is calculated and compared with global minimum. If the local minimum is less than the global minimum then the global minimum is assigned with the local minimum, and the next iteration is continued with the new population. Otherwise, the next iteration is continued with the same old population. This process is repeated for N number of iterations.

2.5.2. K-means++ algorithm

The k-means++ is an algorithm for choosing the initial values called seeds for the k-means clustering algorithm. The k-means++ algorithm addresses the shortcomings of k-means by specifying a procedure to initialize the cluster centers before proceeding with the standard k-means optimization iterations. With the k-means++ initialization, the algorithm is guaranteed to find a solution that is O(log k) competitive to the optimal k-means solution. The intuition behind this approach is that spreading out the *k* initial cluster centers is a positive trait. The first cluster center is chosen uniformly at random from the data points that are being clustered, after which each subsequent cluster center is chosen from the remaining data points with probability proportional to its squared distance from the point's closest existing cluster center.

2.5.3. Mini Batch K-means algorithm

Mini Batch k-means is mostly useful in web applications where the amount of data can be huge, and the time available for clustering maybe limited. The complexity of the original K-Means clustering clustering algorithm is O(n*K*I*f), where n is the number of records, K is the number of clusters we want, I is the number of iterations and f is the number of features in a particular record. For a huge dataset it will take a lifetime for the k-means algorithm to cluster data. The idea of the mini batch k means algorithm is to represent the dataset by a smaller subset of the data. The algorithm makes sure that the model generalizes well over whole of the dataset and does this in far lesser time than the original algorithm. The algorithm takes randomly chosen small batches of the dataset for each iteration. It then assigns a cluster to each data point in the batch, depending on the previous locations of the cluster centroids. It then updates the locations of cluster centroids based on the new points from the batch. The update is a gradient descent update, which is significantly faster than a normal Batch K-Means update.

2.5.4. Geodesic K means algorithm

Typically, the classical k-means is used with Euclidean distance in which case centroids become component-wise mean of cluster points. This leads to a very low computational complexity and makes K-means a suitable candidate to try on very large datasets. However, K-means often fails when clusters are not linearly separable or when the data is cluttered with outliers. A central element in almost all clustering methods is the distance (dissimilarity) measure. The desirable distance measure should capture the connectivity of the points in each cluster, while separating different clusters in presence of outliers in between. A class of geodesic distances rely on a sparse neighborhood graph over the data points together with local density weights on its edges. The claim is that the shortest weighted paths on neighborhood graphs provide the desirable k means properties mentioned.

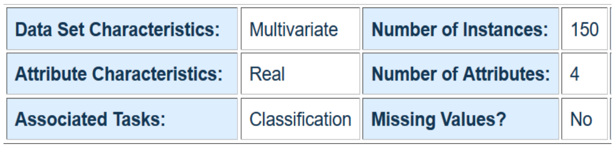
3. IMPLEMENTATION

3.1 Datasets

Formally the implementations of all the algorithms were performed on the following datasets:

3.1.1. Iris Dataset

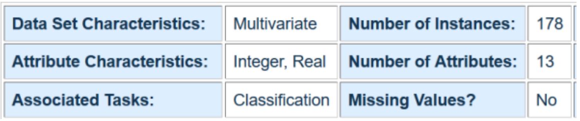
The first dataset that I have worked on is the famous Iris Dataset from the UCI Machine Learning Repository. The data set contains three classes of fifty instances each, where each class refers to a type of iris plant. One class is linearly separable from the other two, the latter are not linearly separable from each other.



Data Set Characteristics:	Multivariate	**Number of Instances:**	150
Attribute Characteristics:	Real	**Number of Attributes:**	4
Associated Tasks:	Classification	**Missing Values?**	No

3.1.2. Wine Dataset

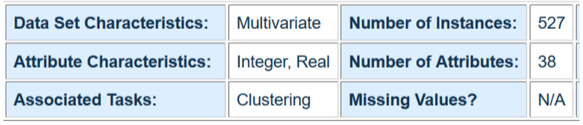
These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of thirteen constituents found in each of the three types of wines. All attributes are continuous.



Data Set Characteristics:	Multivariate	**Number of Instances:**	178
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	13
Associated Tasks:	Classification	**Missing Values?**	No

3.1.3. Water Treatment Plant Dataset

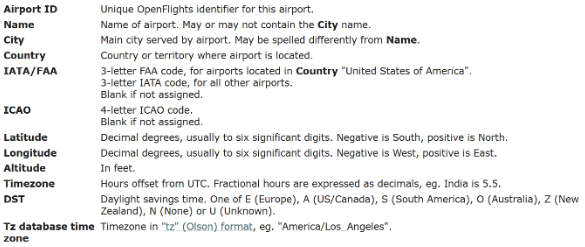
This dataset comes from the daily measures of sensors in an urban waste water treatment plant. The objective is to classify the operational state of the plant in order to predict faults through the state variables of the plant at each of the stages of the treatment process. This domain has been stated as an ill-structured domain. All attributes are numeric and continuous.



Data Set Characteristics:	Multivariate	**Number of Instances:**	527
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	38
Associated Tasks:	Clustering	**Missing Values?**	N/A

3.1.4. Flight Dataset

This dataset contains information about different airports around the world. The data was collected in January 2012. The Open Flights Airports Database contains 6977 airports spanning the globe, as shown in the map above. The data is ISO 8859-1 encoded, with no special characters. Each entry contains the following information:



Airport ID	Unique OpenFlights identifier for this airport.
Name	Name of airport. May or may not contain the **City** name.
City	Main city served by airport. May be spelled differently from **Name**.
Country	Country or territory where airport is located.
IATA/FAA	3-letter FAA code, for airports located in **Country** "United States of America". 3-letter IATA code, for all other airports. Blank if not assigned.
ICAO	4-letter ICAO code. Blank if not assigned.
Latitude	Decimal degrees, usually to six significant digits. Negative is South, positive is North.
Longitude	Decimal degrees, usually to six significant digits. Negative is West, positive is East.
Altitude	In feet.
Timezone	Hours offset from UTC. Fractional hours are expressed as decimals, eg. India is 5.5.
DST	Daylight savings time. One of E (Europe), A (US/Canada), S (South America), O (Australia), Z (New Zealand), N (None) or U (Unknown).
Tz database time zone	Timezone in "tz" (Olson) format, eg. "America/Los_Angeles".

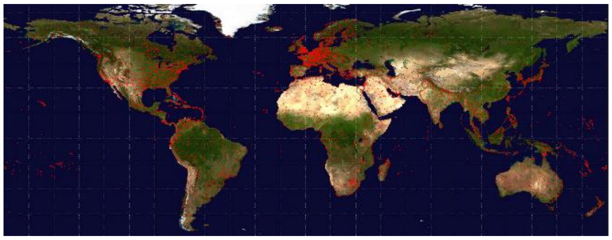


FIGURE 2: This map shows all the airports all over the world

3.2. ALGORITHMS:

Due to time-constraint I was able to implement only two out of the four algorithms. But during research I studied about all four and found them really interesting so I think giving an overview of all the four algorithms is going to be useful.

3.2.1. K Means

Let us start with the classical K means algorithm.

Iterate until stable (= no object moves in group):

1. Determine the centroid coordinate
2. Determine the distance of each object to the centroids
3. Group the object based on minimum distance (find the closest centroid)

3.2.2. K means ++

The K means ++ is a simple improvement to the classical k means algorithm. It mainly solves the shortcoming of sensitivity to initial conditions.

1. Choose one center uniformly at random from among the data points.
2. For each data point x, compute D(x), the distance between x and the nearest center that has already been chosen.
3. Choose one new data point at random as a new center, using a weighted probability distribution where a point x is chosen with probability proportional to D(x)^2.
4. Repeat Steps 2 and 3 until k centers have been chosen.
5. Now that the initial centers have been chosen, proceed using standard k-means clustering.

3.2.3. Genetic k means

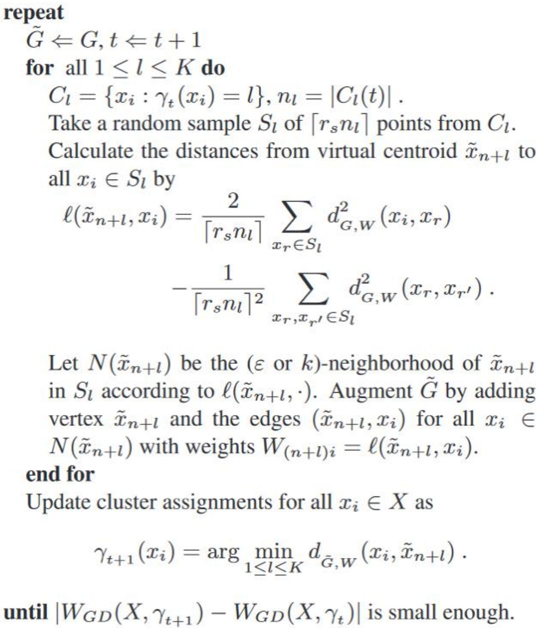
One major problem with the classical k-means is that it can only be applied to numerical attributes. The genetic K-Means is an improved version of the K-Means that is used to map categorical attributes to be able to use k means clustering on them.

```
Begin
1. Multiple sub-samples {S1, S2, ...,Sj};
2. For m = 1to j do K-means (Sm, K’); //executing K-means, produce K' clusters and j groups.
3. Compute $J_c(m) = \sum_{j=1}^{k} \sum_{X_i \in C_j} |X_i - Z_j|^2$;
4. Choose min{Jc} as the refined initial points Zj, j ∈ [1, K'];
5. K-means (S, K'); //executing K-means again with chosen initial, producing K' mediods.
6. Repeat
Combining two near clusters into one cluster, and recalculate the new center generated by two centers merged.
7. Until the number of clusters reduces into k //Merging (K'+K)
    End
```

3.2.4. Geodesic K Means

The geodesic K–means cluster algorithm is used to take care of the sensitivity of the classical k-means algorithm to outliers and noise.

Require: Set of n points X, number of clusters K, neighborhood parameter ε or k, sampling rate rs∊(0,1], an initial clustering assignment γ0 with no empty clusters, t=-1.
Construct the (∊or k)-neighborhood graph G over X. Calculate the matrix of weights W according to general distance k-means.



```
repeat
  $\tilde{G} \Leftarrow G, t \Leftarrow t+1$
  for all $1 \leq l \leq K$ do
    $C_l = \{x_i : \gamma_t(x_i) = l\}, n_l = |C_l(t)|$ .
    Take a random sample $S_l$ of $\lceil r_s n_l \rceil$ points from $C_l$.
    Calculate the distances from virtual centroid $\tilde{x}_{n+l}$ to all $x_i \in S_l$ by
      $\ell(\tilde{x}_{n+l}, x_i) = \frac{2}{\lceil r_s n_l \rceil} \sum_{x_r \in S_l} d^2_{G,W}(x_i, x_r) - \frac{1}{\lceil r_s n_l \rceil^2} \sum_{x_r, x_{r'} \in S_l} d^2_{G,W}(x_r, x_{r'})$ .
    Let $N(\tilde{x}_{n+l})$ be the ($\varepsilon$ or $k$)-neighborhood of $\tilde{x}_{n+l}$ in $S_l$ according to $\ell(\tilde{x}_{n+l}, \cdot)$. Augment $\tilde{G}$ by adding vertex $\tilde{x}_{n+l}$ and the edges $(\tilde{x}_{n+l}, x_i)$ for all $x_i \in N(\tilde{x}_{n+l})$ with weights $W_{(n+l)i} = \ell(\tilde{x}_{n+l}, x_i)$.
  end for
  Update cluster assignments for all $x_i \in X$ as
    $\gamma_{t+1}(x_i) = \arg\min_{1 \leq l \leq K} d_{\tilde{G},W}(x_i, \tilde{x}_{n+l})$ .
until $|W_{GD}(X, \gamma_{t+1}) - W_{GD}(X, \gamma_t)|$ is small enough.
```

3.3. DATA PREPROCESSING:

Data preprocessing is required to find missing values. Incase missing values are found there are many different ways to take care of them- ignore the missing values, replace the missing values with the mean or median of data, replace the missing values with most probable value etc. We can also reduce number of attributes to make data mining easier in the data preprocessing step.

The datasets that I have picked have no missing data, so we are not concerned with that step. The only form of data preprocessing that has been applied to every dataset is the data normalization. Additionally, for the flight dataset, I have grouped together data based on country, daylight savings time, and time-zone.

3.4. EXPLORATORY DATA ANALYSIS

The exploratory data analysis graphs have been put up in the Appendix Section.

Iris Dataset: The graphs show the boxplots containing the minimum, maximum, mean, standard deviation values for the attributes both before and after normalization. I have used StandardScalar() function to normalize the values.

Wine Dataset: The graphs show the boxplots containing the minimum, maximum, mean, standard deviation values for the attributes both before and after normalization. I have used StandardScalar() function to normalize the values.

Water Treatment Plant Dataset: The graphs show the boxplots containing the minimum, maximum, mean, standard deviation values for the attributes both before and after normalization. I have used StandardScalar() function to normalize the values.

Flight Dataset: The graphs show the boxplots containing the minimum, maximum, mean, standard deviation values for the attributes both before and after normalization. I have used StandardScalar() function to normalize the values. For this dataset I have also shown visualizations after grouping the attributes in terms of Country Name, Day light savings and Time-zone.

4. RESULTS

4.1 ELBOW CURVES

This method looks at the percentage of variance explained as a function of the number of clusters. When one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information, but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the "elbow criterion". This "elbow" cannot always be unambiguously identified. Percentage of variance explained is the ratio of the between-group variance to the total variance, also known as an F-test. A slight variation of this method plots the curvature of the within group variance.

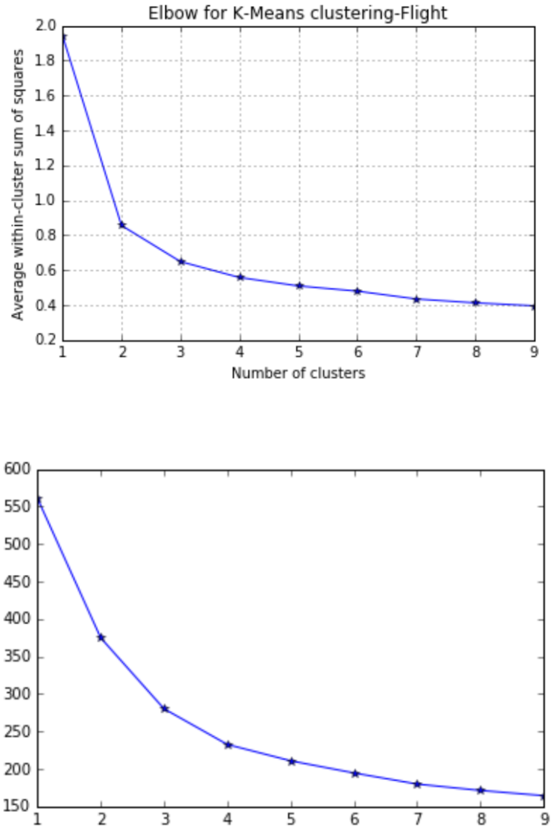


FIGURE 3: Elbow curves of the iris and flight dataset

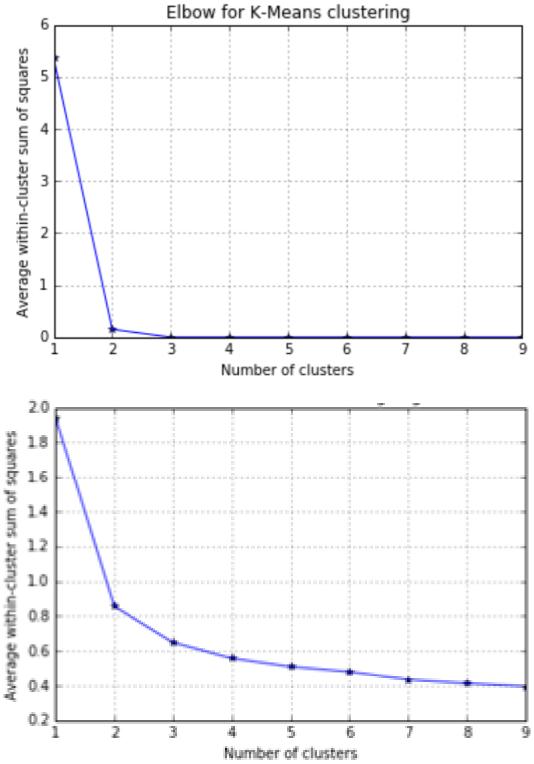
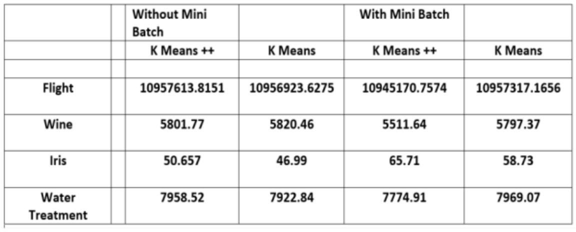


FIGURE 4: Elbow curves for wine and water treatment plant dataset.

4.2 MEAN SQUARED ERROR

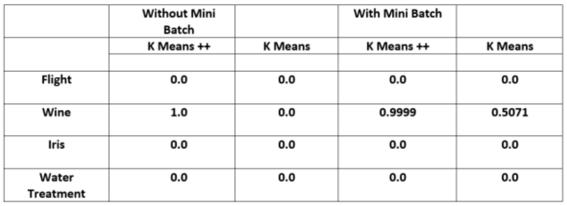
MSE of an estimator measures the average of the squares of the errors or deviations, that is, the difference between the estimator and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss or quadratic loss. The difference occurs because of randomness or because the estimator doesn't account for information that could produce a more accurate estimate.



		Without Mini Batch		With Mini Batch	
		K Means ++	K Means	K Means ++	K Means
Flight		10957613.8151	10956923.6275	10945170.7574	10957317.1656
Wine		5801.77	5820.46	5511.64	5797.37
Iris		50.657	46.99	65.71	58.73
Water Treatment		7958.52	7922.84	7774.91	7969.07

4.3 ADJUSTED RAND INDEX

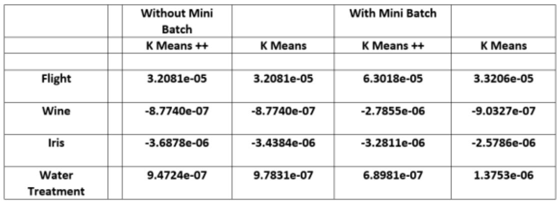
Given the knowledge of the ground truth class assignments labels_true and our clustering algorithm assignments of the same samples labels_pred, the adjusted Rand index is a function that measures the similarity of the two assignments, ignoring permutations and with chance normalization. One can permute 0 and 1 in the predicted labels, rename 2 to 3, and get the same score. Adjusted rand score is symmetric: swapping the argument does not change the score. It can thus be used as a consensus measure. Perfect labeling is scored 1.0.



	Without Mini Batch		With Mini Batch	
	K Means ++	K Means	K Means ++	K Means
Flight	0.0	0.0	0.0	0.0
Wine	1.0	0.0	0.9999	0.5071
Iris	0.0	0.0	0.0	0.0
Water Treatment	0.0	0.0	0.0	0.0

4.4 MUTUAL INFORMATION BASED SCORES

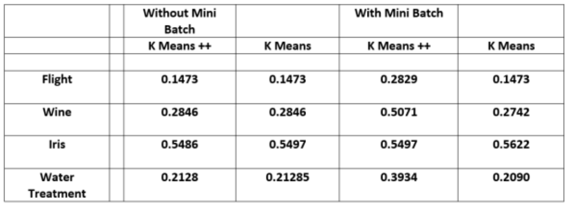
Given the knowledge of the ground truth class assignments labels_true and our clustering algorithm assignments of the same samples labels_pred, the Mutual Information is a function that measures the agreement of the two assignments, ignoring permutations. Two different normalized versions of this measure are available, Normalized Mutual Information(NMI) and Adjusted Mutual Information(AMI**)**. NMI is often used in the literature while AMI was proposed more recently and is normalized against chance.



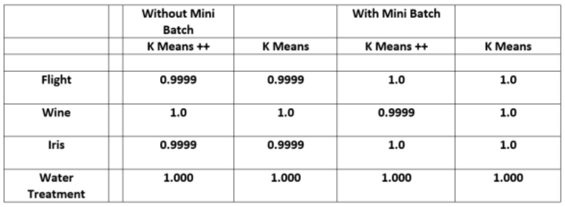
	Without Mini Batch		With Mini Batch	
	K Means ++	K Means	K Means ++	K Means
Flight	3.2081e-05	3.2081e-05	6.3018e-05	3.3206e-05
Wine	-8.7740e-07	-8.7740e-07	-2.7855e-06	-9.0327e-07
Iris	-3.6878e-06	-3.4384e-06	-3.2811e-06	-2.5786e-06
Water Treatment	9.4724e-07	9.7831e-07	6.8981e-07	1.3753e-06

4.5 HOMOGEINEITY, COMPLETENESS AND V- MEASURE

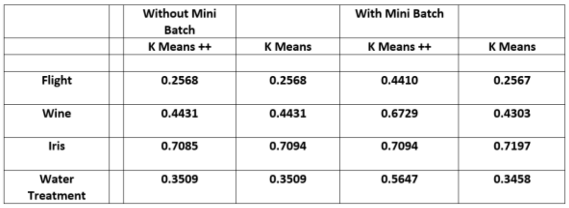
The homogeneity is calculated to see if each cluster contains only members of a single class. The completeness: all members of a given class are assigned to the same cluster. Both are bounded below by 0.0 and above by 1.0 (higher is better). Their harmonic mean called V-measure is computed by v_measure_score. The V-measure is actually equivalent to the mutual information (NMI) discussed above normalized by the sum of the label entropies. The tables below show homogeneity, completeness and v-measure in the above order.



	Without Mini Batch		With Mini Batch	
	K Means ++	K Means	K Means ++	K Means
Flight	0.1473	0.1473	0.2829	0.1473
Wine	0.2846	0.2846	0.5071	0.2742
Iris	0.5486	0.5497	0.5497	0.5622
Water Treatment	0.2128	0.21285	0.3934	0.2090



	Without Mini Batch		With Mini Batch	
	K Means ++	K Means	K Means ++	K Means
Flight	0.9999	0.9999	1.0	1.0
Wine	1.0	1.0	0.9999	1.0
Iris	0.9999	0.9999	1.0	1.0
Water Treatment	1.000	1.000	1.000	1.000



	Without Mini Batch		With Mini Batch	
	K Means ++	K Means	K Means ++	K Means
Flight	0.2568	0.2568	0.4410	0.2567
Wine	0.4431	0.4431	0.6729	0.4303
Iris	0.7085	0.7094	0.7094	0.7197
Water Treatment	0.3509	0.3509	0.5647	0.3458

5. CONCLUSION AND CHALLENGES

This project was a learning experience for me. I learnt several new aspects of the K-means Clustering algorithm and how to deal with its shortcomings. I faced quite a few challenges. I was unable to implement all my algorithms due to lack of time. Also I am not so fluent with python so it took me a long time to get the syntax of the codes correct. Also I had a tough time running the Flight dataset and couldn't generate all graphs for it as it took a long time to process.

6. FUTURE WORK

There are two main areas of future work possible to focus on. One is extending the applications of K means algorithm for categorical data: implementation of the Genetic K means for text clustering using ontology to evaluate the validity of various semantic similarity measures. Second is implementation of the extended K means algorithm using the geodesic distance metric to ensure resistance to outliers.

6 REFERENCES

[1] Geodesic K-means Clustering, By Nima Asgharbeygi, Arian Maleki

[2] Research issues on K-means Algorithm: An Experimental Trial Using Matlab, By Joaquín Pérez Ortega, Ma. Del Rocío Boone Rojas, María J. Somodevilla García

[3] K-means Clustering Algorithm with improved initial Center, By Zhang, Chen; Xia Shixiong.

[4] Local vs global interactions in clustering algorithms: Advances over K-means, by Barbakh Wesam And Colin Fyfe

[5] Analysis of Global k-means, an Incremental Heuristic for Minimum Sum of Squares Clustering, by P. Hansen & E.Nagai.

[6] Selection of K in K-means clustering, by Pham, D. Dimov,S.S. Nguyen, C.D

[7] The Global K-means Clustering Algorithm, by Likas, A., Vlassis, N., Verbeek, J.J.

[8] An empirical comparision of four initialization methods for the k-means algorithm, by J. Peña, J. Lozano and P. Larrañaga

[9] Local vs global interactions in clustering algorithms: Advances over K-means, by Barbakh Wesam And Colin Fyfe.

[10] Improvement the Efficiency and Efficacy of the K-means Clustering Algorithm through a New Convergence Condition, by Joaquín .Pérez O, Rodolfo Pazos R, Laura Cruz R.,Gerardo Reyes S. Rosy Basave T. Héctor and Fraire H.

[11] On the Lower Bound of Local Optimums in K-means Algorithm, by Z. Zhang, B. Tian D. And Tung A.K.H.

[12] Genetic K-means algorithm, by K. Krishna and M.Murty

[13] New Efficient Strategy to Accelerate k-Means Clustering Algorithm, by Moh`d Belal Al Zoubi, Amjad Hudaib, Ammar Huneiti and Bassam Hammo

[14] An Efficient k`-means Clustering Algorithm. Based on the algorithm Rival, it penalizes competitiveLearninig (RPCL), by Zalik, Krista Rizman

[15] GAKREM: A novel hybrid clustering algorithm, by Cao. D. Nguyen & Cios, Krzysztof J.

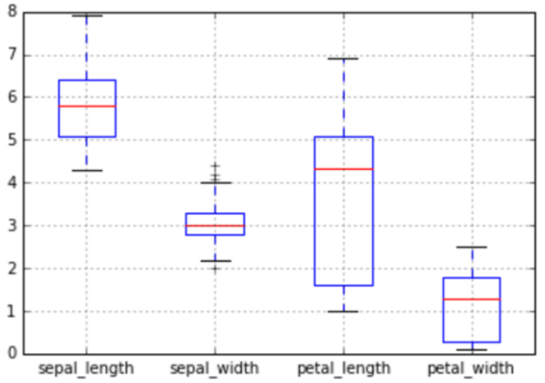
[16] “Improvement of the K-means algorithm using a new approach of convergence and its application to databases cancer population, by J.Pérez, M.F. Henriques, R. Pazos, L. Cruz, G. Reyes, J. Salinas, A. Mexicano

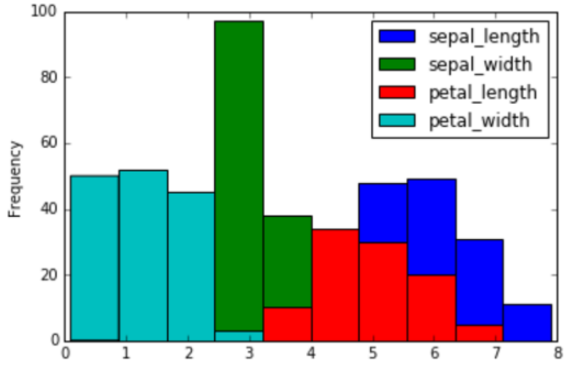
[17] Geodesic K-means clustering, by Asgharbeygi, N.
Maleki, A.

APPENDIX

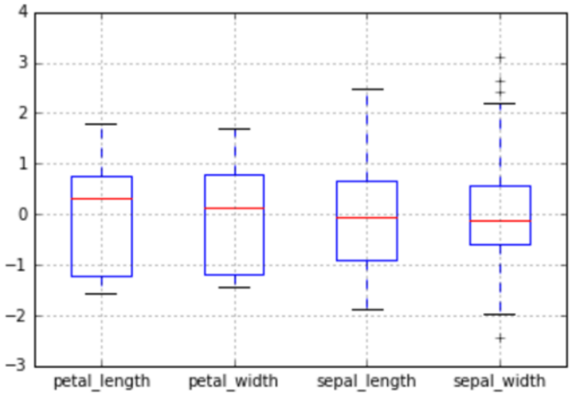
IRIS DATASET:

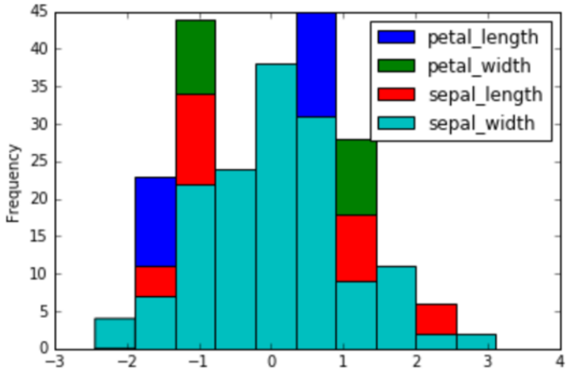
These are the graphs before normalization





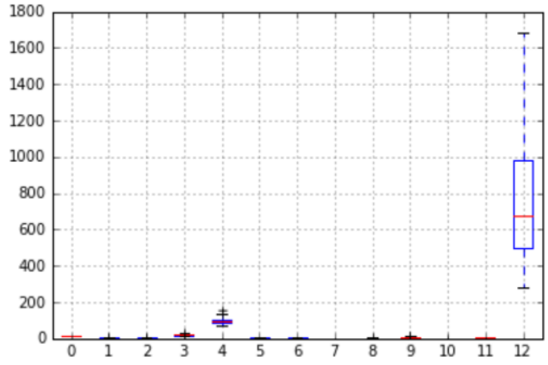
These are the graphs after normalization

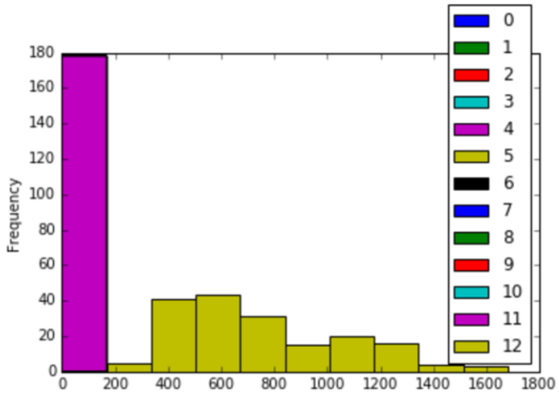




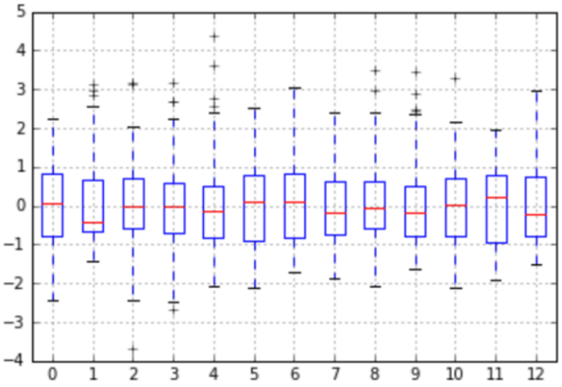
WINE DATASET:

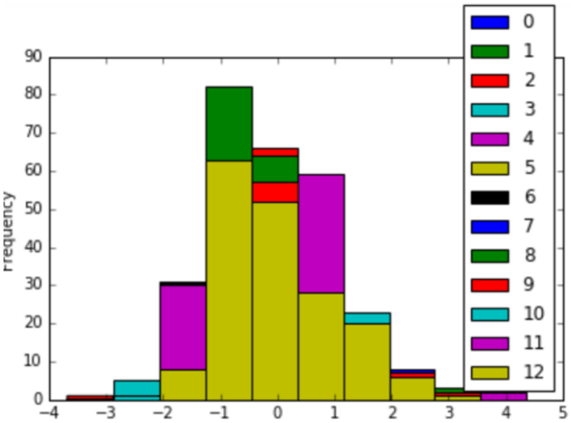
These are the graphs before data normalization



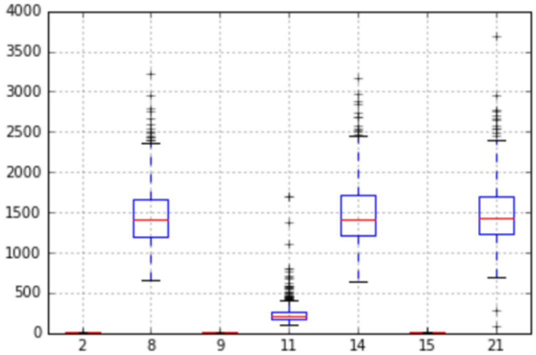


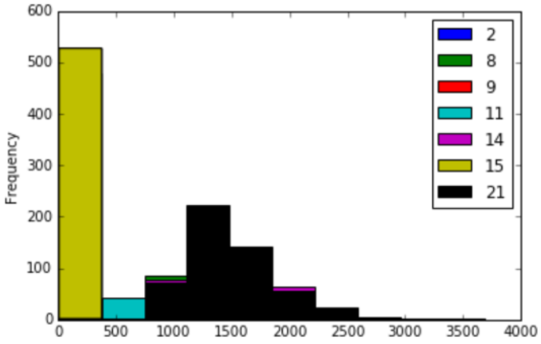
These are the graphs after data normalization



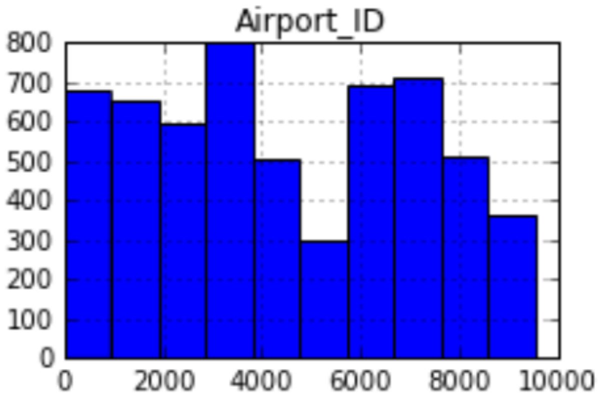


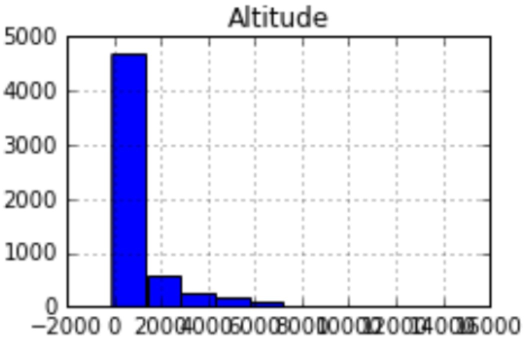
WATER TREATMENT PLANT DATASET:

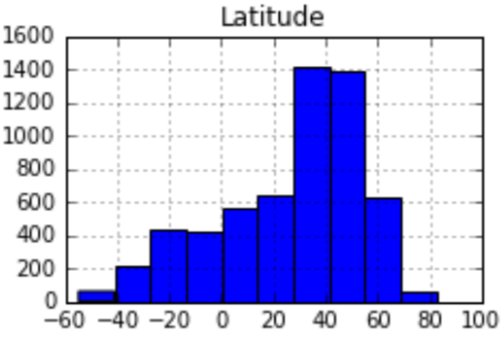


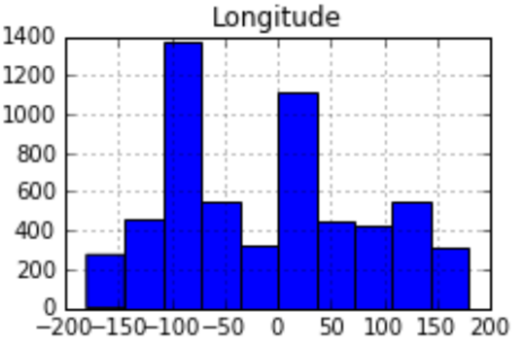


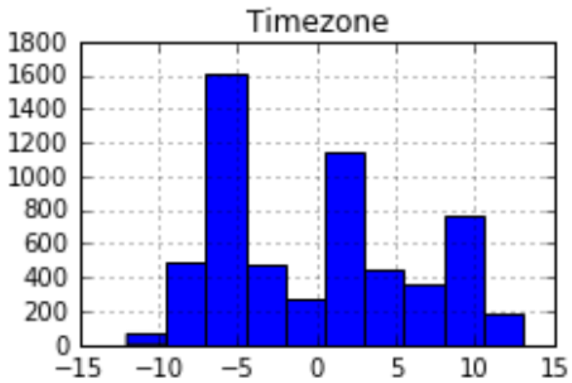
FLIGHT DATASET:

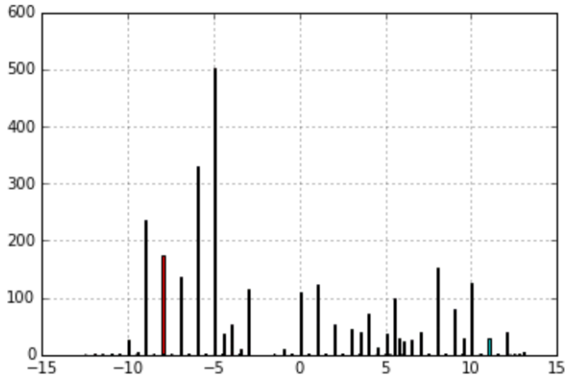


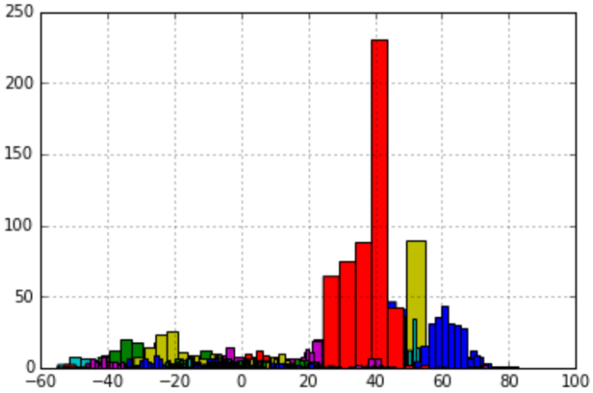


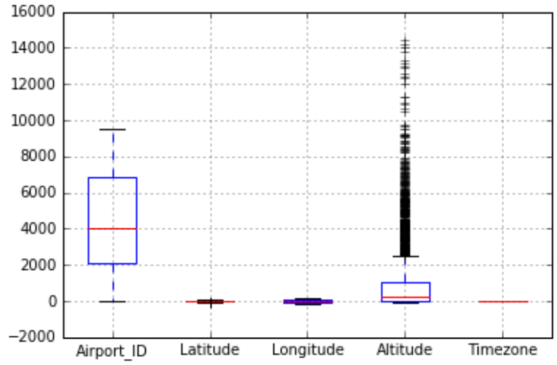


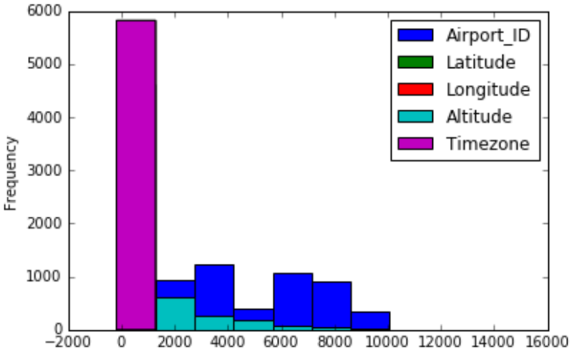


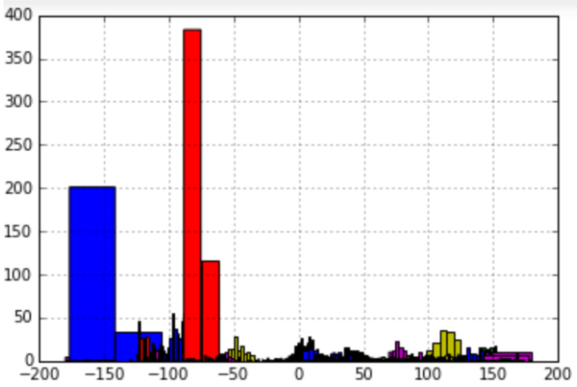

Timezone
1800
1600
1400
1200
1000
800
600
400
200
0
−15
−10
−5
0
5
10
15

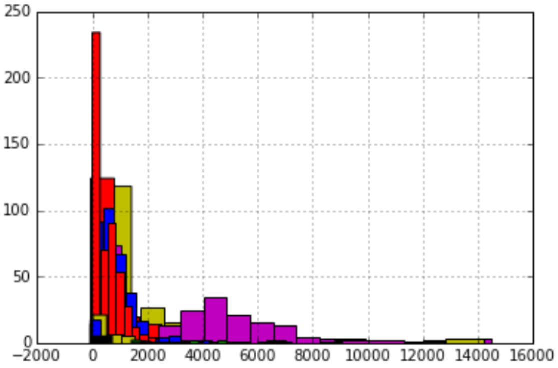

600
500
400
300
200
100
0
−15
−10
−5
0
5
10
15


250
200
150
100
50
0
−60
−40
−20
0
20
40
60
80
100


16000
14000
12000
10000
8000
6000
4000
2000
0
−2000
Airport_ID
Latitude
Longitude
Altitude
Timezone


6000
5000
4000
3000
2000
1000
0
Frequency
−2000
0
2000
4000
6000
8000
10000
12000
14000
16000
Airport_ID
Latitude
Longitude
Altitude
Timezone


400
350
300
250
200
150
100
50
0
−200
−150
−100
−50
0
50
100
150
200


250
200
150
100
50
0
−2000
0
2000
4000
6000
8000
10000
12000
14000
16000