# Supervised Unsupervised Learning in Spark

MSc Reseach Project
Data Analytics

Vidya Sankar Velamuri
x15009653

School of Computing
National College of Ireland

Supervisor: Dr. Simon Caton

**National College of Ireland**
**Project Submission Sheet – 2015/2016**
**School of Computing**

| | |
|---|---|
| **Student Name:** | Vidya Sankar Velamuri |
| **Student ID:** | x15009653 |
| **Programme:** | Data Analytics |
| **Year:** | 2016 |
| **Module:** | MSc Reseach Project |
| **Lecturer:** | Dr. Simon Caton |
| **Submission Due Date:** | 22/08/2016 |
| **Project Title:** | Supervised Unsupervised Learning in Spark |
| **Word Count:** | XXX |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

**ALL** internet material must be referenced in the bibliography section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

| | |
|---|---|
| **Signature:** | |
| **Date:** | 12th September 2016 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS:**

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
3. Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Supervised Unsupervised Learning in Spark

Vidya Sankar Velamuri
x15009653
MSc Reseach Project in Data Analytics

12th September 2016

**Abstract**

Clustering is the unsupervised classification of patterns into groups and is one of the most popular techniques applied to explore and discover naturally occurring patterns within hitherto unlabelled data. The quality of the clusters resulting from a clustering algorithm can be verified using clustering validity indices, which take into account the intra cluster similarity and inter cluster separation of the clusters. However in a distributed setting the computation of pairwise distances between data points of a large data set distributed across the cluster can be computationally very expensive.

This research proposes to evaluate a sampling based approach to computing the cluster validity indices for distributed datasets and embed this methodology into a model selection pipeline that evaluates distributed machine learning jobs in selecting an optimal clustering algorithm. The results suggest the sampling error of the internal validation index so computed is statistically significant.

# Contents

# 1 Introduction

There are different classifications of learning algorithms like supervised, unsupervised, reinforcement learning problems that can be applied to analysis of data. Enterprises face the challenge of selecting a relevant and optimal model for exploratory or predictive analysis of their data.

In supervised learning, the observations to be classified are provided with a label. As the volume and nature of data being generated increases obtaining data with labels is increasingly challenging (Aggarwal and Reddy; 2013). This makes automatic labeling an indispensable step in the modern data analysis pipeline and is relevant to enterprises for understanding and categorising large volumes of unlabelled data being gathered.

"Clustering is the unsupervised classification of patterns into groups" (Jain et al.; 1999) and is an appropriate choice for performing exploratory data analysis of large data sets to identify hitherto unknown inter relationships and patterns within the data. Image classifcation, trend and anamoly detection, customer segmentation are some of the well known application domains in which clustering is used.

# 2 Taxonomy of Clustering Algorithms

Clustering algorithms can be classified into a Hierarchical, Partitional, Density Based, Grid Based algorithms based on their choice of measure for arrving at the partitions of data. Partitional algorithms require mentioning the desired number of clusters upfront, where as density based algorithms require to be told the number of points in space that make up a cluster. Hierarchical algorithms take either a top down approach or bottom up approach aggregating or partitioning data points into clusters (Jain et al.; 1999).

Clustering algorithms are based on the notion of similarity between points and try to maximizie similarity within the group, while minimizing inter-group similarity. The problem of defining similarity is largely dependent on the definition of a desirable cluster and a relevant distance measure, the selection of which requires knowledge of the domain from which the data is being generated. Hence clustering algorithms can only be evaluated against the particular dataset for the measure of the quality of the resulting clusters rather than as a generalised model.

Given the wide taxonomy of clustering algorithms, Lange et al. (2004) define a general model selection process for selecting an optimal clusteirng algorithm for a given dataset as follows :

1. Initialise a list of algorithms

2. On each chosen algorithm perform a parameter sweep to obtain different clusters

3. Compute the cluster validation measures to evaluate the results

However, with no version of ground truth available for the data, the problem of selection and evaluation of a clustering algorithm in an automated manner is a difficult exercise.

Data mining algorithms have traditionally been applied to data sets that could be fit into the memory of a single machine. However the scale of modern enterprise data necessitates scaling out to analyse these larger and diverse data sets. This poses technical challenges like scaling the algorithms to process data on a distributed scale, time and space complexlity, data locality and fault tolerance.

Modern distributed data processing frameworks have abstracted away these complexities and have made it possible for enterprises to build automated data processing pipelines on large distributed datasets. The Apache Hadoop framework based on the functional programming paradigm of map and reduce (Dean and Ghemawat; 2008) can analyse massive amounts of data on large clusters built from afforable and easily replaceable commodity hardware. The ready availability of machine learning algorithms capable of running on distributed data sets, has enabled the industrialisation of large scale data analysis. However it is a significant challenge to build a pipeline atop a framework that runs on shared distributed resources, that is aware of resource-constraints and uses them in an efficient manner. Newer data processing engines like Spark, Tez take advantage of pluggable distributed cluster managers like Apache Mesos and YARN have that efficiently schedule and monitor resource utilisation on shared clusters (Vavilapalli et al.; 2013) and are the building blocks for modern enterprise scale big data analytics platforms.

TN (2015) suggets a novel approach to increasing the efficiency of supervised learning algorithms based on dynamic model evaluation approach. This approach monitors at run time the incremental learning progress of a machine learning algorithm and uses it decide to stop a machine learning job thats has poor incremental learning performance. This information is further embedded into the communication protocol with the cluster resource manager which can use it to request its application master to terminate the job, thus freeing the shared resources.

This research proposes to the explore the feasbility of extending this dynamic model evaluation approach, successfully appplied to a supervised learning algorithm, to an unupservised learning algorithm. However, unlike supervised learning which has precise and objective measures of performance evaluation, an unsupervised learning algorithm can only be evaluated based on relative metrics across different parametric combinations. To illustrate, for selecting and evaluating the optimum number of clusters for a given data set using a crisp clustering algorithm (K-Means), the spark framework on which the aforesaid dynamic model evaluation approach was implemented, only provides a single measure of performance evaluation, the Weighted Sum of Squared Errors (WSSE). As seen in 1, WSSE is a monotonic measure that decreases as the cluster size increases. The optimcal cluster in this scenario can be selected based on the elbow criteria which is a heuristic measure, which makes it harder to evaluate the incremental learning rate objectively in a dynamic model evaluation scenario.

This has lead to the review of other clustering validation measures. Unsupervised learning relies on measures of pairwise distance computation in both arriving at the clusters and in evaluating the quality of the resulting clusters. However in a distributed scenario it becomes highly inefficient to compute the pairwise distances between points spread across multiple nodes for the process of evaluating the results as it has a greater time and space complexity than deriving the clustering model itself.

Hence the research question “Can the computational efficiency of cluster evaluation be improved in a distributed setting and be integrated into a unsupervised modeling pipeline to dynamically evaluate simultaneous distributed machine learning jobs launched with different parameters in choosing an appropriate clustering algorithm”.

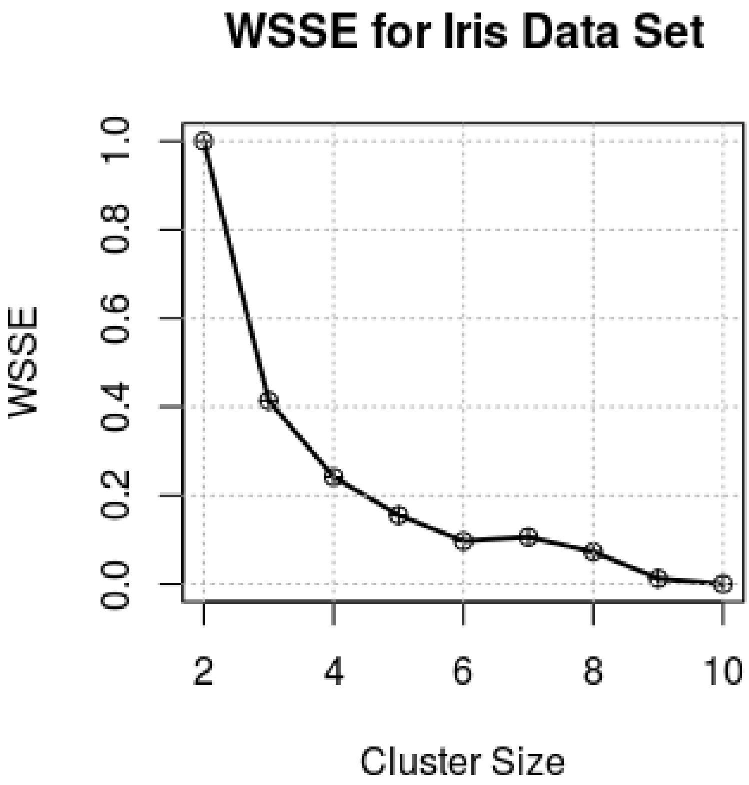

Figure 1: WSSE for KDD 1999 Data Set

# 3 Related Work

## 3.1 Cluster Validation

Since clustering is an unsupervised learning mechanism, there is no apriori knowledge of a ground truth against which the results can be measured against (Von Luxburg and Ben-David; 2005). Hence validation of results of clustering results is an important step in ensuring the quality of the resulting clusters.

The wide variety of clustering algorithms means no single measure can be uniformly and objectively assessed towards the validity of the results for all clustering methods. i.e. some measures may favour a particular algorithm. Density based evaluation measures might align better with clustering algorithms like DBScan, where as an evaluation based on metrics like cluster radius might prefer k-Means algorithm, which is partitional algorithm that is more suited to identifying spherical clusters. Inspite of this limitation, the resulting clusters can be evaluated using various intra cluster metrics like size, radius, density (Leskovec et al.; 2014) and their stability (Von Luxburg; 2010) and cluster validation indices that are based on combination of intra-cluster and inter-cluster measures (Halkidi et al.; 2001). of clustering.

It is also common to employ a variety of clustering techniques to analyse a data set and use visual inspection aided by the prior knowledge of the domain the analyse the results of the clustering algorithm (Handl et al.; 2005). However for large data sets and with high dimensions a manual visual inspection of clusters is not feasible.

In their review of cluster validity measures Halkidi et al. (2001) classify validity criteria for these various clustering mechanisms into two major categories internal validation, external validation measures. External clustering validation rely on prior knowledge of the true clusters and on the existence of this external truth against which the results of clustering are measured. Some example of external validation measures are purity, entropy, precision, recall, f-measure. Since the challenges with large data sets being accumulated is the task of assigning their true labels (Aggarwal and Reddy; 2013), for the purposes of this research, external validation measures have not been further considered.

### 3.1.1 Internal Cluster Validation Indices

Internal Cluster Validity Indices primarily rely on the measures of compactness and seperation. Whereas compactness relates to intra cluster quality , separation relates to inter cluster quality. Liu et al. (2013) emphasise the greater importance of inter cluster separation on the cluster validity.

The measure of compactness can be based on computing either the maximum or average of pairwise distance between points in the cluster or the distance to the cluster centroid where as separation can be computed as the maximum or average of the pair wise distance between points across clusters or distance between their centroids (Aggarwal and Reddy; 2013) and varies according to the chosen validity index measure.

A few of the internal cluster validation indices consider only one of the above criteria. However these primary measures tend to be monotonic and usually only consider one aspect of the principal cluster validation measures of compactness and separation (Liu et al.; 2010)

Validity indices that take in account both measures tend to perform better and handle issues like noise, outliters and skewness of the data better than unary measures of indices that consider only one of the two features of cluster validation indices.

There are a number internal clustering validation indices like the DunnIndex, Davies-BouldenIndex, SDIndex. Each of these indices are based on a variant of the combination of separation and compactness measures of the resultant clusters. Irrespective of the chosen index most of them rely on computing pair wise distances between points across clusters. In a distributed environment this can a very inefficient as it has a greater time and space complexity than performing the clustering itself.

# 4 Methodology

The problem of implementing a distributed cluster validation index is that evaluation can be more expensive than computing the clusters. In its selection of centroids, the Bahmani et al. (2012) algorithm broadcasts the chosen centers to all partitions to compute the nearest centroid locally on each partition. However for the purpose of computing pairwise distances between clusters, a similar approach of relaying of those data points betweeen nodes can lead to a number of expensive network shuffles and is not feasible.

Based on the understanding of existing literature, the principles of distributed computing and statistical inference this research proposes a sampling based approach for the computation of cluster validity indices in computing the validation indices used to determine optimal cluster configuration. To support the methodology of this research, the Davious Boulden Index which compares within cluster scattering to between-cluster separation (Davies and Bouldin; 1979) is proposed to be implemented in the chosen distributed framework.

In sampling for the data points to compute the cluster validity index the following approach is followed. Random sampling, without replacement is done. Samples are drawn at the rates from 10% to 100% of the population and the Davies Bouldin Index is computed locally, on the spark driver node, on the sample so gathered. To eliminate any bias in this process the sampling process is repeated 10 times for each sampling percentage and the average value of the index is computed to arrive at the final index value for that sample percentage. Further a test of statistical signifinance is performed on the sample based Davies Bouldin Index against the index computed on the population to verify the efficacy of this sampling approach.

# 5 Implementation

The choice of framework for this project is the Apache Spark framework. It is based on the Scala language which has in built support for functional programming with its support for map reduce style abstractions and allows parallelising processing of collections and lazy evaluation that can be leveraged for efficient data processing. The Spark framework, developed in scala building on top of its strengths, provides the Resilient Distributed Datasets(RDD) abstraction for programmming on distributed data sets. Unlike the hadoop framework where the intermediate results through the iterations have to be written to disk, the RDD's allow intermediate results to persisted in memory or disk and hence speed up the iterative machine learning process (Zaharia et al.; 2010). However the RDD abstraction does require significantly higher memory allocation as it retains the data lineage of the RDDs to avoid expensive transformations on the parent data set.

Spark provides the MlLib machine learning framework in which the K-means clustering is implemented. It can be run on distributed data sets, and provides a cost metric based on the within cluster sum of squared errors (WCSS) to evaluate its performance.

Performing automated model selection using techniques like paramater sweep and cross validation on a distributed scale is a highly resource intensive process. For each value or combination of the parameters an individual machine learning job can be triggered or alternatively the same job can made to perform the entire parameter sweep. However splitting this up into individual jobs brings with a few benefits like the ability to run those jobs in parallel and monitor each job for its quality metrics and explore the possibility of terminating early the jobs that continually exhibit poor quality metrics.

As there is no centralised job manager available on the Spark Framework currently, each distributed job is submitted individually to the yarn cluster with a particular combination of parameters like the number of clusters, the number of iterations and the sampling rate to be used for gathering sample data points for clustering evaluation and the DBIndex and WSSE are computed on the chosen data sets.

# 6 Evaluation

## 6.1 Experiment Setup

Table 1 provides information on the proposed hardware configuration used for this experiment which primarily consists of a Spark ecosystem setup on top of the YARN cluster.

Table 1: Hardware Setup

| *Component* | *MasterNode* | *DataNode* |
|---|---|---|
| VCPU | 4 | 2 |
| RAM | 8GB | 4GB |
| Hard Disk | 80GB | 40GB |
| Quantity | 1 | 4 |

This implementation proposes an experimental methodology be tested on a scientific data set and two real world-data sets to evaluate their performance.

Since the K-Means implmenetation on the Spark framework only supports the euclidean distance measure, datasets have been chosen for which the euclidean distance is a meaningful measure of distance.

## 6.2 Experiment / Iris Data Set

The Iris data set has been chosen as one of the datasets to evaluate this implementation. The DBIndex for the Iris Data Set 2 shows that quality of the clusters suffers as the number of the cluster increases and this is in contrast to the WSSE that is monotonically decreasing 1. However the DBIndex is not accurate as it shows the Iris data set has the least value at 2 clusters as against the ground truth available for this data set of 3 clusters. This highlights the problem of selection and evaluation of an appropriate clustering algorithm, owing to the lack of a standardised and objective measures of validation.

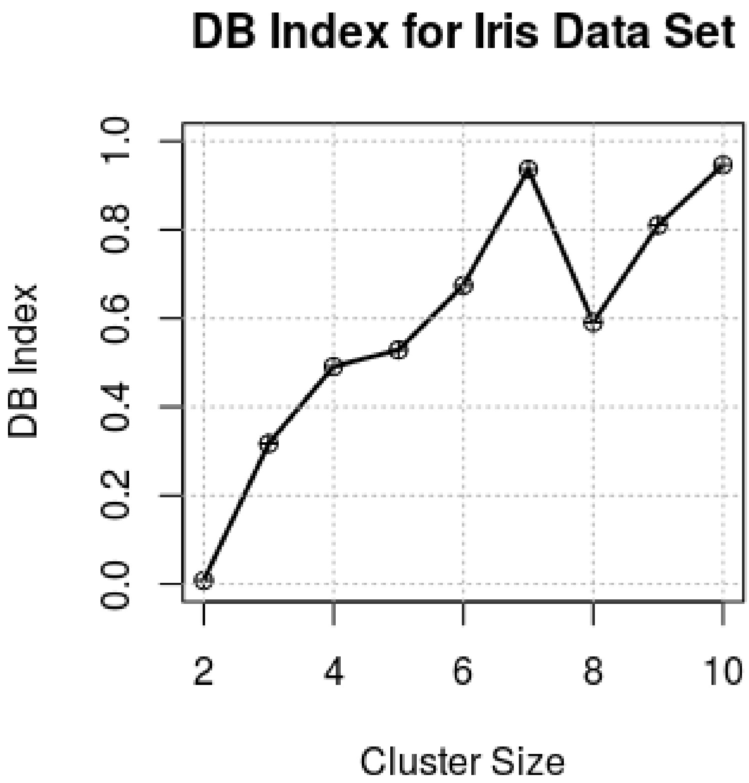

Figure 2: WSSE for KDD 1999 Data Set

### 6.3 Experiment / KDD Data Set

The KDDCup1999 dataset consists of 4.8 million points and has 42 dimensions. The features of the data set have been standardised to void any feature weighing in abnormally on the clustering.

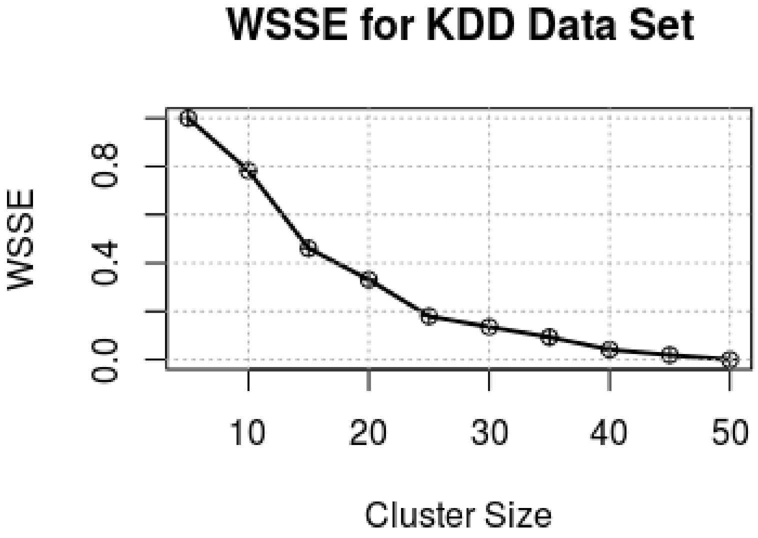

Figure 3: WSSE for KDD 1999 Data Set

A comparision of the normalised WSSE and DBIndex show varying optimal cluster sizes for the KDD 1999 data set. With the WSSE 3 its harder to identify the optimal clusters , whereas the DBIndex 4 is not monotonic in nature and based on its index value, cluster sizes of 20 and 45 can be identified as possible optimal clusters. However this graph is plotted based on the DB Index computed on the entire population.

Looking at the sampling based DB Index computed on cluster size 20 and 45, the optimal clusters identified above, the sampling error in estimating the DBIndex across various percentages is shown in 5 and in 6 respectively. As noticeable from the graphs the sampling error gradually decreases as the sampling rate reaches 100

A one-sample statistical t-test performed across the 30 samples drawn for each of the cluster sizes 20 and 45 has lead to the rejection of the null hypothesis that sample based index is equal to the population based index at 95% confidence interval with the p-values for sampling percentages of 30, 40 and 50 2.

The table 2 provides information on the p-values of the sample percentage and the normalised Standard error of the DBIndex for a poulation mean of 0.9827828.

Table 2: p-value of One Sample T-test of DBIndex

| $SamplePercentage$ | $p - value$ | $95\% Conf.IntervalValues$ |
|---|---|---|
| 0.3 | 0.000007212 | 1.079439 - 1.195468 |
| 0.4 | 0.000001595 | 1.067132 - 1.154388 |
| 0.5 | 0.00001731 | 1.023308 - 1.076916 |

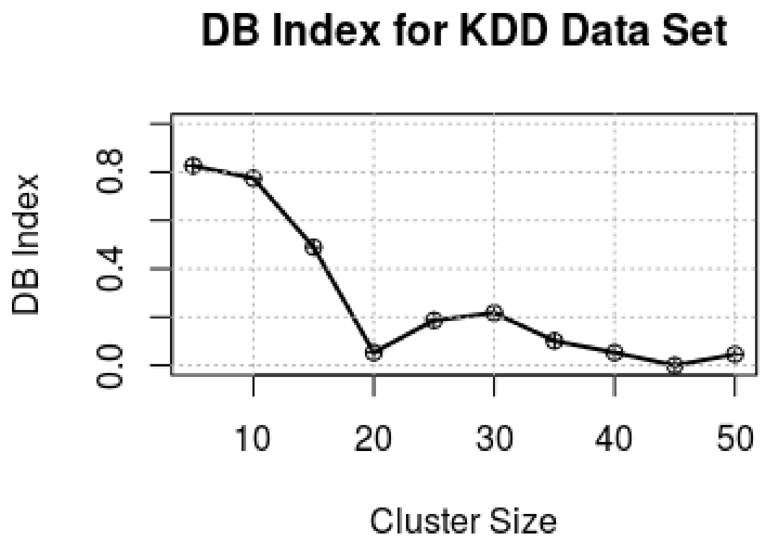

Figure 4: DBIndex for KDD 1999 Data Set

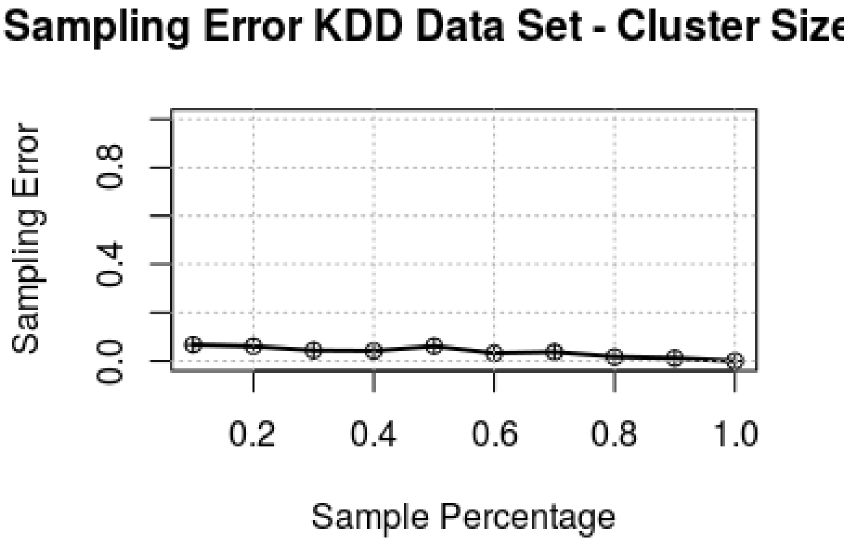

Figure 5: Sampling Based DBIndex for Cluster Size 20

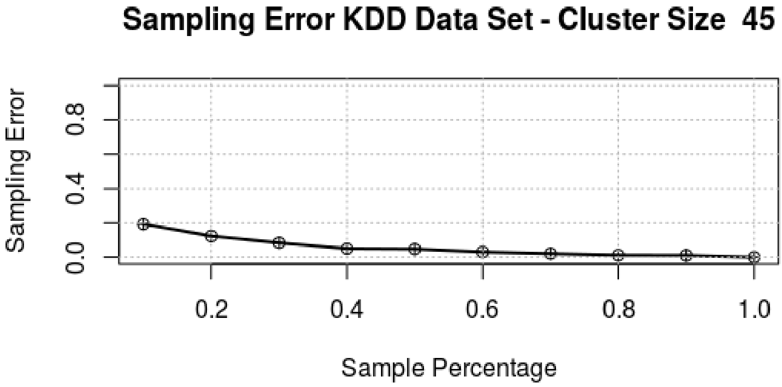

Figure 6: Sampling Based DBIndex for Cluster Size 45

## 6.4 Experiment / GoWalla Data Set

Similarly for the GoWalla DataSet the DBIndex is able to give out a clear indication that 20 is the optimcal cluster size 7 and this is supported by the WSSE Measure as well 8 based on the tapering off of the elbow bend at cluster size of 20.

However, simliar to the results on the t-test for the KDD 1999 dataset, the null hypothesis was rejected on this dataset too.

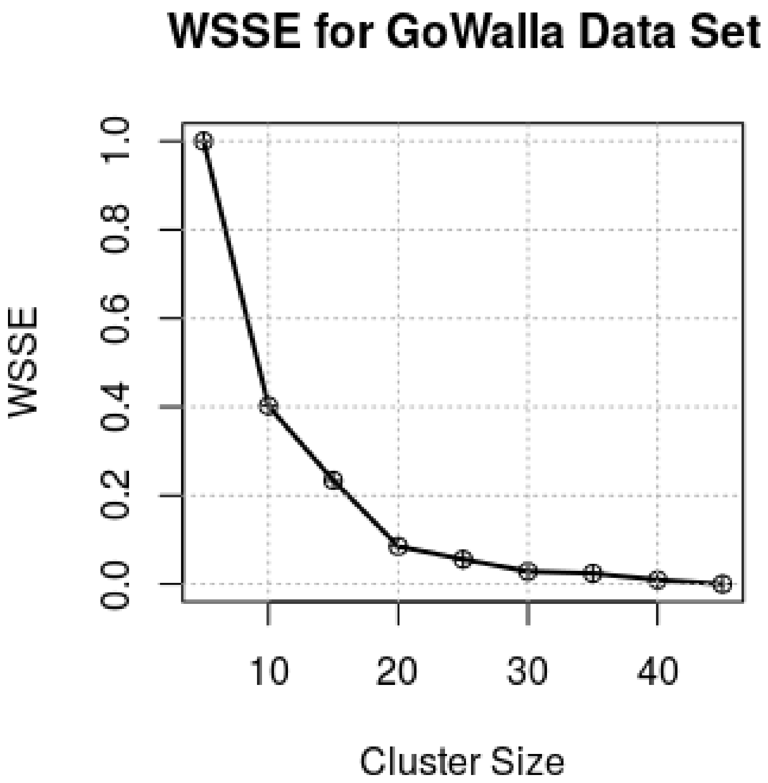

Figure 7: WSSE for GoWalla Data Set

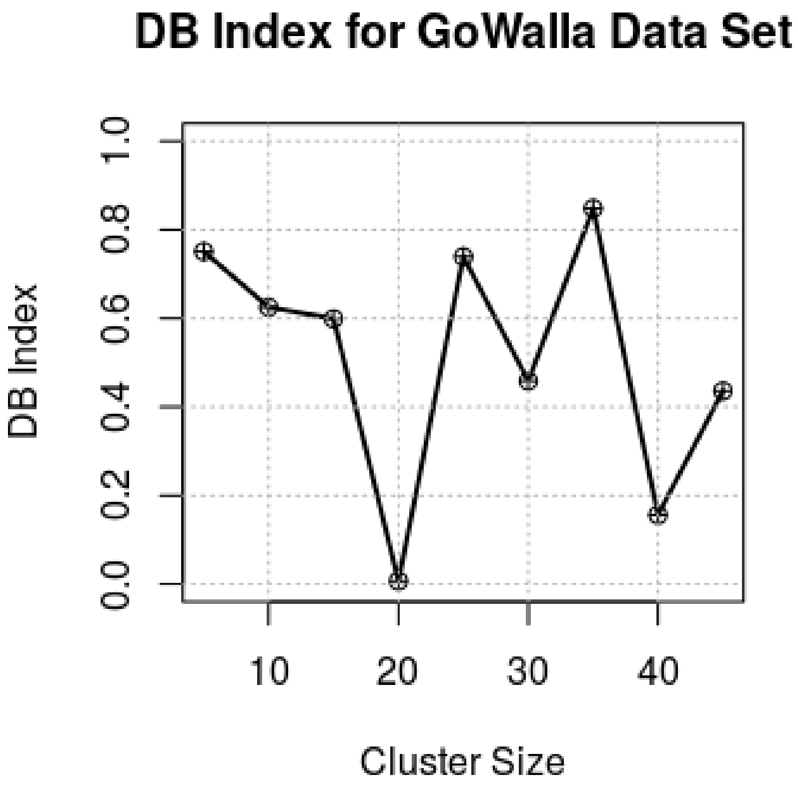

Figure 8: DBIndex for GoWalla Data Set

# 7 Conclusion and Future Work

Inspite of the limitation of the assumption of this approach that the sampled points would fit locally on the driver node and the statistical rejection of the sample based approach to computing cluster validity, in all the three data sets used in the experiment, the Davies Bouldin Index is able to help identify better the optimal size of clusters within the datasets. This reinstates the superiority of the cluster validity indices to measures that only consider one aspect of cluster validation like the WSSE.

Hence possible future work could be to try verifying this approach on other datasets and algorithms and explore alternative approaches to implementing these superior cluster validity indices for evaluating clustering algorithms on a distributed data sets.

# Acknowledgements

My first and foremost thanks to Dr.Horacio Gonzalez Velez for granting me the opportunity to pursue a Masters degree in this college through the RPEL(Recognition of Prior Experiential Learning) Scheme and to Dr.Michael Bradford for allowing a late switch to the Data Analytics stream.

I have been extremely lucky to have Dr.Simon Caton as my thesis supervisor and express my sincere gratitude for his commitment throughout the course of my thesis work. This project would not have been possible without his bundle of exciting research ideas and continued suggestions. And sincere thanks to T N Shashi, who has kindly allowed me build up on his excellent thesis work.

My endeavour as an international student purusing masters would not have been possible without the new friendships made and the ideas and interactions shared with my fantastic colleagues in this campus.

To my beloved dear ones who never cease to amaze and inspire me with their unconditional love and support.

# References

Aggarwal, C. C. and Reddy, C. K. (2013). *Data clustering: algorithms and applications*, CRC Press.

Bahmani, B., Moseley, B., Vattani, A., Kumar, R. and Vassilvitskii, S. (2012). Scalable k-means++, *Proceedings of the VLDB Endowment* **5**(7): 622–633.

Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure, *IEEE transactions on pattern analysis and machine intelligence* (2): 224–227.

Dean, J. and Ghemawat, S. (2008). Mapreduce: simplified data processing on large clusters, *Communications of the ACM* **51**(1): 107–113.

Halkidi, M., Batistakis, Y. and Vazirgiannis, M. (2001). On clustering validation techniques, *Journal of intelligent information systems* **17**(2): 107–145.

Handl, J., Knowles, J. and Kell, D. B. (2005). Computational cluster validation in post-genomic data analysis.

Jain, A. K., Murty, M. N. and Flynn, P. J. (1999). Data clustering: a review, *ACM computing surveys (CSUR)* **31**(3): 264–323.

Lange, T., Roth, V., Braun, M. L. and Buhmann, J. M. (2004). Stability-based validation of clustering solutions, *Neural computation* **16**(6): 1299–1323.

Leskovec, J., Rajaraman, A. and Ullman, J. D. (2014). *Mining of massive datasets*, Cambridge University Press.

Liu, Y., Li, Z., Xiong, H., Gao, X. and Wu, J. (2010). Understanding of internal clustering validation measures, *2010 IEEE International Conference on Data Mining*, IEEE, pp. 911–916.

Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J. and Wu, S. (2013). Understanding and enhancement of internal clustering validation measures, *IEEE transactions on cybernetics* **43**(3): 982–994.

TN, S. B. (2015). *Learning to learn with spark*, Master's thesis, National College of Ireland.

Vavilapalli, V. K., Murthy, A. C., Douglas, C., Agarwal, S., Konar, M., Evans, R., Graves, T., Lowe, J., Shah, H., Seth, S. et al. (2013). Apache hadoop yarn: Yet another resource negotiator, *Proceedings of the 4th annual Symposium on Cloud Computing*, ACM, p. 5.

Von Luxburg, U. (2010). *Clustering Stability*, Now Publishers Inc.

Von Luxburg, U. and Ben-David, S. (2005). Towards a statistical theory of clustering, *Pascal workshop on statistics and optimization of clustering*, Citeseer, pp. 20–26.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S. and Stoica, I. (2010). Spark: Cluster computing with working sets., *HotCloud* **10**: 10–10.