Data Mining:Assignment 3

==========

Krishna Mahajan,0003572903

Q1

A receiver operating characteristics (ROC) graph is a technique for visualizing, organizing and selecting classifi- ers based on their performance

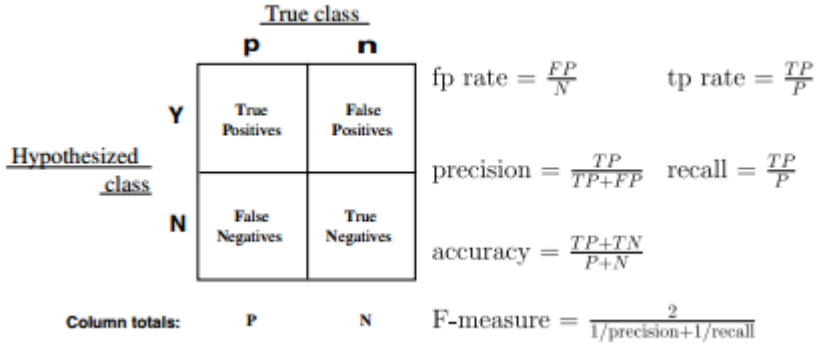
Given:
1. Dataset: $(x_i, y_i)_{i=1}^{i=n}$ where $y_i \epsilon (-1, +1)$

2. Predictions: $(p_i)_{i=1}^{i=n}$ set of predictions (either probability or discrete) for each y_i

3. n_+ and n_- original positive and negative examples in the dataset.

We are asked to find the minimum and maximum number of thresholds for which the true positive and false positive rates have to be computed in order to calculate the area under the ROC curve.

Theory For this Question i went through numerous research paper namely 1,2 and could find a efficient algorithm for generation of ROC points.



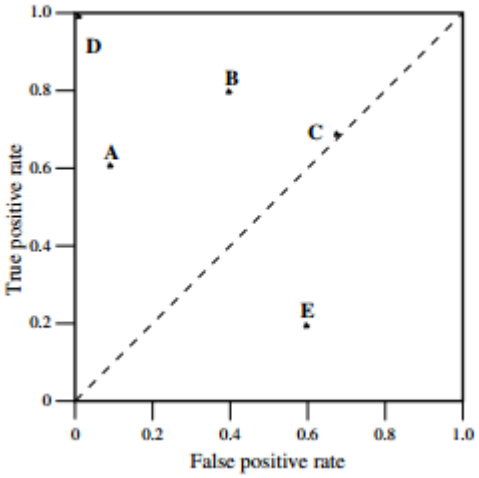
Given a classifier and an instance, there are four possible outcomes. If the instance is positive and it is classified as positive, it is counted as a true positive; if it is classified as negative, it is counted as a false negative. If the instance is negative and it is classified as negative, it is counted as a true negative.if it is classified as positive, it is counted as a false positive. Given a classifier and a set of instances (the test set), a two-by-two confusion matrix. The numbers along the major diagonal represent the correct decisions made, and the numbers of this diagonal represent the errors-the confusion-between the various classes.

The **true positive rate** (also called hit rate and recall) of a classifier is estimated as
tp rate = Positives correctly classified/Total positives

The **false positive rate** (also called false alarm rate) of the classifier is
fp rate = Negatives incorrectly classified/Total negatives

ROC graphs are two-dimensional graphs in which tp rate is plotted on the Y axis and fp rate is plotted on the X axis
An ROC graph depicts relative tradeoffs between benefits (true positives) and costs (false positives)

For example in Below Figure



Several points in ROC space are important to note. The lower left point (0, 0) represents the strategy of never issuing a positive classification; such a classifier commits no false positive errors but also gains no true positives. The opposite strategy, of unconditionally issuing positive classi- fications, is represented by the upper right point (1, 1). The point (0, 1) represents perfect classification. D's performance is perfect as shown.

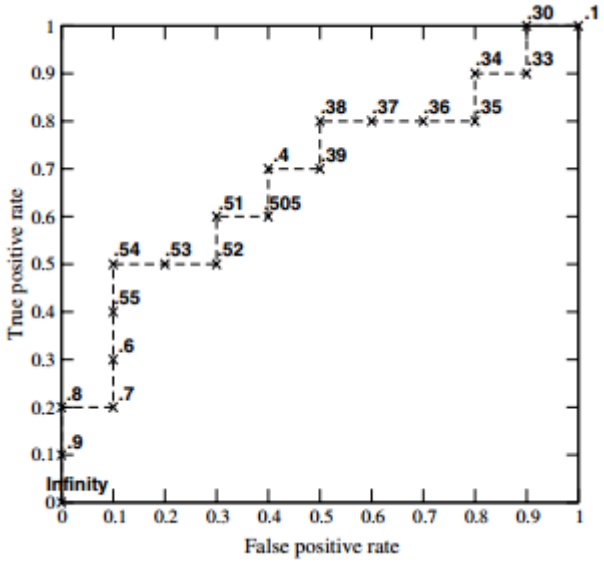
The diagonal line y = x represents the strategy of randomly guessing a class. For example, if a classifier randomly guesses the positive class half the time, it can be expected to get half the positives and half the negatives correct; this yields the point (0.5, 0.5) in ROC space.

Thus a minimum of 2 (Tp rate,FP rate /Threshold) measurements are needed if we assume (0,0) & (1,1) points are not in the ROC curve

** If we consider (0,0) & (1,1) points are present in ROC then only 1 threshold is needed to calculate AUC under ROC**

For maximum no. of thresholds Conceptually, we may imagine varying a threshold from -∞ to ∞ and tracing a curve through ROC space. Computationally, this is a poor way of generating an ROC curve.

Below shows an example of an ROC ''curve" on a test set of 20 instances. The instances, 10 positive and 10 negative, are shown in the table beside the graph. Any ROC curve generated from a finite set of instances is actually a step function, which approaches a true curve as the number of instances approaches infinity



Now for maximum no. of thresholds points we can exploit the monotonicity of thresholded classifications any instance that is classified positive with respect to a given threshold will be classified positive for all lower thresholds as well.Therefore can simply sort the test instances decreasing by f scores and move down the list, processing one instance at a time and updating TP and FP as we go. In this way an ROC graph can be created from a linear scan.
The algorithm is shown in below Figure. TP and FP both start at zero. For each positive instance we increment TP and for every negative instance we increment FP. We maintain a stack R of ROC points, pushing a new point onto R after each instance is processed. The final output is the stack R, which will contain points on the ROC curve.

Algorithm 1. Efficient method for generating ROC points
Inputs: L, the set of test examples; $f(i)$, the probabilistic classifier's estimate that example i is positive; P and N, the number of positive and negative examples.
Outputs: R, a list of ROC points increasing by *fp rate*.
Require: $P > 0$ and $N > 0$

```
1: L_sorted ← L sorted decreasing by f scores
2: FP ← TP ← 0
3: R ← ⟨⟩
4: f_prev ← −∞
5: i ← 1
6: while i ⩽ |L_sorted| do
7:    if f(i) ≠ f_prev then
8:       push (FP/N, TP/P) onto R
9:       f_prev ← f(i)
10:   end if
11:   if L_sorted[i] is a positive example then
12:      TP ← TP + 1
13:   else      /* i is a negative example */
14:      FP ← FP + 1
15:   end if
16:   i ← i + 1
17: end while
18: push (FP/N, TP/P) onto R   /* This is (1,1) */
19: end
```

Let n be the number of points in the test set. This algorithm requires an **O(nlogn)** sort followed by an **O(n)** scan down the list, resulting in **O(nlogn)** total complexity.

Thus from the above Algorithm, the maximum no. of thresholds ever needed for calculation of ROC points woulbe N i,e the no of datapoints

In simple terms we'll create Thresholds only for data points for different posterior probability $P(+/A)$ and max number of different posterior probability we can get is no of datapoints i,e. N

For further detailed answer it'll be unique no' of posterior probability $(p_i)_{i=1}^{i=n}$ which can be N

Q2

Given=$d((x_i, y_i))_{i=1}^{i=n}$, $\binom{n}{2}$ pair of distances , m clusters of these points.

To Find : Algorithm to find which two clusters to merge out of $\binom{m}{2}$ pairs of possible clusters to minimize SSE of new merged cluster.

First , As we know the SSE of a cluster (c_i) is defined as Average squared distance of all the points from the centroid of the cluster. i,e
SSE_{c_i}=$\sum_{x \epsilon C_i} dist(c_i, x)^2$

In the book it is proved that SSE of a cluster is equal to average pairwise distance of all the points in the cluster as follows

SSE_{c_i}=$\sum_{x \epsilon C_i} dist(c_i, x)^2 = \frac{1}{2|c_i|} \sum_{x \epsilon C_i} \sum_{y \epsilon C_i} dist(x, y)^2$

Proof

$$
\begin{aligned}
\frac{1}{2|C_i|}\sum_{x\in C_i}\sum_{y\in C_i}(x-y)^2 &= \frac{1}{2|C_i|}\sum_{x\in C_i}\sum_{y\in C_i}((x-c_i)-(y-c_i))^2 \\
&= \frac{1}{2|C_i|}\Big(\sum_{x\in C_i}\sum_{y\in C_i}(x-c_i)^2 - 2\sum_{x\in C_i}\sum_{y\in C_i}(x-c_i)(y-c_i) \\
&+ \sum_{x\in C_i}\sum_{y\in C_i}(y-c_i)^2\Big) \\
&= \frac{1}{2|C_i|}\Big(\sum_{x\in C_i}\sum_{y\in C_i}(x-c_i)^2 + \sum_{x\in C_i}\sum_{y\in C_i}(y-c_i)^2\Big) \\
&= \frac{1}{|C_i|}\sum_{x\in C_i}|C_i|(x-c_i)^2 \\
&= \text{SSE}
\end{aligned}
$$

In this equation we know everything,as the pairwise distances and all the m clusters are given.So we can calculate SSE of any cluster.

Now, we need to best two clusters C_x and C_y to merge out of set of $\binom{n}{2}$ clusters 'which should have the minimum $SSE_{C_xC_y}$ after merging to form the new cluster
i,e

$SSE_{C_xC_y}$=$\frac{1}{2|C_x|}\sum_{x\epsilon C_x}\sum_{y\epsilon C_x}dist(x,y)^2 + \frac{1}{2|C_y|}\sum_{x\epsilon C_y}\sum_{y\epsilon C_y}dist(x,y)^2 + \frac{1}{2|C_x+C_y|}\sum_{x\epsilon C_x}\sum_{y\epsilon C_y}dist(x,y)^2$

The above equation is self explanatory i,e the SSE of the new cluster 'll be the sum of SSE of two combined clusters + Average sum of distance of points from one cluster to other cluster. Also we have every information to calculate $SSE_{C_xC_y}$

So the Algorithm should look like

For every $C_i\epsilon M$:
....calculate sse_{C_i}
....for every other $C_j\epsilon M$:
.....calculate $sse_{C_iC_j}$
....end for end for

Return C_i, C_j for which $SSE_{C_iC_j}$ is minimum.

Q3

A)

Please find the code for both $F_{k-1}XF_{k-1}$ and $F_{k-1}XF_1$.
Module:aprioriGn.py

I maintained counter for count of candidate itemsets and count of frequent itemsets in actual **apriori** code which can be seen here
Module:apriori.py

Instructions to run the code can be seen in the attached **readme.md** file.

B)

In this problem i took up datasets from UCI repository which were categorical,numerical or both and correspondingly binarized the datasets removing redundant columns.I implemented binarization function in R.
Module:binarization.r

Dataset 1:Car Evaluation

This is completely categorical dataset with dimension $1728 * 7$. After removing some redundant columns i binarized this dataset and converted into csv file.(This binarized file can be seen as binarize_car.csv in ./Data/Car folder).
After executing apriori code on this dataset using the two candidate generation methods i got following results for three different threshold values of support.

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets
0.01	190	84
0.1	78	18
0.3	12	1

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets
0.01	281	84
0.1	64	18
0.3	12	1

All the corresponding frequent itemsets with the their corresponding support values can be seen in ./Code/Data/Car/support_*.csv files.

Dataset 2: Nursery store

This is completely categorical dataset with dimension $12960 * 9$.I binarized this dataset and converted into csv file.(This binarized file can be seen as binarize_car.csv in ./Data/Nursery folder).
After executing apriori code on this dataset using the two candidate generation methods i got following results for three different threshold values of support.

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets
0.01	23880	7028
0.1	870	164
0.3	151	17

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets
0.01	47372	7034
0.1	1550	164
0.3	159	17

All the corresponding frequent itemsets with the their corresponding support values can be seen in ./Code/nursery/Car/support_*.csv files.

Dataset 3:Groceries

This is actual transactional dataset of a belgian store and most practical dataset to apply Apriori Algorithm.This datasets has more than 9K transactions and over 500+ different items. After executing apriori code on this dataset using the two candidate generation methods i got following results for three different threshold values of support.

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets
0.01	3927	217
0.1	178	5
0.3	168	0

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets
0.01	4418	217
0.1	178	5
0.3	168	0

All the corresponding frequent itemsets with the their corresponding support values can be seen in ./Code/Data/Groceries/support_*.csv files.

As can be observed in the data $F_{k-1}XF_{k-1}$ generates less candidate itemsets and thus is more efficient than $F_{k-1}XF_1$

C)

Maximal Frequent Itemset : Frequent item set $X \epsilon F$ is maximal if it does not have any frequent supersets module:maximal_frequent_itemset.py

Closed Frequent Itemset : Frequent item set $X \epsilon F$ is closed if it has no superset with the same frequency. module:closed_frequent_itemset.py

Dataset 1:Car Evaluation

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets	maximal_frequent	closed_frequent
0.01	190	84	40	78
0.1	78	18	9	18
0.3	12	1	1	1

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets	maximal_frequent	closed_frequent
0.01	281	84	40	78
0.1	64	18	9	18
0.3	12	1	1	1

Dataset 2: Nursery store

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets	closed_frequent	maximal_frequent
0.01	23880	7028	3294	2113
0.1	870	164	111	75
0.3	151	17	15	5

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets	closed_frequent	maximal_frequent
0.01	47372	7034	3294	2113
0.1	1550	164	111	75
0.3	159	17	15	5

Dataset 3:Groceries

$F_{k-1}XF_{k-1}$

support	candidate_itemsets	frequent_itemsets	maximal_frequent	closed_frequent
0.01	3927	217	159	172
0.1	178	5	5	5
0.3	168	0	0	0

$F_{k-1}XF_1$

support	candidate_itemsets	frequent_itemsets	maximal_frequent	closed_frequent
0.01	4418	217	159	172
0.1	178	5	5	5
0.3	168	0	0	0

The above results confirm that **Candidate items > Frequent Itemsets > closed Frequent> Maximal Frequent**

d)

The code for confidence based rules can be found here
module:mining.py

Dataset 1:Car Evaluation

support	confidence	rules_generated	pruned_rules
0.01	0.25	175	59
0.01	0.50	175	10
0.01	0.75	175	3
0.1	0.25	16	11
0.1	0.50	16	8
0.1	0.75	16	3
0.3	0.25	0	0
0.3	0.50	0	0
0.3	0.75	0	0

Dataset 2: Nursery store

support	confidence	rules_generated	pruned_rules
0.01	0.25	47372	7034
0.01	0.5	34462	754
0.01	0.75	34460	4
0.1	0.25	281	270
0.1	0.50	281	65
0.1	0.75	281	2
0.3	0.25	2	2
0.3	0.50	2	2
0.3	0.75	2	2

Dataset 3:Groceries

support	confidence	rules_generated	pruned_rules
0.01	0.25	296	75
0.01	0.5	296	2
0.01	0.75	296	2
0.1	0.25	0	0
0.1	0.50	0	0
0.1	0.75	0	0
0.3	0.25	0	0
0.3	0.50	0	0
0.3	0.75	0	0

e)

To generate the top 10 rules i sorted(max to min) rules based on confidence for each support value.
For a particular support value the rule with more confidence 'll be a better rule then the rule with lesser confidence.

Dataset 1:Car Evaluation
Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/Car/rules/confidence)** where each csv files shows the top 10 rules.

Dataset 2: Nursery store
Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/nursery/rules/confidence)** where each csv files shows the top 10 rules.

Dataset 3:Groceries
Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/Groceries/rules/confidence)** where each csv files shows the top 10 rules.

On my Observation for top 10 rules on Groceries dataset (Most practical dataset) i got following rules from the above support and confidence values

rules	support	confidence
'onions' ——> 'other vegetables'	1%	0.510638298
'beef' ———>,'whole milk'	1%	0.504716981
'curd'———>,'whole milk'	1%	0.491916859
'butter'——->,'whole milk'	1%	0.488272921
'root vegetables'———>'whole milk'	1%	0.478316327
'beef'——->,'other vegetables'	1%	0.466981132
'root vegetables'————>,'other vegetables'	1%	0.464285714
'domestic eggs'——->,'whole milk'	1%	0.45979021
'tropical fruit'———>'whole milk'	1%	0.447272727

All these rules seems to be very relevant.1% support seems low but considering this transaction are over 6 months duration 1% 'll be very significant. Rules for Other support Confidence combination can be found in csv files.
Rules with very low support but high confidences then can be thought as peculiar becuase they are rare transactions but are consistent in thier pattern.

f)

Lift is one more parameter of interest in the association analysis. Lift is nothing but the ratio of Confidence to Expected Confidence,Using the above example, expected Confidence in this case means, "confidence, if buying A and B does not enhance the probability of buying C."For the supermarket example the Lift = Confidence/Expected Confidence.Hence, Lift is a value that gives us information about the increase in probability of the then (consequent) given the if (antecedent) part.
A lift ratio larger than 1.0 implies that the relationship between the antecedent and the consequent is more significant than would be expected if the two sets were independent. The larger the lift ratio, the more significant the association

The code for lift based rules can be found here
module:mining_lift.py

Dataset 1:Car Evaluation

support	Lift	rules_generated	pruned_rules
0.01	0.85	175	156
0.01	0.90	175	142
0.01	0.95	175	135
0.1	0.85	16	16
0.1	0.90	16	16
0.1	0.95	16	12
0.3	0.85	0	0
0.3	0.90	0	0
0.3	0.95	0	0

Dataset 2: Nursery store

support	confidence	rules_generated	pruned_rules
0.01	0.85	54589	54295
0.01	0.90	54589	53654
0.01	0.95	54160	52493
0.1	0.85	281	281
0.1	0.90	281	281
0.1	0.95	281	277
0.3	0.85	2	2
0.3	0.90	2	2

support	confidence	rules_generated	pruned_rules
0.3	0.95	2	2

Dataset 3:Belgian store

support	confidence	rules_generated	pruned_rules
0.01	0.85	296	296
0.01	0.90	296	296
0.01	0.95	296	296
0.1	0.85	0	0
0.1	0.90	0	0
0.1	0.95	0	0
0.3	0.85	0	0
0.3	0.90	0	0
0.3	0.95	0	0

To generate the top 10 rules i sorted(max to min) rules based on Lift for each support value.
For a particular support value the rule with more Lift 'll be a better rule then the rule with lesser confidence.

Dataset 1:Car Evaluation Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/Car/rules/lift)** where each csv files shows the top 10 rules.

Dataset 2: Nursery store
Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/nursery/rules/lift)** where each csv files shows the top 10 rules.

Dataset 3:Groceries Top 10 rules for this dataset for each support and each threshold in above table can be in found in folder **(.Data/Groceries/rules/lift)** where each csv files shows the top 10 rules.

Q4

Summary Paper 1

Impossibility Theorem

Clustering means proximating similar objects from a many different observations. The similarity between observations is usually measured in the form of distance/proximation between two observations . Smaller the distance, greater the similarity between objects. Though similarity is one of the measures for forming clusters. Every set of observation has their own definition of similarity Clustering, which is more of an domain related work when we try to change it into an generalised algorithm. There are many different clustering algorithms, which lead into different type of conclusions.According to Jardine and Gibson's paper, work in clustering analysis involves axiomatic approaches which necessisates in hierarchical clustering.
Puzicha on the other hand, emphasis more on the efficiency of partition function for clustering.Some researchers have recently defined properties that are needed to uniquely specify any clustering formulation.

Distance function is used to calculate the similarity. However this metric is not imposed by using the distance metrics. A clustering function is defined that inputs the distances and returns the K partition sets. The

following properties are observed:
1. **Scale Invariance**:
It checks that the selected clustering function is not sensitive to the different scales of measurements. For this, $\alpha > 0$ is used such that f(d) = f(αd).
2. **Richness**:
This property says that each partition of the set is equally favourable to be the output. It is defined by $Range(f)$.
3. **Consistency**:
This shows if the inter-cluster distance is minimized and intra-cluster distance is maximized, the clustering result obtained must be very similar to the previous scale.The publishers uses the transformation function and defines two distances, **(d)** the older distance and **(d')** the new distance. This is imposed to satisfy the conditions of shrinking and expanding the distances. Function is consistent if **f(d) = f(d')** whenever d'is an consistent with d.

With looking on these properties, the publishers characterizes those 1st property that though n >= 2, it will be impossible for whatever grouping capacity to fulfill every last one of over three said properties. This may be demonstrated Toward taking the illustration about single-linkage grouping strategies. The fundamental idea of those methodology is formation about weighted graph and sub graphs might be taken as likewise groups/clusters. For this suitable stoppping states are characterized. There are three sorts for stopping conditions:
1. K - cluster stopping condition: stop when there are 'k' connected parts.
2. Separation - r stopping condition: include edges with weight at most 'r'.
3. Scale - α stopping condition: it means those most extreme pairwise separation Also edges for weight at most α*p may be included.

On basis of these stopping conditions, it is inferred that exactly two of the properties are satisfied by using any of the stopping condition.
a) Single Linkage - (k-cluster) - Scale Invariance, Consistency. (k >= 1, n >= k)
b) Single Linkage - (distance-r) -Consistency, Richness. (n >= 3, α<1)
c) Single Linkage - (scale-α) - Scale Invariance, Richness. (r >0, n >= 2)

Antichains
To prove the above results a special set of clusters is used. The theorem says that every partition will have much refined partition set . This partition will be the subset of the original partitions. For example, if the new cluster is A and B, there will be divison of clusters A such that A $\subseteq$ A and B $\subseteq$ A. From this, the idea of Antichain is defined as the sets of partitions such that no two partitions are the refinement of each other.

From now on ,the Publishers proofs his statement of impossibility. ——>First theorem says that if the function satisfies the Scale Invariance and Consistency, then Range(f) will be an antichain,publishers prove his point with a contradiction. Here the proof assumes that the clustering function is defined on entire set of data points. ——>Next theorem takes help of the sum-of-pairs method to show that for every antichain of partition, if the function satisfies the scale Invariance and Consistency, the Range(f) will be the antichain of partition. Here the sum-of-pairs function is used that can minimize the distances between pairs of points. The entire transformation is shown by series of calculations.

The author has also given a beautiful analysis of how centroid based clustering, which is one of the most common clustering methods, is contrastingly related to the property of Consistency. It is very difficult to generate a consistent k-means or k-median clustering approach as choosing the centroids initially in fairly based on intuition. There is no specific method for generating initial set of centroids.

At the end author introduces the concept of Refinement-Consistency. Even then he argues that even then none of the clustering function can satisfy these properties. However, there are slim chances where some functions can satisfy Scale-Invariance, Refinement-Consistency and Near-Richness properties.

Conclusions

Strengths:
. The paper has very well explained the importance of scale-invariance, richness and consistency on the

generalized form of clustering. The impossibility theorem, explains that two of the above properties could be easily satisfied for any given clustering algorithm. . The author has also proved the impossibility theorem by giving simple and easily understandable proof. He has also gone ahead to give examples of relaxation of every property and how other two properties are satisfied for particular clustering function.
Weakness:
. Kleinberg has not mentioned why he chose only three properties - scale invariance, richness and consistency and what is the importance of other properties such as accuracy, efficiency, etc.

Q5

Summary Paper 2

MEASURES OF CLUSTERING QUALITY

The Publisher of this paper are Margareta Ackerman and Shai Ben-David who are group of researchers at University of Waterloo. This paper is related to the above discussed paper publication about the "Impossibility Theorem". The publisher of this paper rejects the conclusions of Kleinberg's work. The Publishers in this research introduces the notion of Cluster Quality . The publishers argues that instead of using the clustering functions for three measures i,e Scale-Invariance, Richness and Consistency, we can use following quality measures to satisfy the same.

Cluster Quality Measure :It will give a non-negative number as an output that 'll judge how good the partioning algorithm . This Cluster Quality Measure also satisfies all the three basic axioms given in above paper. Initially the Publishers argue that the impossibility is not the inherent feature and it is due to some of the prior assumptions that the function becomes impossible. To eliminate this, the publishers here proves that the Cluster Quality Measure are more flexible measurement than the clustering function in above paper.

However this question arises that what is the need of such measures and how those measures could be used. But on looking carefully it can determines that whether the results of the clustering algorithm can be relied upon. To verify the claim that CQM are better, various axioms in relation to Kleinberg's axioms are formulated. Basically the CQM's running complexity is polynomial time.

Further the author reformulates the three basic properties defined by Kleinberg to incorporate the concept of CQM. The axioms are Scale-Invariance, Richness and Consistency. The set of these properties form the consistent set of requirements. Thereafter the author defines the first CQM.

Relative Margin
. Relative margin works as the average of distances. Basically when a point in cluster is selected, along with it we select two closest centres to that point and take the ratio of their distances. This ratio is defined as Relative Point Margin. . The set of all such distance ratios make up the Representative Set. . Finally the Relative margin is the average of this distance. The smaller value of Relative Margin indicates the better quality of clustering.

Along with this, author gives two notions that should be followed by the axioms
1) Soundness: All the elements should satisfy all the axioms.
2) Completeness: All the properties satisfied by all the elements must be included as axiom.

So with this, the author mentions that the CQM should have soundness and completeness as to satisfy all the axioms, while the clustering functions may fail to have those properties. He mentions that the Kleinberg's approach fails in soundness. They also introduced the new concept of Isomorphism Invariance.

Isomorphism: Publisher defines that 2 cluster of can be isomorphic only if the distance between is preserved. They emphasis that clustering should not be very sensitive to the individual clusters identity.
On basis of these axioms, They defines that a consistent set of axioms that a Cluster Quality Measure should satisfy includes Isomorphism-Invariance, Scale-Invariance, Richness and Consistency.

Then at the last publishers gives examples of Cluster Quality Measure.

Weakest Link . Whenever a cluster is formed, the points belonging to same cluster are tightly packed. They have links between them. However the points belonging to different clusters are loosely packed . This concept of Weakest link focuses on the longest tight links between the points . Then after the maximum value of weakest link is divided by the minimum distance between two clusters. The range of this measure can be (0, ∞) Point Margin. . The Additive Margin the ratio of average Additive Point Margin to average intra cluster distances.

Publisher demonstrates that complexity of this measures is polynomial and function of k and takes O(nˆK+1) time. But if the centres are known, than Relative Margin takes O(nk) time, Additive Margin takes O(n2) time and Weakest Link takes O(n3) time.

Before concluding , Publishers says that there can be influtential dependence on the no of partions. All of thier methods are independent to those, but a final cconclusion on how to handle the number of clusters is still a active topic of research. According their theory, the objective functions will fail to satisfy the Scale-Invariance and Richnes ,but this can be overcome using normalization.

.

Conclusion: On concluding the paper, publisher opposes the Impossibility theorem and gives a solution to use the Cluster Quality Measures instead of Clustering Functions. On reading this I got convinced that the impossibility theorem seems not be accurate. Each methods have their own advantages. However the dependence on the number of clusters is an important part to research still. Also there will be a threshold up to which these Cluster Quality Measures can work.