YUM-Clust

Eriq Augustine EAUGUSTI@UCSC.EDU
University of California, Santa Cruz

Varun Embar VEMBAR@UCSC.EDU
University of California, Santa Cruz

Dhawal Joharapurkar DJOHARAP@UCSC.EDU
University of California, Santa Cruz

Xiao Li XLI111@UCSC.EDU
University of California, Santa Cruz

Abstract

We utilize the data provided in the Yelp Dataset Challenge to help business owners find potential competitors if they choose to open a branch in a new location. Using this information the business owners can not only find potential competitors, but also get an idea of how well their business will be received in the new locality.

We say that a "competitor" is any similar business, with the idea that a similar business caters to a similar clientele and is therefore a competitors. Therefore our task involves finding similar businesses in other regions. We do this by first clustering similar businesses. Once we have the clusters, to find similar businesses in a location, we look into the cluster to which a given restaurant belongs to, and then display restaurants in that cluster that are close to the location.

We use the k-medoids algorithm to cluster the businesses as not all attributes are numeric in nature and we need a richer set of dissimilarity scores. We run experiments with various parameter settings, dissimilarity scores, and features and report the Rand index on a "gold standard" data set as well as a human evaluated data set. Both experiments report a high degree of success.

1. Introduction

TODO(eriq):

2. Algorithm Formulation

2.1. Clustering

We use the K-medoid clustering algorithm to cluster businesses. K-medoids is a clustering technique that tries to minimize the pairwise dissimilarity between data points that are assigned to a cluster and the medoid of that cluster. The medoid is a data point in data set that best represent the cluster center. The medoid is analogous to the centroid in the K-Means algorithm.

The K-Medoids algorithm has the following advantage over K-Means.

- Since we are only using pairwise comparisons, we do not need to be able to calculate the average of a feature. This is better for non-numeric features like strings or sets.
- Since we are only using pairwise comparisons, once we compute the dissimilarity between the data points, we can do away with the actual data points. This leads to a reduction in the memory usage and makes the computation more efficient.
- Since our medoid is a real data point (as opposed to a centroid), we can use this data point as a "representative" when visualizing our data (or perhaps even sampling it).

Our stopping condition is either when we have run through 10 iterations of clustering (experimentally found to be sufficient with this dataset), or when the current set of clusters is the same as either the last run or the run before it. We compare not just the most recent run so that we can prevent unnecessary runs when the clusters are *jittering*, or when

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

outlier points are constantly shifting their membership back and forth between two or more clusters.

2.2. Distance Metrics

Our distance metrics fall into two categories: numeric and set-based. For numeric, we implemented L1 and L2 distances. For set-based, we implemented Jaccard and Dice.

Once the distance for each feature is calculated, they are summed together to make our total distance. To make combining the distance of different features together more reasonable, we also attempt to normalize the output of the distance to ideally be in the range 0 to 1.

2.3. L1 (Manhattan) Distance

The Manhattan distance of two numbers x and y is calculated by:

$$d_{L1} = \sum_{i=1}^{n} |x_i - y_i|$$

2.4. L2 (Euclidean) Distance

The Euclidean distance of two numbers x and y is calculated by:

$$d_{L2} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

2.5. Jaccard Distance

The Jaccard distance of two set A and B is calculated by:

$$d_j(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

2.6. Dice distance

The Dice distance of two set A and B is calculated by:

$$d_d(A,B) = 1 - \frac{2|A \cap B|}{|A| + |B|}$$

2.7. Normalizations

Because different distance metrics work with different output ranges, we need to normalize the distances metrics to something consistent. For example, the range of Jaccard Distance is between 0 and 1 while the range of Manhattan Distance is 0 to infinity.

To figure out what works best, we will try three different normalization methods:

1. **Raw** - No normalization is applied.
2. **Logarithmic** - Use a logarithm to squash down the value closer to zero. Since we did not want to let values less than 1 grow, we put a hinge at 1 and just let all values less than 1 go to zero.

$$distance(d) = ln(max(1, d))$$

3. **Logistic** - Use the Sigmoid function to squash the distance down to a 0-1 range. Since all our distance metrics return non-negative values, we can ensure that the following returns a value in the range of 0 to 1:

$$distance(d) = (\frac{1}{1 - exp(-d)} - 0.5) * 2$$

2.8. Rand Index

To help evaluate the correctness of a cluster, we will be using the Rand index, which is used to measure the similarity between two cluster assignments.

Given a set of elements S, and two partitions of S, $X = \{X_1, X_2, \ldots, X_n\}$ and $Y = \{Y_1, Y_2, \ldots, Y_m\}$, we compute the following

- a - Number of pairs that are assigned to same cluster in both X and Y
- b - Number of pairs that are assigned to different clusters in both X and Y
- c - Number of pairs that are assigned to same cluster in X but to different clusters in Y
- d - Number of pairs that are assigned to different clusters in X but to same cluster in Y

The Rand index is then given by

$$R = \frac{a+b}{a+b+c+d}$$

Intuitively, the Rand index can be thought of as the number pairs that were correctly put together or kept apart over the total number of pairings.

3. Features

Our set of features can be divided into three types of features:

- Numeric Features
  - **Star Rating** - The star ratings of a business, rounded to half-stars.
  - **Total Review Count** - The total number of reviews present for the business.

  - **Available Review Count** - The number of reviews available for the business in the data set.
  - **Mean Review Length** - Average length of a review for the business.
  - **Mean Word Length** - Average length of the words in the reviews for the business.
  - **Number of Words** - Total number of words in the reviews of the business.
  - **Mean Words per Review** - Average number of words per review available for the business.

- Descriptive Features
  The dataset also contains some textual data such as attributes, categories, review text, etc. which describe the businesses. We restrict ourselves to not using the name of the business as a feature since this will make clustering out data set trivial. We construct features using these texts, but to improve efficiency of our model we encode the string data as a set of unique integers by building maps over all possible values that these features can take. So, our output feature is a set of identifiers of the words present for the business.

  - **Attribute Features** - The dataset contains information about attributes of businesses which describe the operations of the business. They exist as key-value pairs in the data, for example - (WiFi, no), (Delivery, false), (Attire, casual). We squash the key-value pairs together and add these together to create a set of attributes that a business has, which we then use as a feature.
  - **Category Features** - The dataset also contains some categorical information about the business, for example, whether the business is a restaurant, cafe, food place, burgers place, etc. We construct the feature which is the set of categories that the business has assigned to it.
  - **Key Words** - These are words that Yelp has defined to help users in filtering out the businesses that appear in the search results. They're words that delineate businesses as they're mostly categorical words such as restaurant, cafes, etc. We use these key words and look for their occurrences in the reviews of the businesses and return a set of key words that they contain.
  - **Top Words** - This set contains the most frequently occurring words in the reviews of the text, after taking care of the stop words. We used a general English language stop words list containing 562 stop words.

- Temporal Features
  We have two time-related features pertaining to the functioning hours of businesses.

  - **Total Hours** - Total number of hours the business is open during the week
  - **Open Hours** - This is a feature encoded information about the functioning hours of the business over the week. We divided the hours in a day in the following way to help us attribute the functioning times to a business.

    ∗ Open between 6AM - 12PM: The restaurant functions in the morning, or serves breakfast.
    ∗ Open between 12PM - 3PM: The restaurant functions in the afternoon, or serves lunch.
    ∗ Open between 5PM - 9PM: The restaurant functions in the evening, or serves dinner.
    ∗ Open between 9PM - 2AM: The restaurant functions post dinner, or late night.

    To make our feature more robust, we specify that a business must be open for at least four days a week in the given time-span to be marked as open in that time-span. We again return the time-spans that a business operates during as a set.

3.1. Weights

To each feature, we attach a weight that ranges from 0 to 4. The inverse of this weight is multiplied by the distance calculated from this weight before it is added to the distance sum. A weight of zero removes this feature from consideration. A weight less than one causes the effective weight to grow. A weight of 1 has no additional effect. A weight greater than one causes the effective weight to shrink.

4. Datasets

We created two datasets from the given Yelp data. Both sets only include "restaurants", businesses that have one of the following categories: "Restaurants", "Food", "Fast Food", "Pizza", "Mexican", "Sandwiches", "American (Traditional)", "Burgers", "Italian", "Chinese", "American (New)", "Breakfast & Brunch", "Cafes".

4.1. Chains

To generate a large set of approximate clusters, we utilized the fact that franchises in a restaurant chain should all be very similar and hence in the same cluster.

We looked at various restaurant chains such as "McDonald's", "Starbucks" etc. that are present in the data and assign all the stores belonging to the same chain as a new cluster. (Note that we have disallowed using the name of the restaurant as a feature.) We extracted the top 15 restaurant chains, listed in Table 1. We only generate pairs within a restaurant chain. Since it is not clear if a McDonald's franchise and a Burger King franchise should be in the

Restaurants	No of branches
Starbucks	527
Subway	408
McDonald’s	365
Taco Bell	193
Burger King	167
Pizza Hut	159
Wendy’s	149
Panda Express	122
Dunkin’ Donuts	122
Domino’s Pizza	107
KFC	99
Chipotle Mexican Grill	97
Dairy Queen	96
Papa John’s Pizza	92
Jack in the Box	81
Fine Dining	100

Table 1. List of restaurant chains.

same or different cluster, we do not look at pairs across various chains.

However if we only have positive pairing, then the Rand index can be trivially maximized by assigning all data points to the same cluster. To counteract this, we also need pairs of restaurants that should not be in the same cluster. To achieve this, we collected a list of 285 “Fine Dining” restaurants (restaurants which have the highest price range), and create pairs such that the first restaurant comes from the “fast food” list like “McDonald’s” and “Taco Bell” while the other comes from the “Fine Dining” list. These pairs should definitely not be in the same cluster.

4.2. Human Annotated

In addition to the automatically generated data set, we manually annotated a set of 100 restaurants. To keep with our problem statement of finding similar restaurants in different regions, we choose the two most represented cities in the Yelp data: **Las Vegas** and **Phoenix**. From each city, we choose one restaurant from each of our chains (including “Fine Dining”) and 34 other restaurants that have at least 50 reviews. This gave us a total of 100 restaurants.

We then manually scored the 4950 unique pairs of restaurants on a scale of 1 (very dissimilar) to 5 (very similar). A score of 3 is considered neutral and does not count towards a positive or negative paring.

5. Experiments

There are five different parameters that we can tune in our algorithm:

Normalization	Set Distance	Rand Index
Log	Dice	0.68435
Log	Jaccard	0.67145
Logistic	**Dice**	**0.82638**
Logistic	Jaccard	0.78539
None	Dice	0.66708
None	Jaccard	0.66708

Table 2. K = 10, D = NAW

1. D - the set of features
2. K - the number of clusters
3. F - the function used to normalize various distance metrics
4. S - the distance measure used to measure set similarity
5. W - the weights applied to each feature

The possible features are any non-empty combination of the following subsets:

- N (*Numeric* features)
- A (*Attribute* descriptive features (attributes and categories))
- W (*Word* descriptive features (key words and top words))
- T (*Temporal* descriptive features (open hours and total hours))

5.1. Chains Data Set

Our first three experiments revolve around the **Chains** data set. In these experiments, we will always keep the weight for each feature constant at 1. Because of time constraints, temporal features were not included in the these experiments. Partial results for temporal features can be found in Appendix A.

The full results for all three experiments can be seen in Appendix B.

5.1.1. Normalization and Set Distance

In our first experiment, we see how the different normalization methods and set distance metrics perform. We keep the number of clusters and the feature set fixed at *10* and *NAW* (*Numeric*, *Attribute*, and *Word*) respectively.

The results are shown in Table 2.

We observe that the best performance is achieved when we use *Dice* similarity and *logistic* normalization.

Features	Rand Index
A	0.90678
W	0.85195
N	0.64897
AW	**0.92933**
WN	0.69408
AN	0.72626
NAW	0.82629

Table 3. K = 10, F = logistic, S = Dice

k	Rand Index
8	0.78670
10	**0.82629**
12	0.77748
14	0.76465
16	0.74801
18	0.74580

Table 4. D = NAW, F = logistic, S = Jaccard

5.1.2. Feature Sets

Next, we examine which set of features perform the best. We keep the number of clusters, normalization method, and set distance metric constant at *10*, *logistic*, and *Dice* respectively.

The results are shown in Table 3.

We observe that the combination of *Attribute* and *Word* features gives the best performance, followed by *Attribute* alone. We also observe that numeric features lead to deterioration of performance.

5.1.3. Number of Clusters

In our third experiment, examine how the number of clusters effect performance. We keep the feature set, normalization method, and set distance metric constant at *NAW*, *logistic*, and *Jaccard* respectively.

The results are shown in Table 4.

We observe that the best results when using *10* clusters. We also observe that as we increase the number of clusters from 10, the Rand index decreases. This could be due to the good clusters being split into many clusters resulting in decreasing Rand index. Another reason could be that the number of “same clusters” pairs in the gold standard dataset is much higher than the number of “across cluster” pairs. This could result in Rand index preferring fewer and larger clusters over many small clusters.

Feature	Weight
Star Rating	0.5
Total Review Count	0.0
Available Review Count	1.0
Mean Review Length	0.5
Mean Word Length	0.5
Number of Words	0.0
Mean Words per Review	0.5
Total Hours	2.0
Attribute Features	3.5
Category Features	0.5
Top Words	1.0
Key Words	1.0
Open Hours	0.0

Table 5. Weight learning on a subset of the **Chains** data set. Yields a 0.919748 Rand index.

5.2. Weight Learning

The weight for each feature can be in the range of 0 to 4 (inclusive). To limit the search space, we will only consider weights that are multiples of 0.5. Unfortunately we have a non-convex objective function (the Rand index), and therefore cannot use any efficient methods to learn the best performing weights. In addition, an exhaustive grid search would result in a little less than 100 billion (96,889,010,407) evaluations.

To search our weight space, we held all but one weight constant and incremented the remaining weight from 0 to 4. The value yielding the heights Rand index was kept and the next weight was probed. The order the weights were probed was randomized. After all the weights have been probed, the process is repeated until either 30 iterations have been completed or there is no change from the previous iteration.

To significantly speed up weight learning, we only used a random subset of 200 restaurants from the **Chains** data set.

The final weights can be seen in Table 5. In general, we see the numeric features getting zeroed out which is consistent with our experiments on the full **Chains** data set. A surprising weight is that attached to the “Category Features”. While the “Attribute Features” gets the highest weight of 3.5, the “Category Features” almost gets zeroed out at 0.5. The Rand index achieved with this subset of the **Chains** data set was 0.919748.

5.3. Human Annotated

We clustered the **Human Annotated** data set using the weights learned from the previous experiment, *10* clusters, *logistic* normalization, and *Jaccard* set distance. We were able to achieve a Rand index of **0.847605**.

Experiment	Data Set	Max Rand Index
Normalization and Set Distance	Chains	0.82638
Feature Sets	Chains	0.92933
Number of Clusters	Chains	0.82629
Weight Learning	Chains (200 Random)	0.919748
Human Annotated	Human Annotated	0.847605

Table 6. Experiment Summary

Although not as high as the 0.92933 achieved using the **Chains** dataset or the 0.919748 achieved in weight learning, this is still a noteable achievement on real-life data. Approximatley 85% of our pairing decisions were correct on this realalistic data.

6. Conclusion

Clustering works fairly well for this dataset. The signals that most strongly indicate that two businesses are the same come from the reviews rather than the more structured data like number of stars. Our work with the textual features was limited to just a few instances of "low-hanging fruit", but if the success of those features is any indication of the richness of the reviews, then the reviews should be the focus of future work.

Appendices

A. Temporal Feature Experiments

Time constraints lead us to not being able to complete the experiments with *Temporal* features. However, we do have partial results.

It does not look like *Temporal* features are stronger than *Attribute* or *Word* features. However, we see *Temporal* features performing better than most combinations which include *Numeric* features. We already know that the *Numeric* features are detrimental, but it is very interesting that *Temporal* beats it when we consider what is represented by each feature. *Numeric* features contain information such as the restaurants star rating, but *Temporal* features just have information about when a place is open.

Feature Set	K	Scalar Normalization	Set Distance	Rand Index
NAWT	8	Log	Dice	0.69343
NAWT	8	Log	Jaccard	0.68537
NAWT	8	Logistic	Dice	0.77411
NAWT	8	Logistic	Jaccard	0.76430
NAWT	10	Log	Dice	0.69944
NAWT	10	Log	Jaccard	0.69522
NAWT	10	Logistic	Dice	0.76119
NAWT	10	Logistic	Jaccard	0.74036
NAWT	12	Log	Dice	0.69350
NAWT	12	Log	Jaccard	0.69780
NAWT	**12**	**Logistic**	**Dice**	**0.78736**
NAWT	12	Logistic	Jaccard	0.74933
NAWT	14	Log	Dice	0.70399
NAWT	14	Log	Jaccard	0.70759
NAWT	14	Logistic	Dice	0.76529
NAWT	14	Logistic	Jaccard	0.74732
NAWT	16	Logistic	Jaccard	0.74728
NAWT	18	Logistic	Jaccard	0.74111
AWT	8	N/A	Jaccard	0.70939
AWT	10	N/A	Jaccard	0.71785
AWT	12	N/A	Jaccard	0.71411
NAT	8	Logistic	Jaccard	0.72766
NAT	10	Logistic	Jaccard	0.74192
NAT	12	Logistic	Jaccard	0.74967
NAT	14	Logistic	Jaccard	0.74361
NAT	16	Logistic	Jaccard	0.76003
NAT	18	Logistic	Jaccard	0.76059
NWT	8	Logistic	Jaccard	0.70685
NWT	10	Logistic	Jaccard	0.70269
NWT	12	Logistic	Jaccard	0.69411
NWT	14	Logistic	Jaccard	0.69536
NWT	16	Logistic	Jaccard	0.71855
NWT	18	Logistic	Jaccard	0.71698
AT	8	N/A	Jaccard	0.72635
AT	10	N/A	Jaccard	0.73351
AT	12	N/A	Jaccard	0.73485
AT	14	N/A	Jaccard	0.72862
AT	16	N/A	Jaccard	0.75446
AT	18	N/A	Jaccard	0.77445
NT	8	Logistic	Jaccard	0.71403
NT	10	Logistic	Jaccard	0.69903
NT	12	Logistic	Jaccard	0.69887
NT	14	Logistic	Jaccard	0.70064
NT	16	Logistic	Jaccard	0.69405
NT	18	Logistic	Jaccard	0.68772
WT	8	N/A	Jaccard	0.72013
WT	10	N/A	Jaccard	0.71753
WT	12	N/A	Jaccard	0.75262
WT	14	N/A	Jaccard	0.75366
WT	16	N/A	Jaccard	0.78053
WT	18	N/A	Jaccard	0.76271
T	8	N/A	Jaccard	0.77966
T	10	N/A	Jaccard	0.77932
T	12	N/A	Jaccard	0.75952
T	14	N/A	Jaccard	0.75414
T	16	N/A	Jaccard	0.77200
T	18	N/A	Jaccard	0.77208

continued...

B. Full Results

Below are the full results from our experiments on different parameters using the **Chains** data set. The "Feature Set" column describes the features that were included in that run.

- **N** - Numeric Features
- **A** - "Attribute" Descriptive Features (attributes and categories)
- **W** - "Word" Descriptive Features (key words and top words)

Note that scalar normalizations only make sense in the presence of numeric features ('N'), while set distance only makes sense in the presence of non-numeric features ('A' & 'W').

YUM-Clust

Feature Set	K	Scalar Normalization	Set Distance	Rand Index
NAW	8	Log	Dice	0.65957
NAW	8	Log	Jaccard	0.66084
NAW	8	Logistic	Dice	0.78673
NAW	8	Logistic	Jaccard	0.79006
NAW	8	None	Dice	0.65833
NAW	8	None	Jaccard	0.65833
NAW	10	Log	Dice	0.68435
NAW	10	Log	Jaccard	0.67145
NAW	10	Logistic	Dice	0.82638
NAW	10	Logistic	Jaccard	0.78539
NAW	10	None	Dice	0.66708
NAW	10	None	Jaccard	0.66708
NAW	12	Log	Dice	0.67951
NAW	12	Log	Jaccard	0.67716
NAW	12	Logistic	Dice	0.77756
NAW	12	Logistic	Jaccard	0.76260
NAW	12	None	Dice	0.66905
NAW	12	None	Jaccard	0.66905
NAW	14	Log	Dice	0.68036
NAW	14	Log	Jaccard	0.67331
NAW	14	Logistic	Dice	0.76466
NAW	14	Logistic	Jaccard	0.75734
NAW	14	None	Dice	0.66945
NAW	14	None	Jaccard	0.66945
NAW	16	Log	Dice	0.67522
NAW	16	Log	Jaccard	0.67415
NAW	16	Logistic	Dice	0.74807
NAW	16	Logistic	Jaccard	0.76035
NAW	16	None	Dice	0.67003
NAW	16	None	Jaccard	0.67014
NAW	18	Log	Dice	0.67559
NAW	18	Log	Jaccard	0.68582
NAW	18	Logistic	Dice	0.74581
NAW	18	Logistic	Jaccard	0.75711
NAW	18	None	Dice	0.66871
NAW	18	None	Jaccard	0.66868
AW	8	N/A	Dice	0.91137
AW	8	N/A	Jaccard	0.86178
AW	10	N/A	Dice	0.92939
AW	10	N/A	Jaccard	0.85487
AW	12	N/A	Dice	0.87015
AW	12	N/A	Jaccard	0.86371
AW	14	N/A	Dice	0.85103
AW	14	N/A	Jaccard	0.84191
AW	16	N/A	Dice	0.84990
AW	16	N/A	Jaccard	0.84398
AW	18	N/A	Dice	0.84842
AW	18	N/A	Jaccard	0.85065
NA	8	Log	Dice	0.65825
NA	8	Log	Jaccard	0.65817
NA	8	Logistic	Dice	0.74196
NA	8	Logistic	Jaccard	0.77038
NA	8	None	Dice	0.65829
NA	8	None	Jaccard	0.65829
NA	10	Log	Dice	0.66995
NA	10	Log	Jaccard	0.67036
NA	10	Logistic	Dice	0.72632
NA	10	Logistic	Jaccard	0.73711
NA	10	None	Dice	0.66708
NA	10	None	Jaccard	0.66708
NA	12	Log	Dice	0.67103
NA	12	Log	Jaccard	0.67230
NA	12	Logistic	Dice	0.72265
NA	12	Logistic	Jaccard	0.72928
NA	12	None	Dice	0.66905
NA	12	None	Jaccard	0.66905
NA	14	Log	Dice	0.66721
NA	14	Log	Jaccard	0.66842
NA	14	Logistic	Dice	0.71373
NA	14	Logistic	Jaccard	0.72141
NA	14	None	Dice	0.66940
NA	14	None	Jaccard	0.66946
NA	16	Log	Dice	0.67745
NA	16	Log	Jaccard	0.67554
NA	16	Logistic	Dice	0.71227
NA	16	Logistic	Jaccard	0.72539
NA	16	None	Dice	0.66978
NA	16	None	Jaccard	0.66984
NA	18	Log	Dice	0.67734
NA	18	Log	Jaccard	0.67551
NA	18	Logistic	Dice	0.70830
NA	18	Logistic	Jaccard	0.71912
NA	18	None	Dice	0.66858
NA	18	None	Jaccard	0.66864
NW	8	Log	Dice	0.66360
NW	8	Log	Jaccard	0.65733
NW	8	Logistic	Dice	0.68185
NW	8	Logistic	Jaccard	0.64403
NW	8	None	Dice	0.65832
NW	8	None	Jaccard	0.65826
NW	10	Log	Dice	0.66894
NW	10	Log	Jaccard	0.68336
NW	10	Logistic	Dice	0.69411
NW	10	Logistic	Jaccard	0.66849
NW	10	None	Dice	0.66697
NW	10	None	Jaccard	0.66697
NW	12	Log	Dice	0.67041
NW	12	Log	Jaccard	0.68355
NW	12	Logistic	Dice	0.69345
NW	12	Logistic	Jaccard	0.67378
NW	12	None	Dice	0.66904
NW	12	None	Jaccard	0.66903
NW	14	Log	Dice	0.67002
NW	14	Log	Jaccard	0.67721
NW	14	Logistic	Dice	0.69369
NW	14	Logistic	Jaccard	0.67517
NW	14	None	Dice	0.66992
NW	14	None	Jaccard	0.66980
NW	16	Log	Dice	0.67170
NW	16	Log	Jaccard	0.67573
NW	16	Logistic	Dice	0.69141
NW	16	Logistic	Jaccard	0.68065
NW	16	None	Dice	0.67011
NW	16	None	Jaccard	0.66993
NW	18	Log	Dice	0.67385
NW	18	Log	Jaccard	0.67157
NW	18	Logistic	Dice	0.69822
NW	18	Logistic	Jaccard	0.68097
NW	18	None	Dice	0.66816
NW	18	None	Jaccard	0.66839
A	8	N/A	Dice	0.83220
A	8	N/A	Jaccard	0.96855
A	10	N/A	Dice	0.90687
A	**10**	**N/A**	**Jaccard**	**0.96930**
A	12	N/A	Dice	0.89807
A	12	N/A	Jaccard	0.91043
A	14	N/A	Dice	0.86913
A	14	N/A	Jaccard	0.91139
A	16	N/A	Dice	0.82648
A	16	N/A	Jaccard	0.90733
A	18	N/A	Dice	0.82387
A	18	N/A	Jaccard	0.88869
N	8	Log	N/A	0.65684
N	8	Logistic	N/A	0.64323
N	8	None	N/A	0.65829
N	10	Log	N/A	0.65649
N	10	Logistic	N/A	0.64903
N	10	None	N/A	0.66707
N	12	Log	N/A	0.66913

continued...

Feature Set	K	Scalar Normalization	Set Distance	Rand Index
N	12	Logistic	N/A	0.66490
N	12	None	N/A	0.66903
N	14	Log	N/A	0.67289
N	14	Logistic	N/A	0.66102
N	14	None	N/A	0.66969
N	16	Log	N/A	0.66826
N	16	Logistic	N/A	0.66552
N	16	None	N/A	0.66995
N	18	Log	N/A	0.66876
N	18	Logistic	N/A	0.66025
N	18	None	N/A	0.66835
W	8	N/A	Dice	0.83930
W	8	N/A	Jaccard	0.81669
W	10	N/A	Dice	0.85203
W	10	N/A	Jaccard	0.81611
W	12	N/A	Dice	0.85990
W	12	N/A	Jaccard	0.81462
W	14	N/A	Dice	0.85966
W	14	N/A	Jaccard	0.81438
W	16	N/A	Dice	0.81891
W	16	N/A	Jaccard	0.79775
W	18	N/A	Dice	0.77338
W	18	N/A	Jaccard	0.76877