Evaluating the effect of distance functions on mining geo-location datasets

Rasaq Otunba and Amarachi Ekedebe
George Mason University
Fairfax, VA 22030, USA
{rotunba, aekedebe}@masonlive.gmu.edu

Abstract

Distance functions are at the core of widely used machine learning or data mining algorithms. Geo-location datasets are subjects of a wide range of data mining tasks. Often, the overall accuracy of the data mining algorithm used can be improved significantly if the inherent distance metric is carefully selected. K-NN is one of the simplest classification algorithms used in data mining. K-NN searches a dataset for the most similar elements to a given query element, with similarity defined by a distance function. This work provides a discussion and presents empirical evidence of how the distance function of K-NN influences its performance. Seven distance functions were evaluated: three popular distance measures and four other distance measures suitable for geo-location dataset domain.

1 Introduction

With current data proliferation, especially in this age of social networks, data mining is gaining increasing importance and advancement daily. Data mining is the extraction of useful novel patterns and relationships from data sources. There are a number of data mining algorithms used for different data mining tasks and suited for different datasets. These data mining algorithms often depend on certain parameters to perform properly. One such parameter is the inherent distance function of the algorithm. Distance functions are used in algorithms such as K-means and K-NN. A simple algorithm was decided upon for this experiment, since the focus is on the inherent distance function. This experiment uses K-NN (K-nearest neighbors) for evaluation. K-NN is a simple machine learning algorithm. Despite its simplicity, it has been applied in a tangible number of domains.

The geo-location domain is one of the areas where K-NN has been, and will continue to be, applied. Geo-location datasets would typically have geographic attributes such as longitude and latitude. The distance functions of data mining algorithms compute similarity of elements in the dataset to a given element using these attributes. Much research has been done on distance functions particularly with respect to geo-coded distances in longitudes and latitudes. These research efforts have resulted in distance functions with considerable accuracy. This experiment evaluates the effect of using different distance functions in mining algorithm on a geo-location dataset. Seven distance functions were evaluated. Four of the functions are calculations on the basis of a spherical earth (ignoring ellipsoidal effects). These include the Haversine distance, Vincenty distance, Spherical Law of Cosines (SLOC) and the equirectangular approximation. The remaining three are popular distance functions used in most data mining algorithms. The three popular distance functions are the Chebyshev, and Manhattan, Euclidean.

The findings in this paper provide evidence that can help data miners know the implications of using distance functions in mining geo-location datasets and mining in general. The paper is organized as follows: Section 2 discusses related work. Sections 3 and 4 describe K-NN and distance functions. Section 5 presents the methodology of the experiment. Section 6 shows the results and Section 7 analyzes and discusses them. Section 8 provides conclusions and ideas for future work.

2 Related Work

Data mining algorithms and distance functions have been studied individually and in relation to one another. Studies have also been conducted on the effect of algorithm parameter tuning [1]. It is well known in the Knowledge Discovery and Data mining (KDD) community that the best selection of parameters for an algorithm should be dependent on the dataset. It is interesting to know that, despite the awareness of this commonplace knowledge, parameter tuning is still not commonly done. This is partly due to the expensive experiment that would result from the exponential growth of parameters combinations on a large number of datasets. This is another study on parameter tuning with focus on distance function as the parameter and geo-location dataset. However, this is the first study that explores popular distance functions and distance functions particularly suited for mining geo-location datasets with geographic coordinate attributes.

3 K-NN

This experiment uses K-NN as a classification algorithm i.e., instances in a training dataset have class labels, and K-NN labels a new instances in a test set with a class from the training set. The labeling is done based on votes from the most similar instances in the training set. Similarity is computed with a distance function. The most frequent class among k closest instances in the training set is assigned to an unlabelled instance in the test set. K is usually selected to maximize performance in the test set. This same approach was followed in this experiment. K-NN is referred to as the nearest neighbor (NN) algorithm when k=1.

4 Distance functions

The seven distance functions used in this experiment are discussed in this section. The section begins with the three popular functions and end with the remaining four. A section on Minkowski distance functions is included as additional information.

4.1 Chebyshev distance function

This is sometimes referred to as the Tchebychev distance, Maximum metric, or L∞ metric. It is defined over a vector space with the distance between two vectors being the greatest of their differences along any coordinate dimension. With respect to its application as a distance function between two instances in a dataset, it is the greatest of their differences along any attribute.

For any two instance i and j, the Chebyshev distance d_{ij} between them is the maximum difference of all their individual k attributes. The formula is shown below in equation 1:

$$\boxed{d_{ij} = \max_k |x_{ik} - x_{jk}|} \quad (1)$$

4.2 Manhattan distance function

This is sometimes referred to as the taxicab metric, rectilinear distance, L1 distance or L-norm, city block distance. It is defined over a vector space with the distance between two vectors being the sum of the absolute differences of their coordinates. With respect to its application as a distance function between two instances in a dataset, it is the sum of the absolute differences of all attributes.

For any two instances (p and q), the distance Manhattan distance $d_1(p, q)$ between them is the sum of the absolute differences of all their individual n attributes. The formula is shown below in equation 2:

$$\boxed{d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|} \quad (2)$$

4.3 Minkowski distance function

This is a metric defined on the Euclidean space. It is however used as distance function on instances in a dataset. The attributes of an instance in a dataset can be mapped to coordinates in the Euclidean space. For any two instances P and Q defined as shown in figure 1 below:

$$P = (x_1, x_2, \ldots, x_n) \text{ and } Q = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$$

Figure 1

The distance between P and Q is defined in figure 2 below:

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

Figure 2

Minkowski distance is typically used with p being 1 or 2. The latter is the Euclidean distance, while the former is sometimes known as the Manhattan distance. In the limiting case of p reaching infinity we obtain the Chebyshev distance [4].

4.4 Euclidean distance function

This is a special case of the Minkowski distance described in section 4.3 with p being equal to 1.

4.5 Haversine distance function

This can be used to calculate the great-circle (shortest) distance over the earth's surface – giving an 'as-the-crow-flies' distance [3]. The formula is shown in figure 3 below:

$a = \sin^2(\Delta lat/2) + \cos(lat_1).\cos(lat_2).\sin^2(\Delta long/2)$
$c = 2.\text{atan2}(\sqrt{a}, \sqrt{(1-a)})$
$d = R.c$

Where R is earth's radius (mean radius = 6,371km), lat is latitude, long is longitude and d is the distance;
angles need to be in radians

Figure 3

4.6 Spherical law of Cosines

This can be used to calculate the great-circle distance between two points to accuracy of approximately 1 meter [3]. See formula in figure 4 below:

$d = \text{acos}(\sin(lat_1).\sin(lat_2)+\cos(lat_1).\cos(lat_2).\cos(long_2-long_1)).R$

Where R is earth's radius (mean radius = 6,371km), lat is latitude, long is longitude and d is the distance;
angles need to be in radians

Figure 4

4.7 Vincenty distance function

This can be used to calculate the great-circle distance between two points to accuracy of approximately 1 millimeter [3]. Space precludes the inclusion of the detailed formulae here. Such details can be found here: http://www.movable-type.co.uk/scripts/latlong-vincenty.html.

4.8 Equirectangular approximation

This is an equirectangular projection of Pythagoras' theorem. It is expected to perform better but less accurate compared other great-circle distance functions. It is good for small distances. The formula is shown in figure 5 below:

$x = \Delta lon.\cos(lat)$
$y = \Delta lat$
$d = R.\sqrt{x^2 + y^2}$

Where R is earth's radius (mean radius = 6,371km), lat is latitude, long is longitude and d is the distance;
angles need to be in radians

Figure 5

5 Methodology

The fundamental premise of the data mining task adopted in this experiment is that close (similar) locations with respect to distance belong to the same city. The geo-location dataset used in this experiment has three attributes, longitude, latitude and city where the city is the class label. Locations from parts of the United States of America are contained in the dataset. The dataset has 1534 unique instances. The data mining task performed on the dataset is to classify unlabelled instances i.e. classify unknown instances based on their longitude and latitude attributes into cities. K-NN is used as the classification algorithm. Over 90% of instances in the dataset have at least two similar instances to enable K-NN perform better than random guessing i.e. K-NN should be perform with less than 50% misclassification error rate. This is because K-NN uses similarity to neighboring instances for classification. An extensible KDD tool, Weka was used to perform the experiment. Weka was extended to include all seven distance functions for experiment. Weka originally has the three popular distance functions stated in the previous sections. F-Measure and misclassification error rate are used as evaluation measures. Misclassification error rate is a classic performance measure but would be inadequate. F-measure is used as a complementary measure for cost of misclassification. K is K-NN parameter indicating the number of neighbors to be used for classification. K was selected to maximize test set accuracy. K, when set to 1 produced best performance. We therefore used the NN algorithm. Ten-fold cross-validation and one-third holdout to partition the datasets into training and test sets respectively. All the artifacts for this experiment including Java Programs, dataset and results can be found at this web address: http://rasaq-random-codes.googlecode.com/svn/trunk/cs750project/.

6 Results

This section shows the results of the experiment. Figure 6 shows the values obtained 10-fold cross validation. Figure 7 is a chart with a clearer picture of the results obtained for the different distance functions with 10-fold cross validation. Figure 8 is a table of the results obtained when one-third of the dataset is held out as test set. Figure is a chart of the results obtained with one-third holdout. All the experiments were performed with default parameters in Weka and kept constant across all runs.

	F-Measure	Error rate (%)
Manhattan	8.7353	0.911
Euclidean	8.8657	0.911
Chebyshev	9.3220	0.906
Haversine	10.3651	0.896
Equirectangular Approximation	10.3651	0.896
Vincenty	10.3651	0.896
Spherical law of cosines	10.3651	0.896

Figure 6: 10-fold cross validation table

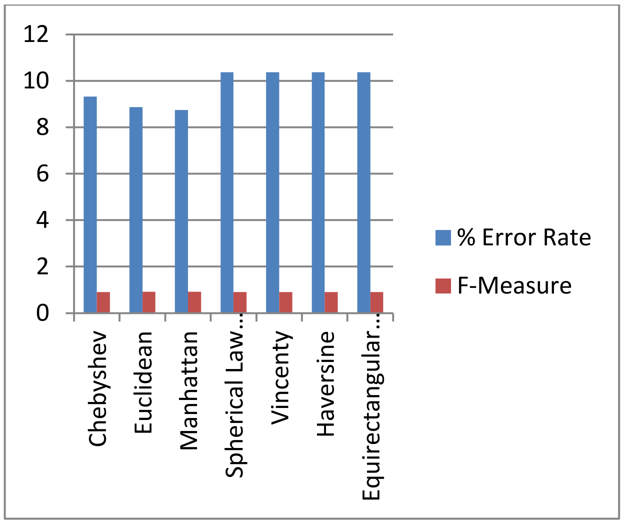

Figure 7: 10-fold cross validation chart

	F-Measure	Error rate (%)
Manhattan	9.5785	0.897
Euclidean	10.7280	0.884
Chebyshev	12.4521	0.867
Haversine	13.2184	0.857
Equirectangular Approximation	13.2184	0.857
Vincenty	13.2184	0.857
Spherical law of cosines	13.2184	0.857

Figure 8: one-third holdout table

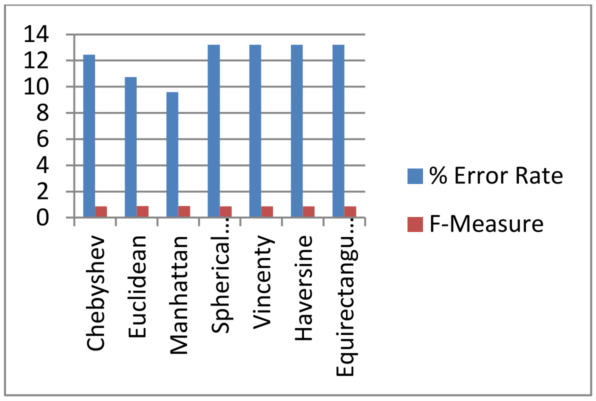

Figure 9: one-third holdout chart

7 Analysis and discussion

As seen from the results in the previous section, Manhattan distance function performed better overall. It has the desired lower error rates and higher F-measure. This should be expected for the dataset used. For the type of dataset, L-norm distances are a natural choice (as they reflect what "similarity" means geographically), and it makes sense that they perform the best. Although the four distance functions specifically suited for great–circle distances, the areas contained in the datasets (part of Unites States of America) are relatively small compared to the surface of the earth. The amount of curvature might not be significant enough to reap the benefits of these specific distance functions.

The Euclidean performed second to the Manhattan distance function. Chebyshev performed better than all the four great-circle distance functions. The four-great circle distance functions have equal performance so this implies they have similar accuracy.

A threat to construct validity is the assumption made about similar places being in the same city. Locations along separate city borders could be very similar. The effect of this is not expected to be colossal and would be uniform across all distance functions.

8 Conclusion and Future Work

It is evident from this experiment that due consideration should be given to parameter tuning in experiments. More specifically, Manhattan distance function is more appropriate for distance computations on geographic coordinates. This is especially the case when the locations being considered are pretty close. Domain knowledge is desirable for most experiments, but it is interesting to find that generic parameters might perform better than domain specific parameters as seen in this experiment.

For future work, it would be expedient to expand the investigation on distance measures and other parameters of algorithms used in KDD. More datasets, distance functions and parameters could be explored e.g., investigating the appropriate distance functions for text data and multimedia data. Experiments on generic solutions to problems posed by parameter tuning are also worthy of consideration.

References

[1] BATISTA, G. E. A. P. A., Silva, D. F. How k-Nearest Neighbor Parameters Affect its Performance, Proceedings of the Argentine Symposium on Artificial Intelligence, 2009. Pg 1 - 12.

[2] Charu C. Aggarwal. IBM T. J. Watson Research Center. charu@us.ibm.com, 2003.

[3] Movable type scripts. Calculate distance, bearing and more between Latitude/Longitude points. Last accessed: May 5, 2012.

[4] Minkowski distance. http://en.wikipedia.org/wiki/Minkowski_distance. Last accessed: May 6, 2012.

[5] Chebyshev distance. http://en.wikipedia.org/wiki/Chebyshev_distance. Last accessed: May 6, 2012.

[6] Taxicab geometry. http://en.wikipedia.org/wiki/Taxicab_geometry. Last accessed: May 6, 2012.

[7] Pang-Ning Tan, Michael Steinbach, Vipin Kumar. Introduction to Data Mining. Addison-Wesley, 2005. ISBN: 0321321367.