Term Paper Data Science 1

Docent: Prof. Dr. Lena Wiese
Semester: Summer Term 2021

Institute of Computer Science
Goethe-Universität Frankfurt a. M.

Authors: Franziska Hicking
your Student ID
your.email@ddre.ss
branch of study (Bachelor/Master, semester count)
Jonas Elpelt
6673181
elpelt@stud.uni-frankfurt.de
Master Bioinformatics, 2nd semester
Julian Rummel
6673334
s9594673@stud.uni-frankfurt.de
Master Bioinformatics, 2
Niklas Conen
6599913
conen@stud.uni-frankfurt.de
branch of study (Bachelor Computer Science, 8)

Date: June 1, 2021

Chosen Project Topic:
T4 - DISTANCE MEASURES AND CLUSTERING

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet,

Contents

1 Definition of Distance Measure

A distance measure is a function $d(x, y)$ that calculates a real value between two points in a space, containing two sets of points. If $d(x, y)$ satisfies the following three axioms the distance measure is classified as a *metric*:

$$d(x, y) = 0 \Leftrightarrow x = y \qquad \text{Identity of indiscernibles} \tag{1.1}$$
$$d(x, y) = d(y, x) \qquad \text{Symmetry} \tag{1.2}$$
$$d(x, y) \leq d(x, z) + d(z, y) \qquad \text{Triangle inequality} \tag{1.3}$$

The triangle-inequality imposes the condition that a distance reflects the shortest path between two points. Thus, it is not possible to achieve a distance improvement by traveling via an intermediate point z. [1]

Moreover all axioms enforce non negative distances as an additional condition.

$$d(x, y) \geq 0 \qquad \text{Non Negativity} \tag{1.4}$$

2 Different Distance Measurements

2.1 Euclidean Distances

The Euclidean distance is part of the L_p-*metrics* which are defined as

$$d(x, y) = \sum_{i=1}^{n} (|x_i - y_i|^p)^{\frac{1}{p}} \tag{2}$$

Setting $p = 2$ expresses the Euclidean distance, which is defined as the positive square root of the sum of all squared distances in each dimension:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (|x_i - y_i|)^2} \tag{3}$$

The first two axioms defined in section 1 are easily shown to apply:

1. Identity of indiscernibles:
   For $x = y$ the value is obviously 0. Let $x = y$, then $(|x - y|)^2 = 0$ and $\sqrt{0} = 0$.
2. Symmetry:
   Symmetry is cleary given by the square of each distance.
   $(x - y)^2 = (y - x)^2$.

Non negativity is also shown quite easily. The square of any real number is always positive and the squareroot of any real positive number is always positive. Hence $d(x, y) \geq 0$.

The traingle inequality requires a more difficult proof. However, to keep it simple, the Euclidean space possesses the property that the sum of the lengths of Cathetus and Ancathetus is always longer than the length of the Hypothenuse. [1]

2.2 Angular cosine Distance

The angular cosine distance gives the (normalized) angle between two points x and y represented as vectors in an n-dimensional space. It does not make a difference between a vector and a multiple of that vector. The cosine distance can be calculated by applying the arc-cosine function to the cosine of the angle θ between x and y [1].
It is based on the cosine similarity (cosine between two vectors x and y), which is definied as:

$$\text{cosine similarity} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sum_{i=1}^{n} y_i^2} \tag{4}$$

The cosine similarity, however, is not a distance as it is defined for positive values only. Therefore it has to be converted to the normalized angle between x and y as followed [2]:

$$\text{angular cosine distance} = \frac{\arccos(\text{cosine similarity})}{\pi} \tag{5}$$

Note, that if x or y are zero vectors, the cosine similarity would not be defined. To prevent a division by zero the cosine similarity is set to 1 in this special case (based on the implementation of the pairwise distance in scikit-learn).

The axioms for a distance measure are fulfilled for the cosine distance [1]:

1. Identity of indiscernibles:
   Two vectors can have a cosine distance of 0 if and only if they are located in the same direction. (This applies also to vectors that are multiples of one another and therefore are in the same direction.)
2. Symmetry:
   Symmetry is obviously given by the equality to measure an angle between x and y and an angle between y and x.
3. Triangle inequality:
   A rotation from x to y can be explained by a rotation from x to z and then to y. Therefore a sum of these two rotations is always bigger or equal than the rotation directly from x to y
4. No negative distances:
   Regardless of the dimensionality of the space the values of the cosine distance are between 0 and 180 degrees, therefore no negative distances can occur.

2.3 Chebyshev Distance

The Chebyshev distance (also known as Tschebyscheff distance, Maximum Value distance or L_∞ distance) is the limit of the before mentioned L_p-metrics (equation 2). On a vector space this metric is induced by the Supremum Norm (also called Chebyshev Norm or Infinity Norm), which again is the limit of the L_p-norms.
Descriptively the Chebyshev metric is the greatest distance between two vectors on one axis. Formally it is defined as:

$$d(x, y) = \max(|x_i - y_i|) \tag{6}$$

which is the aforementioned limit of the L_p-metric and is therefore also called L_∞-metric:

$$d(x,y) = \lim_{p\to\infty} \left(\sum_{i=1}^{n} (|x_i - y_i|^p)^{\frac{1}{p}} \right) \tag{7}$$

The three axioms for a metric (section 1) are proven below:

1. For $x = y$ all entries of a vector are identical and all differences between $x_i - y_i$ are 0. Thus: $d(x,x) = \max(|x_i - x_i|) = \max(0) = 0$
2. Symmetry is given because of the symmetry of the absolute value function: $|x_i - y_i| = |y_i - x_i|$
3. The triangle equation can be shown using some estimates:

$$\begin{aligned} \max(|x_i - y_i|) &= \max(|x_i - z_i + z_i - y_i|) \\ &\leq \max(|x_i - z_i| + |z_i - y_i|) \\ &\leq \max(|x_i - z_i|) + \max(|z_i - y_i|) \\ \Rightarrow d(x,y) &\leq d(x,z) + d(z,y) \end{aligned}$$

Non negativity also results from the non negativity of the absolute value function. Therefore the Chebyshev distance is classified as a metric.

3 Data Set Description

3.1 Housevotes

The housevotes dataset, created by Jeff Schlimmer in April 1987, was taken from the UCI Machine Learning Repository [3]. The dataset consists of voting results of the U.S. House of Representatives Congressmen on 16 key votes during the second session of Congress in 1984. The key votes and the voting results are identified by the Congressional Quarterly Almanac (CQA) documenting this session of Congress. The voting results are split into nine different types by the CQA, which are consolidated into three results used in the dataset.

Voted for, paired for, and announced for count as a yes vote. Voted against,

paired against, and announced against count as a no vote. Voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known are denoted as a unknown state.

The set consists of two classes, 267 democrats and 168 republicans.

3.2 Wine recognition dataset

This dataset contains the chemical analysis results of Italian wines from 3 different cultivators. It is also taken from the UCI Machine Learning Repository [3]. The dataset consists of 178 instances, each of them having 13 numeric attributes according to different measurements taken for different constitutuents (alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines, proline). Each instance belongs to either one of three classes containing 59, 71 and 48 data points. It was created by R. A. Fisher in July 1988.

4 Clustering Algorithms

4.1 K-Means

4.2 K-Medoids

The K-Medoids clustering method is related to the well-known K-means algorithm, but uses medoids (representative points for each cluster) instead of means to define new cluster centers, which makes it more robust to outliers. It partitions the dataset by assigning each data point to the closest of k cluster centers, which are defined by the most centrally located medoids. A medoid is a point with a minimal average dissimilarity to all other data points in the same cluster. The most commonly used algorithm to solve this NP-hard problem heuristically is the PAM (Partitoning Around Medoids) algorithm, that works as following:

1. First initialize the algorithm by selecting k data points to be the medoids and assigning every data point to its closest medoid.

2. Compare the average dissimilarity coefficient of a swap of each medoid m and a non-medoid data point $\bar{m}$. Find a swap between m and $\bar{m}$ that would decrease the average dissimilarity coefficient the most.

3. If no change of a medoid happened in the second step, terminate the algorithm, else re-assign the data points to the new medoids and go back to step 2.

4.3 K-Median

4.4 DBSCAN

DBSCAN was developed by Martin Ester, Hans-Peter Kriegel, Jiirg Sander and Xiaowei Xu. All following definitions and descriptions are taken from their original publication [4] or their revisit of DBSCAN [5] and only apply to this algorithm.
Contrary to the aforementioned centroid-based partitioning algorithms (k-means, k-medoids and k-median) the DBSCAN (*Density Based Spatial Clustering of Applications with Noise*) algorithm uses point densities to determine clusters.
To introduce the definition of the density of a cluster, first the Eps-neighbourhood of a point is defined:

Definition 1: *Eps-neighbourhood*
A point q is part of the Eps-neighbourhood N_{Eps} of point p if the distance between them is smaller than a threshold distance called Eps.
The Eps-neighbourhood therefore is defined as $N_{Eps} = \{q \in D \mid ||p, q|| \leq Eps\}$ with D denoting the entirety of points that are supposed to be clustered and $||p, q||$ being the distance between p and q for an arbitrary distance measure.

The Eps-neighbourhood fails at being a reliable measure for the point density if a point is located at the border of a cluster. These points are called *border point*. Points that are located on the inside of a cluster are called *core points*. Hence the following definition is made:

Definition 2: *directly density-reachable and density-reachable*
A point p is directly density-reachable from a point q when

1. $p \in N_{Eps}(q)$

2. $|N_{Eps}(q)| \geq$ MinPts

with MinPts being the minimal number of points that $N_{eps}(q)$ should contain so that q is considered a core point of a cluster.
A point is *density-reachable* if there is a chain of points between p and q so that all neighbouring points in the chain are directly density reachable.

To complete the definition of what is considered part of a cluster density-connectivity is defined:

Definition 3: *density-connected*
Two points p and q are considered density-connected if there is a common point o which is density-reachable from p and q.

Now a cluster can be described as:

Definition 4: *cluster*
A cluster is a non empty subset $C \in D$ so that:

1. $\forall p, q : p \in C \wedge q$ is density reachable from $p \Rightarrow q \in C$
2. $\forall p, q \in C :$ p is density-connected to q

Noise is easily defined as every point that is not part of a Cluster C_i.

Using these definitions DBSCAN can begin the clustering process with given values for Eps and MinPts. In the beginning all points are not labeled. Beginning with an arbitrary point p all points are iterated in a linear fashion. For each point a `RangeQuery` function is executed finding all density-reachable neighbours of p. If `RangeQuery` finds more than MinPts neighbours then p is a core point and is labeled as such. Otherwise p is marked as Noise.
In the next step every point in the Neighbourhood excluding p is expanded. Unlabled Points get checked for the core point condition (which equals a `RangeQuery` call). Points that got labeled as Noise before are labeled as core points. When the expansion comes to an end a cluster is yielded and the next unlabeled point is chosen as p.

Two clusters may be merged if their distance is below Eps. The distance between two clusters C_1 and C_2 is defined as $||C_1, C_2|| = \min\{||q, p|| \mid p \in C_1, q \in C_2\}$.

The runtime complexity of DBSCAN heavily depends on the runtime of the `RangeQuery` function and the distance measure. Thus the runtime can exceed $\mathcal{O}(n^2)$ depending on the chosen implementations. A detailed discussion of DBSCANs runtime can be found in [5].

5 Additional Methods Used

5.1 k++-Initialiser

5.2 One-Hot-Encoding

Categorical data is represented by specific discrete values or labels. This is case for the housevotes dataset (see section 3.1), where voting results can have one of three values (y, n, ?). The distance measures described in section 2 need numerical data to work. The categorical data therefore needs to be converted (encoded) to numbers which accurately describe their distance to another. Using simple integer encoding where no is encoded as 0, yes is encoded as 1 and the unknown state is encoded as 2 results in a yes vote being classified as closer to the unknown state than a no vote by the distance measures.

For so called non ordinal data (data which has no known order) like the votes, One-Hot-Encoding provides better results when using distance based clustering.

With One-Hot-Encoding every attribute is represented as binary vector. Each element of this vector represents a category value. The corresponding value of a sample is set to 1 in the binary vector. This increases the dimensionality of the problem, but represents an equal distance between every value an attribute can have.

The scikit-learn library [6] was used to implement One-Hot-Encoding on the solar flare dataset.

6 Description of Python libraries used

Libraries:

- numpy
- pyclustering
- seaborn
- sklearn
- sklearn-extra

7 Description of Evaluation Module

ANMERKUNGEN:

- What are the results and how are they measured?

8 Web Frontend and User Manual

ANMERKUNGEN:

- Describe the implementation and write a brief user manual with screenshots.

9 Conclusion

ANMERKUNGEN:

- Summarize the main points and achievements
- Add your own assessment/criticism on the topic

References

[1] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. Definition of a distance measure. In *Mining of Massive Datasets - THIRD EDITION*, page 97. Infolab Stanford EDU, 2020.

[2] Cosine distance, cosine similarity, angular cosine distance, angular cosine similarity, Jul 2017.

[3] Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017.

[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[5] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Trans. Database Syst.*, 42(3), July 2017.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[7] Andrei Novikov. PyClustering: Data Mining Library. *Journal of Open Source Software*, 4(36):1230, apr 2019.

[8] Moshe Lichman. UCI Machine Learning Repository, 2013.

[9] Harro Heuser. Lehrbuch der analysis : Teil 2, 2000.