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
6673525
franziska.hicking@stud.uni-frankfurt.de
Master Bioinformatics, 2
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 5, 2021

Chosen Project Topic:
T4 - DISTANCE MEASURES AND CLUSTERING

Abstract

Clustering algorithms can be important tools during the analysis of datasets. They divide a dataset into groups of items based on a certain measure of similarity such as the distances between each of the items. In this work, we implemented and compared four different clustering algorithms (K-Means, K-Medoids, K-Median, DBSCAN). For this, we selected four distinct datasets as well as multiple distance measures (Manhattan, Euclidean, Angular cosine, Chebyshev). For efficient comparison of the clustering results we made use of multiple clustering indices. Additionally, we implemented a web frontend which provides the ability to run all clustering algorithms with distance measures, datasets and clustering indices chosen by the user. The results will be visualized afterwards. After running all algorithms with each of the datasets respectively and all distance measures where they could be applied, we compared the resulting values of the clustering indices. Our results show that

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*:

$$d(x, y) = 0 \Leftrightarrow x = y \qquad \text{Identity of indiscernibles} \qquad (1.1)$$

$$d(x, y) = d(y, x) \qquad \text{Symmetry} \qquad (1.2)$$

$$d(x, y) \leq d(x, z) + d(z, y) \qquad \text{Triangle inequality} \qquad (1.3)$$

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. [**?**]

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

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

2 Different Distance Measurements

2.1 Manhattan distance

To determine the distance between two items the Manhattan distance, also referred to as taxicab distance, may be used. This distance measure assumes a n-dimensional vector space with a fixed cartesian coordinate system. It is defined as following:

$$d(x, y) = ||x - y||_1 = \sum_{i=1}^{n} |x_i - y_i|$$

where x and y are vectors

$$x = (x_1, x_2, ..., x_n) \text{ and } y = (y_1, y_2, ..., y_n)$$

The Manhattan distance is, like the Euclidean distance, part of the $L_p - metrics$ (see 2.2), where the value for p is set to 1. Assuming a two dimensional space, the distance between two points is the shortest path between them with the restriction of only being able to move vertically and horizontally. Prove of axioms described in section 1:

1. Identity of indiscernibles:
   Let $x = y$, then $|x - y| = 0$ and hence $\sum_{i=1}^{n} |x_i - y_i| = 0$

2. Symmetry:
   This axiom is fulfilled since $|x - y| = |y - x|$ for any x and y , which implies that $\sum_{i=1}^{n} |x_i - y_i| = \sum_{i=1}^{n} |y_i - x_i|$.

2.2 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. [**?**]

2.3 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 [**?**].
It is based on the cosine similarity (cosine between two vectors), 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 [**?**]:

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

1. 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. 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.)

3. Symmetry:
   Symmetry is obviously given by the equality to measure an angle between x and y and an angle between y and x.

4. 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 [**?**].

2.4 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 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.

3 Data Set Description

3.1 Solar Flare

The solar flare dataset is taken from the UCI Machine Learning Repository [**?**]. Each point contains data recorded for on active region of the sun. The first three attributes are the McIntosh classification of sunspot groups:

1. Z-value: modified Zurich sunspot class, $\{A, B, C, D, E, F, H\}$
2. p-value: description of the penumbra of the largest spot. A penumbra is a part of a sunspot that is darker than the suns surface. $\{x, r, s, a, h, k\}$
3. c-value: description of the distribution of sunspots in a group $\{x, o, i, c\}$

A detailed descriptions of the letter codes can be found in the original paper by McIntosh [**?**]. The following entries are as follows:

4. Activity (1: reduced, 2: unchanged)
5. Evolution (1: decay, 2: no growth, 3: growth)
6. Code for the previous 24h flare activity
7. Is the region historically complex? (1: yes, 2: no)
8. Todoooooo
9. Area (1: small, 2: large)
10. Area of the largest spot (1: ≥ 5 deg^2, 2: > 5 deg^2)

The last three entries are a predicted flare classes

3.2 Wine recognition dataset

This dataset contains the chemical analysis results of Italian wines from 3 different cultivators [**?**]. 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.

3.3 Iris dataset

The Iris Dataset, introduced by Ronald Fisher in 1936, contains the petal and sepal measurements of three different species of Iris [**?**]. The considered Irises are Setosa, Versicolour and Virginica. For each Iris, 50 samples are included and for each sample the dataset contains the sepal length, sepal width, petal length and petal width in cm.

4 Clustering Algorithms

4.1 K-Means

The K-means algorithm aims to group together similar items of a given dataset into clusters. The total number of clusters is predefined and represented as the value for k. All considered items can be referred to as points, as this clustering algorithm assumes an Euclidean space. Hence, only distance measures which assume an Eulicdean space, such as the Manhattan distance or the Euclidean distance, are sensibly appicable. The K-means algorithm belongs to the point-assignment algorithms in clustering, as all points are considered successively and assigned to the most fitting cluster. The algorithms operates in the following steps:

1. Initially, the algorithm picks k points whose positions each represent one cluster centroid
2. All points are considered in turn:
    - Find the nearest centroid/mean of the considered point (Euclidean distance measure)
    - Assign point to cluster of that centroid
    - Adapt position of this centroid
3. (optional) fix all centroids and reassign all points with the inclusion of the initial k points

The essential first step of initializing the clusters requires k points that have a high chance of being in separate clusters. This can be achieved by different approaches. One possible approach consists of picking points which are as far

away as possible from each other. This can be achieved by the conducting the following steps:

1. A random point is picked as the first of k cluster centorids
2. For k-1 passes:
   Pick the point whose minimum distance is the largest considering all previously chosen points

After the K-means algorithm assigned all points, an optional step af reassigning the points with fixed centroids can be conducted. This can be sensible since it is possible that after a point has been assigned to cluster the centroids move so far that the point would now belong to a different cluster.

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 [**?**] or their revisit of DBSCAN [**?**] 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 Description of Python libraries used

Libraries:

- numpy
- pyclustering
- seaborn
- sklearn

6 Description of Evaluation Module

ANMERKUNGEN:

- What are the results and how are they measured?

7 Web Frontend and User Manual

ANMERKUNGEN:

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

8 Conclusion

ANMERKUNGEN:

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

References