Online Test 4: Clustering and Outlier Detection

Test und Assessment – Druckansicht

Online Test 4: Clustering and Outlier Detection

Datum: Sun Mar 13 14:14:36 2022 Maximale Punktezahl: 18

Frage 1 - Clustering Algorithms - Categories (1 Punkt) [ID: 1051807]

Match the clustering algorithms to their corresponding category!

Partitioning	passt zu	CLARANS	*(0.25 Punkte)*
Hierarchical	passt zu	DIANA	*(0.25 Punkte)*
Density-based	passt zu	DBSCAN	*(0.25 Punkte)*
Probabilistic	passt zu	Mixture models	*(0.25 Punkte)*

See the lecture.

Frage 2 - Clustering Algorithms - Complexity (1 Punkt) [ID: 1078967]

Order the following clustering algorithms according to their runtime complexity regarding the number of data objects n! Start with the lowest complexity, i.e., the fastest algorithm.

k-means

DBSCAN (with spatial index structure)

PAM

agglomerative hierarchical (naive implementation)

divisive hierarchical (naive implementation)

k-means only depends linearly on n. The number of iterations is an input parameter and thus independent from n, though you can't tell beforehand how well the algorithm will have converged after a fixed number of iterations.

DBSCAN has a complexity of $O(n \cdot \log n)$ with a spatial index structure to speed up neighborhood queries. Without such an index structure, the complexity would be $O(n^2)$. Each point is processed once. To process a point, the density of its neighborhood needs to be checked (does the neighborhood contain at least $MinPts$ points or not?).

PAM is quadratic in n if $k \ll n$ (else, it's rather $O(n^3)$). In each iteration of the outermost loop, it tests each non-medoid object as a medoid. For each of these tests, it needs to loop over all non-medoid objects again. These nested loops yield a quadratic complexity. Similar to k-means, the number of iterations of the outermost loop is an input parameter, so it doesn't count towards the complexity estimate here.

Agglomerative hierarchical clustering does $n - 1$ iterations of the outermost loop. In each of these iterations, it needs to decide which two clusters to merge by finding the most similar current clusters. As there

are up to $O(n)$ clusters, the cluster-wise distance matrix has a size of $O(n^2)$. (Also, $O(n)$ distances need to be updated to remove information from the old, now removed, clusters and add the new cluster.) This yields a total complexity of $O(n^3)$. For special linkage criteria, there are customized implementations with an overall complexity of $O(n^2)$. Even for an arbitrary linkage criterion, one can save time with an efficient data structure like a heap, reducing the effort per iteration to $O(n \cdot \log n)$, yielding an overall complexity of $O(n^2 \cdot \log n)$.

Divise hierarchical clustering with a naive implementation needs to check all possible splits of existing clusters. Starting with one cluster, there are $2^{n-1} - 1$ splitting options just in the first iteration.

Frage 3 - Clustering Algorithms - Custom Distance Measure (1 Punkt) [ID: 1078963]

Which of the following clustering algorithms allow using an arbitrary distance measure between points, i.e., they are not tied to the Euclidean distance or some other specific notion of distance?

- [x] DBSCAN *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- [ ] Gaussian mixture models *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- [x] agglomerative hierarchical *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- [ ] k-means *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- [x] PAM *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*

DBSCAN, PAM, and agglomerative hierarchical treat distance computations as a black box. This means you can easily exchange the distance measure.

In contrast, k-means requires that the distance measure allows averaging (to compute the cluster centers). While averaging is straightforward for the Euclidean distance, generalizing it to more sophisticated distance measures might be tricky.

Gaussian mixture models are tied to the normal distribution to express "distances" (probabilities act as similarity measures). Of course, there are other mixture models as well, but they all assume a certain distribution of data objects.

Frage 4 - Clustering Algorithms - Number of Clusters (1 Punkt) [ID: 1078965]

In which of the following clustering algorithms do you need to know the number of clusters before clustering?

- [ ] DBSCAN *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- [x] Gaussian mixture models *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- [ ] agglomerative hierarchical *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- [x] k-means *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- [x] PAM *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*

See the lecture. One might argue a bit about agglomerative hierarchical clustering: If you want just one fixed clustering and not a dendrogram, you need to decide on some k. (In contrast, DBSCAN directly yields you a clustering.) However, you can still choose k after creating the dendrogram, which provides you a lot of flexibility.

Frage 5 - Clustering Algorithms - Complex Clustering Structure (1 Punkt) [ID:

1051808]

Assume you have a two-dimensional dataset in which the clusters form the letters "K", "I" and "T". Further assume that the clusters are located next to each other with some space between them, as in the word "KIT", and that there are no outliers. Which of the following clustering algorithms are most suitable for this kind data (ignoring that we also have to parameterize the algorithms correctly)?

- ☐ k-means *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- ☐ hierarchical with complete linkage *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*
- ☑ hierarchical with single linkage *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- ☑ DBSCAN *(Ausgewählt = 0.2 Punkte, Nicht ausgewählt = 0 Punkte)*
- ☐ Gaussian mixture models *(Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.2 Punkte)*

k-means targets at spherical clusters around cluster centers, so it might have difficulties separating the natural clusters (= letters) here.

Complete linkage also keeps clusters rather compact.

Mixture models are more flexible than k-means and support elliptical clusters, but might have problems with the irregular distribution of points in our example as well.

To detect clusters of arbitrary shape, DBSCAN and hierarchical clustering with single linkage are best-suited here, assuming some further conditions are met (e.g., clusters should have similar density for DBSCAN and there should not be a small chain of outliers between clusters to disturb single linkage).

Frage 6 - Silhouette Coefficient - Range (1 Punkt) [ID: 1078971]

Match the (approximate) silhouette values of individual data objects or clusterings to the clustering results!

A data object in the middle of its cluster.	passt zu	1	*(0.2 Punkte)*
A data object between two clusters.	passt zu	0	*(0.2 Punkte)*
A data object assigned to the wrong cluster.	passt zu	-1	*(0.2 Punkte)*
A clustering consisting of only one cluster.	passt zu	undefined	*(0.2 Punkte)*
A clustering consisting only of one-object clusters.	passt zu	0	*(0.2 Punkte)*

The silhouette coefficient takes values in $[-1, 1]$, having higher values if data objects are close to objects in their own cluster, but far from data objects in other clusters (to be precise, in the closest other cluster). The silhouette of a data object is set to zero by definition if the data object is the only one in its cluster. If there are no other clusters, you cannot compute the silhouette of a data object at all.

Frage 7 - PAM - Algorithm (1 Punkt) [ID: 1051809]

Order the steps of Partitioning Around Medoids!

Choose initial medoids

Compute current quality

[BEGIN] Loop until no change

[BEGIN] Loop over medoid/non-medoid pairs

Compute quality for swapping

[END] Loop over medoid/non-medoid pairs

Select pair with best quality

[BEGIN] If swapping improves quality

Swap medoid and non-medoid

Update current quality

[END] If swapping improves quality

[END] Loop until no change

See the lecture. In the steps outlined here, we left out the assignment of non-medoids to medoids; we treat this as a part of computing quality.

Frage 8 - Gaussian Mixture Models - Number of Parameters (1 Punkt) [ID: 1078969]

Assume we want to train a Gaussian mixture model with three components for a five-dimensional dataset. How many parameters does the model have, assuming that we store all vectors and matrices naively (i.e., we do not consider redundancies among the parameters and treat each entry of a matrix or vector as one parameter)?

Der Wert muss zwischen 93 und 93 liegen

- For three components, we have three mean vectors, with five entries each due to the five dimensions.
- Also, we have three covariance matrices with 25 entries each, as we store the covariance between each pair of dimensions.
- Finally, we need to store three mixing coefficients.

Overall, this yields $3 * 5 + 3 * 5^2 + 3 = 93$ parameters if we do not consider redundancies. In fact, as the covariance matrices are symmetric, the actual number of parameters is only roughly the half of that figure. Also, knowing $k - 1$ mixing coefficients determines the last one, as they sum up to one.

Frage 9 - DBSCAN - Algorithm (1 Punkt) [ID: 1051811]

Order the steps of DBSCAN!

[BEGIN] Loop: For each unprocessed point x

Mark x as processed

N := neighbors(x, epsilon)

[BEGIN] If |N| >= MinPts:

C := new cluster

Add x to C

Extend C recursively

[END] If

[BEGIN] Else:

Mark x as "noise"

[END] Else

[END] Loop

See the lecture.

Frage 10 - DBSCAN - Object Types (1 Punkt) [ID: 1078973]

Match where the three types of objects in DBSCAN might end up!

Cluster	passt zu	Core object	*(0.25 Punkte)*
Cluster	passt zu	Border object	*(0.25 Punkte)*
Noise	passt zu	Border object	*(0.25 Punkte)*
Noise	passt zu	Neither core nor border object	*(0.25 Punkte)*

See the lecture. All data objects are either assigned to a cluster or marked as noise.

Frage 11 - OPTICS - Insertion Order (1 Punkt) [ID: 1081570]

Which criterion does OPTICS use to select a new point to be processed for the output list?

It's the minimum *(0.2 Punkte)* reachability distance *(0.4 Punkte)* of an unprocessed point from all processed points *(0.4 Punkte)*

See the lecture for the formulas and the full pseudo-code of the OPTICS algorithm. Note that the **reachability distance** combines (i.e., calculates the maximum) of the actual distance between two points and the core distance of one of the points (here: the already processed point). As we want consecutive points in the output list to be close together, we take the **minimum** reachability distance. Also, as we update the priority list each time a point is processed, this list contains the minimum reachability distance of each unprocessed point **to all processed points**.

Frage 12 - BIRCH - Clustering Features (1 Punkt) [ID: 1081572]

Assume we conduct BIRCH on one-dimensional data. Let the clustering feature CF = (n, LS, SS) = (3, 6, 18) be given.

The centroid of the cluster is 2 *(0.5 Punkte)* . The radius of the cluster is 1.414 *(0.5 Punkte)* .

If necessary, use a decimal point (.) as decimal separator and round your solution to three decimal places.

According to the formulas from the lecture,

- $Centroid(CF) = \frac{LS}{n} = \frac{6}{3} = 2$

- $Radius(CF) = \sqrt{\frac{SS}{n} - (\frac{LS}{n})^2} = \sqrt{\frac{18}{3} - (\frac{6}{3})^2} = \sqrt{2} = 1.414$

An exemplary one-dimensional set of points for this CF vector is $\{1, 1, 4\}$. The radius corresponds to the (population) standard deviation of this set.

Frage 13 - Outlier Detection - Statistical Approach (1 Punkt) [ID: 1051813]

Which fraction of normally distributed data points has a distance of at least three standard deviations to the mean? Hint: If you don't want to consult a probability table, ***scipy.stats.norm.cdf()*** might come in handy.

Use a decimal point (.) as decimal separator and round your solution to three decimal places.

Der Wert muss zwischen 0.003 und 0.003 liegen

You can compute the solution from the cumulative distribution function: ***round(2 * (1 - scipy.stats.norm.cdf(3)), 3)*** (after importing ***scipy.stats***). We evaluate the cumulative distribution at x=3 (for a standard normal distribution, this point is exactly three standard deviations from the mean), invert the probability (because we are interested in points lying outside) and multiply with two (since the distribution is symmetrical). (We could also have calculated ***round(2 * scipy.stats.norm.cdf(-3), 3)*** instead.)

Frage 14 - Outlier Detection - Distance-Based Approach (1 Punkt) [ID: 1080880]

The expected number of *DB(p,d)*-outliers increases if ...

- ☑ p decreases
  (Ausgewählt = 0.25 Punkte, Nicht ausgewählt = 0 Punkte)
- ☐ p increases
  (Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.25 Punkte)
- ☑ d decreases
  (Ausgewählt = 0.25 Punkte, Nicht ausgewählt = 0 Punkte)
- ☐ d increases
  (Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.25 Punkte)

For both hyperparameters, lowering them makes the outlier criterion less restrictive und thus more points likely to be labeled as outliers.

A lower p means that fewer points need to have a distance of d to the point under discussion.

A lower d means that the other (at least p) points need to satisfy a lower minimum distance from the point under discussion.

Frage 15 - Outlier Detection - Density-Based Approach (1 Punkt) [ID:

1080882]

Assume all pairwise distances between points as well as k-distances of points are distinct, i.e., there cannot be ties.

To compute LOF for $k = 5$, one has to consider the reachability distances of at least [5] *(0.5 Punkte)* further points (i.e., excluding the point under discussion) and at most [25] *(0.5 Punkte)* further points.

To compute LOF, one needs to consider the local reachability density of the point itself (which we do not consider here) and k neighbors (if there aren't ties of distances, as stated in the task). To compute the local **reachability density** of a point, we need to consider the **reachability distances** of its k neighbors. If all neighbors of the point under discussion are close together and further points are at a higher distance, the former might also be each other's nearest neighbors. Thus, you might only need to consider k reachability distances. As the other extreme, all neighbors of the point under discussion might have different nearest neighbors, resulting in k^2 reachability distances to be computed.

Frage 16 - Outlier Detection - Neural-Based Approach (1 Punkt) [ID: 1081650]

Match the types of neural-based outlier detection to their architectural characteristics!

Autoencoder	passt zu	Input layer and output layer have same size, smaller bottleneck layer between them.	*(0.34 Punkte)*
Self-Organizing Map	passt zu	Grid of neurons.	*(0.33 Punkte)*
Restricted Boltzmann Machine	passt zu	Layer of visible neurons and layer of hidden neurons.	*(0.33 Punkte)*

See the lecture.

Frage 17 - Outlier Detection - Subspace Search (1 Punkt) [ID: 1051815]

Let a dataset with 50 features be given. How many subspaces with five features are there?

Der Wert muss zwischen 2118760 und 2118760 liegen

This is a simple combinatorial problem: select 5 out of 50 elements without replacement, order does not matter. Thus, the solution is $\frac{50!}{(50-5)!*5!}$. You can compute this in Python with ***math.comb(50, 5)***.

Frage 18 - Outlier Detection - Non-Trivial and Strong Outliers (1 Punkt) [ID: 1051816]

Assume a chair is evaluating an exam with two tasks. The goal is to find outliers in the 2D data space.

Scenario 1: Somebody gets an overall score that is clearly higher than that of all other students. However, for each of the two tasks alone, there are students with similar scores as the student under discussion. Also, there are a few exceptionally low or high scores of other students for individual tasks.

Scenario 2: Somebody has the best score overall and is clearly better than all other students are for each task.

Evaluate the following statements!

- ☐ Scenario 1 is an example of a strong outlier.
  (Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.25 Punkte)
- ☑ Scenario 1 is an example of a non-trivial outlier.
  (Ausgewählt = 0.25 Punkte, Nicht ausgewählt = 0 Punkte)
- ☐ Scenario 2 is an example of a strong outlier.
  (Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.25 Punkte)
- ☐ Scenario 2 is an example of a non-trivial outlier.
  (Ausgewählt = 0 Punkte, Nicht ausgewählt = 0.25 Punkte)

Scenario 1: As the score is only exceptional overall, but not in the subspaces (tasks), the outlier is non-trivial. However, the outlier is not strong, as other students are outliers in subspaces.

Scenario 2: The outlier is trivial, as it already occurs in subspaces. Thus, it cannot be strong as well.