Hyperiondev

TASK

Clustering I

Visit our website

Copyright © 2021 HyperionDev. All rights reserved.

Introduction

WELCOME TO THE CLUSTERING TASK!

Supervised learning involves datasets with both input and output variables. However, there is another class of problems, unsupervised learning problems, where we only observe input variables. In this task, we will introduce our unsupervised method: clustering.

Get in touch

Connect for support

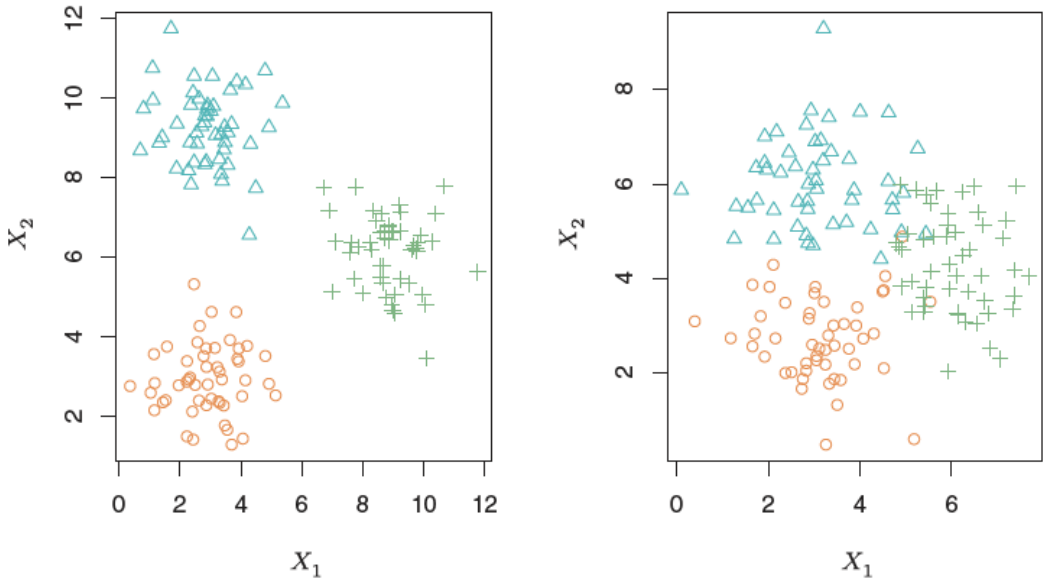
Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to Discord at **https://discord.com/invite/hyperdev** where our specialist team is ready to support you.

Our team is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

Copyright © 2021 HyperionDev. All rights reserved.

INTRODUCTION TO CLUSTERING



The graphs above show two datasets that are good candidates for applying clustering. The data on the left exhibits a very clear grouping that a clustering algorithm could readily identify for us. The data on the right has groups with more overlap and will be harder to identify, but still more suited to a clustering approach than, for example, linear regression.

If a clustering approach seems suitable, you can use cluster analysis to ascertain, on the basis of your input variables x1, . . . , xn, whether or not the observations fall into relatively distinct groups by asserting that observations within a group are similar to each other, while observations in different groups are different from each other.

Note that, as usual, we use examples that can be visualised on a two-dimensional plane. In practice, data sets can contain many more than two variables with more complicated relationships among them.

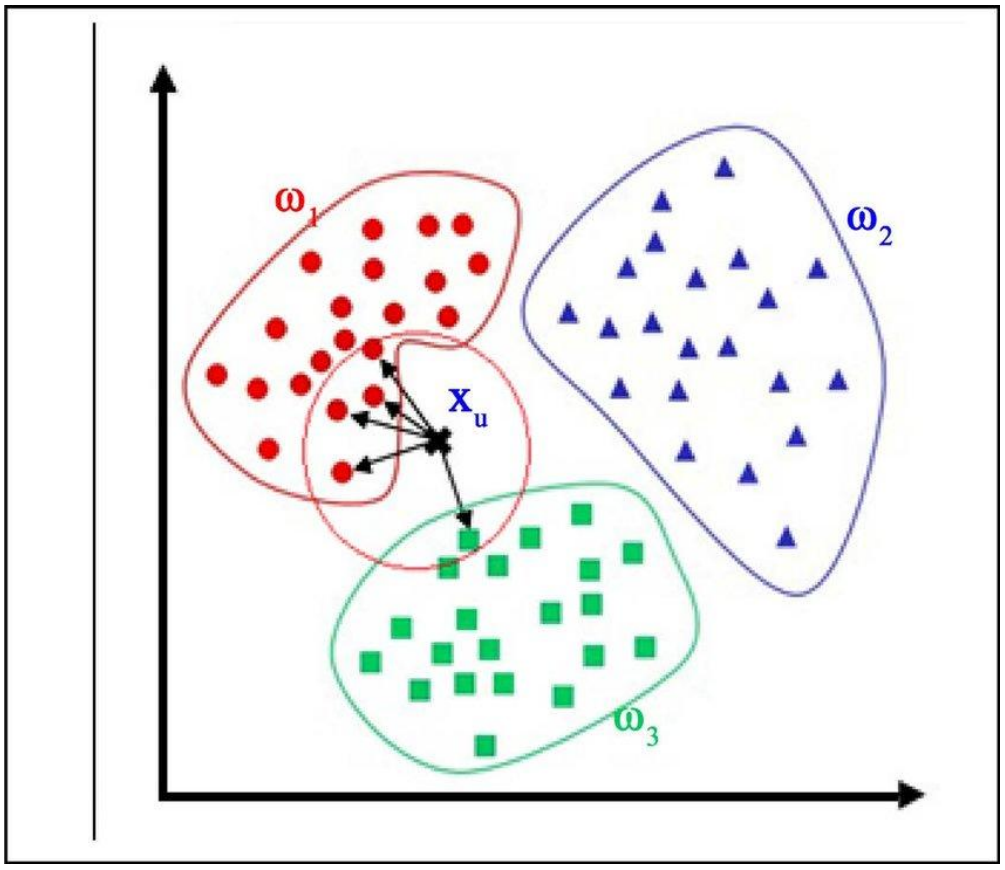
K-NEAREST NEIGHBOURS

K-Nearest Neighbours (KNN) is a very simple *supervised* clustering algorithm. It works on the assumption that any point in the dataset will probably be in the same class as its K nearest neighbours (where you can set the value of K as you wish).

Examine the diagram below: we have three clusters w_1, w_2 and w_3. We have a new point to be predicted: x_u. To determine the class of this new point, we examine the 5 closest neighbours to that point. We see that there are 4 red neighbours and 1

Copyright © 2021 HyperionDev. All rights reserved.

green neighbour. Because most of the neighbours are red, we assign the new point to the red class.



K-MEANS CLUSTERING

K-means clustering is the most well-known clustering algorithm. It is a simple and elegant approach for partitioning a data set into K distinct clusters. To perform K-means clustering, we first specify the desired number of clusters, K, and then assign each observation to exactly one of the K clusters.

The principle behind K-means clustering is that a good clustering is one for which the within-cluster variation is as small as possible. The within-cluster variation — measured as inertia or within-cluster sum-of-squares — is a measure of the amount by which the observations within a cluster differ from each other. The within-cluster variation should be low and the without-cluster variation should be high.

Feature space

The similarity between observations is determined by the distance between their points in space. If observation #1 in a dataset of flowers has a height of 40 cm, and

Copyright © 2021 HyperionDev. All rights reserved.

a flower width of 2 cm, that data point has the coordinates [40, 2] in the *feature space* of the dataset. If observation #2 has the coordinates [43, 2], and observation #3 has the values [10, 0.5], we can say that observation #2 is 'closer' to #1 than #3 is. A clustering algorithm will place #1 and #2 in the same cluster based on that closeness. There are a number of different distance metrics that are used in algorithms to decide how similar instances are. The most common one is called Euclidean distance. In two dimensions, the Euclidean distance between the points (x_i, y_i) and (x_j, y_j) is $\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$.

To compute the mean (or average) of a number of observations, you divide the sum of those observations by the number of observations. A multi-dimensional mean is equally straightforward. To compute the mean of a certain number of (x, y) points, you compute the mean of all x values and the mean of all the y values.

The K-means algorithm

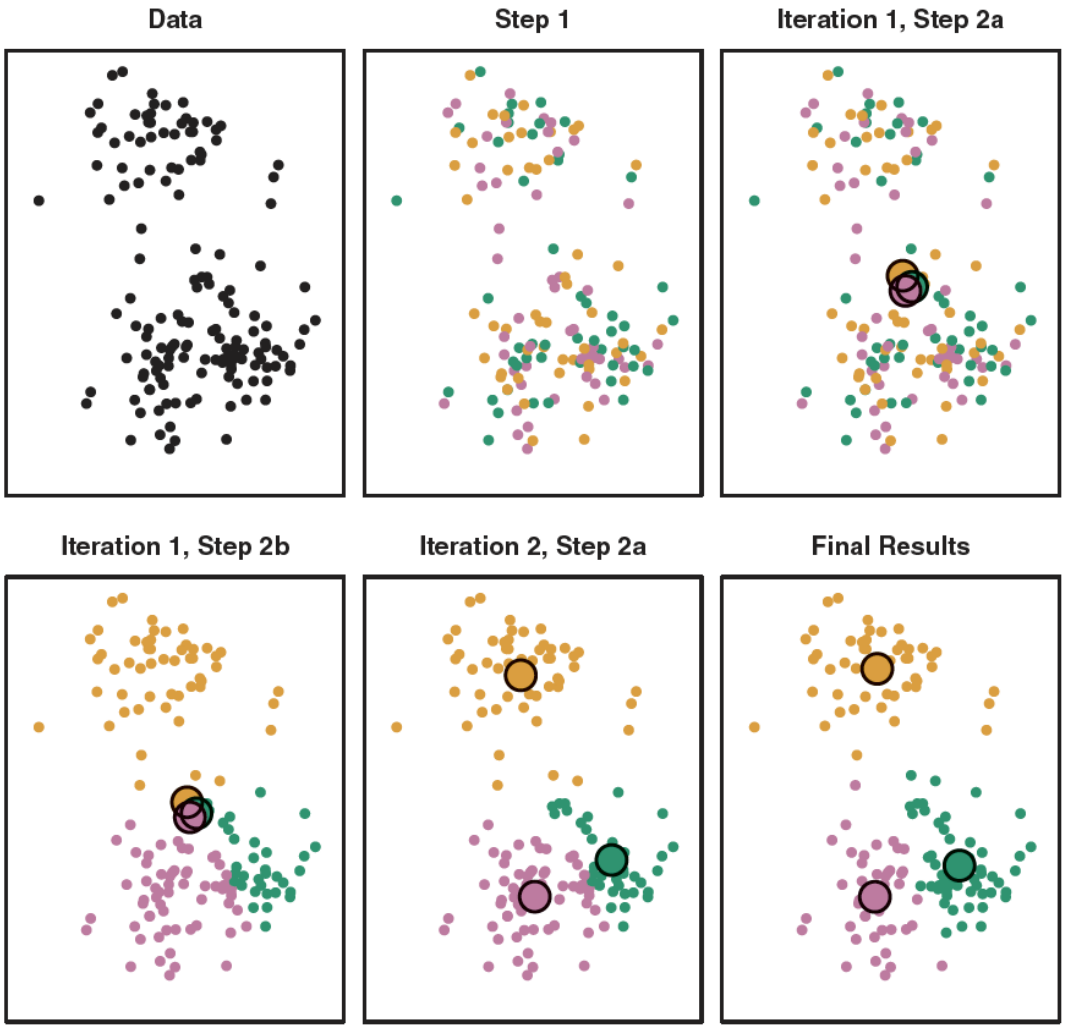
The K-means algorithm follows the following steps:

1. Select a number of clusters, K.
2. Select random points from the data as starting values and initialise the mean of each cluster (E.g. using the `sample()` function in Python).
3. For *n* number of iterations:
    a. Assign each point to the cluster whose mean (or "centroid") is the nearest.
    b. Re-compute the means for each cluster based on its current members.
4. Repeat 3 until convergence.

Convergence here means that the means of each cluster no longer or barely change between iterations, i.e. the value has 'stabilised'. When the K-means algorithm converges, it may have reached a local optimum. This means that the algorithm found the best values given the initialisation, but that there exists a clustering of the data with an even lower within-cluster variation. To avoid this, run the algorithm multiple times and select the most optimum solution.

The following figure shows 6 iterations of the K-means algorithm. At the start of iteration 1, the centres are based on random points and are therefore close to the global average, producing a very poor clustering of the points. Subsequent iterations drag the cluster centres out. By the 6th iteration, a sensible clustering has been found.

Copyright © 2021 HyperionDev. All rights reserved.

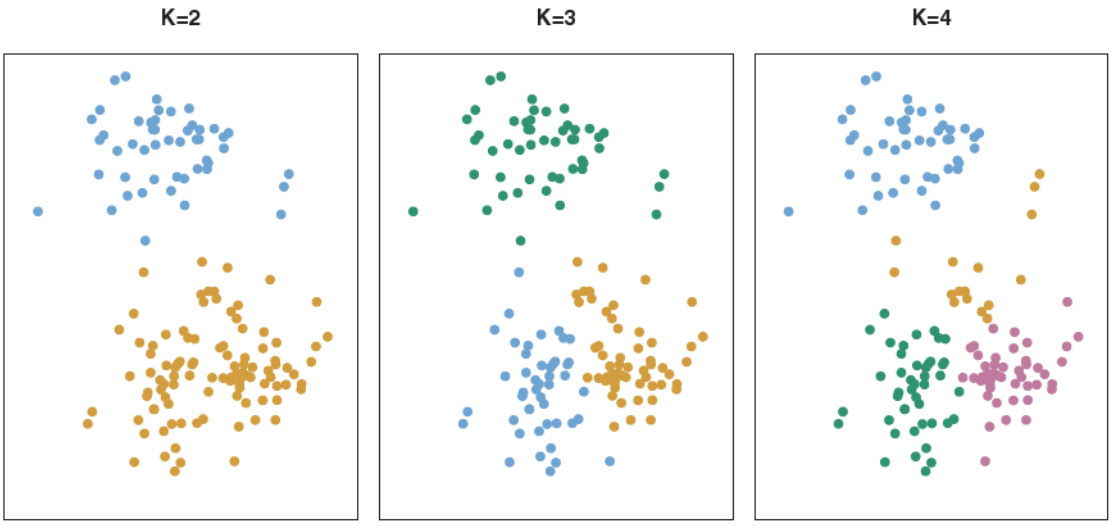


(Image source: Lin, 2015)

Choosing K

An important thing to consider when applying the K-means algorithm is that you need to choose K before running the analysis. Choosing K will greatly affect the outcome and the accuracy of the clusters. The following plots show different outcomes of the algorithm depending on the value chosen for K.

Copyright © 2021 HyperionDev. All rights reserved.



(Image source: Lin, 2015)

Validating the clusters

It is possible to find clusters in any data, but it is important to determine if these clusters actually represent underlying subgroups in the data or are merely groupings with similar noise.

This is a very hard question to answer. There exists a number of techniques for assigning a significance value to a cluster in order to assess whether there is more evidence for the cluster than one would expect due to chance. However, there has been no consensus on a single best approach. The Silhouette Coefficient (**sklearn.metrics.silhouette_score**) is an example of an evaluation metric which indicates how similar samples within a cluster are, compared to other clusters. A higher Silhouette Coefficient score relates to a model with better-defined clusters.

The Silhouette Score

This score is a measure of how well-defined the clusters are. The documentation defined two metrics, **a** and **b**. **a** is defined as the mean *intra*-cluster distance (the average of distances within a specific cluster), and **b** is defined as the mean *nearest*-cluster distance (the distance from a specific point to the nearest cluster). The computation for the Silhouette Score is (b - a) / max(a, b). This gives a value between -1 and 1, where -1 is the worst score and 1 is the best score.

If the Silhouette Score is close to -1, this means that there are many samples belonging to the wrong cluster. This is because **b** is smaller than **a**. Practically speaking, it means that, for many points within a specific cluster, there is actually another cluster that is closer to that point than most of the rest of the cluster. This just means that the point is not in the right cluster.

Copyright © 2021 HyperionDev. All rights reserved.

If the Silhouette Score is close to 0, this denotes overlapping clusters. Visualise two clusters that are basically next to each other: for about half the points there, the other cluster will be closer than the rest of its own cluster. **a** and **b** will be about the same at that stage, as the distances will be very similar.

If the Silhouette Score is close to 1, this means that the clusters are well-defined. This is because, in all cases, **b** will be much larger than **a**, as the nearest cluster will be so far away that it will always produce a positive number.

Instructions

- Read the Jupyter notebook in this task's folder before trying the compulsory task.
- Take a look at **this blog** to read more about the drawbacks of K-means clustering.

Compulsory Task 1

Follow these steps:

- Load the **Iris.csv** into a notebook.
- Plot six different scatter plots with the different combinations of the variables. For example, sepal length vs petal length.

  In each scatter plot:

  - Code the different species observations with different colours.
  - Which of these plots looks the most promising for separating into clusters?
- Select two of the most promising plots and build a K-Nearest Neighbours model with k = 3 for each of these pairs of features.
- For each of these two models:
  - What is the accuracy of your model?
  - Create a scatter plot showing the clusters predicted by the model.

Copyright © 2021 HyperionDev. All rights reserved.

If you are having any difficulties, please feel free to contact our specialist team **on Discord** for support.

Completed the task(s)?

Ask an expert to review your work!

Review work

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we've done a good job?

Click here to share your thoughts anonymously.

REFERENCES

Lin, P. (2015). Ch10: Unsupervised Learning. Retrieved 28 August 2020, from **http://datasciencehc.github.io/Study-ISLR/Ch10_Unsupervised%20Learning/Ch10.html#(6)**

Copyright © 2021 HyperionDev. All rights reserved.