

Microsoft: DAT210x Programming with Python for Data Science

Help

Course > 5. Data Modeling > Lecture: Clustering > Video

Video

☐ Bookmark this page

SciKit-Learn and K-Means

Start of transcript. Skip to the end. Let's take a peek at how to use KMeans for clustering with scikit-learn. KMeans is a class in scikitlearn's clustering module. So you'll have to import that (Caption will be displayed when you start playing in order to use it, of course. the video.) And there are many other different clustering classes or techniques that scikit-learn also supports as well. 0:00 / 7:18 1.25x X CC We'll take a look at some of

Video

Download video file

Transcripts

Download SubRip (.srt) file Download Text (.txt) file

It's very simple to get up and running with K-Means in SKLearn. Given a dataframe $\mathrm{d} f$, you can compute its labels and centroids as follows:

```
>>> from <u>sklearn.cluster</u> import KMeans
>>> kmeans = KMeans(n clusters=5)
>>> kmeans.fit(df)
KMeans(copy_x=True, init='k-means++', max_iter=300, n_clusters=5, n_init=10,
    n_jobs=1, precompute_distances='auto', random_state=None, tol=0.0001,
    verbose=0)
>>> labels = kmeans.predict(df)
>>> centroids = kmeans.cluster_centers_
```

The most important factor for you to focus on being n_clusters, the "K" number of clusters you want K-Means to place for you. Also experiment with different initialization methods, including rolling your own and in the positions as an NDArray shaped as [n_clusters, n_features]. We've include more details for you on that in the dive deeper section.

© All Rights Reserved





© 2012-2017 edX Inc. All rights reserved except where noted. EdX, Open edX and the edX and Open EdX logos are registered trademarks or trademarks of edX Inc. | 粤ICP备17044299号-2

















