Clustering

K-means clustering - example I

K-means clustering will divide data in k clusters, each with its own center. The algorithm works iteratively to group together data points that are spatially closer to one-another.

We will generate a synthetic data set that has natural clusters and then will use clustering functions to see how it works.

Here is the function that will create the synthetic data:

```
from sklearn.datasets import make_blobs

help(make_blobs)
```

Ex.1 Please analyse the function parameters and generated output.

Ex.2 Please call `make_blob` and generate a dataset with 120 samples, 2 features, 4 centres (clusters), and cluster_std=0.40.

Ex.3 Analyse the x and y shape.

Ex.4 Plot the data samples in X to see the clusters.

All sklearn algorithms have a similar way of running:

- initialize the algorithm
- fit the model
- predict the outcome for the data

Ex.5 Run the implemented `KMeans` function K-Means, fit the model and predict the outcome.

Ex.6 Plot the results in order to see whether the clustering worked.

Ex.7 Calculate inertia Inertia_ or Dunn index Package jgmcvi (optional).

K-means clustering - example II

Change the parameters of our data generation process to create different-looking clusters (eg. about 5 clusters and cluster_std=0.8).

Ex.1 Plot the data

Ex.2 Predict for the same nb. of clusters = 5.
Ex.3 Predict for 2 instead of 5 clusters.
Ex.4 Calculate inertia or Dunn index (optiona).

K-means ++

Remember how we randomly initialize the centroids in k-means clustering? This is also potentially problematic because we might get different clusters every time. So, to solve this problem of random initialization, there is an algorithm called K-Means++ that can be used to choose the initial values (discussed during the lecture). In latest version of `sklearn` library kmeans++ method is the default version. In this case, change it to a random one and compare the results.

Ex.1 Run K-Means for the previous ex. while changing the parameter in KMeans function:

```
init='k-means++'
# or init='random' in later version
```

Ex.2 Calculate inertia or Dunn index.

Elbow method

Please implement the elbow method to choose the correct parameter value for k.

Ex.1 Run K-Means for a range of clusters using a for loop and collecting the distortions (inertia) into a list.

Ex.2 Plot the distortions of K-Means.

Maximum number of a clasters is 20 Please check whether the K-Means algorithm will correctly cluster the data:

```
from sklearn.datasets import make_moons
import seaborn as sns

x, y = make_moons(1000, noise=.05, random_state=0)
X_moon = pd.DataFrame(x, columns = ['f1','f2'])
#TODO: Kmeans

#Plot
X_moon['k_means'] = y_km
sns.lmplot(data=X_moon, x='f1', y='f2', fit_reg=False, hue = 'k_means',
palette = ['#eb6c6a', '#6aeb6c']).set(title='Algorytm k-srednich')
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=100, alpha=0.5)
plt.show()
```

Clustering II - hierarchical methods

Dendrograms

The dendrogram illustrates how each cluster is composed by drawing a U-shaped link between a non-singleton cluster and its children. The top of the U-link indicates a cluster merge. The two legs of the U-link indicate which clusters were merged. The length of the two legs of the U-link represents the distance between the child clusters. It is also the cophenetic distance between original observations in the two children clusters.
More:Dendograms

Dataset:
Create the dataset using `make blobs` function:

```
X, y_true = make_blobs(n_samples=300, centers=5,
                       cluster_std=0.8, random_state=0)
```

Ex.1 Plot the data.
Ex.2 Use the Means function with k=2 to assign points to clusters.

In Kmeans clustering, we provide the number of clusters and then the algorithm partitions the data. In agglomerative clustering, the data is grouped together based on the distance, and we can decide how many clusters we want, once we see how the data are grouped together.

Ex.3 Use the scipy libraries to draw the dendrogram for the data:

```
from scipy.cluster.hierarchy import dendrogram, linkage
```

where:

- `linkage` is the the function that performs the clustering more
- `dendrogram`more

Ex.4 Something this makes clear is that visualizing the dengrogram is useful, but it's not for big dataset. Create a smaller dataset to draw a dendogram.

Agglomerative clustering

Hierarchical clustering is a type of unsupervised machine learning algorithm used to cluster unlabeled data points. Like K-means clustering, hierarchical clustering also groups together the data points with similar characteristics.

There are two types of hierarchical clustering: Agglomerative and Divisive. In the former, data points are clustered using a bottom-up approach starting with individual data points, while in the latter top-down approach is followed where all the data points are treated as one big cluster and the clustering process involves dividing the one big cluster into several small clusters.

In this ex. we will focus on agglomerative clustering.

Dataset: Please download the dataset

Data

The database contains following information:

```
CustomerID  Genre    Age    Annual Income (k$)    Spending Score (1-100)
1           Male    19    15                    39
2           Male    21    15                    81
3           Female    20    16                    6
4           Female    23    16                    77
5           Female    31    17                    40
```

Ex.5 Please read the dataset and choose columns to further analysis. The dataset contains 200 records and 5 attributes. Use only Annual Income (in thousands of dollars) and Spending Score (1-100) columns for clustering.
Ex.6 Create a dendrogram to chose the appropriate k value.
Ex.7 Analyse results for different `linkage` values - single, average,complete linkage and Ward. The linkage method takes the dataset and the method to minimize distances as parameters. We use 'ward' as the method since it minimizes then variants of distances between the clusters. (pl. Celem zadania będzie porównanie działania metody średnich połączeń (ang. *avarage linkage*), metody Warda, pojedynczego połączenia (ang. *single linkage*) oraz pełnego wiązania (ang. *complete linkage*).).
Ex.8 Use the AgglomerativeClustering class of the sklearn.cluster library to build the model predict clusters.

Ex.9 Plot the clusters to see how actually our data has been clustered.

Selecting the number of clusters with silhouette analysis on KMeans clustering

During our last meeting you have used the `elbow method` to asses the quality of the clusters. Today we will use the silhouette analysis on KMeans clustering to assess the clusters and choose the right k value more.
Ex.1 Create the dataset:

```
X, y = make_blobs(n_samples=500,
                  n_features=2,
                  centers=4,
                  cluster_std=1,
                  center_box=(-10.0, 10.0),
                  shuffle=True,
                  random_state=1)
```

Ex.2 Determine the cluster range [2,3….]
Ex.3 Iterate for all the values in cluster range:

- Perform k-Means
- Calculate and print the silhouette_score
- Compute the silhouette scores for each sample more
- Aggregate the silhouette scores for samples belonging to cluster i, sort them and plot (visualization the result)
- Have a look here to check you solution and plot the silhouette plot more

DBSCAN - density based clusterring

A density based alghorithm has saveral advantages:

- It does not require the number of clusters
- It can detect outliers - points from outside of any clusters
- It can solve all the various shapes and sizes.

However, it is pretty slow and requires a cerefull parameter selection.

Ex. Compare DBSCAN algorithm with k-means on the moon dataset.

Use DBSCAN from `sklearn` library.

```
from sklearn.datasets import make_blobs, make_moons, make_circles
x, y = make_moons(n_samples=200, noise=.05, random_state=0)
plt.scatter(x[:,0],x[:,1])
plt.show()
```

You can use `make_moons`, `make_blobs` or `make_circles` to experiment with the alghorithms. DBSCAN parameters have to be chosen “empirically”.

Interesting paper or forum post about DBSCAN parameter selection can be found in the links.

Additionally

https://www.efavdb.com/mean-shift