CE807 Lab 2
Document clustering with SciKit Learn

All the scripts is in Python files. However, all experiments needs to be done on the GoogleColab (https://colab.research.google.com/).
Google Colab tutorials

- https://algotrading101.com/learn/google-colab-guide/
- https://www.youtube.com/watch?v=iMlMfrXJYSg

Scikit Learn includes in the package `sklearn.cluster` implementations of a number of clustering algorithms. In this lab we are going to learn about this package to cluster documents, focusing in particular on kMeans. Scikit-learn also provides support for several feature reduction / cluster evaluation methods.

All the materials for this lab, including this script, can be found in moodle.

Copy all the material in that folder, including the data sub-folder, in a directory in your file space.

1. The implementation of KMeans in scikit-learn

The main aspects of the implementation of KMeans in scikit-learn are:

- the function `KMeans`, which takes as input a number of clusters and an initialization of the centroids and returns an object of type KMeans;
- the function `fit`, which takes as input a set of feature vectors and outputs clusters;
- the function `predict`, which specifies the cluster each element belongs to.

Exercise study the documentation for KMeans at:

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

2. A toy example

The script `plot_kmeans_example.py` in the `Lab2` directory contains an example of using KMeans to cluster a toy dataset. If you haven't yet done so, copy

the script, the script `utils.py`, and the `data` subdirectory in one of your folders.

The script

1. Creates a few random points using the norm function, saved in the image `charts/1400_03_01.png` (i.e., the file `1400_03_01.png` in the subdirectory charts that should be created if the script executes correctly).
2. Runs one iteration of KMeans with random initialization over these points, and outputs the results in `charts/1400_03_02.png`
3. `Runs two iterations` and outputs the results in `charts/1400_03_03.png` – notice how the centroids have moved
4. Runs 10 iterations and saves the output in `charts/1400_03_04.png`

Exercise examine the diagrams, and make sure you understand what happens with the centroids after 1, 2, and 10 iterations.

A lot of the code in the script is scipy trickery to visualize the diagrams properly. Ignore that. But do make sure you understand the calls to the KMeans implementation.

Exercise in particular: what are the arguments to KMeans? How is `fit` used to create the clusters? For bonus points: what are `xw1, yw1, x, and y`? What is `meshgrid`?

3. Understanding TFIDF: a toy implementation

Although scikit-learn and many other packages provide implementations of tfidf, in order to understand what it does it is a good idea to see how it can be implemented and how it works with a toy document collection.

The script `tfidf.py` contains an implementation of tfidf, and shows what values it is going to take on the 'documents' in a toy collection D. Copy the script and the script `utils.py` in one of your folders if you haven't yet done so.

Exercise run the script.
Exercise make sure you understand what the script does. How are documents represented? How is the collection of documents represented? How are the tf and idf terms computed?

4. Clustering 20Newsgroups using KMeans

The script `rel_post_20news.py`, also in the `Lab2` page, uses KMeans to cluster a real life dataset –the 20Newsgroups dataset of news articles belonging to 20 different newsgroups already used in last week's lab. This script puts everything that we have seen in the previous lab and in the current lab together:

1. It downloads a subset of the entire 20newsgroups dataset

   Exercise can you find the parts of the script that choose a subset of the categories and download the documents belonging to those categories?

2. It creates vectors for these documents using `StemmedTfidfVectorizer`, a type of vectorizer that stems and uses TFIDF to create vectors.

   Exercise study the description of the TfidfVectorizer class in http://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.Tfid fVectorizer.html

   Notice in particular the argument `decode_error` that prevents the method to crash if it encounters character encoding errors, which are all too common.

3. It clusters these vectors into 50 different clusters using KMeans ran for 17 iterations.

   This is done by the code:

   ```
   num_clusters = 50
   from sklearn.cluster import KMeans km =
   KMeans(n_clusters=num_clusters, n_init=1,
   verbose=1, random_state=3)
   clustered = km.fit(vectorized)
   ```

   Exercise if you haven't yet done so, look at the description of the use of the `KMeans` and `fit` functions.

4. After fitting, we get the clustering information out of the members of `km`. For every vectorized post, there is an integer label in `km.labels_`

   ```
   print("km.labels_=%s" % km.labels_) #
   km.labels_=[ 6 34 22 ...,  2 21 26]

   print("km.labels_.shape=%s" % km.labels_.shape)
   # km.labels_.shape=3529
   ```

5. The script then computes and prints out several assessments of the quality of the clusters, using metrics such as *homogeneity* and the *rand index*:

```
from sklearn import metrics print("Homogeneity:
%0.3f" %
metrics.homogeneity_score(labels, km.labels_))
# Homogeneity: 0.400 ….
```

Exercise study the metrics of cluster quality implemented in the sklearn.metrics.cluster package:
http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation
making sure you understand at least homogeneity.

6. Finally, the script shows you how you can find the most similar news posts to a new post by assigning the new post to one of the existing clusters.

```
new_post = \     """Disk drive problems. Hi, I
have a problem with my hard disk.
After 1 year it is working only sporadically now. I
tried to format it, but now it doesn't boot any
more.
Any ideas? Thanks.
"""
```

The first step is to vectorize the new post:

```
new_post_vec = vectorizer.transform([new_post])
new_post_label = km.predict(new_post_vec)[0]
```

Now that the post has been assigned to a cluster, we can find the most similar posts in that cluster only:

```
similar_indices = (km.labels_ ==
new_post_label).nonzero()[0]
```

We can now find the most similar posts, and print the top 3:

```
similar = [] for i in similar_indices:
dist = sp.linalg.norm((new_post_vec -
vectorized[i]).toarray())
    similar.append((dist, train_data.data[i]))

similar = sorted(similar)
print("Count similar: %i" % len(similar))
```

```
show_at_1 = similar[0]
show_at_2 = similar[int(len(similar) / 10)]
show_at_3 = similar[int(len(similar) / 2)]
```

Exercise run `rel_post_20news.py` and make sure you understand what it does.

5. Experimenting with kmeans

The script `document_clustering.py` can be used to test the effect of using

- different versions of the kmeans algorithm
- using different numbers of features
- using idf or not
- using latent semantic analysis to reduce the number of features (to be discussed in the next lectures)

On various measures of cluster quality.

You can use the script from a Terminal command line by typing

```
>python document_clustering.py
```

and then specifying the options.

Exercise study carefully the script, and then try to run it with different configurations.

6. Dealing with noise

You will find that the clustering is not perfect: some posts will be clustered with posts to a different newsgroup.

The script `noise_analysis.py` shows examples of why this happens.

Exercise try to understand what the script does.

7. Hierarchical clustering

The `AgglomerativeClustering` scikit-learn object performs hierarchical clustering `bottom-up', starting from each object as a separate cluster, and then

merging these clusters two at a time. The class supports three different *linkage criteria*—metrics used to decide which clusters are closer:

- Maximum distance
- Average distance
- Ward distance: this is a criterion that minimizes the sum of square differences.

The script `plot_ward_structured_vs_unstructured.py` (included with the scikitlearn documentation) shows how to use `AgglomerativeClustering` with the 'ward' option.

Exercise study the script. Notice in particular how the same functions are used to fit and access the model.

References

- scikit-learn's documentation at http://scikit-learn.org/stable/index.html
- Richert & Coelho, Building Machine Learning Systems with Python (2nd ed), Packt Publishing. (We covered material from chapter 3) ○ Code for the book at: https://github.com/luispedro/BuildingMachineLearningSystemsWithPython