Coursework Description

Comparing SKLEARN Clustering Algorithms

There are a lot of clustering algorithms to choose from. The standard `sklearn` clustering suite has thirteen different clustering classes alone. So, what clustering algorithms should you be using? As with every question in data science and machine learning, it depends on your data. A number of those thirteen classes in `sklearn` are specialised for certain tasks. Obviously, an algorithm specialising in text clustering is going to be the right choice for clustering text data. Thus, if you know enough about your data, you can narrow down on the clustering algorithm that best suits that kind of data or the sorts of important properties your data has. But what if you don't know much about your data? If, for example, you are 'just looking' and doing some exploratory data analysis (EDA), it is not so easy to choose a specialised algorithm.

So, what algorithm is good for exploratory data analysis?

Some scope (domain) for Exploratory Data Analysis (EDA) with clustering

To start, let's lay down some ground rules of what we need a good EDA clustering algorithm to do; then, we can set about seeing how the algorithms available stack up.

- **Don't be wrong!** If you are doing EDA, you are trying to learn and gain intuitions about your data. In that case, it is far better to get no result at all than a result that is wrong. Bad results lead to false intuitions, which in turn send you down completely the wrong path. Not only do you not understand your data, but you also *misunderstand* your data.
- **Intuitive Parameters**: All clustering algorithms have parameters; you need some knobs to adjust things. The question is: how do you pick settings for those parameters? If you know little about your data, it can be hard to determine what value or setting a parameter should have. This means parameters need to be intuitive enough that you can hopefully set them without having to know a lot about your data.
- **Stable Clusters**: If you run the algorithm twice with a different random initialisation, you should expect to get roughly the same clusters back. If you vary the clustering algorithm parameters, you want the clustering to change in a somewhat stable, predictable fashion.
- **Performance**: Data sets are only getting bigger. Ultimately, you need a clustering algorithm that can scale to large data sizes.

Experimental Design:

In this coursework, you will be testing (assessing) various clustering algorithms in various ways. In order to compare the clustering algorithms, you must use the same code implementation to test all of them. Thus, **it is critical to use the code provided below** to cluster the data and produce results. Since this is an experiment, you will be required to set it up first in two steps: (A) Getting your environment set up and (B) Preparing and Testing Clustering Algorithms.

Note: The dataset provided ***data.npy*** is already clean and requires no further preprocessing.

A. Getting the environment set up.

If we are going to compare clustering algorithms, we'll need a few things: first, some libraries to load and cluster the data, and second, some visualisation tools so we can look at the results of clustering.

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn.cluster as cluster
import time
%matplotlib inline
sns.set_context('poster')
sns.set_color_codes()
plot_kwds = {'alpha' : 0.5, 's' : 20, 'linewidths':3}
```

Next, we need some data; download the ***data.npy*** from the Blackboard.

This is an artificial dataset that will give clustering algorithms a challenge – it contains some non-globular clusters, some noise, etc.; the sorts of things we expect to crop up in messy real-world data. So that we can actually visualise clustering, the dataset is two-dimensional; this is not something we expect from real-world data where you generally can't just visualise and see what is going on. By running the following code, you will be able to visualise the dataset in 2D space, see Figure 1.

```
data = np.load('/content/data.npy')
```

So let's have a look at the data and see what we have.

```
plt.scatter(data.T[0], data.T[1], c='b', **plot_kwds)
frame = plt.gca()
frame.axes.get_xaxis().set_visible(False)
frame.axes.get_yaxis().set_visible(False)
```

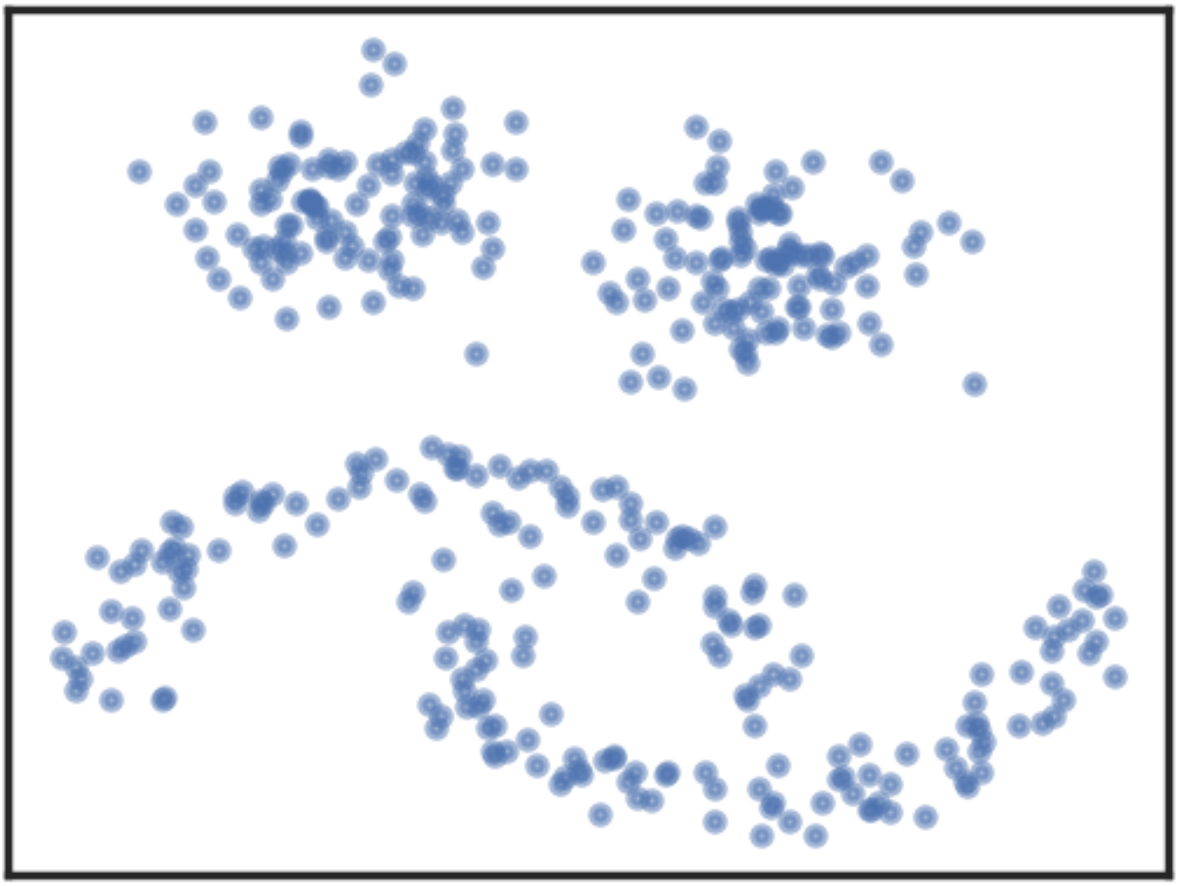

Fig.1. Visualisation of the coursework dataset '***data.npy***'

It's messy, but there are certainly some clusters that you can pick out by eye; determining the exact boundaries of those clusters is harder of course, but we can hope that our clustering algorithms will find at least some of those clusters. So, on to testing with Sklearn Clustering Algorithms

B. Preparing Clustering Algorithms for Testing

To start, let's set up a little utility function to do the clustering and plot the results for us. When running the utility function call, the data will be clustered automatically for you.

We can also time the clustering algorithm while we're at it and add that to the plot since we do care about performance.

```
#This is the clustering and clusters plotting function

def plot_clusters(data, algorithm, args, kwds):
    start_time = time.time()
    labels = algorithm(*args, **kwds).fit_predict(data)
    end_time = time.time()
    palette = sns.color_palette('deep', np.unique(labels).max() + 1)
    colors = [palette[x] if x >= 0 else (0.0, 0.0, 0.0) for x in labels]
    plt.scatter(data.T[0], data.T[1], c=colors, **plot_kwds)
    frame = plt.gca()
    frame.axes.get_xaxis().set_visible(False)
    frame.axes.get_yaxis().set_visible(False)
    plt.title('Clusters found by {}'.format(str(algorithm.__name__)), fontsize=24)
    plt.text(-0.5, 0.7, 'Clustering took {:.2f} s'.format(end_time - start_time),
fontsize=14)
```

Before we try doing the clustering, there are some things to keep in mind as we look at the results:

a) In real use cases, we *can't* look at the data and realise points are not really in a cluster; we have to take the clustering algorithm at its word.

b) This is a small dataset, so poor performance here bodes very badly.

c) To plan your experiments' preparation with any algorithm, you must establish the **function little utility call** by modifying **plot_clusters (data, algorithm, args, kwds)**. This means when establishing the little utility call, you need the data, algorithm name, the arguments and the kwrds which you want to experiment with. *Note that the arguments and kwrds represent the algorithm's parameters.*

d) The modifications show only the names of the parameters and/or arguments you intend to use in your test. For example,
To establish k-means with the n-clusters parameter**,** you document**:**
plot_clusters (data, KMeans, (), {'n_clusters':value}), and so on for the rest of the algorithms and their intended associated parameters for your testing.

e) To execute your test, you load your data, then insert values of your choice for the parameters and arguments in the function call, which were established in the preparation, and run it. For example:
plot_clusters(data, cluster.KMeans, (), {'n_clusters':6, })

Important Note: You must save all your tests and executions in a single Python notebook (.ipynb). DO NOT create individual notebooks for each clustering algorithm.

Coursework Tasks & Marking Scheme

TASK 1: Preparing your test Environment:

Set the Python environment for testing the sklearn clustering **algorithms for performing exploratory data analysis outlined in section 1,2**. Answer each point in the same given order.

a. Show evidence of Loading the 'data.npy' data file. **[2 Marks]**
b. Scatter plot the 'data.npy' data. **[3 Marks]**
c. On your Scatter plot, annotate the various groups of data as you observe them. **[7 Marks]**
d. Establish the **plot_clusters (data, algorithm, args, kwds)** little utility function call (syntax only), which does the clustering and the plotting of the results for each of the algorithms: **K-Means**, **DBSCAN**, **Birch**, **OPTICS**, **Affinity Propagation**, **HDBSCAN**, **Agglomerative Clustering**, **Spectral Clustering** **[8 Marks]**

[TASK 1 Total: 20 Marks]

TASK 2: Choosing each algorithm's parameters values

Review and research various published literature articles in the field to advise the clustering function on the (algorithm, args, kwds) values for the following clustering algorithms. You need to **justify your chosen parameters and their values.** You must include in-text citations for your justification. Answer each point in the same given order.

a. K-Means **[3.5 Marks]**
b. Affinity Propagation **[3.5 Marks]**
c. DBSCAN **[3.5 Marks]**
d. Birch **[3.5 Marks]**
e. OPTICS **[3.5 Marks]**
f. HDBSCAN **[3.5 Marks]**
g. Agglomerative Clustering **[3.5 Marks]**
h. Spectral Clustering **[3.5 Marks]**

Substituted the kwds and args values in the little utility call syntax. However, you may find using the following table per algorithm useful (using a table is optional but highly recommended).

[TASK 2 Total: 28 Marks]

Algorithm Name	args or kwds used	Args or kwds values	Justification
	⋮	⋮	⋮

Algorithm Name	args or kwds used	args or kwds values	Justification
	⋮	⋮	⋮

Algorithm Name	args or kwds used	args or kwds values	Justification
	⋮	⋮	⋮

⋮

Algorithm Neame	args or kwds used	args or kwds values	Justification
	⋮	⋮	⋮

TASK 3: Testing and documenting the output of the algorithms

Execute the following sklearn clustering algorithms as per your setup in tasks 1 & 2. **For performing exploratory data analysis, plot the output.**

a. K-Means **[2 Marks]**
b. Affinity Propagation **[2 Marks]**
c. DBSCAN **[2 Marks]**
d. Birch **[2 Marks]**
e. OPTICS **[2 Marks]**
f. HDBSCAN **[2 Marks]**
g. Agglomerative Clustering **[2 Marks]**
h. Spectral Clustering **[2 Marks]**

[TASK 3 Total: 16 Marks]

TASK 4: Writing a short technical report analysing the "Clustering Results"

Write a report that critically summarises the quality of the obtained clusters for each clustering algorithm based on the following aspects:

a. The given domain. **[4 Marks]**
b. The specified (algorithm, args, kwds) values as part of the plot_clusters (data, algorithm, args, kwds) employed function **[12Marks]**
c. Intra-cluster versus inter-cluster distance **[8 Marks]**
d. The documented scatter plots after applying each clustering algorithm. **[8 Marks]**
e. Draft a final conclusion of your analysis on the best-performing clustering algorithm/s for this clustering task. **[4 Marks]**

Present your findings for this task (Task 4) as a technical report. The paper must express your own conclusions and findings. The paper size should be between [950-1500] words, excluding any references and plots. Minimum font size is 10.
Penalty Warning: Papers violating the lower limit or exceeding the upper limit of allowable words will be subject to a penalty of 10% (2 Marks out of 20)

[TASK 4 Total: 36 Marks]

.

All Coursework Tasks Total 100 Marks