Introduction to Data Science Portfolio

Week 8 – k-means and UMAP

TianQiu

## Text using k-means

In this notebook, we will use the [LPL 2018 Summer Playoffs - Player Stats](https://oracleselixir.com/stats/players/byTournament/LPL%2F2018%20Season%2FSummer% 20Playoffs) dataset to try the k-means clustering algorithm. We can then examine the clusters to see if they look reasonable.

### Correlation Matrix

Use the 'df.corr()' function to quickly see the correlation between features.A screenshot of a computer

Description automatically generated

A table of numbers and symbols

Description automatically generated with medium confidence

We can see that a small portion of the data is extremely correlated, but this is probably due to the fact that the data are all derived from the same type of data (e.g. damage per minute and damage percentage).

### Process data

In order to improve the clustering results, we prepared the data, analysed the clusters, visualised the clusters, determined the number of clusters using the 'elbow method', re-dimensioned and re-clustered the data, and analysed the clustering results prior to performing the cluster analysis.

The text data was preprocessed and features were extracted and then the text data was analysed for clustering using the K-means algorithm. By visualising the clustering results, we can observe the distribution and similarity between different clusters. The most appropriate number of clusters is determined to avoid overfitting or underfitting. Subsequently, based on the determined optimal number of clusters, the text data is subjected to dimensionality reduction.

After the above data processing, we can use K-means algorithm for clustering and analysis.

### Cluster Results

A screenshot of a black and white table

Description automatically generated A screenshot of a black and white table

Description automatically generated

Cluster:0 Cluster:1

A screenshot of a game

Description automatically generated A table with numbers and letters

Description automatically generated Cluster:2 Cluster:3

A screenshot of a black and white table

Description automatically generated Cluster:4

We can clearly notice from the clustered data with different results that there are two clusters in which all the players have a 'POS' of 'Junngle' or all of them are 'Support'; and in another cluster most of his players are 'Top'; in another cluster all the players in it have a win rate much higher than 50 per cent; and in the last cluster all the players in it have a high value of 'K': this suggests that the effect of the clustering has a significant effect.

## Image using k-means

A blue x-shaped graph

Description automatically generated with medium confidence In this notebook, we will try the k-means clustering algorithm using the [OSCA]( https://www.pinterest.com/ceo\_of\_bennefischl/asuka-is-just-like-me/) image. Then, we can check if the clustering looks reasonable. Unlike textual data which can be processed using the K-means algorithm, textual data can usually be converted into a numerical feature representation, whereas image data has to be analysed by obtaining features from its pixel information and converting it into a two-dimensional array.

*2D dataset of images plotted on 2D axes*

The image data has to be characterised by its pixel information and converted into a two-dimensional array for analysis.

### Elbow plot

We can use elbow plot to help us determine the optimal number of clusters in K-means clustering by means of a graph.

A red line graph with numbers

Description automatically generatedA red line graph with a red x

Description automatically generated

The graph on the left shows the sum of squares of the in-sample distances for 20 images, but the score is still above 680,000 and I didn't notice the decline becoming flat in the image, so I set the RANGE to (1, 500) and looked at the score change. In Figure 2 we can see that the score is approaching 0 and at the end the graph has taken on a shape similar to the bend of an arm, which we can roughly determine to be the elbow point.

### Cluster and Plot

We can start by clustering all features and then reduce to 2 dimensions for plotting.

A diagram of a number of dots

Description automatically generated with medium confidence

We can find that the clusters are more centralised and do not show a good ability to classify clusters. So we decided to reduce Dimensions before clustering.

A diagram of a number of dots

Description automatically generated with medium confidence

We can find from the figure that this time the clusters are classified better and the main clusters are separated. And we can find two clusters with less data on top of the image, this may be because we set the classification to 7 clusters, we can merge it into one cluster, which is k=6.

### Examine Clusters

A collage of images of anime characters

Description automatically generatedAfter cluster classification, we can judge whether the cluster classification is successful or not by checking the representative images of each cluster, by means of images.

We can see from the picture that each group of pictures basically has a distinct style, such as Cluster4 which is a black and white comic book style. However, there are also pictures in the classification that don't quite match such as the fourth picture of Cluster5, other pictures have obvious black and red backgrounds, while it has a blue sky background. This shows that there is still room for improvement in the model.

## UMAP

In this notebook, we will try UMAP using [Rei | Evangelion] ( https://www.pinterest.com/salmon00\_601/rei-evangelion/) image datasets of different kinds (manga, original drawings, emoticons).UMAP can map high dimensional data to low dimensional space while preserving the local structure of the data, which can help us to cluster the image data.

The source code is downloaded from the URL of the image obtained through the API and connected to the model. I already had a database of images in a local file, so I modified the code to reference the local file.

def collect\_data(directory\_path, max\_images=500):

# Use glob pattern matching to find all image files

pattern = os.path.join(directory\_path, '\*.jpg')

image\_paths = glob.glob(pattern)

# Limit the maximum number of images

image\_paths = image\_paths[:max\_images]

if not image\_paths:

print('No images found. Check the directory path.')

return []

print(f'Found {len(image\_paths)} images. Processing...')

# Here we simply return the paths of the images, or you can further process as needed

return image\_paths

# usage

directory\_path = 'data/Rei | Evangelion'

images = collect\_data(directory\_path)

print(f'Collected {len(images)} images.')

### UMAP Visualisation 1: Colour palettes

The first UMAP visualisation technique we tried was palette-based.

A collage of images

Description automatically generated

We can see from the diagram that the images are mostly in RGB colours have a good classification.

### UMAP Visualisation 2: ImageNet Features

Next we will perform UMAP visualisation using ImageNet features to reduce the original image to 1000 semantic dimensions

.A collage of images

Description automatically generated

From the figure, we can find that most of the pictures with convergent painting style and convergent colours are in the same area, which can assist our values to show the distribution of image features, and we can find that some of the images in our image dataset are duplicated from where the images are clustered in the figure.

### Code

All code is in the GitHub repository https://git.arts.ac.uk/23009764/intro-to-ds-portfolio

Most code is started by prompting ChatGPT then tailoring for specific use cases