Fall 2022 MAT265 Clustering & Dimensionality Reduction

- Shaokang Li

Introduction

Clustering and Dimensionality Reduction are 2 effective approaches used in data analysis. Clustering is often used to see if there's any grouping pattern in the data, while dimensionality reduction are helpful in visualizing high-dimesional data.

In this project, I collect 3 dimensional data for each of the title in database. `Number of copies`, `Number of checkout`s and `average borrow duration`. Using clustering and dimensionality reduction algorithm, I am able to find some patterns with respect to the data and also perform data visualization in a 2D plane.

Description:

The whole process is divided into these steps:

Collect data using MySQL queries

- Query below is used to retrieve 3 numerical data for each title.
- To make the query faster, I add a restriction to CD category and a time range starting from 2010.

```sql
SELECT
    count(cout) as checkouts,
    title,
    count(distinct itemNumber) as copies,
    AVG( (DATEDIFF(cin, cout))) as avg_duration
FROM spl_2016.inraw
where
    itemtype in (
        'arcd',
        'nacd',
        'jrcd',
        'accd',
        'cacd',
        'cccd',
        'jccd',
        'nccd'
    )
    and cout > '2010-01-01'
group by title
order by
    checkouts,
    copies,
    avg_duration;
```

> See data.csv for the data

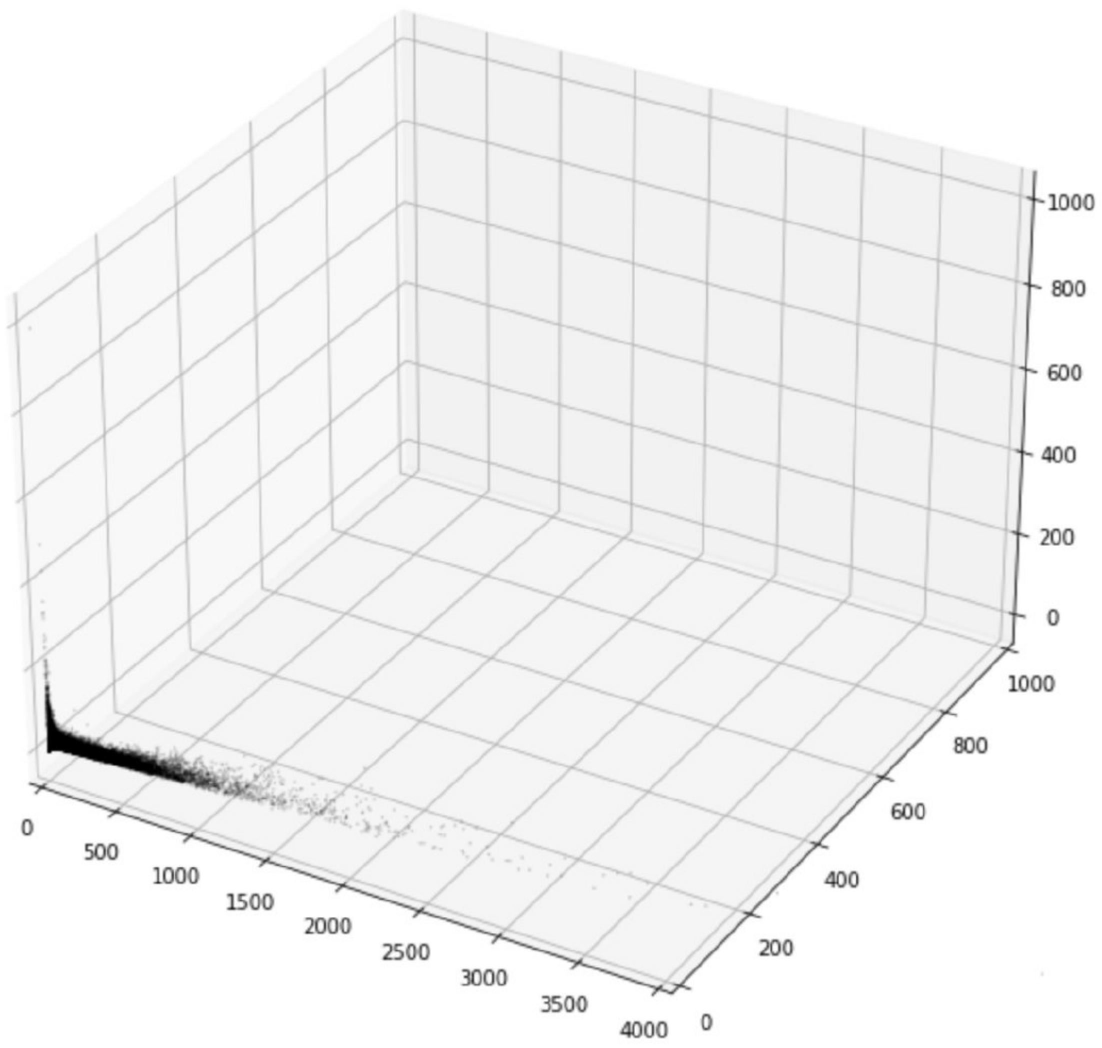
Perform Dimensionality Reduction Algorithm with Python (sci-kit learn library)

- A common dimensionality reduction algorithm use in data analysis is called Principle Component Analysis. It is a linear algebra approach.
- Applying the approach to the dataset could give us some insights on the principle component (important factor) of the dataset. And also helps with data visualization on 2D plane.

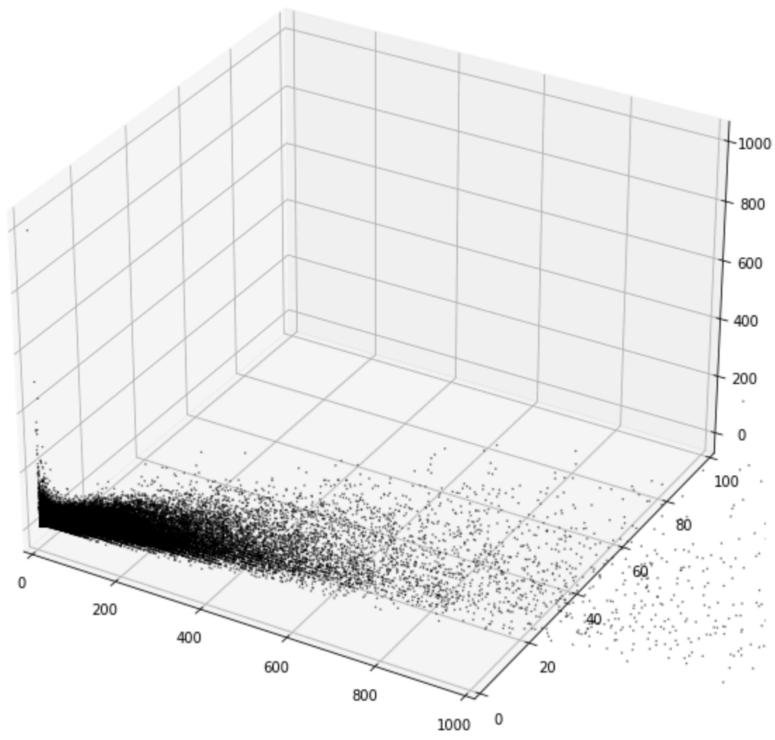
```python
import matplotlib

ax = plt.axes(projection='3d')
# Data for a three-dimensional line
zline = np.linspace(0, 15, 1000)
yline = np.linspace(0, 15, 1000)
xline = np.linspace(0, 15, 1000)
ax.plot3D(xline, yline, zline, 'gray')

# Data for three-dimensional scattered points
ax.scatter3D(X.checkouts, X.copies, X.avg_duration, c='black', cmap='Greens');
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(18.5, 10.5)
```

We can directly plot the original data in 3D space, it looks like this on with a broad scope:

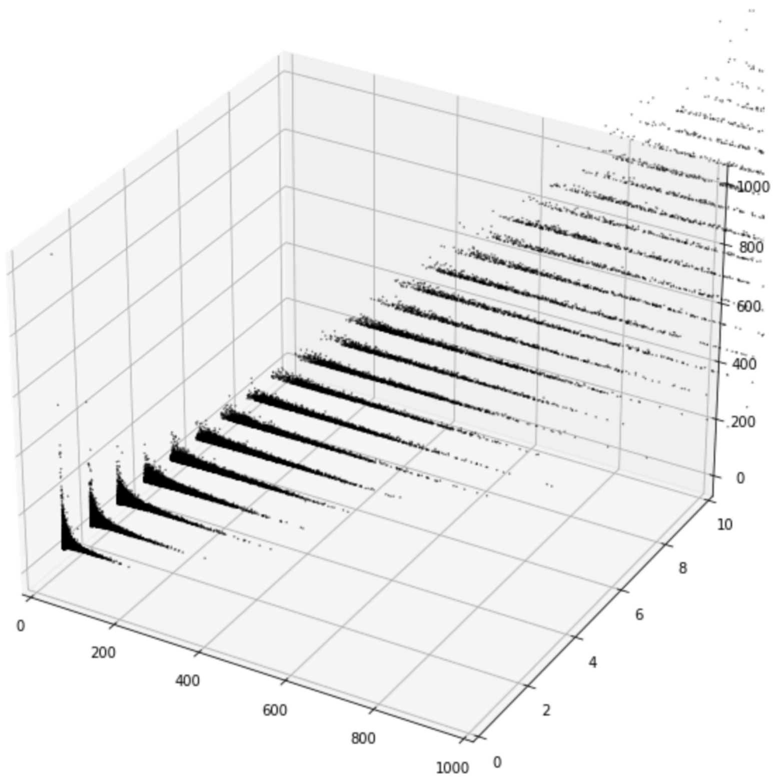
Most of the data lies within a range of 0~200 copies, however, the checkouts varies from nearly 0 to 4000+.



If we take a closer look, there is a radiating pattern in the plot.



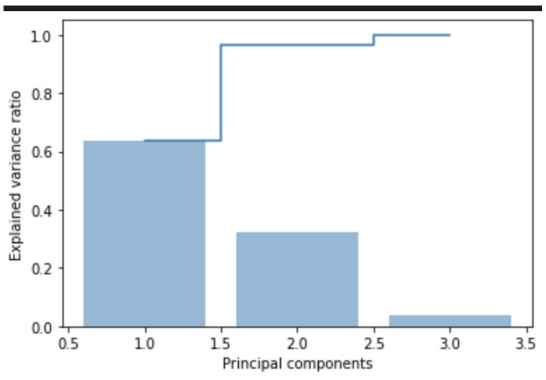
If we narrow down the copies from 0 to 10. We may find an interesting pattern in the distribution of the data. Data within the same cluster are of the same number of copies. Their checkouts are relatively low. And the average borrow duration is longer. As the copies increases, their checkouts are higher and the average borrow duration is shorter.



Such plot can be hard to understand if the range isn't correctly set. An 2D graph can be easier and intuitive to understand. To convert a 3D space into 2D plane, we need to apply PCA method to the dataset.

```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

X = df[['checkouts','copies','avg_duration']]
sc = StandardScaler()
X_std = sc.fit_transform(X)

from sklearn.decomposition import PCA

pca = PCA()
X_pca = pca.fit_transform(X_std)
print(pca.explained_variance_)
print(pca.explained_variance_ratio_)
plt.bar(range(1, len(pca.explained_variance_ratio_)+1),
pca.explained_variance_ratio_, alpha=0.5, align='center')
plt.step(range(1, len(pca.explained_variance_ratio_)+1),
np.cumsum(pca.explained_variance_ratio_), where='mid')
plt.ylabel('Explained variance ratio')
plt.xlabel('Principal components')

plt.show()
```

To compress the data into 2D plane. We need to find 2 eigen vectors of the covariance matrix by their eigen value. And the 2 eigen vectors are the principle components of the dataset. Principle component is mathematically defined as along which, data have the largest variance. Loosely speaking, 2 principle components represents 2 distinct qualities of the dataset, lowering the correlations between different attributes. (e.g. `checkouts` and `copies` can be positively correlated)

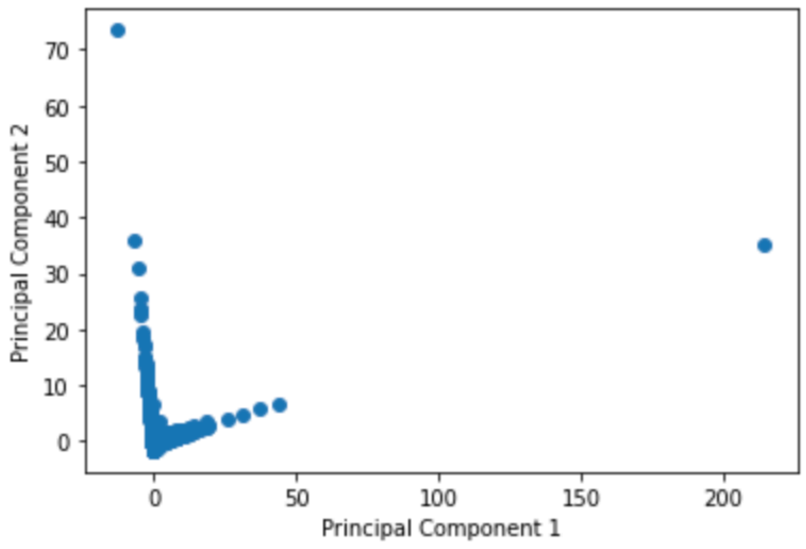


> PCA explained variance ratio.

Graph above indicates 2 principle components explained 96% of the variance of the dataset. Compress the data into a 2D plane will preserve 96% of the information (variance) of the original data.

```python
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_std)
plt.scatter(X_pca[:, 0], X_pca[:, 1])
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

Now we can perform the dimenionality reduction by matrix projection. Each datapoint now has 2 attributes, `principle component 1` and `principle component 2`. And their distribution looks like this on the plane.



Each principle component is a weighted sum of the original attributes. And on the 2D plane, the pattern is easier to observe.

There are mainly 2 groupings of the data: Some datapoints have higher `principle component 2` and low `principle component 1`, while others are have high level `principle component 1` and low-medium level `principle component 2`.

One question to think about here is: what does these principle components mean?

Note that each principle component is a weighted sum of the original attributes. We can look into the composition of the component:

```python
pcs = np.array(pca.components_) # (n_comp, n_features)

df_pc = pd.DataFrame(pcs, columns=['checkouts','copies','avg_duration'])
df_pc.index = [f"{c}-th Component" for c in['One', 'Two']]
df_pc.style\
    .background_gradient(cmap='bwr_r', axis=None)\
    .format("{:.2}")
```

	checkouts	copies	avg_duration
One-th Component	0.7	0.7	-0.16
Two-th Component	0.11	0.12	0.99

`principle component 1` equals to 0.7 checkouts and 0.7 copies and -0.16 average durations.

`principle component 2` equals to 0.11 checkouts and 0.12 copies and 0.99 average durations.

In other word, `principle component 1` represents such quality: Many checkouts and abudant copies, and the borrow duration is usually shorter. Loosely speaking, it indicates the popularity of an item.

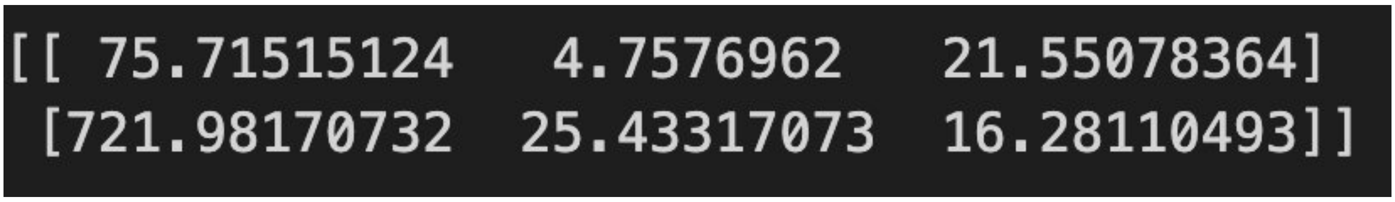
`principle component 2` represents such quality: It contributes some to the checkouts and copies. However, the borrow duration is usually longer. Loosly speaking, it mainly indicates the borrow duration of an item.

If we look back at the 2D graph, the boarderline of 2 different grouping are clear. We can just check it's location on the graph. If it's on the left-upper side, it's unpopular and has a long borrow duration, otherwise, it's popular and has a shorter borrow duration.

K-means Clustering

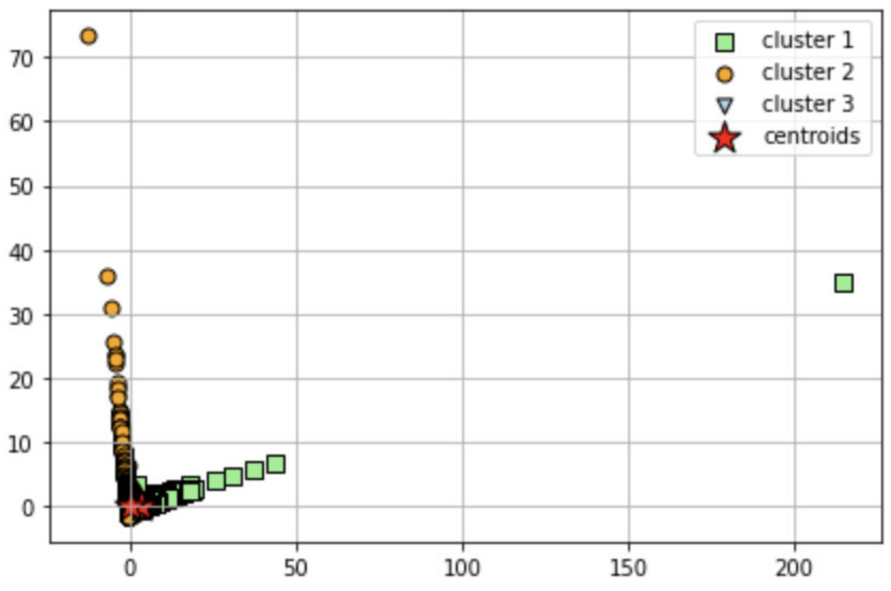
Another way to find pattern in the dataset is to use clustering algorithm, this algorithm may tell us how many different types of data are in the dataset. And often the centroids of the dataset represent the typical value of the data within the group.

```python
from sklearn.cluster import KMeans
import numpy as np
from sklearn.cluster import KMeans

km = KMeans(n_clusters=2,
            init='random',
            n_init=10,
            max_iter=300,
            tol=1e-04,
            random_state=0)

y_km = km.fit_predict(X)
print(km.cluster_centers_)
```



```
[[ 75.71515124   4.7576962   21.55078364]
 [721.98170732  25.43317073  16.28110493]]
```

The 2 centroids mainly differs in checkout times and copies. If we apply this approach on a 2D plane with data from PCA method, it looks like this:



In this case, the 2 centroids are very close to each other, and they are very close to the point (0,0). This indicates that most of the items are unpopular and has a shorter borrow durations.

Conclusions:

By using Clusterings and Dimensionality Reduction algorithms on the database, I find that:

- The attributes `checkouts` `copies` `borrow_duration` can be reduced to 2 principle components. And one represents the popularity and the other represent the borrow_duration. This means the `checkouts` and `copies` are highly correlated, and they are not very correlated with the `borrow_duration`.
- By reduce the dimensionality, we are able to project the data on a 2D plane, and easier to find pattern.
- By checking the composition of the principle components, we are able to know what's the distinct quality in the dataset. In this case, one is `popularity`, one is `borrow_duration`.
- Most of the datapoints are unpopular and has a shorter borrow durations since both centroids are positioned near (0,0).

Reference:

1. Elbow Method in K-means: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/
2. K-means documentation: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
3. Principle Component Analysis (in zh-tw) https://leemeng.tw/essence-of-principal-component-analysis.html