

WSM Project 3

110753201 曹昱維
110753132 馬行遠
110753207 林依樺
111753124 巫謹任

1) Introduction	P2
2) Data Preprocessing	P2
a) Exploring Data	
b) Feature Engineering	
i) Filtering the DateTime	
ii) Filtering the candidate item	
iii) session preprocessing	
iv) one-hot encoding	
c) Analyzing and Visualizing the Data	
i) Clustering	
3) Training Model	P7
a) TF-IDF	
b) Item-CF	
c) Ensemble Model	
4) Conclusion	P9
5) Teamwork	P10

1) Introduction

This Project is to participate in the RecSys Challenge 2022 competition.

- Task: Fashion Recommendation
 - Given: User sessions, descriptive labels of the items, purchase data, and content data about items.
 - Output: The item will be bought at the end of the session.

2) Data Preprocessing

a) Exploring Data

The data sources are provided by RecSys Challenge 2022, and the values have been sorted by the organizer, so the steps of processing missing values, checking outliers, and data integration are omitted here.

When checking the data, we found that in the item_features.csv file, there may be repeated feature_category_id in one item. In other words, there may be two or three colors of a piece of clothing, in which case multi-value will appear. So let's first explore which item feature_category_id has a repeating value:

```
• feature_category 最大的重複次數(該feature_category 在同一個item中 會有複數值)
  ◦ less item_features.csv | awk -F "," '{print $1, $2}' | sort | uniq -d -c | awk -F " " '{if ($3== 1 ) {print $1, $3}}' | sort | uniq
    ■ 將綠色的部份改成 1 | 28 | 30 | 4 | 46 | 53
    ■ feature_category : max number of multiple value
      • 1 : 2
      • 28 : 3
      • 30 : 8
      • 4 : 4
      • 46 : 2
      • 53 : 2
```

b) Feature Engineering

i) Filtering the DateTime

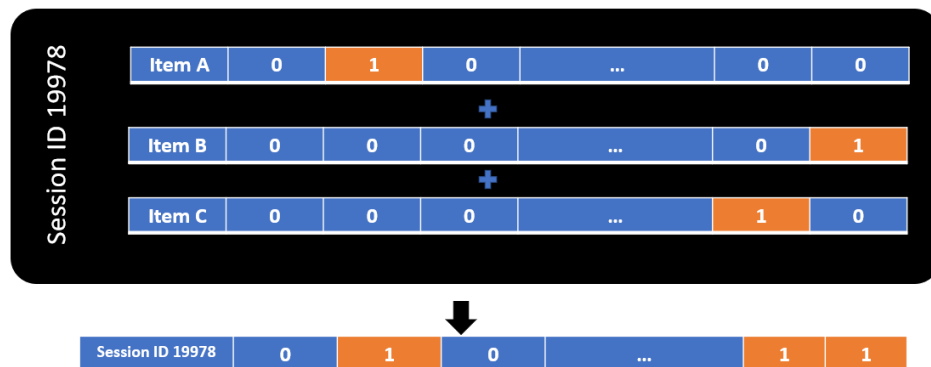
- Select data and candidates by filtering dates during training
e.g. We only select the session of 2021/5/15~2021/5/31 to calculate TF-IDF, and the candidate part also selects the items that have appeared in 2021/5/15~2021/5/31.

ii) Filtering the candidate

- Some models may output the recommended items beyond the candidate item set, so we adjust the outputs by filtering the candidate
e.g. We filter the outputs of Item-CF with the candidate itemset.

- Linear superposition

- Linear stacking of items in the same session. The main purpose of this method is to combine all items in the same session to make a session into a vector.



iv) One-hot encoding

Since the two columns of `feature_category_id` and `feature_category_value` are of category type and have no sequential relationship, these features are converted into binary features. Each category is a new feature, if it is this category, give 1, if not, give 0. However, because the `category_id` of an item may have multi-values, we can deal with multi-values in the following ways:

- I. Feature expanded from 73 columns to 904 columns

We combined each `feature_category_id` and `feature_category_value` group into a new feature value. For example, if `feature_category_id=1` and `feature_category_value=60`, the new feature name would be `1_60`. Refer to Figure 1 for an example.

- II. After processing multi-value, expand from 73 columns to 88 columns

We ignore `feature_category_value` here. If `feature_category_id` occurs twice in an item, the number of occurrences is listed after `feature_category_id`. For example, if there are two `feature_category_id=4` items in `item3`, the number `feature_category_id` will be added to two columns: `4_1` and `4_2`. See Figure 2 for an example.

feature_category_id	feature_value_id	feature_name
1	60	1_60
1	143	1_143
1	358	1_358

Figure 1.

item_id	feature_category_id
30	16_1
30	57_1
30	4_1
30	68_1
30	61_1
30	8_1
30	55_1
30	4_2

Figure 2.

III. Combine : $73+904=977$

The results obtained in the preceding 1 are combined with the one-hot-encoding results of 73 features in the original data. Finally, 23691 rows \times 977 columns can be obtained, as shown in Figure 3.

	10_147	10_159	10_184	10_217	10_22	10_287	10_361	10_407	10_464	10_561	...	64	65	66	67	68	69	70	71	72	73
item_id																					
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	1.0

Figure 3.

c) Analyzing and Visualizing the data

We use the multi-value processed data mentioned above for data analysis. For visualization and computational efficiency, we use PCA to reduce the dimension of the data and divide it into two tasks for analysis:

- Clustering feature
 1. Goal: To group the encoded 88 features and compare the similarity between features.
 2. Data dimension: We applied 88 rows \times 23691 columns data on PCA, and then an 88rows \times 2 columns matrix was obtained. Explained variance ratio is 0.536 and 0.104.
- Clustering item
 1. Goal: Group all items according to the feature to find items with similar properties.
 2. Data dimension: We applied 23691rows \times 88 columns data on PCA, and then a 23691rows \times 2 columns matrix was obtained. Explained variance ratio is 0.187 and 0.1.

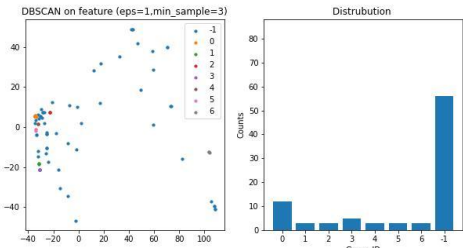
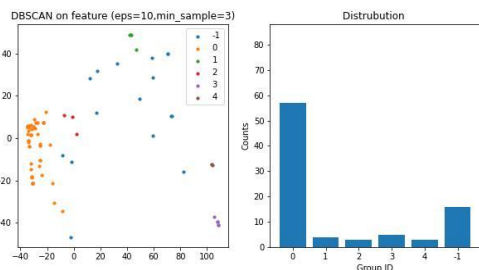
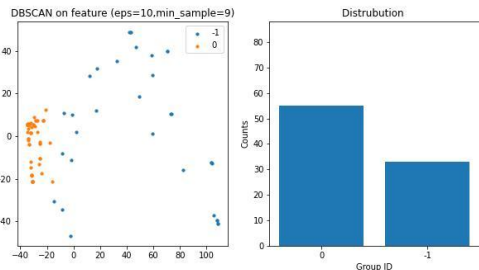
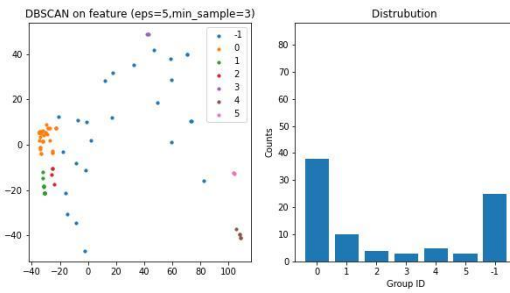
We tried to use DBSCAN and K-means for clustering.

1) DBSCAN

DBSCAN is a density-based clustering algorithm. Good at identifying noise in data without specifying clustering categories.

→ DBSCAN on feature

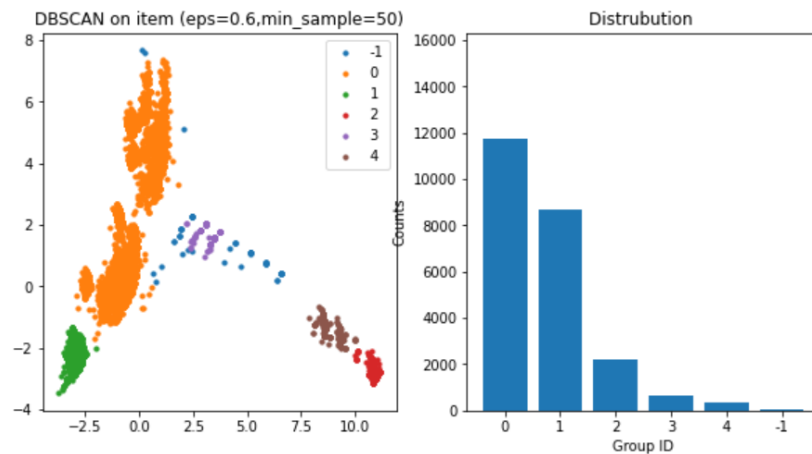
The filtering test results are as follows:

eps	min_sample		
1	3		Most of the data is considered as noise
10	3		Grouping results are too concentrated
10	9		We tend to classify data into specific groups and outliers
5	3		Grouping results with better results

This clustering result is not ideal. No matter the density requirement of the clustering algorithm is adjusted, there will be many groups with large differences in the number of members (even noise occupies the majority) or most features will be divided into the same group.

→ DBSCAN on item

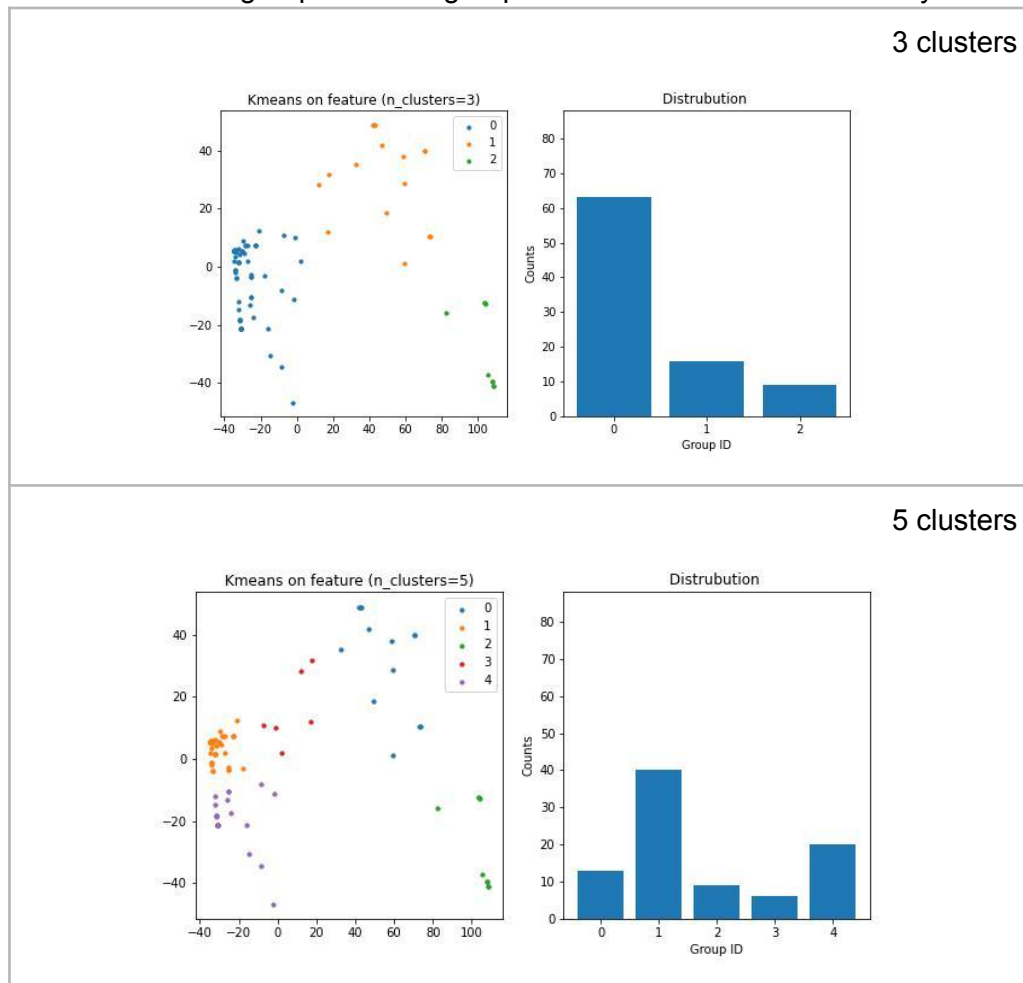
Judgment of outliers and grouping results are greatly affected by parameters.



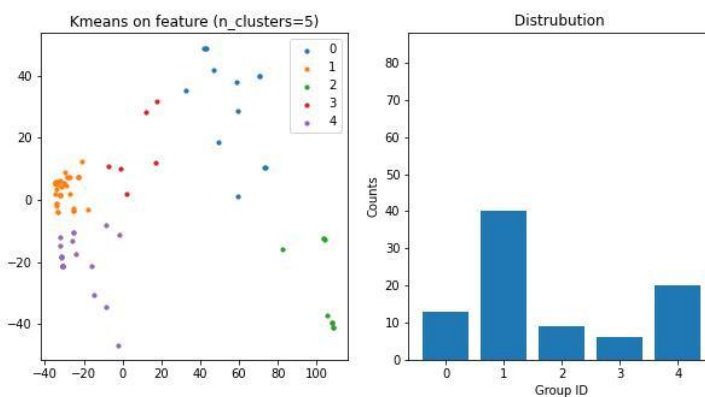
2) K-means

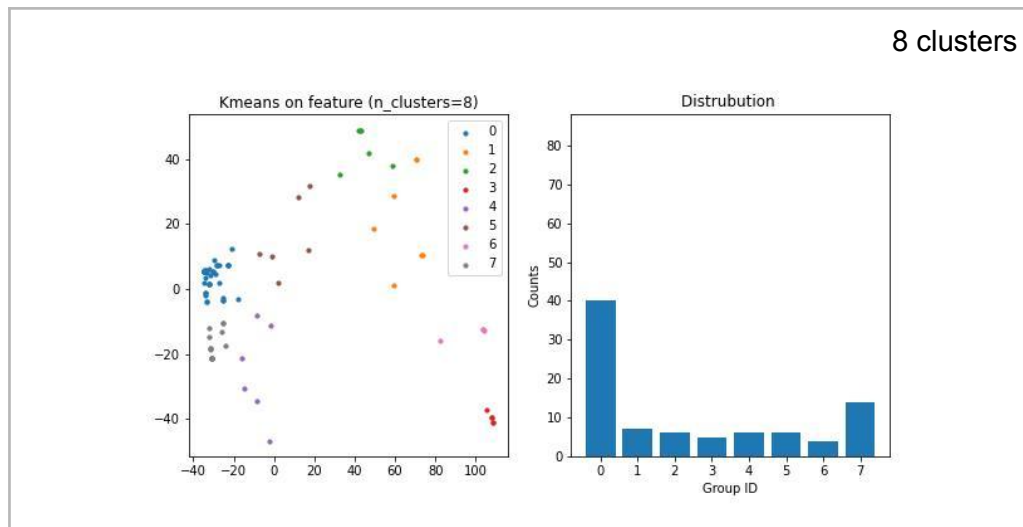
→ K-means on feature

It is better able to group data into groups that match what the human eye sees.



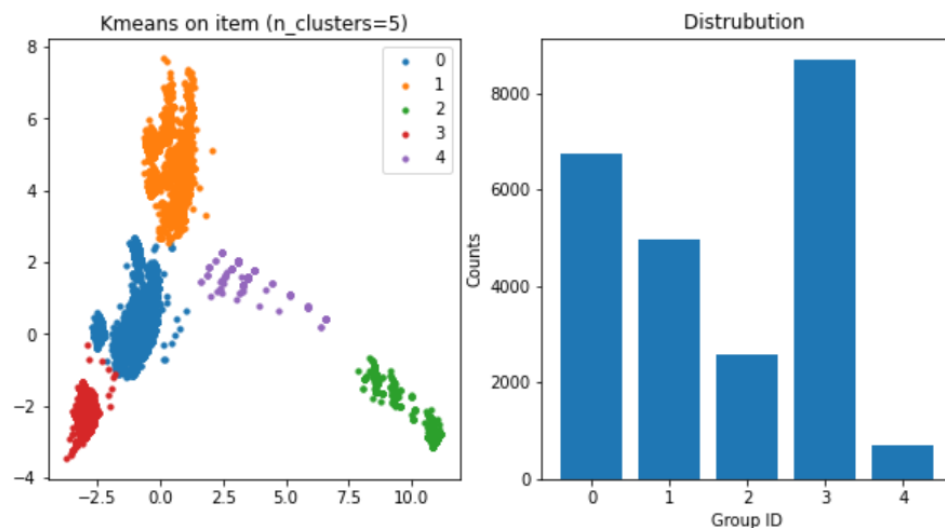
5 clusters





→ K-means on item

It is better able to group data into groups that match what the human eye sees than DBSCAN results.



3) Training Model

- TF-IDF method

We use the TF-IDF method taught in class to test the data, and use cosine similarity to calculate the similarity. We regard each session as a document and encode the attributes of the item in a one-hot-encoding way.

ID	Method			Score
	Period of train data	Feature Engineer method	model	Leader Broad
1	2021/5/15~ 2021/5/31	<ul style="list-style-type: none"> • one-hot(904 columns) • session preprocess • Filtering the DateTime 	TF-IDF	0.04953

ID	Method			Score
	Period of train data	Feature Engineer method	model	Leader Broad
2	2021/5/1~ 2021/5/31	<ul style="list-style-type: none"> one-hot(904 columns) session preprocess Filtering the DateTime 	TF-IDF	0.04867
3	2020/1/1~ 2021/5/31	<ul style="list-style-type: none"> one-hot(904 columns) session preprocess 	TF-IDF	0.04770

○ Item-CF

- We regard each session as a user and calculate the similarity of item i and item j when both of them co-occur in a user's list .
- Parameter K : We compute the utility function $p(u,j)$ by considering the set of K items most similar to item j .

ID	Method			Score
	Period of train data	Feature Engineer method	model	Leader Broad
1	2020/1/1~ 2021/6/30 (only leaderboard)	<ul style="list-style-type: none"> Filtering the candidate 	Item-CF with K=4	0.14584
2	2020/1/1~ 2021/6/30 (only leaderboard)	<ul style="list-style-type: none"> Filtering the candidate 	Item-CF with K=2000	0.16909
3	2020/1/1~ 2021/6/30 (only leaderboard)		Item-CF with K=2000	0.16486
4	2021/1/1~ 2021/6/30 (only leaderboard)	<ul style="list-style-type: none"> Filtering the candidate Filtering the DateTime 	Item-CF with K=2000	0.17068
5	2021/1/1~ 2021/6/30 (only leaderboard)	<ul style="list-style-type: none"> Filtering the candidate Filtering the DateTime 	Item-CF with K=8000	0.17071

○ Ensemble Model

- We combine Item-CF and TF-IDF models into an ensemble model with a voting ratio R . The ratio indicates the contribution of two models.
- Re-rank recommended item with the score:

$$\text{score}_i = 1 * \frac{1}{\text{rank}_{i,\text{itemcf}}} + R * \frac{1}{\text{rank}_{i,\text{tfidf}}}$$

$$\frac{1}{\text{rank}_{i,\text{model}}} = \begin{cases} \frac{1}{\text{rank}_{i,\text{model}}} & \text{if item } i \text{ exist} \\ 0 & \text{if item } i \text{ not exist} \end{cases}$$

ID	Method			Score
	Period of train data	Feature Engineer method	model	Leader Broad
1	2021/5/15~ 2021/5/31	<ul style="list-style-type: none"> one-hot(904 columns) session preprocess 	TF-IDF	0.049539
2	2020/1/1~ 2021/6/30 (only leaderboard)	<ul style="list-style-type: none"> Filtering the candidate Filtering the DateTime 	Item-CF with K=8000	0.170718 ²⁴²²² 743115
3			Ensemble * ratio = 0.02 * Item-CF with K = 8000 * TF-IDF	0.170718 ³⁶⁸⁷¹ ¹⁶⁸⁶¹⁴

4) Conclusion

In this final project, we first spent some time to understand the data, and we also thought about analyzing the correlation between feature, item, feature_category_id and feature_category_value from the perspective of statistics. We want to measure whether the original data types of different characteristic data are continuous or categorical. But when we consider that the data are hashed values, the data may not be statistically meaningful.

There are also discussions on using other Encoding Methods, such as Leave-One-Out Encoding for processing and so on. At the same time, we also use Data Mining, such as clustering and other methods. Although we encountered thinking and technical obstacles in the process (such as code optimization for preprocessing), we also used the suggestions provided by the teaching assistants in class (the teaching assistants provided preprocessing solutions and different thinking perspectives) and continued to experiment.

In terms of models, we have completed a total of 3 model submission results, TF-IDF model, Item-CF model, and ensemble model. The performance of the TF-IDF model is 0.4953, which is worse than expected. The overall performance of item-CF mode is better than that of TF-IDF, which is 0.17182. The overall forecast was slightly improved to 0.17184 after the Ensemble model. The overall results are as follows:

Model	TF-IDF	Item-CF	Ensemble
Best performance	0.04953	0.1707182	0.1707184

We have also tried machine learning models. However, the training model took a long time (estimated to be more than three days), which may require further performance optimization, so it has not been completed yet.

We hope that our model results can be improved in the following ways:

- Use different encoding methods, such as Leave-One-Out Encoding
- Due to the limited number of uploads, we compare the relationship between different encoding methods and the final model performance.
e.g. The effect of TF-IDF model + multi value and the current comparison and further reason analysis.
- Add the clustering information as a feature to the data for training
- Prediction using ML methods

We learned a lot during the course of this project and can apply what we learned in class to this project. Further understanding attempts to implement various theories. The challenges we encountered in the process of implementation were invaluable experiences and gave us a better understanding of the field of data science.

5) Teamwork

110753201 曹昱維: Data Preprocessing & Training Model & Testing Model

110753132 馬行遠: Training Model & Clustering & Testing Model

110753207 林依樺: Data Preprocessing & Clustering & Presentation & Paper Report

111753124 巫謹任: Data Preprocessing & Presentation & Paper Report