POLITECNICO DI MILANO
DEI

Thesis Presentation

POLITECNICO
MILANO 1863

Tourism Personalized Recommender System Core Design

Chaofeng Guan

Politecnico di Milano

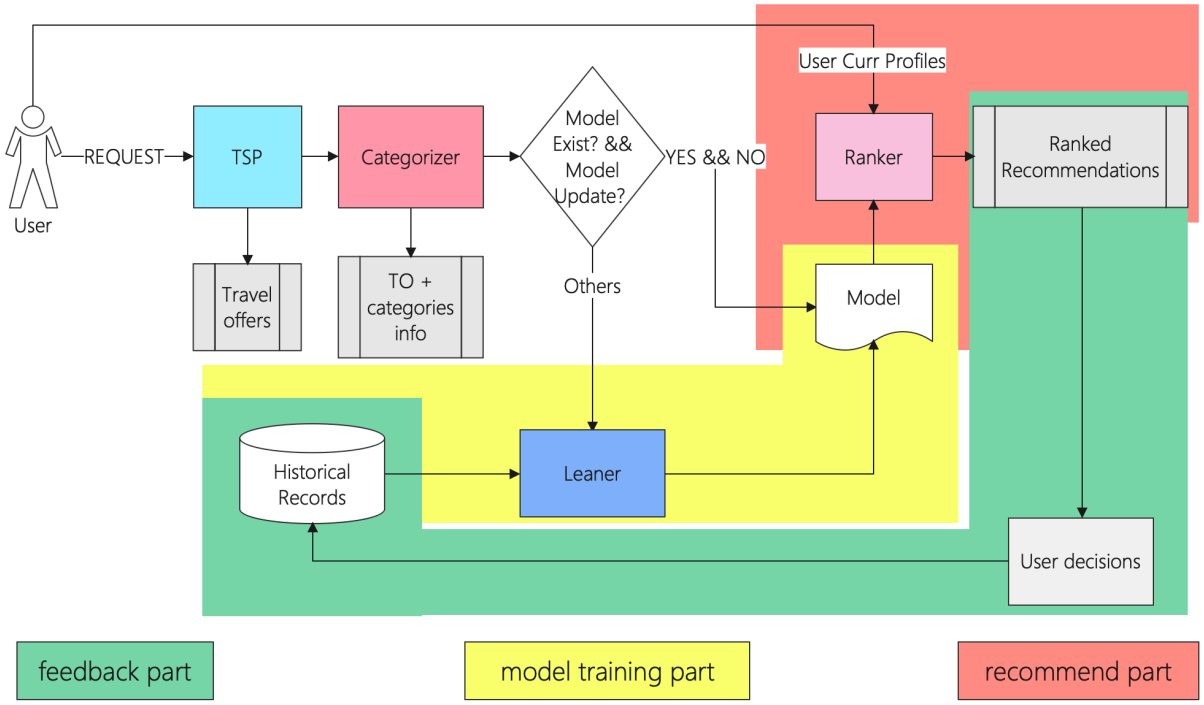
Summary

- Design Idea
- Program Running
- Results Display

Design Idea

Design Idea

The system high-level overview



Three parts of the system have been marked in different colors.

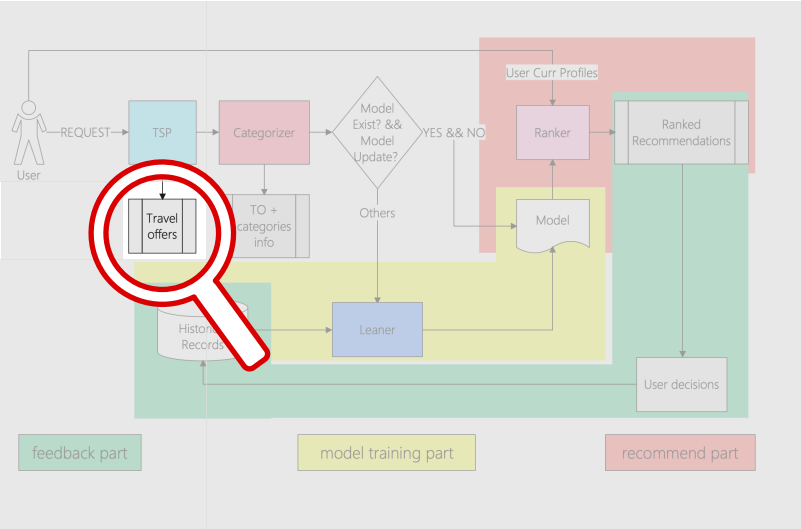
User's request will be accepted by the Travel Service Provider and a table of travel offers can be recommended will be given.

Categorizer will add the categories' points for each travel offer.

- Recommend part will use the specific user's model to calculate the scores of all the travel offers, and the top 30 travel offers will be recommended to the user.
- Model training part in charge of providing the user's recommender model.
- All the recommendations with the user's decision will be recorded into the database.

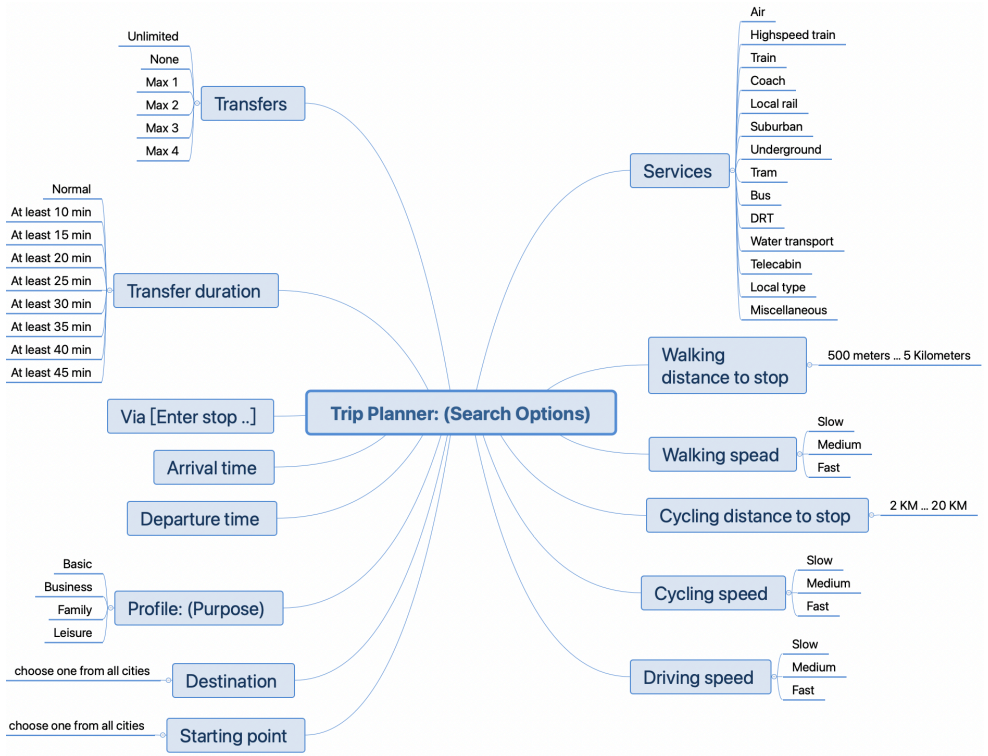
Design Idea

The table content of travel offers



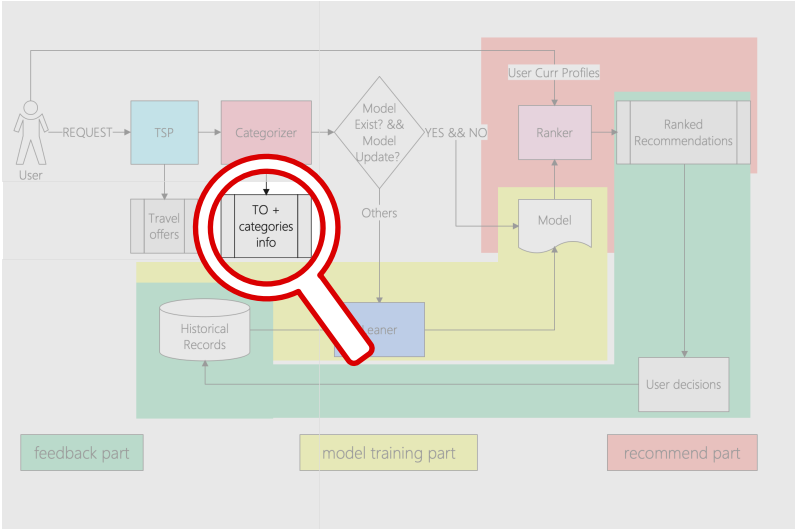
The user may has these requests on a travel offer, and TSP will provide the travel offers according to the request.

Therefore, the content of one travel record should have the right information.



Design Idea

Content of the table given by the Categorizer




```
CREATE TABLE `20200514_023200` (
  `Travel Offer ID` varchar(255) NOT NULL,
  `Quick` float DEFAULT NULL,
  `Reliable` float DEFAULT NULL,
  `Cheap` float DEFAULT NULL,
  `Comfortable` float DEFAULT NULL,
  `Door-to-door` float DEFAULT NULL,
  `Environmentally friendly` float DEFAULT NULL,
  `Short` float DEFAULT NULL,
  `Multitasking` float DEFAULT NULL,
  `Social` float DEFAULT NULL,
  `Panoramic` float DEFAULT NULL,
  `Healthy` float DEFAULT NULL,
  `Legs Number` int(255) DEFAULT NULL,
  `Profile` varchar(255) DEFAULT NULL,
  `Starting point` varchar(255) DEFAULT NULL,
  `Destination` varchar(255) DEFAULT NULL,
  `Via` varchar(255) DEFAULT NULL,
  `LegMode` varchar(255) DEFAULT NULL,
  `LegCarrier` varchar(255) DEFAULT NULL,
  `LegSeat` varchar(255) DEFAULT NULL,
  `LegLength` varchar(255) DEFAULT NULL,
  `Departure time` datetime DEFAULT NULL,
  `Arrival time` datetime DEFAULT NULL,
  `Services` varchar(255) DEFAULT NULL,
  `Transfers` varchar(255) DEFAULT NULL,
  `Transfer duration` varchar(255) DEFAULT NULL,
  `Walking distance to stop` varchar(255) DEFAULT NU
  `Walking speed` varchar(255) DEFAULT NULL,
  `Cycling distance to stop` varchar(255) DEFAULT NU
  `Cycling speed` varchar(255) DEFAULT NULL,
  `Driving speed` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Travel Offer ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Quick	0.4574
Reliable	0.6839
Cheap	0.2011
Comfortable	0.3352
Door-to-door	0.8713
Environmentally friendly	0.6428
Short	0.5837
Multitasking	0.1171
Social	0.9371
Panoramic	0.0617
Healthy	0.1401

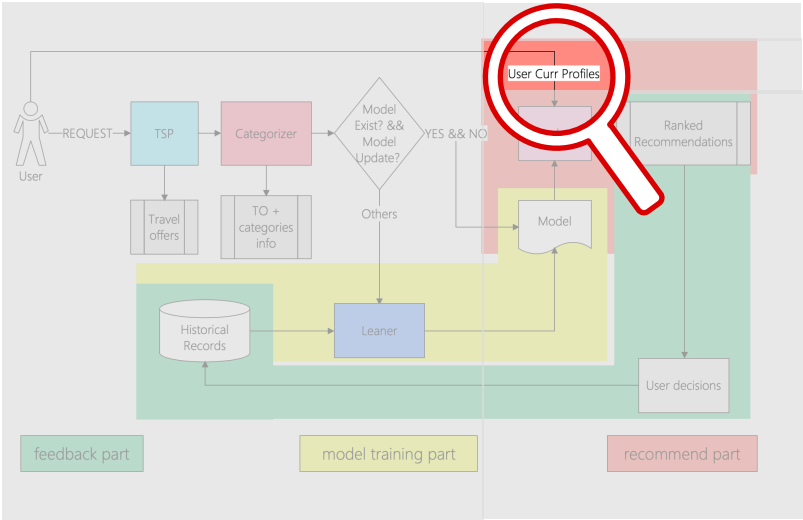
LegMode	['Funicular', 'Funicular', 'Intercity', 'Urban', 'Trolely Bus', 'Other', 'Bus', 'Park']
LegCarrier	['FlixBus', 'RegioJet', 'Trenitalia', 'Trenitalia', 'VBB', 'FlixBus', 'Renfe', 'FlixBus']
LegSeat	['Aisle', 'Large', 'Aisle', 'Window', 'Window', 'Large', 'Large', 'Aisle']
LegLength	[85, 52, 15, 23, 38, 55, 30, 78]

Besides the original information in Travel Offer table, Categorizer adds two parts of information for each travel offer in the table.

- RED block indicates the categories' points of the travel offer.
- GREEN block indicates the leg information for each travel offer.

Design Idea

Content of the User Current Profile

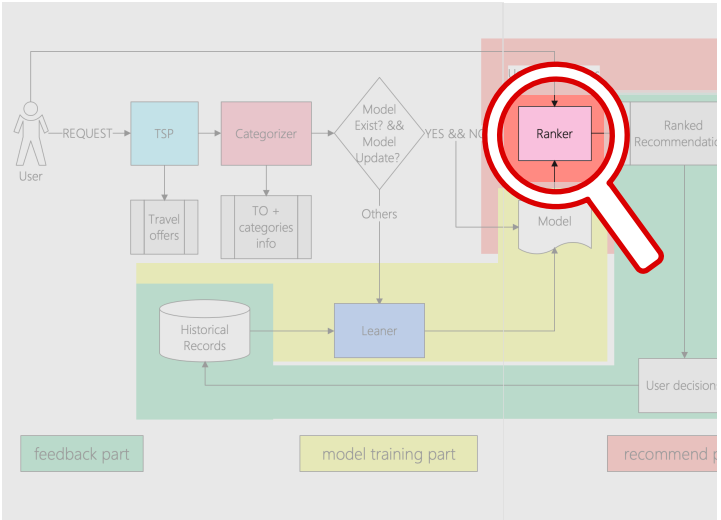


A full table of the user profiles records all the historical information, and the current status can be recognized by the TimeStamp.

TimeStamp	User ID	Date Of Birth	city	country	Loyalty Card	Payment Card
2021-05-01 01:12:58.436566	CLUSTERtrain_0	1971-07-08	Lisbon	Portugal	['Cartafreccia', 'Golden Card']	['Paypal']
PRM Type						
['Persons with blind or visual impairments', 'Wheelchair users in mainstreaming seat', 'Persons with impairments in their members / users of temporary wheelchair', 'Person with deafness or auditory impairments']						
Preferred means of transportation			Preferred carrier	Class	Seat	Refund Type
['Cable Way', 'Funicular', 'Urban', 'Metro', 'Other', 'Train', 'Toll', 'Park', 'Car Sharing', 'Trolely Bus', 'Ship', 'Tram', 'Coach']			[2.0, 3.5, 4.0, 5.0, 2.0, 4.0, 3.5, 0.0, 4.5, 3.0]	Economy	Large	Manual refund

Design Idea

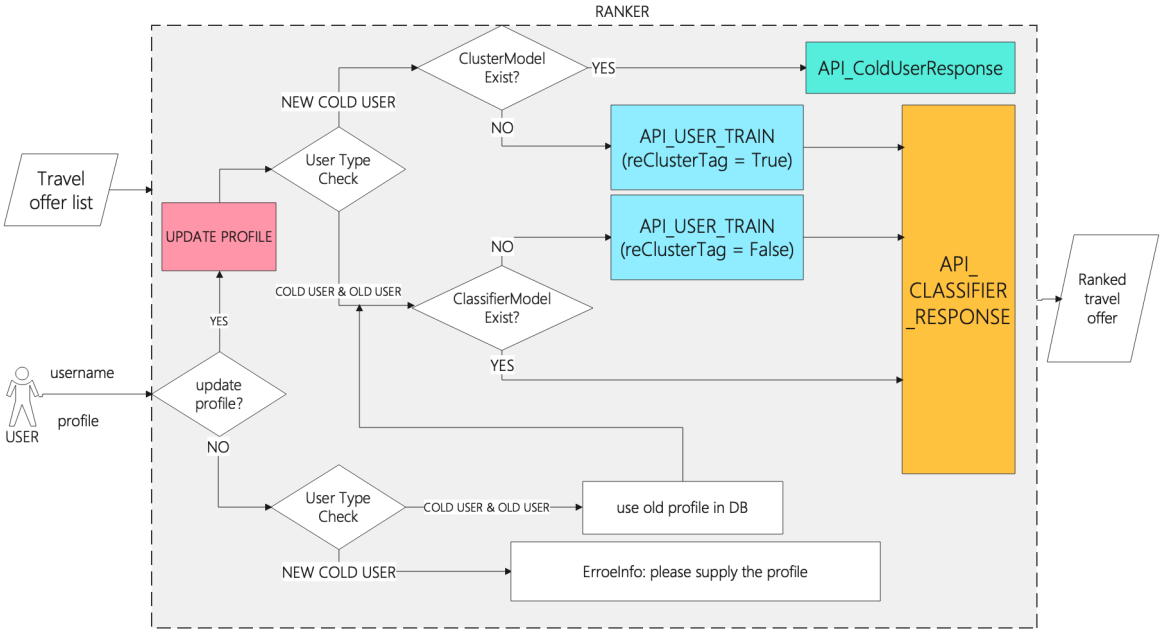
Ranker(1/2)



In the program, the logic before the ranker is also implemented in this block and details are shown in right.

The travel offer list table is given by Categorizer, and profile means the user current profile.

Step1 is checking the user current profile, fetch the new current profile if user updated their profile.

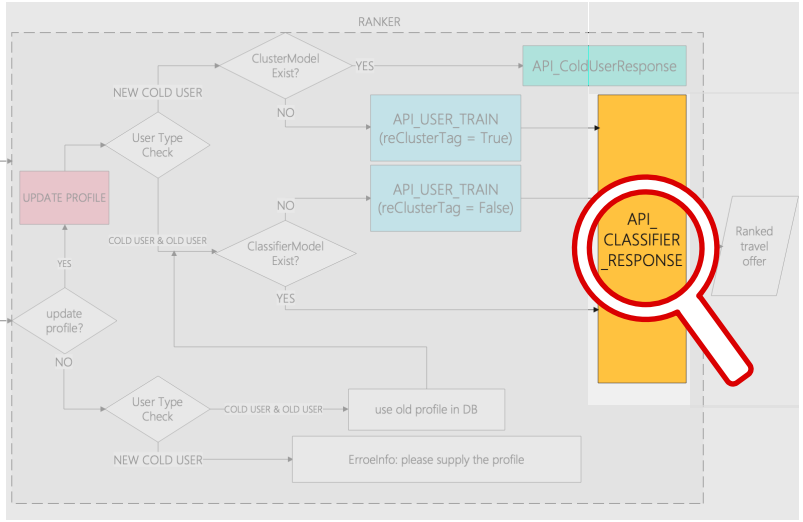


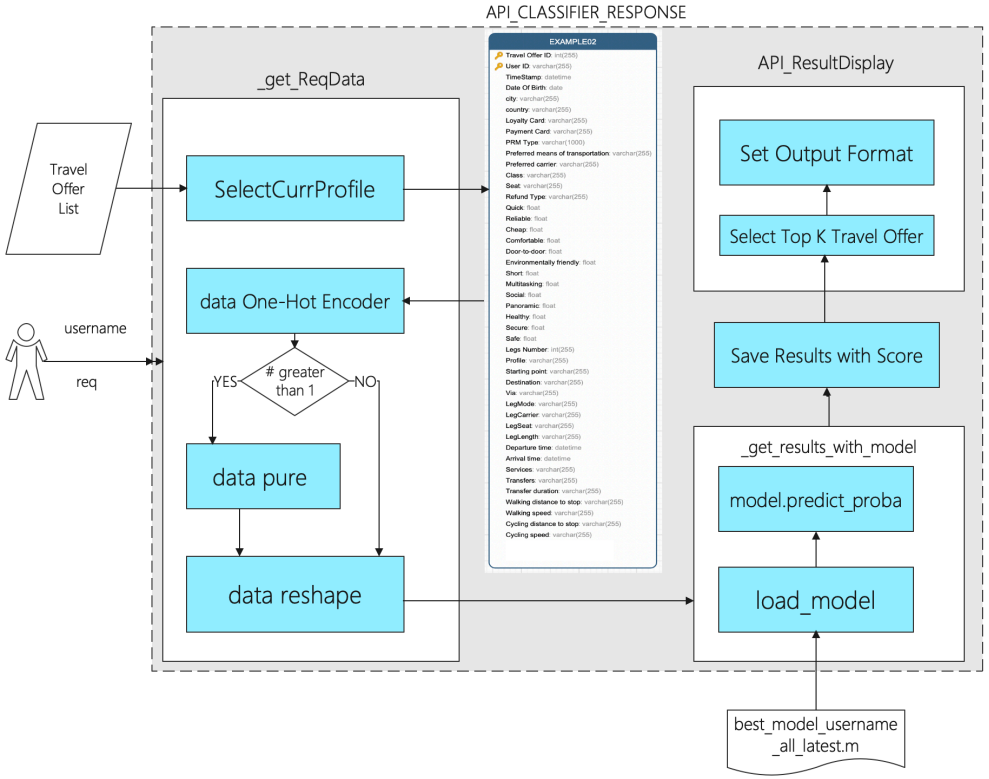
Step2 is checking the user type, different users will trigger different functions.

Only when the user has the recommender model and do not need to update the model, the response function will give the recommendations.

Design Idea

Ranker(2/2) - Core: response function





Travel offer list and user current profile will mix into the response data.

Data after one-hot encoding, deleting the same columns, reshaping the size and normalization will be put into the model and get the score for each travel offer.

A simply ranker can be used to rank all the recommendations based on the score.

Design Idea

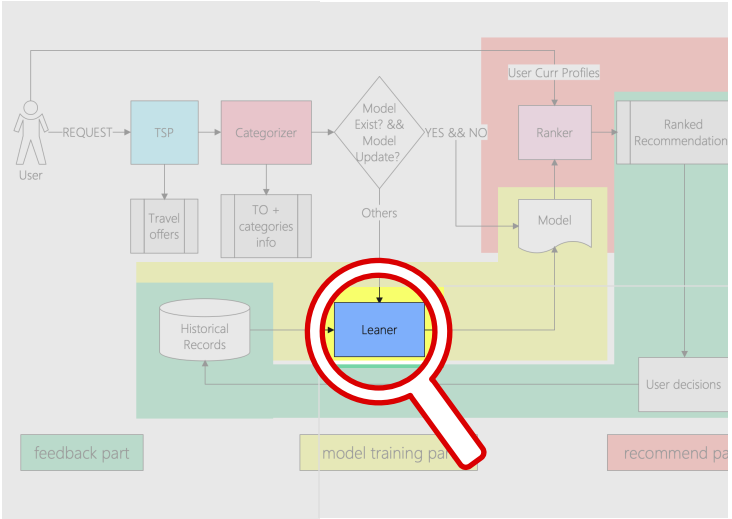
Content of Response Data

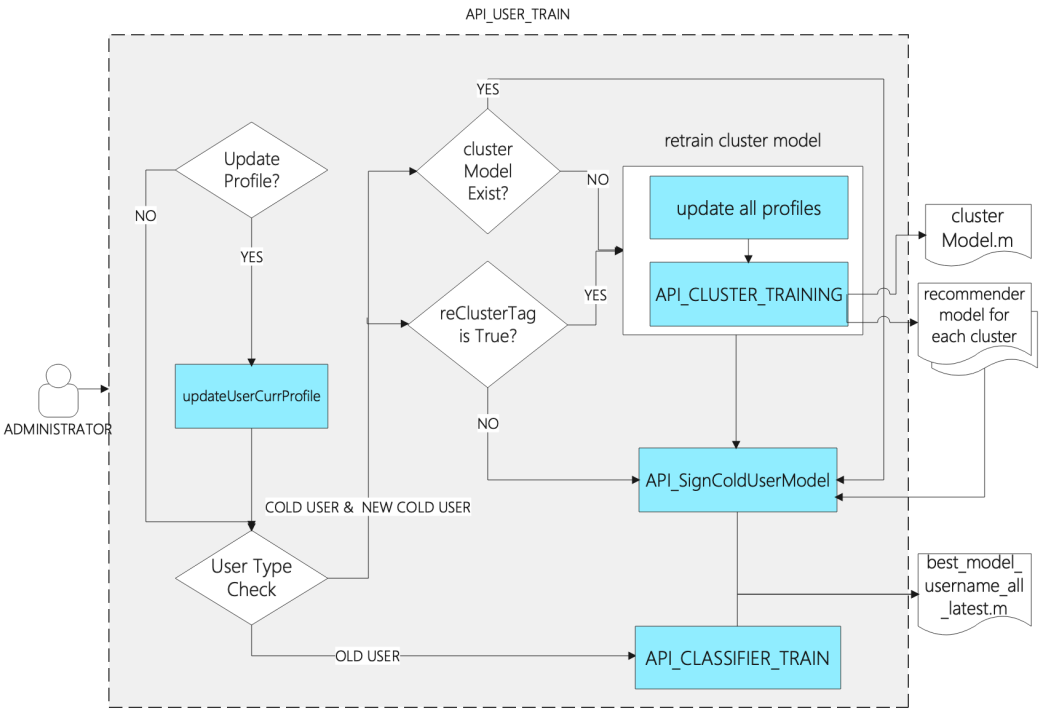
User Current Profile will be appended on every travel offer in the table given by Categorizer.
Then we get the response data for making recommendation

TimeStamp	User ID	Date Of Birth	city	country	Loyalty Card	Payment Card
2021-05-01 01:12:58.436566	CLUSTERtrain_0	1971-07-08	Lisbon	Portugal	['Cartafreccia', 'Golden Card']	['Paypal']
PRM Type						
['Persons with blind or visual impairments', 'Wheelchair users in mainstreaming seat', 'Persons with impairments in their members / users of temporary wheelchair', 'Person with deafness or auditory impairments']						
Preferred means of transportation			Preferred carrier	Class	Seat	Refund Type
['Cable Way', 'Funicular', 'Urban', 'Metro', 'Other', 'Train', 'Toll', 'Park', 'Car Sharing', 'Trolely Bus', 'Ship', 'Tram', 'Coach']			[2.0, 3.5, 4.0, 5.0, 2.0, 4.0, 3.5, 0.0, 4.5, 3.0]	Economy	Large	Manual refund
	Quick	Reliable	Cheap	Comfortable	Door-to-door	Environmentally friendly
	0.6922	0.9641	0.139	0.7825	0.4149	0.8703
Short	Multitasking	Social	Panoramic	Healthy	Legs Number	Profile
0.0921	0.1923	0.8833	0.5652	0.4101	7	Family
Starting point	Destination	Via		LegMode		
Dublin	Frankfurt	['Milan', 'London', 'Brugge', 'Warsaw', 'Praha', 'Oxford']		['Urban', 'Other', 'Intercity', 'Coach', 'Toll', 'Other', 'Cable Way']		
LegCarrier			LegSeat		LegLength	
['VBB', 'Renfe', 'Iberia', 'FlixBus', 'FlixBus', 'SNFC', 'KLM']			['Aisle', 'Large', 'Aisle', 'Window', 'Window', 'Window', 'Aisle']		[76, 25, 16, 19, 13, 53, 25]	
Departure time	Arrival time	Services				Transfers
2022-10-11 17:03	2022-11-20 16:58	['Air', 'Highspeed train', 'Bus', 'Suburban', 'Water transport', 'Coach', 'Miscellaneous', 'Telecabin', 'DRT']				Max 1
Transfer duration	Walking distance to stop	Walking speed	Cycling distance to stop	Cycling speed	Driving speed	
normal	2000m	Slow	10000m	Medium	Medium	

Design Idea

Learner(1/3)





First Step is checking the user type.

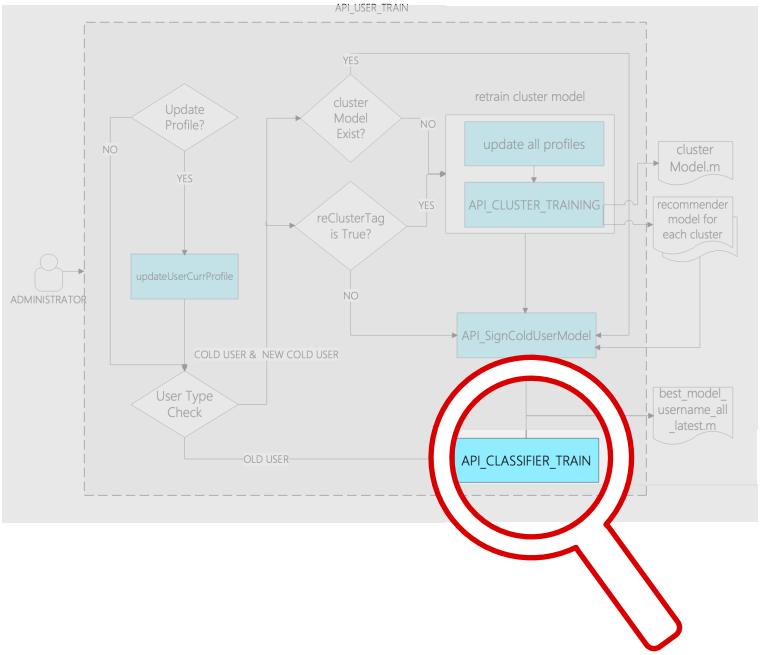
For a general user who has enough historical records in database, we use classifier train function to get the recommender model.

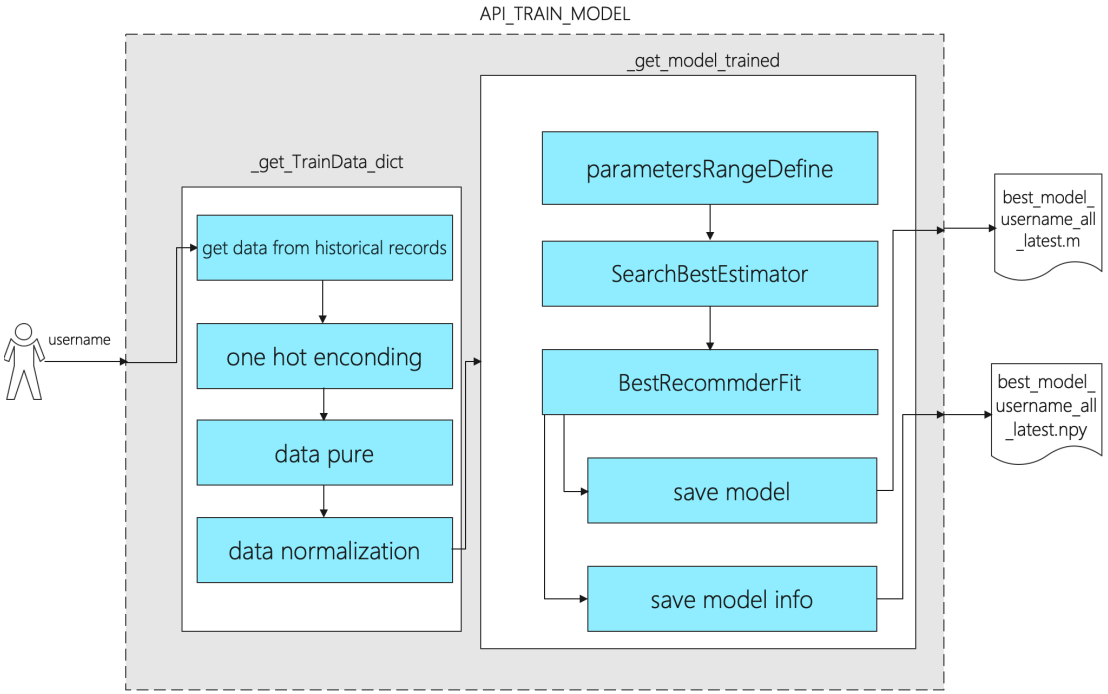
For a new user or a cold user who does not have enough historical records, we use cluster model to find the similar group users and sign the predefined recommender model for the user.

If the cluster model is not exist or system need to update the cluster model, system need fetch all the general users' profiles and use cluster training function to get the cluster model. The predefined models are also required to be trained at this stage.

Design Idea

Learner(2/3)



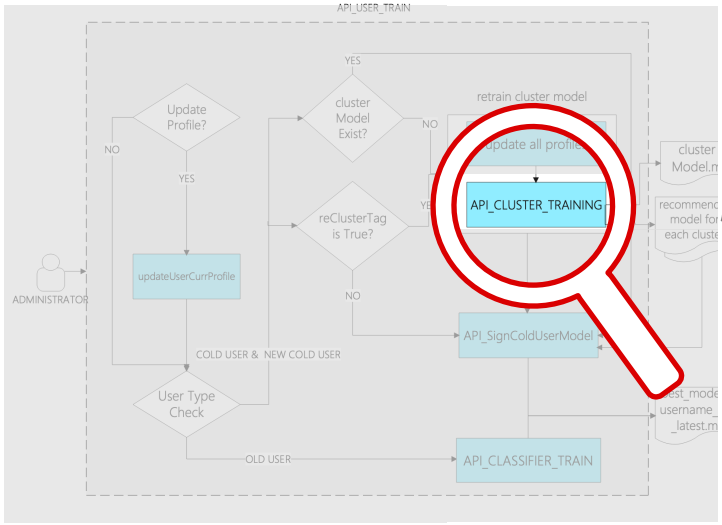


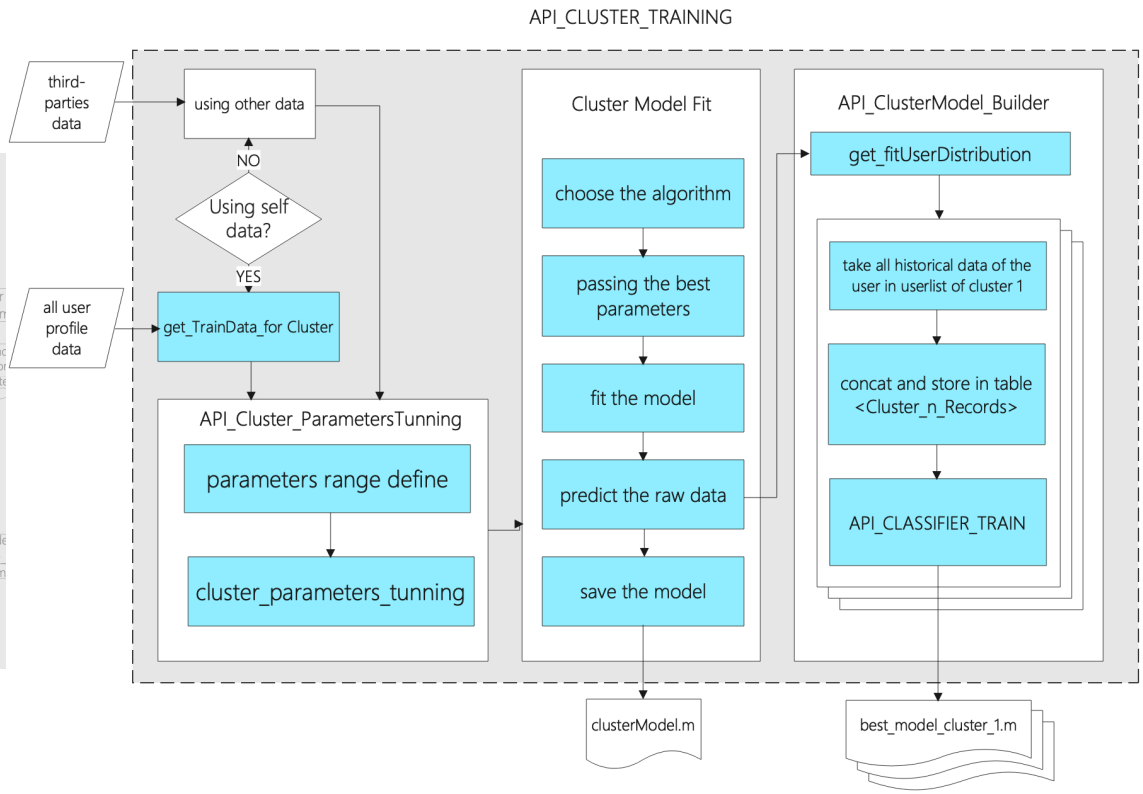
For a general user, all their historical records will be collected, and all the data are labeled with Bought Tag. The offer user has bought in the past will be labeled 1.

After all the preprocessing methods, the dataset will be test on different classifier algorithms, like KNN, SVC, Decision Tree, Logistic Regression, Random Forest. Bayes Search Cross Validation method will be used to find the best model automatically for the given dataset and save the related model in the local file.

Design Idea

Learner(3/3)



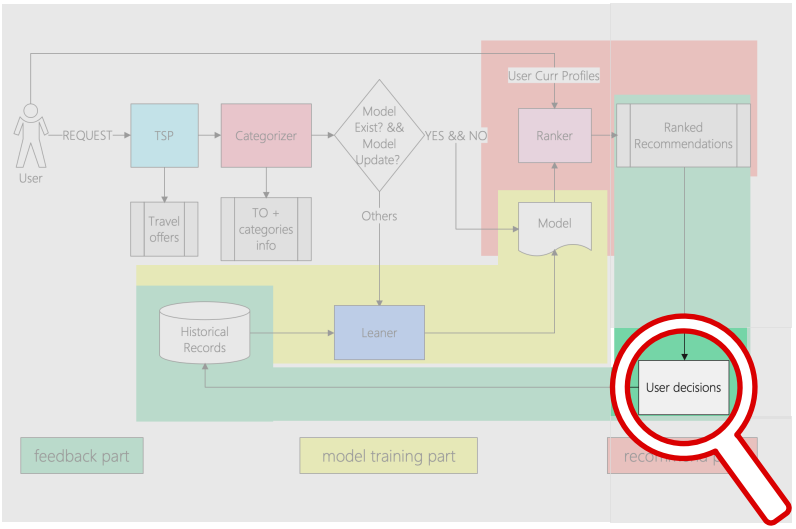


For the new user, we aim to find their similar group and use the recommender model of the group to help the new user to make recommendations.

All the general users' profile will be collected for training the cluster model; two famous algorithms (KMeans , DBSCAN) are used and silhouette coefficient score can be the judgement for picking the better cluster model.
For making the predefined model for a cluster, we collect all the historical records of all the valid users in the cluster and use classifier algorithms to train the corresponding recommender model.

Design Idea

Feedback



In my design, only top 30 recommendations will be pushed into users, this number can be modified at any time.

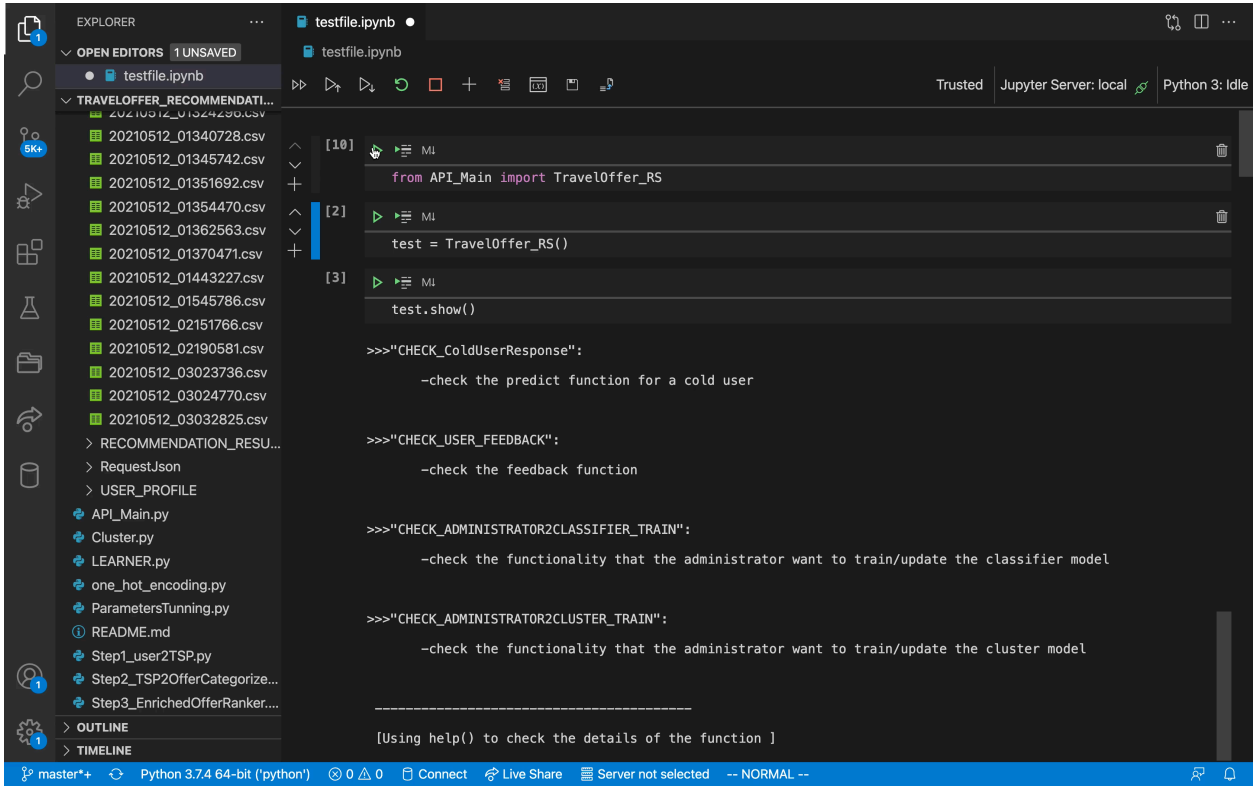
All the top 30 recommendations with the Bought Tag will be written into historical records.

If the user do not buy any of recommendations and choose another one in the system, the bought travel offer will also be recorded.

Program Running

Program Running

Make Recommendations for Different Users



More information can be checked in pdf - guide for using.pdf

Three types of user will give different response in this process.

- A general user (old user) who had recommender model in system

```
test.API_USER_PREDICT(test.OLD_USER,test.req)
```

- A cold user who had predefined model in system

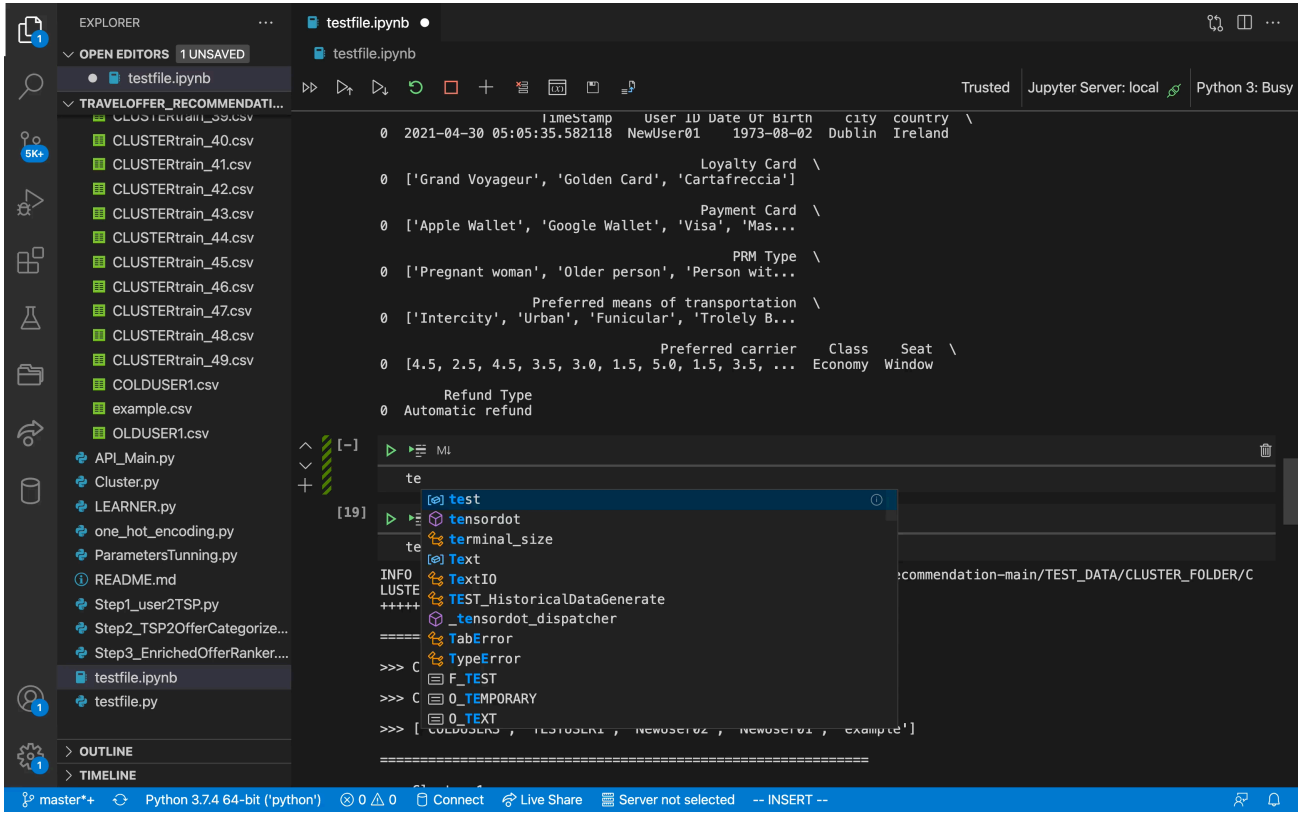
```
test.API_USER_PREDICT(test.COLD_USER,test.req)
```

- A new user who do not have the model in system

```
newuser = 'NewUser01'
new_profile = test.df_profile
new_profile.loc[0,'User ID'] = newuser
test.API_USER_PREDICT(newuser,test.req,new_profile)
```

Program Running

Model Training for Different Users



More information can be checked in pdf - guide for using.pdf

Administrator can choose to update the model by API_USER_TRAIN

-Update the recommender model for an old user:

```
test.API_USER_TRAIN(test.OLD_USER)
```

-Update the cluster model and predefined model for cold user:

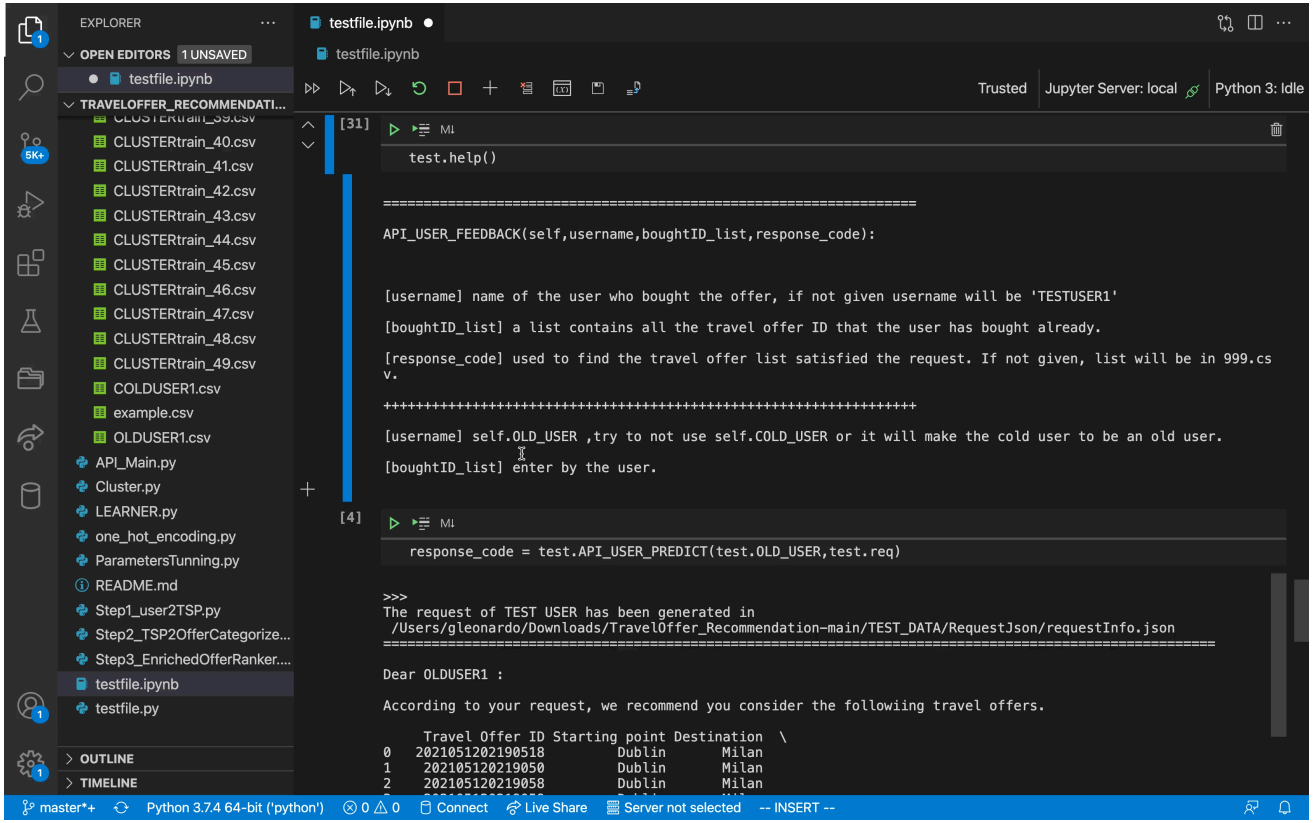
```
test.API_USER_TRAIN(test.COLD_USER)
```

-Sign the predefined model without retraining the cluster model

```
test.API_USER_TRAIN(test.COLD_USER,re
ClusterTag=False)
```

Program Running

User Feedback Test



More information can be checked in pdf - guide for using.pdf

Choose the response_code according to the travel offer list given by TSP, it means the unique code of a response. You can use predict function to get the code.

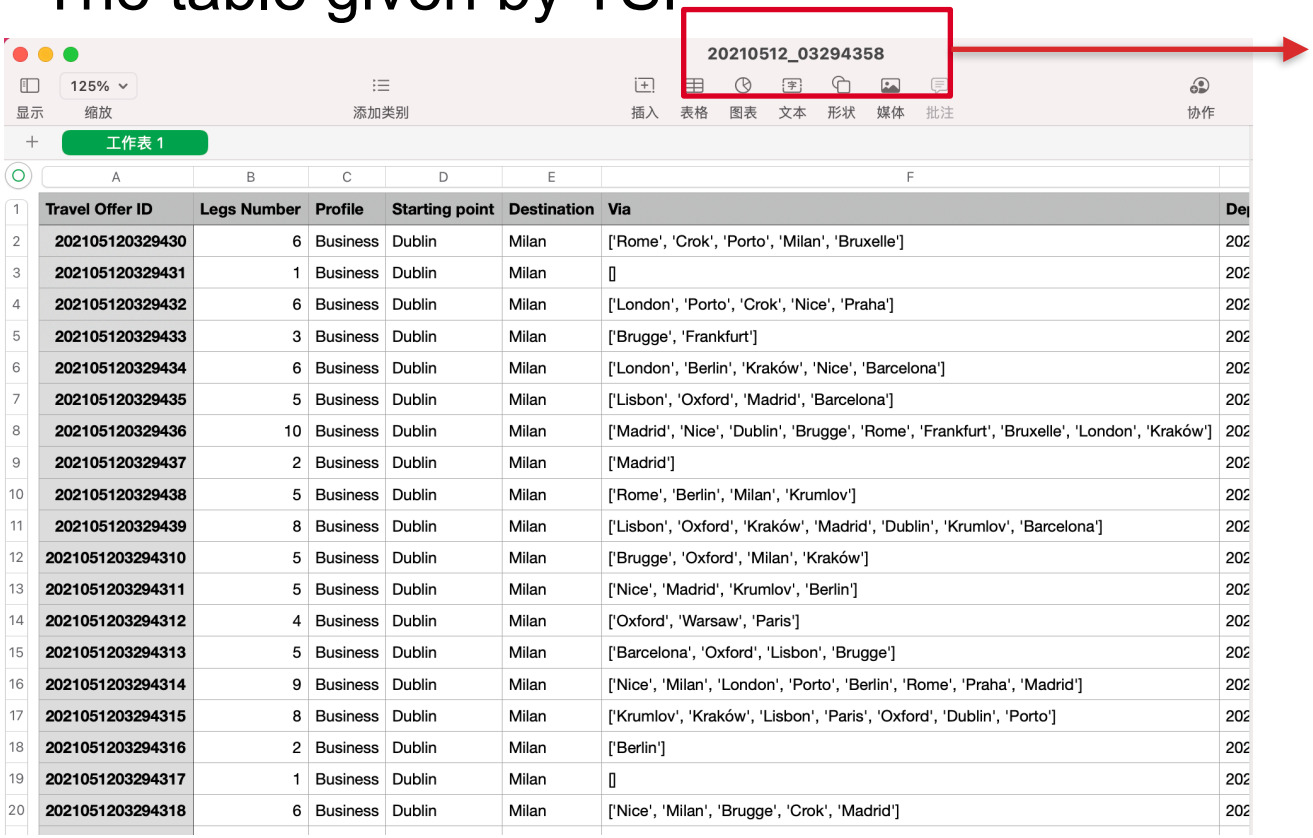
Choose parts of travel offer ID from the recommendation as the boughtList.

```
response_code=test.API_USER_PREDICT(test.OLD_USER,test.req)

boughtList = [2021051202190515]
# change by yourself

test.API_USER_FEEDBACK(test.OLD_USER,boughtList,response_code)
```

Results Display

Results Display

The table given by TSP



Travel Offer ID	Legs Number	Profile	Starting point	Destination	Via	Dep
202105120329430	6	Business	Dublin	Milan	['Rome', 'Crok', 'Porto', 'Milan', 'Bruxelle']	202
202105120329431	1	Business	Dublin	Milan	[]	202
202105120329432	6	Business	Dublin	Milan	['London', 'Porto', 'Crok', 'Nice', 'Praha']	202
202105120329433	3	Business	Dublin	Milan	['Brugge', 'Frankfurt']	202
202105120329434	6	Business	Dublin	Milan	['London', 'Berlin', 'Kraków', 'Nice', 'Barcelona']	202
202105120329435	5	Business	Dublin	Milan	['Lisbon', 'Oxford', 'Madrid', 'Barcelona']	202
202105120329436	10	Business	Dublin	Milan	['Madrid', 'Nice', 'Dublin', 'Brugge', 'Rome', 'Frankfurt', 'Bruxelle', 'London', 'Kraków']	202
202105120329437	2	Business	Dublin	Milan	['Madrid']	202
202105120329438	5	Business	Dublin	Milan	['Rome', 'Berlin', 'Milan', 'Krumlov']	202
202105120329439	8	Business	Dublin	Milan	['Lisbon', 'Oxford', 'Kraków', 'Madrid', 'Dublin', 'Krumlov', 'Barcelona']	202
2021051203294310	5	Business	Dublin	Milan	['Brugge', 'Oxford', 'Milan', 'Kraków']	202
2021051203294311	5	Business	Dublin	Milan	['Nice', 'Madrid', 'Krumlov', 'Berlin']	202
2021051203294312	4	Business	Dublin	Milan	['Oxford', 'Warsaw', 'Paris']	202
2021051203294313	5	Business	Dublin	Milan	['Barcelona', 'Oxford', 'Lisbon', 'Brugge']	202
2021051203294314	9	Business	Dublin	Milan	['Nice', 'Milan', 'London', 'Porto', 'Berlin', 'Rome', 'Praha', 'Madrid']	202
2021051203294315	8	Business	Dublin	Milan	['Krumlov', 'Kraków', 'Lisbon', 'Paris', 'Oxford', 'Dublin', 'Porto']	202
2021051203294316	2	Business	Dublin	Milan	['Berlin']	202
2021051203294317	1	Business	Dublin	Milan	[]	202
2021051203294318	6	Business	Dublin	Milan	['Nice', 'Milan', 'Brugge', 'Crok', 'Madrid']	202

The name of the table represent the response code, and it should be unique.

All the columns has been introduced in chapter Design Idea, and each row represents one travel offer that satisfy the user's request.

Results Display

The table given by Categorizer

20210512_03294358

75% 显示 缩放 添加类别 插入 表格 图表 文本 形状 媒体 批注 协作 格式 整理

工作表 1

Quick	Reliable	Cheap	Comfortable	Door-to-door	Environmentally friendly	Short	Multitasking	Social	Panoramic	Healthy	LegMode	LegCarrier
0.7507	0.9211	0.6857	0.0054	0.4135	0.7846	0.3657	0.6577	0.6614	0.7398	0.89	['Taxi', 'Airline', 'Trolely Bus', 'Intercity', 'Coach', 'Toll']	['Renfe', 'VBB', 'Trenitalia', 'AirFrance', 'SNFC', 'AirFrance']
0.0461	0.116	0.8716	0.0459	0.4357	0.775	0.3412	0.8333	0.6691	0.4375	0.8293	['Toll']	['KLM']
0.3741	0.2453	0.6421	0.9779	0.544	0.0993	0.3244	0.1679	0.6616	0.2113	0.8905	['Coach', 'Bike Sharing', 'Taxi', 'Car Sharing', 'Trolely Bus', 'Park']	['Iberia', 'VBB', 'Renfe', 'Trenitalia', 'KLM', 'SNFC']
0.824	0.3839	0.6977	0.9038	0.8319	0.6129	0.8057	0.6978	0.2894	0.9403	0.5382	['Other', 'Train', 'Tram']	['AirFrance', 'SNFC', 'SNFC']
0.1926	0.5959	0.1578	0.3567	0.4432	0.177	0.8054	0.6718	0.5649	0.4767	0.0569	['Bus', 'Tram', 'Trolely Bus', 'Car Sharing', 'Intercity', 'Toll']	['Iberia', 'KLM', 'TMB', 'Renfe', 'VBB', 'SNFC']
0.1378	0.5564	0.8562	0.7715	0.787	0.3834	0.8188	0.9367	0.5711	0.1186	0.9071	['Funicular', 'Bus', 'Metro', 'Urban', 'Airline']	['AirFrance', 'Renfe', 'Renfe', 'SNFC', 'RegioJet']
0.0418	0.12	0.2404	0.6967	0.1474	0.8458	0.3371	0.9448	0.4619	0.9185	0.846	['Funicular', 'Urban', 'Metro', 'Cable Way', 'Park', 'Other', 'Other', 'Airline', 'Urban', 'Other']	['Iberia', 'SNFC', 'SNFC', 'Trenitalia', 'FlixBus', 'AirFrance', 'Trenitalia', 'Sl
0.9195	0.7152	0.28	0.964	0.7008	0.3524	0.047	0.9294	0.0695	0.3989	0.8816	['Car Sharing', 'Trolely Bus']	['FlixBus', 'FlixBus']
0.7508	0.0563	0.7239	0.6739	0.6141	0.8244	0.6838	0.584	0.3237	0.577	0.778	['Coach', 'Cable Way', 'Funicular', 'Tram', 'Park']	['Trenitalia', 'Iberia', 'TMB', 'AirFrance', 'Renfe']
0.1829	0.5508	0.0141	0.1962	0.2347	0.2026	0.1825	0.5884	0.6118	0.9192	0.7021	['Funicular', 'Urban', 'Intercity', 'Tram', 'Coach', 'Trolely Bus', 'Trolely Bus', 'Urban']	['SNFC', 'SNFC', 'SNFC', 'Iberia', 'Iberia', 'SNFC', 'FlixBus', 'SNFC']
0.911	0.4553	0.3224	0.9468	0.241	0.2342	0.6333	0.1352	0.4403	0.2339	0.6555	['Toll', 'Urban', 'Bike Sharing', 'Intercity', 'Funicular']	['Trenitalia', 'SNFC', 'Trenitalia', 'Iberia', 'Iberia']
0.1985	0.6012	0.2629	0.5924	0.1178	0.4392	0.5525	0.0106	0.9627	0.0146	0.1069	['Funicular', 'Airline', 'Train', 'Car Sharing', 'Cable Way']	['SNFC', 'RegioJet', 'KLM', 'VBB', 'KLM']
0.2614	0.8123	0.4043	0.2281	0.8667	0.2452	0.5231	0.2268	0.5761	0.0141	0.7205	['Trolely Bus', 'Coach', 'Ship', 'Park']	['FlixBus', 'VBB', 'FlixBus', 'Renfe']
0.7976	0.4719	0.0981	0.3096	0.7796	0.0123	0.034	0.3837	0.6173	0.2663	0.5814	['Bike Sharing', 'Car Sharing', 'Bus', 'Funicular', 'Park']	['RegioJet', 'TMB', 'KLM', 'Trenitalia', 'RegioJet']
0.4241	0.0725	0.4373	0.227	0.309	0.2299	0.2407	0.9414	0.3217	0.5933	0.3491	['Metro', 'Bike Sharing', 'Intercity', 'Trolely Bus', 'Bike Sharing', 'Other', 'Park', 'Metro', 'Cable Way']	['Renfe', 'RegioJet', 'SNFC', 'KLM', 'Renfe', 'Renfe', 'KLM', 'RegioJet', 'F
0.0088	0.6978	0.9617	0.2364	0.2349	0.0013	0.9963	0.3221	0.3357	0.2631	0.6364	['Metro', 'Other', 'Other', 'Other', 'Cable Way', 'Cable Way', 'Park', 'Train']	['Iberia', 'Iberia', 'Renfe', 'KLM', 'Iberia', 'TMB', 'Renfe', 'VBB']
0.7188	0.1881	0.0073	0.8671	0.5256	0.7037	0.7261	0.0565	0.978	0.2035	0.2291	['Urban', 'Tram']	['SNFC', 'AirFrance']
0.002	0.6704	0.6827	0.97	0.1985	0.1925	0.6858	0.2689	0.3517	0.661	0.9588	['Car Sharing']	['Renfe']
0.8043	0.4302	0.2358	0.5119	0.0593	0.0385	0.2642	0.6152	0.6484	0.2615	0.7897	['Car Sharing', 'Coach', 'Taxi', 'Coach', 'Bike Sharing', 'Funicular']	['VBB', 'Trenitalia', 'Trenitalia', 'RegioJet', 'SNFC', 'KLM']
0.4569	0.8407	0.4548	0.2955	0.4058	0.6413	0.1749	0.5381	0.4781	0.7237	0.7417	['Coach', 'Bike Sharing', 'Taxi', 'Toll', 'Metro', 'Airline', 'Tram']	['Renfe', 'KLM', 'SNFC', 'VBB', 'TMB', 'VBB', 'Iberia']
0.6926	0.1968	0.3839	0.9703	0.6509	0.9767	0.6522	0.9551	0.2544	0.357	0.6578	['Train', 'Taxi', 'Park', 'Park', 'Toll', 'Toll', 'Bike Sharing']	['Iberia', 'VBB', 'Iberia', 'VBB', 'SNFC', 'AirFrance', 'FlixBus']
0.4547	0.9944	0.6733	0.6381	0.5121	0.0137	0.1197	0.656	0.2	0.3846	0.8812	['Ship', 'Cable Way']	['FlixBus', 'AirFrance']
0.7624	0.1758	0.0092	0.8784	0.5852	0.5599	0.4451	0.4335	0.764	0.9976	0.0127	['Taxi', 'Funicular']	['SNFC', 'AirFrance']
0.5305	0.3724	0.9969	0.075	0.2025	0.8543	0.3438	0.8316	0.2263	0.6763	0.3083	['Bus', 'Intercity', 'Coach', 'Coach', 'Airline', 'Taxi']	['SNFC', 'FlixBus', 'AirFrance', 'FlixBus', 'TMB', 'Iberia']
0.1508	0.7305	0.1163	0.9294	0.0626	0.1332	0.1947	0.2733	0.2328	0.402	0.8688	['Funicular', 'Other', 'Toll', 'Tram', 'Intercity', 'Funicular', 'Train']	['Trenitalia', 'VBB', 'FlixBus', 'RegioJet', 'KLM', 'SNFC', 'TMB']
0.1406	0.7802	0.8252	0.098	0.9425	0.8856	0.6101	0.0234	0.8719	0.9102	0.094	['Funicular', 'Train']	['Trenitalia', 'Trenitalia']

The Categories' points information and leg related information are added into every travel offer

Results Display

The table after one-hot encoding

checkOneHot_20210430_170759

Legs Number	Bought Tag	DoB-year	DoB-month	DoB-day	Loyalty Card-Cartafreccia	Loyalty Card-FlyingBlue	Loyalty Card-Golden Card	Loyalty Card-Grand Voyageur	Payment Card-Mastercard	Payment Card-
8	0	1995	6	2	0	1	1	1	1	
3	1	1995	6	2	0	1	1	1	1	
6	0	1995	6	2	0	1	1	1	1	
1	1	1995	6	2	0	1	1	1	1	
7	0	1995	6	2	0	1	1	1	1	
8	0	1995	6	2	0	1	1	1	1	
5	0	1995	6	2	0	1	1	1	1	
6	1	1995	6	2	0	1	1	1	1	
9	1	1995	6	2	0	1	1	1	1	
1	0	1995	6	2	0	1	1	1	1	
7	0	1995	6	2	0	1	1	1	1	
10	0	1995	6	2	0	1	1	1	1	
10	0	1995	6	2	0	1	1	1	1	
2	1	1995	6	2	0	1	1	1	1	
3	0	1995	6	2	0	1	1	1	1	
4	0	1995	6	2	0	1	1	1	1	
6	1	1995	6	2	0	1	1	1	1	
5	1	1995	6	2	0	1	1	1	1	
3	0	1995	6	2	0	1	1	1	1	
4	1	1995	6	2	0	1	1	1	1	
4	1	1995	6	2	0	1	1	1	1	
3	0	1995	6	2	0	1	1	1	1	
6	0	1995	6	2	0	1	1	1	1	
2	0	1995	6	2	0	1	1	1	1	
3	1	1995	6	2	0	1	1	1	1	
2	1	1995	6	2	0	1	1	1	1	
7	1	1995	6	2	0	1	1	1	1	
10	1	1995	6	2	0	1	1	1	1	
7	1	1995	6	2	0	1	1	1	1	
7	1	1995	6	2	0	1	1	1	1	
7	0	1995	6	2	0	1	1	1	1	
7	0	1995	6	2	0	1	1	1	1	
7	0	1995	6	2	0	1	1	1	1	

All the textual columns are transferred into numeric.

Results Display

The table after Information-free Columns Deletion

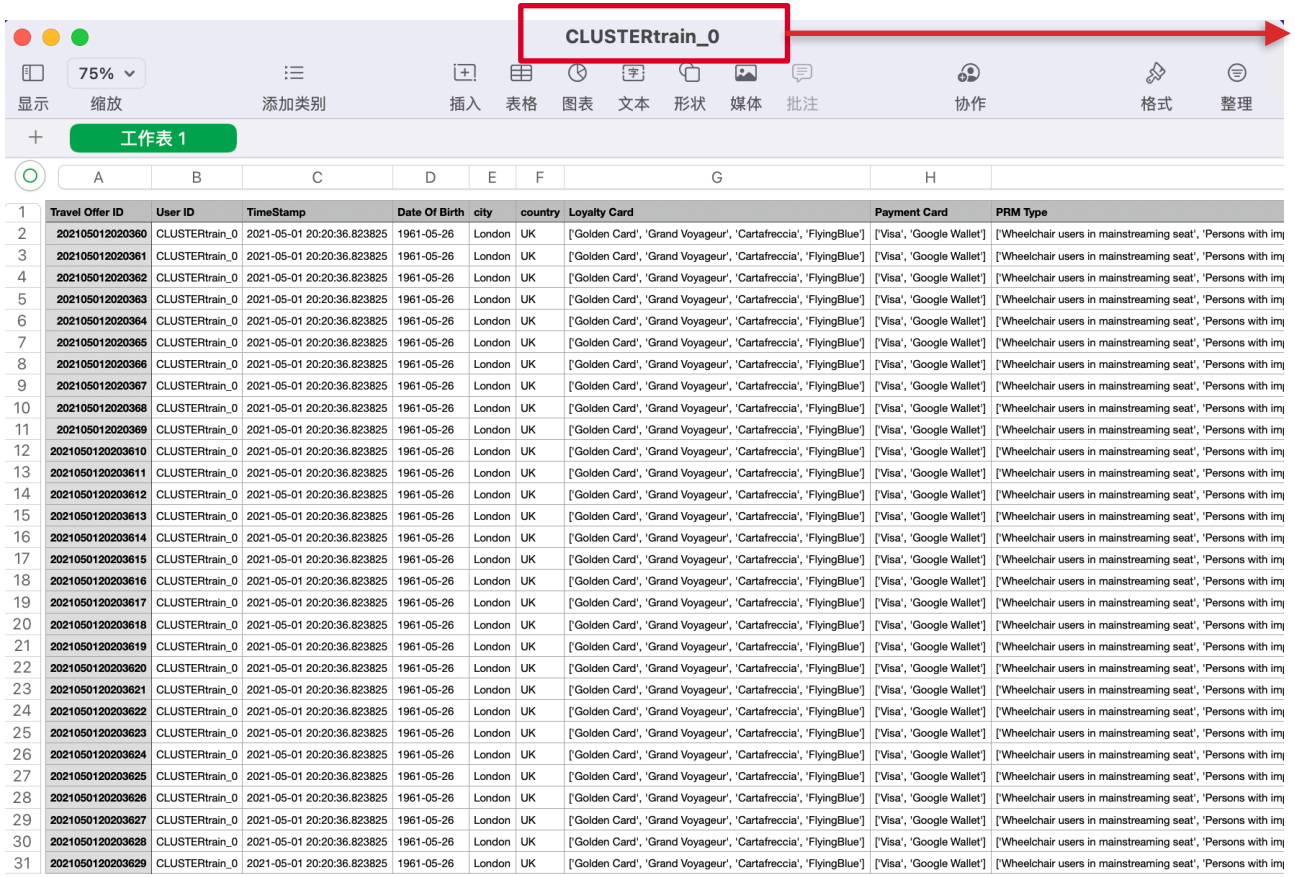
Travel Offer ID	Quick	Reliable	Cheap	Comfortable	Door-to-door	Environmentally friendly	Short	Multitasking	Social	Panoramic	Healthy
1	0.5818	0.3621	0.2227	0.6497	0.3435	0.2692	0.1215	0.646	0.3185	0.4839	0.5479
2	0.4717	0.1103	0.9273	0.395	0.2099	0.0827	0.856	0.4057	0.315	0.4894	0.3038
3	0.0531	0.4085	0.5388	0.5598	0.6167	0.2522	0.7713	0.5165	0.4961	0.6954	0.0028
4	0.4357	0.7781	0.2402	0.5757	0.8395	0.8249	0.3572	0.2833	0.1184	0.4102	0.2494
5	0.575	0.1458	0.7976	0.6189	0.9728	0.8224	0.2394	0.0244	0.0285	0.0716	0.6656
6	0.0735	0.3075	0.6138	0.0522	0.5706	0.1507	0.2939	0.346	0.5301	0.1459	0.2278
7	0.5577	0.3268	0.7451	0.5322	0.9762	0.7085	0.2633	0.2586	0.1171	0.7382	0.8847
8	0.212	0.4496	0.8377	0.1564	0.0007	0.4657	0.2574	0.5902	0.3872	0.3755	0.3084
9	0.7597	0.7423	0.4961	0.6802	0.525	0.6101	0.9457	0.6196	0.7599	0.2771	0.0482
10	0.2561	0.1142	0.4288	0.3775	0.0149	0.4141	0.9955	0.0713	0.4181	0.2513	0.9378
11	0.7792	0.677	0.3146	0.6777	0.3608	0.8648	0.7382	0.8842	0.8808	0.7117	0.4639
12	0.0077	0.614	0.9784	0.6617	0.5935	0.3237	0.0694	0.4058	0.8447	0.7417	0.6899
13	0.5005	0.7786	0.5709	0.3939	0.7127	0.3243	0.1649	0.921	0.4396	0.6006	0.2289
14	0.2936	0.2048	0.9986	0.0649	0.7212	0.9996	0.3953	0.0199	0.7147	0.1102	0.1332
15	0.6073	0.0425	0.3058	0.783	0.8089	0.7176	0.7846	0.458	0.184	0.1258	0.3084
16	0.441	0.8668	0.0216	0.9475	0.0359	0.4587	0.9396	0.0147	0.7496	0.4605	0.3496
17	0.5231	0.2284	0.6781	0.2692	0.128	0.6817	0.2728	0.3478	0.4628	0.2764	0.8788
18	0.2122	0.014	0.718	0.6867	0.5367	0.8408	0.8452	0.6896	0.6482	0.3446	0.9149
19	0.1125	0.8733	0.5467	0.285	0.5135	0.7424	0.4047	0.857	0.1623	0.544	0.4682
20	0.3304	0.4098	0.373	0.8595	0.9816	0.8314	0.3102	0.4443	0.1071	0.3115	0.8332
21	0.8707	0.8908	0.8011	0.0186	0.561	0.2532	0.9842	0.6672	0.451	0.0026	0.3594
22	0.943	0.6538	0.4181	0.1954	0.6536	0.6186	0.3288	0.9371	0.6775	0.0869	0.2925
23	0.037	0.5752	0.5985	0.3014	0.2131	0.3117	0.0569	0.3238	0.8407	0.4084	0.5747
24	0.8416	0.8547	0.1693	0.6929	0.9681	0.0852	0.5322	0.6072	0.5479	0.0074	0.5514
25	0.2222	0.4568	0.6021	0.6	0.229	0.6531	0.8002	0.599	0.4329	0.8768	0.7862

All the columns that have the same value means nothing for our system, and we call this type of column as Information-free columns.

This process aims to delete all the Information-free columns for reducing the training time cost.

Results Display

The table of user ‘CLUSTERtrain_0”s historical records



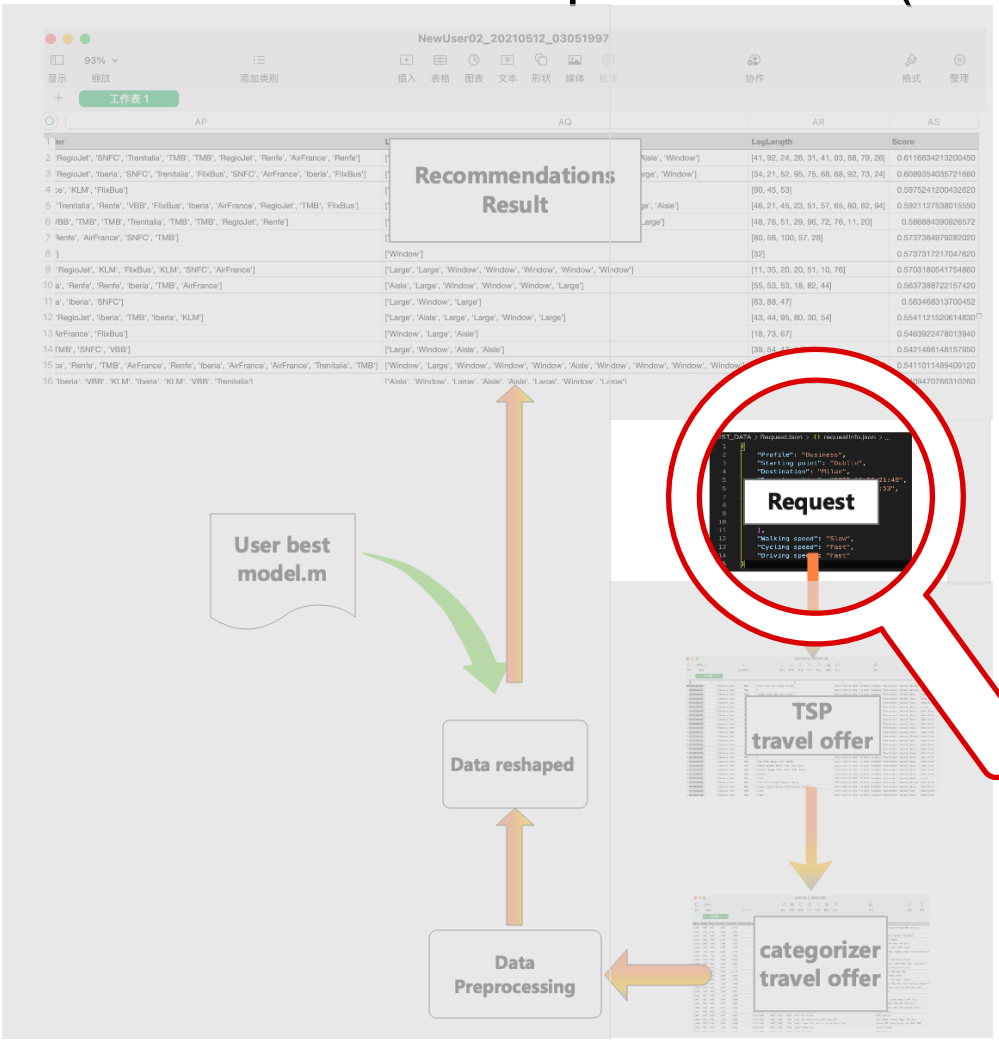
	Travel Offer ID	User ID	TimeStamp	Date Of Birth	city	country	Loyalty Card	Payment Card	PRM Type
2	202105012020360	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
3	202105012020361	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
4	202105012020362	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
5	202105012020363	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
6	202105012020364	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
7	202105012020365	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
8	202105012020366	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
9	202105012020367	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
10	202105012020368	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
11	202105012020369	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
12	202105012020610	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
13	202105012020611	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
14	202105012020612	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
15	202105012020613	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
16	202105012020614	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
17	202105012020615	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
18	202105012020616	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
19	202105012020617	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
20	202105012020618	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
21	202105012020619	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
22	202105012020620	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
23	202105012020621	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
24	202105012020622	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
25	202105012020623	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
26	202105012020624	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
27	202105012020625	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
28	202105012020626	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
29	202105012020627	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
30	202105012020628	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im
31	202105012020629	CLUSTERtrain_0	2021-05-01 20:20:36.823825	1961-05-26	London	UK	['Golden Card', 'Grand Voyageur', 'Cartafreccia', 'FlyingBlue']	['Visa', 'Google Wallet']	['Wheelchair users in mainstreaming seat', 'Persons with im

The name of the table represent the user ID and it should be unique

This table contains all the historical records of the specific user, and the Bought Tag information is recorded for every record.
The user’s profile at the corresponding period is also recorded.

Results Display

General User Response Flow(1/5)



Consider a general user named “OldUser”

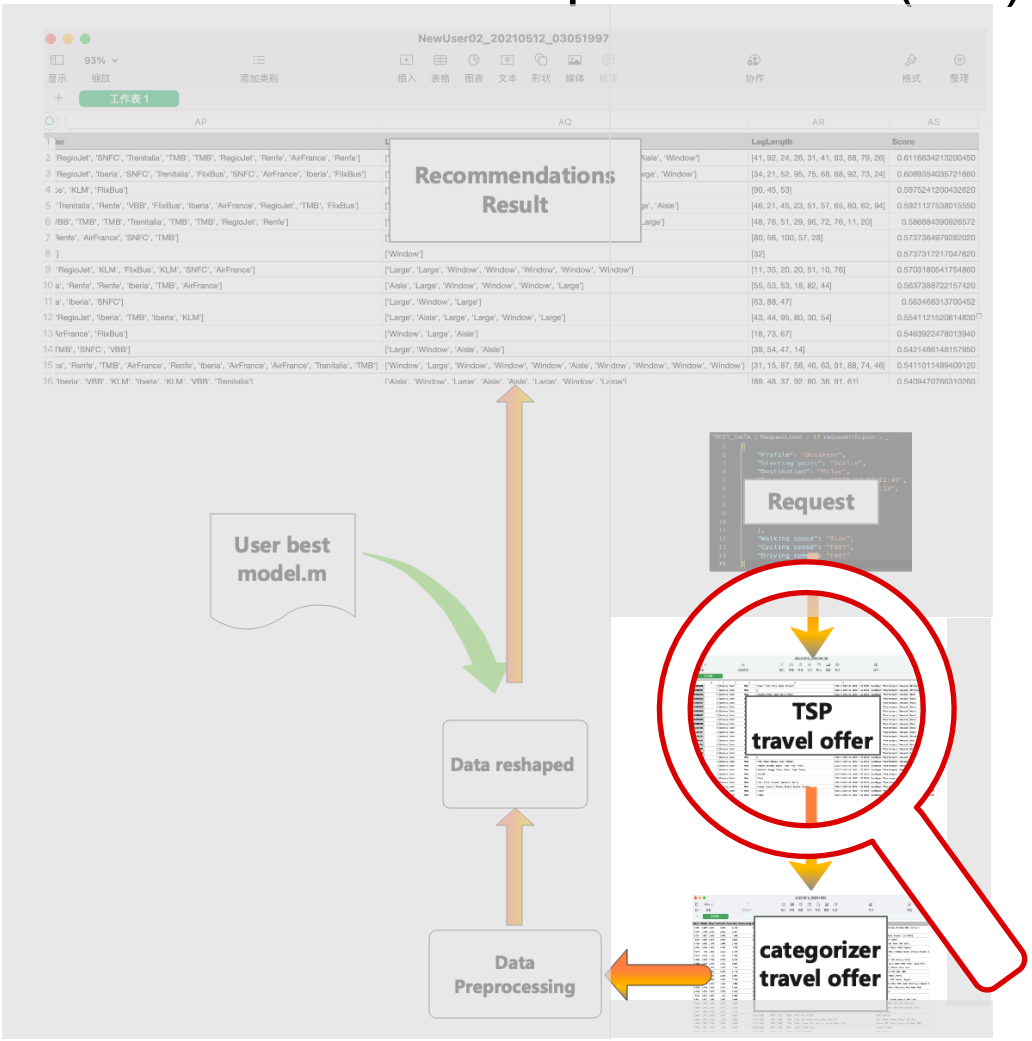
He gives his request like following:

TEST_DATA > RequestJson > {} requestInfo.json > ...

```
{
    "Profile": "Business",
    "Starting point": "Dublin",
    "Destination": "Milan",
    "Departure time": "2022-10-20 21:49",
    "Arrival time": "2022-11-02 08:53",
    "Services": [
        "Local type",
        "Water transport",
        "Telecabin"
    ],
    "Walking speed": "Slow",
    "Cycling speed": "Fast",
    "Driving speed": "Fast"
}
```

Results Display

General User Response Flow(2/5)

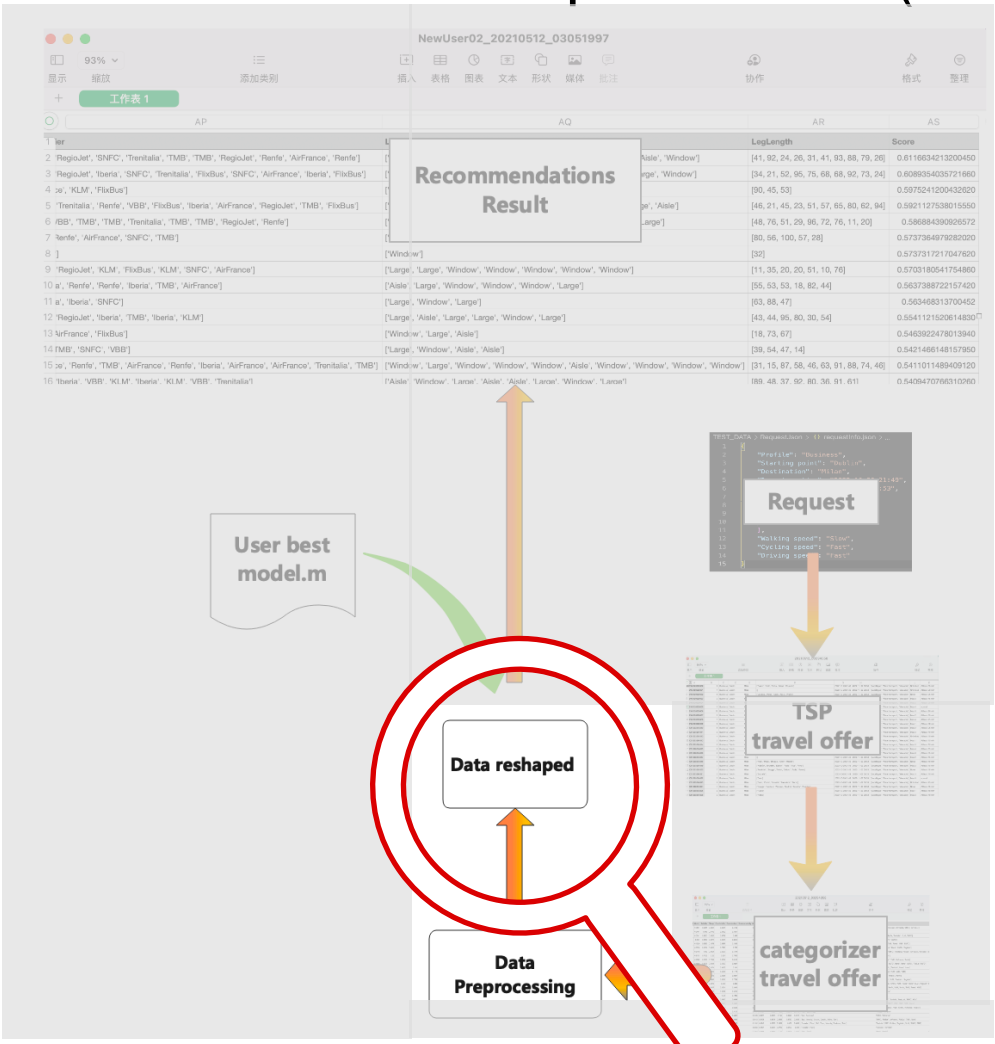


The Request is accepted by the TSP, and Travel Offer list is generated in a table named with the response code.

TO table then will be modified by the Categorizer.

Results Display

General User Response Flow(3/5)



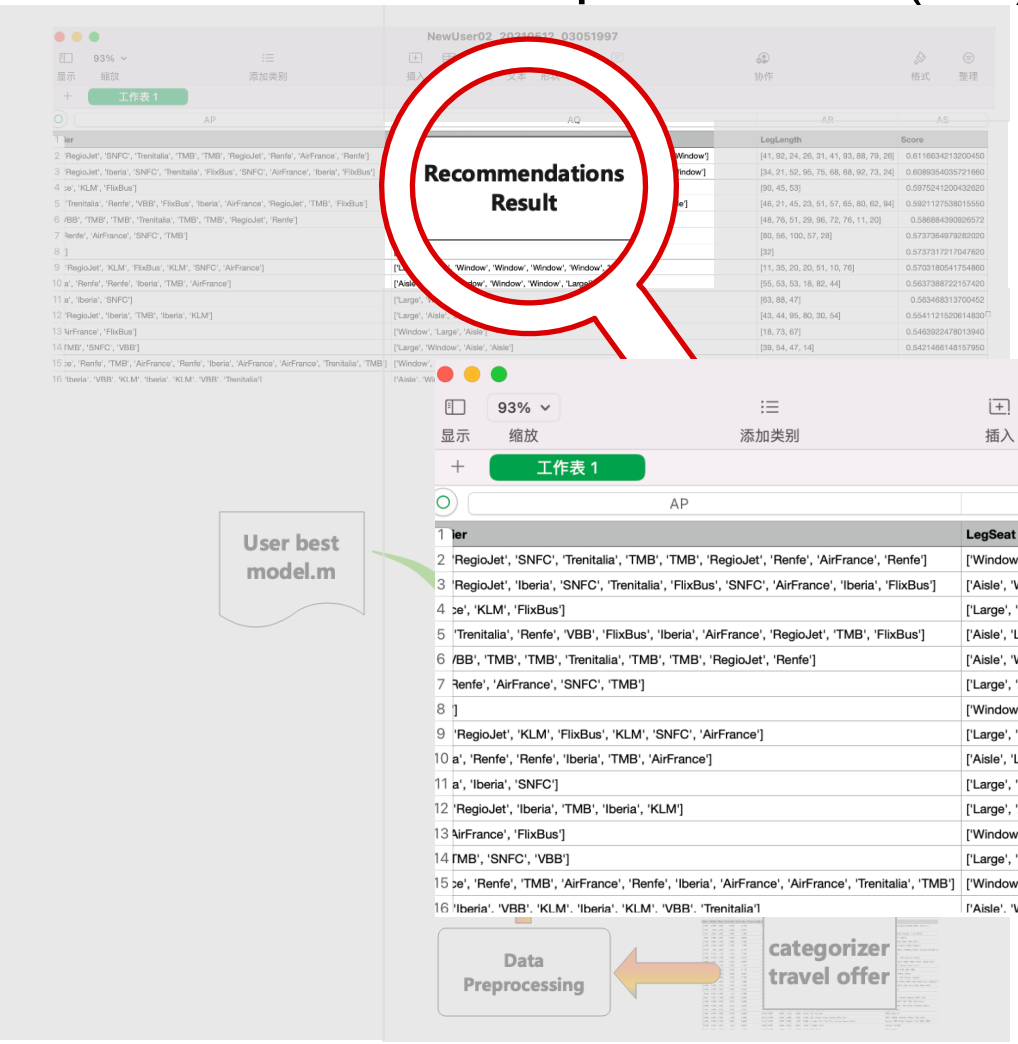
One hot encoding, IFCD, normalization and data reshape methods will transform the dataset to a numeric matrix so that can be learnt by the algorithms.

·*Why we need Data reshape?*

To keep the response data have the same size with the train data used in the model, so that the response data can be accepted by the model.

Results Display

General User Response Flow(4/5)



NewUser02_20210512_03051997

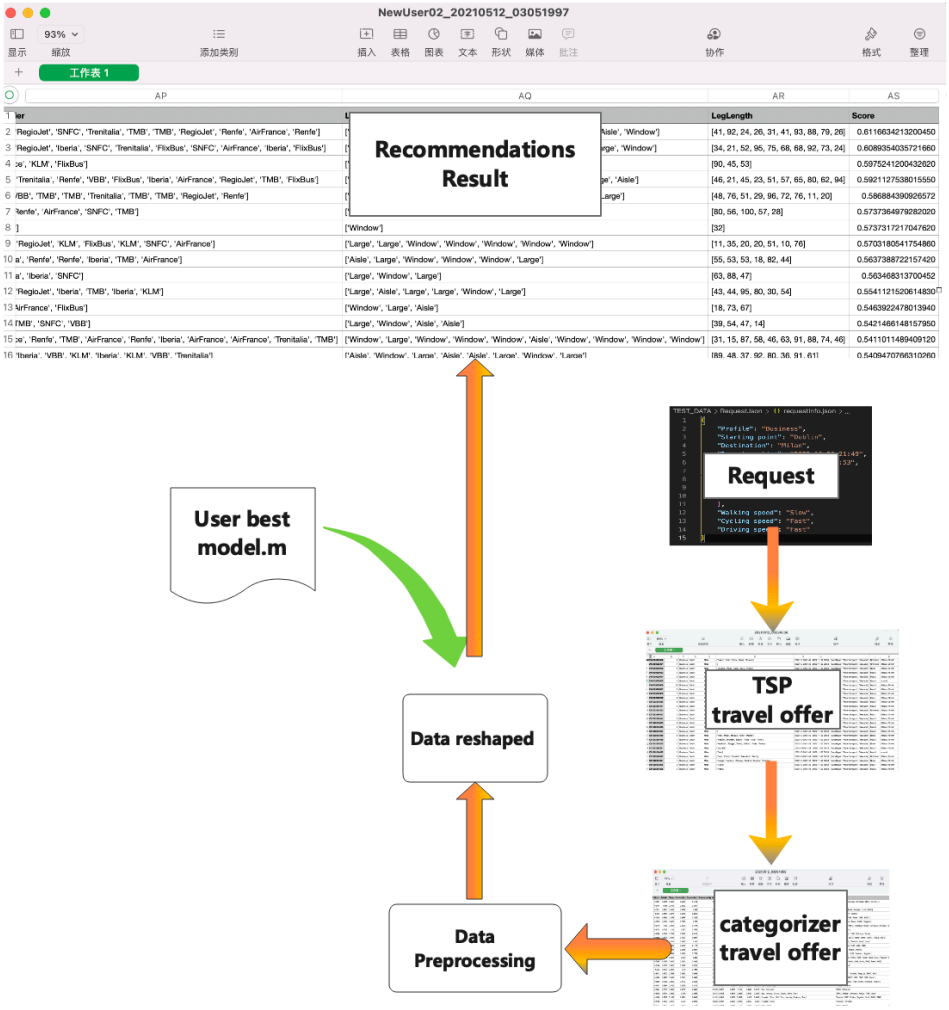
	AP	AQ	AR	AS
1	ier	LegSeat	LegLength	Score
2	'RegioJet', 'SNFC', 'Trenitalia', 'TMB', 'TMB', 'RegioJet', 'Renfe', 'AirFrance', 'Renfe']	['Window', 'Aisle', 'Window', 'Large', 'Aisle', 'Large', 'Aisle', 'Window', 'Aisle', 'Window']	[41, 92, 24, 26, 31, 41, 93, 88, 79, 2	0.6116634213200450
3	'RegioJet', 'Iberia', 'SNFC', 'Trenitalia', 'FlixBus', 'SNFC', 'AirFrance', 'Iberia', 'FlixBus']	['Aisle', 'Window', 'Large', 'Large', 'Window', 'Large', 'Large', 'Aisle', 'Large', 'Window']	[34, 21, 52, 95, 75, 68, 68, 92, 73, 2	0.6089354035721660
4	ce', 'KLM', 'FlixBus']	['Large', 'Window', 'Large']	[90, 45, 53]	0.5975241200432620
5	'Trenitalia', 'Renfe', 'VBB', 'FlixBus', 'Iberia', 'AirFrance', 'RegioJet', 'TMB', 'FlixBus']	['Aisle', 'Large', 'Large', 'Large', 'Large', 'Large', 'Window', 'Large', 'Large', 'Aisle']	[46, 21, 45, 23, 51, 57, 65, 80, 62, 9	0.5921127538015550
6	VBB', 'TMB', 'TMB', 'Trenitalia', 'TMB', 'TMB', 'RegioJet', 'Renfe']	['Aisle', 'Window', 'Aisle', 'Window', 'Window', 'Aisle', 'Large', 'Large', 'Large']	[48, 76, 51, 29, 96, 72, 76, 11, 20]	0.586884390926572
7	Renfe', 'AirFrance', 'SNFC', 'TMB']	['Large', 'Aisle', 'Aisle', 'Window', 'Large']	[80, 56, 100, 57, 28]	0.5737364979282020
8	]	['Window']	[32]	0.5737317217047620
9	'RegioJet', 'KLM', 'FlixBus', 'KLM', 'SNFC', 'AirFrance']	['Large', 'Large', 'Window', 'Window', 'Window', 'Window', 'Window']	[11, 35, 20, 20, 51, 10, 76]	0.5703180541754860
10	a', 'Renfe', 'Renfe', 'Iberia', 'TMB', 'AirFrance']	['Aisle', 'Large', 'Window', 'Window', 'Window', 'Large']	[55, 53, 53, 18, 82, 44]	0.5637388722157420
11	a', 'Iberia', 'SNFC']	['Large', 'Window', 'Large']	[63, 88, 47]	0.563468313700452
12	'RegioJet', 'Iberia', 'TMB', 'Iberia', 'KLM']	['Large', 'Aisle', 'Large', 'Large', 'Window', 'Large']	[43, 44, 95, 80, 30, 54]	0.5541121520614830
13	AirFrance', 'FlixBus']	['Window', 'Large', 'Aisle']	[18, 73, 67]	0.5463922478013940
14	TMB', 'SNFC', 'VBB']	['Large', 'Window', 'Aisle', 'Aisle']	[39, 54, 47, 14]	0.5421466148157950
15	ce', 'Renfe', 'TMB', 'AirFrance', 'Renfe', 'Iberia', 'AirFrance', 'AirFrance', 'Trenitalia', 'TMB']	['Window', 'Large', 'Window', 'Window', 'Window', 'Aisle', 'Window', 'Window', 'Window', 'Window']	[31, 15, 87, 58, 46, 63, 91, 88, 74, 4	0.5411011489409120
16	'Iberia', 'VBB', 'KLM', 'Iberia', 'KLM', 'VBB', 'Trenitalia']	['Aisle', 'Window', 'Large', 'Aisle', 'Aisle', 'Large', 'Window', 'Large']	[89, 48, 37, 92, 80, 36, 91, 61]	0.5409470766310260

Attribute Score reflects the possibility of buying this travel offer for the specific user.

All the related information will be saved in the table.

Results Display

General User Response Flow(5/5)

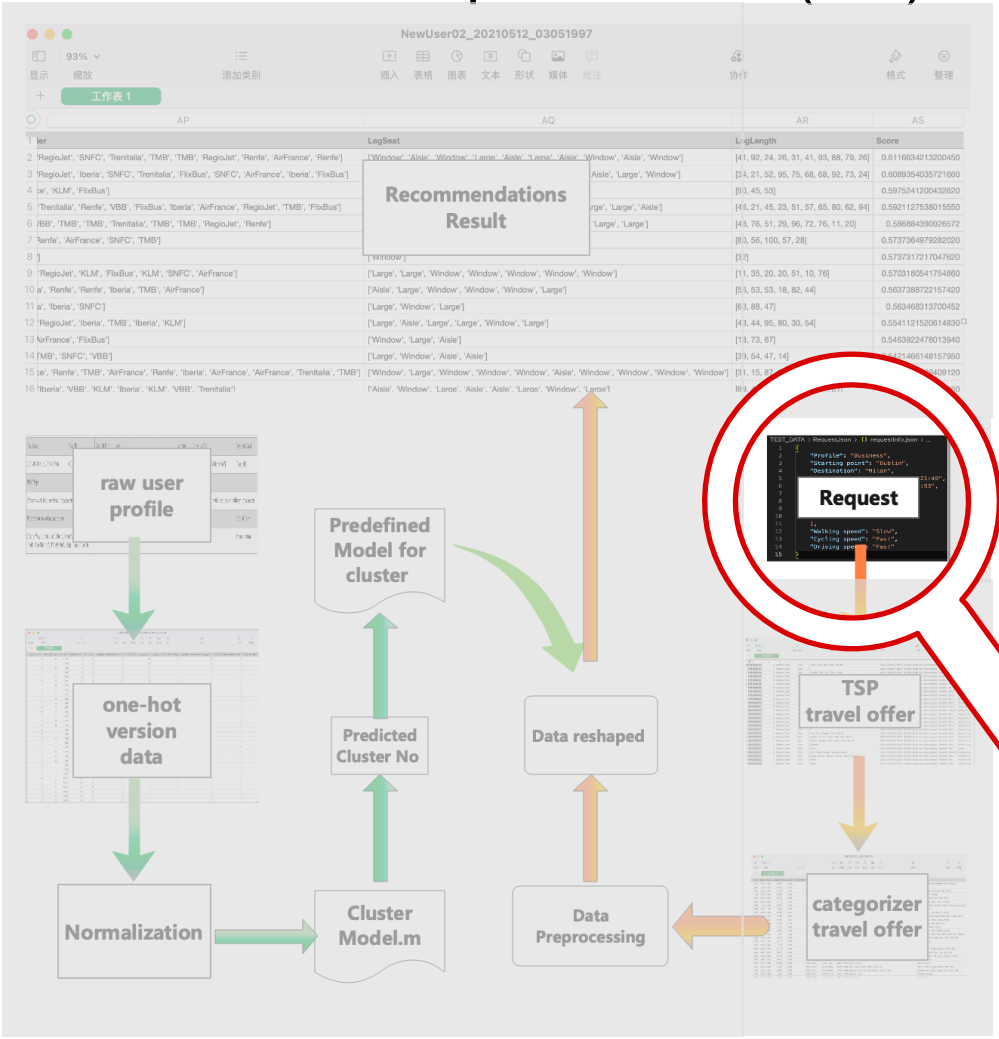


In the screen, we can see the list of recommendations.

```
[14]  M↓
    test.API_USER_PREDICT(test.OLD_USER,test.req)

>>>
The request of TEST USER has been generated in
 /Users/gleonardo/Downloads/TravelOffer_Recommendat
==========================================================

Dear OLDUSER1 :

According to your request, we recommend you conside

     Travel Offer ID Starting point Destination  \
0     202105120304399         Dublin       Milan
1    2021051203043933         Dublin       Milan
2    2021051203043930         Dublin       Milan
3    2021051203043934         Dublin       Milan
4     202105120304394         Dublin       Milan
5    2021051203043912         Dublin       Milan
6    2021051203043919         Dublin       Milan
7     202105120304398         Dublin       Milan
8    2021051203043917         Dublin       Milan
9    2021051203043929         Dublin       Milan
10   2021051203043924         Dublin       Milan
11    202105120304397         Dublin       Milan
12    202105120304395         Dublin       Milan
```

Results Display

New User Response Flow(1/3)



Consider a general user named “NewUser”

The right part flow marked in orange are same with the general user. We assume the user’s request like following:

TEST_DATA > RequestJson > {} requestInfo.json > ...

```
{
    "Profile": "Business",
    "Starting point": "Dublin",
    "Destination": "Milan",
    "Departure time": "2022-10-20 21:49",
    "Arrival time": "2022-11-02 08:53",
    "Services": [
        "Local type",
        "Water transport",
        "Telecabin"
    ],
    "Walking speed": "Slow",
    "Cycling speed": "Fast",
    "Driving speed": "Fast"
}
```

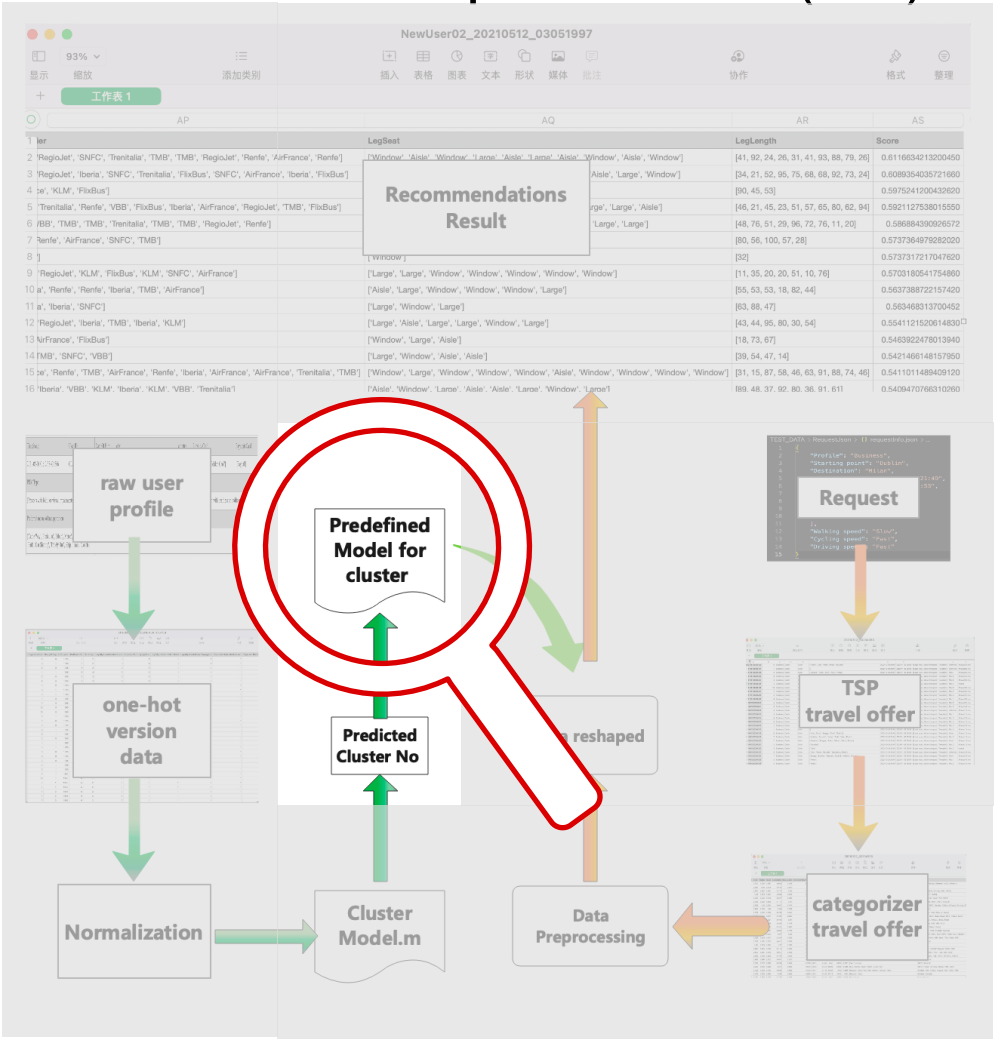
Results Display

New User Response Flow(2/3)

The new user's profile data will be preprocessed and put into cluster model to get the similar cluster.

Results Display

New User Response Flow(3/3)



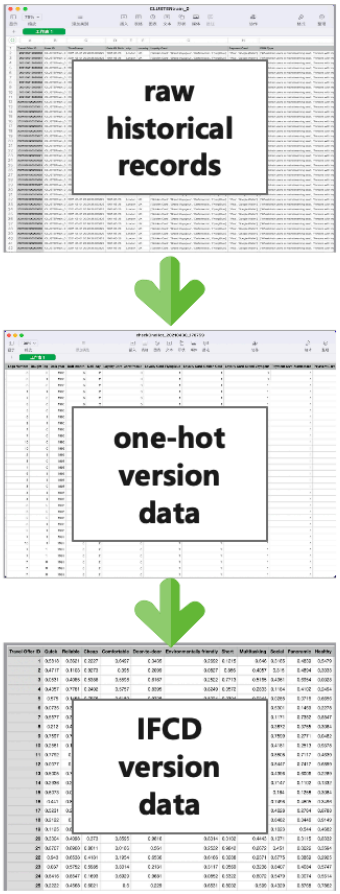
-RED block shows the predicted result

-GREEN block shows the predefined model has been copied to the user

```
[16]
newuser = 'NewUser02'
new_profile = test.df_profile
new_profile.loc[0,'User ID'] = newuser

[18]
test.API_USER_PREDICT('NewUser02',req=test.req,df_profile=new_profile)

Profile Has Been Updated in  /Users/gleonardo/Downloads/TravelOffer_Recommendation-

>>>
The request of TEST USER has been generated in
 /Users/gleonardo/Downloads/TravelOffer_Recommendation-main/TEST_DATA/RequestJson/r
The ClusterNo for the user is : [0]
DONE! THE Pre trained model has been copied into the cold user
==========

Dear NewUser02 :

According to your request, we recommend you consider the followiing travel offers.

     Travel Offer ID Starting point Destination  \
0   2021051203051915         Dublin       Milan
1   2021051203051948         Dublin       Milan
2   2021051203051922         Dublin       Milan
3   2021051203051942         Dublin       Milan
4   2021051203051935         Dublin       Milan
5   2021051203051930         Dublin       Milan
6   2021051203051916         Dublin       Milan
7   2021051203051924         Dublin       Milan
8    202105120305192         Dublin       Milan
9   2021051203051925         Dublin       Milan
10  2021051203051945         Dublin       Milan
11  2021051203051934         Dublin       Milan
12   202105120305193         Dublin       Milan
```

Results Display

General User Training Model Flow(1/4)



System Train Model API

- best_model_OLDUSER1_all_latest.m
- best_model_TESTUSER1_all_latest.m
- best_model_TESTUSER1_train_latest.m

For a general user, all their historical records will be selected and given to the classifier algorithms.

Results Display

General User Training Model Flow(2/4)

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Start Searching The best recommender<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Current Checking Recommender is >>>> [[ KNeighborsClassifier ]]

Time Consuming for this fit is :0.19162321090698242

===> val. score: 0.5397350993377483

===> best parameter: OrderedDict([('algorithm', 'auto'), ('n_neighbors', 26)])

===> Actual Estimator is: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=26, p=2,
                     weights='uniform')



+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Current Checking Recommender is >>>> [[ SVC ]]

Time Consuming for this fit is :3.298020839691162

===> val. score: 0.5562913907284768

===> best parameter: OrderedDict([('C', 0.002773590321759536), ('degree', 2), ('gamma', 0.025374946874608124), ('
kernel', 'linear'), ('probability', True)])

===> Actual Estimator is: SVC(C=0.002773590321759536, break_ties=False, cache_size=200, class_weight=None,
    coef0=0.0, decision_function_shape='ovr', degree=2,
    gamma=0.025374946874608124, kernel='linear', max_iter=-1, probability=True,
    random_state=None, shrinking=True, tol=0.001, verbose=False)
```

System will search the best model in the parameters' range we defined before, and give each best model of different algorithms a score to help the system find the best model from different algorithms.
- Here is the snapshot of KNN, SVC 's best model.

Results Display

General User Training Model Flow(3/4)

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Current Checking Recommender is >>>> [[ DecisionTreeClassifier ]]

Time Consuming for this fit is :0.03579878807067871

===> val. score: 0.5231788079470199

===> best parameter: OrderedDict([('criterion', 'gini'), ('max_depth', 4), ('max_features', 21), ('max_leaf_nodes
', 16), ('min_samples_leaf', 5)])

===> Actual Estimator is: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=4, max_features=21, max_leaf_nodes=16,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=5, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')



++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Current Checking Recommender is >>>> [[ LogisticRegression ]]

Time Consuming for this fit is :0.5745196342468262

===> val. score: 0.5529801324503312

===> best parameter: OrderedDict([('C', 0.9692748387114193), ('fit_intercept', True), ('solver', 'lbfgs')])

===> Actual Estimator is: LogisticRegression(C=0.9692748387114193, class_weight=None, dual=False,
                    fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                    max_iter=100, multi_class='auto', n_jobs=None, penalty='l2',
                    random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                    warm_start=False)
```

- Here shows the snapshot of Decision Tree and Logistic Regression ’s best model

Results Display

General User Training Model Flow(4/4)

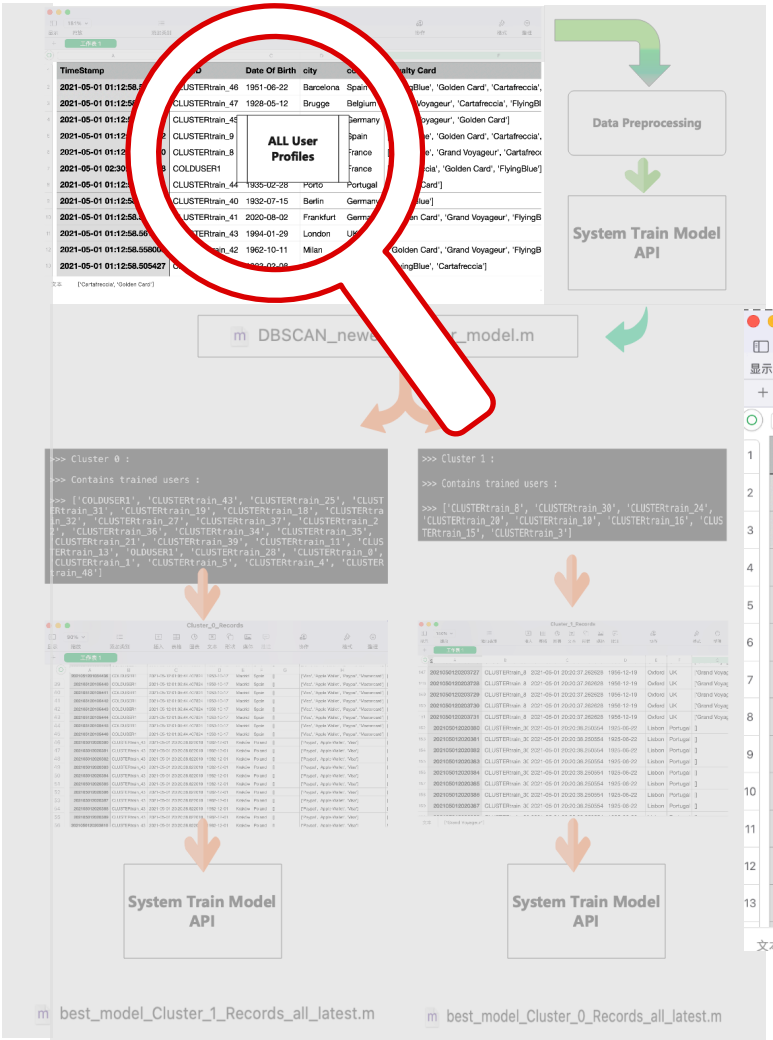
```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Current Checking Recommender is >>>> [[ RandomForestClassifier ]]

Time Consuming for this fit is :1.8376047611236572

===> val. score: 0.5711920529801324

===> best parameter: OrderedDict([('bootstrap', True), ('criterion', 'gini'), ('max_depth', 37), ('max_features',
 38), ('n_estimators', 92)])

===> Actual Estimator is: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=37, max_features=38,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=92,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)



----------------------------------------------------------------------------------------------------------------

SEARCHING FINISH :
 The best recommder is RandomForestClassifier
 its tune parameter is OrderedDict([('bootstrap', True), ('criterion', 'gini'), ('max_depth', 37), ('max_features
', 38), ('n_estimators', 92)])
 its score is 0.5711920529801324
 its evaluation score is 0 (0 means do not have the evaluation set)
 its model train time is 1.8376047611236572
 The actually estimator is RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=37, max_features=38,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=92,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)

----------------------------------------------------------------------------------------------------------------

MODEL HAS BEEN SAVED IN  /Users/gleonardo/Desktop/研究生毕业设计/RS_thesis_for_polimi/Meeting2/experimentResults/
MODEL/best_model_EXAMPLE02_all_10000.m
INFO HAS BEEN SAVED IN  /Users/gleonardo/Desktop/研究生毕业设计/RS_thesis_for_polimi/Meeting2/experimentResults/I
NFO/best_model_EXAMPLE02_all_10000.npy
GLeonardodeMacBook-Pro:RS_thesis_for_polimi gleonardo$
```

- Here is the snapshot of Random Forest's best model.

The best model picked from all the algorithms is also shown in the left, and this model will be saved as the user's recommender model.

Results Display

Cluster Model Training Flow(1/5)



All user's profiles table contains all the general user's profiles.

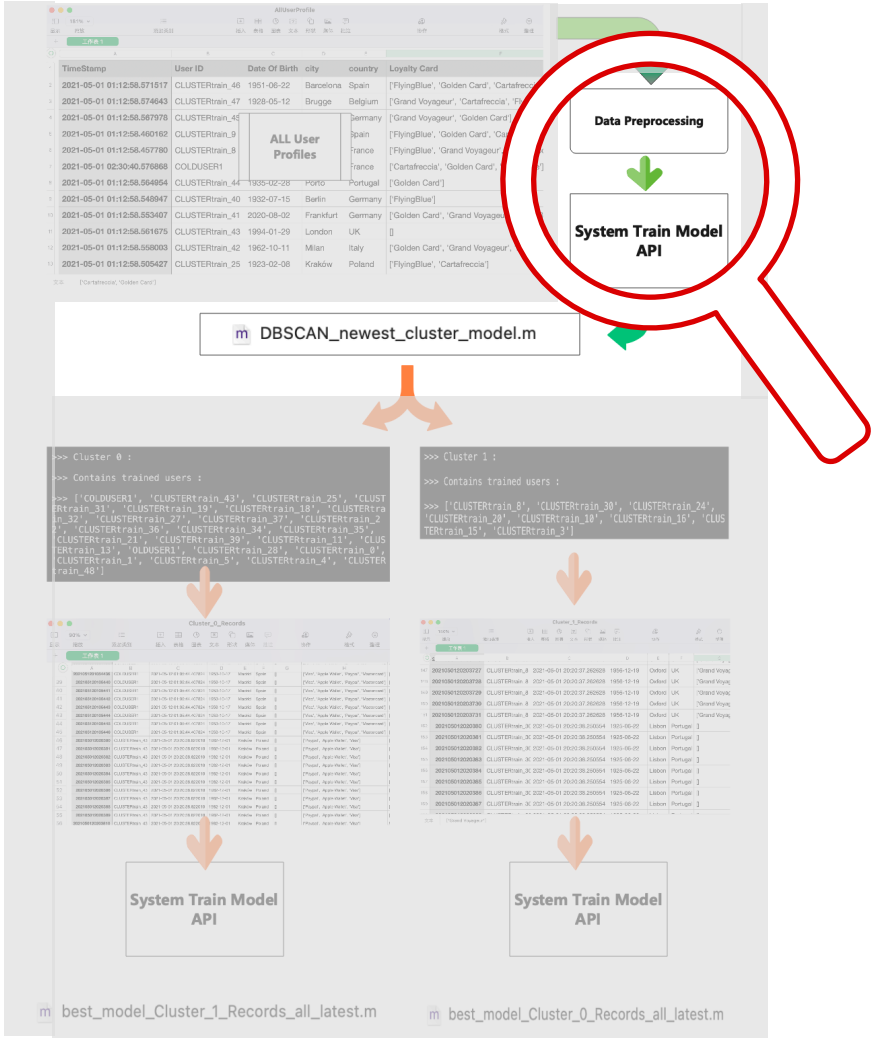
AllUserProfile

TimeStamp	User ID	Date Of Birth	city	country	Loyalty Card
2021-05-01 01:12:58.571517	CLUSTERtrain_46	1951-06-22	Barcelona	Spain	['FlyingBlue', 'Golden Card', 'Cartafreccia
2021-05-01 01:12:58.574643	CLUSTERtrain_47	1928-05-12	Brugge	Belgium	['Grand Voyageur', 'Cartafreccia', 'FlyingB
2021-05-01 01:12:58.567978	CLUSTERtrain_45	2008-02-13	Frankfurt	Germany	['Grand Voyageur', 'Golden Card']
2021-05-01 01:12:58.460162	CLUSTERtrain_9	2009-02-12	Madrid	Spain	['FlyingBlue', 'Golden Card', 'Cartafreccia
2021-05-01 01:12:58.457780	CLUSTERtrain_8	1954-06-04	Paris	France	['FlyingBlue', 'Grand Voyageur', 'Cartafrec
2021-05-01 02:30:40.576868	COLDUSER1	2016-07-10	Nice	France	['Cartafreccia', 'Golden Card', 'FlyingBlue
2021-05-01 01:12:58.564954	CLUSTERtrain_44	1935-02-28	Porto	Portugal	['Golden Card']
2021-05-01 01:12:58.548947	CLUSTERtrain_40	1932-07-15	Berlin	Germany	['FlyingBlue']
2021-05-01 01:12:58.553407	CLUSTERtrain_41	2020-08-02	Frankfurt	Germany	['Golden Card', 'Grand Voyageur', 'FlyingE
2021-05-01 01:12:58.561675	CLUSTERtrain_43	1994-01-29	London	UK	[]
2021-05-01 01:12:58.558003	CLUSTERtrain_42	1962-10-11	Milan	Italy	['Golden Card', 'Grand Voyageur', 'FlyingE
2021-05-01 01:12:58.505427	CLUSTERtrain_25	1923-02-08	Kraków	Poland	['FlyingBlue', 'Cartafreccia']

['Cartafreccia', 'Golden Card']

Results Display

Cluster Model Training Flow(2/5)



We use silhouette coefficient score to judge the performance of the model, and help the system find the best parameters settings.

```
test.API_USER_TRAIN(test.COLD_USER)
>>>Curr BEST PARAMETERS:

BEST_EPS : 11.350000000000033

BEST_MIN_SAMPLES : 8

BEST_SCORE : 0.009023677211671096



       eps  min_samples  n_clusters     score  ratio_outliners  outliners  \
105  11.35            8           2  0.009024         0.396226         21

       stats
105  [24  8]

================================================================

>>> The BEST parameters for DBSCAN algorithm is:

>>> eps : 11.350000000000033

>>> min_samples: 8

>>> its sihouette coeficient score is : 0.009023677211671096


CLUSTER MODEL HAS BEEN SAVED IN  /Users/gleonardo/Downloads/TravelOffer_Recom
R/CLUSTER_MODEL_INFO/Zero_cluserFitScaler.m
```

Results Display

Cluster Model Training Flow(3/5)

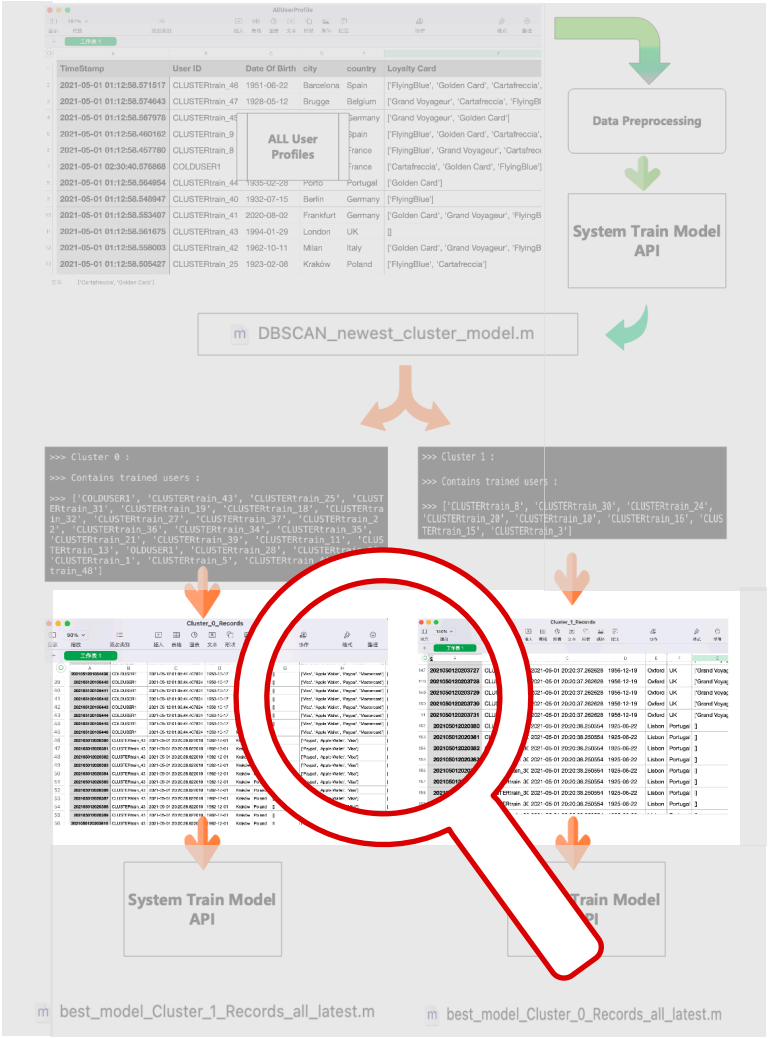
The user list in different clusters are then fetched by the system.

```
============================================================

>>> Cluster 0 :

>>> Contains trained users :

>>> ['COLDUSER1', 'CLUSTERtrain_43', 'CLUSTERtrain_25', 'CLUSTERtrain_31', 'CLUSTERtrain_19', 'CLUSTERtrain_18', 'CLUSTERtrain_32', 'CLUSTERtrain_27', 'CLUSTERtrain_37', 'CLUSTERtrain_22', 'CLUSTERtrain_36', 'CLUSTERtrain_34', 'CLUSTERtrain_35', 'CLUSTERtrain_21', 'CLUSTERtrain_39', 'CLUSTERtrain_11', 'CLUSTERtrain_13', 'OLDUSER1', 'CLUSTERtrain_28', 'CLUSTERtrain_0', 'CLUSTERtrain_1', 'CLUSTERtrain_5', 'CLUSTERtrain_4', 'CLUSTERtrain_48']

============================================================

>>> Cluster 1 :

>>> Contains trained users :

>>> ['CLUSTERtrain_8', 'CLUSTERtrain_30', 'CLUSTERtrain_24', 'CLUSTERtrain_20', 'CLUSTERtrain_10', 'CLUSTERtrain_16', 'CLUSTERtrain_15', 'CLUSTERtrain_3']

All the historical data for each cluster is ready!
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Start Searching The best recommender<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

Results Display

Cluster Model Training Flow(4/5)

All the historical records are collected and put into a new table as the cluster's historical records.



Cluster_0_Records

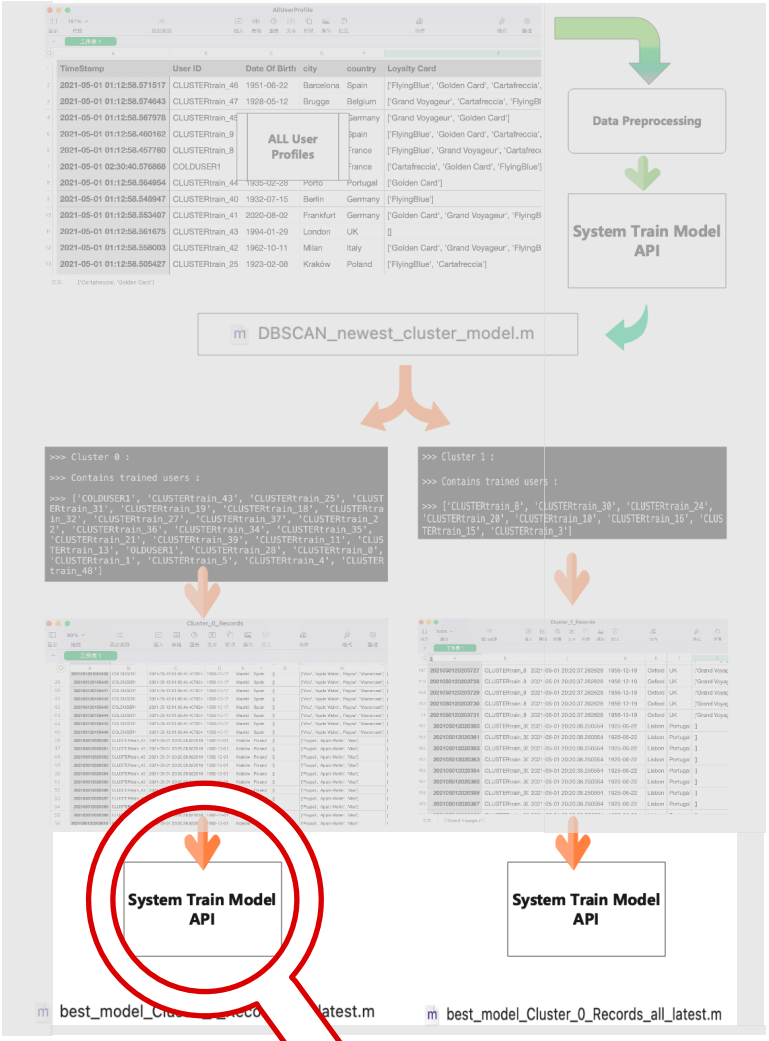
	A	B	C	D	E	F	G	H
	202105120105436	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
39	202105120105440	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
40	202105120105441	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
41	202105120105442	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
42	202105120105443	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
43	202105120105444	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
44	202105120105445	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
45	202105120105446	COLDUSER1	2021-05-12 01:05:44.407824	1953-10-17	Madrid	Spain	[]	['Visa', 'Apple Wallet', 'Paypal', 'Mastercard']
46	202105012020380	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
47	202105012020381	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
48	202105012020382	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
49	202105012020383	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
50	202105012020384	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
51	202105012020385	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
52	202105012020386	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']
53	202105012020387	CLUSTERtrain_43	2021-05-01 20:20:38.822010	1982-12-01	Kraków	Poland	[]	['Paypal', 'Apple Wallet', 'Visa']

Cluster_1_Records

	A	B	C	D	E	F	G
147	2021050120203727	CLUSTERtrain_8	2021-05-01 20:20:37.262628	1956-12-19	Oxford	UK	['Grand Voyag
148	2021050120203728	CLUSTERtrain_8	2021-05-01 20:20:37.262628	1956-12-19	Oxford	UK	['Grand Voyag
149	2021050120203729	CLUSTERtrain_8	2021-05-01 20:20:37.262628	1956-12-19	Oxford	UK	['Grand Voyag
150	2021050120203730	CLUSTERtrain_8	2021-05-01 20:20:37.262628	1956-12-19	Oxford	UK	['Grand Voyag
151	2021050120203731	CLUSTERtrain_8	2021-05-01 20:20:37.262628	1956-12-19	Oxford	UK	['Grand Voyag
152	202105012020380	CLUSTERtrain_30	2021-05-01 20:20:38.250554	1925-06-22	Lisbon	Portugal	[]
153	202105012020381	CLUSTERtrain_30	2021-05-01 20:20:38.250554	1925-06-22	Lisbon	Portugal	[]
154	202105012020382	CLUSTERtrain_30	2021-05-01 20:20:38.250554	1925-06-22	Lisbon	Portugal	[]

Results Display

Cluster Model Training Flow(5/5)



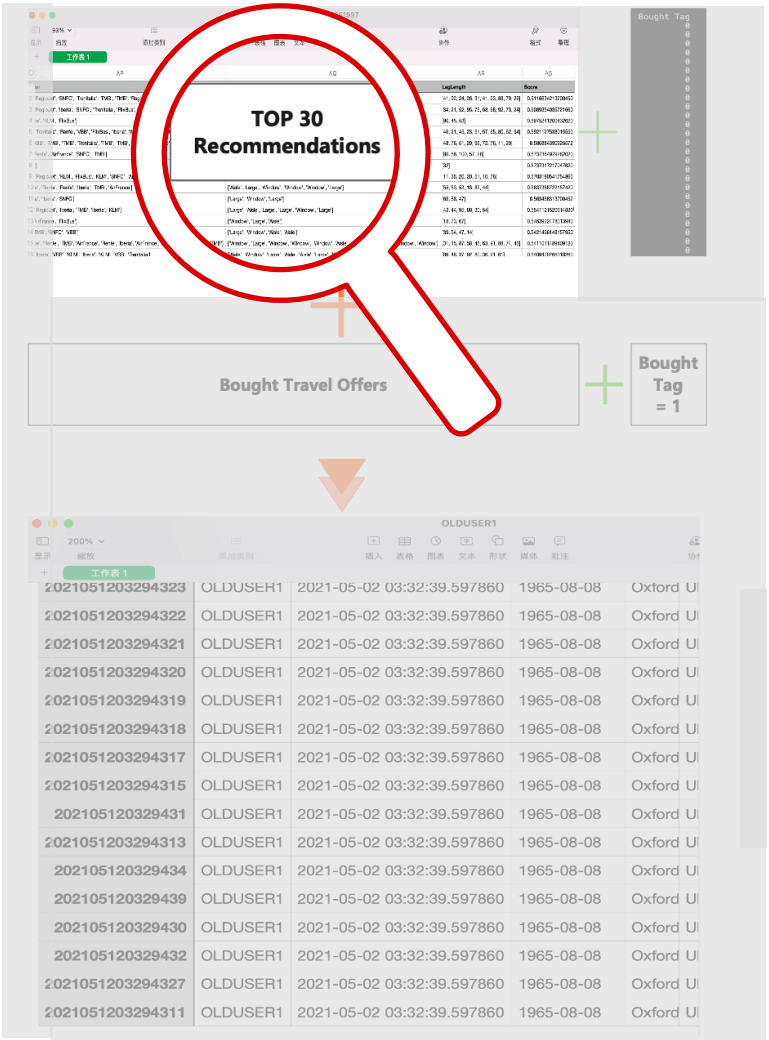
Cluster can be seen as a general user cause it has its own historical records, and the classifier algorithms can be used for getting the predefined recommender model.

```
SEARCHING FINISH :
 The best recommder is DecisionTreeClassifier
 its tune parameter is OrderedDict([('criterion', 'entropy'), ('max_depth', 3), ('max_features', 16), ('max_leaf_nodes', 7), ('min_samples_leaf', 6)])
 its score is 0.5126050420168067
 its evaluation score is 0 (0 means do not have the evaluation set)
 its model train time is 0.22274398803710938
 The actually estimator is DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=3, max_features=16, max_leaf_nodes=7,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=6, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')

MODEL HAS BEEN SAVED IN  /Users/gleonardo/Downloads/TravelOffer_Recommendation-main/TEST_DATA/CLUSTER_FOLDER/CLUSTER_REC_MODEL/best_model_Cluster_0_Records_all_latest.m
INFO HAS BEEN SAVED IN  /Users/gleonardo/Downloads/TravelOffer_Recommendation-main/TEST_DATA/CLUSTER_FOLDER/CLUSTER_REC_MODEL/best_model_Cluster_0_Records_all_latest.npy
```

```
SEARCHING FINISH :
 The best recommder is KNeighborsClassifier_distance
 its tune parameter is OrderedDict([('algorithm', 'kd_tree'), ('n_neighbors', 5), ('p', 6)])
 its score is 0.5225496091401083
 its evaluation score is 0 (0 means do not have the evaluation set)
 its model train time is 9.881953954696655
 The actually estimator is KNeighborsClassifier(algorithm='kd_tree', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=6,
                     weights='distance')

MODEL HAS BEEN SAVED IN  /Users/gleonardo/Downloads/TravelOffer_Recommendation-main/TEST_DATA/CLUSTER_FOLDER/CLUSTER_REC_MODEL/best_model_Cluster_1_Records_all_latest.m
INFO HAS BEEN SAVED IN  /Users/gleonardo/Downloads/TravelOffer_Recommendation-main/TEST_DATA/CLUSTER_FOLDER/CLUSTER_REC_MODEL/best_model_Cluster_1_Records_all_latest.npy
The ClusterNo for the user is : [0]
DONE! THE Pre trained model has been copied into the cold user
```

Results Display

User Feedback Flow(1/3)



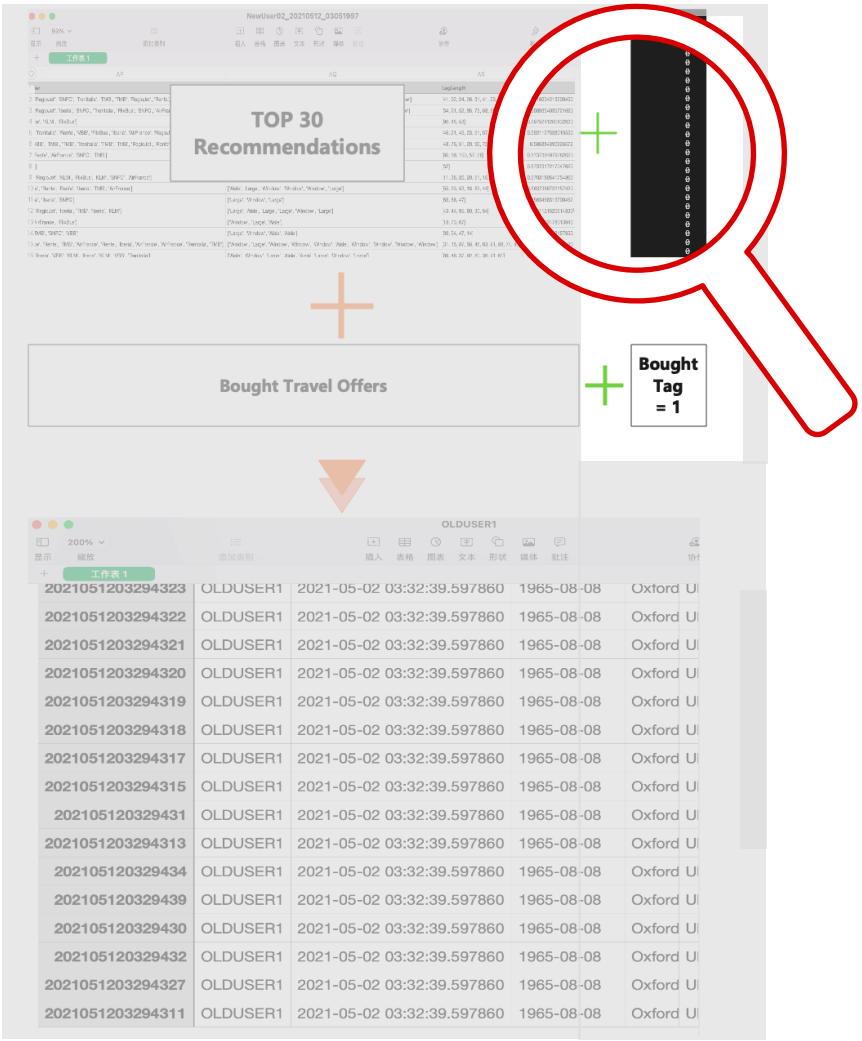
Top 30 recommendations will be recommended to the user, and the table contains all the information about the travel offer, user profile and score.

OLDUSER1 _20210512_03051997

	AP	AQ	AR	AS
1	ler	LegSeat	LegLength	Score
2	'RegioJet', 'SNFC', 'Trenitalia', 'TMB', 'TMB', 'RegioJet', 'Renfe', 'AirFrance', 'Renfe']	['Window', 'Aisle', 'Window', 'Large', 'Aisle', 'Large', 'Aisle', 'Window', 'Aisle', 'Window']	[41, 92, 24, 26, 31, 41, 93, 88, 79, 26]	0.6116634213200450
3	'RegioJet', 'Iberia', 'SNFC', 'Trenitalia', 'FlixBus', 'SNFC', 'AirFrance', 'Iberia', 'FlixBus']	['Aisle', 'Window', 'Large', 'Large', 'Window', 'Large', 'Large', 'Aisle', 'Large', 'Window']	[34, 21, 52, 95, 75, 68, 68, 92, 73, 24]	0.6089354035721660
4	:e', 'KLM', 'FlixBus']	['Large', 'Window', 'Large']	[90, 45, 53]	0.5975241200432620
5	'Trenitalia', 'Renfe', 'VBB', 'FlixBus', 'Iberia', 'AirFrance', 'RegioJet', 'TMB', 'FlixBus']	['Aisle', 'Large', 'Large', 'Large', 'Large', 'Large', 'Window', 'Large', 'Large', 'Aisle']	[46, 21, 45, 23, 51, 57, 65, 80, 62, 94]	0.5921127538015550
6	/BB', 'TMB', 'TMB', 'Trenitalia', 'TMB', 'TMB', 'RegioJet', 'Renfe']	['Aisle', 'Window', 'Aisle', 'Window', 'Window', 'Aisle', 'Large', 'Large', 'Large']	[48, 76, 51, 29, 96, 72, 76, 11, 20]	0.586884390926572
7	Renfe', 'AirFrance', 'SNFC', 'TMB']	['Large', 'Aisle', 'Aisle', 'Window', 'Large']	[80, 56, 100, 57, 28]	0.5737364979282020
8	]	['Window']	[32]	0.5737317217047620
9	'RegioJet', 'KLM', 'FlixBus', 'KLM', 'SNFC', 'AirFrance']	['Large', 'Large', 'Window', 'Window', 'Window', 'Window', 'Window']	[11, 35, 20, 20, 51, 10, 76]	0.5703180541754860
10	a', 'Renfe', 'Renfe', 'Iberia', 'TMB', 'AirFrance']	['Aisle', 'Large', 'Window', 'Window', 'Window', 'Large']	[55, 53, 53, 18, 82, 44]	0.5637388722157420
11	a', 'Iberia', 'SNFC']	['Large', 'Window', 'Large']	[63, 88, 47]	0.563468313700452
12	'RegioJet', 'Iberia', 'TMB', 'Iberia', 'KLM']	['Large', 'Aisle', 'Large', 'Large', 'Window', 'Large']	[43, 44, 95, 80, 30, 54]	0.5541121520614830
13	AirFrance', 'FlixBus']	['Window', 'Large', 'Aisle']	[18, 73, 67]	0.5463922478013940
14	TMB', 'SNFC', 'VBB']	['Large', 'Window', 'Aisle', 'Aisle']	[39, 54, 47, 14]	0.5421466148157950
15	:e', 'Renfe', 'TMB', 'AirFrance', 'Renfe', 'Iberia', 'AirFrance', 'AirFrance', 'Trenitalia', 'TMB']	['Window', 'Large', 'Window', 'Window', 'Window', 'Aisle', 'Window', 'Window', 'Window', 'Window']	[31, 15, 87, 58, 46, 63, 91, 88, 74, 46]	0.5411011489409120
16	'Iberia', 'VBB', 'KLM', 'Iberia', 'KLM', 'VBB', 'Trenitalia']	['Aisle', 'Window', 'Large', 'Aisle', 'Aisle', 'Large', 'Window', 'Large']	[89, 48, 37, 92, 80, 36, 91, 61]	0.5409470766310260

Results Display

User Feedback Flow(2/3)



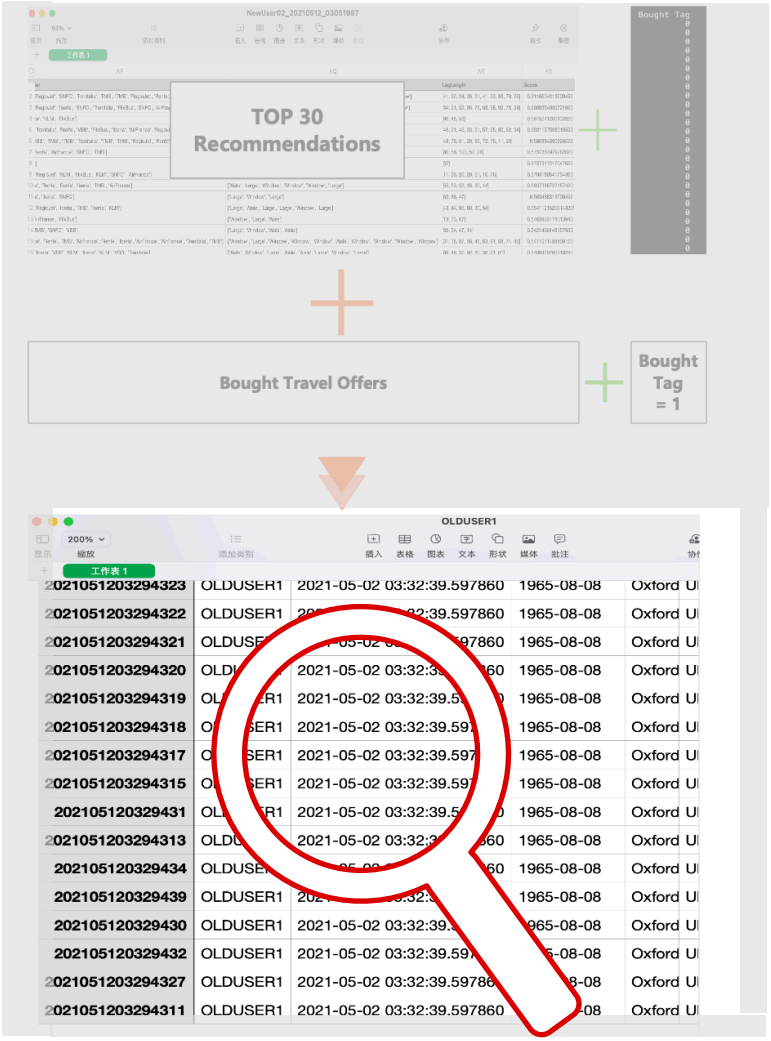
Bought Tag
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0

All the recommended records will have 0 Bought Tag value if user do not choose them.

And the travel offer user bought in the end will be tagged 1.

Results Display

User Feedback Flow(3/3)



Then, the feedback data will be joint into the user's historical records.

Once a new user has more than 100 records(number can be modified), the new user can get their recommender model trained with their own data.

```
[8]
boughtList = [2021051203294311]

[9]
test.API_USER_FEEDBACK(test.OLD_USER,boughtList,response_code)

[2021051203294311]
After Buying, the following historical records will be recorded:
    Travel Offer ID   User ID              TimeStamp Date Of Birth  \
0   2021051203294314  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
1   2021051203294324  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
2   2021051203294316  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
3   2021051203294310  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
4    202105120329437  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
5   2021051203294312  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
6   2021051203294328  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
7    202105120329436  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
8    202105120329435  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
9    202105120329433  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
10   202105120329438  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
11  2021051203294326  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
12  2021051203294325  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
13  2021051203294323  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
14  2021051203294322  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
15  2021051203294321  OLDUSER1  2021-05-02 03:32:39.597860    1965-08-08
```