OLX Code & the Curious

Popularity Based Ad Recommender System

Akhil Punia

https://www.linkedin.com/in/akhil-punia-0922bab6/

30 July 2017

Part 1- Dataset Exploration

Comments on Dataset Quality & Errors Found

(Refer to OLX-Data-Analysis-Graphlab.ipynb)

#1. ads_data.csv

Most redundant variables were ***latitude, longitude and product description.***
Most useful variables include ***ad_id, category_id , seller_id , title , price and source.***

ad_id

dtype:	int
num_unique (est.):	5,70,262
num_undefined:	0
min:	0
max:	29,22,041
median:	23,75,611
mean:	21,08,882.133
std:	7,57,628.493

category_id

dtype:	int
num_unique (est.):	14
num_undefined:	0
min:	5
max:	888
median:	815
mean:	810.952
std:	95.939

seller_id

dtype:	int
num_unique (est.):	1,00,956
num_undefined:	0
min:	2
max:	6,46,815
median:	3,86,256
mean:	3,66,124.074
std:	1,88,590.59

creation_time

dtype:	str
num_unique (est.):	5,57,693
num_undefined:	0

frequent items:

No values appear with ≥ 0.01% occurrence.

title

dtype:	str
num_unique (est.):	4,30,873
num_undefined:	0

frequent items:

Campera
Zapatos
Vestido
Zapatillas
Remera
Vestido de fiesta
Botas
Cartera
Camisa
Jeans
Campera de cuero
Musculosa

description

dtype:	str
num_unique (est.):	4,62,669
num_undefined:	0

frequent items:

''
Excelente estado
Nuevo
Talle M
Muy buen estado
Talle S
Nueva
Buen estado
Sin uso
Talle L
Impecable
En buen estado

price

dtype:	int
num_unique (est.):	2,158
num_undefined:	0
min:	0
max:	1.127e+8
median:	300
mean:	4,013.705
std:	1,77,488.926

distribution of values:

lat

dtype:	str
num_unique (est.):	49,621
num_undefined:	0

frequent items:

''
-34.60373306
-34.60399999
0E-8
-34.59000015
-34.57099999
-34.70381546
-34.57114959
-34.65490729
-34.60373450
-34.56208440
-34.52812060

#2. userdata.csv

Most redundant variables were ***ad_messges, user_long and user_lat***.
Most useful variables include ***user_id, event , channel , origin , ad_id , images_count and ad_views.***

user_id

dtype:	int
num_unique (est.):	12,284
num_undefined:	0
min:	1
max:	15,065
median:	7,237
mean:	7,308.38
std:	4,344.247

distribution of values:

event

dtype:	str
num_unique (est.):	2
num_undefined:	0

frequent items:

- view
- first_message

channel

dtype:	str
num_unique (est.):	3
num_undefined:	0

frequent items:

- android
- ios
- desktop

user_lat

dtype:	float
num_unique (est.):	1,36,414
num_undefined:	0
min:	-54.84
max:	52.31
median:	-34.606
mean:	-34.596
std:	1.111

distribution of values:

origin

dtype:	str
num_unique (est.):	9
num_undefined:	0

frequent items:

- home
- search
- notification_cente ...
- browse
- browse_search
- push
- drawer
- ''
- deeplink

ad_id

dtype:	int
num_unique (est.):	5,23,179
num_undefined:	0
min:	9
max:	28,74,067
median:	25,85,021
mean:	22,74,923.968
std:	6,68,617.411

distribution of values:

images_count

dtype:	str
num_unique (est.):	16
num_undefined:	0

frequent items:

- 1.0
- 2.0
- 3.0
- 4.0
- 5.0
- 6.0
- ''
- 8.0
- 7.0
- 0.0
- 10.0
- 9.0

ad_impressions

dtype:	float
num_unique (est.):	4,404
num_undefined:	0
min:	0
max:	6,94,012
median:	0
mean:	1,745.52
std:	20,979.317

distribution of values:

Part 2- Data Handling

Summary of Dataset Preprocessing Steps

Step 1: Reformatting user_messages.csv

(Refer to OLX-User-Data-Restructuring.ipynb)

Step 2: Segmenting the User Message Data based on 10 Unique Categories

In [77]:
```
user_messages.sort_values('category_id',inplace=True)
user_messages.head(5)
```

Out[77]:

	user_id	category_id	ad_id
12167	7597	362	1730320
19091	11938	362	2388804
11279	6452	362	1238103
11280	6452	362	2187575
11281	6452	362	2863136

Step 3: Finding the the Ads with the Highest Frequency in the Given Category ID and sorting them in Descending Order

In [109]:
```
pred = list()
for i in xrange(10):
    df1 = user_messages[user_messages['category_id']== category[i]]
    ads_all = list(df1['ad_id'])
    ads_uni = df1['ad_id'].unique()

    add_freq = list()

    for i in xrange(len(ads_uni)):
        c = ads_all.count(ads_uni[i])
        add_freq.append(c)

    d = {'Frequency': add_freq,'ad_id': ads_uni}
    d1 = df(data = d)

    d1.sort_values('Frequency',ascending = False,inplace = True)
```

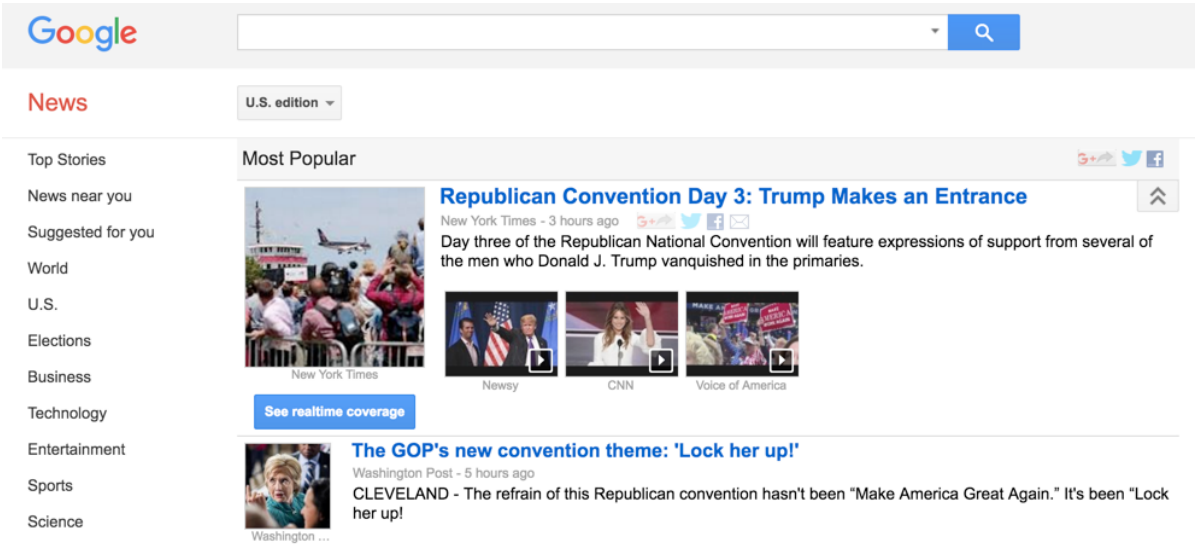
Part 3- Model Justification

Why I chose Popularity based recommender system model?

The **simplest approach** could be to recommend the ads which will be suitable for most number of users. This is a blazing fast and dirty approach and thus has a major drawback.

The things is, there is **no personalisation involved** with this approach.

Basically, the 10 most popular ads within a particular category (1 out of 10 at a time) would be same for each user since, popularity is defined on the entire user pool. So everybody will see the same ads. It sounds like, 'a website recommends you to buy HP Laptop in the Electronics category just because it's been liked by other users and doesn't care if you are even interested in buying or not'.



Surprisingly, such approach still works in places like news portals. Whenever you login to say Google News where we will see a column of "Popular News" which is subdivided into sections and the most read articles of each sections are displayed. This approach works in this case because:

- There is division by section so user can look at the section of his interest.
- At a time there are only a few hot topics and there is a high chance that a user wants to read the news which is being read by most others

Similarly, it can be used OLX dataset as the users are divided into 10 separate categories.