EDA and Data Cleaning

May 8, 2020

1 EDA and Data Cleaning

- 1. Data
  - 1.1. Data Overview
  - 1.2. Data Manipulation
- 2. Exploratory Data Analysis
  - 2.1. Hotel Cluster Distribution
  - 2.2. Other Attrition Distribution
  - 1.1. Data Overview
- 3. Data Processing
  - 3.1. Filling Missing Value
  - 3.2. Target Encoding
  - 3.3. Ref Variable Creation

In [1]:
```
#Importing libraries
import zipfile
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
# Input data files are available in the "../input/" directory.
import os
import matplotlib.pyplot as plt#visualization
from PIL import  Image
%matplotlib inline
import pandas as pd
import seaborn as sns#visualization
import itertools
import warnings
warnings.filterwarnings("ignore")
import io
import plotly.offline as py#visualization
py.init_notebook_mode(connected=True)#visualization
import plotly.graph_objs as go#visualization
import plotly.tools as tls#visualization
import plotly.figure_factory as ff#visualization
```

```
import plotly.express as px

start_time = pd.datetime.now()
```

2 1.Data

2.1 1.1.Data Overview

In [3]:
```
%%time
#load file from local zip
zf = zipfile.ZipFile('expedia-hotel-recommendations.zip')
df = pd.read_csv(zf.open('train.csv'))
```

```
CPU times: user 1min 27s, sys: 43.7 s, total: 2min 11s
Wall time: 2min 34s
```

In [5]:
```
#load file from s3 with pyspark
'''
def read_csv_s3(path):
    import os
    from pyspark.sql import SparkSession
    AWS_USER = os.environ.get('AWSUSER')
    AWS_PASSWORD = os.environ.get('AWSPASSWORD')
    os.environ['PYSPARK_SUBMIT_ARGS'] = ("--packages=org.apache.hadoop:hadoop-aws:2.7.
    spark = SparkSession.builder.appName('S3CSVRead').getOrCreate()
    spark._jsc.hadoopConfiguration().set("fs.s3a.impl", "org.apache.hadoop.fs.s3native
    spark._jsc.hadoopConfiguration().set("fs.s3a.awsAccessKeyId", AWS_USER)
    spark._jsc.hadoopConfiguration().set("fs.s3a.awsSecretAccessKey", AWS_PASSWORD)
    df= spark.read.csv(path, header=True)
    return df,spark
'''
```

In [6]:
```
#%%time
#path="s3a://593research/train.csv"
#df,spark = read_csv_s3()
```

```
CPU times: user 2.86 ms, sys: 1.68 ms, total: 4.54 ms
Wall time: 567 ms
```

In [168]:
```
len(df)
```

Out[168]:
```
37670293
```

In [4]:
```
df.head()
```

Out[4]:
```
             date_time  site_name  posa_continent  user_location_country  \
0  2014-08-11 07:46:59          2               3                     66
```

```
1  2014-08-11 08:22:12         2               3                    66
2  2014-08-11 08:24:33         2               3                    66
3  2014-08-09 18:05:16         2               3                    66
4  2014-08-09 18:08:18         2               3                    66

   user_location_region  user_location_city  orig_destination_distance  \
0                   348               48862                  2234.2641
1                   348               48862                  2234.2641
2                   348               48862                  2234.2641
3                   442               35390                   913.1932
4                   442               35390                   913.6259

   user_id  is_mobile  is_package  ...  srch_children_cnt srch_rm_cnt  \
0       12          0           1  ...                  0           1
1       12          0           1  ...                  0           1
2       12          0           0  ...                  0           1
3       93          0           0  ...                  0           1
4       93          0           0  ...                  0           1

  srch_destination_id  srch_destination_type_id  is_booking  cnt  \
0                8250                         1           0    3
1                8250                         1           1    1
2                8250                         1           0    1
3               14984                         1           0    1
4               14984                         1           0    1

   hotel_continent  hotel_country  hotel_market  hotel_cluster
0                2             50           628              1
1                2             50           628              1
2                2             50           628              1
3                2             50          1457             80
4                2             50          1457             21

[5 rows x 24 columns]
```

In [173]:
```
#randomly sample 1% from training dataset for data manipulation and training
data = df.sample(frac=0.01, random_state=99)
```

In [174]:
```
%%time

print ("Rows     : " ,data.shape[0])
print ("Columns  : " ,data.shape[1])
print ("\nFeatures : \n" ,data.columns.tolist())
print ("\nMissing values :  ", data.isnull().sum().values.sum())
print ("\nUnique values :  \n",data.nunique())
```

```
Rows     :  376703
Columns  :  24
```

```
Features :
 ['date_time', 'site_name', 'posa_continent', 'user_location_country', 'user_location_region',

Missing values :   136337

Unique values :
 date_time                    375047
site_name                        42
posa_continent                    5
user_location_country           213
user_location_region            871
user_location_city            18645
orig_destination_distance    223465
user_id                      262797
is_mobile                         2
is_package                        2
channel                          11
srch_ci                        1105
srch_co                        1106
srch_adults_cnt                  10
srch_children_cnt                10
srch_rm_cnt                       9
srch_destination_id           15938
srch_destination_type_id          9
is_booking                        2
cnt                              37
hotel_continent                   7
hotel_country                   197
hotel_market                   2045
hotel_cluster                   100
dtype: int64
CPU times: user 1.89 s, sys: 1.82 s, total: 3.71 s
Wall time: 4.35 s
```

In [175]:
```
##missing value percentage
percent_missing = data.isnull().sum() * 100 / len(data)
missing_value_df = pd.DataFrame({'column_name': data.columns,
                                 'percent_missing': percent_missing})
missing_value_df
```

Out[175]:

	column_name	percent_missing
date_time	date_time	0.000000
site_name	site_name	0.000000
posa_continent	posa_continent	0.000000
user_location_country	user_location_country	0.000000
user_location_region	user_location_region	0.000000

```
user_location_city                  user_location_city         0.000000
orig_destination_distance    orig_destination_distance        35.933613
user_id                                        user_id         0.000000
is_mobile                                    is_mobile         0.000000
is_package                                  is_package         0.000000
channel                                        channel         0.000000
srch_ci                                        srch_ci         0.129280
srch_co                                        srch_co         0.129280
srch_adults_cnt                        srch_adults_cnt         0.000000
srch_children_cnt                    srch_children_cnt         0.000000
srch_rm_cnt                                srch_rm_cnt         0.000000
srch_destination_id                srch_destination_id         0.000000
srch_destination_type_id      srch_destination_type_id         0.000000
is_booking                                  is_booking         0.000000
cnt                                                cnt         0.000000
hotel_continent                        hotel_continent         0.000000
hotel_country                            hotel_country         0.000000
hotel_market                              hotel_market         0.000000
hotel_cluster                            hotel_cluster         0.000000
```

In [41]:
```
data.describe()
```

Out[41]:
```
           site_name  posa_continent  user_location_country  \
count  376703.000000   376703.000000          376703.000000
mean        9.811220        2.679803              86.197123
std        11.985748        0.748484              59.239274
min         2.000000        0.000000               0.000000
25%         2.000000        3.000000              66.000000
50%         2.000000        3.000000              66.000000
75%        15.000000        3.000000              71.000000
max        53.000000        4.000000             239.000000

       user_location_region  user_location_city  orig_destination_distance  \
count         376703.000000       376703.000000              241340.000000
mean             308.132784        27793.453418                1970.497121
std              208.878863        16762.012048                2233.786783
min                0.000000            0.000000                   0.005600
25%              174.000000        13087.000000                 312.906825
50%              312.000000        27655.000000                1137.305950
75%              385.000000        42396.000000                2553.017375
max             1027.000000        56507.000000               12052.021000

            user_id      is_mobile     is_package        channel  ...  \
count  3.767030e+05  376703.000000  376703.000000  376703.000000  ...
mean   6.044845e+05       0.135207       0.248278       5.868459  ...
std    3.508230e+05       0.341945       0.432014       3.717852  ...
min    3.000000e+00       0.000000       0.000000       0.000000  ...
25%    2.983800e+05       0.000000       0.000000       2.000000  ...
```

```
50%    6.042430e+05      0.000000      0.000000       9.000000  ...
75%    9.109655e+05      0.000000      0.000000       9.000000  ...
max    1.198783e+06      1.000000      1.000000      10.000000  ...

       srch_children_cnt    srch_rm_cnt  srch_destination_id  \
count      376703.000000  376703.000000        376703.000000
mean            0.331980       1.112752         14440.410429
std             0.729092       0.458563         11075.148990
min             0.000000       0.000000             4.000000
25%             0.000000       1.000000          8267.000000
50%             0.000000       1.000000          9147.000000
75%             0.000000       1.000000         18790.000000
max             9.000000       8.000000         65068.000000

       srch_destination_type_id     is_booking            cnt  \
count             376703.000000  376703.000000  376703.000000
mean                   2.580457       0.078972       1.482130
std                    2.151231       0.269695       1.199764
min                    1.000000       0.000000       1.000000
25%                    1.000000       0.000000       1.000000
50%                    1.000000       0.000000       1.000000
75%                    5.000000       0.000000       2.000000
max                    9.000000       1.000000      49.000000

       hotel_continent  hotel_country   hotel_market  hotel_cluster
count    376703.000000  376703.000000  376703.000000  376703.000000
mean          3.155796      81.296385     600.239868      49.832327
std           1.622880      56.169040     511.170282      28.928797
min           0.000000       0.000000       0.000000       0.000000
25%           2.000000      50.000000     160.000000      25.000000
50%           2.000000      50.000000     597.000000      49.000000
75%           4.000000     106.000000     701.000000      73.000000
max           6.000000     212.000000    2117.000000      99.000000

[8 rows x 21 columns]
```

In [42]:
```
#to check data types
data.dtypes
```

Out[42]:
```
date_time                     object
site_name                      int64
posa_continent                 int64
user_location_country          int64
user_location_region           int64
user_location_city             int64
orig_destination_distance    float64
user_id                        int64
is_mobile                      int64
```

```
is_package                      int64
channel                         int64
srch_ci                        object
srch_co                        object
srch_adults_cnt                 int64
srch_children_cnt               int64
srch_rm_cnt                     int64
srch_destination_id             int64
srch_destination_type_id        int64
is_booking                      int64
cnt                             int64
hotel_continent                 int64
hotel_country                   int64
hotel_market                    int64
hotel_cluster                   int64
dtype: object
```

2.2 1.2.Data Manipulation

In [176]:
```
#change data types
columns = data.columns.tolist()
datetime_var = ['date_time','srch_ci','srch_co']

numeric_var = ['orig_destination_distance','is_mobile','is_package','srch_adults_cnt
               'is_booking','cnt']

cat_var = [i for i in columns if(i not in datetime_var and i not in numeric_var)]
```

In [177]:
```
def change_datatype():
    for i in datetime_var:
        data[i] = data[i].astype('datetime64[ns]')
    return data

data = change_datatype()
```

In [45]:
```
#check data types
data.dtypes
```

Out[45]:
```
date_time                     datetime64[ns]
site_name                              int64
posa_continent                         int64
user_location_country                  int64
user_location_region                   int64
user_location_city                     int64
orig_destination_distance            float64
user_id                                int64
is_mobile                              int64
is_package                             int64
channel                                int64
```

```
srch_ci                              datetime64[ns]
srch_co                              datetime64[ns]
srch_adults_cnt                               int64
srch_children_cnt                             int64
srch_rm_cnt                                   int64
srch_destination_id                           int64
srch_destination_type_id                      int64
is_booking                                    int64
cnt                                           int64
hotel_continent                               int64
hotel_country                                 int64
hotel_market                                  int64
hotel_cluster                                 int64
dtype: object
```

3 2.Exploratory Data Analysis

3.1 2.1.Hotel Cluster Distribution

In [45]:
```
#define a function to plot interactive distrbution graph
def distribution_plot(dataset,column,title,xtitle,ytitle):
    trace = go.Histogram(x=dataset[column], opacity=0.7, marker={"line": {"color": "#2
    layout = go.Layout(title=title, xaxis={"title": xtitle, "showgrid": False},
                       yaxis={"title": ytitle, "showgrid": False},plot_bgcolor='rgba(0
                      paper_bgcolor='rgba(0,0,0,0)') #showgrid:False to remove gridli
    figure = {"data": [trace], "layout": layout}

    py.iplot(figure)
```

In [46]:
```
distribution_plot(data,'hotel_cluster',f"Hotel Cluster Distribution","Hotel Cluster",'
```

In [48]:
```
#hotel cluster counts
data['hotel_cluster'].value_counts()
```

Out[48]:
```
91    10363
41     7792
48     7449
64     7063
65     6778
      ...
35     1359
53     1353
88     1098
27     1044
74      508
Name: hotel_cluster, Length: 100, dtype: int64
```

3.2 2.2.Other Attrition Distribution

In [55]:
```
plot_data = data.copy()
plot_data['is_mobile'] = plot_data["is_mobile"].replace({1:"Yes",0:"No"})
plot_data['is_package'] = plot_data["is_package"].replace({1:"Yes",0:"No"})
plot_data['is_booking'] = plot_data["is_booking"].replace({1:"Yes",0:"No"})
book = plot_data[plot_data['is_booking']=="Yes"]
notbook = plot_data[plot_data['is_booking']=="No"]
```

In [52]:
```
#define a function to plot distribution by booking decision
def histogram(column) :
    trace1 = go.Histogram(x  = book[column],
                          histnorm= "percent",
                          name = "Booked",
                          marker = dict(line = dict(width = .5,
                                                    color = "black"
                                                    )
                                        ),
                         opacity = .9
                         )

    trace2 = go.Histogram(x  = notbook[column],
                          histnorm = "percent",
                          name = "Not Booked",
                          marker = dict(line = dict(width = .5,
                                              color = "black"
                                             )
                                ),
                          opacity = .9
                         )

    data = [trace1,trace2]
    layout = go.Layout(dict(title =column + " distribution in booking attrition ",
                            plot_bgcolor  = "rgb(243,243,243)",
                            paper_bgcolor = "rgb(243,243,243)",
                            xaxis = dict(gridcolor = 'rgb(255, 255, 255)',
                                             title = column,
                                             zerolinewidth=1,
                                             ticklen=5,
                                             gridwidth=2
                                            ),
                            yaxis = dict(gridcolor = 'rgb(255, 255, 255)',
                                             title = "percent",
                                             zerolinewidth=1,
                                             ticklen=5,
                                             gridwidth=2
                                            ),
                           )
                      )
```

```
            fig  = go.Figure(data=data,layout=layout)

            py.iplot(fig)
```

```
In [57]: col = ['orig_destination_distance','is_mobile','is_package','channel','srch_adults_cnt
                      'srch_rm_cnt','srch_destination_type_id','hotel_continent']
         for i in col:
             histogram(i)
```

4 3. Data processing

4.1 3.1. Filling Missing Value

```
In [178]: #check number of rows that have missing values
          dis_na = data['orig_destination_distance'].isnull().values.ravel().sum()
          print(dis_na)
```

```
135363
```

```
In [179]: def filled_distance():
              dis_na = data['orig_destination_distance'].isnull().values.ravel().sum()
              #first step fill by the user lovation city and srch destimation id group
              #filled about 4k record, 131808 NA left
              data['orig_destination_distance'] = data.groupby(['user_location_city','srch_des
              lambda x: x.fillna(np.mean(x)))
              dis_na1 = data['orig_destination_distance'].isnull().values.ravel().sum()
              print('Filled {} records and there are {} NA records left'.format(dis_na-dis_na1
              #second step fill by the user location country and hotel country group
              #filled 64651 records and 67157 left
              data['orig_destination_distance'] = data.groupby(['user_location_city','hotel_cou
              lambda x: x.fillna(np.mean(x)))
              dis_na2 = data['orig_destination_distance'].isnull().values.ravel().sum()
              print('Filled {} records and there are {} NA records left'.format(dis_na1-dis_na2
              #third step fill by the user location country and search destination type id gro
              #filled 32507 records and 34650 left
              data['orig_destination_distance'] = data.groupby(['user_location_country','srch_d
              lambda x: x.fillna(np.mean(x)))
              dis_na3 = data['orig_destination_distance'].isnull().values.ravel().sum()
              print('Filled {} records and there are {} NA records left'.format(dis_na2-dis_na3
              #forth step fill by the posa continent group
              #filled all the missing value
              data['orig_destination_distance'] = data.groupby(['posa_continent'])['orig_destin
              lambda x: x.fillna(np.mean(x)))
              dis_na4 = data['orig_destination_distance'].isnull().values.ravel().sum()
              print('Filled {} records and there are {} NA records left'.format(dis_na3-dis_na4
              return data
```

In [180]: filled_distance()

```
Filled 3555 records and there are 131808 NA records left
Filled 9541 records and there are 122267 NA records left
Filled 82791 records and there are 39476 NA records left
Filled 39476 records and there are 0 NA records left
```

Out[180]:

```
                    date_time  site_name  posa_continent  \
14135162 2014-01-26 14:49:27         11               3
18984109 2014-01-13 19:33:04         22               2
26309215 2013-06-03 21:47:36          2               3
6926715  2014-01-02 16:11:51         13               1
21882923 2014-09-30 10:29:23         37               1
...                       ...        ...             ...
8251929  2014-09-23 19:55:38          2               3
32607965 2014-12-31 14:00:53         37               1
30217917 2014-08-06 08:08:43          2               3
29750021 2014-02-18 13:31:30         10               0
549466   2013-03-28 03:08:19          2               3

          user_location_country  user_location_region  user_location_city  \
14135162                    168                    45                7182
18984109                      0                   147               10568
26309215                     66                   184               11878
6926715                      46                   171               19639
21882923                     69                   573               33444
...                         ...                   ...                 ...
8251929                     194                    38               42328
32607965                     69                   926               53290
30217917                     66                   220                2086
29750021                    182                   416               32100
549466                       66                   348               48862

          orig_destination_distance  user_id  is_mobile  is_package  ...  \
14135162                1862.949101   178107          0           0  ...
18984109                2854.719471  1024412          0           0  ...
26309215                 648.255400   216672          0           0  ...
6926715                 5566.765700  1179327          0           0  ...
21882923                3419.160513   128601          0           0  ...
...                             ...      ...        ...         ...  ...
8251929                 1862.949101   118510          0           0  ...
32607965                3419.160513   302897          0           0  ...
30217917                 528.022400  1184505          1           1  ...
29750021                5041.088800  1185797          0           0  ...
549466                   209.120700   667832          0           0  ...

          srch_children_cnt srch_rm_cnt srch_destination_id  \
```

```
14135162                  0           1             33648
18984109                  0           1               468
26309215                  2           1             26663
6926715                   0           1             22890
21882923                  0           1             23479
...                     ...         ...               ...
8251929                   0           1              8242
32607965                  0           1             25967
30217917                  1           1              1152
29750021                  0           1              8218
549466                    0           1              8291

          srch_destination_type_id  is_booking  cnt  hotel_continent  \
14135162                         1           1    1                5
18984109                         1           0    1                3
26309215                         1           0    4                2
6926715                          5           0    1                3
21882923                         6           0    5                6
...                            ...         ...  ...              ...
8251929                          1           0    1                3
32607965                         6           0    1                6
30217917                         1           0    1                4
29750021                         1           0    2                2
549466                           1           0    1                2

          hotel_country  hotel_market  hotel_cluster
14135162            194          1555             76
18984109             48           153              9
26309215             50           707             15
6926715             182            46             57
21882923             70            19             21
...                 ...           ...            ...
8251929             171            61             37
32607965             70           134             18
30217917             47          1502             65
29750021             50           743             70
549466               50           191             91

[376703 rows x 24 columns]
```

In [ ]:
```
print('duration: ',pd.datetime.now() - start_time)
```

4.2 3.2. Target Encoding

In [49]:
```
target = ['hotel_cluster']
userid = ['user_id']
cat_var = [i for i in cat_var if i not in target]+userid
print(target)
```

```
print(cat_var)
```

```
['hotel_cluster']
['site_name', 'posa_continent', 'user_location_country', 'user_location_region', 'user_locatio
```

```
In [50]: #train test split
         from sklearn.model_selection import train_test_split
         train,test = train_test_split(data,test_size =.25 ,random_state = 111)
         cols    = [i for i in data.columns if i not in target]
         train_X = train[cols]
         train_Y = train[target]
         test_X  = test[cols]
         test_Y  = test[target]
         train1 = train.copy()
```

```
In [18]: def target_encoder(df,df_toencoded, column, target, method='mean'):
             if method == 'mean':
                 df1 = df.groupby(column)[target].mean().reset_index()
             elif method == 'median':
                 df1 = df.groupby(column)[target].median().reset_index()
             elif method == 'std':
                 df1 = df.groupby(column)[target].std().reset_index()
             elif method == 'mode':
                 df1 = df.groupby(column)[target].apply(pd.Series.mode).reset_index()
             else:
                 raise ValueError("Incorrect method supplied: '{}'. Must be one of 'mean', 'me

             encode_dict = {k:v for k, v in zip(df1[column],df1[target])}
             encoded_column = [encode_dict[k] for k in df_toencoded[column]]

             return encoded_column

         encode_col = ['hotel_country']

         #for i in encode_col:
         #    train[i] = target_encoder(train1,train,i,'hotel_cluster','mode')

         #train.head()
```

```
Out[18]:                    date_time  site_name  posa_continent  \
         34608371 2014-05-23 09:42:03          2               3
         33215220 2014-08-24 10:52:53         24               2
         31322656 2013-12-26 19:15:58         34               3
         9852244  2014-06-12 01:38:38          2               3
         3315432  2014-08-09 08:03:05         11               3

                   user_location_country  user_location_region  user_location_city  \
         34608371                     66                   348               48862
```

```
33215220                      3                 70            32441
31322656                    205                354            16084
9852244                      66                462            46651
3315432                     205                354            49492

          orig_destination_distance  user_id  is_mobile  is_package  ...  \
34608371                3831.303900   666517          0           0  ...
33215220                7363.881774   929132          0           0  ...
31322656                2096.279100   457168          0           0  ...
9852244                  861.480200   752047          0           0  ...
3315432                 6131.312300   206212          0           1  ...

          srch_children_cnt srch_rm_cnt srch_destination_id  \
34608371                  0           1                8788
33215220                  0           1                8785
31322656                  0           1                8288
9852244                   0           1                8250
3315432                   0           1                8863

          srch_destination_type_id  is_booking  cnt  hotel_continent  \
34608371                         1           0    1                6
33215220                         1           0    1                6
31322656                         1           0    1                2
9852244                          1           0    1                2
3315432                          1           0    1                0

          hotel_country  hotel_market  hotel_cluster
34608371             82          1880             46
33215220              8            35             58
31322656             55           399             95
9852244              91           628             45
3315432              92            59             78

[5 rows x 24 columns]
```

4.3 3.3. Ref Variable Creation

In [161]:
```
##previous hotel cluster
train_user = set(train.user_id)
test_user = set(test.user_id)
new_user = test_user-train_user
new_user = list(new_user)
```

In [52]:
```
train['rec_hotel_cluster'] = train.groupby(['user_id'])['hotel_cluster'].transform(
    lambda x:pd.Series.mode(x)[0])
```

In [ ]:
```
#for existing user, match rec_hotel_cluster
#for new_user, find similar user
```

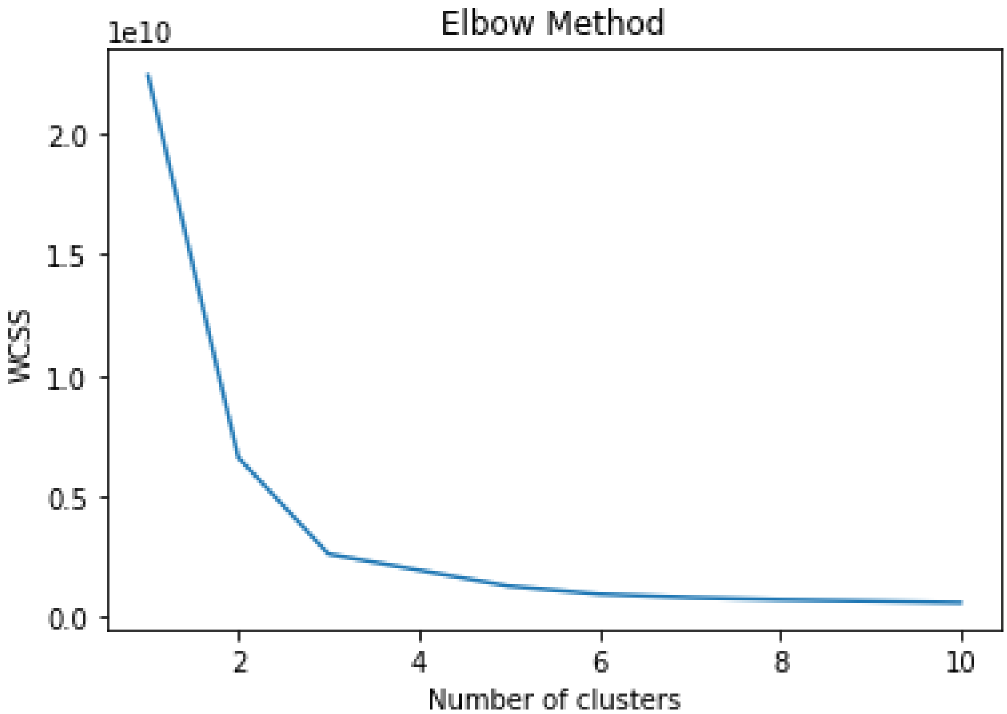
```
In [56]: data[data['user_id'].isin(train_user)]
```

```
Out[56]:                     date_time  site_name  posa_continent  \
         14135162 2014-01-26 14:49:27         11               3
         18984109 2014-01-13 19:33:04         22               2
         26309215 2013-06-03 21:47:36          2               3
         6926715  2014-01-02 16:11:51         13               1
         21882923 2014-09-30 10:29:23         37               1
         ...                      ...        ...             ...
         34757536 2013-01-25 00:22:37         24               2
         8251929  2014-09-23 19:55:38          2               3
         32607965 2014-12-31 14:00:53         37               1
         29750021 2014-02-18 13:31:30         10               0
         549466   2013-03-28 03:08:19          2               3

                  user_location_country  user_location_region  user_location_city  \
         14135162                   168                    45                7182
         18984109                     0                   147               10568
         26309215                    66                   184               11878
         6926715                     46                   171               19639
         21882923                    69                   573               33444
         ...                        ...                   ...                 ...
         34757536                     3                    48               48781
         8251929                    194                    38               42328
         32607965                    69                   926               53290
         29750021                   182                   416               32100
         549466                      66                   348               48862

                  orig_destination_distance  user_id  is_mobile  is_package  ...  \
         14135162               1877.175685   178107          0           0  ...
         18984109               2854.719471  1024412          0           0  ...
         26309215                648.255400   216672          0           0  ...
         6926715                5566.765700  1179327          0           0  ...
         21882923               2083.922543   128601          0           0  ...
         ...                            ...      ...        ...         ...  ...
         34757536               8300.060200    12803          0           0  ...
         8251929                1877.175685   118510          0           0  ...
         32607965               2083.922543   302897          0           0  ...
         29750021               5041.088800  1185797          0           0  ...
         549466                  209.120700   667832          0           0  ...

                  srch_children_cnt srch_rm_cnt srch_destination_id  \
         14135162                 0           1               33648
         18984109                 0           1                 468
         26309215                 2           1               26663
         6926715                  0           1               22890
         21882923                 0           1               23479
         ...                    ...         ...                 ...
```

```
34757536                  0          1              8253
8251929                   0          1              8242
32607965                  0          1             25967
29750021                  0          1              8218
549466                    0          1              8291

          srch_destination_type_id  is_booking  cnt  hotel_continent  \
14135162                         1           1    1                5
18984109                         1           0    1                3
26309215                         1           0    4                2
6926715                          5           0    1                3
21882923                         6           0    5                6
...                            ...         ...  ...              ...
34757536                         1           0    1                6
8251929                          1           0    1                3
32607965                         6           0    1                6
29750021                         1           0    2                2
549466                           1           0    1                2

          hotel_country  hotel_market  hotel_cluster
14135162            194          1555             76
18984109             48           153              9
26309215             50           707             15
6926715             182            46             57
21882923             70            19             21
...                 ...           ...            ...
34757536             70            19             25
8251929             171            61             37
32607965             70           134             18
29750021             50           743             70
549466               50           191             91

[322406 rows x 24 columns]
```

In [136]:
```
#get unique user_id record and plot kmeans elbow plot to find optimal number of clus
test['earlist'] = test.groupby(['user_id'])['date_time'].transform(
    lambda x:x.min())
test_df = test[test['date_time']==test['earlist']]
from sklearn.datasets.samples_generator import make_blobs
from sklearn.cluster import KMeans
X_df = test_df[['site_name','user_location_country','hotel_country','hotel_market','i
X_df = X_df.set_index('user_id')
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_s
    kmeans.fit(X_df)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
```

```
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```




```
In [137]: kmeans = KMeans(n_clusters=3, init='k-means++', max_iter=300, n_init=10, random_state
          pred_y = kmeans.fit_predict(X_df)
          X_df['cluster'] = pred_y
          X_df = X_df.reset_index()
          cluster_0 = list(set(X_df[X_df['cluster']==0]['user_id']))
          cluster_1 = list(set(X_df[X_df['cluster']==1]['user_id']))
          cluster_2 = list(set(X_df[X_df['cluster']==2]['user_id']))
```

```
In [154]: def rec_cluster():
              #find exisiting user then match the rec_hotel_cluster value based on the train d
              cluster_dict = {k:v for k,v in zip(train['user_id'],train['rec_hotel_cluster'])}
              df_old = test[test['user_id'].isin(train_user)]
              df_new = test[~test['user_id'].isin(train_user)]
              col_old = [cluster_dict[k] for k in df_old['user_id']]
              df_old['rec_hotel_cluster'] = col_old
              #for new user, find mode in similar cluster
              df_new0 = df_new[df_new['user_id'].isin(cluster_0)]
              df_new0['rec_hotel_cluster'] = df_new0['hotel_cluster'].mode()[0]
              df_new1 = df_new[df_new['user_id'].isin(cluster_1)]
```

```
    df_new1['rec_hotel_cluster'] = df_new1['hotel_cluster'].mode()[0]
    df_new2 = df_new[df_new['user_id'].isin(cluster_2)]
    df_new2['rec_hotel_cluster'] = df_new2['hotel_cluster'].mode()[0]
    frames = [train,df_old, df_new0, df_new1,df_new2]
    result = pd.concat(frames)
    return result
```

In [152]:
```
df_new = test[~test['user_id'].isin(train_user)]
df_new0 = df_new[df_new['user_id'].isin(cluster_0)]
```

In [153]:
```
df_new0
```

Out[153]:
```
                    date_time  site_name  posa_continent  \
15467234 2014-06-07 07:29:22          2               3
19922510 2014-10-04 05:43:36         34               3
11981320 2014-10-22 18:50:19         10               0
7854674  2014-11-18 11:37:10          2               3
35772501 2014-08-28 18:47:32          2               3
...                       ...        ...             ...
903321   2014-11-19 18:00:38          2               3
14840496 2014-09-22 18:58:08          2               3
18889807 2014-12-28 13:02:59          2               3
20923988 2013-04-28 20:58:26         34               3
446714   2014-09-16 08:53:21          2               3

          user_location_country  user_location_region  user_location_city  \
15467234                     80                    38               52486
19922510                    205                   354               21728
11981320                    182                   376               14405
7854674                      66                   351               52368
35772501                     66                   442               23258
...                         ...                   ...                 ...
903321                       66                   337                3591
14840496                     66                   260               22666
18889807                     66                   318               51298
20923988                    205                   155               14703
446714                       66                   331                2639

          orig_destination_distance  user_id  is_mobile  is_package  ...  \
15467234                1877.175685   299996          0           0  ...
19922510                 298.178700      420          0           0  ...
11981320                4106.045000   380410          0           0  ...
7854674                  881.810500   163201          0           0  ...
35772501                1233.609700   177098          0           0  ...
...                             ...      ...        ...         ...  ...
903321                    62.753600   590085          1           0  ...
14840496                 103.351700  1015569          0           0  ...
18889807                 249.053800   751293          1           0  ...
```

```
20923988            2427.830200   544688          0            0  ...
446714               144.724700   367278          0            0  ...

          srch_rm_cnt srch_destination_id srch_destination_type_id  \
15467234            1                8230                        1
19922510            1                 108                        1
11981320            1                8260                        1
7854674             1               13436                        4
35772501            1               27140                        6
...               ...                 ...                      ...
903321              1               11569                        1
14840496            2               12014                        1
18889807            1                8230                        1
20923988            1                8267                        1
446714              2                8274                        1

          is_booking  cnt  hotel_continent  hotel_country  hotel_market  \
15467234           0    1                2             50           637
19922510           0    1                2             50           829
11981320           0    1                2             50           701
7854674            0    1                2             50           682
35772501           0    1                2             50           637
...              ...  ...              ...            ...           ...
903321             0    3                2             50           623
14840496           0    2                2             50           644
18889807           0    1                2             50           637
20923988           0    1                2             50           675
446714             0    1                2             50           684

          hotel_cluster             earlist
15467234             69 2014-06-07 07:29:22
19922510             13 2014-10-04 05:43:36
11981320             18 2014-10-22 18:50:19
7854674              31 2014-11-18 11:37:10
35772501             91 2014-08-28 18:47:32
...                 ...                 ...
903321               79 2014-11-19 18:00:38
14840496             55 2014-09-22 18:58:08
18889807             91 2014-12-28 13:02:59
20923988             98 2013-04-28 20:58:26
446714               28 2014-09-16 08:53:21

[19895 rows x 25 columns]
```

In [166]:
```
data = rec_cluster()
data = data.drop(['earlist'], axis=1)
```

In [167]:
```
data.to_csv ('data.csv')
```

In [ ]:
```
print('duration: ',pd.datetime.now() - start_time)
```