Quickr - Assignment

January 30, 2019

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        from tqdm import tqdm
        import warnings
        import pandas_profiling
```

```
In [19]: tqdm.pandas()
         plt.style.use('ggplot')
         %config InlineBackend.figure_format = 'retina'
         sns.set(rc={'figure.figsize':(25,12)})
         warnings.filterwarnings('ignore', category=RuntimeWarning)
         warnings.filterwarnings('ignore', category=UserWarning)
         warnings.filterwarnings('ignore', category=DeprecationWarning)
```

```
In [3]: data = pd.read_csv('assignment2_data.csv')
```

```
In [4]: data.shape
```

```
Out[4]: (14471, 45)
```

```
In [5]: data.columns
```

```
Out[5]: Index(['user_id', 'dvJun13', 'S30d_events', 'S30d_totalsessiontime',
               'S30d_sessioncount', 'S30d_propertiesviewed',
               'S30d_distinctpropertiesviewed', 'S30d_Saleviews', 'S30d_Rentviews',
               'S30d_Otherviews', 'S30d_distinctlocalitiesviewed',
               'S30d_distinctcitiesviewed', 'S30d_pricevariance', 'SMax_property',
               'SMax_city_name', 'SMax_subcategory_name', 'SMax_source',
               'SMax_utm_traffic_source', 'SMax_price', 'SMax_Area_Sq_Feet',
               'R30d_responses', 'R30d_propertiesresponded',
               'R30d_distinctpropertiesresponded', 'R30d_Saleresponses',
               'R30d_Rentresponses', 'R30d_Otherresponses',
               'R30d_distinctlocalitiesresponded', 'R30d_distinctcitiesresponded',
               'R30d_pricevariance', 'R30d_Projectresponses', 'RMax_property',
               'RMax_city_name', 'RMax_subcategory_name', 'RMax_source',
               'RMax_utm_traffic_source', 'RMax_price', 'RMax_Area_Sq_Feet',
```

```
       'S30d_seriousness', 'S30d_Saleseriousness', 'S30d_avgsessiontime',
       'S30d_distinctpropertiesviewedRatio', 'SMax_priceperSqft',
       'R30d_Saleseriousness', 'R30d_distinctpropertiesrespondedRatio',
       'RMax_priceperSqft'],
      dtype='object')
```

In [6]:
```
data.head()
```

Out[6]:

	user_id	dvJun13	S30d_events	S30d_totalsessiontime	S30d_sessioncount	\
0	156453324	0	6	4.457883	2	
1	154292288	0	2	1.409117	1	
2	140097221	0	6	4.405150	1	
3	128872598	0	2	0.000000	2	
4	150399061	0	1	0.000000	1	

	S30d_propertiesviewed	S30d_distinctpropertiesviewed	S30d_Saleviews	\
0	6	2	6	
1	2	1	2	
2	6	2	6	
3	2	1	2	
4	0	0	0	

	S30d_Rentviews	S30d_Otherviews	...	RMax_price	\
0	0	0	...	1.0	
1	0	0	...	NaN	
2	0	0	...	1.0	
3	0	0	...	NaN	
4	0	1	...	4.0	

	RMax_Area_Sq_Feet	S30d_seriousness	S30d_Saleseriousness	\
0	2.0	0.123830	1.0	
1	NaN	0.352279	1.0	
2	3.0	0.122365	1.0	
3	NaN	0.000000	1.0	
4	4.0	NaN	NaN	

	S30d_avgsessiontime	S30d_distinctpropertiesviewedRatio	SMax_priceperSqft	\
0	2.228942	0.333333	0.333333	
1	1.409117	0.500000	0.250000	
2	4.405150	0.333333	1.000000	
3	0.000000	0.500000	0.500000	
4	0.000000	NaN	1.000000	

	R30d_Saleseriousness	R30d_distinctpropertiesrespondedRatio	\
0	1.0	1.0	
1	NaN	NaN	
2	1.0	1.0	
3	NaN	NaN	

```
4                   NaN                                          NaN

   RMax_priceperSqft
0           0.500000
1                NaN
2           0.333333
3                NaN
4           1.000000

[5 rows x 45 columns]
```

```
In [11]: pandas_profiling.ProfileReport(data)
```

```
Out[11]: <pandas_profiling.ProfileReport at 0x7ff25c192438>
```

```
In [9]: corr = np.round(data.corr(), 2)
```

```
In [10]: corr.loc['dvJun13']
```

```
Out[10]: user_id                                0.02
         dvJun13                                1.00
         S30d_events                           -0.05
         S30d_totalsessiontime                  0.00
         S30d_sessioncount                     -0.01
         S30d_propertiesviewed                 -0.05
         S30d_distinctpropertiesviewed         -0.05
         S30d_Saleviews                        -0.05
         S30d_Rentviews                        -0.05
         S30d_Otherviews                       -0.05
         S30d_distinctlocalitiesviewed         -0.05
         S30d_distinctcitiesviewed             -0.04
         S30d_pricevariance                    -0.00
         SMax_price                            -0.04
         SMax_Area_Sq_Feet                     -0.04
         R30d_responses                        -0.06
         R30d_propertiesresponded              -0.06
         R30d_distinctpropertiesresponded      -0.06
         R30d_Saleresponses                    -0.06
         R30d_Rentresponses                    -0.06
         R30d_Otherresponses                   -0.06
         R30d_distinctlocalitiesresponded      -0.06
         R30d_distinctcitiesresponded          -0.08
         R30d_pricevariance                    -0.03
         R30d_Projectresponses                  0.01
         RMax_price                            -0.08
         RMax_Area_Sq_Feet                     -0.06
         S30d_seriousness                       0.00
         S30d_Saleseriousness                   0.06
         S30d_avgsessiontime                    0.01
```

```
S30d_distinctpropertiesviewedRatio      -0.06
SMax_priceperSqft                       -0.00
R30d_Saleseriousness                     0.07
R30d_distinctpropertiesrespondedRatio    0.06
RMax_priceperSqft                       -0.04
Name: dvJun13, dtype: float64
```

Conclusion: Most of the columns are not very hightly corelelated and hence we need to use almost all the columns in our analysis.

```
In [11]: numerical_cols = ['S30d_events', 'S30d_totalsessiontime',
                'S30d_sessioncount', 'S30d_propertiesviewed',
                'S30d_distinctpropertiesviewed', 'S30d_Saleviews', 'S30d_Rentviews',
                'S30d_Otherviews', 'S30d_distinctlocalitiesviewed',
                'S30d_distinctcitiesviewed', 'S30d_pricevariance',
                'SMax_price', 'SMax_Area_Sq_Feet',
                'R30d_responses', 'R30d_propertiesresponded',
                'R30d_distinctpropertiesresponded', 'R30d_Saleresponses',
                'R30d_Rentresponses', 'R30d_Otherresponses',
                'R30d_distinctlocalitiesresponded', 'R30d_distinctcitiesresponded',
                'R30d_pricevariance', 'R30d_Projectresponses',
                 'RMax_price', 'RMax_Area_Sq_Feet',
                'S30d_seriousness', 'S30d_Saleseriousness', 'S30d_avgsessiontime',
                'S30d_distinctpropertiesviewedRatio', 'SMax_priceperSqft',
                'R30d_Saleseriousness', 'R30d_distinctpropertiesrespondedRatio',
                'RMax_priceperSqft']
```

```
In [12]: for col in numerical_cols:
             print('----------------------')
             print('Column: ', col)
             print('----------------------')
             print('\nColumn Description:\n\n{}'.format(data[col].describe()))
             print('----------------------')
             print(data[col].quantile(np.arange(0, 1.1, 0.1)))
             print(data[col].quantile(np.arange(0.9, 1.01, 0.01)))
             print(data[col].quantile(np.arange(0.99, 1.001, 0.001)))
             print('_' * 100)
```

```
----------------------
Column:  S30d_events
----------------------

Column Description:

count    14471.000000
mean       142.818465
std       2277.147797
min          0.000000
25%          2.000000
```

```
50%          6.000000
75%         15.000000
max      46332.000000
Name: S30d_events, dtype: float64
---------------------
0.0        0.0
0.1        0.0
0.2        1.0
0.3        3.0
0.4        4.0
0.5        6.0
0.6        9.0
0.7       12.0
0.8       18.0
0.9       31.0
1.0    46332.0
Name: S30d_events, dtype: float64
0.90       31.0
0.91       34.0
0.92       37.0
0.93       40.0
0.94       44.0
0.95       51.0
0.96       58.0
0.97       68.0
0.98       88.0
0.99      151.6
1.00    46332.0
Name: S30d_events, dtype: float64
0.990      151.60
0.991      170.77
0.992      191.96
0.993      231.42
0.994      276.26
0.995      404.40
0.996      714.36
0.997    24442.83
0.998    41151.46
0.999    43252.71
1.000    46332.00
Name: S30d_events, dtype: float64
--------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_totalsessiontime
---------------------

Column Description:
```

```
count    14471.000000
mean        19.595616
std         76.592819
min          0.000000
25%          1.244375
50%          6.179367
75%         19.724883
max       4971.235900
Name: S30d_totalsessiontime, dtype: float64
---------------------
0.0       0.000000
0.1       0.000000
0.2       0.562950
0.3       1.952133
0.4       3.758883
0.5       6.179367
0.6       9.822750
0.7      15.536317
0.8      25.241050
0.9      44.541400
1.0    4971.235900
Name: S30d_totalsessiontime, dtype: float64
0.90      44.541400
0.91      47.833993
0.92      51.994710
0.93      56.606198
0.94      62.937607
0.95      69.410725
0.96      80.070900
0.97      96.725350
0.98     124.943460
0.99     187.002000
1.00    4971.235900
Name: S30d_totalsessiontime, dtype: float64
0.990     187.002000
0.991     198.350572
0.992     211.140728
0.993     222.949655
0.994     234.421635
0.995     247.484309
0.996     276.780654
0.997     325.065079
0.998     423.964302
0.999     644.647459
1.000    4971.235900
Name: S30d_totalsessiontime, dtype: float64
---------------------------------------------------------------------------------------------------
---------------------
```

```
Column:  S30d_sessioncount
----------------------

Column Description:

count    14471.000000
mean         2.716606
std          4.072365
min          0.000000
25%          1.000000
50%          1.000000
75%          3.000000
max        122.000000
Name: S30d_sessioncount, dtype: float64
----------------------
0.0      0.0
0.1      1.0
0.2      1.0
0.3      1.0
0.4      1.0
0.5      1.0
0.6      2.0
0.7      2.0
0.8      3.0
0.9      6.0
1.0    122.0
Name: S30d_sessioncount, dtype: float64
0.90      6.0
0.91      6.0
0.92      6.0
0.93      7.0
0.94      8.0
0.95      9.0
0.96      9.0
0.97     11.0
0.98     13.0
0.99     19.0
1.00    122.0
Name: S30d_sessioncount, dtype: float64
0.990     19.00
0.991     20.00
0.992     20.24
0.993     22.00
0.994     23.00
0.995     25.00
0.996     27.00
0.997     30.59
0.998     36.00
```

```
0.999     48.00
1.000    122.00
Name: S30d_sessioncount, dtype: float64
--------------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_propertiesviewed
---------------------

Column Description:

count    14471.00000
mean       130.47343
std       2036.55981
min          0.00000
25%          3.00000
50%          7.00000
75%         16.00000
max      40514.00000
Name: S30d_propertiesviewed, dtype: float64
---------------------
0.0        0.0
0.1        1.0
0.2        2.0
0.3        3.0
0.4        5.0
0.5        7.0
0.6        9.0
0.7       13.0
0.8       20.0
0.9       34.0
1.0    40514.0
Name: S30d_propertiesviewed, dtype: float64
0.90       34.0
0.91       37.0
0.92       40.0
0.93       43.0
0.94       48.0
0.95       54.0
0.96       61.0
0.97       73.0
0.98       94.0
0.99      161.0
1.00    40514.0
Name: S30d_propertiesviewed, dtype: float64
0.990      161.00
0.991      180.00
0.992      212.96
0.993      234.71
```

```
0.994      288.54
0.995      410.00
0.996      714.36
0.997    20646.81
0.998    36155.74
0.999    38727.49
1.000    40514.00
Name: S30d_propertiesviewed, dtype: float64
--------------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_distinctpropertiesviewed
---------------------

Column Description:

count    14471.000000
mean        40.061571
std        588.690922
min          0.000000
25%          1.000000
50%          3.000000
75%          6.000000
max      12107.000000
Name: S30d_distinctpropertiesviewed, dtype: float64
---------------------
0.0        0.0
0.1        1.0
0.2        1.0
0.3        1.0
0.4        2.0
0.5        3.0
0.6        4.0
0.7        5.0
0.8        8.0
0.9       14.0
1.0    12107.0
Name: S30d_distinctpropertiesviewed, dtype: float64
0.90       14.0
0.91       15.0
0.92       17.0
0.93       19.0
0.94       21.0
0.95       23.0
0.96       27.0
0.97       32.0
0.98       43.6
0.99       75.3
1.00    12107.0
```

```
Name: S30d_distinctpropertiesviewed, dtype: float64
0.990       75.30
0.991       84.00
0.992       99.24
0.993      112.71
0.994      123.90
0.995      165.20
0.996      366.80
0.997     6788.15
0.998    10545.38
0.999    11078.23
1.000    12107.00
Name: S30d_distinctpropertiesviewed, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_Saleviews
---------------------

Column Description:

count    14471.000000
mean        73.345035
std       1066.596087
min          0.000000
25%          2.000000
50%          6.000000
75%         14.000000
max      21940.000000
Name: S30d_Saleviews, dtype: float64
---------------------
0.0        0.0
0.1        1.0
0.2        2.0
0.3        3.0
0.4        4.0
0.5        6.0
0.6        8.0
0.7       12.0
0.8       17.0
0.9       30.0
1.0    21940.0
Name: S30d_Saleviews, dtype: float64
0.90       30.0
0.91       32.0
0.92       35.0
0.93       38.0
0.94       42.0
0.95       47.0
```

```
0.96       54.0
0.97       63.0
0.98       81.6
0.99      136.3
1.00    21940.0
Name: S30d_Saleviews, dtype: float64
0.990      136.30
0.991      147.77
0.992      164.00
0.993      181.00
0.994      222.54
0.995      291.00
0.996      419.56
0.997    11092.86
0.998    18902.06
0.999    20048.77
1.000    21940.00
Name: S30d_Saleviews, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_Rentviews
---------------------

Column Description:

count    14471.000000
mean        52.918734
std        901.482736
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max      18579.000000
Name: S30d_Rentviews, dtype: float64
---------------------
0.0        0.0
0.1        0.0
0.2        0.0
0.3        0.0
0.4        0.0
0.5        0.0
0.6        0.0
0.7        0.0
0.8        0.0
0.9        3.0
1.0    18579.0
Name: S30d_Rentviews, dtype: float64
0.90        3.0
```

```
0.91        4.0
0.92        5.0
0.93        6.0
0.94        7.0
0.95        9.0
0.96       13.0
0.97       18.0
0.98       29.0
0.99       53.0
1.00    18579.0
Name: S30d_Rentviews, dtype: float64
0.990       53.00
0.991       56.00
0.992       62.48
0.993       76.71
0.994      102.36
0.995      130.00
0.996      201.12
0.997     8807.67
0.998    15822.26
0.999    17069.43
1.000    18579.00
Name: S30d_Rentviews, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_Otherviews
---------------------

Column Description:

count    14471.000000
mean        18.791514
std        315.038068
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max       7207.000000
Name: S30d_Otherviews, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       0.0
0.3       0.0
0.4       0.0
0.5       0.0
0.6       0.0
0.7       0.0
```

```
0.8       1.0
0.9       2.0
1.0    7207.0
Name: S30d_Otherviews, dtype: float64
0.90       2.0
0.91       2.0
0.92       2.0
0.93       3.0
0.94       3.0
0.95       4.0
0.96       5.0
0.97       7.0
0.98      11.0
0.99      29.0
1.00    7207.0
Name: S30d_Otherviews, dtype: float64
0.990      29.00
0.991      36.77
0.992      45.24
0.993      53.71
0.994      71.18
0.995     118.75
0.996     256.44
0.997    3453.76
0.998    5129.16
0.999    5949.38
1.000    7207.00
Name: S30d_Otherviews, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_distinctlocalitiesviewed
---------------------

Column Description:

count    14471.000000
mean        12.360445
std        153.395446
min          0.000000
25%          1.000000
50%          2.000000
75%          4.000000
max       3023.000000
Name: S30d_distinctlocalitiesviewed, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       1.0
```

```
0.3       1.0
0.4       1.0
0.5       2.0
0.6       2.0
0.7       3.0
0.8       5.0
0.9       8.0
1.0    3023.0
Name: S30d_distinctlocalitiesviewed, dtype: float64
0.90       8.0
0.91       8.0
0.92       9.0
0.93      10.0
0.94      11.0
0.95      12.0
0.96      14.0
0.97      16.0
0.98      20.0
0.99      31.0
1.00    3023.0
Name: S30d_distinctlocalitiesviewed, dtype: float64
0.990      31.00
0.991      33.00
0.992      37.00
0.993      44.00
0.994      51.54
0.995      74.60
0.996     178.00
0.997    2350.70
0.998    2649.30
0.999    2773.53
1.000    3023.00
Name: S30d_distinctlocalitiesviewed, dtype: float64
--------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_distinctcitiesviewed
---------------------

Column Description:

count    14471.000000
mean         1.162048
std          0.591053
min          1.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          8.000000
```

```
Name: S30d_distinctcitiesviewed, dtype: float64
--------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    1.0
0.5    1.0
0.6    1.0
0.7    1.0
0.8    1.0
0.9    2.0
1.0    8.0
Name: S30d_distinctcitiesviewed, dtype: float64
0.90    2.0
0.91    2.0
0.92    2.0
0.93    2.0
0.94    2.0
0.95    2.0
0.96    2.0
0.97    2.0
0.98    2.0
0.99    2.0
1.00    8.0
Name: S30d_distinctcitiesviewed, dtype: float64
0.990    2.00
0.991    3.00
0.992    3.00
0.993    3.00
0.994    4.18
0.995    6.00
0.996    7.00
0.997    8.00
0.998    8.00
0.999    8.00
1.000    8.00
Name: S30d_distinctcitiesviewed, dtype: float64
--------------------------------------------------------------------------------------------------------
--------------------
Column:  S30d_pricevariance
--------------------

Column Description:

count    12819.000000
mean         0.576814
std          0.843340
```

```
min          0.000000
25%          0.000000
50%          0.045455
75%          1.000000
max          4.500000
Name: S30d_pricevariance, dtype: float64
----------------------
0.0    0.000000
0.1    0.000000
0.2    0.000000
0.3    0.000000
0.4    0.000000
0.5    0.045455
0.6    0.285714
0.7    0.717650
0.8    1.285714
0.9    1.978022
1.0    4.500000
Name: S30d_pricevariance, dtype: float64
0.90    1.978022
0.91    2.064188
0.92    2.142857
0.93    2.225255
0.94    2.250000
0.95    2.286175
0.96    2.376070
0.97    2.410714
0.98    2.700000
0.99    3.000000
1.00    4.500000
Name: S30d_pricevariance, dtype: float64
0.990    3.0
0.991    3.0
0.992    3.0
0.993    3.0
0.994    3.0
0.995    3.0
0.996    4.5
0.997    4.5
0.998    4.5
0.999    4.5
1.000    4.5
Name: S30d_pricevariance, dtype: float64
----------------------------------------------------------------------------------------------------
----------------------
Column:  SMax_price
----------------------
```

```
Column Description:

count    14457.000000
mean         1.703604
std          1.156744
min          1.000000
25%          1.000000
50%          1.000000
75%          2.000000
max          4.000000
Name: SMax_price, dtype: float64
----------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    1.0
0.5    1.0
0.6    1.0
0.7    2.0
0.8    3.0
0.9    4.0
1.0    4.0
Name: SMax_price, dtype: float64
0.90    4.0
0.91    4.0
0.92    4.0
0.93    4.0
0.94    4.0
0.95    4.0
0.96    4.0
0.97    4.0
0.98    4.0
0.99    4.0
1.00    4.0
Name: SMax_price, dtype: float64
0.990    4.0
0.991    4.0
0.992    4.0
0.993    4.0
0.994    4.0
0.995    4.0
0.996    4.0
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: SMax_price, dtype: float64
```

```
--------------------------------------------------------------------------------------------------------
---------------------
Column:  SMax_Area_Sq_Feet
---------------------

Column Description:

count    14399.000000
mean         2.587332
std          1.271623
min          1.000000
25%          1.000000
50%          3.000000
75%          4.000000
max          4.000000
Name: SMax_Area_Sq_Feet, dtype: float64
---------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    2.0
0.5    3.0
0.6    3.0
0.7    4.0
0.8    4.0
0.9    4.0
1.0    4.0
Name: SMax_Area_Sq_Feet, dtype: float64
0.90    4.0
0.91    4.0
0.92    4.0
0.93    4.0
0.94    4.0
0.95    4.0
0.96    4.0
0.97    4.0
0.98    4.0
0.99    4.0
1.00    4.0
Name: SMax_Area_Sq_Feet, dtype: float64
0.990    4.0
0.991    4.0
0.992    4.0
0.993    4.0
0.994    4.0
0.995    4.0
0.996    4.0
```

```
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: SMax_Area_Sq_Feet, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_responses
---------------------

Column Description:

count    10573.000000
mean        82.739620
std       1184.045021
min          1.000000
25%          1.000000
50%          2.000000
75%          5.000000
max      21184.000000
Name: R30d_responses, dtype: float64
---------------------
0.0        1.0
0.1        1.0
0.2        1.0
0.3        1.0
0.4        2.0
0.5        2.0
0.6        3.0
0.7        4.0
0.8        6.0
0.9       10.0
1.0    21184.0
Name: R30d_responses, dtype: float64
0.90       10.0
0.91       10.0
0.92       12.0
0.93       12.0
0.94       14.0
0.95       16.0
0.96       19.0
0.97       23.0
0.98       28.0
0.99       48.0
1.00    21184.0
Name: R30d_responses, dtype: float64
0.990       48.000
0.991       54.000
```

```
0.992       59.848
0.993       66.996
0.994       91.112
0.995      193.560
0.996    11330.152
0.997    17870.384
0.998    18170.816
0.999    19357.100
1.000    21184.000
Name: R30d_responses, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_propertiesresponded
---------------------

Column Description:

count    10573.000000
mean        63.702923
std        901.602558
min          0.000000
25%          1.000000
50%          2.000000
75%          4.000000
max      15366.000000
Name: R30d_propertiesresponded, dtype: float64
---------------------
0.0        0.0
0.1        0.0
0.2        1.0
0.3        1.0
0.4        1.0
0.5        2.0
0.6        2.0
0.7        4.0
0.8        5.0
0.9        9.0
1.0    15366.0
Name: R30d_propertiesresponded, dtype: float64
0.90        9.0
0.91       10.0
0.92       10.0
0.93       12.0
0.94       13.0
0.95       14.0
0.96       17.0
0.97       21.0
0.98       26.0
```

```
0.99       46.0
1.00    15366.0
Name: R30d_propertiesresponded, dtype: float64
0.990       46.000
0.991       51.852
0.992       59.000
0.993       63.992
0.994       90.680
0.995      185.260
0.996     8433.488
0.997    13573.620
0.998    14227.712
0.999    14545.992
1.000    15366.000
Name: R30d_propertiesresponded, dtype: float64
----------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_distinctpropertiesresponded
---------------------

Column Description:

count    10573.000000
mean        31.190391
std        438.066975
min          0.000000
25%          1.000000
50%          1.000000
75%          2.000000
max       7875.000000
Name: R30d_distinctpropertiesresponded, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       1.0
0.3       1.0
0.4       1.0
0.5       1.0
0.6       1.0
0.7       2.0
0.8       3.0
0.9       4.0
1.0    7875.0
Name: R30d_distinctpropertiesresponded, dtype: float64
0.90       4.00
0.91       5.00
0.92       5.00
0.93       5.00
```

```
0.94       6.00
0.95       7.00
0.96       8.00
0.97       9.84
0.98      12.00
0.99      18.00
1.00    7875.00
Name: R30d_distinctpropertiesresponded, dtype: float64
0.990      18.000
0.991      19.852
0.992      21.000
0.993      23.996
0.994      40.000
0.995      81.280
0.996    4322.536
0.997    6120.284
0.998    6640.272
0.999    7378.388
1.000    7875.000
Name: R30d_distinctpropertiesresponded, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_Saleresponses
---------------------

Column Description:

count    10573.000000
mean        34.071125
std        456.506434
min          0.000000
25%          1.000000
50%          2.000000
75%          4.000000
max       8129.000000
Name: R30d_Saleresponses, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       1.0
0.3       1.0
0.4       1.0
0.5       2.0
0.6       2.0
0.7       3.0
0.8       5.0
0.9       8.0
1.0    8129.0
```

```
Name: R30d_Saleresponses, dtype: float64
0.90       8.00
0.91       9.00
0.92      10.00
0.93      11.00
0.94      12.00
0.95      14.00
0.96      16.00
0.97      20.00
0.98      25.00
0.99      44.56
1.00    8129.00
Name: R30d_Saleresponses, dtype: float64
0.990      44.560
0.991      49.852
0.992      57.424
0.993      62.000
0.994      73.136
0.995     133.700
0.996    4263.112
0.997    6814.684
0.998    7033.712
0.999    7306.840
1.000    8129.000
Name: R30d_Saleresponses, dtype: float64
--------------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_Rentresponses
---------------------

Column Description:

count    10573.000000
mean        26.277783
std        392.560857
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max       6757.000000
Name: R30d_Rentresponses, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       0.0
0.3       0.0
0.4       0.0
0.5       0.0
```

```
0.6       0.0
0.7       0.0
0.8       0.0
0.9       0.0
1.0    6757.0
Name: R30d_Rentresponses, dtype: float64
0.90       0.0
0.91       0.0
0.92       0.0
0.93       0.0
0.94       1.0
0.95       1.0
0.96       2.0
0.97       3.0
0.98       5.0
0.99      10.0
1.00    6757.0
Name: R30d_Rentresponses, dtype: float64
0.990      10.000
0.991      13.000
0.992      14.000
0.993      16.996
0.994      21.000
0.995      38.280
0.996    3649.104
0.997    5806.372
0.998    6199.256
0.999    6472.424
1.000    6757.000
Name: R30d_Rentresponses, dtype: float64
--------------------------------------------------------------------------------------------------------
--------------------
Column:  R30d_Otherresponses
--------------------

Column Description:

count    10573.000000
mean        22.390712
std        340.225006
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max       6727.000000
Name: R30d_Otherresponses, dtype: float64
--------------------
0.0       0.0
```

```
0.1       0.0
0.2       0.0
0.3       0.0
0.4       0.0
0.5       0.0
0.6       0.0
0.7       0.0
0.8       0.0
0.9       1.0
1.0    6727.0
Name: R30d_Otherresponses, dtype: float64
0.90       1.0
0.91       1.0
0.92       1.0
0.93       2.0
0.94       2.0
0.95       2.0
0.96       3.0
0.97       3.0
0.98       5.0
0.99      10.0
1.00    6727.0
Name: R30d_Otherresponses, dtype: float64
0.990      10.000
0.991      10.852
0.992      13.000
0.993      15.000
0.994      20.568
0.995      34.560
0.996    3305.624
0.997    4550.708
0.998    4971.000
0.999    6131.676
1.000    6727.000
Name: R30d_Otherresponses, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_distinctlocalitiesresponded
---------------------

Column Description:

count    10573.000000
mean        11.890381
std        156.543469
min          0.000000
25%          1.000000
50%          1.000000
```

```
75%          2.000000
max       2724.000000
Name: R30d_distinctlocalitiesresponded, dtype: float64
---------------------
0.0       0.0
0.1       0.0
0.2       0.0
0.3       1.0
0.4       1.0
0.5       1.0
0.6       1.0
0.7       1.0
0.8       2.0
0.9       3.0
1.0    2724.0
Name: R30d_distinctlocalitiesresponded, dtype: float64
0.90       3.0
0.91       3.0
0.92       3.0
0.93       3.0
0.94       4.0
0.95       4.0
0.96       5.0
0.97       6.0
0.98       7.0
0.99      11.0
1.00    2724.0
Name: R30d_distinctlocalitiesresponded, dtype: float64
0.990      11.000
0.991      11.000
0.992      12.000
0.993      14.000
0.994      18.000
0.995      44.000
0.996    2043.808
0.997    2272.568
0.998    2349.424
0.999    2522.564
1.000    2724.000
Name: R30d_distinctlocalitiesresponded, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_distinctcitiesresponded
---------------------

Column Description:

count    10573.000000
```

```
mean         1.145181
std          0.563265
min          1.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          8.000000
Name: R30d_distinctcitiesresponded, dtype: float64
---------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    1.0
0.5    1.0
0.6    1.0
0.7    1.0
0.8    1.0
0.9    2.0
1.0    8.0
Name: R30d_distinctcitiesresponded, dtype: float64
0.90    2.0
0.91    2.0
0.92    2.0
0.93    2.0
0.94    2.0
0.95    2.0
0.96    2.0
0.97    2.0
0.98    2.0
0.99    2.0
1.00    8.0
Name: R30d_distinctcitiesresponded, dtype: float64
0.990    2.0
0.991    2.0
0.992    2.0
0.993    2.0
0.994    2.0
0.995    3.0
0.996    8.0
0.997    8.0
0.998    8.0
0.999    8.0
1.000    8.0
Name: R30d_distinctcitiesresponded, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_pricevariance
```

```
---------------------

Column Description:

count    6917.000000
mean        0.889835
std         1.354163
min         0.000000
25%         0.000000
50%         0.000000
75%         1.800000
max         4.500000
Name: R30d_pricevariance, dtype: float64
---------------------
0.0    0.000000
0.1    0.000000
0.2    0.000000
0.3    0.000000
0.4    0.000000
0.5    0.000000
0.6    0.216623
0.7    1.203848
0.8    2.250000
0.9    3.000000
1.0    4.500000
Name: R30d_pricevariance, dtype: float64
0.90    3.0
0.91    3.0
0.92    3.0
0.93    3.0
0.94    4.5
0.95    4.5
0.96    4.5
0.97    4.5
0.98    4.5
0.99    4.5
1.00    4.5
Name: R30d_pricevariance, dtype: float64
0.990    4.5
0.991    4.5
0.992    4.5
0.993    4.5
0.994    4.5
0.995    4.5
0.996    4.5
0.997    4.5
0.998    4.5
0.999    4.5
```

```
1.000    4.5
Name: R30d_pricevariance, dtype: float64
--------------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_Projectresponses
---------------------

Column Description:

count    10573.000000
mean         0.497967
std          1.690742
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max         40.000000
Name: R30d_Projectresponses, dtype: float64
---------------------
0.0     0.0
0.1     0.0
0.2     0.0
0.3     0.0
0.4     0.0
0.5     0.0
0.6     0.0
0.7     0.0
0.8     1.0
0.9     2.0
1.0    40.0
Name: R30d_Projectresponses, dtype: float64
0.90     2.0
0.91     2.0
0.92     2.0
0.93     2.0
0.94     2.0
0.95     2.0
0.96     3.0
0.97     4.0
0.98     4.0
0.99     7.0
1.00    40.0
Name: R30d_Projectresponses, dtype: float64
0.990     7.000
0.991     8.000
0.992     8.000
0.993     9.000
0.994     9.568
```

```
0.995    11.000
0.996    11.712
0.997    13.000
0.998    15.856
0.999    20.428
1.000    40.000
Name: R30d_Projectresponses, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  RMax_price
---------------------

Column Description:

count    10558.000000
mean         2.140841
std          1.377211
min          1.000000
25%          1.000000
50%          1.000000
75%          4.000000
max          4.000000
Name: RMax_price, dtype: float64
---------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    1.0
0.5    1.0
0.6    2.0
0.7    4.0
0.8    4.0
0.9    4.0
1.0    4.0
Name: RMax_price, dtype: float64
0.90    4.0
0.91    4.0
0.92    4.0
0.93    4.0
0.94    4.0
0.95    4.0
0.96    4.0
0.97    4.0
0.98    4.0
0.99    4.0
1.00    4.0
Name: RMax_price, dtype: float64
```

```
0.990    4.0
0.991    4.0
0.992    4.0
0.993    4.0
0.994    4.0
0.995    4.0
0.996    4.0
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: RMax_price, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  RMax_Area_Sq_Feet
---------------------

Column Description:

count    10487.000000
mean         2.847907
std          1.265469
min          1.000000
25%          1.000000
50%          3.000000
75%          4.000000
max          4.000000
Name: RMax_Area_Sq_Feet, dtype: float64
---------------------
0.0    1.0
0.1    1.0
0.2    1.0
0.3    2.0
0.4    3.0
0.5    3.0
0.6    4.0
0.7    4.0
0.8    4.0
0.9    4.0
1.0    4.0
Name: RMax_Area_Sq_Feet, dtype: float64
0.90    4.0
0.91    4.0
0.92    4.0
0.93    4.0
0.94    4.0
0.95    4.0
0.96    4.0
```

```
0.97    4.0
0.98    4.0
0.99    4.0
1.00    4.0
Name: RMax_Area_Sq_Feet, dtype: float64
0.990    4.0
0.991    4.0
0.992    4.0
0.993    4.0
0.994    4.0
0.995    4.0
0.996    4.0
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: RMax_Area_Sq_Feet, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_seriousness
---------------------

Column Description:

count    13317.000000
mean         0.264490
std          1.218278
min          0.000000
25%          0.020561
50%          0.065745
75%          0.177379
max         52.742167
Name: S30d_seriousness, dtype: float64
---------------------
0.0     0.000000
0.1     0.000000
0.2     0.013202
0.3     0.027825
0.4     0.044691
0.5     0.065745
0.6     0.095211
0.7     0.141586
0.8     0.225752
0.9     0.442908
1.0    52.742167
Name: S30d_seriousness, dtype: float64
0.90     0.442908
0.91     0.491330
```

```
0.92     0.546928
0.93     0.620370
0.94     0.731478
0.95     0.864908
0.96     1.066030
0.97     1.359171
0.98     1.981909
0.99     3.640040
1.00    52.742167
Name: S30d_seriousness, dtype: float64
0.990     3.640040
0.991     4.102897
0.992     4.499020
0.993     4.877940
0.994     5.428696
0.995     5.915078
0.996     6.748484
0.997     8.350823
0.998    11.067466
0.999    16.155309
1.000    52.742167
Name: S30d_seriousness, dtype: float64
--------------------------------------------------------------------------------------------------------------
----------------------
Column:  S30d_Saleseriousness
----------------------

Column Description:

count    13832.000000
mean         0.905788
std          0.256658
min          0.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          1.000000
Name: S30d_Saleseriousness, dtype: float64
----------------------
0.0    0.000000
0.1    0.571429
0.2    1.000000
0.3    1.000000
0.4    1.000000
0.5    1.000000
0.6    1.000000
0.7    1.000000
0.8    1.000000
```

```
0.9    1.000000
1.0    1.000000
Name: S30d_Saleseriousness, dtype: float64
0.90    1.0
0.91    1.0
0.92    1.0
0.93    1.0
0.94    1.0
0.95    1.0
0.96    1.0
0.97    1.0
0.98    1.0
0.99    1.0
1.00    1.0
Name: S30d_Saleseriousness, dtype: float64
0.990    1.0
0.991    1.0
0.992    1.0
0.993    1.0
0.994    1.0
0.995    1.0
0.996    1.0
0.997    1.0
0.998    1.0
0.999    1.0
1.000    1.0
Name: S30d_Saleseriousness, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_avgsessiontime
---------------------

Column Description:

count    14451.000000
mean         6.801188
std          9.791506
min          0.000000
25%          0.843844
50%          3.540900
75%          8.835108
max        144.902473
Name: S30d_avgsessiontime, dtype: float64
---------------------
0.0      0.000000
0.1      0.000000
0.2      0.365433
0.3      1.266847
```

```
0.4      2.285700
0.5      3.540900
0.6      5.115557
0.7      7.333583
0.8     10.730050
0.9     17.229467
1.0    144.902473
Name: S30d_avgsessiontime, dtype: float64
0.90     17.229467
0.91     18.445625
0.92     19.885708
0.93     21.350608
0.94     22.923217
0.95     24.882700
0.96     27.561167
0.97     31.175017
0.98     35.878683
0.99     45.420325
1.00    144.902473
Name: S30d_avgsessiontime, dtype: float64
0.990     45.420325
0.991     47.184810
0.992     49.952213
0.993     52.243839
0.994     54.719192
0.995     56.262399
0.996     59.683226
0.997     65.282636
0.998     71.234151
0.999     87.897194
1.000    144.902473
Name: S30d_avgsessiontime, dtype: float64
----------------------------------------------------------------------------------------------------
---------------------
Column:  S30d_distinctpropertiesviewedRatio
---------------------

Column Description:

count    13317.000000
mean         0.530280
std          0.290794
min          0.003322
25%          0.291667
50%          0.500000
75%          0.750000
max          1.000000
Name: S30d_distinctpropertiesviewedRatio, dtype: float64
```

```
---------------------
0.0    0.003322
0.1    0.185185
0.2    0.250000
0.3    0.333333
0.4    0.400000
0.5    0.500000
0.6    0.545455
0.7    0.666667
0.8    0.875000
0.9    1.000000
1.0    1.000000
Name: S30d_distinctpropertiesviewedRatio, dtype: float64
0.90    1.0
0.91    1.0
0.92    1.0
0.93    1.0
0.94    1.0
0.95    1.0
0.96    1.0
0.97    1.0
0.98    1.0
0.99    1.0
1.00    1.0
Name: S30d_distinctpropertiesviewedRatio, dtype: float64
0.990    1.0
0.991    1.0
0.992    1.0
0.993    1.0
0.994    1.0
0.995    1.0
0.996    1.0
0.997    1.0
0.998    1.0
0.999    1.0
1.000    1.0
Name: S30d_distinctpropertiesviewedRatio, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  SMax_priceperSqft
---------------------

Column Description:

count    14386.000000
mean         0.757467
std          0.454493
min          0.250000
```

```
25%          0.333333
50%          1.000000
75%          1.000000
max          4.000000
Name: SMax_priceperSqft, dtype: float64
---------------------
0.0    0.250000
0.1    0.250000
0.2    0.333333
0.3    0.500000
0.4    0.500000
0.5    1.000000
0.6    1.000000
0.7    1.000000
0.8    1.000000
0.9    1.000000
1.0    4.000000
Name: SMax_priceperSqft, dtype: float64
0.90    1.0
0.91    1.0
0.92    1.0
0.93    1.0
0.94    1.0
0.95    1.0
0.96    1.0
0.97    2.0
0.98    2.0
0.99    2.0
1.00    4.0
Name: SMax_priceperSqft, dtype: float64
0.990    2.0
0.991    2.0
0.992    2.0
0.993    3.0
0.994    3.0
0.995    4.0
0.996    4.0
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: SMax_priceperSqft, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  R30d_Saleseriousness
---------------------

Column Description:
```

```
count    9777.000000
mean        0.946402
std         0.207014
min         0.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         1.000000
Name: R30d_Saleseriousness, dtype: float64
--------------------
0.0    0.0
0.1    1.0
0.2    1.0
0.3    1.0
0.4    1.0
0.5    1.0
0.6    1.0
0.7    1.0
0.8    1.0
0.9    1.0
1.0    1.0
Name: R30d_Saleseriousness, dtype: float64
0.90    1.0
0.91    1.0
0.92    1.0
0.93    1.0
0.94    1.0
0.95    1.0
0.96    1.0
0.97    1.0
0.98    1.0
0.99    1.0
1.00    1.0
Name: R30d_Saleseriousness, dtype: float64
0.990    1.0
0.991    1.0
0.992    1.0
0.993    1.0
0.994    1.0
0.995    1.0
0.996    1.0
0.997    1.0
0.998    1.0
0.999    1.0
1.000    1.0
Name: R30d_Saleseriousness, dtype: float64
--------------------------------------------------------------------------------
```

```
---------------------
Column:  R30d_distinctpropertiesrespondedRatio
---------------------

Column Description:

count    9058.000000
mean        0.718123
std         0.290313
min         0.010000
25%         0.500000
50%         0.750000
75%         1.000000
max         1.000000
Name: R30d_distinctpropertiesrespondedRatio, dtype: float64
---------------------
0.0    0.010000
0.1    0.333333
0.2    0.500000
0.3    0.500000
0.4    0.571429
0.5    0.750000
0.6    1.000000
0.7    1.000000
0.8    1.000000
0.9    1.000000
1.0    1.000000
Name: R30d_distinctpropertiesrespondedRatio, dtype: float64
0.90    1.0
0.91    1.0
0.92    1.0
0.93    1.0
0.94    1.0
0.95    1.0
0.96    1.0
0.97    1.0
0.98    1.0
0.99    1.0
1.00    1.0
Name: R30d_distinctpropertiesrespondedRatio, dtype: float64
0.990    1.0
0.991    1.0
0.992    1.0
0.993    1.0
0.994    1.0
0.995    1.0
0.996    1.0
0.997    1.0
```

```
0.998    1.0
0.999    1.0
1.000    1.0
Name: R30d_distinctpropertiesrespondedRatio, dtype: float64
--------------------------------------------------------------------------------------------------------
---------------------
Column:  RMax_priceperSqft
---------------------

Column Description:

count    10473.000000
mean         0.826204
std          0.514141
min          0.250000
25%          0.500000
50%          1.000000
75%          1.000000
max          4.000000
Name: RMax_priceperSqft, dtype: float64
---------------------
0.0    0.250000
0.1    0.250000
0.2    0.333333
0.3    0.500000
0.4    0.750000
0.5    1.000000
0.6    1.000000
0.7    1.000000
0.8    1.000000
0.9    1.000000
1.0    4.000000
Name: RMax_priceperSqft, dtype: float64
0.90    1.000000
0.91    1.000000
0.92    1.000000
0.93    1.000000
0.94    1.000000
0.95    1.000000
0.96    1.333333
0.97    2.000000
0.98    2.000000
0.99    4.000000
1.00    4.000000
Name: RMax_priceperSqft, dtype: float64
0.990    4.0
0.991    4.0
0.992    4.0
```

```
0.993    4.0
0.994    4.0
0.995    4.0
0.996    4.0
0.997    4.0
0.998    4.0
0.999    4.0
1.000    4.0
Name: RMax_priceperSqft, dtype: float64
```

0.0.1 Removing the outliers from the data

```
In [13]: threshold_values = [
             715, 424, 48, 410, 367, 419, 201, 257, 178, 8, 5, 4, 4,
             194, 186, 82, 134, 39, 35, 44, 8, 4.5, 40, 4, 4, 17, 1, 145,
             1, 4, 1, 1, 4
         ]
```

```
In [14]: for col, thresh in zip(numerical_cols, threshold_values):
             n = data.shape[0]
             print('Removing outliers from column: {}'.format(col))
             new_n = data[data[col] <= thresh].shape[0]
             if (n - new_n) / n * 100.0 <= 5.0:
                 data = data[data[col] <= thresh]
             print('Lost data points: {}({})'.format(n - data.shape[0], (n - data.shape[0]) / n
             print('_' * 100)
```

```
Removing outliers from column: S30d_events
Lost data points: 58(0.4008016032064128)
```

```
Removing outliers from column: S30d_totalsessiontime
Lost data points: 18(0.12488725456185389)
```

```
Removing outliers from column: S30d_sessioncount
Lost data points: 5(0.03473428273706148)
```

```
Removing outliers from column: S30d_propertiesviewed
Lost data points: 7(0.048644892286309936)
```

```
Removing outliers from column: S30d_distinctpropertiesviewed
Lost data points: 0(0.0)
```

```
Removing outliers from column: S30d_Saleviews
Lost data points: 0(0.0)
```

```
Removing outliers from column: S30d_Rentviews
```

```
Lost data points: 4(0.027810609747618713)
-------------------------------------------------------------------------------------------------
Removing outliers from column: S30d_Otherviews
Lost data points: 4(0.02781834619931845)
-------------------------------------------------------------------------------------------------
Removing outliers from column: S30d_distinctlocalitiesviewed
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: S30d_distinctcitiesviewed
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: S30d_pricevariance
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: SMax_price
Lost data points: 14(0.09739130434782609)
-------------------------------------------------------------------------------------------------
Removing outliers from column: SMax_Area_Sq_Feet
Lost data points: 71(0.49439454077014133)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_responses
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_propertiesresponded
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_distinctpropertiesresponded
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_Saleresponses
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_Rentresponses
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_Otherresponses
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_distinctlocalitiesresponded
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_distinctcitiesresponded
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_pricevariance
Lost data points: 0(0.0)
-------------------------------------------------------------------------------------------------
Removing outliers from column: R30d_Projectresponses
```

```
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: RMax_price
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: RMax_Area_Sq_Feet
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: S30d_seriousness
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: S30d_Saleseriousness
Lost data points: 636(4.450664800559832)
-----------------------------------------------------------------------------------------------
Removing outliers from column: S30d_avgsessiontime
Lost data points: 20(0.14647722279185588)
-----------------------------------------------------------------------------------------------
Removing outliers from column: S30d_distinctpropertiesviewedRatio
Lost data points: 609(4.466774240868418)
-----------------------------------------------------------------------------------------------
Removing outliers from column: SMax_priceperSqft
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: R30d_Saleseriousness
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: R30d_distinctpropertiesrespondedRatio
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
Removing outliers from column: RMax_priceperSqft
Lost data points: 0(0.0)
-----------------------------------------------------------------------------------------------
```

```
In [15]: data.shape

Out[15]: (13025, 45)
```

0.1 Filling the missing values (For numerical Features)

```
In [16]: data[numerical_cols].std()

Out[16]: S30d_events                          22.672839
         S30d_totalsessiontime                30.895706
         S30d_sessioncount                     3.384026
         S30d_propertiesviewed                23.598938
         S30d_distinctpropertiesviewed        11.599967
         S30d_Saleviews                       20.863635
         S30d_Rentviews                        8.182616
```

```
S30d_Otherviews                          6.823955
S30d_distinctlocalitiesviewed            5.609493
S30d_distinctcitiesviewed                0.413690
S30d_pricevariance                       0.853906
SMax_price                               0.975674
SMax_Area_Sq_Feet                        1.250832
R30d_responses                           7.635026
R30d_propertiesresponded                 7.377752
R30d_distinctpropertiesresponded         3.321296
R30d_Saleresponses                       7.151640
R30d_Rentresponses                       1.662826
R30d_Otherresponses                      1.447002
R30d_distinctlocalitiesresponded         2.012781
R30d_distinctcitiesresponded             0.322457
R30d_pricevariance                       1.384587
R30d_Projectresponses                    1.502514
RMax_price                               1.278973
RMax_Area_Sq_Feet                        1.270843
S30d_seriousness                         1.214875
S30d_Saleseriousness                     0.260821
S30d_avgsessiontime                      9.751364
S30d_distinctpropertiesviewedRatio       0.290044
SMax_priceperSqft                        0.468003
R30d_Saleseriousness                     0.209946
R30d_distinctpropertiesrespondedRatio    0.290365
RMax_priceperSqft                        0.542957
dtype: float64
```

```
In [17]: from sklearn.preprocessing import Imputer
```

```
In [20]: imp1 = Imputer(strategy='mean')
         imp2 = Imputer(strategy='most_frequent')
```

```
In [21]: from sklearn.model_selection import train_test_split
```

```
In [22]: from collections import Counter
```

```
In [23]: y = data.dvJun13.values
         X = data.drop('dvJun13', axis=1)
```

```
In [24]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state
```

```
In [25]: Counter(y_train)
```

```
Out[25]: Counter({0: 5856, 1: 3912})
```

```
In [26]: Counter(y_test)
```

```
Out[26]: Counter({0: 1962, 1: 1295})
```

```
In [27]: X_train_num = X_train[numerical_cols]
         X_test_num = X_test[numerical_cols]

In [28]: imp1.fit(X_train_num)
         X_train_num_v1 = imp1.transform(X_train_num)
         X_test_num_v1 = imp1.transform(X_test_num)

In [29]: imp2.fit(X_train_num)
         X_train_num_v2 = imp1.transform(X_train_num)
         X_test_num_v2 = imp1.transform(X_test_num)

In [30]: X_train_num_v1 = pd.DataFrame(X_train_num_v1, columns=numerical_cols)
         X_test_num_v1 = pd.DataFrame(X_test_num_v1, columns=numerical_cols)

         X_train_num_v2 = pd.DataFrame(X_train_num_v2, columns=numerical_cols)
         X_test_num_v2 = pd.DataFrame(X_test_num_v2, columns=numerical_cols)
```

0.2 Filling the missing values (For non-numerical Features)

```
In [31]: cat_cols = set(data.columns) - set(numerical_cols)

In [32]: cat_cols

Out[32]: {'RMax_city_name',
          'RMax_property',
          'RMax_source',
          'RMax_subcategory_name',
          'RMax_utm_traffic_source',
          'SMax_city_name',
          'SMax_property',
          'SMax_source',
          'SMax_subcategory_name',
          'SMax_utm_traffic_source',
          'dvJun13',
          'user_id'}

In [33]: cat_cols.remove('user_id')

In [34]: cat_cols.remove('dvJun13')

In [35]: X_train_cat = X_train[list(cat_cols)]
         X_test_cat = X_test[list(cat_cols)]

In [36]: mode_cat = X_train_cat.mode()

In [37]: for col in cat_cols:
             X_train_cat[col] = X_train_cat[col].fillna(mode_cat[col][0])
             X_test_cat[col] = X_test_cat[col].fillna(mode_cat[col][0])
```

```
/home/mayukhpay/test/lib/python3.5/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html

/home/mayukhpay/test/lib/python3.5/site-packages/ipykernel_launcher.py:3: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  This is separate from the ipykernel package so we can avoid doing imports until
```

```
In [38]: for col in cat_cols:
             print((col, len(X_train_cat[col].unique())))
```

```
('RMax_city_name', 8)
('SMax_property', 2)
('RMax_property', 2)
('RMax_subcategory_name', 8)
('SMax_subcategory_name', 8)
('SMax_city_name', 8)
('RMax_utm_traffic_source', 10)
('SMax_utm_traffic_source', 11)
('RMax_source', 5)
('SMax_source', 5)
```

0.2.1 OneHotEncoding for all categorical columns. (Except `SMax_utm_traffic_source` & `RMax_utm_traffic_source`)

```
In [39]: hot_encode = ['SMax_property', 'RMax_subcategory_name', 'RMax_city_name',
                       'RMax_property', 'SMax_city_name', 'SMax_subcategory_name',
                       'RMax_source', 'SMax_source']
```

```
In [71]: X_train_cat_dummy = pd.get_dummies(X_train_cat[hot_encode])
         X_test_cat_dummy = pd.get_dummies(X_test_cat[hot_encode])
```

0.2.2 Response coding for `SMax_utm_traffic_source` & `RMax_utm_traffic_source`

```
In [41]: from collections import defaultdict
```

```
In [42]: def get_response_coding_for(data, feature, alpha=1, classes=(0, 1, )):
             counts = data[feature].value_counts()
             feature_dict = defaultdict(list)
             for each in data[feature].unique():
                 for each_class in classes:
                     match_count = data[(data['dvJun13'] == each_class) &
```

```
                                    (data[feature] == each)].shape[0]
                    ## Calculate the probability with laplace smoothing
                    feature_dict[each].append((match_count + alpha * 10) / (counts[each] + alp
            return feature_dict
```

```
In [43]: df = X_train_cat[['SMax_utm_traffic_source', 'RMax_utm_traffic_source']]
         df['dvJun13'] = y_train
```

```
/home/mayukhpay/test/lib/python3.5/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
```

```
In [44]: %%time
         SMax_utm_traffic_source_resp_coding = get_response_coding_for(df, 'SMax_utm_traffic_so
         RMax_utm_traffic_source_resp_coding = get_response_coding_for(df, 'RMax_utm_traffic_so
```

```
CPU times: user 88 ms, sys: 0 ns, total: 88 ms
Wall time: 84.7 ms
```

```
In [45]: SMax_utm_traffic_source_resp_coding
```

```
Out[45]: defaultdict(list,
                     {'Alerts': [0.5857033051498847, 0.36049192928516527],
                      'CF': [0.5221445221445221, 0.3146853146853147],
                      'Direct': [0.3470319634703196, 0.3333333333333333],
                      'Facebook': [0.12087912087912088, 0.10989010989010989],
                      'Internal': [0.4874274661508704, 0.3771760154738878],
                      'Notification': [0.15, 0.15],
                      'Organic': [0.5432960893854749, 0.35893854748603354],
                      'OtherPaid': [0.5906702754338481, 0.39818500238815474],
                      'Referral': [0.49523809523809526, 0.3380952380952381],
                      'SEM': [0.36904761904761907, 0.3531746031746032],
                      'SMS': [0.5231481481481481, 0.3148148148148148]})
```

```
In [46]: RMax_utm_traffic_source_resp_coding
```

```
Out[46]: defaultdict(list,
                     {'Alerts': [0.5654761904761905, 0.36507936507936506],
                      'CF': [0.3357142857142857, 0.16428571428571428],
                      'Direct': [0.4280639431616341, 0.44760213143872113],
                      'Facebook': [0.12087912087912088, 0.10989010989010989],
                      'Internal': [0.5064377682403434, 0.45064377682403434],
                      'Organic': [0.619414154372461, 0.36561898652982683],
                      'OtherPaid': [0.6059001512859304, 0.3411497730711044],
```

```
                    'Referral': [0.450402144772118, 0.36193029490616624],
                    'SEM': [0.4264069264069264, 0.42207792207792205],
                    'SMS': [0.4937655860349127, 0.3316708229426434]})
```

```
In [47]: def featurize_SMax_utm_traffic(SMax_utm_traffic):
             try:
                 return SMax_utm_traffic_source_resp_coding[SMax_utm_traffic]
             except KeyError:
                 return [1/2] * 2
```

```
In [48]: def featurize_RMax_utm_traffic(RMax_utm_traffic):
             try:
                 return RMax_utm_traffic_source_resp_coding[RMax_utm_traffic]
             except KeyError:
                 return [1/2] * 2
```

```
In [49]: X_train_cat_2_cols = X_train_cat[['SMax_utm_traffic_source', 'RMax_utm_traffic_source
         X_test_cat_2_cols = X_test_cat[['SMax_utm_traffic_source', 'RMax_utm_traffic_source']]
```

```
In [50]: X_train_cat_responseCoding_SMax = np.array(X_train_cat_2_cols.SMax_utm_traffic_source
         X_train_cat_responseCoding_RMax = np.array(X_train_cat_2_cols.RMax_utm_traffic_source
```

```
In [51]: X_test_cat_responseCoding_SMax = np.array(X_test_cat_2_cols.SMax_utm_traffic_source.a
         X_test_cat_responseCoding_RMax = np.array(X_test_cat_2_cols.RMax_utm_traffic_source.a
```

```
In [52]: X_train_cat_responseCoding_SMax.shape
```

```
Out[52]: (9768, 2)
```

```
In [53]: X_test_cat_responseCoding_SMax.shape
```

```
Out[53]: (3257, 2)
```

```
In [54]: X_train_cat_responseCoding_RMax.shape
```

```
Out[54]: (9768, 2)
```

```
In [55]: X_test_cat_responseCoding_RMax.shape
```

```
Out[55]: (3257, 2)
```

0.2.3 Merging all the data parts

```
In [67]: train1 = np.hstack((X_train_num_v1.values, X_train_cat_dummy.values,
                  X_train_cat_responseCoding_SMax, X_train_cat_responseCoding_RMax))
```

```
In [68]: train2 = np.hstack((X_train_num_v2.values, X_train_cat_dummy.values,
                  X_train_cat_responseCoding_SMax, X_train_cat_responseCoding_RMax))
```

```
In [72]: test1 = np.hstack((X_test_num_v1.values, X_test_cat_dummy.values,
                  X_test_cat_responseCoding_SMax, X_test_cat_responseCoding_RMax))
```

```
In [73]: test2 = np.hstack((X_test_num_v2.values, X_test_cat_dummy.values,
                            X_test_cat_responseCoding_SMax, X_test_cat_responseCoding_RMax))

In [74]: np.save('train1', train1)
         np.save('train2', train2)
         np.save('test1', test1)
         np.save('test2', test2)
         np.save('y_train', y_train)
         np.save('y_test', y_test)
```

0.2.4 Model building

We are skipping the feature selection because most of the features show very less importances. We shall build 2 models. One is a GLM model and another is a tree based ensemble. Looking at the number of features, we shall use LogisticRegression and as number of data points are less, we shall lead to overfitting the model and hence we shall use an RBF kernel SVM to benchmark an XGBoost model and we refrain from Random Forest.

GLM - Logistic Regression (On type 1)

```
In [82]: from sklearn.model_selection import GridSearchCV
         from sklearn.linear_model import LogisticRegression
         from sklearn.model_selection import KFold

In [83]: params = {
             'C': [1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3, 1e4],
             'penalty': ['l1', 'l2']
         }

         estimator = LogisticRegression(random_state=42)

         grid = GridSearchCV(estimator=estimator,
                             param_grid=params,
                             scoring={'roc_auc', 'neg_log_loss', 'f1', 'accuracy'},
                             refit='roc_auc', # Because we are using multiple evaluation metrics
                             cv=KFold(n_splits=3, shuffle=True, random_state=42),
                             return_train_score=True,
                             verbose=2,
                             n_jobs=16)

In [84]: grid.fit(train1, y_train)

Fitting 3 folds for each of 18 candidates, totalling 54 fits


[Parallel(n_jobs=16)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done   9 tasks      | elapsed:    2.1s
[Parallel(n_jobs=16)]: Done  51 out of  54 | elapsed:    6.9s remaining:    0.4s
[Parallel(n_jobs=16)]: Done  54 out of  54 | elapsed:    7.2s finished
```

```
/home/mayukhpay/test/lib/python3.5/site-packages/sklearn/linear_model/logistic.py:433: FutureWa
  FutureWarning)


Out[84]: GridSearchCV(cv=KFold(n_splits=3, random_state=42, shuffle=True),
                error_score='raise-deprecating',
                estimator=LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercep
                   intercept_scaling=1, max_iter=100, multi_class='warn',
                   n_jobs=None, penalty='l2', random_state=42, solver='warn',
                   tol=0.0001, verbose=0, warm_start=False),
                fit_params=None, iid='warn', n_jobs=16,
                param_grid={'penalty': ['l1', 'l2'], 'C': [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0
                pre_dispatch='2*n_jobs', refit='roc_auc', return_train_score=True,
                scoring={'f1', 'roc_auc', 'accuracy', 'neg_log_loss'}, verbose=2)

In [85]: from sklearn.metrics import roc_auc_score

In [86]: predict_train = grid.best_estimator_.predict(train1)
         print("The train AUC is:",roc_auc_score(y_train, predict_train))

         predict_test = grid.best_estimator_.predict(test1)
         print("The test AUC is:",roc_auc_score(y_test, predict_test))

The train AUC is: 0.569175026819538
The test AUC is: 0.5584276937487946
```

GLM - Logistic Regression (On type 2)

```
In [87]: grid.fit(train2, y_train)

Fitting 3 folds for each of 18 candidates, totalling 54 fits


[Parallel(n_jobs=16)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done   9 tasks      | elapsed:    0.6s
[Parallel(n_jobs=16)]: Done  23 out of  54 | elapsed:    1.9s remaining:    2.5s
[Parallel(n_jobs=16)]: Done  54 out of  54 | elapsed:    6.5s finished
/home/mayukhpay/test/lib/python3.5/site-packages/sklearn/linear_model/logistic.py:433: FutureWa
  FutureWarning)


Out[87]: GridSearchCV(cv=KFold(n_splits=3, random_state=42, shuffle=True),
                error_score='raise-deprecating',
                estimator=LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercep
                   intercept_scaling=1, max_iter=100, multi_class='warn',
                   n_jobs=None, penalty='l2', random_state=42, solver='warn',
                   tol=0.0001, verbose=0, warm_start=False),
                fit_params=None, iid='warn', n_jobs=16,
```

```
                param_grid={'penalty': ['l1', 'l2'], 'C': [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0
                pre_dispatch='2*n_jobs', refit='roc_auc', return_train_score=True,
                scoring={'f1', 'roc_auc', 'accuracy', 'neg_log_loss'}, verbose=2)
```

```
In [89]: predict_train = grid.best_estimator_.predict(train2)
         print("The train AUC is:",roc_auc_score(y_train, predict_train))

         predict_test = grid.best_estimator_.predict(test2)
         print("The test AUC is:",roc_auc_score(y_test, predict_test))
```

```
The train AUC is: 0.569175026819538
The test AUC is: 0.5584276937487946
```

Non-Linear -SVM (RBF) for benchmarking (On type 1)

```
In [90]: from sklearn.svm import SVC
```

```
In [93]: params = {
             'C': [1e-4, 1e-3, 1e-2, 1e-1,
                   1e0, 1e1, 1e2, 1e3, 1e4],
         }
         estimator = SVC(random_state=42, kernel='rbf')

         grid = GridSearchCV(estimator=estimator,
                         param_grid=params,
                         scoring={'roc_auc', 'f1', 'accuracy'},
                         refit='roc_auc', # Because we are using multiple evaluation metrics
                         cv=KFold(n_splits=3, shuffle=True, random_state=42),
                         return_train_score=True,
                         verbose=2,
                         n_jobs=16)
```

```
In [94]: grid.fit(train1, y_train)
```

```
Fitting 3 folds for each of 9 candidates, totalling 27 fits


[Parallel(n_jobs=16)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done  10 out of  27 | elapsed:   49.4s remaining:  1.4min
[Parallel(n_jobs=16)]: Done  24 out of  27 | elapsed:  1.7min remaining:   12.7s
[Parallel(n_jobs=16)]: Done  27 out of  27 | elapsed:  2.3min finished
/home/mayukhpay/test/lib/python3.5/site-packages/sklearn/svm/base.py:196: FutureWarning: The de
  "avoid this warning.", FutureWarning)
```

```
Out[94]: GridSearchCV(cv=KFold(n_splits=3, random_state=42, shuffle=True),
                error_score='raise-deprecating',
                estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
```

```
          decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
          kernel='rbf', max_iter=-1, probability=False, random_state=42,
          shrinking=True, tol=0.001, verbose=False),
               fit_params=None, iid='warn', n_jobs=16,
               param_grid={'C': [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0, 10000.0]
               pre_dispatch='2*n_jobs', refit='roc_auc', return_train_score=True,
               scoring={'f1', 'roc_auc', 'accuracy'}, verbose=2)
```

```
In [95]: predict_train = grid.best_estimator_.predict(train1)
         print("The train AUC is:",roc_auc_score(y_train, predict_train))

         predict_test = grid.best_estimator_.predict(test1)
         print("The test AUC is:",roc_auc_score(y_test, predict_test))
```

```
The train AUC is: 0.5074890417043816
The test AUC is: 0.504184918863818
```

Non-Linear -XGBoost

```
In [98]: import xgboost as xgb
         from sklearn.model_selection import RandomizedSearchCV
```

```
In [97]: x_cfl = xgb.XGBClassifier()

         prams={
              'learning_rate':[0.01,0.03,0.05,0.1,0.15,0.2],
              'n_estimators':[100, 200, 500, 1000, 2000],
              'max_depth':[3,5,10],
              'colsample_bytree':[0.1,0.3,0.5,1],
              'subsample':[0.1,0.3,0.5,1]
         }
```

```
In [99]: random_cfl = RandomizedSearchCV(x_cfl, param_distributions=prams, verbose=1, n_jobs=-
         random_cfl.fit(train1, y_train)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits


[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=-1)]: Done  18 tasks      | elapsed:   23.7s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:  1.0min finished


Out[99]: RandomizedSearchCV(cv=5, error_score='raise-deprecating',
                    estimator=XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=
                colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,
                max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
                n_jobs=1, nthread=None, objective='binary:logistic', random_state=0,
```

```
                reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
                silent=True, subsample=1),
                   fit_params=None, iid='warn', n_iter=10, n_jobs=-1,
                   param_distributions={'n_estimators': [100, 200, 500, 1000, 2000], 'colsample
                   pre_dispatch='2*n_jobs', random_state=None, refit=True,
                   return_train_score='warn', scoring=None, verbose=1)

In [100]: predict_train = grid.best_estimator_.predict(train1)
          print("The train AUC is:",roc_auc_score(y_train, predict_train))

          predict_test = grid.best_estimator_.predict(test1)
          print("The test AUC is:",roc_auc_score(y_test, predict_test))

The train AUC is: 0.5074890417043816
The test AUC is: 0.504184918863818


In [ ]:
```