Customer Revenue Prediction - Kaggle

October 23, 2018

1 Google Customer Revenue Prediction

We aim to predict how much Google Store customers will spend on the Google products

Data Import and Exploration

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import gc
        import os
        import seaborn as sb
        import time
        import json
        import datetime as dt

        from sklearn import metrics
        from pandas.io.json import json_normalize
        %matplotlib inline
```

```
In [2]: train_df =  pd.read_csv('./train3.csv')
        train_df.dtypes
```

```
/home/jaytorasakar8/anaconda3/lib/python3.6/site-packages/IPython/core/interactiveshell.py:2785:
  interactivity=interactivity, compiler=compiler, result=result)
```

```
Out[2]: channelGrouping         object
        date                     int64
        device                  object
        fullVisitorId           object
        geoNetwork              object
        sessionId               object
        socialEngagementType    object
        totals                  object
        trafficSource           object
        visitId                  int64
```

```
visitNumber              int64
visitStartTime           int64
dtype: object
```

```
In [3]: print(train_df.shape[0])
```

```
903653
```

We see that our dataset is of 900K tuples

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

```
Out[4]:   channelGrouping      date  \
        0  Organic Search  20160902
        1  Organic Search  20160902
        2  Organic Search  20160902
        3  Organic Search  20160902
        4  Organic Search  20160902

                                                   device        fullVisitorId  \
        0  {"browser": "Chrome", "browserVersion": "not a...  1131660440785968503
        1  {"browser": "Firefox", "browserVersion": "not ...   377306020877927890
        2  {"browser": "Chrome", "browserVersion": "not a...  3895546263509774583
        3  {"browser": "UC Browser", "browserVersion": "n...  4763447161404445595
        4  {"browser": "Chrome", "browserVersion": "not a...    27294437909732085

                                              geoNetwork  \
        0  {"continent": "Asia", "subContinent": "Western...
        1  {"continent": "Oceania", "subContinent": "Aust...
        2  {"continent": "Europe", "subContinent": "South...
        3  {"continent": "Asia", "subContinent": "Southea...
        4  {"continent": "Europe", "subContinent": "North...

                             sessionId  socialEngagementType  \
        0  1131660440785968503_1472830385  Not Socially Engaged
        1   377306020877927890_1472880147  Not Socially Engaged
        2  3895546263509774583_1472865386  Not Socially Engaged
        3  4763447161404445595_1472881213  Not Socially Engaged
        4    27294437909732085_1472822600  Not Socially Engaged

                                                   totals  \
        0  {"visits": "1", "hits": "1", "pageviews": "1",...
        1  {"visits": "1", "hits": "1", "pageviews": "1",...
        2  {"visits": "1", "hits": "1", "pageviews": "1",...
        3  {"visits": "1", "hits": "1", "pageviews": "1",...
        4  {"visits": "1", "hits": "1", "pageviews": "1",...
```

```
                                          trafficSource     visitId  visitNumber  \
0  {"campaign": "(not set)", "source": "google", ...  1472830385            1
1  {"campaign": "(not set)", "source": "google", ...  1472880147            1
2  {"campaign": "(not set)", "source": "google", ...  1472865386            1
3  {"campaign": "(not set)", "source": "google", ...  1472881213            1
4  {"campaign": "(not set)", "source": "google", ...  1472822600            2

   visitStartTime
0      1472830385
1      1472880147
2      1472865386
3      1472881213
4      1472822600
```

Data is in JSON Format for the columns: Device, geonetworks, totals and traffic source Since the data is in JSON Format we convert the given data into standard format

```
In [5]: columns_in_json = ['device', 'geoNetwork', 'totals', 'trafficSource']

        #We need to reload the Dataframe with all the data formatting

        def load_df(path_name):
            df = pd.read_csv(path_name, converters = {column: json.loads for column in columns_i

            for column in columns_in_json:
                json_column_as_df = json_normalize(df[column])
                json_column_as_df.columns = [f"{column}.{subcolumn}" for subcolumn in json_colum
                df = df.drop(column, axis = 1).merge(json_column_as_df, right_index = True, left

            return df

        #Reference: https://medium.com/@gis10kwo/converting-nested-json-data-to-csv-using-python

In [6]: train_df = load_df('./train3.csv')
        test_df = load_df('./test3.csv')

In [7]: pd.set_option('display.max_columns', None)
        test_df.head()

Out[7]:   channelGrouping      date        fullVisitorId  \
        0  Organic Search  20171016  6167871330617112363
        1  Organic Search  20171016  0643697640977915618
        2  Organic Search  20171016  6059383810968229466
        3  Organic Search  20171016  2376720078563423631
        4  Organic Search  20171016  2314544520795440038

                                sessionId  socialEngagementType     visitId  \
        0  6167871330617112363_1508151024  Not Socially Engaged  1508151024
```

```
1  0643697640977915618_1508175522  Not Socially Engaged  1508175522
2  6059383810968229466_1508143220  Not Socially Engaged  1508143220
3  2376720078563423631_1508193530  Not Socially Engaged  1508193530
4  2314544520795440038_1508217442  Not Socially Engaged  1508217442

   visitNumber  visitStartTime device.browser            device.browserSize  \
0            2      1508151024         Chrome  not available in demo dataset
1            1      1508175522         Chrome  not available in demo dataset
2            1      1508143220         Chrome  not available in demo dataset
3            1      1508193530         Safari  not available in demo dataset
4            1      1508217442         Safari  not available in demo dataset

          device.browserVersion device.deviceCategory  \
0  not available in demo dataset               desktop
1  not available in demo dataset               desktop
2  not available in demo dataset               desktop
3  not available in demo dataset                mobile
4  not available in demo dataset               desktop

            device.flashVersion  device.isMobile  \
0  not available in demo dataset            False
1  not available in demo dataset            False
2  not available in demo dataset            False
3  not available in demo dataset             True
4  not available in demo dataset            False

                device.language    device.mobileDeviceBranding  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

        device.mobileDeviceInfo device.mobileDeviceMarketingName  \
0  not available in demo dataset    not available in demo dataset
1  not available in demo dataset    not available in demo dataset
2  not available in demo dataset    not available in demo dataset
3  not available in demo dataset    not available in demo dataset
4  not available in demo dataset    not available in demo dataset

       device.mobileDeviceModel     device.mobileInputSelector  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

  device.operatingSystem  device.operatingSystemVersion  \
```

```
0              Macintosh  not available in demo dataset
1                Windows  not available in demo dataset
2              Macintosh  not available in demo dataset
3                    iOS  not available in demo dataset
4              Macintosh  not available in demo dataset

            device.screenColors        device.screenResolution  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

                geoNetwork.city             geoNetwork.cityId  \
0                     (not set)  not available in demo dataset
1                      Zaragoza  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3                  Mountain View  not available in demo dataset
4                       San Jose  not available in demo dataset

  geoNetwork.continent geoNetwork.country           geoNetwork.latitude  \
0                 Asia          Singapore  not available in demo dataset
1               Europe              Spain  not available in demo dataset
2               Europe             France  not available in demo dataset
3             Americas      United States  not available in demo dataset
4             Americas      United States  not available in demo dataset

           geoNetwork.longitude                    geoNetwork.metro  \
0  not available in demo dataset                           (not set)
1  not available in demo dataset                           (not set)
2  not available in demo dataset      not available in demo dataset
3  not available in demo dataset  San Francisco-Oakland-San Jose CA
4  not available in demo dataset  San Francisco-Oakland-San Jose CA

  geoNetwork.networkDomain     geoNetwork.networkLocation  \
0       myrepublic.com.sg  not available in demo dataset
1            rima-tde.net  not available in demo dataset
2                sfr.net  not available in demo dataset
3              (not set)  not available in demo dataset
4              (not set)  not available in demo dataset

              geoNetwork.region geoNetwork.subContinent totals.bounces  \
0                     (not set)          Southeast Asia            NaN
1                        Aragon         Southern Europe            NaN
2  not available in demo dataset          Western Europe            NaN
3                    California        Northern America            NaN
4                    California        Northern America            NaN
```

```
  totals.hits totals.newVisits totals.pageviews totals.visits  \
0           4              NaN                4             1
1           5                1                5             1
2           7                1                7             1
3           8                1                4             1
4           9                1                4             1

  trafficSource.adContent trafficSource.adwordsClickInfo.adNetworkType  \
0                     NaN                                           NaN
1                     NaN                                           NaN
2                     NaN                                           NaN
3                     NaN                                           NaN
4                     NaN                                           NaN

  trafficSource.adwordsClickInfo.criteriaParameters  \
0                     not available in demo dataset
1                     not available in demo dataset
2                     not available in demo dataset
3                     not available in demo dataset
4                     not available in demo dataset

  trafficSource.adwordsClickInfo.gclId  \
0                                  NaN
1                                  NaN
2                                  NaN
3                                  NaN
4                                  NaN

  trafficSource.adwordsClickInfo.isVideoAd  \
0                                      NaN
1                                      NaN
2                                      NaN
3                                      NaN
4                                      NaN

  trafficSource.adwordsClickInfo.page trafficSource.adwordsClickInfo.slot  \
0                                 NaN                                  NaN
1                                 NaN                                  NaN
2                                 NaN                                  NaN
3                                 NaN                                  NaN
4                                 NaN                                  NaN

  trafficSource.campaign trafficSource.isTrueDirect trafficSource.keyword  \
0             (not set)                       True        (not provided)
1             (not set)                        NaN        (not provided)
2             (not set)                        NaN        (not provided)
3             (not set)                        NaN        (not provided)
4             (not set)                        NaN        (not provided)
```

```
  trafficSource.medium trafficSource.referralPath trafficSource.source
0              organic                       NaN               google
1              organic                       NaN               google
2              organic                       NaN               google
3              organic                       NaN               google
4              organic                       NaN               google
```

In [8]:
```
pd.set_option('display.max_columns', None)
train_df.head()

#For displaying all columns
#Source: https://stackoverflow.com/questions/28775813/not-able-to-view-all-columns-in-pa
```

Out[8]:
```
  channelGrouping      date        fullVisitorId  \
0  Organic Search  20160902  1131660440785968503
1  Organic Search  20160902   377306020877927890
2  Organic Search  20160902  3895546263509774583
3  Organic Search  20160902  4763447161404445595
4  Organic Search  20160902    27294437909732085

                        sessionId  socialEngagementType     visitId  \
0  1131660440785968503_1472830385  Not Socially Engaged  1472830385
1   377306020877927890_1472880147  Not Socially Engaged  1472880147
2  3895546263509774583_1472865386  Not Socially Engaged  1472865386
3  4763447161404445595_1472881213  Not Socially Engaged  1472881213
4    27294437909732085_1472822600  Not Socially Engaged  1472822600

   visitNumber  visitStartTime device.browser            device.browserSize  \
0            1      1472830385         Chrome  not available in demo dataset
1            1      1472880147        Firefox  not available in demo dataset
2            1      1472865386         Chrome  not available in demo dataset
3            1      1472881213     UC Browser  not available in demo dataset
4            2      1472822600         Chrome  not available in demo dataset

          device.browserVersion device.deviceCategory  \
0  not available in demo dataset              desktop
1  not available in demo dataset              desktop
2  not available in demo dataset              desktop
3  not available in demo dataset              desktop
4  not available in demo dataset               mobile

            device.flashVersion  device.isMobile  \
0  not available in demo dataset           False
1  not available in demo dataset           False
2  not available in demo dataset           False
3  not available in demo dataset           False
4  not available in demo dataset            True
```

```
                device.language    device.mobileDeviceBranding  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

        device.mobileDeviceInfo device.mobileDeviceMarketingName  \
0  not available in demo dataset    not available in demo dataset
1  not available in demo dataset    not available in demo dataset
2  not available in demo dataset    not available in demo dataset
3  not available in demo dataset    not available in demo dataset
4  not available in demo dataset    not available in demo dataset

       device.mobileDeviceModel     device.mobileInputSelector  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

  device.operatingSystem  device.operatingSystemVersion  \
0                Windows  not available in demo dataset
1              Macintosh  not available in demo dataset
2                Windows  not available in demo dataset
3                  Linux  not available in demo dataset
4                Android  not available in demo dataset

            device.screenColors        device.screenResolution  \
0  not available in demo dataset  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

                geoNetwork.city              geoNetwork.cityId  \
0                         Izmir  not available in demo dataset
1  not available in demo dataset  not available in demo dataset
2                        Madrid  not available in demo dataset
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

  geoNetwork.continent geoNetwork.country           geoNetwork.latitude  \
0                 Asia             Turkey  not available in demo dataset
1              Oceania          Australia  not available in demo dataset
2               Europe              Spain  not available in demo dataset
3                 Asia          Indonesia  not available in demo dataset
```

```
4              Europe     United Kingdom  not available in demo dataset

           geoNetwork.longitude               geoNetwork.metro  \
0  not available in demo dataset                     (not set)
1  not available in demo dataset  not available in demo dataset
2  not available in demo dataset                     (not set)
3  not available in demo dataset  not available in demo dataset
4  not available in demo dataset  not available in demo dataset

  geoNetwork.networkDomain     geoNetwork.networkLocation  \
0             ttnet.com.tr  not available in demo dataset
1              dodo.net.au  not available in demo dataset
2          unknown.unknown  not available in demo dataset
3          unknown.unknown  not available in demo dataset
4          unknown.unknown  not available in demo dataset

              geoNetwork.region geoNetwork.subContinent totals.bounces  \
0                         Izmir            Western Asia              1
1  not available in demo dataset             Australasia              1
2           Community of Madrid         Southern Europe              1
3  not available in demo dataset          Southeast Asia              1
4  not available in demo dataset         Northern Europe              1

  totals.hits totals.newVisits totals.pageviews totals.transactionRevenue  \
0           1                1                1                        NaN
1           1                1                1                        NaN
2           1                1                1                        NaN
3           1                1                1                        NaN
4           1              NaN                1                        NaN

  totals.visits trafficSource.adContent  \
0             1                     NaN
1             1                     NaN
2             1                     NaN
3             1                     NaN
4             1                     NaN

  trafficSource.adwordsClickInfo.adNetworkType  \
0                                          NaN
1                                          NaN
2                                          NaN
3                                          NaN
4                                          NaN

  trafficSource.adwordsClickInfo.criteriaParameters  \
0                     not available in demo dataset
1                     not available in demo dataset
2                     not available in demo dataset
```

```
3                     not available in demo dataset
4                     not available in demo dataset

  trafficSource.adwordsClickInfo.gclId  \
0                                  NaN
1                                  NaN
2                                  NaN
3                                  NaN
4                                  NaN

  trafficSource.adwordsClickInfo.isVideoAd  \
0                                      NaN
1                                      NaN
2                                      NaN
3                                      NaN
4                                      NaN

  trafficSource.adwordsClickInfo.page trafficSource.adwordsClickInfo.slot  \
0                                 NaN                                  NaN
1                                 NaN                                  NaN
2                                 NaN                                  NaN
3                                 NaN                                  NaN
4                                 NaN                                  NaN

  trafficSource.campaign trafficSource.campaignCode  \
0             (not set)                        NaN
1             (not set)                        NaN
2             (not set)                        NaN
3             (not set)                        NaN
4             (not set)                        NaN

  trafficSource.isTrueDirect trafficSource.keyword trafficSource.medium  \
0                        NaN        (not provided)              organic
1                        NaN        (not provided)              organic
2                        NaN        (not provided)              organic
3                        NaN       google + online              organic
4                       True        (not provided)              organic

  trafficSource.referralPath trafficSource.source
0                        NaN               google
1                        NaN               google
2                        NaN               google
3                        NaN               google
4                        NaN               google
```

Format the given date in regular format of Y/M/D format from the given POSIX format

```
In [9]: temp = train_df['date'].apply(lambda x: dt.datetime.strptime(str(x), "%Y%m%d") )
        train_df['date'] = temp

        #Reference: https://stackoverflow.com/questions/30132282/datetime-to-string-with-series-
```

We are calculating the Log value of the Transaction Revenue!

```
In [10]: log_values = train_df['totals.transactionRevenue'].fillna(0).astype(float)
         log_values = log_values.apply(lambda x: np.log1p(x))
         train_df['totals.transactionRevenue'] = log_values
         train_df['totals.transactionRevenue'].describe()

Out[10]: count    903653.000000
         mean          0.227118
         std           2.003710
         min           0.000000
         25%           0.000000
         50%           0.000000
         75%           0.000000
         max          23.864375
         Name: totals.transactionRevenue, dtype: float64
```

We are trying to find the columns which are constant and not having any impact on our prediction

```
In [11]: const_cols = [c for c in train_df.columns if train_df[c].nunique(dropna=False)==1 ]
         const_cols

Out[11]: ['socialEngagementType',
          'device.browserSize',
          'device.browserVersion',
          'device.flashVersion',
          'device.language',
          'device.mobileDeviceBranding',
          'device.mobileDeviceInfo',
          'device.mobileDeviceMarketingName',
          'device.mobileDeviceModel',
          'device.mobileInputSelector',
          'device.operatingSystemVersion',
          'device.screenColors',
          'device.screenResolution',
          'geoNetwork.cityId',
          'geoNetwork.latitude',
          'geoNetwork.longitude',
          'geoNetwork.networkLocation',
          'totals.visits',
          'trafficSource.adwordsClickInfo.criteriaParameters']

In [12]: print("Number of unique visitors in train set : ",train_df.fullVisitorId.nunique())
```

```
Number of unique visitors in train set :  714167
```

```
In [13]: print("Variables not in test but in train : ", set(train_df.columns).difference(set(tes

Variables not in test but in train :  {'totals.transactionRevenue', 'trafficSource.campaignCode'
```

We need to drop columns which are not present in both the train and test data set

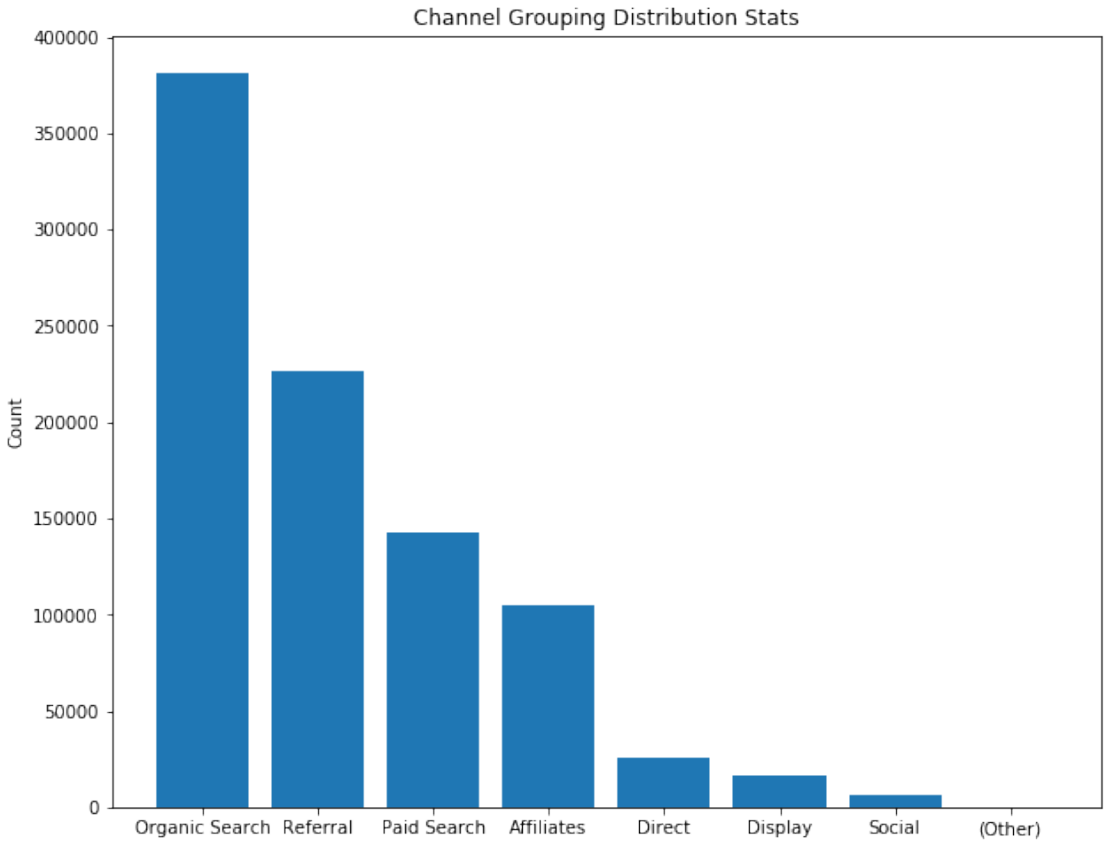
```
In [14]: cols_to_drop = const_cols + ['sessionId']

         train_df = train_df.drop(cols_to_drop + ["trafficSource.campaignCode"], axis=1)
         test_df = test_df.drop(cols_to_drop, axis=1)
```

```
In [15]: train_df.columns.values
```

```
Out[15]: array(['channelGrouping', 'date', 'fullVisitorId', 'visitId',
                'visitNumber', 'visitStartTime', 'device.browser',
                'device.deviceCategory', 'device.isMobile',
                'device.operatingSystem', 'geoNetwork.city',
                'geoNetwork.continent', 'geoNetwork.country', 'geoNetwork.metro',
                'geoNetwork.networkDomain', 'geoNetwork.region',
                'geoNetwork.subContinent', 'totals.bounces', 'totals.hits',
                'totals.newVisits', 'totals.pageviews',
                'totals.transactionRevenue', 'trafficSource.adContent',
                'trafficSource.adwordsClickInfo.adNetworkType',
                'trafficSource.adwordsClickInfo.gclId',
                'trafficSource.adwordsClickInfo.isVideoAd',
                'trafficSource.adwordsClickInfo.page',
                'trafficSource.adwordsClickInfo.slot', 'trafficSource.campaign',
                'trafficSource.isTrueDirect', 'trafficSource.keyword',
                'trafficSource.medium', 'trafficSource.referralPath',
                'trafficSource.source'], dtype=object)
```

We are looking in Channel Grouping Distribution Statistics

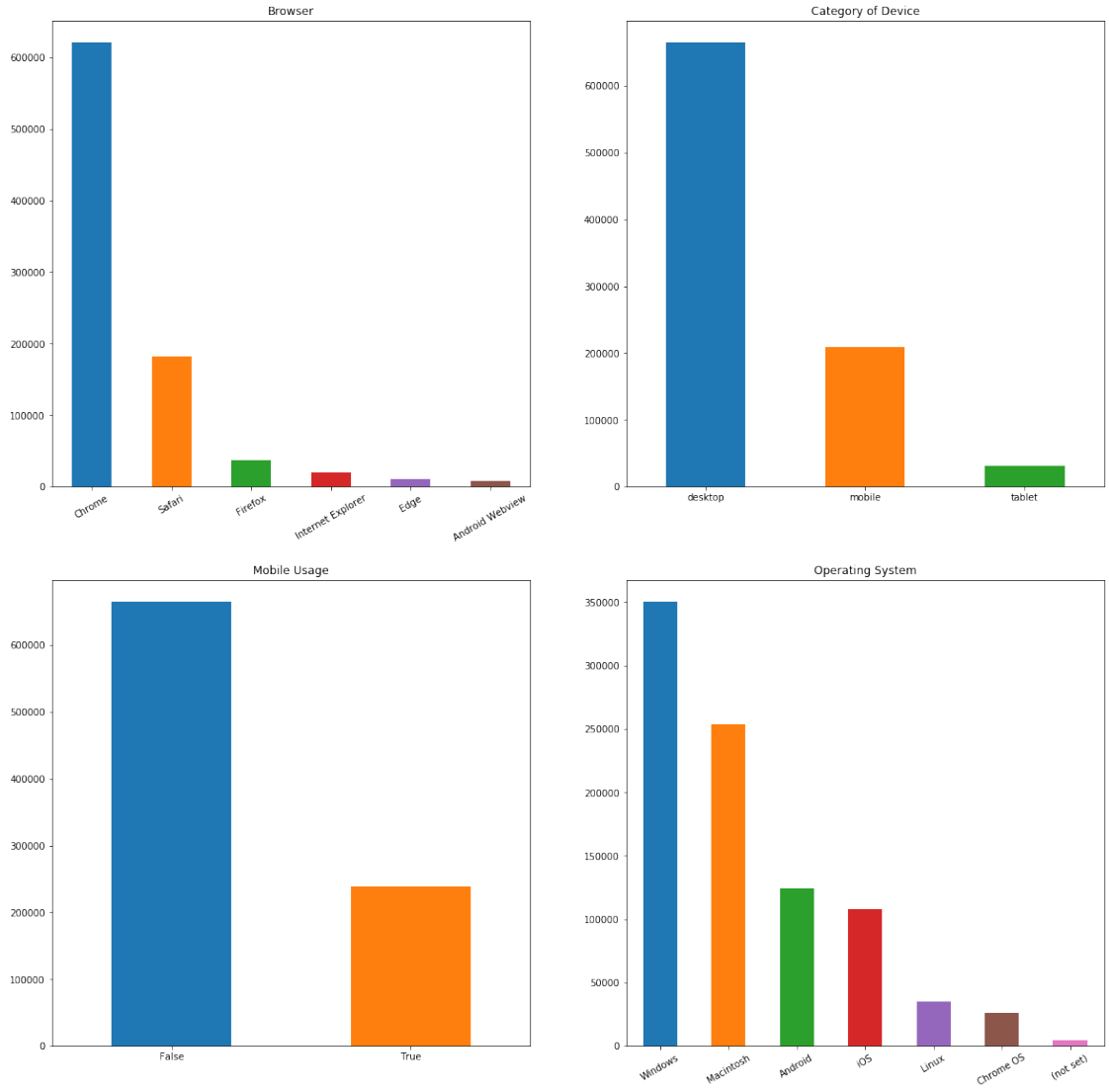
```
In [16]: channel_group_column = train_df["channelGrouping"].unique()
         channel_group_count = train_df["channelGrouping"].value_counts()

         plt.figure(figsize = (10,8))
         plt.bar(channel_group_column, channel_group_count, align='center', alpha=1)
         plt.ylabel('Count')
         plt.title('Channel Grouping Distribution Stats')
         plt.show()
```



1.0.1 Plots based on the Devices and it's various Categories

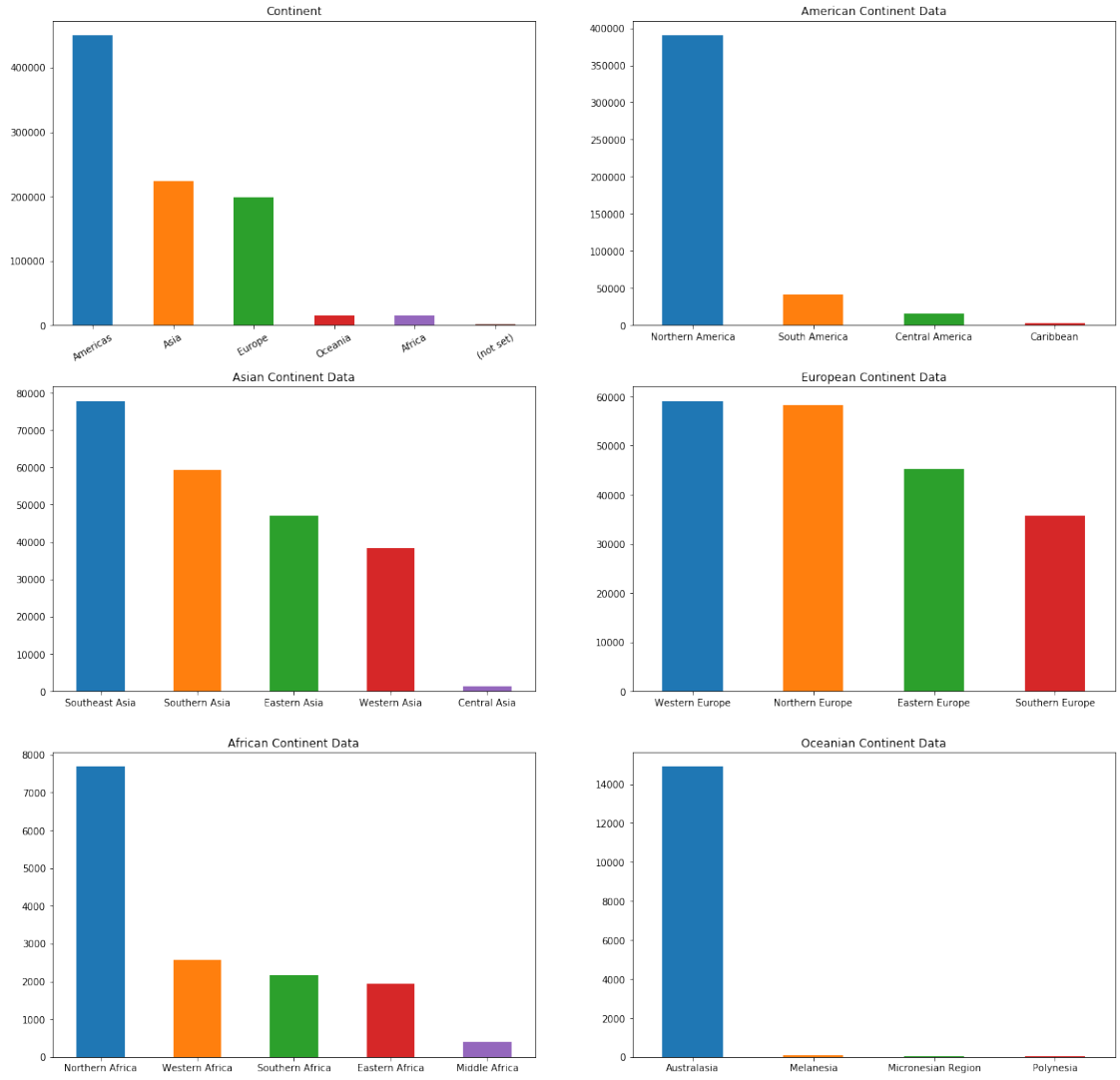
In [17]:
```
fig, axes = plt.subplots(2,2,figsize=(20,20))

train_df["device.browser"].value_counts().head(6).plot.bar(ax=axes[0][0],rot=30, title=
train_df["device.deviceCategory"].value_counts().plot.bar(ax=axes[0][1],rot=0,title="Ca
train_df["device.isMobile"].value_counts().plot.bar(ax=axes[1][0],rot=0,title="Mobile U
train_df["device.operatingSystem"].value_counts().head(7).plot.bar(ax=axes[1][1],rot=30

#Reference: https://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFra
```

Out[17]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6c54988320>
```



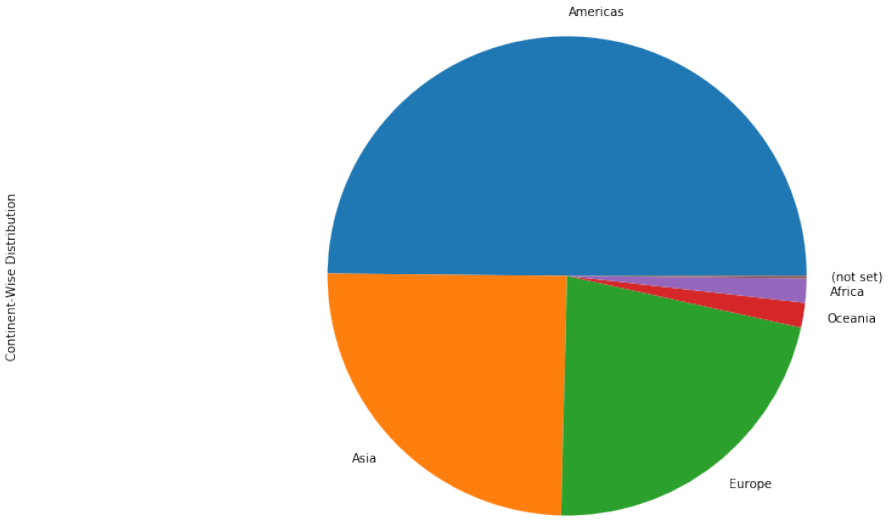
In [18]:
```
fig, axes = plt.subplots(3,2,figsize=(20,20))

train_df["geoNetwork.continent"].value_counts().plot.bar(ax=axes[0][0],rot=30, title="C
train_df[train_df["geoNetwork.continent"] == "Americas"]["geoNetwork.subContinent"].val
train_df[train_df["geoNetwork.continent"] == "Asia"]["geoNetwork.subContinent"].value_c
train_df[train_df["geoNetwork.continent"] == "Europe"]["geoNetwork.subContinent"].value
train_df[train_df["geoNetwork.continent"] == "Africa"]["geoNetwork.subContinent"].value
train_df[train_df["geoNetwork.continent"] == "Oceania"]["geoNetwork.subContinent"].valu

#Reference: https://stackoverflow.com/questions/29498652/plot-bar-graph-from-pandas-dat
```

Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6c54beca58>
```

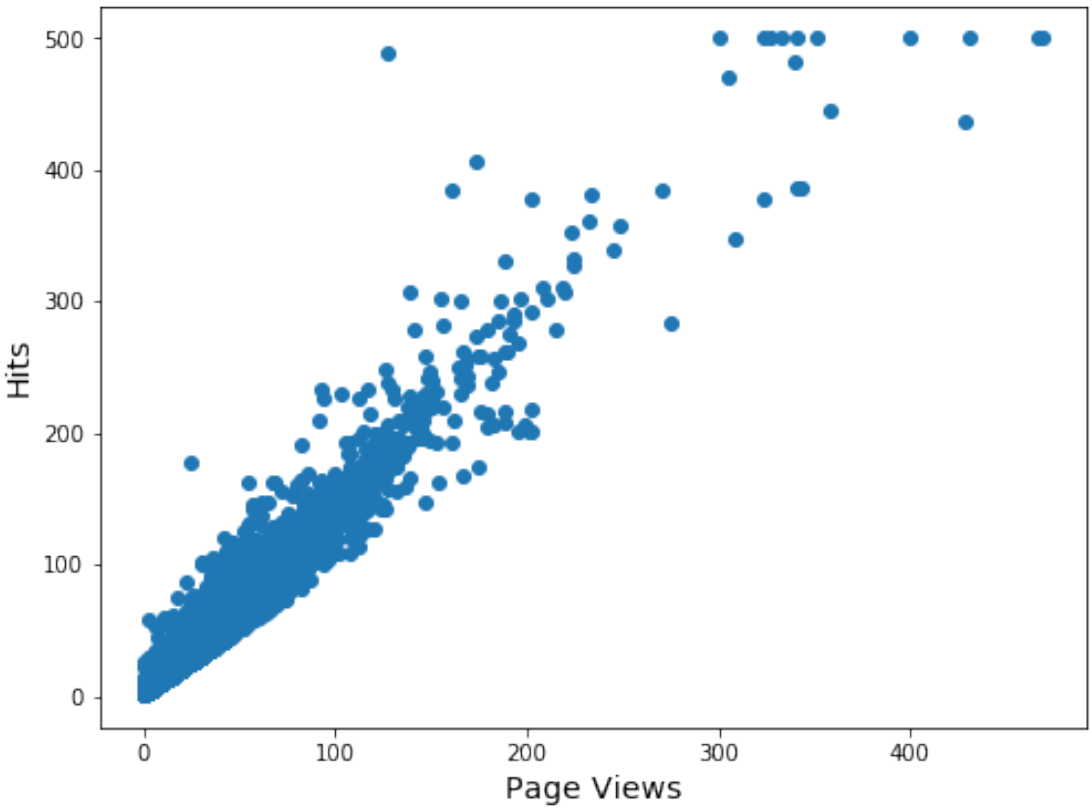



```
In [19]: plt.figure(figsize=(16,8))
         train_df["geoNetwork.continent"].value_counts().plot.pie(label='Continent-Wise Distribu
         plt.axis('equal')
```

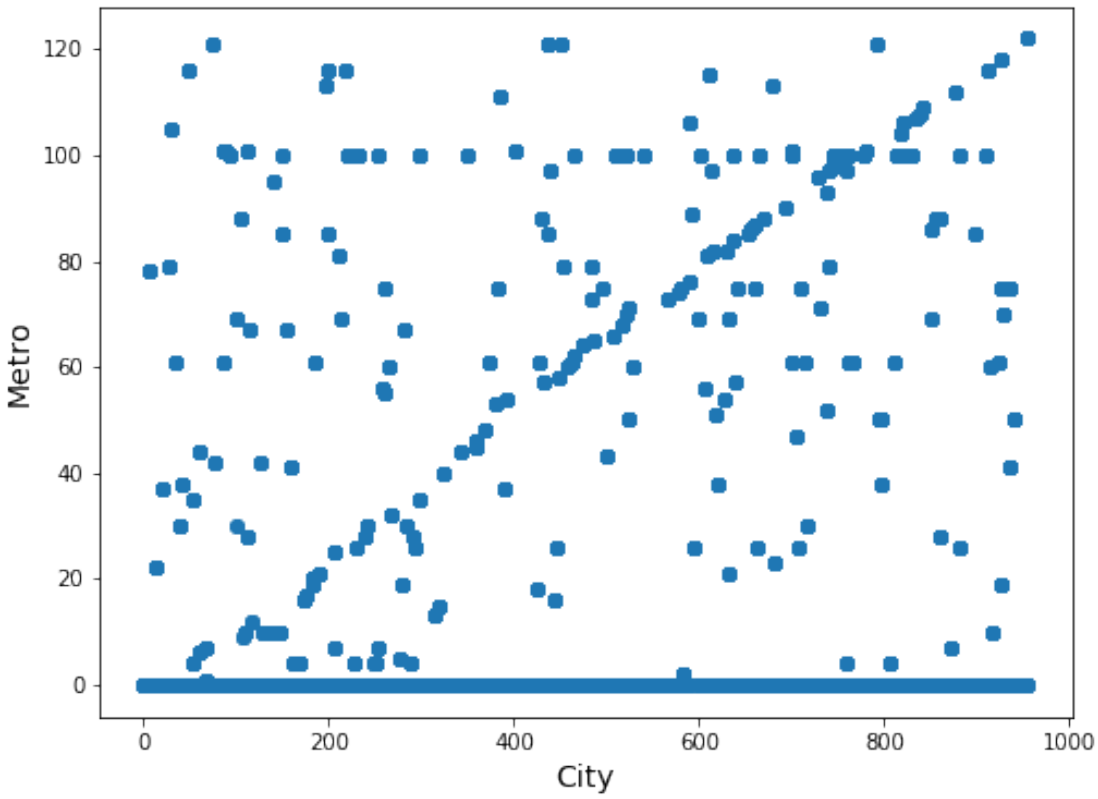
```
Out[19]: (-1.1027240173209008,
          1.1001297151105192,
          -1.105933359216009,
          1.1016642372813983)
```




```
In [52]: plt.figure(figsize = (8,6))
         plt.xlabel('Page Views', fontsize = 14)
         plt.ylabel('Hits', fontsize = 14)
         plt.scatter(train_df["totals.pageviews"], train_df["totals.hits"])
         plt.show()
```



In [55]:
```
plt.figure(figsize = (8,6))
plt.xlabel('City', fontsize = 14)
plt.ylabel('Metro', fontsize = 14)
plt.scatter(train_df["geoNetwork.city"], train_df["geoNetwork.metro"])
plt.show()
```



In [20]:
```
revenue_datetime_df = train_df[["totals.transactionRevenue" , "date"]].dropna()
revenue_datetime_df["revenue"] = revenue_datetime_df["totals.transactionRevenue"].astyp
revenue_datetime_df.head()
```

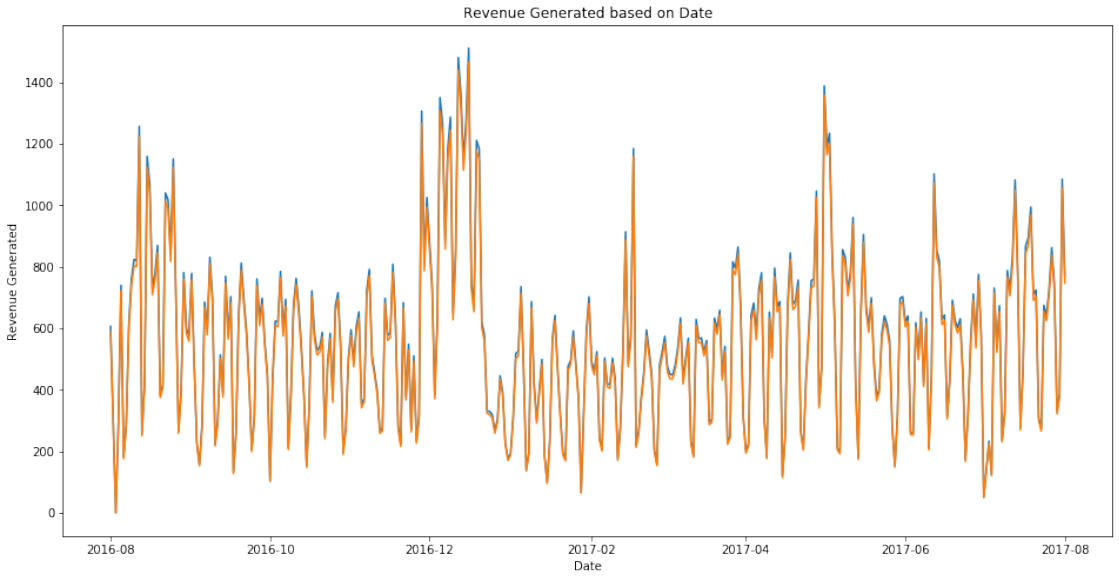
Out[20]:

	totals.transactionRevenue	date	revenue
0	0.0	2016-09-02	0
1	0.0	2016-09-02	0
2	0.0	2016-09-02	0
3	0.0	2016-09-02	0
4	0.0	2016-09-02	0

In [21]:
```
daily_revenue_df = revenue_datetime_df.groupby(by=["date"],axis = 0 ).sum()
```

```
fig, axes = plt.subplots(figsize=(16,8))

axes.set_title("Revenue Generated based on Date")
axes.set_xlabel("Date")
axes.set_ylabel("Revenue Generated")
axes.plot(daily_revenue_df)
```

```
Out[21]: [<matplotlib.lines.Line2D at 0x7f6c54daa518>,
          <matplotlib.lines.Line2D at 0x7f6c54d80470>]
```



1.0.2 Data Processing based on Category and Numerical Variables

```
In [22]: #Splitting the categorical variables and Numerical Variables

from sklearn import preprocessing
categorical_columns = ['channelGrouping', 'device.browser',
       'device.deviceCategory','device.operatingSystem', 'geoNetwork.city',
       'geoNetwork.continent', 'geoNetwork.country', 'geoNetwork.metro',
       'geoNetwork.networkDomain', 'geoNetwork.region',
       'geoNetwork.subContinent', 'trafficSource.adContent',
       'trafficSource.adwordsClickInfo.adNetworkType',
       'trafficSource.adwordsClickInfo.gclId',
       'trafficSource.adwordsClickInfo.isVideoAd',
       'trafficSource.adwordsClickInfo.page',
       'trafficSource.adwordsClickInfo.slot', 'trafficSource.campaign',
       'trafficSource.isTrueDirect', 'trafficSource.keyword',
       'trafficSource.medium', 'trafficSource.referralPath',
       'trafficSource.source']
```

```
    for column in categorical_columns:
        lbl = preprocessing.LabelEncoder()
        lbl.fit(list(train_df[column].values.astype('str')) + list(test_df[column].values.a
        train_df[column] = lbl.transform(list(train_df[column].values.astype('str')))
        test_df[column] = lbl.transform(list(test_df[column].values.astype('str')))
```

In [23]:
```
#For columns with numerical values
numerical_columns = ['totals.bounces', 'totals.hits',
       'totals.newVisits', 'totals.pageviews',
       'visitNumber', 'visitStartTime']

for column in numerical_columns:
    train_df[column] = train_df[column].astype(float)
    test_df[column] = test_df[column].astype(float)
```

In [24]:
```
train_df.describe()
```

Out[24]:

	channelGrouping	visitId	visitNumber	visitStartTime	\
count	903653.000000	9.036530e+05	903653.000000	9.036530e+05	
mean	4.632267	1.485007e+09	2.264897	1.485007e+09	
std	1.774791	9.022124e+06	9.283735	9.022124e+06	
min	0.000000	1.470035e+09	1.000000	1.470035e+09	
25%	4.000000	1.477561e+09	1.000000	1.477561e+09	
50%	4.000000	1.483949e+09	1.000000	1.483949e+09	
75%	7.000000	1.492759e+09	1.000000	1.492759e+09	
max	7.000000	1.501657e+09	395.000000	1.501657e+09	

	device.browser	device.deviceCategory	device.operatingSystem	\
count	903653.000000	903653.000000	903653.000000	
mean	44.014666	0.298370	12.949865	
std	15.389741	0.526058	8.159630	
min	0.000000	0.000000	0.000000	
25%	35.000000	0.000000	7.000000	
50%	35.000000	0.000000	20.000000	
75%	47.000000	1.000000	20.000000	
max	117.000000	2.000000	23.000000	

	geoNetwork.city	geoNetwork.continent	geoNetwork.country	\
count	903653.000000	903653.000000	903653.000000	
mean	740.380483	2.716869	163.035326	
std	302.648488	0.885558	69.196953	
min	0.000000	0.000000	0.000000	
25%	540.000000	2.000000	97.000000	
50%	955.000000	2.000000	210.000000	
75%	955.000000	3.000000	218.000000	
max	955.000000	5.000000	227.000000	

```
       geoNetwork.metro  geoNetwork.networkDomain  geoNetwork.region  \
count     903653.000000             903653.000000      903653.000000
mean          85.871039              18911.072901         352.329262
std           50.605625              16539.152347         173.492619
min            0.000000                  0.000000           0.000000
25%           46.000000                  0.000000         187.000000
50%          122.000000              16269.000000         482.000000
75%          122.000000              37466.000000         482.000000
max          122.000000              41980.000000         482.000000

       geoNetwork.subContinent  totals.bounces    totals.hits  \
count            903653.000000        450630.0  903653.000000
mean                 13.310106             1.0       4.596538
std                   4.678611             0.0       9.641437
min                   0.000000             1.0       1.000000
25%                  12.000000             1.0       1.000000
50%                  12.000000             1.0       2.000000
75%                  16.000000             1.0       4.000000
max                  22.000000             1.0     500.000000

       totals.newVisits  totals.pageviews  totals.transactionRevenue  \
count          703060.0     903553.000000              903653.000000
mean                1.0          3.849764                   0.227118
std                 0.0          7.025274                   2.003710
min                 1.0          1.000000                   0.000000
25%                 1.0          1.000000                   0.000000
50%                 1.0          1.000000                   0.000000
75%                 1.0          4.000000                   0.000000
max                 1.0        469.000000                  23.864375

       trafficSource.adContent  trafficSource.adwordsClickInfo.adNetworkType  \
count            903653.000000                                  903653.000000
mean                 61.549617                                       2.952512
std                   4.377804                                       0.304490
min                   0.000000                                       1.000000
25%                  62.000000                                       3.000000
50%                  62.000000                                       3.000000
75%                  62.000000                                       3.000000
max                  76.000000                                       3.000000

       trafficSource.adwordsClickInfo.gclId  \
count                         903653.000000
mean                           58408.270643
std                             4404.462630
min                                1.000000
25%                            59008.000000
50%                            59008.000000
75%                            59008.000000
```

```
max                            59008.000000

       trafficSource.adwordsClickInfo.isVideoAd  \
count                             903653.000000
mean                                   0.976252
std                                    0.152263
min                                    0.000000
25%                                    1.000000
50%                                    1.000000
75%                                    1.000000
max                                    1.000000

       trafficSource.adwordsClickInfo.page  \
count                         903653.000000
mean                             10.739166
std                               1.672862
min                               0.000000
25%                              11.000000
50%                              11.000000
75%                              11.000000
max                              11.000000

       trafficSource.adwordsClickInfo.slot  trafficSource.campaign  \
count                         903653.000000           903653.000000
mean                               2.975694                4.280819
std                                0.157577                1.510145
min                                1.000000                4.000000
25%                                3.000000                4.000000
50%                                3.000000                4.000000
75%                                3.000000                4.000000
max                                3.000000               34.000000

       trafficSource.isTrueDirect  trafficSource.keyword  \
count                903653.000000          903653.000000
mean                      0.696781            1897.047624
std                       0.459649            1640.605290
min                       0.000000               0.000000
25%                       0.000000              11.000000
50%                       1.000000            3327.000000
75%                       1.000000            3327.000000
max                       1.000000            5391.000000

       trafficSource.medium  trafficSource.referralPath  trafficSource.source
count         903653.000000                903653.000000         903653.000000
mean               4.456896                  2684.468135            254.317229
std                2.076703                   974.303431            166.515464
min                0.000000                     0.000000              0.000000
25%                5.000000                  2604.000000            208.000000
```

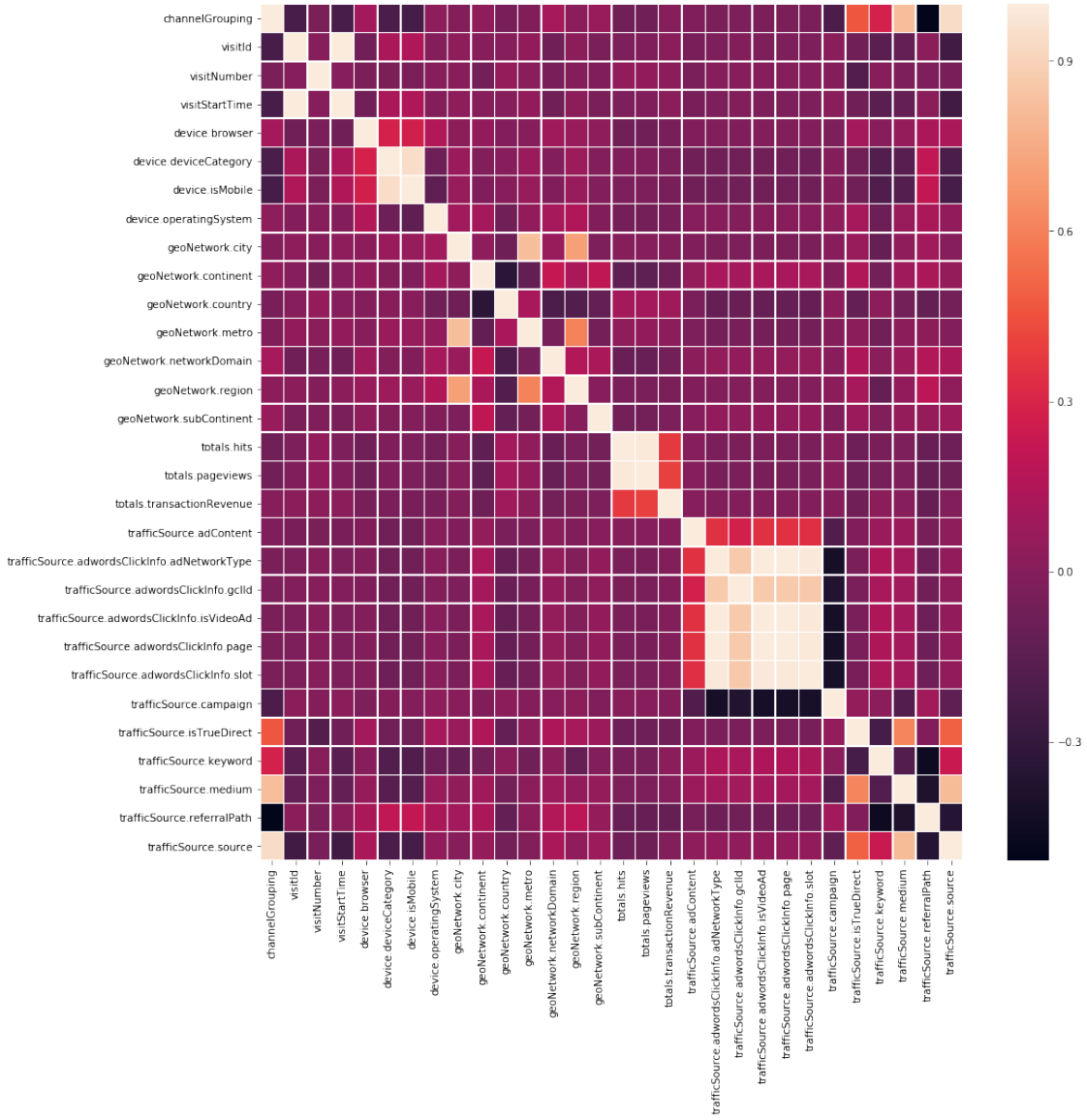
50%	5.000000	3196.000000	208.000000
75%	6.000000	3196.000000	417.000000
max	6.000000	3196.000000	499.000000

1.0.3 Coorelation and HeatMap

We are generating heatmap by taking the corelation between multiple parameters provided to us

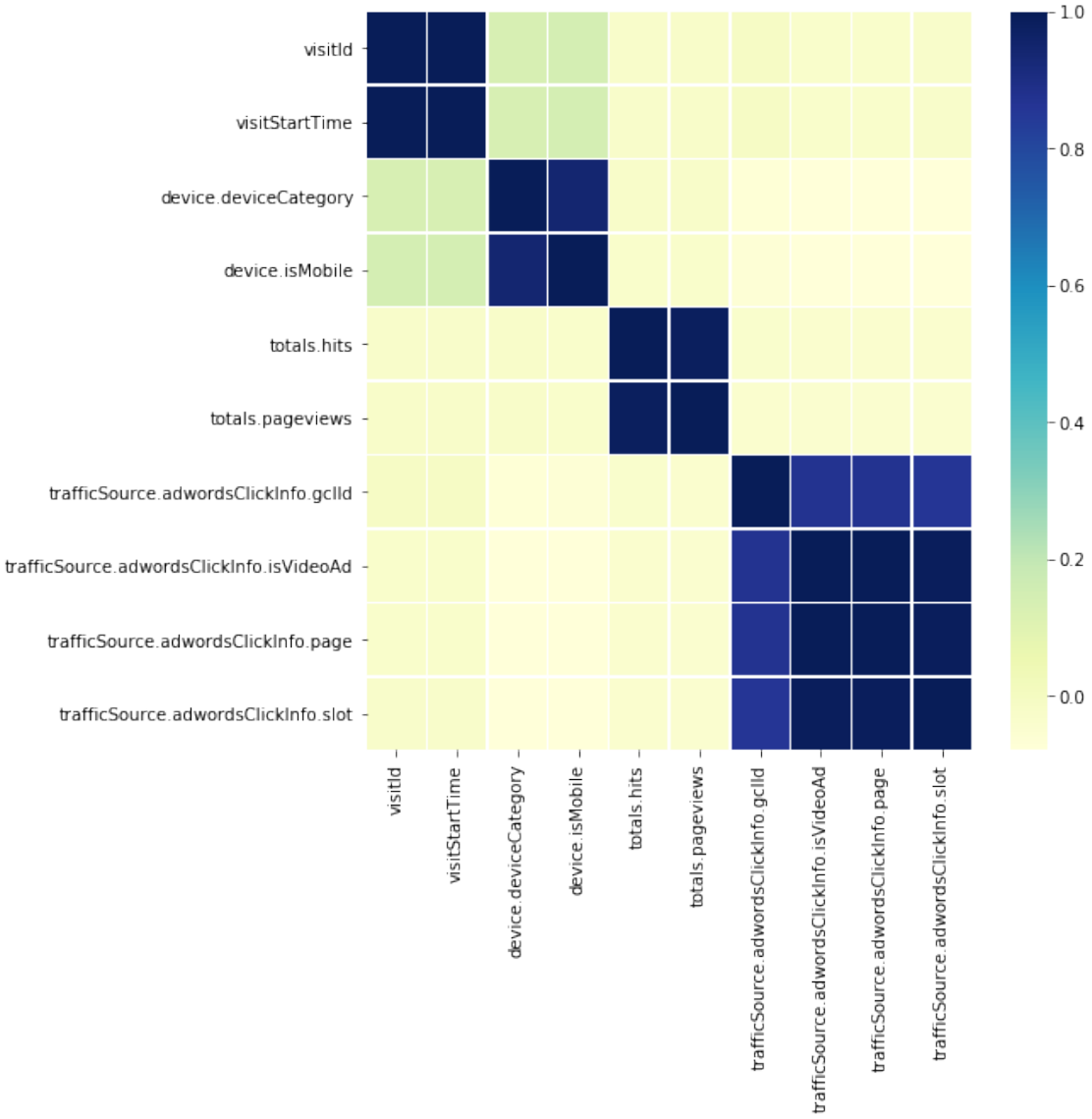
In [25]:
```
#Generating Coorelation And HeatMap

correlation_df = train_df[[i for i in list(train_df.columns) if i not in ['totals.bounc
corr = correlation_df.corr()
fig, ax = plt.subplots(figsize=(15, 15))
sb.heatmap(corr,  linewidths=.5, annot=False)

#Reference: https://seaborn.pydata.org/generated/seaborn.heatmap.html
```

Out[25]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6c54cf0da0>
```



In [26]:
```
correlation_df1 = train_df[[i for i in list(train_df.columns) if i in ['visitId', 'visi
corr1 = correlation_df1.corr()
fig, ax = plt.subplots(figsize=(8, 8))
sb.heatmap(corr1,  linewidths=.5, cmap="YlGnBu", annot=False)
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f6c54a10c18>



- The above heatmap is generated based on a few selected features which are having high coorelation value amongst the features

1.0.4 External Data Set

I have taken data from the source: https://www.kaggle.com/satian/exported-google-analytics-data. We are trying to see if we can get meaningful datasets from the given dataset. Using the external dataset, we can enhance our prediction model and we can improve our results which will help us improve the efficiency of our result

```
In [27]: train_copy_df = train_df
         test_copy_df = test_df
```

```
In [28]: external_train_df =  pd.read_csv('./ExternalData/Train_external_data.csv', dtype = {"Cl
         external_train_df.head()
```

Out[28]:

	Client Id	Sessions	Avg. Session Duration	Bounce Rate	Revenue	\
0	1071704239	2	0:03:10	0.00%	$234.16	
1	1073240909	2	0:03:55	0.00%	$234.16	
2	1073518268	4	0:09:36	0.00%	$234.16	
3	1073673252	4	0:03:05	0.00%	$234.16	
4	1074548295	2	0:05:30	0.00%	$234.16	

	Transactions	Goal Conversion Rate
0	2	400.00%
1	2	300.00%
2	2	250.00%
3	2	200.00%
4	2	300.00%

```
In [29]: external_test_df =  pd.read_csv('./ExternalData/Test_external_data.csv', dtype = {"Clie
         external_test_df.head()
```

Out[29]:

	Client Id	Sessions	Avg. Session Duration	Bounce Rate	Revenue	\
0	1216956233	120	0:17:48	1.67%	$54,632.64	
1	907102415.2	4	0:14:40	50.00%	$18,615.84	
2	833580000.2	55	0:07:04	34.55%	$13,954.70	
3	595590681.2	18	0:10:28	11.11%	$11,391.00	
4	72230301.15	32	0:07:35	0.00%	$7,238.45	

	Transactions	Goal Conversion Rate
0	11	65.00%
1	4	175.00%
2	11	61.82%
3	2	144.44%
4	5	100.00%

Data Cleaning of the External Data set and merging them

```
In [30]: for df in [external_train_df,external_test_df]:
             df["visitId"] = df["Client Id"].apply(lambda x: x.split('.', 1)[0]).astype(str)

         categorical_columns = ['Revenue', 'Sessions', 'Avg. Session Duration', 'Bounce Rate', '

         for column in categorical_columns:
             lbl = preprocessing.LabelEncoder()
             lbl.fit(list(external_train_df[column].values.astype('str')) + list(external_test_d
             external_train_df[column] = lbl.transform(list(external_train_df[column].values.ast
             external_test_df[column] = lbl.transform(list(external_test_df[column].values.astyp

         external_train_df = external_train_df.merge(external_train_df, how="left", on="visitId"
         external_test_df = external_test_df.merge(external_test_df, how="left", on="visitId")
```

```
for df in [external_train_df,external_test_df]:
    df.drop("Client Id", axis=1, inplace=True)
```

1.1 Baseline Model - Light GDM model

```
In [31]: import lightgbm as lgb
         params = {
                 "objective" : "regression",
                 "metric" : "rmse",
                 "num_leaves" : 30,
                 "min_child_samples" : 100,
                 "learning_rate" : 0.1,
                 "bagging_fraction" : 0.7,
                 "feature_fraction" : 0.5,
                 "bagging_frequency" : 5,
                 "bagging_seed" : 2018,
                 "verbosity" : -1
             }

In [32]: #Remove transactionRev from train_df. That will be train_y
         final_train_y = train_df["totals.transactionRevenue"]

         del train_df["totals.transactionRevenue"]
         final_train_df = train_df

In [33]: val_X = final_train_df[categorical_columns + numerical_columns]
         val_y = final_train_y

In [34]: test_X = test_df[categorical_columns + numerical_columns]

In [35]: date_data = final_train_df["date"]
         fullVisitorId_data = final_train_df["fullVisitorId"]

In [36]: del final_train_df["date"]
         del final_train_df["fullVisitorId"]

In [37]: final_train_df.dtypes

Out[37]: channelGrouping                    int64
         visitId                            int64
         visitNumber                      float64
         visitStartTime                   float64
         device.browser                     int64
         device.deviceCategory              int64
         device.isMobile                     bool
         device.operatingSystem             int64
         geoNetwork.city                    int64
         geoNetwork.continent               int64
```

```
geoNetwork.country                                  int64
geoNetwork.metro                                    int64
geoNetwork.networkDomain                            int64
geoNetwork.region                                   int64
geoNetwork.subContinent                             int64
totals.bounces                                    float64
totals.hits                                       float64
totals.newVisits                                  float64
totals.pageviews                                  float64
trafficSource.adContent                             int64
trafficSource.adwordsClickInfo.adNetworkType        int64
trafficSource.adwordsClickInfo.gclId                int64
trafficSource.adwordsClickInfo.isVideoAd            int64
trafficSource.adwordsClickInfo.page                 int64
trafficSource.adwordsClickInfo.slot                 int64
trafficSource.campaign                              int64
trafficSource.isTrueDirect                          int64
trafficSource.keyword                               int64
trafficSource.medium                                int64
trafficSource.referralPath                          int64
trafficSource.source                                int64
dtype: object
```

In [38]:
```
test_X.dtypes
```

Out[38]:
```
channelGrouping                                     int64
device.browser                                      int64
device.deviceCategory                               int64
device.operatingSystem                              int64
geoNetwork.city                                     int64
geoNetwork.continent                                int64
geoNetwork.country                                  int64
geoNetwork.metro                                    int64
geoNetwork.networkDomain                            int64
geoNetwork.region                                   int64
geoNetwork.subContinent                             int64
trafficSource.adContent                             int64
trafficSource.adwordsClickInfo.adNetworkType        int64
trafficSource.adwordsClickInfo.gclId                int64
trafficSource.adwordsClickInfo.isVideoAd            int64
trafficSource.adwordsClickInfo.page                 int64
trafficSource.adwordsClickInfo.slot                 int64
trafficSource.campaign                              int64
trafficSource.isTrueDirect                          int64
trafficSource.keyword                               int64
trafficSource.medium                                int64
trafficSource.referralPath                          int64
trafficSource.source                                int64
```

```
totals.bounces                                    float64
totals.hits                                       float64
totals.newVisits                                  float64
totals.pageviews                                  float64
visitNumber                                       float64
visitStartTime                                    float64
dtype: object
```

```
In [39]: print("Difference:", set(final_train_df.columns).difference(set(test_X.columns)))
         del final_train_df["visitId"]
         del final_train_df["device.isMobile"]
```

```
Difference: {'visitId', 'device.isMobile'}
```

```
In [40]: def data_train(train_X, train_y, val_X, val_y, test_X):
             lgtrain = lgb.Dataset(train_X, label=train_y)
             lgval = lgb.Dataset(val_X, label=val_y)

             model = lgb.train(params, lgtrain, 1000, valid_sets=[lgval], early_stopping_rounds=

             pred_test_y = model.predict(test_X, num_iteration=model.best_iteration)
             pred_val_y = model.predict(val_X, num_iteration=model.best_iteration)
             return pred_test_y, model, pred_val_y

         pred_test, model, pred_val = data_train(final_train_df, final_train_y, val_X, val_y, te

         #Reference: https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-li
```

```
Training until validation scores don't improve for 100 rounds.
[100]        valid_0's rmse: 2.01569
Early stopping, best iteration is:
[1]        valid_0's rmse: 2.00379
```

```
In [42]: #Finding the Mean Squared Error

         pred_val[pred_val<0] = 0

         val_pred_df = pd.DataFrame({"fullVisitorId":fullVisitorId_data.values})
         val_pred_df["transactionRevenue"] = final_train_y.values
         val_pred_df["PredictedRevenue"] = np.expm1(pred_val)
         val_pred_df = val_pred_df.groupby("fullVisitorId")["transactionRevenue", "PredictedReve

         print(np.sqrt(metrics.mean_squared_error(np.log1p(val_pred_df["transactionRevenue"].val
```

```
0.3964828805195681
```

```
In [43]: test_id = test_df["fullVisitorId"].values
```

```
In [44]: sub_df = pd.DataFrame({"fullVisitorId":test_id})
         pred_test[pred_test<0] = 0

         sub_df["PredictedLogRevenue"] = np.expm1(pred_test)
         sub_df = sub_df.groupby("fullVisitorId")["PredictedLogRevenue"].sum().reset_index()
         sub_df.columns = ["fullVisitorId", "PredictedLogRevenue"]

         sub_df["PredictedLogRevenue"] = np.log1p(sub_df["PredictedLogRevenue"])
         sub_df.to_csv("result.csv", index=False)
```

Predicting of Buying with Probablility Funciton using Logistic Regression

```
In [154]: from sklearn.linear_model import LogisticRegression

          logistic_df = final_train_df
          logistic_columns = ['totals.hits','totals.newVisits', 'totals.pageviews', 'totals.boun
          logistic_df[logistic_columns] = logistic_df[logistic_columns].fillna(0.0).astype(int)
          #logistic_df.dtypes

          temp = final_train_y > 0.0
          temp = temp.astype(int)
          new_df = pd.DataFrame()
          new_df['target_value'] = temp


          feature_vars = numerical_columns + categorical_columns
          log_features = logistic_df[feature_vars].drop(['geoNetwork.region','trafficSource.adwo
                                                         'trafficSource.campaign','trafficSource.keyword
                                                         'trafficSource.adwordsClickInfo.gclId'], axis=1

          feature_vars_1 = ['totals.bounces', 'totals.hits', 'totals.newVisits',
                 'totals.pageviews', 'visitNumber', 'visitStartTime',
                 'channelGrouping', 'device.browser', 'device.deviceCategory',
                 'device.operatingSystem', 'geoNetwork.city',
                 'geoNetwork.continent', 'geoNetwork.country', 'geoNetwork.metro',
                 'geoNetwork.networkDomain', 'geoNetwork.subContinent',
                 'trafficSource.adContent',
                 'trafficSource.adwordsClickInfo.adNetworkType',
                 'trafficSource.adwordsClickInfo.isVideoAd',
                 'trafficSource.adwordsClickInfo.page',
                 'trafficSource.isTrueDirect', 'trafficSource.medium',
                 'trafficSource.referralPath', 'trafficSource.source']

          clf = LogisticRegression(random_state=0, solver='lbfgs', multi_class='multinomial').fi
          clf.predict_proba(logistic_df[feature_vars_1])[:,0]
```

```
/home/jaytorasakar8/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:578: DataC
  y = column_or_1d(y, warn=True)
```

```
Out[154]: array([0.89631606, 0.89632283, 0.89632082, ..., 0.89776278, 0.89777171,
                 0.89776924])

In [155]: regression_df = train_df
          regression_features = list(regression_df.columns.values)

          regression_df = regression_df[regression_features].drop(['geoNetwork.region','trafficS

In [156]: regression_df['probs'] = clf.predict_proba(log_features)[:,0]
          regression_df = regression_df.sort_values(by='probs', ascending=False)

In [158]: regression_df.head(10)

Out[158]:       channelGrouping  visitNumber  visitStartTime  device.browser  \
          64223                4          1.0    1.501657e+09              72
          65294                7          1.0    1.501657e+09              72
          65054                4          3.0    1.501657e+09              47
          64767                4          1.0    1.501657e+09              35
          64597                1          1.0    1.501657e+09              35
          62993                4          1.0    1.501657e+09              35
          65301                2          1.0    1.501657e+09              35
          65085                7          1.0    1.501657e+09              47
          64403                6          1.0    1.501657e+09              35
          64680                2          1.0    1.501657e+09              35

                 device.deviceCategory  device.operatingSystem  geoNetwork.city  \
          64223                      2                      23              458
          65294                      1                      23               70
          65054                      0                      20              458
          64767                      0                      20              955
          64597                      0                      20              955
          62993                      0                      20              955
          65301                      0                      20              179
          65085                      0                      20              955
          64403                      0                       7              341
          64680                      0                      20               91

                 geoNetwork.continent  geoNetwork.country  geoNetwork.metro  \
          64223                     4                 217                60
          65294                     3                 204                 0
          65054                     4                 217                60
          64767                     3                  43               122
          64597                     3                  99               122
          62993                     3                  52               122
          65301                     3                  93                 0
          65085                     3                 160               122
          64403                     4                  75                 0
          64680                     3                  93                 0
```

```
       geoNetwork.networkDomain  geoNetwork.subContinent  totals.bounces  \
64223                     39737                       13               1
65294                      2774                       16               1
65054                     19919                       13               1
64767                        48                        6               0
64597                      3329                       21               1
62993                      8936                       21               0
65301                     38725                       18               1
65085                      8379                       16               1
64403                         0                       22               1
64680                     38725                       18               1

       totals.hits  totals.newVisits  totals.pageviews  \
64223            1                 1                 1
65294            1                 1                 1
65054            1                 0                 1
64767            2                 1                 2
64597            1                 1                 1
62993            4                 1                 4
65301            1                 1                 1
65085            1                 1                 1
64403            1                 1                 1
64680            1                 1                 1

       trafficSource.adContent  trafficSource.adwordsClickInfo.adNetworkType  \
64223                       62                                             3
65294                       62                                             3
65054                       62                                             3
64767                       62                                             3
64597                       62                                             3
62993                       62                                             3
65301                       62                                             3
65085                       62                                             3
64403                       62                                             3
64680                       62                                             3

       trafficSource.adwordsClickInfo.isVideoAd  \
64223                                         1
65294                                         1
65054                                         1
64767                                         1
64597                                         1
62993                                         1
65301                                         1
65085                                         1
64403                                         1
64680                                         1
```

```
       trafficSource.adwordsClickInfo.page  trafficSource.isTrueDirect  \
64223                                   11                           1
65294                                   11                           1
65054                                   11                           0
64767                                   11                           1
64597                                   11                           1
62993                                   11                           1
65301                                   11                           0
65085                                   11                           1
64403                                   11                           1
64680                                   11                           0

       trafficSource.medium  trafficSource.referralPath  trafficSource.source  \
64223                     5                        3196                   208
65294                     6                        1840                   497
65054                     5                        3196                   208
64767                     5                        3196                   208
64597                     2                        3196                    81
62993                     5                        3196                   208
65301                     0                        3196                     0
65085                     6                        2589                   497
64403                     6                        2198                   352
64680                     0                        3196                     0

          probs
64223  0.900174
65294  0.900174
65054  0.900174
64767  0.900174
64597  0.900174
62993  0.900174
65301  0.900174
65085  0.900174
64403  0.900174
64680  0.900174
```

1.1.1 Using another Model - Random Forest for prediction

In [159]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split

permutation_df = final_train_df

train_x, test_x, train_y, test_y = train_test_split(permutation_df, final_train_y, tes
rf = RandomForestRegressor(n_estimators = 10)
model = rf.fit(train_x, train_y)
```

In [169]:
```
y_pred = model.predict(train_x)
```

- We don't see any improvement in the results as compared to LGDM model, so didn't advance further

Permutation Test p-values We are doing the Permutation Test, and it is done inorder to see the effects of the data shuffle on the final RMSE value

```
In [160]: import eli5
          from eli5.sklearn import PermutationImportance

          perm = PermutationImportance(rf).fit(test_x, test_y)
          eli5.show_weights(perm, feature_names = test_x.columns.tolist())

          #Reference: https://eli5.readthedocs.io/en/latest/blackbox/permutation_importance.html
```

```
Out[160]: <IPython.core.display.HTML object>
```

From above we can see that the feature of pageviews is the most important one in this given dataset