In [2]:
```
import lightgbm
import pandas as pd
import numpy as np
import time
import lightgbm as lgb
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import KFold
import matplotlib.pyplot as plt
import seaborn as sns

import os
import json
import numpy as np
import pandas as pd
from pandas.io.json import json_normalize

train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
```

```
/anaconda3/lib/python3.6/site-packages/IPython/core/interactiveshell.p
y:2785: DtypeWarning: Columns (3) have mixed types. Specify dtype optio
n on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

In [3]:
```
columns_to_normalize = ['device', 'geoNetwork','totals', 'trafficSource'
]
def normalize_json_data(filename):
    path = filename
    df = pd.read_csv(path, converters={column: json.loads for column in
columns_to_normalize},
                     dtype={'fullVisitorId': 'str'})

    for column in columns_to_normalize:
        column_as_df = json_normalize(df[column])
        column_as_df.columns = [f"{column}_{subcolumn}" for subcolumn in
 column_as_df.columns]
        df = df.drop(column, axis=1).merge(column_as_df, right_index=Tru
e, left_index=True)
    return df

train = normalize_json_data("train.csv")
test = normalize_json_data("test.csv")
```

In [5]:
```
train.shape
```

Out[5]: (903653, 55)

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

Out[6]:

	channelGrouping	date	fullVisitorId	sessionId
0	Organic Search	20160902	1131660440785968503	1131660440785968503_1472830385
1	Organic Search	20160902	377306020877927890	377306020877927890_1472880147
2	Organic Search	20160902	3895546263509774583	3895546263509774583_1472865386
3	Organic Search	20160902	4763447161404445595	4763447161404445595_1472881213
4	Organic Search	20160902	27294437909732085	27294437909732085_1472822600

5 rows × 55 columns

In [7]:
```
train_numerical_features = train.select_dtypes(include=[np.number])
```

In [8]:
```
test_numerical_features = test.select_dtypes(include=[np.number])
```

In [9]:
```
train_category_features = train.select_dtypes(include=[np.object])
test_category_features = test.select_dtypes(include=[np.object])
train_category_features.columns
test_category_features.columns
```

Out[9]:
```
Index(['channelGrouping', 'fullVisitorId', 'sessionId', 'socialEngageme
ntType',
       'device_browser', 'device_browserSize', 'device_browserVersion',
       'device_deviceCategory', 'device_flashVersion', 'device_languag
e',
       'device_mobileDeviceBranding', 'device_mobileDeviceInfo',
       'device_mobileDeviceMarketingName', 'device_mobileDeviceModel',
       'device_mobileInputSelector', 'device_operatingSystem',
       'device_operatingSystemVersion', 'device_screenColors',
       'device_screenResolution', 'geoNetwork_city', 'geoNetwork_cityI
d',
       'geoNetwork_continent', 'geoNetwork_country', 'geoNetwork_latitu
de',
       'geoNetwork_longitude', 'geoNetwork_metro', 'geoNetwork_networkD
omain',
       'geoNetwork_networkLocation', 'geoNetwork_region',
       'geoNetwork_subContinent', 'totals_bounces', 'totals_hits',
       'totals_newVisits', 'totals_pageviews', 'totals_visits',
       'trafficSource_adContent',
       'trafficSource_adwordsClickInfo.adNetworkType',
       'trafficSource_adwordsClickInfo.criteriaParameters',
       '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')
```

In [10]:
```
train = train.loc[:, (train != train.iloc[0]).any()]
test = test.loc[:, (test != test.iloc[0]).any()]
```

In [11]:
```
train["totals_transactionRevenue"] = train["totals_transactionRevenue"].
astype('float')
```

In [12]:
```
for df in [train, test]:
    df['v_date'] = pd.to_datetime(df['visitStartTime'], unit='s')
    df['dayofweek'] = df['v_date'].dt.dayofweek
    df['hours'] = df['v_date'].dt.hour
    df['day'] = df['v_date'].dt.day
    df.drop('visitStartTime', axis=1)
```

In [13]:
```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

agg_dict = {}
for col in ["totals_bounces", "totals_hits", "totals_newVisits", "totals
_pageviews", "totals_transactionRevenue"]:
    train[col] = train[col].astype('float')
    agg_dict[col] = "sum"
tmp = train.groupby("fullVisitorId").agg(agg_dict).reset_index()
tmp.head()
```

Out[13]:

	fullVisitorId	totals_bounces	totals_hits	totals_newVisits	totals_pageviews
0	0000010278554503158	0.0	11.0	1.0	8.0
1	0000020424342248747	0.0	17.0	1.0	13.0
2	0000027376579751715	0.0	6.0	1.0	5.0
3	0000039460501403861	0.0	2.0	1.0	2.0
4	0000040862739425590	0.0	5.0	1.0	5.0

In [14]:
```
constant_columns = []
for column in train.columns:
    if len(train[column].value_counts()) == 1:
        constant_columns.append(column)

irrelevant_columns = ["visitNumber", "date", "fullVisitorId", "sessionI
d", "visitId", "visitStartTime", "v_date",
                      "month", "weekday"]
```

In [15]:
```
colms = irrelevant_columns + constant_columns
```

In [16]:
```
from sklearn.preprocessing import LabelEncoder

category_columns = [c for c in train.columns if not c.startswith("total"
)]
category_columns = [c for c in category_columns if c not in colms]

#print(category_columns)

for c in category_columns:

    labelencode = LabelEncoder()
    train_vals = list(train[c].values.astype(str))
    test_vals = list(test[c].values.astype(str))

    labelencode.fit(train_vals + test_vals)

    train[c] = labelencode.transform(train_vals)
    test[c] = labelencode.transform(test_vals)
```

In [17]:
```
def normalize_numerical_columns(df, isTrain = True):
    df["totals_hits"] = df["totals_hits"].astype(float)
    df["totals_hits"] = (df["totals_hits"] - min(df["totals_hits"])) / (
max(df["totals_hits"]) - min(df["totals_hits"]))

    df["totals_pageviews"] = df["totals_pageviews"].astype(float)
    df["totals_pageviews"] = (df["totals_pageviews"] - min(df["totals_pa
geviews"])) / (max(df["totals_pageviews"]) - min(df["totals_pageviews"
]))

    if isTrain:
        df["totals_transactionRevenue"] = df["totals_transactionRevenue"
].fillna(0.0)
    return df
```

In [18]:
```
train = normalize_numerical_columns(train)
test = normalize_numerical_columns(test, isTrain = False)
```

In [19]:
```
from sklearn.model_selection import train_test_split
features = [c for c in train.columns if c not in colms]
features.remove("totals_transactionRevenue")
train["totals_transactionRevenue"] = np.log1p(train["totals_transactionR
evenue"].astype(float))
```

In [20]:
```
train.head()
```

Out[20]:

	channelGrouping	date	fullVisitorId	sessionId
0	4	20160902	1131660440785968503	1131660440785968503_1472830385
1	4	20160902	377306020877927890	377306020877927890_1472880147
2	4	20160902	3895546263509774583	3895546263509774583_1472865386
3	4	20160902	4763447161404445595	4763447161404445595_1472881213
4	4	20160902	27294437909732085	27294437909732085_1472822600

5 rows × 40 columns

In [21]:
```
print(train.corrwith(train['totals_transactionRevenue']))
```

```
channelGrouping                                 -0.000807
date                                             0.007897
visitId                                          0.010491
visitNumber                                      0.023666
visitStartTime                                   0.010491
device_browser                                  -0.046152
device_deviceCategory                           -0.042843
device_isMobile                                 -0.046071
device_operatingSystem                          -0.045766
geoNetwork_city                                 -0.013983
geoNetwork_continent                            -0.088376
geoNetwork_country                               0.080750
geoNetwork_metro                                 0.015912
geoNetwork_networkDomain                        -0.065680
geoNetwork_region                               -0.032555
geoNetwork_subContinent                         -0.031715
totals_bounces                                        NaN
totals_hits                                      0.378804
totals_newVisits                                      NaN
totals_pageviews                                 0.400732
totals_transactionRevenue                        1.000000
trafficSource_adContent                          0.001028
trafficSource_adwordsClickInfo.adNetworkType    -0.011333
trafficSource_adwordsClickInfo.gclId            -0.009010
trafficSource_adwordsClickInfo.page             -0.011369
trafficSource_adwordsClickInfo.slot             -0.010855
trafficSource_campaign                          -0.007297
trafficSource_keyword                            0.024183
trafficSource_medium                             0.001241
trafficSource_referralPath                      -0.112841
trafficSource_source                            -0.009591
dayofweek                                       -0.015258
hours                                           -0.008287
day                                             -0.005075
dtype: float64
```

In [22]:
```
train_x, valid_x, train_y, valid_y = train_test_split(train[features], t
rain["totals_transactionRevenue"],
                                      test_size=0.25, random_state=20)
```

In [23]:
```
import lightgbm as lgb

lgb_params = {"objective" : "regression", "metric" : "rmse",
              "num_leaves" : 50, "learning_rate" : 0.02,
              "bagging_fraction" : 0.75, "feature_fraction" : 0.8, "bagg
ing_frequency" : 9}

lgb_train = lgb.Dataset(train_x, label=train_y)
lgb_val = lgb.Dataset(valid_x, label=valid_y)
model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val], earl
y_stopping_rounds=250, verbose_eval=100)
```

```
Training until validation scores don't improve for 250 rounds.
[100]   valid_0's rmse: 1.68274
[200]   valid_0's rmse: 1.65902
[300]   valid_0's rmse: 1.65559
[400]   valid_0's rmse: 1.65468
[500]   valid_0's rmse: 1.65422
[600]   valid_0's rmse: 1.65415
[700]   valid_0's rmse: 1.65447
Did not meet early stopping. Best iteration is:
[584]   valid_0's rmse: 1.65401
```

In [24]:
```
model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val], earl
y_stopping_rounds=250, verbose_eval=100)
```

```
Training until validation scores don't improve for 250 rounds.
[100]   valid_0's rmse: 1.68274
[200]   valid_0's rmse: 1.65902
[300]   valid_0's rmse: 1.65559
[400]   valid_0's rmse: 1.65468
[500]   valid_0's rmse: 1.65422
[600]   valid_0's rmse: 1.65415
[700]   valid_0's rmse: 1.65447
Did not meet early stopping. Best iteration is:
[584]   valid_0's rmse: 1.65401
```

In [25]:
```
test_prediction = model.predict(test[features], num_iteration=model.best
_iteration)
test["PredictedLogRevenue"] = np.expm1(test_prediction)
submission = test.groupby("fullVisitorId").agg({"PredictedLogRevenue" :
"sum"}).reset_index()
submission["PredictedLogRevenue"] = np.log1p(submission["PredictedLogRev
enue"])
submission["PredictedLogRevenue"] =  submission["PredictedLogRevenue"].a
pply(lambda x : 0.0 if x < 0 else x)
submission.to_csv("submission6.csv", index=False)
submission.head()
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:4: Runtime
Warning: invalid value encountered in log1p
  after removing the cwd from sys.path.
```

Out[25]:

	fullVisitorId	PredictedLogRevenue
0	0000000259678714014	0.127458
1	0000049363351866189	0.000000
2	0000053049821714864	0.000000
3	0000059488412965267	0.000000
4	0000085840370633780	0.017774

In [26]:
```
train_x, valid_x, train_y, valid_y = train_test_split(train[features], t
rain["totals_transactionRevenue"],
                                      test_size=0.25, random_state=20)

def permutation_test(train_X, train_y, val_X, val_y):
    params = {
        "objective" : "regression",
        "metric" : "rmse",
        "num_leaves" : 50,
        "min_child_samples" : 100,
        "learning_rate" : 0.1,
        "bagging_fraction" : 0.7,
        "feature_fraction" : 0.5,
        "bagging_frequency" : 5,
        "bagging_seed" : 2018,
        "verbosity" : -1,
        "njobs" : 3
    }

    lgtrain = lgb.Dataset(train_x, label=train_y)
    lgval = lgb.Dataset(valid_x, label=valid_y)
    model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_
stopping_rounds=1000, verbose_eval=500)

#     pred_test_y = model.predict(test_X, num_iteration=model.best_itera
tion)
    pred_val_y = model.predict(valid_X, num_iteration=model.best_iterati
on)
    return model, pred_val_y
```

In [27]:
```
train.columns
```

Out[27]:
```
Index(['channelGrouping', 'date', 'fullVisitorId', 'sessionId', 'visitI
d',
       'visitNumber', 'visitStartTime', 'device_browser',
       'device_deviceCategory', 'device_isMobile', 'device_operatingSys
tem',
       'geoNetwork_city', 'geoNetwork_continent', 'geoNetwork_country',
       'geoNetwork_metro', 'geoNetwork_networkDomain', 'geoNetwork_regi
on',
       'geoNetwork_subContinent', 'totals_bounces', 'totals_hits',
       'totals_newVisits', 'totals_pageviews', 'totals_transactionReven
ue',
       'trafficSource_adContent',
       'trafficSource_adwordsClickInfo.adNetworkType',
       'trafficSource_adwordsClickInfo.gclId',
       'trafficSource_adwordsClickInfo.isVideoAd',
       'trafficSource_adwordsClickInfo.page',
       'trafficSource_adwordsClickInfo.slot', 'trafficSource_campaign',
       'trafficSource_campaignCode', 'trafficSource_isTrueDirect',
       'trafficSource_keyword', 'trafficSource_medium',
       'trafficSource_referralPath', 'trafficSource_source', 'v_date',
       'dayofweek', 'hours', 'day'],
      dtype='object')
```

In [38]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[cat_cols + num_cols]
        val_X = val_df[cat_cols + num_cols]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews   0

---------------------------------------------------------------------------
----
NameError                                 Traceback (most recent call l
ast)
<ipython-input-38-a0e2f9579d31> in <module>()
     11         val_y = np.log1p(val_df["totals_transactionRevenue"].va
lues)
     12
---> 13         dev_X = dev_df[cat_cols + num_cols]
     14         val_X = val_df[cat_cols + num_cols]
     15         model, pred_val = permutation_test(dev_X, dev_y, val_X,
 val_y)

NameError: name 'cat_cols' is not defined
```

In [42]:
```
train.columns
```

Out[42]:
```
Index(['channelGrouping', 'date', 'fullVisitorId', 'sessionId', 'visitI
d',
       'visitNumber', 'visitStartTime', 'device_browser',
       'device_deviceCategory', 'device_isMobile', 'device_operatingSys
tem',
       'geoNetwork_city', 'geoNetwork_continent', 'geoNetwork_country',
       'geoNetwork_metro', 'geoNetwork_networkDomain', 'geoNetwork_regi
on',
       'geoNetwork_subContinent', 'totals_bounces', 'totals_hits',
       'totals_newVisits', 'totals_pageviews', 'totals_transactionReven
ue',
       'trafficSource_adContent',
       'trafficSource_adwordsClickInfo.adNetworkType',
       'trafficSource_adwordsClickInfo.gclId',
       'trafficSource_adwordsClickInfo.isVideoAd',
       'trafficSource_adwordsClickInfo.page',
       'trafficSource_adwordsClickInfo.slot', 'trafficSource_campaign',
       'trafficSource_campaignCode', 'trafficSource_isTrueDirect',
       'trafficSource_keyword', 'trafficSource_medium',
       'trafficSource_referralPath', 'trafficSource_source', 'v_date',
       'dayofweek', 'hours', 'day'],
      dtype='object')
```

In [43]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 903653 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 903653 non-null int64
date                                            903653 non-null int64
fullVisitorId                                   903653 non-null object
sessionId                                       903653 non-null object
visitId                                         903653 non-null int64
visitNumber                                     903653 non-null int64
visitStartTime                                  903653 non-null int64
device_browser                                  903653 non-null int64
device_deviceCategory                           903653 non-null int64
device_isMobile                                 903653 non-null int64
device_operatingSystem                          903653 non-null int64
geoNetwork_city                                 903653 non-null int64
geoNetwork_continent                            903653 non-null int64
geoNetwork_country                              903653 non-null int64
geoNetwork_metro                                903653 non-null int64
geoNetwork_networkDomain                        903653 non-null int64
geoNetwork_region                               903653 non-null int64
geoNetwork_subContinent                         903653 non-null int64
totals_bounces                                  450630 non-null float64
totals_hits                                     903653 non-null float64
totals_newVisits                                703060 non-null float64
totals_pageviews                                903553 non-null float64
totals_transactionRevenue                       903653 non-null float64
trafficSource_adContent                         903653 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    903653 non-null int64
trafficSource_adwordsClickInfo.gclId            903653 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        21460 non-null object
trafficSource_adwordsClickInfo.page             903653 non-null int64
trafficSource_adwordsClickInfo.slot             903653 non-null int64
trafficSource_campaign                          903653 non-null int64
trafficSource_campaignCode                      1 non-null object
trafficSource_isTrueDirect                      274005 non-null object
trafficSource_keyword                           903653 non-null int64
trafficSource_medium                            903653 non-null int64
trafficSource_referralPath                      903653 non-null int64
trafficSource_source                            903653 non-null int64
v_date                                          903653 non-null datetim
e64[ns]
dayofweek                                       903653 non-null int64
hours                                           903653 non-null int64
day                                             903653 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 275.8+ MB
```

In [45]:
```
numerical_colums = ["totals_hits", "totals_bounces", "totals_pageviews",
 "totals_newVisits", "visitNumber"]
```

In [46]:
```
category_columns
```

Out[46]:
```
['channelGrouping',
 'device_browser',
 'device_deviceCategory',
 'device_isMobile',
 '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.page',
 'trafficSource_adwordsClickInfo.slot',
 'trafficSource_campaign',
 'trafficSource_keyword',
 'trafficSource_medium',
 'trafficSource_referralPath',
 'trafficSource_source',
 'dayofweek',
 'hours',
 'day']
```

In [47]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[category_columns + numerical_colums]
        val_X = val_df[category_columns + numerical_colums]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews    0
```

```
---------------------------------------------------------------------------
LightGBMError                             Traceback (most recent call last)
<ipython-input-47-ea3a519919fc> in <module>()
     13         dev_X = dev_df[category_columns + numerical_colums]
     14         val_X = val_df[category_columns + numerical_colums]
---> 15         model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)

<ipython-input-26-a9742b21269c> in permutation_test(train_X, train_y, val_X, val_y)
     19     lgtrain = lgb.Dataset(train_x, label=train_y)
     20     lgval = lgb.Dataset(valid_x, label=valid_y)
---> 21     model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_stopping_rounds=1000, verbose_eval=500)
     22
     23 #     pred_test_y = model.predict(test_X, num_iteration=model.best_iteration)

/anaconda3/lib/python3.6/site-packages/lightgbm/engine.py in train(params, train_set, num_boost_round, valid_sets, valid_names, fobj, feval, init_model, feature_name, categorical_feature, early_stopping_rounds, evals_result, verbose_eval, learning_rates, keep_training_booster, callbacks)
    190     # construct booster
    191     try:
--> 192         booster = Booster(params=params, train_set=train_set)
    193         if is_valid_contain_train:
    194             booster.set_train_data_name(train_data_name)

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in __init__(self, params, train_set, model_file, silent)
   1485             self.handle = ctypes.c_void_p()
   1486             _safe_call(_LIB.LGBM_BoosterCreate(
-> 1487                 train_set.construct().handle,
   1488                 c_str(params_str),
   1489                 ctypes.byref(self.handle)))

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in construct(self)
    983                                 init_score=self.init_score, predictor=self._predictor,
    984                                 silent=self.silent, feature_name=self.feature_name,
--> 985                                 categorical_feature=self.categorical_feature, params=self.params)
    986             if self.free_raw_data:
    987                 self.data = None

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _lazy_init(self, data, label, reference, weight, group, init_score, predictor, silent, feature_name, categorical_feature, params)
    779                 raise TypeError('Cannot initialize Dataset from {}'.format(type(data).__name__))
    780         if label is not None:
--> 781             self.set_label(label)
```

```
    782         if self.get_label() is None:
    783             raise ValueError("Label should not be None")

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_label(s
elf, label)
   1275         if self.handle is not None:
   1276             label = list_to_1d_numpy(_label_from_pandas(label),
 name='label')
-> 1277             self.set_field('label', label)
   1278         return self
   1279

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_field(s
elf, field_name, data)
   1122             ptr_data,
   1123             ctypes.c_int(len(data)),
-> 1124             ctypes.c_int(type_data)))
   1125         return self
   1126

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _safe_call
(ret)
     43     """
     44     if ret != 0:
---> 45         raise LightGBMError(decode_string(_LIB.LGBM_GetLastErro
r()))
     46
     47

LightGBMError: Length of label is not same with #data
```

In [50]:
```
model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_stop
ping_rounds=1000, verbose_eval=500)
```

```
---------------------------------------------------------------------------
----
NameError                                 Traceback (most recent call l
ast)
<ipython-input-50-9f26e1d6cbe8> in <module>()
----> 1 model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], e
arly_stopping_rounds=1000, verbose_eval=500)

NameError: name 'lgtrain' is not defined
```

In [49]:
```
    params = {
        "objective" : "regression",
        "metric" : "rmse",
        "num_leaves" : 50,
        "min_child_samples" : 100,
        "learning_rate" : 0.1,
        "bagging_fraction" : 0.7,
        "feature_fraction" : 0.5,
        "bagging_frequency" : 5,
        "bagging_seed" : 2018,
        "verbosity" : -1,
        "njobs" : 3
    }
```

In [51]:
```
lgb_params = {"objective" : "regression", "metric" : "rmse",
              "num_leaves" : 50, "learning_rate" : 0.02,
              "bagging_fraction" : 0.75, "feature_fraction" : 0.8, "bagg
ing_frequency" : 9}

lgb_train = lgb.Dataset(train_x, label=train_y)
lgb_val = lgb.Dataset(valid_x, label=valid_y)
model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val], earl
y_stopping_rounds=250, verbose_eval=100)
```

```
Training until validation scores don't improve for 250 rounds.
[100]   valid_0's rmse: 1.68274
[200]   valid_0's rmse: 1.65902
[300]   valid_0's rmse: 1.65559
[400]   valid_0's rmse: 1.65468
[500]   valid_0's rmse: 1.65422
[600]   valid_0's rmse: 1.65415
[700]   valid_0's rmse: 1.65447
Did not meet early stopping. Best iteration is:
[584]   valid_0's rmse: 1.65401
```

In [56]:
```
train_x, valid_x, train_y, valid_y = train_test_split(train[features], t
rain["totals_transactionRevenue"],
                                      test_size=0.25, random_state=20)

def permutation_test(train_X, train_y, val_X, val_y):
    params = {
        "objective" : "regression",
        "metric" : "rmse",
        "num_leaves" : 50,
        "min_child_samples" : 100,
        "learning_rate" : 0.1,
        "bagging_fraction" : 0.7,
        "feature_fraction" : 0.5,
        "bagging_frequency" : 5,
        "bagging_seed" : 2018,
        "verbosity" : -1,
        "njobs" : 3
    }

    lgtrain = lgb.Dataset(train_x, label=train_y)
    lgval = lgb.Dataset(valid_x, label=valid_y)
    model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_
stopping_rounds=250, verbose_eval=100)

#     pred_test_y = model.predict(test_X, num_iteration=model.best_itera
tion)
    pred_val_y = model.predict(valid_X, num_iteration=model.best_iterati
on)
    return model, pred_val_y
```

In [57]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[category_columns + numerical_colums]
        val_X = val_df[category_columns + numerical_colums]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews    0
```

```
---------------------------------------------------------------------------
LightGBMError                             Traceback (most recent call last)
<ipython-input-57-ea3a519919fc> in <module>()
     13         dev_X = dev_df[category_columns + numerical_colums]
     14         val_X = val_df[category_columns + numerical_colums]
---> 15         model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)

<ipython-input-56-865f3e4084aa> in permutation_test(train_X, train_y, val_X, val_y)
     19     lgtrain = lgb.Dataset(train_x, label=train_y)
     20     lgval = lgb.Dataset(valid_x, label=valid_y)
---> 21     model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_stopping_rounds=250, verbose_eval=100)
     22
     23 #     pred_test_y = model.predict(test_X, num_iteration=model.best_iteration)

/anaconda3/lib/python3.6/site-packages/lightgbm/engine.py in train(params, train_set, num_boost_round, valid_sets, valid_names, fobj, feval, init_model, feature_name, categorical_feature, early_stopping_rounds, evals_result, verbose_eval, learning_rates, keep_training_booster, callbacks)
    190     # construct booster
    191     try:
--> 192         booster = Booster(params=params, train_set=train_set)
    193         if is_valid_contain_train:
    194             booster.set_train_data_name(train_data_name)

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in __init__(self, params, train_set, model_file, silent)
   1485             self.handle = ctypes.c_void_p()
   1486             _safe_call(_LIB.LGBM_BoosterCreate(
-> 1487                 train_set.construct().handle,
   1488                 c_str(params_str),
   1489                 ctypes.byref(self.handle)))

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in construct(self)
    983                                 init_score=self.init_score, predictor=self._predictor,
    984                                 silent=self.silent, feature_name=self.feature_name,
--> 985                                 categorical_feature=self.categorical_feature, params=self.params)
    986             if self.free_raw_data:
    987                 self.data = None

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _lazy_init(self, data, label, reference, weight, group, init_score, predictor, silent, feature_name, categorical_feature, params)
    779                 raise TypeError('Cannot initialize Dataset from {}'.format(type(data).__name__))
    780         if label is not None:
--> 781             self.set_label(label)
```

```
    782         if self.get_label() is None:
    783             raise ValueError("Label should not be None")

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_label(s
elf, label)
   1275         if self.handle is not None:
   1276             label = list_to_1d_numpy(_label_from_pandas(label),
 name='label')
-> 1277             self.set_field('label', label)
   1278         return self
   1279

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_field(s
elf, field_name, data)
   1122             ptr_data,
   1123             ctypes.c_int(len(data)),
-> 1124             ctypes.c_int(type_data)))
   1125         return self
   1126

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _safe_call
(ret)
     43     """
     44     if ret != 0:
---> 45         raise LightGBMError(decode_string(_LIB.LGBM_GetLastErro
r()))
     46
     47

LightGBMError: Length of label is not same with #data
```

In [58]:
```
dev_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 765707 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 765707 non-null int64
date                                            765707 non-null int64
fullVisitorId                                   765707 non-null object
sessionId                                       765707 non-null object
visitId                                         765707 non-null int64
visitNumber                                     765707 non-null int64
visitStartTime                                  765707 non-null int64
device_browser                                  765707 non-null int64
device_deviceCategory                           765707 non-null int64
device_isMobile                                 765707 non-null int64
device_operatingSystem                          765707 non-null int64
geoNetwork_city                                 765707 non-null int64
geoNetwork_continent                            765707 non-null int64
geoNetwork_country                              765707 non-null int64
geoNetwork_metro                                765707 non-null int64
geoNetwork_networkDomain                        765707 non-null int64
geoNetwork_region                               765707 non-null int64
geoNetwork_subContinent                         765707 non-null int64
totals_bounces                                  381281 non-null float64
totals_hits                                     765707 non-null float64
totals_newVisits                                598244 non-null float64
totals_pageviews                                765616 non-null float64
totals_transactionRevenue                       765707 non-null float64
trafficSource_adContent                         765707 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    765707 non-null int64
trafficSource_adwordsClickInfo.gclId            765707 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        16421 non-null object
trafficSource_adwordsClickInfo.page             765707 non-null int64
trafficSource_adwordsClickInfo.slot             765707 non-null int64
trafficSource_campaign                          765707 non-null int64
trafficSource_campaignCode                      1 non-null object
trafficSource_isTrueDirect                      228716 non-null object
trafficSource_keyword                           765707 non-null int64
trafficSource_medium                            765707 non-null int64
trafficSource_referralPath                      765707 non-null int64
trafficSource_source                            765707 non-null int64
v_date                                          765707 non-null datetim
e64[ns]
dayofweek                                       765707 non-null int64
hours                                           765707 non-null int64
day                                             765707 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 239.5+ MB
```

In [60]:
```
train["totals_bounces"].fillna(0, inplace=True)
```

In [63]:
```
train["totals_bounces"].fillna(0, inplace=True)
train["totals_newVisits"].fillna(0, inplace=True)
train["totals_pageviews"].fillna(0, inplace=True)
train["totals_bounces"].fillna(0, inplace=True)
```

In [64]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 903653 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 903653 non-null int64
date                                            903653 non-null int64
fullVisitorId                                   903653 non-null object
sessionId                                       903653 non-null object
visitId                                         903653 non-null int64
visitNumber                                     903653 non-null int64
visitStartTime                                  903653 non-null int64
device_browser                                  903653 non-null int64
device_deviceCategory                           903653 non-null int64
device_isMobile                                 903653 non-null int64
device_operatingSystem                          903653 non-null int64
geoNetwork_city                                 903653 non-null int64
geoNetwork_continent                            903653 non-null int64
geoNetwork_country                              903653 non-null int64
geoNetwork_metro                                903653 non-null int64
geoNetwork_networkDomain                        903653 non-null int64
geoNetwork_region                               903653 non-null int64
geoNetwork_subContinent                         903653 non-null int64
totals_bounces                                  903653 non-null float64
totals_hits                                     903653 non-null float64
totals_newVisits                                903653 non-null float64
totals_pageviews                                903653 non-null float64
totals_transactionRevenue                       903653 non-null float64
trafficSource_adContent                         903653 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    903653 non-null int64
trafficSource_adwordsClickInfo.gclId            903653 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        21460 non-null object
trafficSource_adwordsClickInfo.page             903653 non-null int64
trafficSource_adwordsClickInfo.slot             903653 non-null int64
trafficSource_campaign                          903653 non-null int64
trafficSource_campaignCode                      1 non-null object
trafficSource_isTrueDirect                      274005 non-null object
trafficSource_keyword                           903653 non-null int64
trafficSource_medium                            903653 non-null int64
trafficSource_referralPath                      903653 non-null int64
trafficSource_source                            903653 non-null int64
v_date                                          903653 non-null datetim
e64[ns]
dayofweek                                       903653 non-null int64
hours                                           903653 non-null int64
day                                             903653 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 275.8+ MB
```

In [65]:
```
train_x, valid_x, train_y, valid_y = train_test_split(train[features], t
rain["totals_transactionRevenue"],
                                      test_size=0.25, random_state=20)

def permutation_test(train_X, train_y, val_X, val_y):
    params = {
        "objective" : "regression",
        "metric" : "rmse",
        "num_leaves" : 50,
        "min_child_samples" : 100,
        "learning_rate" : 0.1,
        "bagging_fraction" : 0.7,
        "feature_fraction" : 0.5,
        "bagging_frequency" : 5,
        "bagging_seed" : 2018,
        "verbosity" : -1,
        "njobs" : 3
    }

    lgtrain = lgb.Dataset(train_x, label=train_y)
    lgval = lgb.Dataset(valid_x, label=valid_y)
    model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_
stopping_rounds=250, verbose_eval=100)

#     pred_test_y = model.predict(test_X, num_iteration=model.best_itera
tion)
    pred_val_y = model.predict(valid_X, num_iteration=model.best_iterati
on)
    return model, pred_val_y
```

In [66]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[category_columns + numerical_colums]
        val_X = val_df[category_columns + numerical_colums]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews    0
```

```
---------------------------------------------------------------------------
LightGBMError                             Traceback (most recent call last)
<ipython-input-66-ea3a519919fc> in <module>()
     13         dev_X = dev_df[category_columns + numerical_colums]
     14         val_X = val_df[category_columns + numerical_colums]
---> 15         model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)

<ipython-input-65-865f3e4084aa> in permutation_test(train_X, train_y, val_X, val_y)
     19     lgtrain = lgb.Dataset(train_x, label=train_y)
     20     lgval = lgb.Dataset(valid_x, label=valid_y)
---> 21     model = lgb.train(params, lgtrain, 10000, valid_sets=[lgval], early_stopping_rounds=250, verbose_eval=100)
     22
     23 #     pred_test_y = model.predict(test_X, num_iteration=model.best_iteration)

/anaconda3/lib/python3.6/site-packages/lightgbm/engine.py in train(params, train_set, num_boost_round, valid_sets, valid_names, fobj, feval, init_model, feature_name, categorical_feature, early_stopping_rounds, evals_result, verbose_eval, learning_rates, keep_training_booster, callbacks)
    190     # construct booster
    191     try:
--> 192         booster = Booster(params=params, train_set=train_set)
    193         if is_valid_contain_train:
    194             booster.set_train_data_name(train_data_name)

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in __init__(self, params, train_set, model_file, silent)
   1485             self.handle = ctypes.c_void_p()
   1486             _safe_call(_LIB.LGBM_BoosterCreate(
-> 1487                 train_set.construct().handle,
   1488                 c_str(params_str),
   1489                 ctypes.byref(self.handle)))

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in construct(self)
    983                                 init_score=self.init_score, predictor=self._predictor,
    984                                 silent=self.silent, feature_name=self.feature_name,
--> 985                                 categorical_feature=self.categorical_feature, params=self.params)
    986             if self.free_raw_data:
    987                 self.data = None

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _lazy_init(self, data, label, reference, weight, group, init_score, predictor, silent, feature_name, categorical_feature, params)
    779                 raise TypeError('Cannot initialize Dataset from {}'.format(type(data).__name__))
    780         if label is not None:
--> 781             self.set_label(label)
```

```
    782         if self.get_label() is None:
    783             raise ValueError("Label should not be None")

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_label(s
elf, label)
   1275         if self.handle is not None:
   1276             label = list_to_1d_numpy(_label_from_pandas(label),
 name='label')
-> 1277             self.set_field('label', label)
   1278         return self
   1279

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_field(s
elf, field_name, data)
   1122             ptr_data,
   1123             ctypes.c_int(len(data)),
-> 1124             ctypes.c_int(type_data)))
   1125         return self
   1126

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _safe_call
(ret)
     43     """
     44     if ret != 0:
---> 45         raise LightGBMError(decode_string(_LIB.LGBM_GetLastErro
r()))
     46
     47

LightGBMError: Length of label is not same with #data
```

In [69]:
```
lgb_params = {"objective" : "regression", "metric" : "rmse",
              "num_leaves" : 50, "learning_rate" : 0.02,
              "bagging_fraction" : 0.75, "feature_fraction" : 0.8, "bagg
ing_frequency" : 9}

lgb_train = lgb.Dataset(train_x, label=train_y)
lgb_val = lgb.Dataset(valid_x, label=valid_y)
model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val], earl
y_stopping_rounds=1000, verbose_eval=250)
```

```
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 1.65654
[500]   valid_0's rmse: 1.65422
Did not meet early stopping. Best iteration is:
[584]   valid_0's rmse: 1.65401
```

In [68]:
```
    pred_val_y = model.predict(valid_x, num_iteration=model.best_iterati
on)
    pred_val_y
```

Out[68]:
```
array([ 1.21677678e-02, -2.61379075e-04, -3.63743323e-03, ...,
        4.80294321e-02, -2.44224749e-03,  3.94914822e-01])
```

In [77]:
```
def permutation_test(train_X, train_y, val_X, val_y):
    lgb_params = {"objective" : "regression", "metric" : "rmse",
              "num_leaves" : 50, "learning_rate" : 0.02,
              "bagging_fraction" : 0.75, "feature_fraction" : 0.8, "bagg
ing_frequency" : 9, "bagging_seed" : 2018,
               "verbosity" : -1,"njobs" : 3}

    lgb_train = lgb.Dataset(train_X, label=train_y)
    lgb_val = lgb.Dataset(val_X, label=val_y)
    model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val],
early_stopping_rounds=1000, verbose_eval=250)
    pred_val_y = model.predict(val_X, num_iteration=model.best_iteration
)

    return model, pred_val_y
```

In [82]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[category_columns + numerical_colums]
        val_X = val_df[category_columns + numerical_colums]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews    0
```

```
---------------------------------------------------------------------------
LightGBMError                             Traceback (most recent call last)
<ipython-input-82-ea3a519919fc> in <module>()
     13         dev_X = dev_df[category_columns + numerical_colums]
     14         val_X = val_df[category_columns + numerical_colums]
---> 15         model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)

<ipython-input-77-86614e56b4e9> in permutation_test(train_X, train_y, val_X, val_y)
      7     lgb_train = lgb.Dataset(train_x, label=train_y)
      8     lgb_val = lgb.Dataset(valid_x, label=valid_y)
----> 9     model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val], early_stopping_rounds=1000, verbose_eval=250)
     10     pred_val_y = model.predict(valid_x, num_iteration=model.best_iteration)
     11

/anaconda3/lib/python3.6/site-packages/lightgbm/engine.py in train(params, train_set, num_boost_round, valid_sets, valid_names, fobj, feval, init_model, feature_name, categorical_feature, early_stopping_rounds, evals_result, verbose_eval, learning_rates, keep_training_booster, callbacks)
    190     # construct booster
    191     try:
--> 192         booster = Booster(params=params, train_set=train_set)
    193         if is_valid_contain_train:
    194             booster.set_train_data_name(train_data_name)

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in __init__(self, params, train_set, model_file, silent)
   1485             self.handle = ctypes.c_void_p()
   1486             _safe_call(_LIB.LGBM_BoosterCreate(
-> 1487                 train_set.construct().handle,
   1488                 c_str(params_str),
   1489                 ctypes.byref(self.handle)))

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in construct(self)
    983                                 init_score=self.init_score, predictor=self._predictor,
    984                                 silent=self.silent, feature_name=self.feature_name,
--> 985                                 categorical_feature=self.categorical_feature, params=self.params)
    986             if self.free_raw_data:
    987                 self.data = None

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _lazy_init(self, data, label, reference, weight, group, init_score, predictor, silent, feature_name, categorical_feature, params)
    779                 raise TypeError('Cannot initialize Dataset from {}'.format(type(data).__name__))
    780         if label is not None:
--> 781             self.set_label(label)
```

```
    782         if self.get_label() is None:
    783             raise ValueError("Label should not be None")

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_label(s
elf, label)
   1275         if self.handle is not None:
   1276             label = list_to_1d_numpy(_label_from_pandas(label),
 name='label')
-> 1277             self.set_field('label', label)
   1278         return self
   1279

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in set_field(s
elf, field_name, data)
   1122             ptr_data,
   1123             ctypes.c_int(len(data)),
-> 1124             ctypes.c_int(type_data)))
   1125         return self
   1126

/anaconda3/lib/python3.6/site-packages/lightgbm/basic.py in _safe_call
(ret)
     43     """
     44     if ret != 0:
---> 45         raise LightGBMError(decode_string(_LIB.LGBM_GetLastErro
r()))
     46
     47

LightGBMError: Length of label is not same with #data
```

In [83]:
```
dev_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 765707 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 765707 non-null int64
date                                            765707 non-null int64
fullVisitorId                                   765707 non-null object
sessionId                                       765707 non-null object
visitId                                         765707 non-null int64
visitNumber                                     765707 non-null int64
visitStartTime                                  765707 non-null int64
device_browser                                  765707 non-null int64
device_deviceCategory                           765707 non-null int64
device_isMobile                                 765707 non-null int64
device_operatingSystem                          765707 non-null int64
geoNetwork_city                                 765707 non-null int64
geoNetwork_continent                            765707 non-null int64
geoNetwork_country                              765707 non-null int64
geoNetwork_metro                                765707 non-null int64
geoNetwork_networkDomain                        765707 non-null int64
geoNetwork_region                               765707 non-null int64
geoNetwork_subContinent                         765707 non-null int64
totals_bounces                                  765707 non-null float64
totals_hits                                     765707 non-null float64
totals_newVisits                                765707 non-null float64
totals_pageviews                                765707 non-null float64
totals_transactionRevenue                       765707 non-null float64
trafficSource_adContent                         765707 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    765707 non-null int64
trafficSource_adwordsClickInfo.gclId            765707 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        765707 non-null object
trafficSource_adwordsClickInfo.page             765707 non-null int64
trafficSource_adwordsClickInfo.slot             765707 non-null int64
trafficSource_campaign                          765707 non-null int64
trafficSource_campaignCode                      765707 non-null object
trafficSource_isTrueDirect                      765707 non-null object
trafficSource_keyword                           765707 non-null int64
trafficSource_medium                            765707 non-null int64
trafficSource_referralPath                      765707 non-null int64
trafficSource_source                            765707 non-null int64
v_date                                          765707 non-null datetim
e64[ns]
dayofweek                                       765707 non-null int64
hours                                           765707 non-null int64
day                                             765707 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 239.5+ MB
```

In [80]:
```
train["trafficSource_isTrueDirect"].fillna(0, inplace=True)
```

In [81]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 903653 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 903653 non-null int64
date                                            903653 non-null int64
fullVisitorId                                   903653 non-null object
sessionId                                       903653 non-null object
visitId                                         903653 non-null int64
visitNumber                                     903653 non-null int64
visitStartTime                                  903653 non-null int64
device_browser                                  903653 non-null int64
device_deviceCategory                           903653 non-null int64
device_isMobile                                 903653 non-null int64
device_operatingSystem                          903653 non-null int64
geoNetwork_city                                 903653 non-null int64
geoNetwork_continent                            903653 non-null int64
geoNetwork_country                              903653 non-null int64
geoNetwork_metro                                903653 non-null int64
geoNetwork_networkDomain                        903653 non-null int64
geoNetwork_region                               903653 non-null int64
geoNetwork_subContinent                         903653 non-null int64
totals_bounces                                  903653 non-null float64
totals_hits                                     903653 non-null float64
totals_newVisits                                903653 non-null float64
totals_pageviews                                903653 non-null float64
totals_transactionRevenue                       903653 non-null float64
trafficSource_adContent                         903653 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    903653 non-null int64
trafficSource_adwordsClickInfo.gclId            903653 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        903653 non-null object
trafficSource_adwordsClickInfo.page             903653 non-null int64
trafficSource_adwordsClickInfo.slot             903653 non-null int64
trafficSource_campaign                          903653 non-null int64
trafficSource_campaignCode                      903653 non-null object
trafficSource_isTrueDirect                      903653 non-null object
trafficSource_keyword                           903653 non-null int64
trafficSource_medium                            903653 non-null int64
trafficSource_referralPath                      903653 non-null int64
trafficSource_source                            903653 non-null int64
v_date                                          903653 non-null datetim
e64[ns]
dayofweek                                       903653 non-null int64
hours                                           903653 non-null int64
day                                             903653 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 275.8+ MB
```

In [79]:
```
dev_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 765707 entries, 0 to 903652
Data columns (total 40 columns):
channelGrouping                                 765707 non-null int64
date                                            765707 non-null int64
fullVisitorId                                   765707 non-null object
sessionId                                       765707 non-null object
visitId                                         765707 non-null int64
visitNumber                                     765707 non-null int64
visitStartTime                                  765707 non-null int64
device_browser                                  765707 non-null int64
device_deviceCategory                           765707 non-null int64
device_isMobile                                 765707 non-null int64
device_operatingSystem                          765707 non-null int64
geoNetwork_city                                 765707 non-null int64
geoNetwork_continent                            765707 non-null int64
geoNetwork_country                              765707 non-null int64
geoNetwork_metro                                765707 non-null int64
geoNetwork_networkDomain                        765707 non-null int64
geoNetwork_region                               765707 non-null int64
geoNetwork_subContinent                         765707 non-null int64
totals_bounces                                  765707 non-null float64
totals_hits                                     765707 non-null float64
totals_newVisits                                765707 non-null float64
totals_pageviews                                765707 non-null float64
totals_transactionRevenue                       765707 non-null float64
trafficSource_adContent                         765707 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    765707 non-null int64
trafficSource_adwordsClickInfo.gclId            765707 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        765707 non-null object
trafficSource_adwordsClickInfo.page             765707 non-null int64
trafficSource_adwordsClickInfo.slot             765707 non-null int64
trafficSource_campaign                          765707 non-null int64
trafficSource_campaignCode                      765707 non-null object
trafficSource_isTrueDirect                      228716 non-null object
trafficSource_keyword                           765707 non-null int64
trafficSource_medium                            765707 non-null int64
trafficSource_referralPath                      765707 non-null int64
trafficSource_source                            765707 non-null int64
v_date                                          765707 non-null datetim
e64[ns]
dayofweek                                       765707 non-null int64
hours                                           765707 non-null int64
day                                             765707 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 239.5+ MB
```

In [84]:
```
val_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 137946 entries, 4822 to 901262
Data columns (total 40 columns):
channelGrouping                                 137946 non-null int64
date                                            137946 non-null int64
fullVisitorId                                   137946 non-null object
sessionId                                       137946 non-null object
visitId                                         137946 non-null int64
visitNumber                                     137946 non-null int64
visitStartTime                                  137946 non-null int64
device_browser                                  137946 non-null int64
device_deviceCategory                           137946 non-null int64
device_isMobile                                 137946 non-null int64
device_operatingSystem                          137946 non-null int64
geoNetwork_city                                 137946 non-null int64
geoNetwork_continent                            137946 non-null int64
geoNetwork_country                              137946 non-null int64
geoNetwork_metro                                137946 non-null int64
geoNetwork_networkDomain                        137946 non-null int64
geoNetwork_region                               137946 non-null int64
geoNetwork_subContinent                         137946 non-null int64
totals_bounces                                  137946 non-null float64
totals_hits                                     137946 non-null float64
totals_newVisits                                137946 non-null float64
totals_pageviews                                137946 non-null float64
totals_transactionRevenue                       137946 non-null float64
trafficSource_adContent                         137946 non-null int64
trafficSource_adwordsClickInfo.adNetworkType    137946 non-null int64
trafficSource_adwordsClickInfo.gclId            137946 non-null int64
trafficSource_adwordsClickInfo.isVideoAd        137946 non-null object
trafficSource_adwordsClickInfo.page             137946 non-null int64
trafficSource_adwordsClickInfo.slot             137946 non-null int64
trafficSource_campaign                          137946 non-null int64
trafficSource_campaignCode                      137946 non-null object
trafficSource_isTrueDirect                      137946 non-null object
trafficSource_keyword                           137946 non-null int64
trafficSource_medium                            137946 non-null int64
trafficSource_referralPath                      137946 non-null int64
trafficSource_source                            137946 non-null int64
v_date                                          137946 non-null datetim
e64[ns]
dayofweek                                       137946 non-null int64
hours                                           137946 non-null int64
day                                             137946 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(29), object(5)
memory usage: 43.2+ MB
```

In [85]:
```
dev_df = train_permute[train['date']<=20170531]
val_df = train_permute[train['date']>20170531]
dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
val_y = np.log1p(val_df["totals_transactionRevenue"].values)
dev_X = dev_df[category_columns + numerical_colums]
val_X = val_df[category_columns + numerical_colums]
```

In [86]:
```
def permutation_test(train_X, train_y, val_X, val_y):
    lgb_params = {"objective" : "regression", "metric" : "rmse",
                "num_leaves" : 50, "learning_rate" : 0.02,
                "bagging_fraction" : 0.75, "feature_fraction" : 0.8, "bagg
ing_frequency" : 9, "bagging_seed" : 2018,
                 "verbosity" : -1,"njobs" : 3}

    lgb_train = lgb.Dataset(train_X, label=train_y)
    lgb_val = lgb.Dataset(val_X, label=val_y)
    model = lgb.train(lgb_params, lgb_train, 700, valid_sets=[lgb_val],
early_stopping_rounds=1000, verbose_eval=250)
    pred_val_y = model.predict(val_X, num_iteration=model.best_iteration
)

    return model, pred_val_y
```

In [87]:
```
from datetime import *
List = ["totals_pageviews", "totals_hits", "visitNumber", "geoNetwork_co
untry"]
for column_perm in List:
    for i in range(20):
        print(column_perm," ",i)
        train_permute = train.copy()
        train_permute[column_perm] = np.random.permutation(train_permute
[column_perm])
        dev_df = train_permute[train_permute['date']<=20170531]
        val_df = train_permute[train_permute['date']>20170531]
        dev_y = np.log1p(dev_df["totals_transactionRevenue"].values)
        val_y = np.log1p(val_df["totals_transactionRevenue"].values)

        dev_X = dev_df[category_columns + numerical_colums]
        val_X = val_df[category_columns + numerical_colums]
        model, pred_val = permutation_test(dev_X, dev_y, val_X, val_y)
```

```
totals_pageviews   0
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293957
[500]   valid_0's rmse: 0.293376
Did not meet early stopping. Best iteration is:
[416]   valid_0's rmse: 0.293341
totals_pageviews   1
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293929
[500]   valid_0's rmse: 0.293402
Did not meet early stopping. Best iteration is:
[427]   valid_0's rmse: 0.293347
totals_pageviews   2
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294092
[500]   valid_0's rmse: 0.293641
Did not meet early stopping. Best iteration is:
[524]   valid_0's rmse: 0.29361
totals_pageviews   3
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293988
[500]   valid_0's rmse: 0.293477
Did not meet early stopping. Best iteration is:
[400]   valid_0's rmse: 0.293388
totals_pageviews   4
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293943
[500]   valid_0's rmse: 0.293467
Did not meet early stopping. Best iteration is:
[419]   valid_0's rmse: 0.29344
totals_pageviews   5
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294109
[500]   valid_0's rmse: 0.293544
Did not meet early stopping. Best iteration is:
[425]   valid_0's rmse: 0.29349
totals_pageviews   6
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294005
[500]   valid_0's rmse: 0.293499
Did not meet early stopping. Best iteration is:
[566]   valid_0's rmse: 0.293476
totals_pageviews   7
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294175
[500]   valid_0's rmse: 0.293657
Did not meet early stopping. Best iteration is:
[407]   valid_0's rmse: 0.293597
totals_pageviews   8
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294264
[500]   valid_0's rmse: 0.293806
Did not meet early stopping. Best iteration is:
[466]   valid_0's rmse: 0.293778
totals_pageviews   9
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294119
```

```
[500]   valid_0's rmse: 0.293625
Did not meet early stopping. Best iteration is:
[439]   valid_0's rmse: 0.293549
totals_pageviews   10
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.29419
[500]   valid_0's rmse: 0.29368
Did not meet early stopping. Best iteration is:
[593]   valid_0's rmse: 0.293617
totals_pageviews   11
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293982
[500]   valid_0's rmse: 0.293465
Did not meet early stopping. Best iteration is:
[447]   valid_0's rmse: 0.293432
totals_pageviews   12
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294061
[500]   valid_0's rmse: 0.293498
Did not meet early stopping. Best iteration is:
[439]   valid_0's rmse: 0.293452
totals_pageviews   13
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294098
[500]   valid_0's rmse: 0.293582
Did not meet early stopping. Best iteration is:
[446]   valid_0's rmse: 0.293513
totals_pageviews   14
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293986
[500]   valid_0's rmse: 0.293515
Did not meet early stopping. Best iteration is:
[396]   valid_0's rmse: 0.293502
totals_pageviews   15
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294215
[500]   valid_0's rmse: 0.293685
Did not meet early stopping. Best iteration is:
[516]   valid_0's rmse: 0.293665
totals_pageviews   16
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294127
[500]   valid_0's rmse: 0.293542
Did not meet early stopping. Best iteration is:
[408]   valid_0's rmse: 0.2935
totals_pageviews   17
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294014
[500]   valid_0's rmse: 0.29349
Did not meet early stopping. Best iteration is:
[495]   valid_0's rmse: 0.293473
totals_pageviews   18
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.293935
[500]   valid_0's rmse: 0.293392
Did not meet early stopping. Best iteration is:
[474]   valid_0's rmse: 0.293356
```

```
totals_pageviews   19
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.294089
[500]   valid_0's rmse: 0.293597
Did not meet early stopping. Best iteration is:
[434]   valid_0's rmse: 0.293512
totals_hits   0
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287155
[500]   valid_0's rmse: 0.286349
Did not meet early stopping. Best iteration is:
[490]   valid_0's rmse: 0.286322
totals_hits   1
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287381
[500]   valid_0's rmse: 0.286622
Did not meet early stopping. Best iteration is:
[531]   valid_0's rmse: 0.286594
totals_hits   2
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287302
[500]   valid_0's rmse: 0.286821
Did not meet early stopping. Best iteration is:
[388]   valid_0's rmse: 0.286711
totals_hits   3
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287289
[500]   valid_0's rmse: 0.286739
Did not meet early stopping. Best iteration is:
[530]   valid_0's rmse: 0.286718
totals_hits   4
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287601
[500]   valid_0's rmse: 0.286977
Did not meet early stopping. Best iteration is:
[496]   valid_0's rmse: 0.286964
totals_hits   5
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287343
[500]   valid_0's rmse: 0.286745
Did not meet early stopping. Best iteration is:
[512]   valid_0's rmse: 0.286724
totals_hits   6
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287318
[500]   valid_0's rmse: 0.286654
Did not meet early stopping. Best iteration is:
[506]   valid_0's rmse: 0.286646
totals_hits   7
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287329
[500]   valid_0's rmse: 0.286638
Did not meet early stopping. Best iteration is:
[529]   valid_0's rmse: 0.286607
totals_hits   8
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287192
```

```
[500]   valid_0's rmse: 0.286643
Did not meet early stopping. Best iteration is:
[473]   valid_0's rmse: 0.28663
totals_hits   9
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287247
[500]   valid_0's rmse: 0.286707
Did not meet early stopping. Best iteration is:
[441]   valid_0's rmse: 0.286684
totals_hits   10
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287364
[500]   valid_0's rmse: 0.286804
Did not meet early stopping. Best iteration is:
[455]   valid_0's rmse: 0.286784
totals_hits   11
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.28726
[500]   valid_0's rmse: 0.286686
Did not meet early stopping. Best iteration is:
[501]   valid_0's rmse: 0.286678
totals_hits   12
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287296
[500]   valid_0's rmse: 0.286538
Did not meet early stopping. Best iteration is:
[679]   valid_0's rmse: 0.28651
totals_hits   13
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287269
[500]   valid_0's rmse: 0.28656
Did not meet early stopping. Best iteration is:
[428]   valid_0's rmse: 0.286497
totals_hits   14
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287442
[500]   valid_0's rmse: 0.286793
Did not meet early stopping. Best iteration is:
[631]   valid_0's rmse: 0.286698
totals_hits   15
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287417
[500]   valid_0's rmse: 0.286795
Did not meet early stopping. Best iteration is:
[530]   valid_0's rmse: 0.286777
totals_hits   16
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287312
[500]   valid_0's rmse: 0.286758
Did not meet early stopping. Best iteration is:
[655]   valid_0's rmse: 0.286719
totals_hits   17
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287357
[500]   valid_0's rmse: 0.286783
Did not meet early stopping. Best iteration is:
[481]   valid_0's rmse: 0.286749
```

```
totals_hits   18
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287233
[500]   valid_0's rmse: 0.286552
Did not meet early stopping. Best iteration is:
[630]   valid_0's rmse: 0.2865
totals_hits   19
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.287295
[500]   valid_0's rmse: 0.286548
Did not meet early stopping. Best iteration is:
[621]   valid_0's rmse: 0.286494
visitNumber   0
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283635
[500]   valid_0's rmse: 0.283071
Did not meet early stopping. Best iteration is:
[562]   valid_0's rmse: 0.28302
visitNumber   1
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283709
[500]   valid_0's rmse: 0.283045
Did not meet early stopping. Best iteration is:
[426]   valid_0's rmse: 0.282973
visitNumber   2
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283798
[500]   valid_0's rmse: 0.283358
Did not meet early stopping. Best iteration is:
[611]   valid_0's rmse: 0.28326
visitNumber   3
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283608
[500]   valid_0's rmse: 0.283082
Did not meet early stopping. Best iteration is:
[488]   valid_0's rmse: 0.283041
visitNumber   4
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283689
[500]   valid_0's rmse: 0.283122
Did not meet early stopping. Best iteration is:
[482]   valid_0's rmse: 0.283085
visitNumber   5
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283621
[500]   valid_0's rmse: 0.283019
Did not meet early stopping. Best iteration is:
[618]   valid_0's rmse: 0.282961
visitNumber   6
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283617
[500]   valid_0's rmse: 0.283002
Did not meet early stopping. Best iteration is:
[477]   valid_0's rmse: 0.282969
visitNumber   7
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283629
```

```
[500]   valid_0's rmse: 0.283161
Did not meet early stopping. Best iteration is:
[519]   valid_0's rmse: 0.283117
visitNumber   8
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283668
[500]   valid_0's rmse: 0.283049
Did not meet early stopping. Best iteration is:
[505]   valid_0's rmse: 0.283022
visitNumber   9
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283675
[500]   valid_0's rmse: 0.283176
Did not meet early stopping. Best iteration is:
[544]   valid_0's rmse: 0.283107
visitNumber   10
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283767
[500]   valid_0's rmse: 0.283141
Did not meet early stopping. Best iteration is:
[495]   valid_0's rmse: 0.283128
visitNumber   11
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.28369
[500]   valid_0's rmse: 0.283058
Did not meet early stopping. Best iteration is:
[535]   valid_0's rmse: 0.283021
visitNumber   12
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283665
[500]   valid_0's rmse: 0.28309
Did not meet early stopping. Best iteration is:
[529]   valid_0's rmse: 0.283026
visitNumber   13
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283739
[500]   valid_0's rmse: 0.283311
Did not meet early stopping. Best iteration is:
[544]   valid_0's rmse: 0.283267
visitNumber   14
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283789
[500]   valid_0's rmse: 0.283214
Did not meet early stopping. Best iteration is:
[517]   valid_0's rmse: 0.283202
visitNumber   15
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283702
[500]   valid_0's rmse: 0.283172
Did not meet early stopping. Best iteration is:
[603]   valid_0's rmse: 0.283063
visitNumber   16
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283643
[500]   valid_0's rmse: 0.283321
Did not meet early stopping. Best iteration is:
[413]   valid_0's rmse: 0.28323
```

```
visitNumber   17
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283608
[500]   valid_0's rmse: 0.283112
Did not meet early stopping. Best iteration is:
[394]   valid_0's rmse: 0.283069
visitNumber   18
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283661
[500]   valid_0's rmse: 0.283176
Did not meet early stopping. Best iteration is:
[513]   valid_0's rmse: 0.283131
visitNumber   19
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283704
[500]   valid_0's rmse: 0.283191
Did not meet early stopping. Best iteration is:
[543]   valid_0's rmse: 0.283145
geoNetwork_country   0
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284059
[500]   valid_0's rmse: 0.283112
Did not meet early stopping. Best iteration is:
[621]   valid_0's rmse: 0.282976
geoNetwork_country   1
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.283963
[500]   valid_0's rmse: 0.283208
Did not meet early stopping. Best iteration is:
[569]   valid_0's rmse: 0.283122
geoNetwork_country   2
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284267
[500]   valid_0's rmse: 0.283418
Did not meet early stopping. Best iteration is:
[614]   valid_0's rmse: 0.283297
geoNetwork_country   3
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284322
[500]   valid_0's rmse: 0.283479
Did not meet early stopping. Best iteration is:
[684]   valid_0's rmse: 0.283304
geoNetwork_country   4
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.28413
[500]   valid_0's rmse: 0.283294
Did not meet early stopping. Best iteration is:
[673]   valid_0's rmse: 0.283137
geoNetwork_country   5
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.28418
[500]   valid_0's rmse: 0.283343
Did not meet early stopping. Best iteration is:
[491]   valid_0's rmse: 0.283322
geoNetwork_country   6
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284356
```

```
[500]   valid_0's rmse: 0.283515
Did not meet early stopping. Best iteration is:
[689]   valid_0's rmse: 0.283382
geoNetwork_country   7
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284303
[500]   valid_0's rmse: 0.283379
Did not meet early stopping. Best iteration is:
[663]   valid_0's rmse: 0.283316
geoNetwork_country   8
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284227
[500]   valid_0's rmse: 0.283349
Did not meet early stopping. Best iteration is:
[665]   valid_0's rmse: 0.283297
geoNetwork_country   9
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284319
[500]   valid_0's rmse: 0.283571
Did not meet early stopping. Best iteration is:
[638]   valid_0's rmse: 0.283543
geoNetwork_country   10
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284236
[500]   valid_0's rmse: 0.283387
Did not meet early stopping. Best iteration is:
[649]   valid_0's rmse: 0.283332
geoNetwork_country   11
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284112
[500]   valid_0's rmse: 0.283397
Did not meet early stopping. Best iteration is:
[620]   valid_0's rmse: 0.283352
geoNetwork_country   12
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284175
[500]   valid_0's rmse: 0.283263
Did not meet early stopping. Best iteration is:
[616]   valid_0's rmse: 0.283146
geoNetwork_country   13
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284245
[500]   valid_0's rmse: 0.283155
Did not meet early stopping. Best iteration is:
[590]   valid_0's rmse: 0.283052
geoNetwork_country   14
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284112
[500]   valid_0's rmse: 0.28323
Did not meet early stopping. Best iteration is:
[621]   valid_0's rmse: 0.283157
geoNetwork_country   15
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284216
[500]   valid_0's rmse: 0.28333
Did not meet early stopping. Best iteration is:
[639]   valid_0's rmse: 0.28329
```

```
geoNetwork_country   16
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284283
[500]   valid_0's rmse: 0.283315
Did not meet early stopping. Best iteration is:
[531]   valid_0's rmse: 0.283247
geoNetwork_country   17
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284279
[500]   valid_0's rmse: 0.283323
Did not meet early stopping. Best iteration is:
[663]   valid_0's rmse: 0.283153
geoNetwork_country   18
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284205
[500]   valid_0's rmse: 0.283445
Did not meet early stopping. Best iteration is:
[634]   valid_0's rmse: 0.283317
geoNetwork_country   19
Training until validation scores don't improve for 1000 rounds.
[250]   valid_0's rmse: 0.284167
[500]   valid_0's rmse: 0.283387
Did not meet early stopping. Best iteration is:
[640]   valid_0's rmse: 0.283291
```

In [88]:
```
cnt_srs = train.groupby('fullVisitorId')['totals_transactionRevenue'].ag
g(['size', 'count', 'mean'])
```

```
---------------------------------------------------------------------
----
KeyError                                  Traceback (most recent call l
ast)
<ipython-input-88-eecefcbd0f58> in <module>()
----> 1 cnt_srs = train.groupby('fullVisitorId')['transactionRevenue'].
agg(['size', 'count', 'mean'])

/anaconda3/lib/python3.6/site-packages/pandas/core/base.py in __getitem
__(self, key)
    265         else:
    266             if key not in self.obj:
--> 267                 raise KeyError("Column not found: {key}".format
(key=key))
    268             return self._gotitem(key, ndim=1)
    269

KeyError: 'Column not found: transactionRevenue'
```

In [92]:
```
train['Purchase'] =  train['totals_transactionRevenue'].apply(lambda x :
 [0,1][x>0])
```

In [94]:
```
cnt_srs = train.groupby('fullVisitorId')['Purchase'].agg(['size', 'sum'
])
```

In [96]:
```
cnt_mean = train.groupby('fullVisitorId')['totals_transactionRevenue'].a
gg(['size', 'mean'])
```

In [97]:
```
cnt_srs['mean'] = cnt_mean['mean']
```

In [99]:
```
cnt_srs['prob'] = cnt_srs['sum']/cnt_srs['size']
```

In [102]:
```
sorted(cnt_srs)
```

Out[102]:
```
['mean', 'prob', 'size', 'sum']
```

In [108]:
```
cnt_srs.sort_values(by=['prob'], ascending=False)
```

Out[108]:

	size	sum	mean	prob
fullVisitorId				
8544342563894929 78	1	1	17.488007	1.0
9799169384521023462	1	1	16.905188	1.0
8263279530140831487	1	1	17.280621	1.0
4339064982507273483	1	1	16.759423	1.0
2364782563241908647	1	1	16.873278	1.0
6463910848287283554	1	1	17.942161	1.0
6124362361380921120	1	1	18.155934	1.0
3943384499857599795	1	1	18.875190	1.0
646337888671074736	1	1	18.197037	1.0
1766466254683326555	1	1	16.617051	1.0
0725021958135646864	1	1	18.077909	1.0
8535572302467268502	1	1	16.670831	1.0
1766574900434071803	1	1	18.197412	1.0
0532638128006986594	1	1	17.072067	1.0
7979010430069561046	1	1	15.518439	1.0
3597020423335627391	1	1	17.567365	1.0
7000425720623417338	1	1	18.826013	1.0
4525601131116258606	1	1	17.375988	1.0
7004942470658795502	1	1	17.225998	1.0
555059843722195488	1	1	16.379690	1.0
9475218131155083092	1	1	16.992731	1.0
4525480404637930448	1	1	16.759423	1.0
1490558553818530030	1	1	18.953307	1.0
6730027102407598454	1	1	18.091621	1.0
6462196953273318864	2	2	18.174098	1.0
0533361455920078502	1	1	17.477789	1.0
646210848993699313	1	1	16.587474	1.0
0839560865423027275	1	1	17.825749	1.0
928560936983652295	1	1	17.643717	1.0
2367296881084974931	1	1	19.470593	1.0
...	...	...	...	...

	size	sum	mean	prob
fullVisitorId				
3363022075690047509	1	0	0.000000	0.0
3363049454115599878	1	0	0.000000	0.0
3363072230242290275	1	0	0.000000	0.0
3363083173079066254	1	0	0.000000	0.0
3363125424528665036	1	0	0.000000	0.0
3363127210013647940	2	0	0.000000	0.0
3363130045994395264	1	0	0.000000	0.0
3363142491866750712	1	0	0.000000	0.0
3362914540532840857	1	0	0.000000	0.0
3362895436652161023	1	0	0.000000	0.0
3362887836351430530	1	0	0.000000	0.0
3362887286509658825	1	0	0.000000	0.0
3362692566739287520	1	0	0.000000	0.0
3362719132308434864	1	0	0.000000	0.0
336273002445019042	1	0	0.000000	0.0
336273879604982296	1	0	0.000000	0.0
3362748478613029748	1	0	0.000000	0.0
3362764928219635352	1	0	0.000000	0.0
3362776858163814059	3	0	0.000000	0.0
3362801972340023716	1	0	0.000000	0.0
3362814773077129271	1	0	0.000000	0.0
3362828719758803736	1	0	0.000000	0.0
3362833331111996336	1	0	0.000000	0.0
3362842813912284200	2	0	0.000000	0.0
3362844321977235129	1	0	0.000000	0.0
3362854807171208318	1	0	0.000000	0.0
3362855878337917591	1	0	0.000000	0.0
336286767069572297	3	0	0.000000	0.0
3362869754320830266	2	0	0.000000	0.0
9999986437109498564	1	0	0.000000	0.0

714167 rows × 4 columns

In [ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt
spearman_correlation = train.corr(method='spearman')
pick_columns=spearman_correlation.nlargest(20, 'totals_transactionRevenu
e').index
correlationmap = np.corrcoef(train[pick_columns].values.T)
sns.set(font_scale=1.0)
heatmap = sns.heatmap(correlationmap, cbar=True, annot=True, square=True
 , fmt='.2f',
                      yticklabels=train.values, xticklabels=train.values
)
plt.show()
```