Zinvestor

Research Factors Impacting Real Estate Assessment using Zillow data (Zestimate)

September 9, 2017

Darius Bailey, Nasir Sayed, Uzair Siddiqui

Georgetown University Data Science Capstone Project

convert_pandoc

Project Goal:

- Identify drivers of Zestimate error at the zip code level within Washington DC.
- Model residual of Zestimate to actual sales price within 2017.
- Covariates used to predict Zestimate error include individual property data, neighborhood demographic data, and business license data.

Hypothesis:

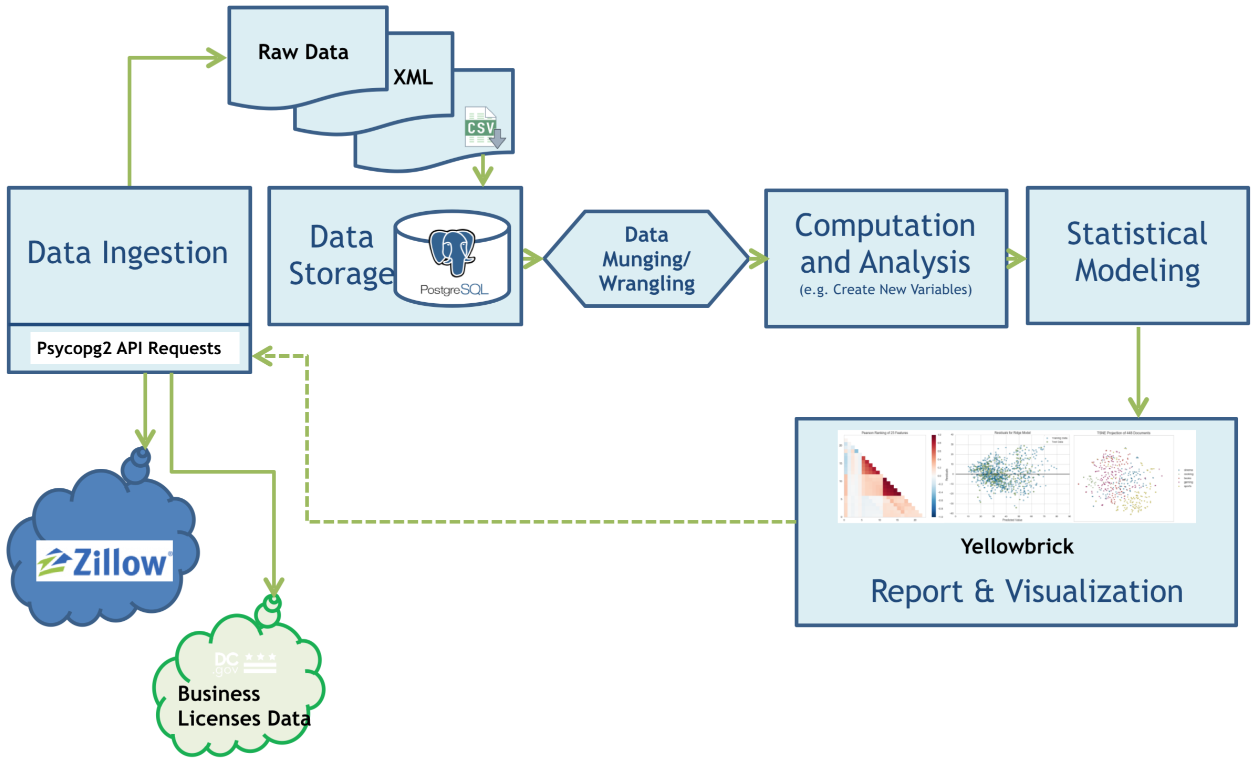
Real Estate values fluctuate based on local activities at zip code/neighborhood level. These local activities (esp. businesses and population demographics) highly impact Zestimate.

What is Zestimate?:

- The Zestimate® home value is Zillow's estimated market value for an individual home and is calculated for about 100 million homes nationwide.
- It is a starting point in determining a home's value but it is not an official appraisal.
- The Zestimate is automatically computed daily based on millions of public and user-submitted data points.

Architecture

Architecture Diagram: “Zinvestor”



Data Ingestion

The following data sources were explored and appropriately incorporated into the model.

Zillow Property Data:

- Zillow APIs were initially used to obtain zillow data for properties
  - However, Zillow APIs only allow data for one property at a time based on exact physical address or Zillow Property ID (ZPID)
- Team used Web Scrapping methodology (BeautifulSoup) to obtain ZPID for Houses for Sale in Washington DC (~11k records)
- Used Zillow API to obtain detailed property information on each scrapped ZPID
- Zillow limits API calls to 3000 calls per day
- Needed to collect 6 weeks of data
- Distributed ZPID over team members; Ran Zillow APIs calls 3 times a week for six weeks.
- The detailed property data from API calls were stored as XML and additional challenges were encountered as we imported to CSV format

Open Data DC:

- Obtained 5 year Business License data from http://opendata.dc.gov/ (http://opendata.dc.gov/) as CSVs
- Zillow Neighborhood Data:
- Downloaded Zillow Neighborhood data as CSVs from Zillow website

Census:

- Developed API calls using Psycopg2 to obtain ACS data from US Census but we ended up not using this data in our analysis

In [1]:

```
# Import all needed packages
import psycopg2
import pandas as pd
import pandas.io.sql as pdsql
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from pandas.tools.plotting import scatter_matrix
from sklearn import cross_validation as cv
from sklearn.cross_validation import train_test_split as tts
from sklearn.decomposition import PCA
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import Ridge, RandomizedLasso, ElasticNet, LinearR
from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error as mse
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/sklearn/cross_v
alidation.py:44: DeprecationWarning: This module was deprecated in versio
n 0.18 in favor of the model_selection module into which all the refactor
ed classes and functions are moved. Also note that the interface of the n
ew CV iterators are different from that of this module. This module will
 be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

Data Munging & Wrangling

Initially we used locally installed PostgreSQL database but soon had to move to AWS PostgreSQL so all team members can work from the same data set.

First step was to combine primary ZPID files with CSVs from 3 team members capturing 3 CSV per week for 6 weeks Essentially combine 55 CSV in a single table ('zillow_weekly) in Postgres. During the process we encountered challenges in data types, null values, missing Zestimate etc..

Uploaded Business License (license_summary_float & lic_cat_crosstab_6m_yoy) and Zillow Neighborhood data (zhvi) into AWS PostgreSQL and organized it by zip code.

Using psycopg2, connected to AWS PostgreSQL database

In [2]:

```
# Connect to AWS Database
con=psycopg2.connect(dbname= 'DCZillow',
                     host='dczillow.cfdlhqngxmri.us-east-1.rds.amazonaws.com
                     port='5432',
                     user= 'DCZillow',
                     password= 'DCZillow');

cur = con.cursor()
```

In [3]:

```
# Pull in weekly Zillow API call data
df = pdsql.read_sql("""SELECT * FROM zillow_weekly""", con)

# Select only the columns needed for analysis / modeling
df1 = df[['bedrooms', 'bathrooms', 'yearbuilt', 'lotsizesqft', 'taxassessmen
         'for_sale','zpid','street','city','state','zipcode',
          'zestimate_1', 'zestimate_6', 'percentile_1', 'percentile_6', 'las
```

In [4]:

```
# Pull Aggregated Basic Business License Change by Zipcode
df_lic_sum = pdsql.read_sql("""SELECT * FROM license_summary_float""", con)

df_lic_sum.loc[:,'bbl_pct_chg_1617'] = df_lic_sum['growth_yoy']
df_lic_sum.loc[:,'bbl_pct_chg_1217'] = df_lic_sum['growth_yoy_5']

df_lic_sum1 = df_lic_sum[['zipcode', 'bbl_pct_chg_1617', 'bbl_pct_chg_1217']
```

In [5]:

```
# Pull Basic Business License Category Change by Zipcode
df_lic_cat_xtab_6m= pdsql.read_sql("""SELECT * FROM lic_cat_crosstab_6m_yoy'
df_lic_cat_xtab_6m.fillna(0)
```

Out[5]:

	zipcode	barber_shop	cigarette_retail	delicatessen	food_products	parking_facility_atte
0	20016	0.0	0.000000	0.000000	0.000000	0.000000
1	20005	0.0	0.000000	0.000000	0.000000	2.727273
2	20036	0.0	0.000000	0.000000	0.000000	-0.080000
3	20037	0.0	0.000000	0.000000	0.000000	0.000000
4	20004	0.0	0.000000	2.666667	-0.571429	1.444444
5	20007	0.0	0.000000	1.272727	-0.950704	0.000000
6	20017	0.0	0.000000	3.666667	0.000000	0.000000
7	20008	0.0	0.000000	0.000000	0.000000	0.000000
8	20002	0.0	-0.150000	0.377778	2.057692	0.000000
9	20003	0.0	-0.944444	0.000000	0.000000	1.000000
10	20009	0.0	-0.421053	0.687500	0.000000	0.000000
11	20011	0.0	0.062500	4.090909	0.250000	0.000000
12	20019	0.0	0.000000	0.000000	-0.222222	0.000000
13	20020	0.0	1.000000	2.900000	-0.300000	0.000000
14	20032	0.0	-0.250000	-0.307692	0.000000	0.000000
15	20001	-0.8	0.068966	1.925926	-0.400000	2.200000
16	20024	0.0	0.000000	0.000000	0.000000	0.000000
17	20010	0.0	0.000000	0.000000	0.000000	0.000000
18	20012	0.0	0.000000	0.000000	0.000000	0.000000
19	20015	0.0	0.000000	0.000000	0.000000	0.000000
20	20018	0.0	0.000000	0.000000	0.000000	0.000000

21 rows × 23 columns

In [6]:

```
# Pull ZHVI Change by Zipcode
df_zhvi= pdsql.read_sql("""SELECT * FROM zhvi""", con)

columns = ('mom','qoq','yoy','fiveyear','tenyear','peakmonth','peakmonth','p
col_prefix = 'zhvi_'

for i in columns:
    newvar = col_prefix + str(i)
    df_zhvi.rename(columns={i : newvar}, inplace=True)

df_zhvi.rename(columns={'regionname' : 'zipcode'}, inplace=True)
df_zhvi.loc[:,'zhvi_peakmonth'] = pd.to_datetime(df_zhvi['zhvi_peakmonth'])
df_zhvi.loc[:,'lasttimeatcurrzhvi'] = pd.to_datetime(df_zhvi['lasttimeatcurr

df_zhvi1 = df_zhvi[['zipcode', 'sizerank', 'zhvi','zhvi_mom','zhvi_qoq','zhv
                    'zhvi_peakmonth','peakzhvi','zhvi_pctfallfrompeak','last
df_zhvi1.fillna(0)
```

Out[6]:

	zipcode	sizerank	zhvi	zhvi_mom	zhvi_qoq	zhvi_yoy	zhvi_fiveyear	zhvi_tenyear
0	20009	27	565400	-0.006152	-0.005977	0.064383	0.052053	0.024720
1	20002	28	612700	-0.007291	-0.003091	0.048246	0.101941	0.000000
2	20011	140	568900	-0.003852	0.001408	0.107671	0.123579	0.037577
3	20019	205	293900	0.006507	0.021195	0.136944	0.119687	0.020936
4	20001	265	647600	-0.004764	-0.005681	0.065657	0.083636	0.040571
5	20020	361	307800	-0.006135	-0.010608	0.081518	0.114632	0.018479
6	20008	1225	856100	-0.004072	0.003634	0.054570	0.044144	0.018017
7	20032	1576	278300	0.012368	0.032653	0.108323	0.100896	0.014032
8	20016	1883	983100	-0.004557	-0.003548	0.052119	0.052494	0.026411
9	20003	2048	742500	-0.004158	-0.010528	0.061472	0.077594	0.036841
10	20010	2417	665300	-0.003744	-0.004787	0.042137	0.091722	0.043934
11	20007	2448	960200	-0.005901	-0.005798	0.047110	0.050291	0.022360
12	20005	4824	513700	-0.005421	-0.003492	0.060706	0.044807	0.023858
13	20017	4983	514900	-0.004639	0.003704	0.130654	0.115956	0.036234
14	20018	5062	511700	-0.005829	-0.004087	0.140660	0.120349	0.000000
15	20024	5082	399000	-0.005979	-0.005236	0.061453	0.073634	0.021546
16	20037	5520	518800	-0.006321	-0.005559	0.074343	0.035339	0.018313
17	20015	6037	1018700	-0.011163	-0.013557	0.027019	0.043944	0.020771
18	20012	6543	723900	-0.001517	0.004858	0.100152	0.091015	0.036954
19	20036	8427	380700	-0.013475	-0.019572	0.044444	0.040271	0.018705
20	20004	10800	484100	-0.013651	-0.018053	0.073155	0.031183	0.020034

```
In [7]: # Close connection to the database
        cur.close()
        con.close()
```

```
In [8]: # Merge Weekly Zillow + Aggregated BBL Change + BBL Category Change + ZHVI C
        merged = pd.merge(df1,
                          df_lic_sum1,
                          on='zipcode',
                          how='left')

        merged1 = pd.merge(merged,
                          df_lic_cat_xtab_6m,
                          on='zipcode',
                          how='left')

        merged2 = pd.merge(merged1,
                          df_zhvi1,
                          on='zipcode',
                          how='left')

        merged2.shape
```

```
Out[8]: (4690, 53)
```

Treat Variables and Create Target

Target is defined as (Sold Price / Zestimate)

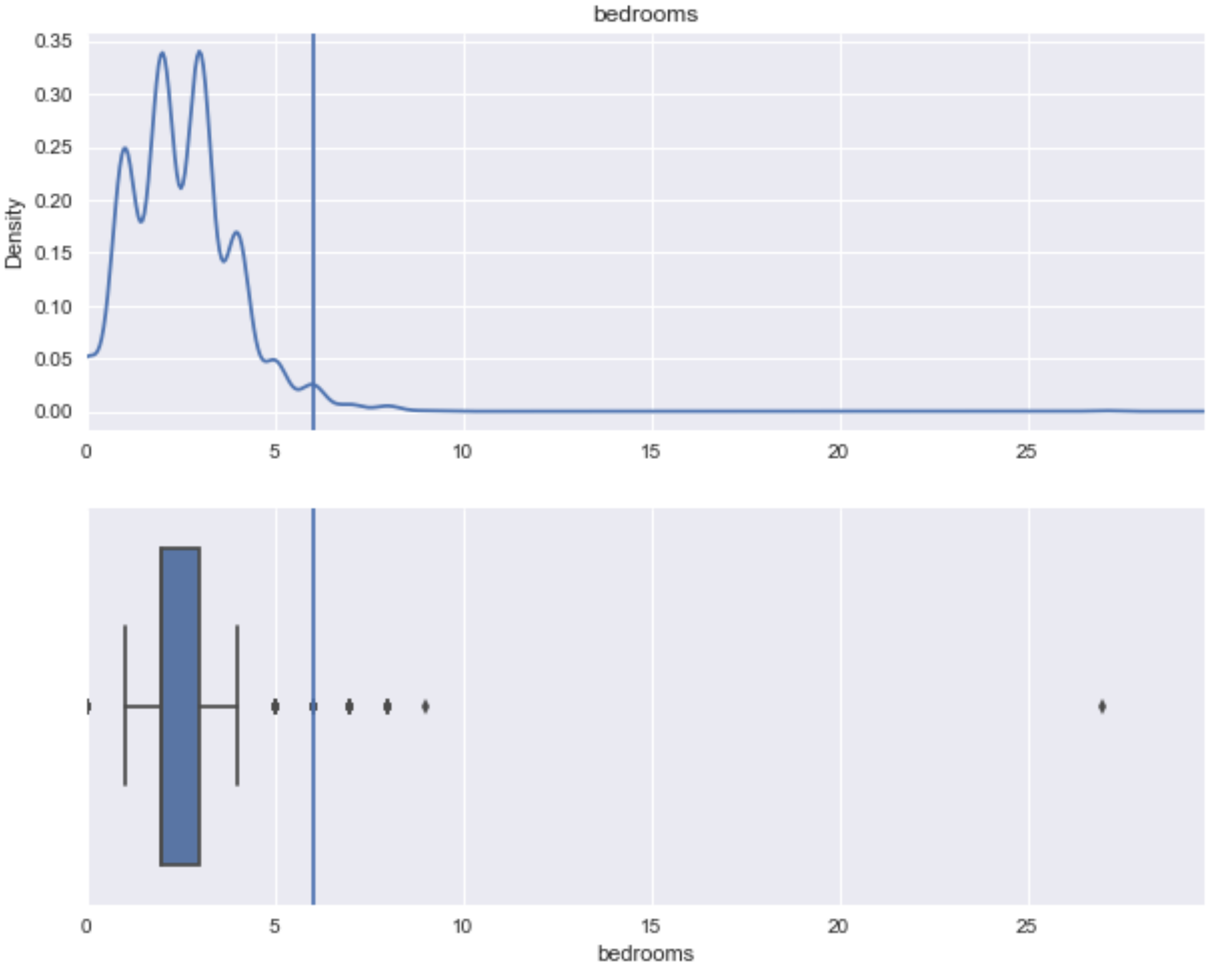
```
In [9]: # Some variables were pulled into the database as the incorrect type. Correc

        merged2.loc[:,'lastsolddate_6'] = pd.to_datetime(merged2['lastsolddate_6'])
        merged2.loc[:,'bathrooms'] = pd.to_numeric(merged2['bathrooms'], errors='coe
        merged2.loc[:,'lotsizesqft'] = merged2['lotsizesqft'].astype(int)
        merged2.loc[:,'target_2017'] = (merged2['lastsoldprice_6'] / merged2['zestim

        merged3= merged2.dropna(subset=['zpid'])

        merged3.index = merged3['lastsolddate_6']
        merged4=merged3['1/1/2017':]
        list(merged4)
```

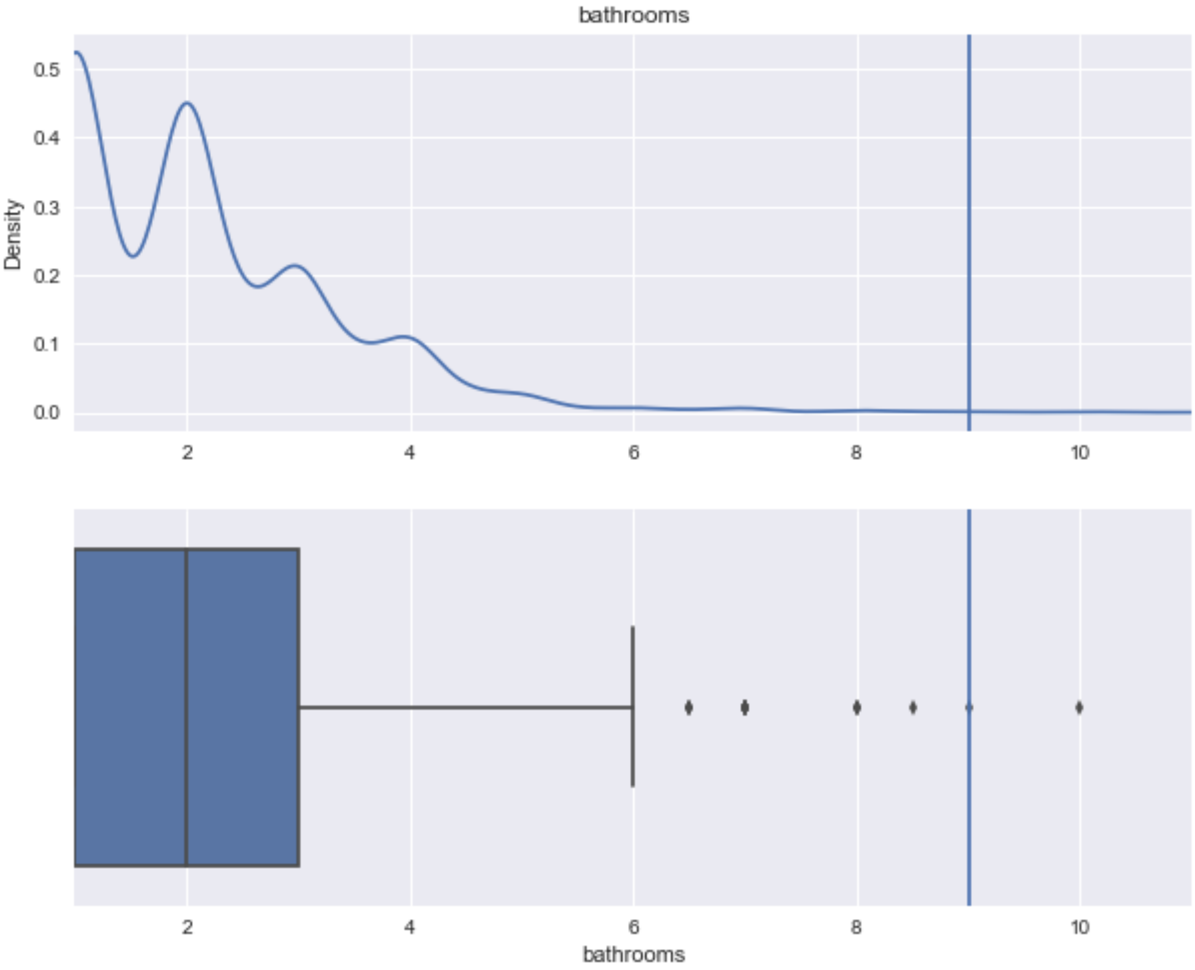
```
Out[9]: ['bedrooms',
         'bathrooms',
         'yearbuilt',
         'lotsizesqft',
         'taxassessment_6',
         'finishedsqft',
         'for_sale',
         'zpid',
         'street',
         'city',
         'state',
         'zipcode',
         'zestimate_1',
         'zestimate_6',
         'percentile_1',
         'percentile_6',
         'lastsolddate_6',
         'lastsoldprice_6',
         'bbl_pct_chg_1617',
         'bbl_pct_chg_1217',
         'barber_shop',
         'cigarette_retail',
         'delicatessen',
         'food_products',
         'parking_facility_attendant',
         'grocery_store',
         'special_events',
         'charitable_solicitation',
         'home_improvement_salesman',
         'secondhand_dealers_a',
         'motor_vehicle_salesman',
         'beauty_shop',
         'parking_facility',
         'gen_contr_construction_mngr',
         'consumer_goods_auto_repair',
         'apartment',
         'two_family_rental',
         'home_improvement_contractor',
         'general_business_licenses',
         'patent_medicine',
         'restaurant',
         'one_family_rental',
         'sizerank',
         'zhvi',
         'zhvi_mom',
         'zhvi_qoq',
         'zhvi_yoy',
         'zhvi_fiveyear',
         'zhvi_tenyear',
         'zhvi_peakmonth',
         'peakzhvi',
         'zhvi_pctfallfrompeak',
         'lasttimeatcurrzhvi',
         'target_2017']
```

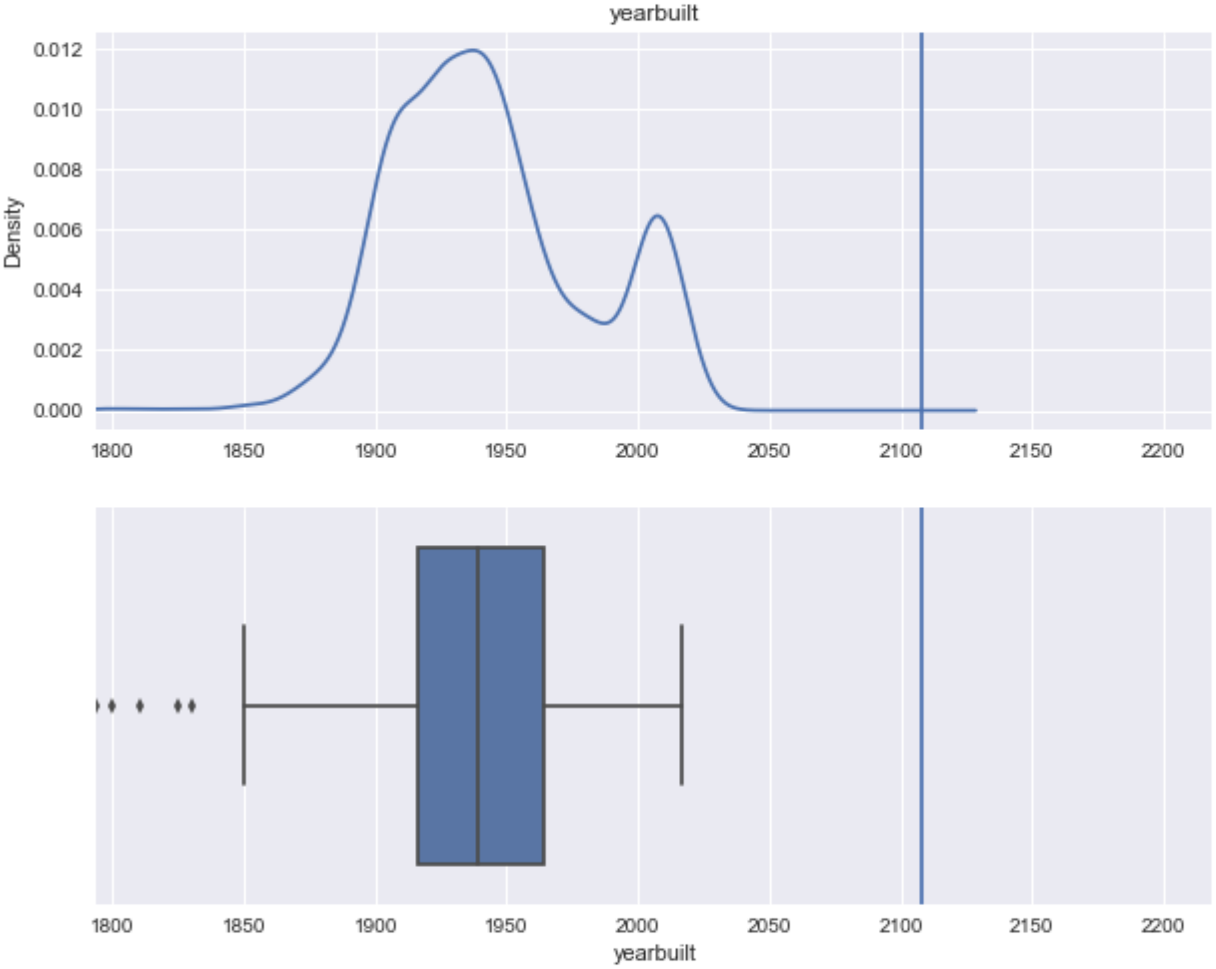
Visual Data Analysis

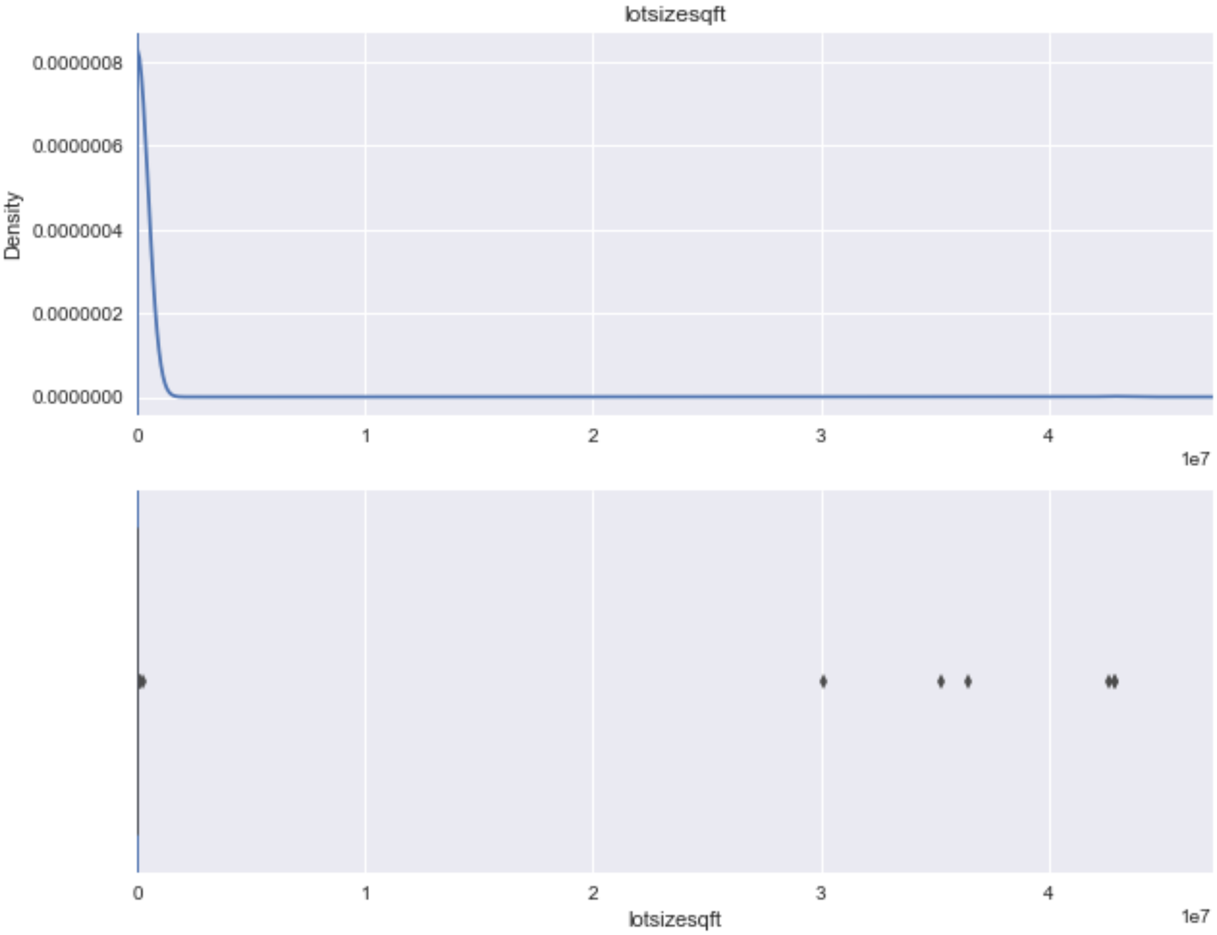
In [10]:

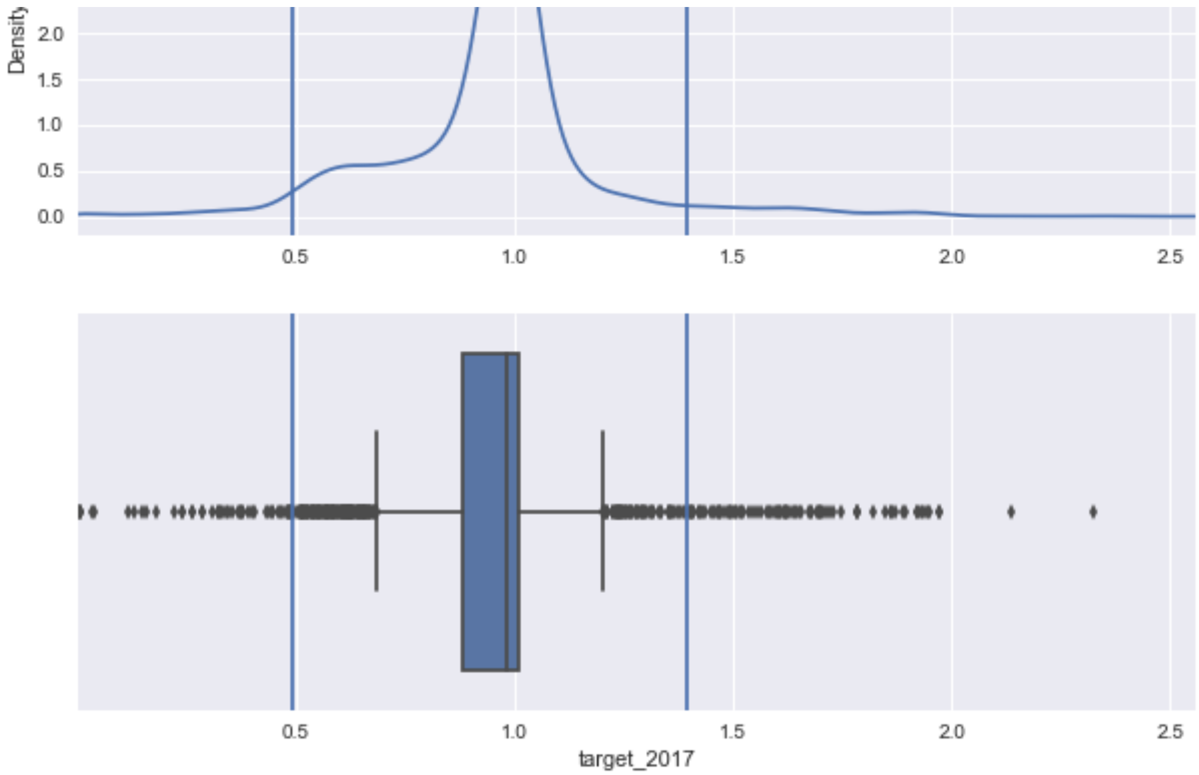
```
# Plot Density and BoxPlots to understand feature distribution

numerics = list(merged4.select_dtypes(include=[np.number]).columns.values)

for i in numerics:


    q25, q75 = np.percentile(merged4[i].dropna(), [25 ,75])
    iqr = q75 - q25

    min = q25 - (iqr*3)
    max = q75 + (iqr*3)

    plt.figure(figsize=(10,8))
    plt.subplot(211)
    plt.xlim(merged4[i].min(), merged4[i].max()*1.1)
    plt.axvline(x=min)
    plt.axvline(x=max)
    plt.title(i)
    ax = merged4[i].plot(kind='kde')

    plt.subplot(212)
    plt.xlim(merged4[i].min(), merged4[i].max()*1.1)
    sns.boxplot(x=merged4[i])
    plt.axvline(x=min)
    plt.axvline(x=max)

    sns.plt.show()
```


bedrooms
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Density
0
5
10
15
20
25
0
5
10
15
20
25
bedrooms


bathrooms
Density
0.5
0.4
0.3
0.2
0.1
0.0
2
4
6
8
10
2
4
6
8
10
bathrooms


yearbuilt
Density
0.012
0.010
0.008
0.006
0.004
0.002
0.000
1800
1850
1900
1950
2000
2050
2100
2150
2200
yearbuilt


lotsizesqft
Density
0.0000008
0.0000006
0.0000004
0.0000002
0.0000000
0
1
2
3
4
1e7
lotsizesqft



Takeaways from Initial Visual Analysis

1. Functional form of some variables may need to be changed

For example, lotsizesqft and taxassessment may benefit from transforming into Log form

2. Many fields have extreme values that need to be treated

In this case, exclude values above 99th percentile

3. Features w/multimodal distributions need to be assessed more closely

For example, two building booms can be observed in DC. Features leveraging this info may be useful.

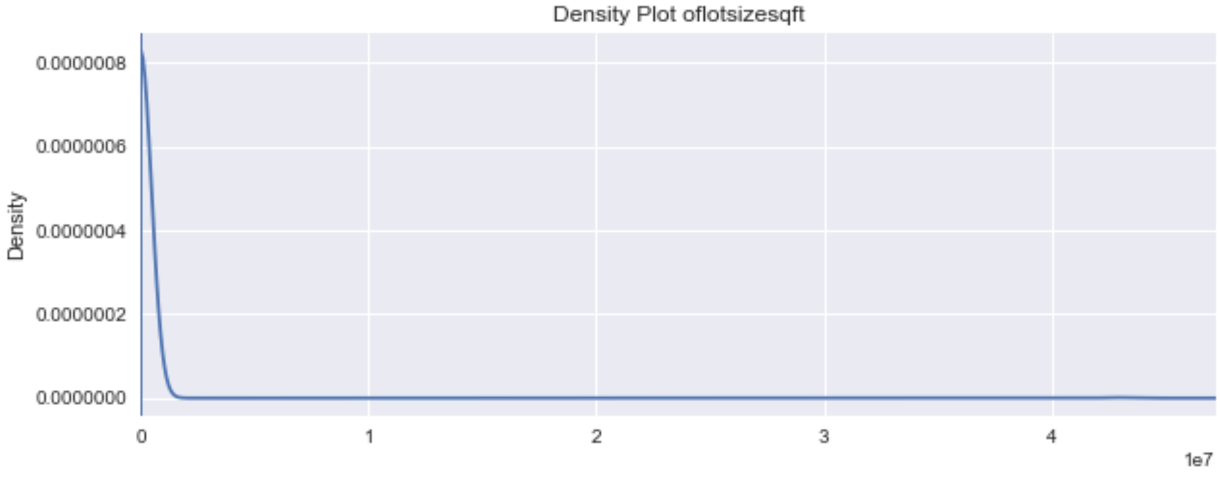
4. Additional derived features can be created

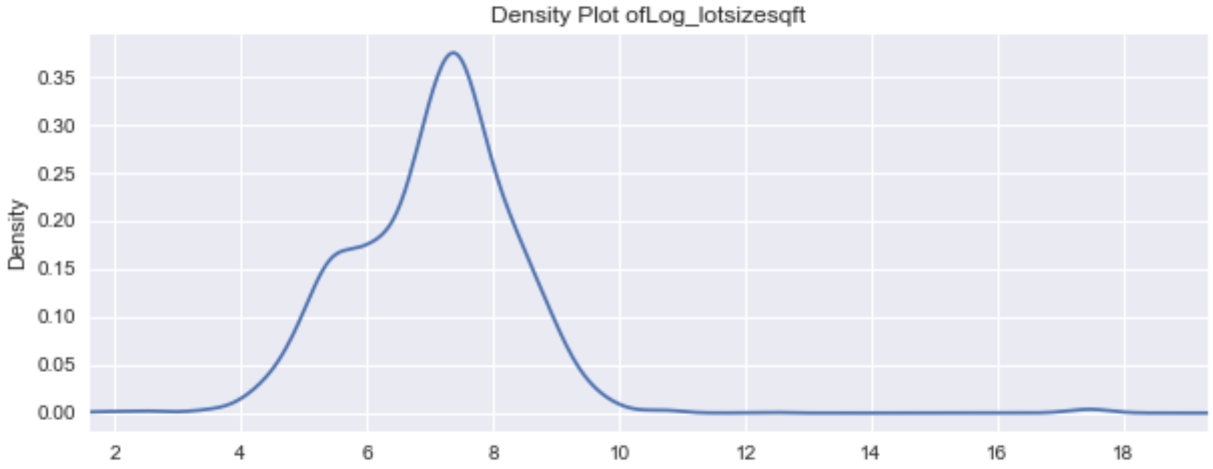
For example, performance and feature importance may differ by property sqft

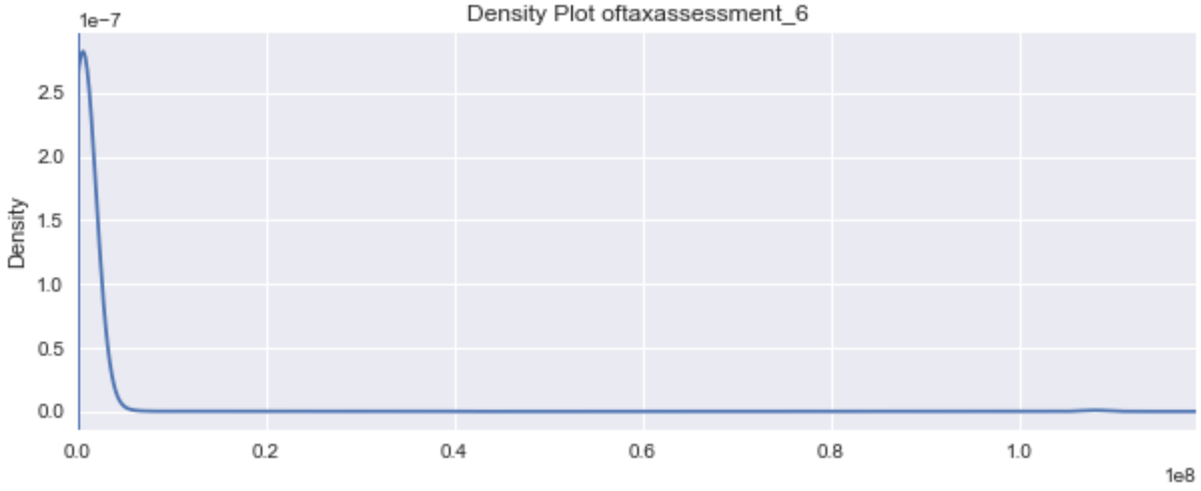
In [11]:

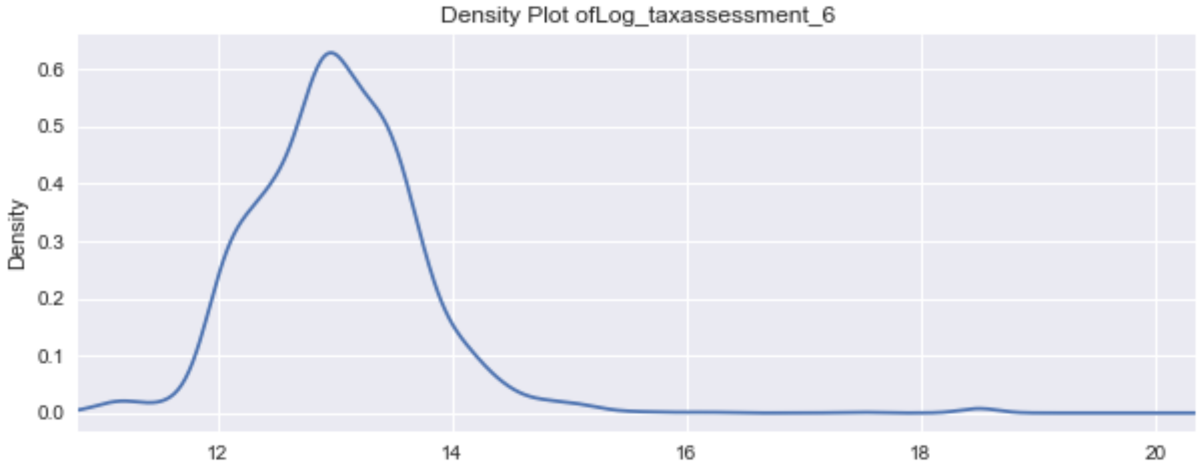
```
# Create Log form of lotsizesqft and taxassessment
merged5 = merged4
var = ('lotsizesqft', 'taxassessment_6')

for j in var:
    i = j
    i2 = 'Log_'+j
    merged5.loc[:,i2] = np.log(merged5[i])

    plt.figure(figsize=(10,8))
    plt.subplot(211)
    plt.xlim(merged5[i].min(), merged5[i].max()*1.1)
    plt.axvline(x=min)
    plt.axvline(x=max)
    ax = merged5[i].plot(kind='kde')
    plt.title('Density Plot of'+i)

    plt.figure(figsize=(10,8))
    plt.subplot(211)
    plt.xlim(merged5[i2].min(), merged5[i2].max()*1.1)
    plt.axvline(x=min)
    plt.axvline(x=max)
    ax = merged5[i2].plot(kind='kde')
    plt.title('Density Plot of'+i2)

    sns.plt.show()
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:297: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[key] = _infer_fill_value(value)
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:477: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[item] = s
```


Density Plot oflotsizesqft
Density
0.0000008
0.0000006
0.0000004
0.0000002
0.0000000
0
1
2
3
4
1e7


Density Plot ofLog_lotsizesqft
Density
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2
4
6
8
10
12
14
16
18


Density Plot oftaxassessment_6
1e−7
Density
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1e8



In [12]:

```
# Remove obs w/values above 99th percentile

numerics = list(merged5.select_dtypes(include=[np.number]).columns.values)
numerics.remove('zpid')
numerics.remove('target_2017')

for i in numerics:
    qmax = merged5[i].quantile(0.99)
    merged5[merged5[i] < qmax]
```

In [13]:
```
# Create flags for build date
merged5['built_after_2000'] = merged5['yearbuilt'] > 2000
merged5.loc[:,'built_after_2000'] = merged5['built_after_2000'].astype(int)

# Create flags for property size
merged5.loc[:,'sqft_lt1000'] = merged5['finishedsqft'] < 1000
merged5.loc[:,'sqft_lt1000'] = merged5['sqft_lt1000'].astype(int)

merged5.loc[:,'sqft_lt1500'] = merged5['finishedsqft'] < 1500
merged5.loc[:,'sqft_lt1500'] = merged5['sqft_lt1500'].astype(int)

merged5.loc[:,'sqft_lt2000'] = merged5['finishedsqft'] < 2000
merged5.loc[:,'sqft_lt2000'] = merged5['sqft_lt2000'].astype(int)

merged5.loc[:,'sqft_lt2500'] = merged5['finishedsqft'] < 2500
merged5.loc[:,'sqft_lt2500'] = merged5['sqft_lt2500'].astype(int)

merged5.loc[:,'sqft_ge2500'] = merged5['finishedsqft'] >= 2500
merged5.loc[:,'sqft_ge2500'] = merged5['sqft_ge2500'].astype(int)

# Create flags for ZHVI
merged5.loc[:,'ZHVI_lt400k'] = merged5['zhvi'] < 400000
merged5.loc[:,'ZHVI_lt400k'] = merged5['ZHVI_lt400k'].astype(int)

merged5.loc[:,'ZHVI_lt800k'] = merged5['zhvi'] < 800000
merged5.loc[:,'ZHVI_lt800k'] = merged5['ZHVI_lt800k'].astype(int)

merged5.loc[:,'ZHVI_gt1m'] = merged5['zhvi'] >= 1000000
merged5.loc[:,'ZHVI_gt1m'] = merged5['ZHVI_gt1m'].astype(int)
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/ipykernel/__mai
n__.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  from ipykernel import kernelapp as app
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:477: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[item] = s
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:297: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[key] = _infer_fill_value(value)
```

In [14]:

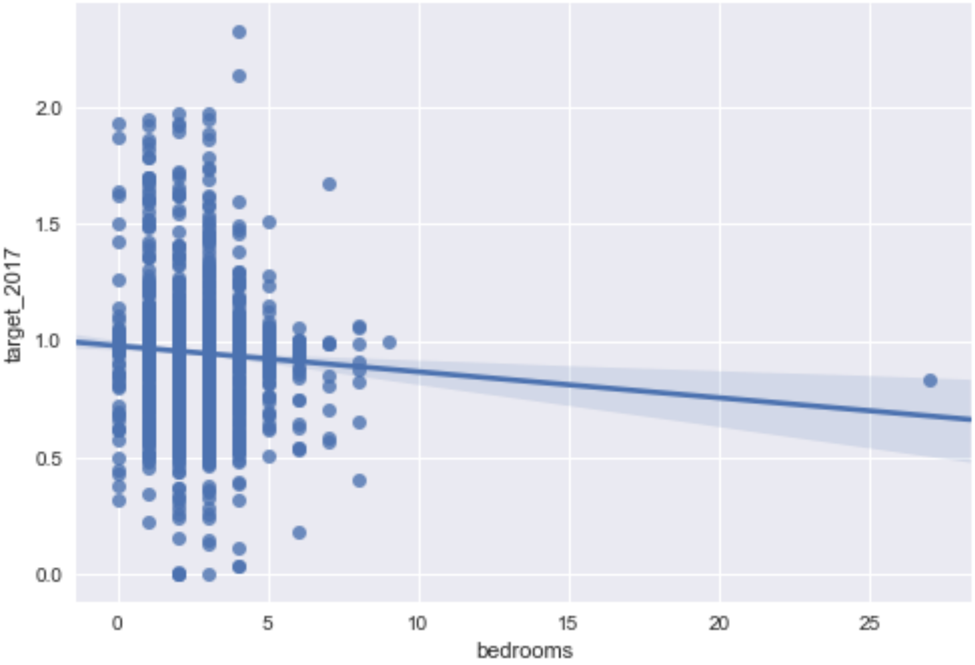
```
merged5.head(10)
```

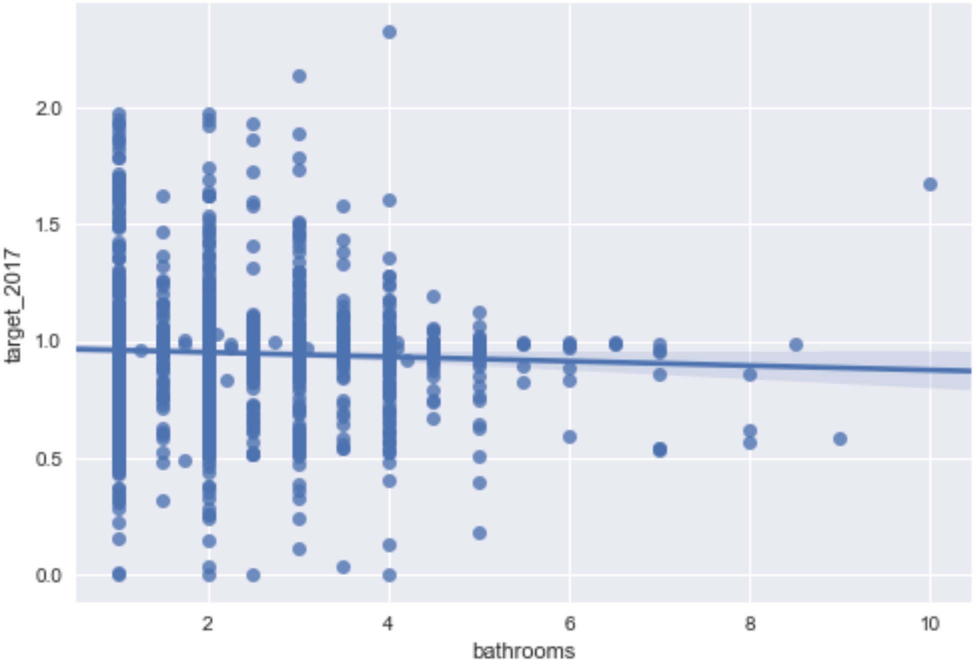
Out[14]:

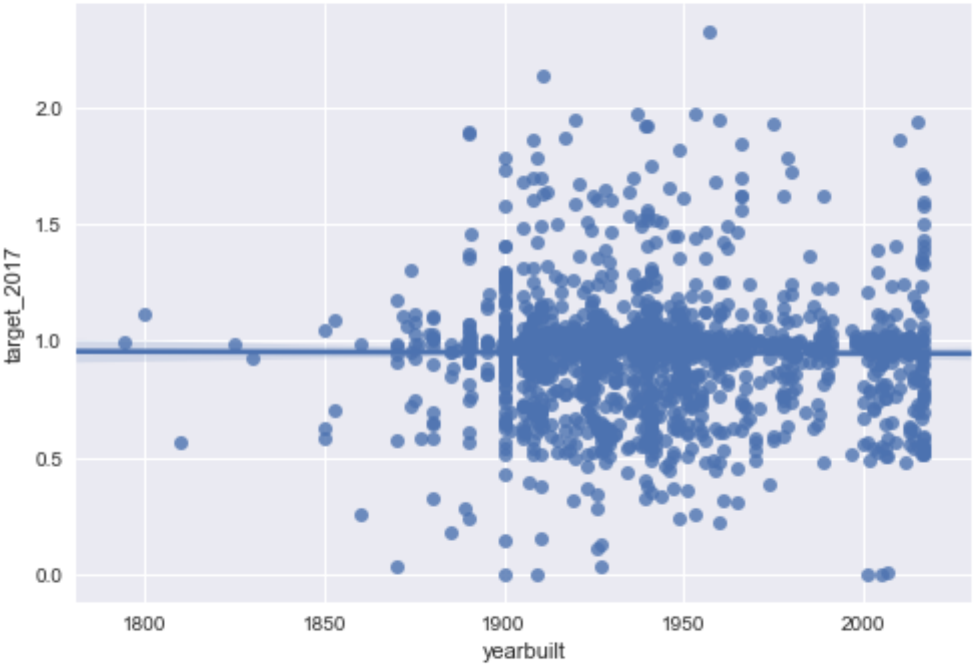
	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft
lastsolddate_6						
2017-01-18	4	7.0	2010	17777	10600000.0	10897
2017-06-30	2	2.0	1908	565	367380.0	946
2017-05-12	2	2.0	1938	1552	162050.0	832
2017-07-10	4	4.0	2017	756	523750.0	1700
2017-04-05	3	1.5	1946	1999	186130.0	1998
2017-05-01	2	1.0	1942	1564	435760.0	1175
2017-01-10	2	1.0	1942	1502	162280.0	960
2017-04-12	3	2.0	1920	5000	219310.0	1456
2017-06-05	3	2.0	1905	12000	315050.0	1802
2017-04-21	3	4.0	1927	3519	911010.0	2500

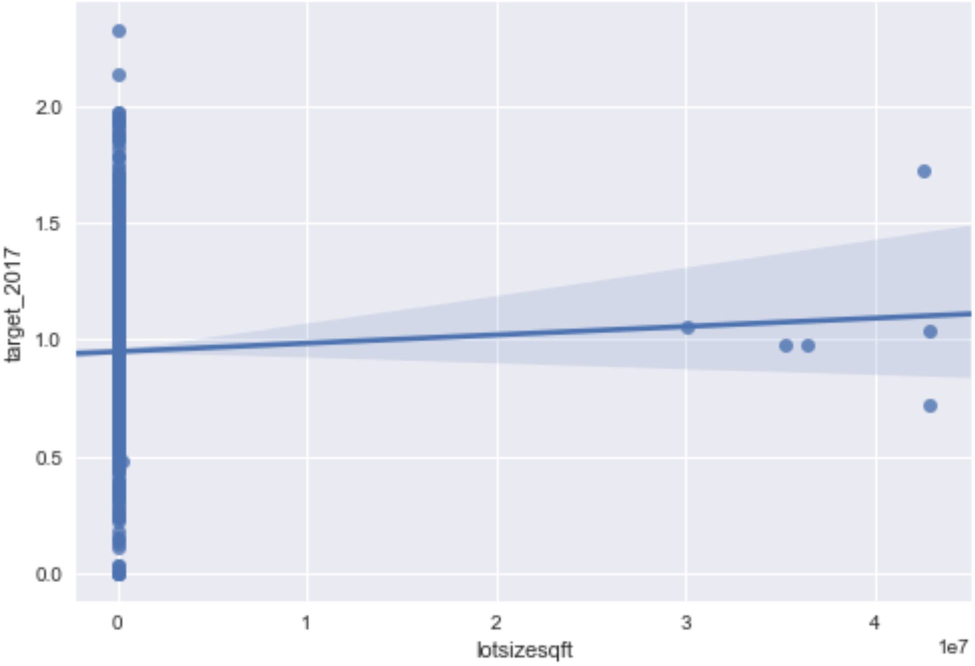
10 rows × 65 columns

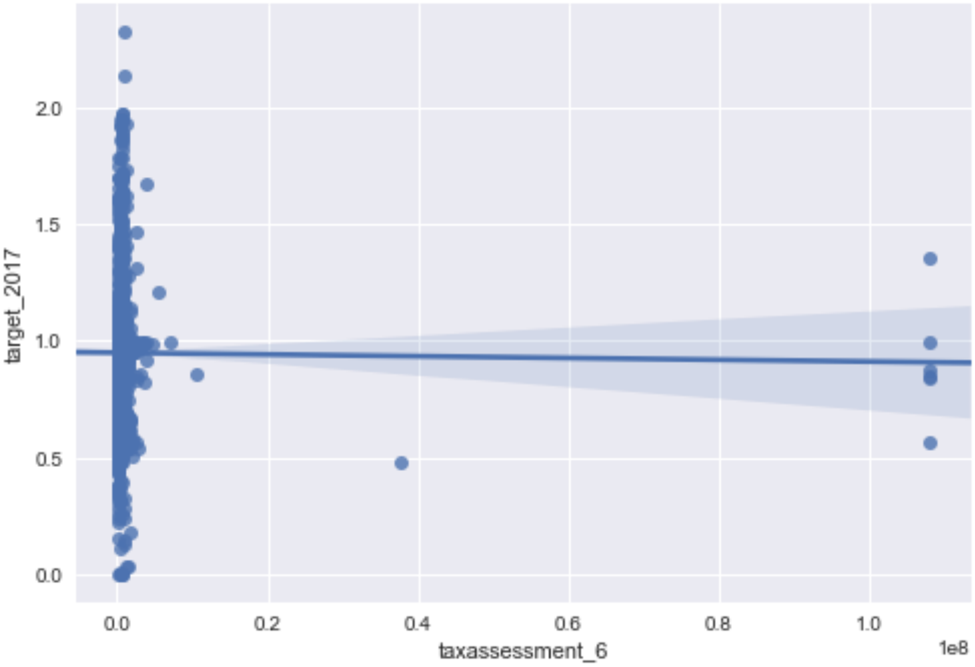
```
In [15]: # Plot metrics against the target

         numerics = list(merged5.select_dtypes(include=[np.number]).columns.values)
         numerics.remove('zpid')
         numerics.remove('target_2017')

         for i in numerics:

             g = sns.regplot(x=i, y="target_2017", data=merged5)
             sns.plt.show()
```

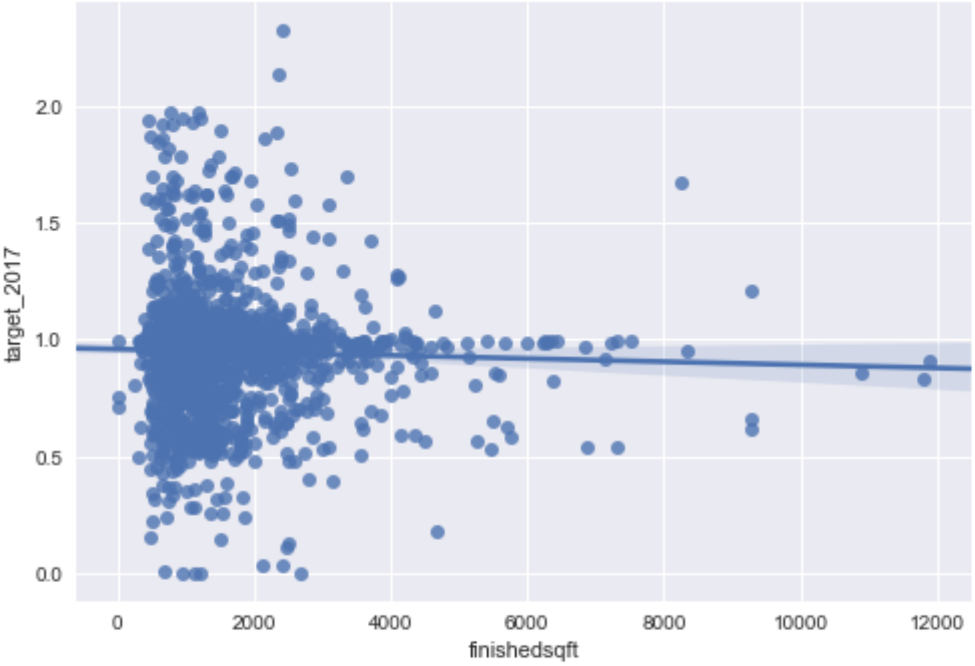


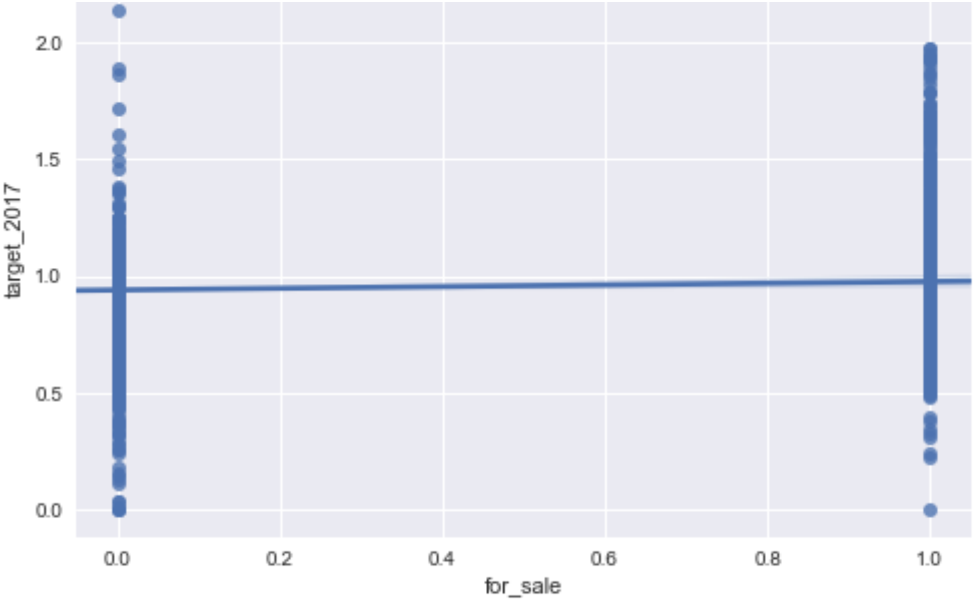


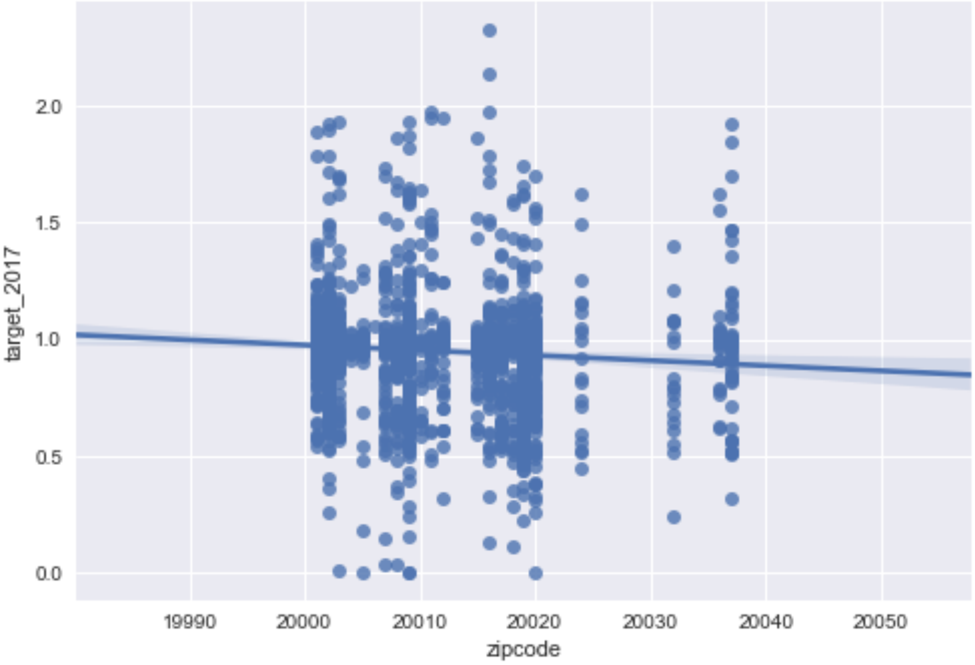

target_2017
0.0
0.5
1.0
1.5
2.0
1800
1850
1900
1950
2000
yearbuilt

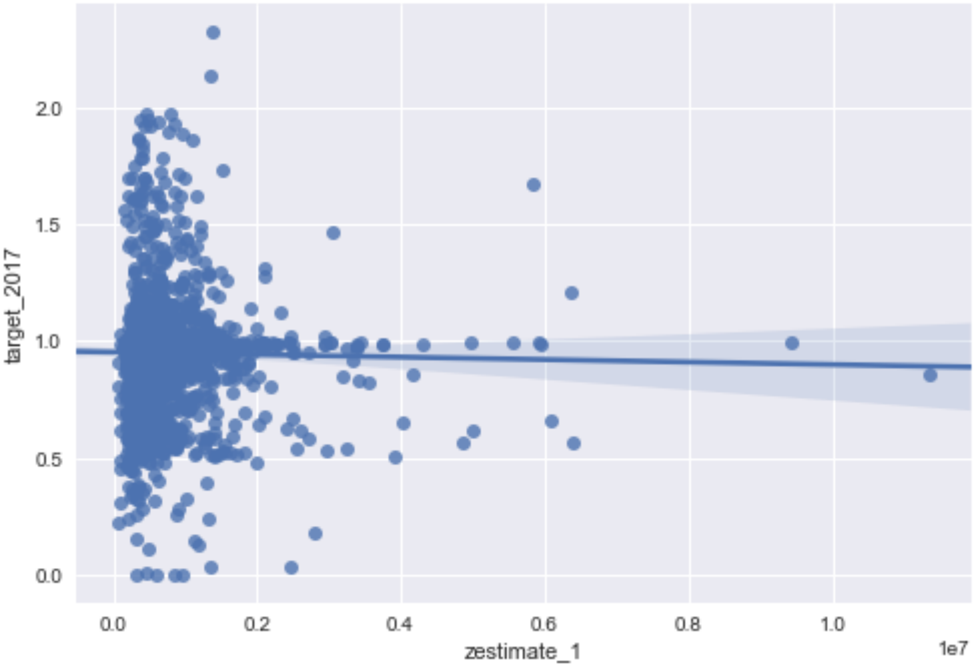

target_2017
0.0
0.5
1.0
1.5
2.0
0
1
2
3
4
lotsizesqft
1e7

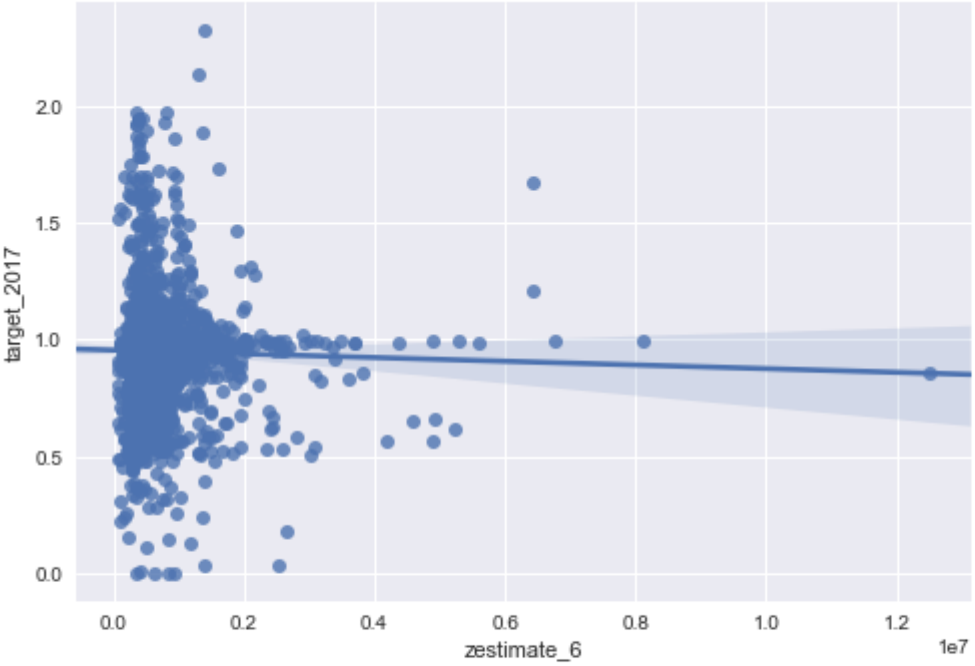

target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
1e8
taxassessment_6

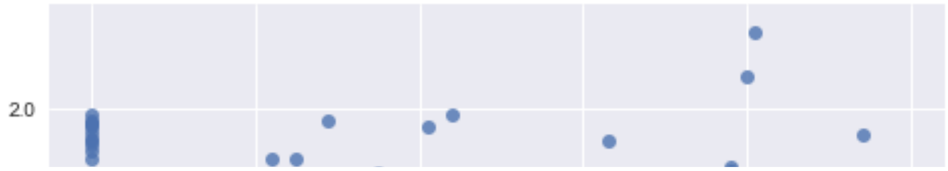

target_2017
0.0
0.5
1.0
1.5
2.0
0
2000
4000
6000
8000
10000
12000
finishedsqft

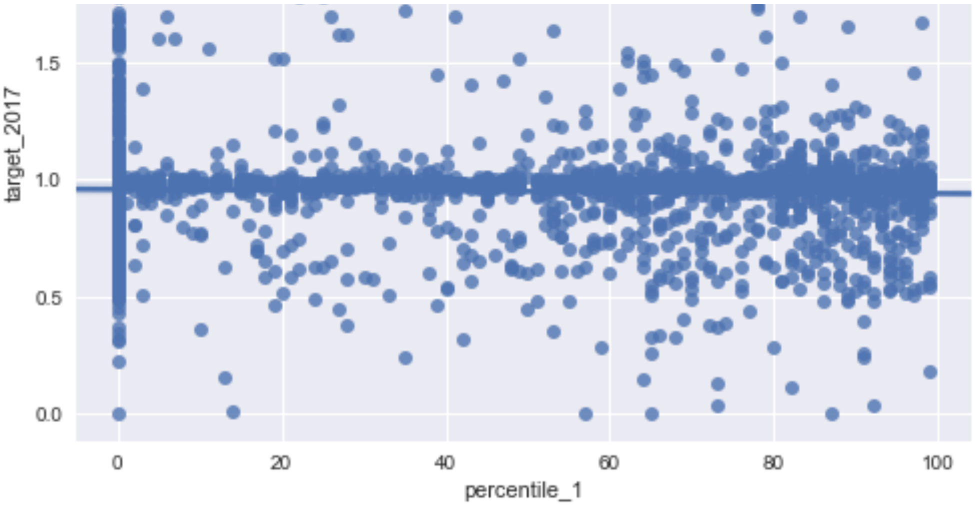

target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
for_sale

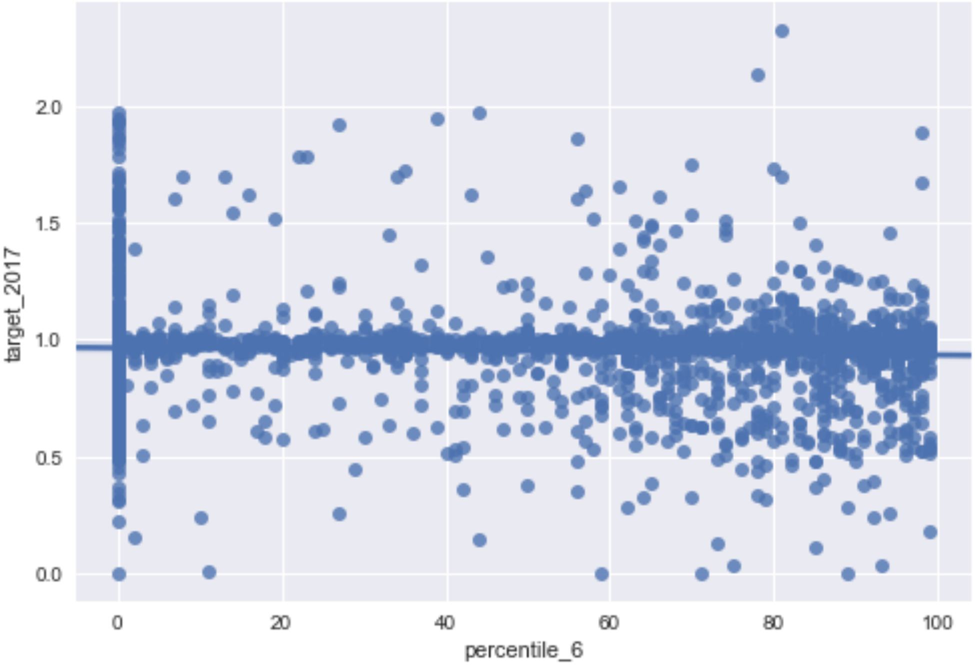

target_2017
0.0
0.5
1.0
1.5
2.0
19990
20000
20010
20020
20030
20040
20050
zipcode

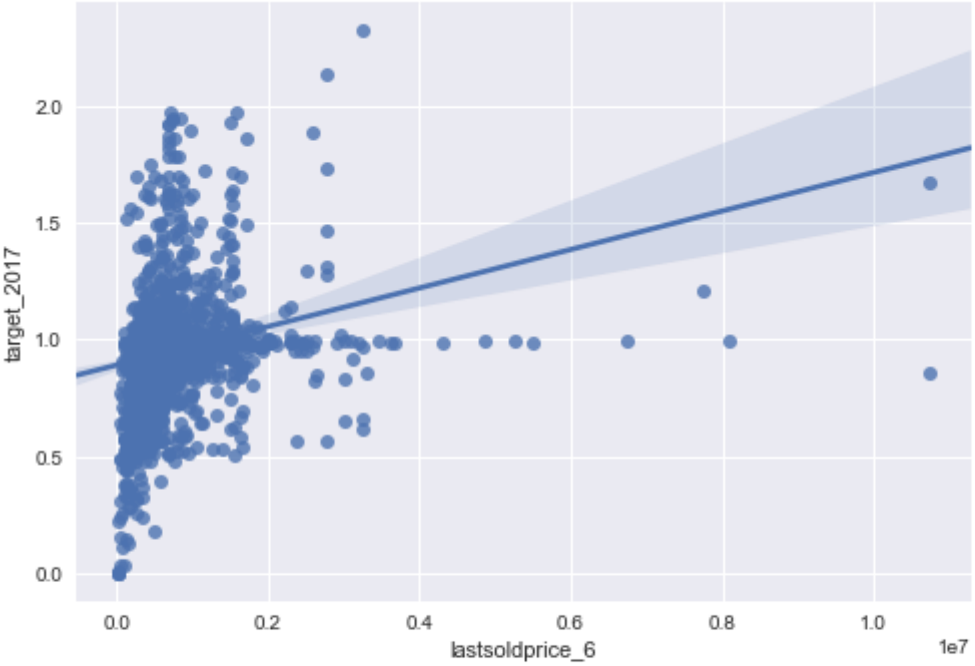

2.0
1.5
1.0
0.5
0.0
target_2017
0.0
0.2
0.4
0.6
0.8
1.0
zestimate_1
1e7

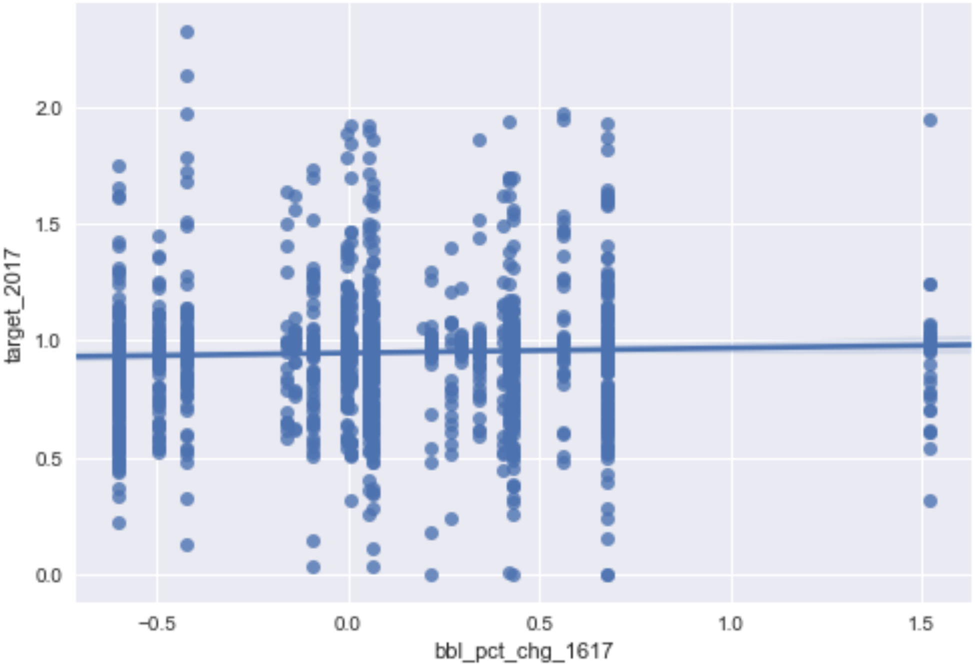

2.0
1.5
1.0
0.5
0.0
target_2017
0.0
0.2
0.4
0.6
0.8
1.0
1.2
zestimate_6
1e7

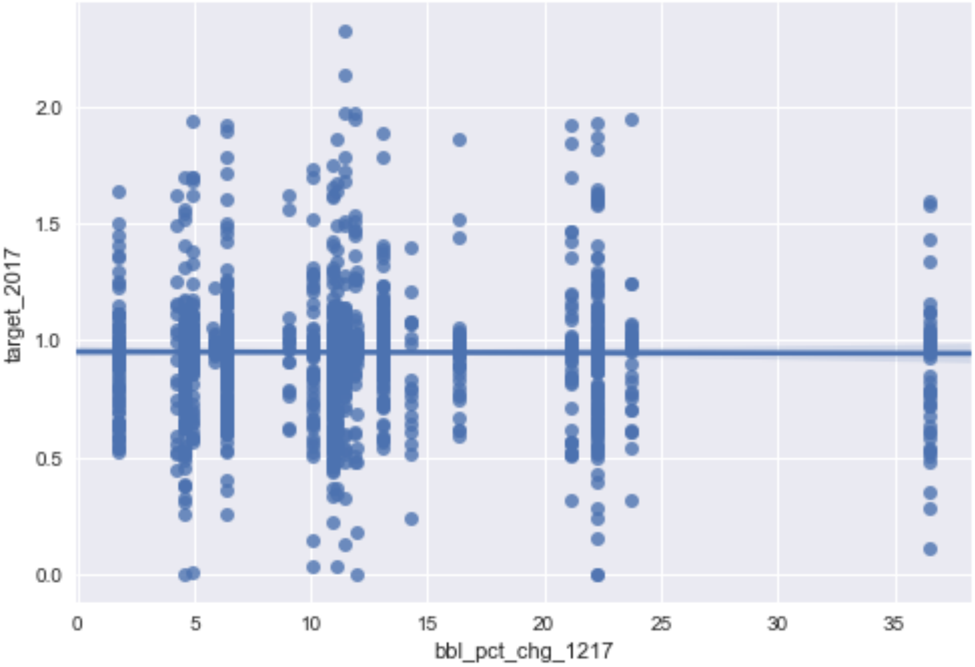

2.0

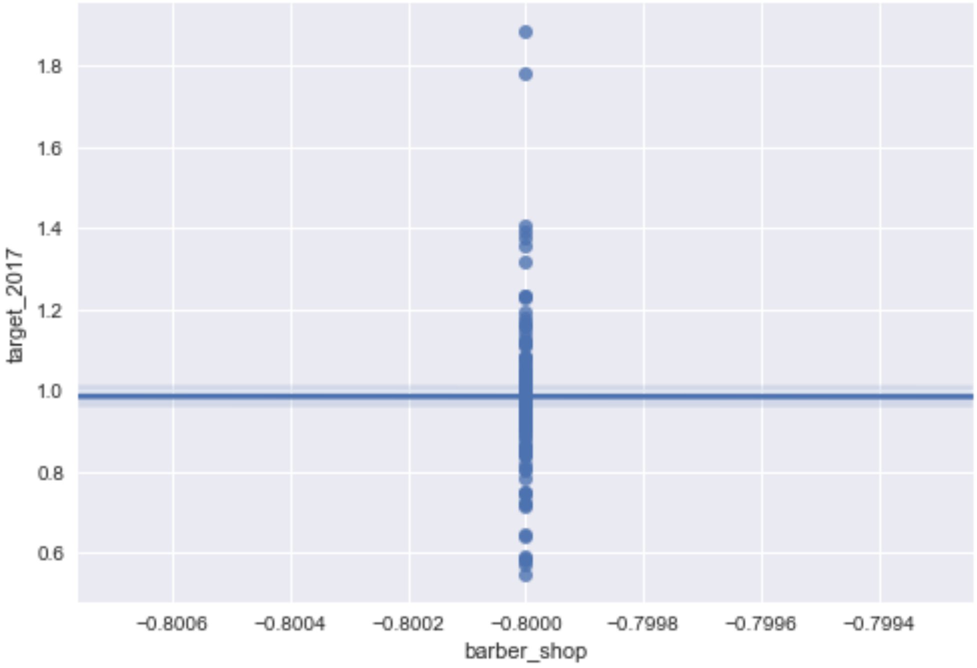

target_2017
0.0
0.5
1.0
1.5
0
20
40
60
80
100
percentile_1

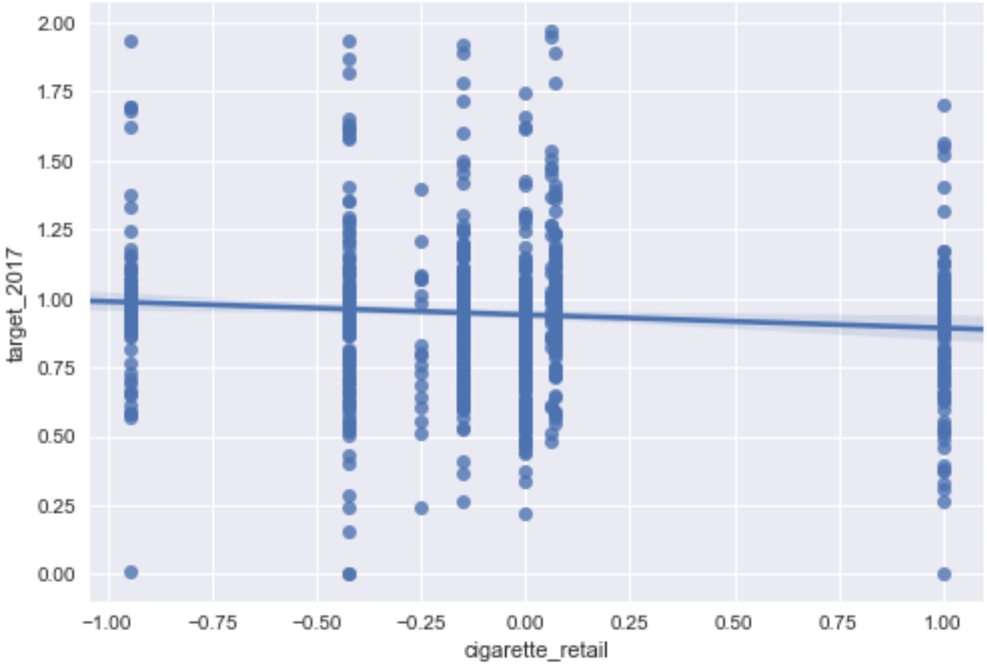

target_2017
0.0
0.5
1.0
1.5
2.0
0
20
40
60
80
100
percentile_6

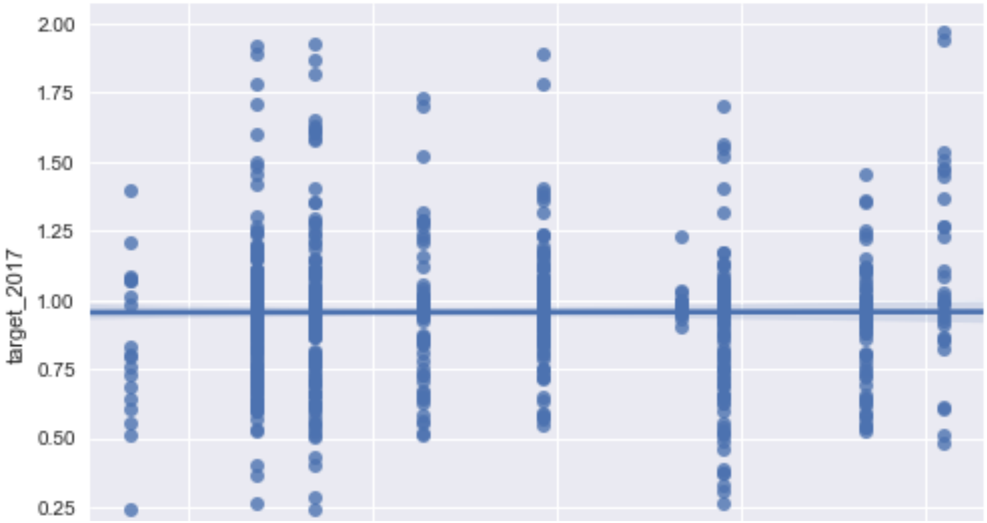

target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
1e7
lastsoldprice_6

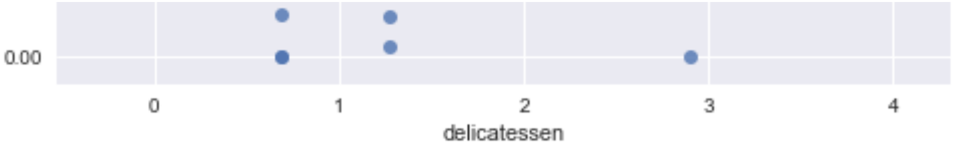

target_2017
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0
1.5
bbl_pct_chg_1617

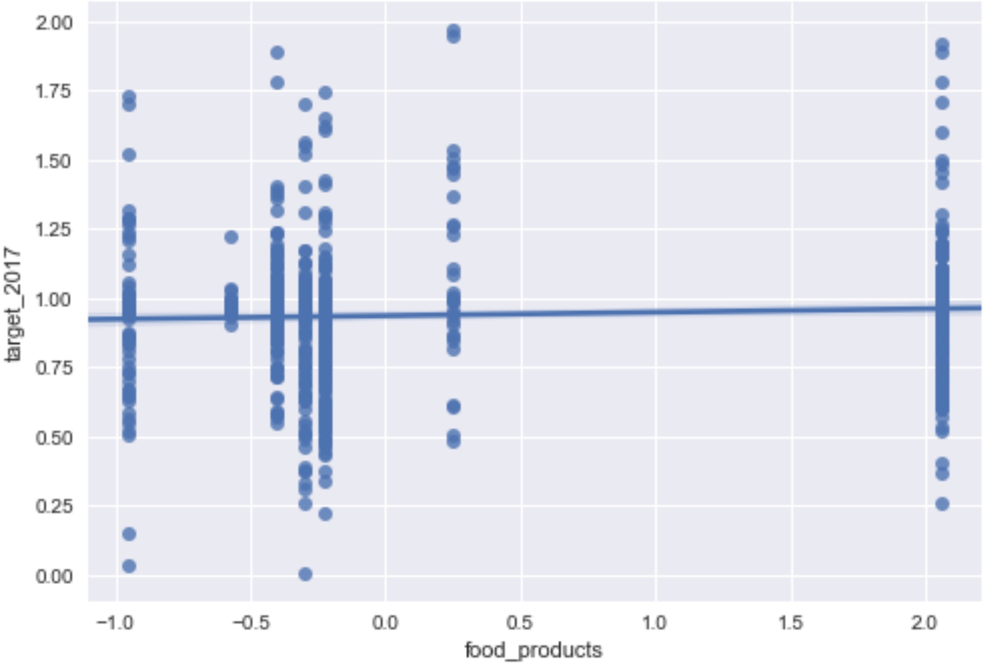

target_2017
0.0
0.5
1.0
1.5
2.0
0
5
10
15
20
25
30
35
bbl_pct_chg_1217

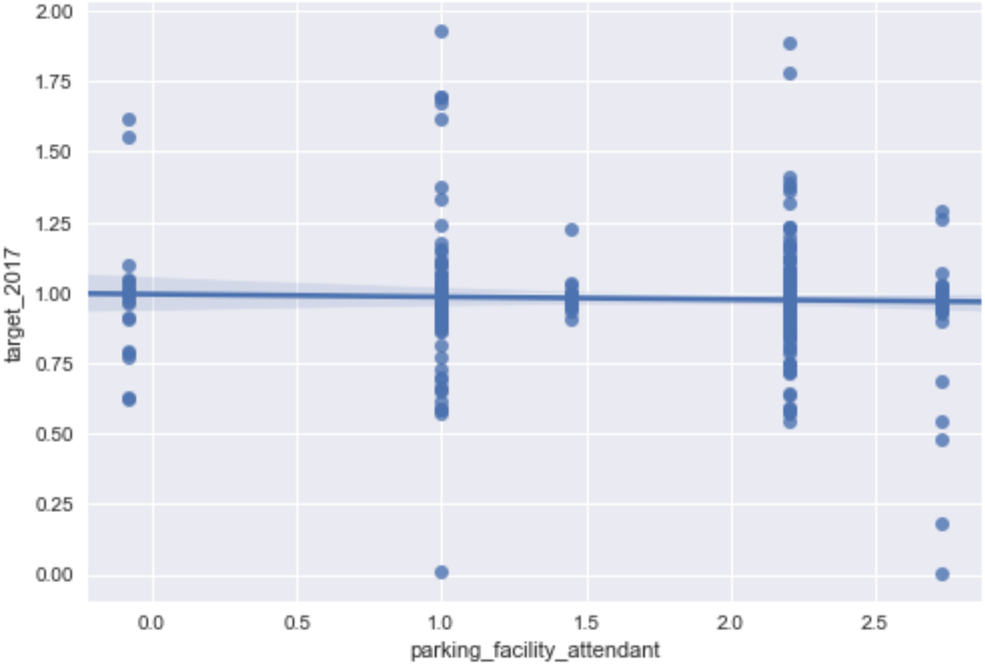

target_2017
1.8
1.6
1.4
1.2
1.0
0.8
0.6
−0.8006
−0.8004
−0.8002
−0.8000
−0.7998
−0.7996
−0.7994
barber_shop


target_2017
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
cigarette_retail

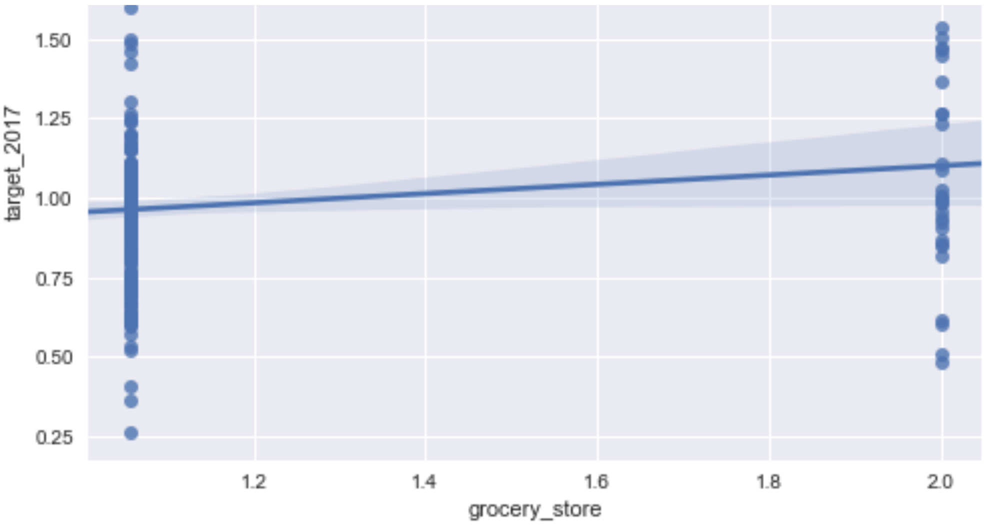

target_2017
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25

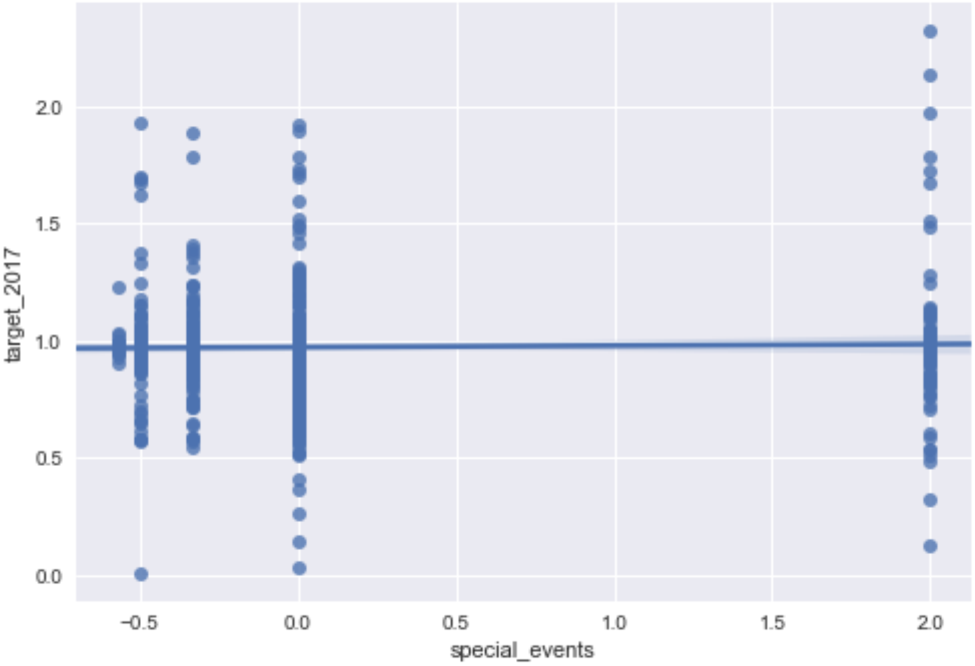

0.00
0
1
2
3
4
delicatessen

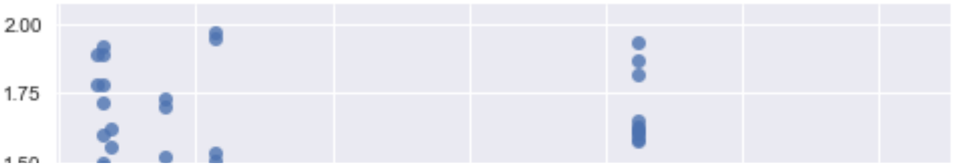

target_2017
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
food_products

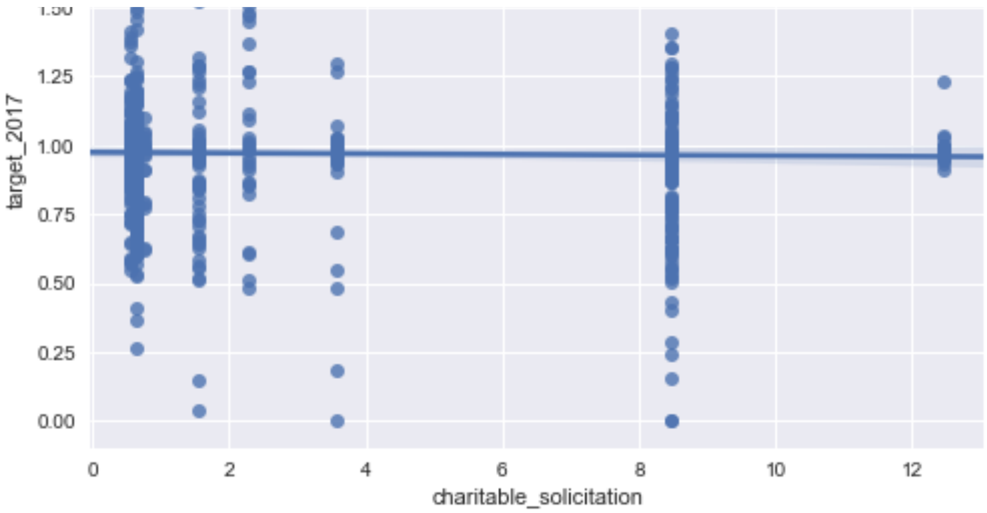

target_2017
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.5
2.0
2.5
parking_facility_attendant

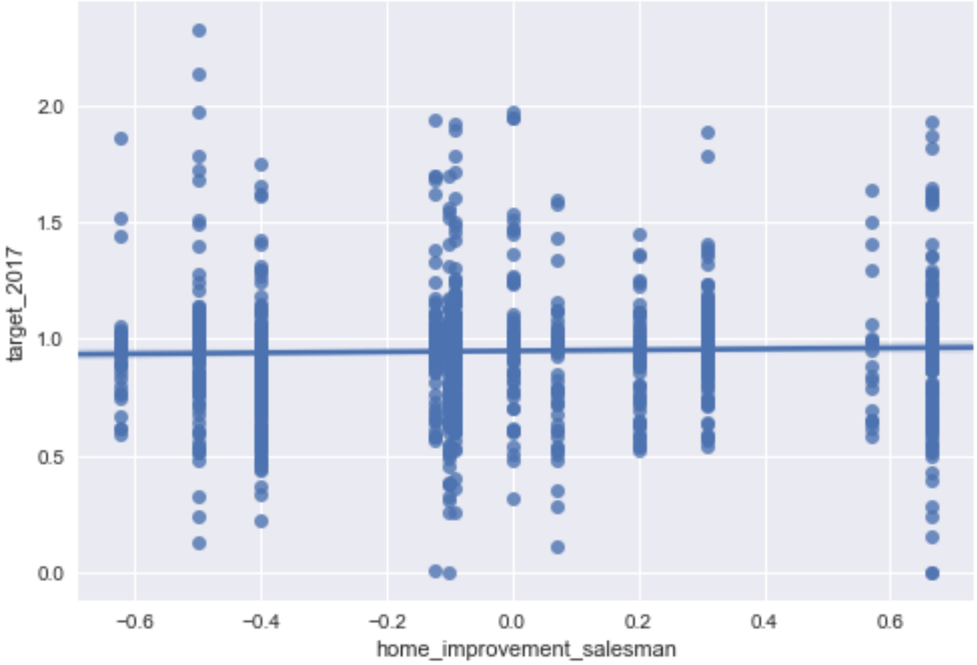
2.00
1.75

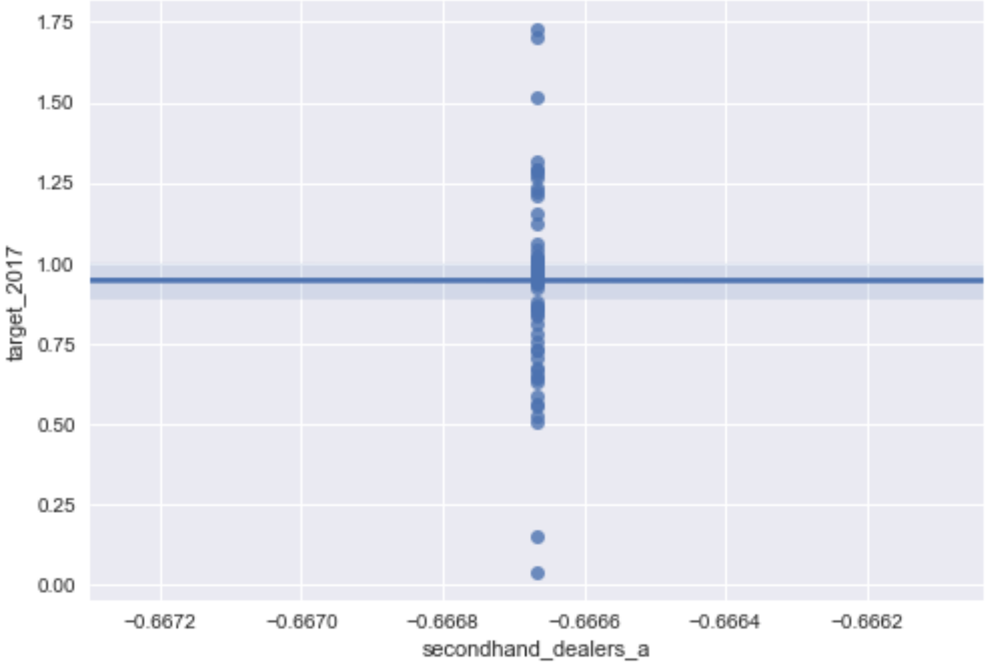

target_2017
1.50
1.25
1.00
0.75
0.50
0.25
1.2
1.4
1.6
1.8
2.0
grocery_store

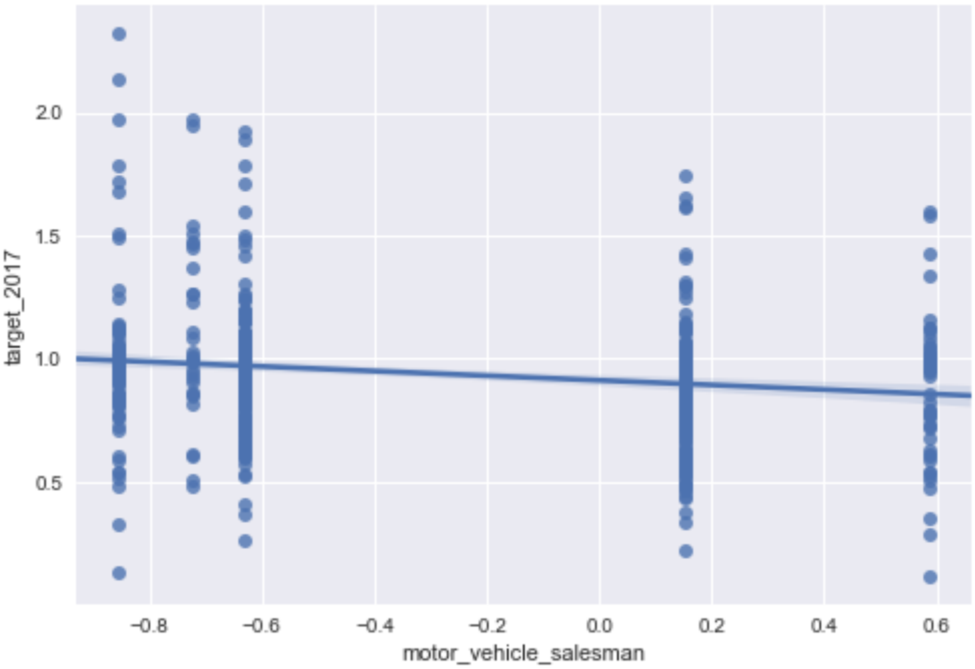

target_2017
2.0
1.5
1.0
0.5
0.0
−0.5
0.0
0.5
1.0
1.5
2.0
special_events

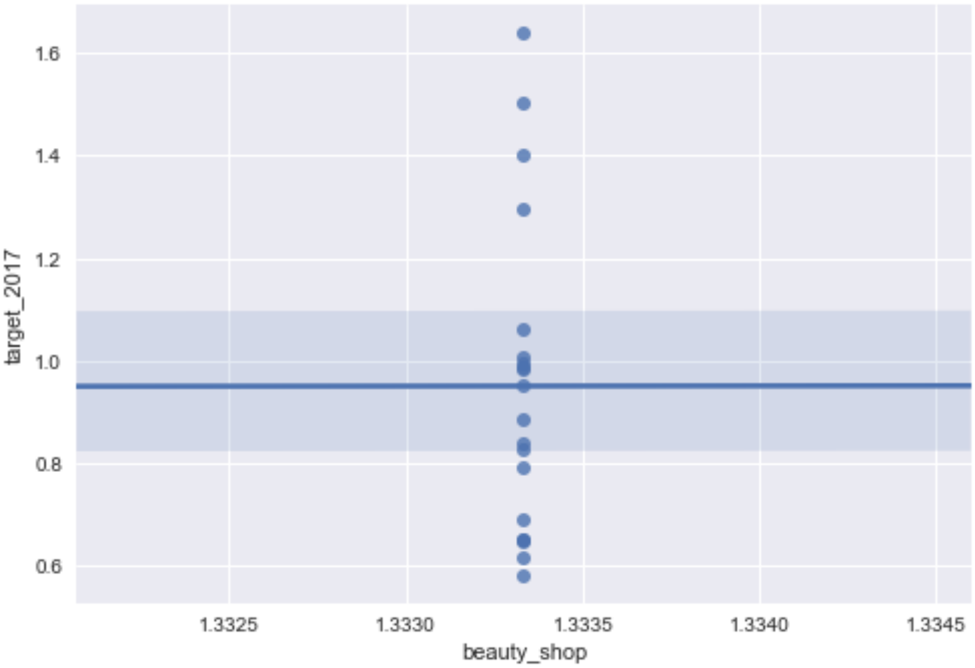

2.00
1.75
1.50

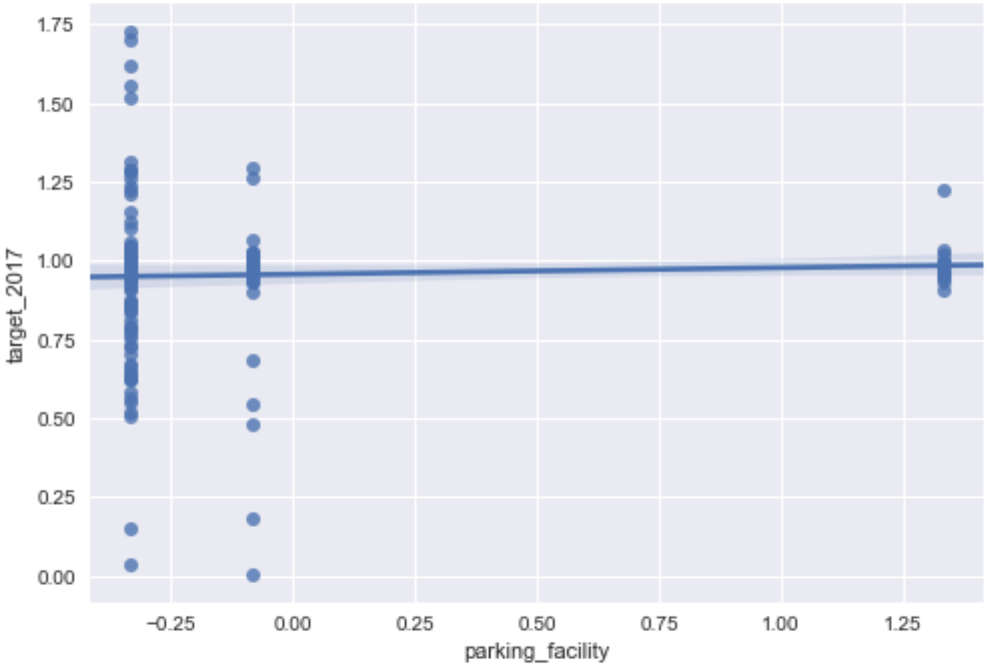

1.50
1.25
1.00
0.75
0.50
0.25
0.00
target_2017
0
2
4
6
8
10
12
charitable_solicitation

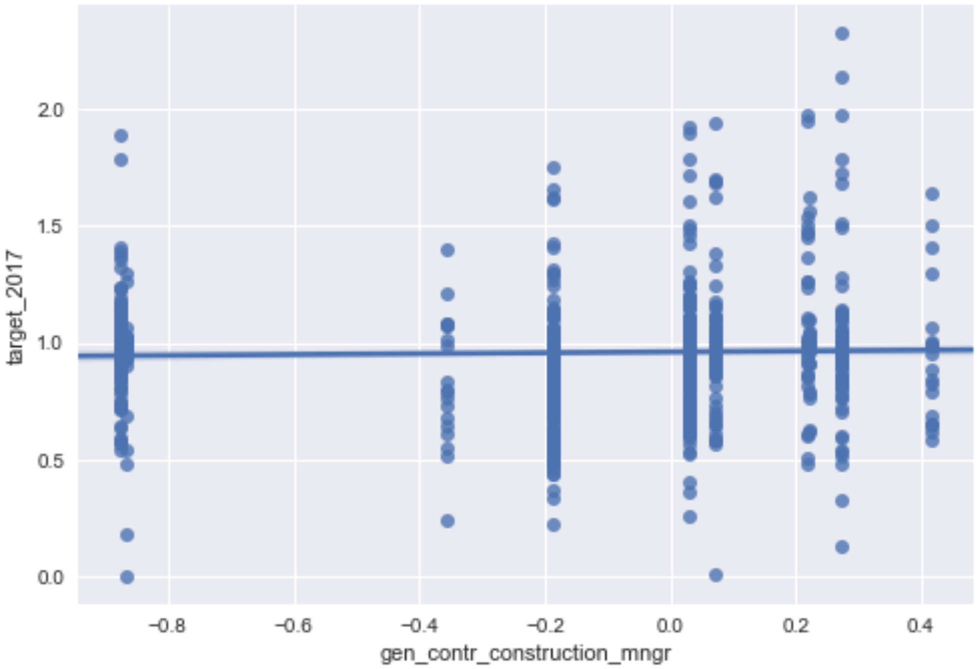

2.0
1.5
1.0
0.5
0.0
target_2017
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
home_improvement_salesman


1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
target_2017
−0.6672
−0.6670
−0.6668
−0.6666
−0.6664
−0.6662
secondhand_dealers_a

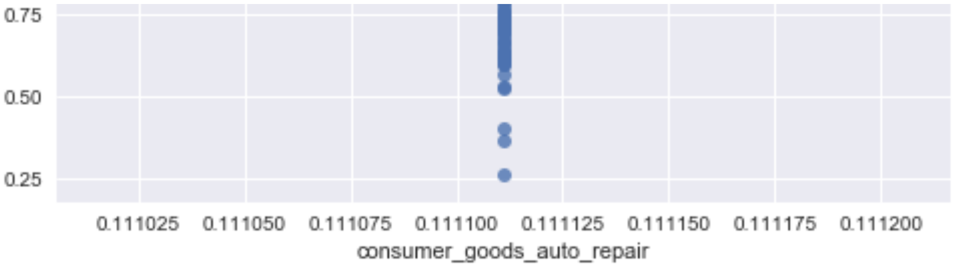

target_2017
2.0
1.5
1.0
0.5
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
motor_vehicle_salesman

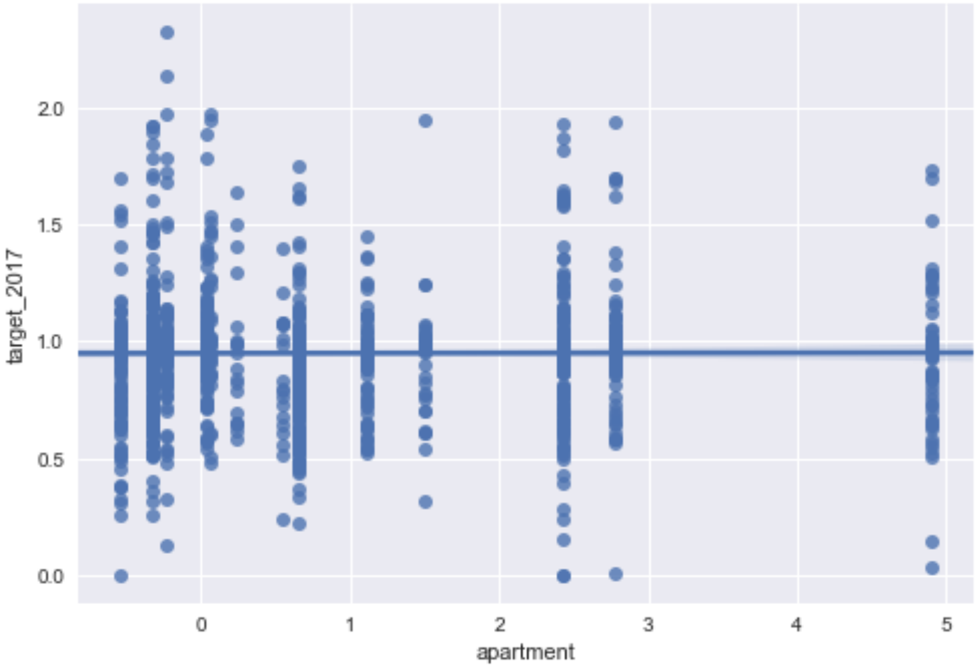

target_2017
1.6
1.4
1.2
1.0
0.8
0.6
1.3325
1.3330
1.3335
1.3340
1.3345
beauty_shop

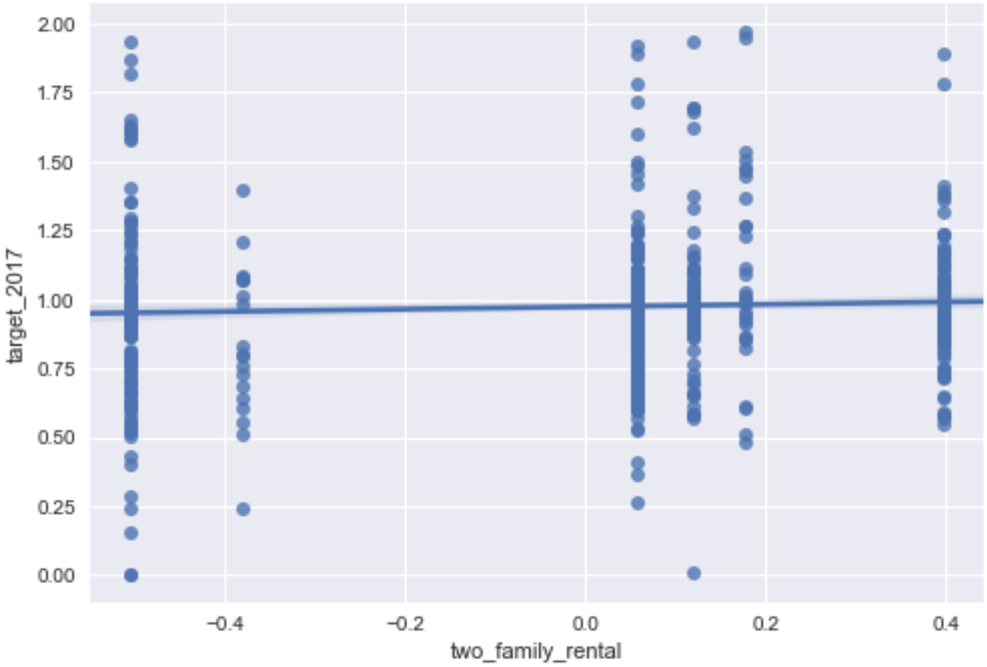

target_2017
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
parking_facility

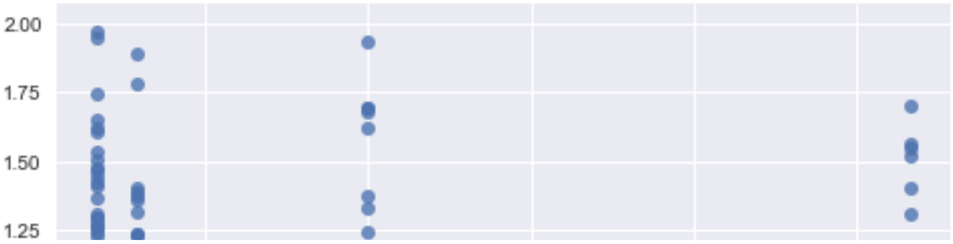

target_2017
0.0
0.5
1.0
1.5
2.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
gen_contr_construction_mngr

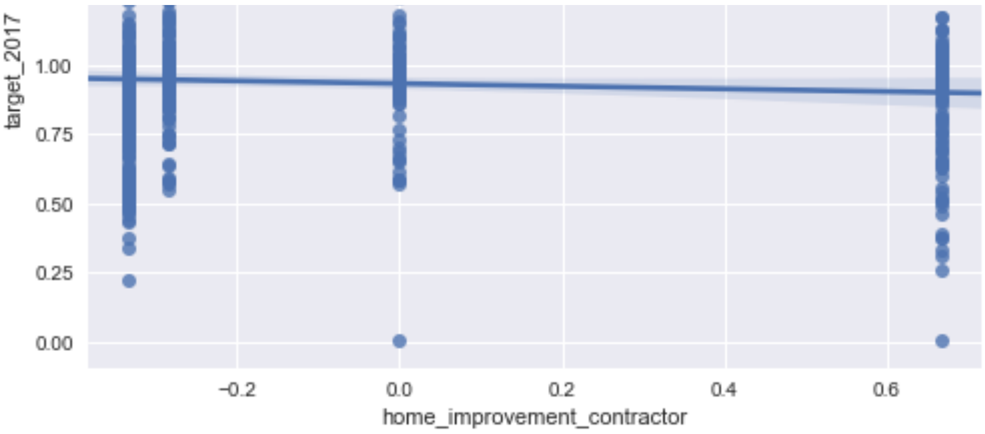
target_2017
1.00
1.25
1.50
1.75
2.00

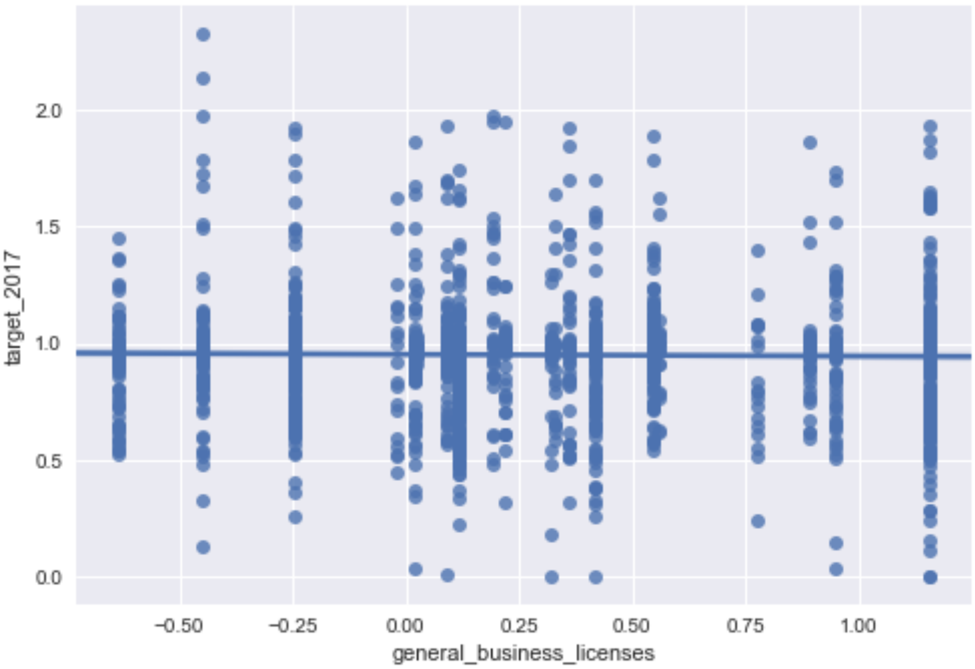

0.75
0.50
0.25
0.111025
0.111050
0.111075
0.111100
0.111125
0.111150
0.111175
0.111200
consumer_goods_auto_repair

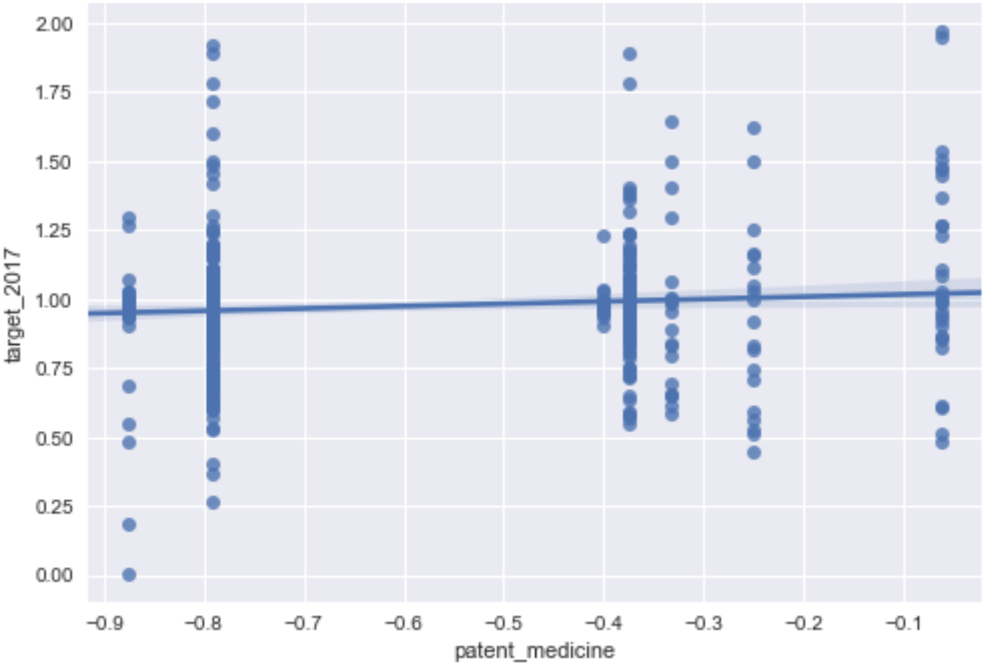

target_2017
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5
apartment

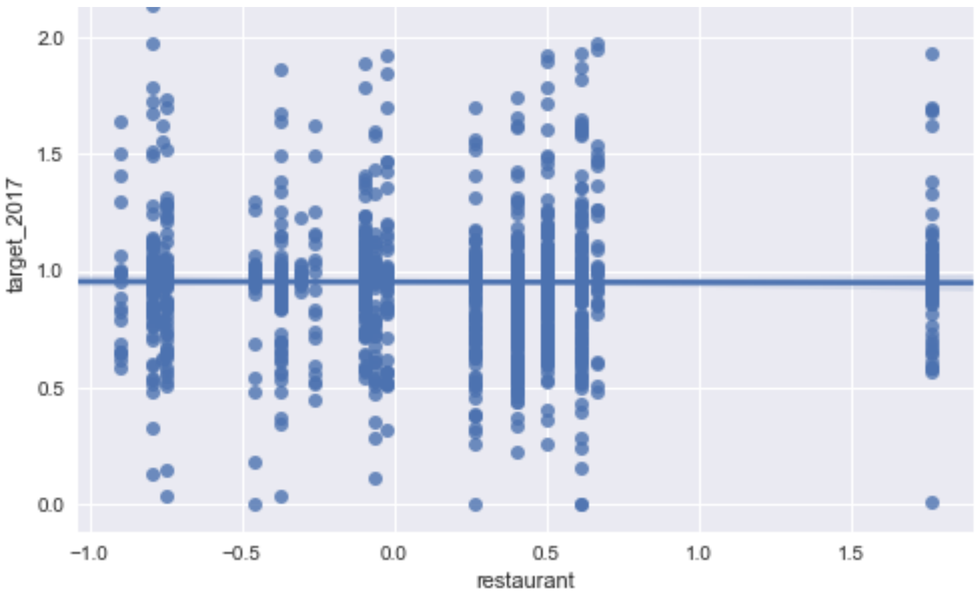

target_2017
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.4
−0.2
0.0
0.2
0.4
two_family_rental

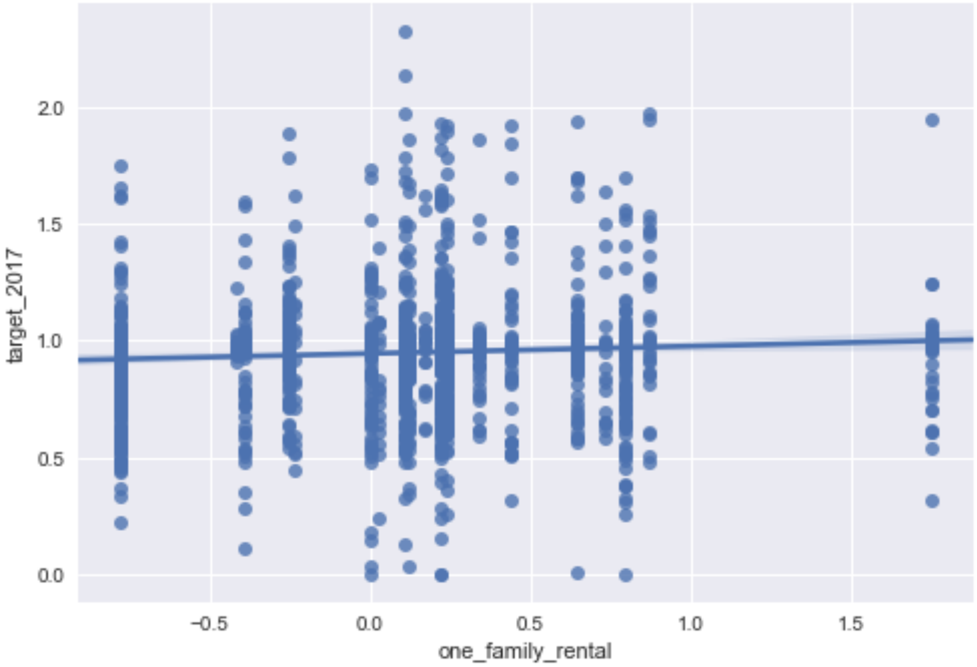

2.00
1.75
1.50
1.25

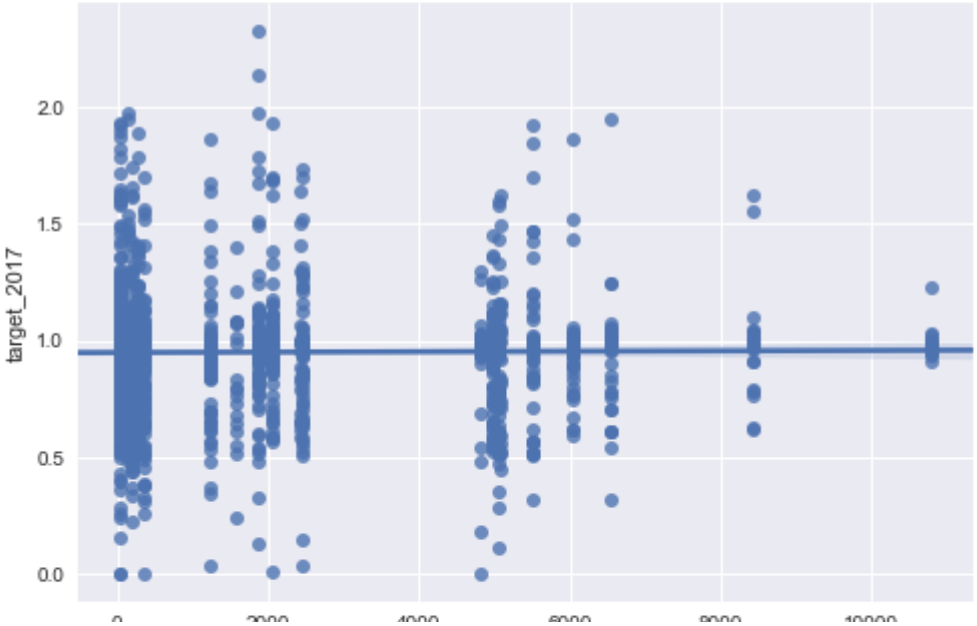

target_2017
0.00
0.25
0.50
0.75
1.00
−0.2
0.0
0.2
0.4
0.6
home_improvement_contractor

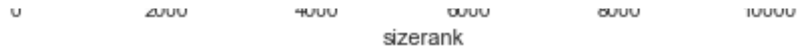

target_2017
0.0
0.5
1.0
1.5
2.0
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
general_business_licenses

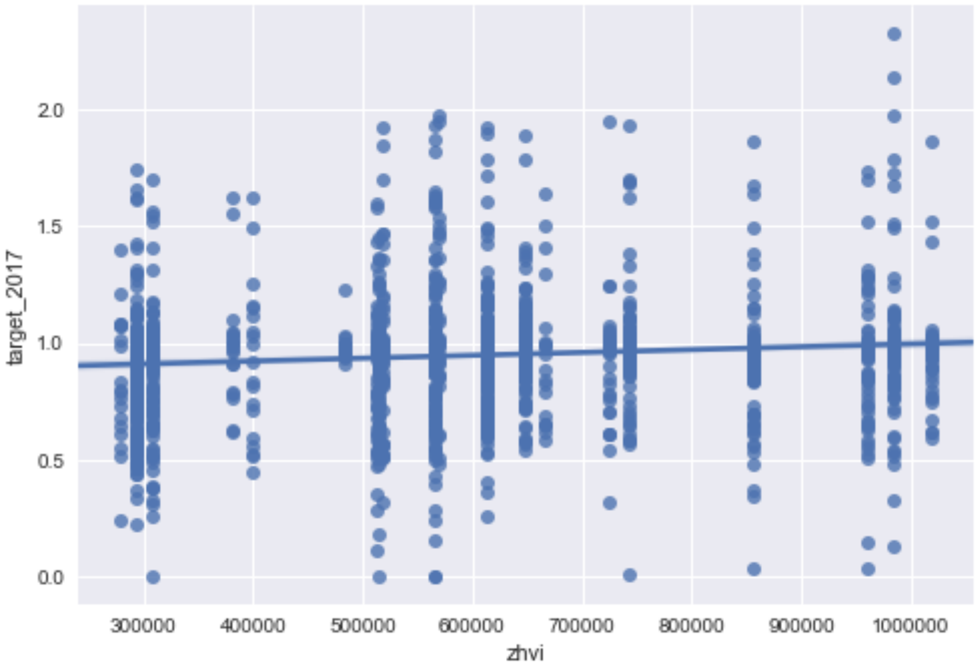

target_2017
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
−0.9
−0.8
−0.7
−0.6
−0.5
−0.4
−0.3
−0.2
−0.1
patent_medicine

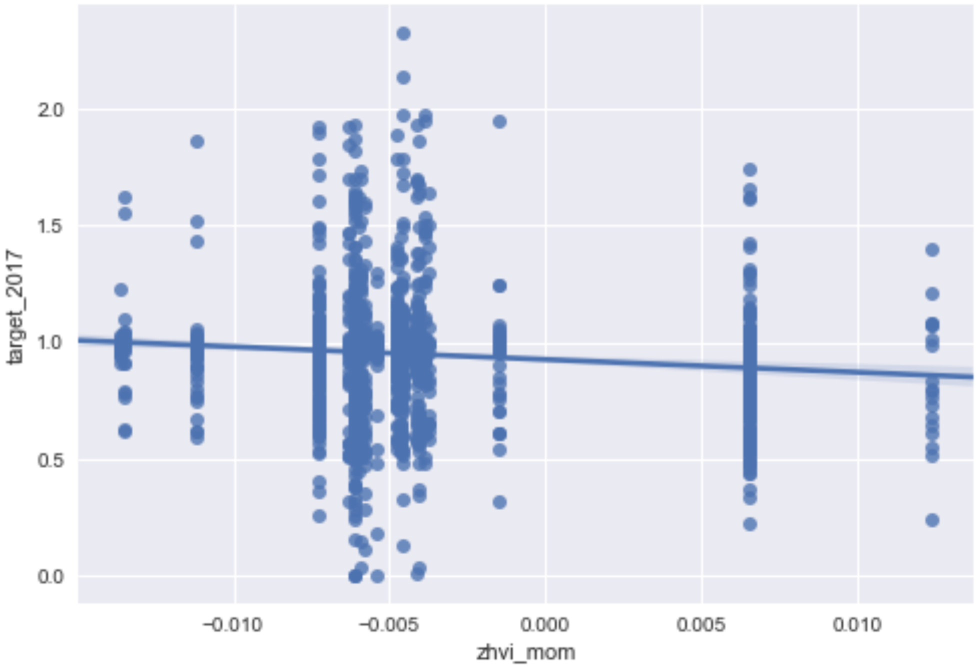

target_2017
0.0
0.5
1.0
1.5
2.0
−1.0
−0.5
0.0
0.5
1.0
1.5
restaurant

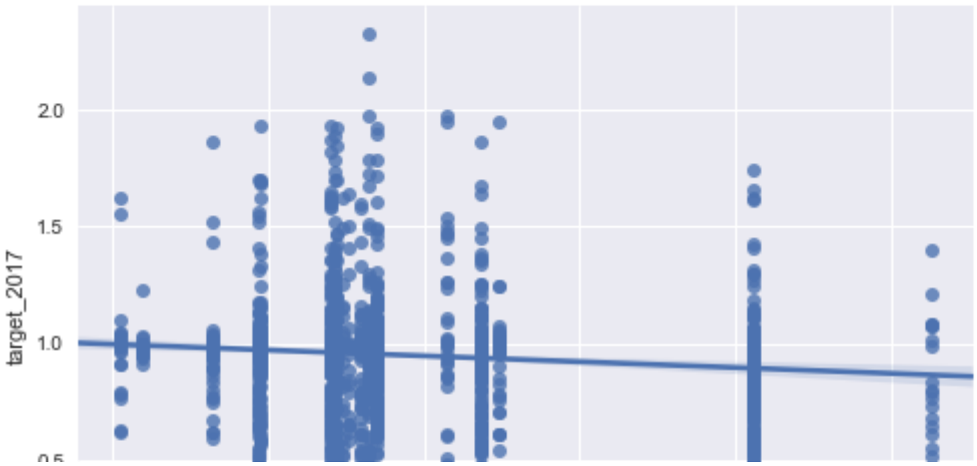

target_2017
0.0
0.5
1.0
1.5
2.0
−0.5
0.0
0.5
1.0
1.5
one_family_rental

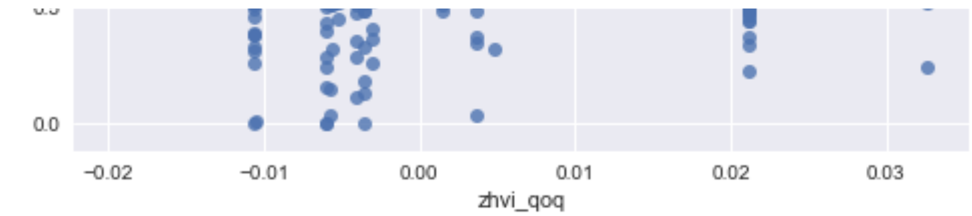

target_2017
0.0
0.5
1.0
1.5
2.0
0
2000
4000
6000
8000
10000

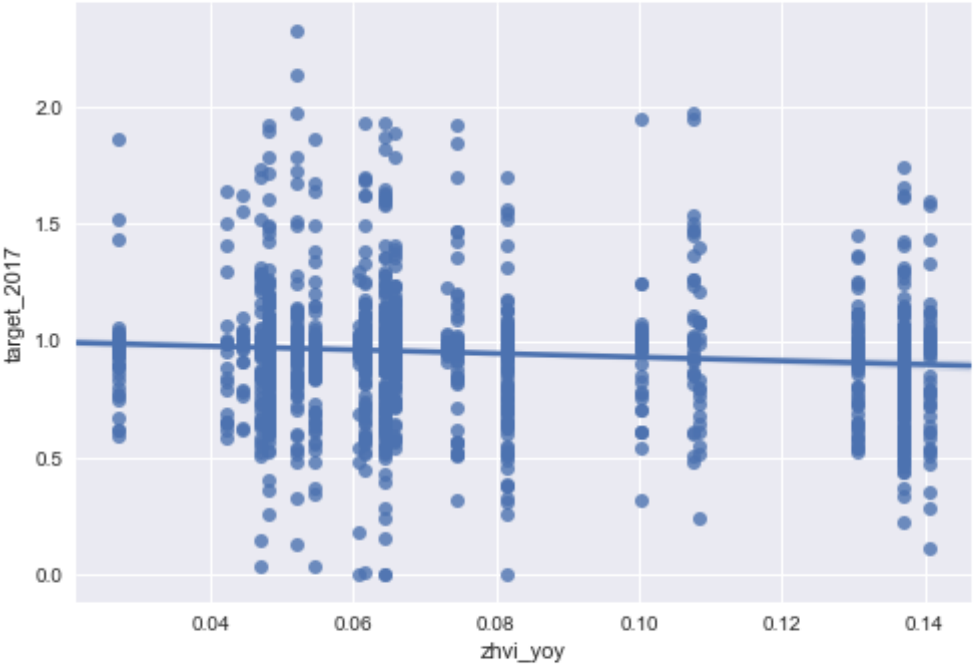

0
2000
4000
6000
8000
10000
sizerank

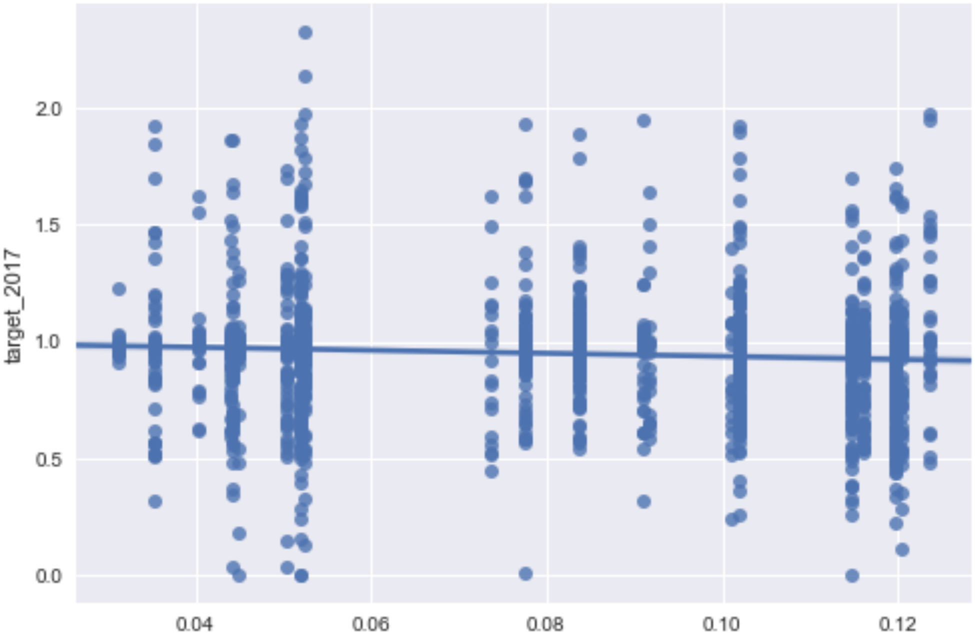

target_2017
0.0
0.5
1.0
1.5
2.0
300000
400000
500000
600000
700000
800000
900000
1000000
zhvi


target_2017
0.0
0.5
1.0
1.5
2.0
−0.010
−0.005
0.000
0.005
0.010
zhvi_mom

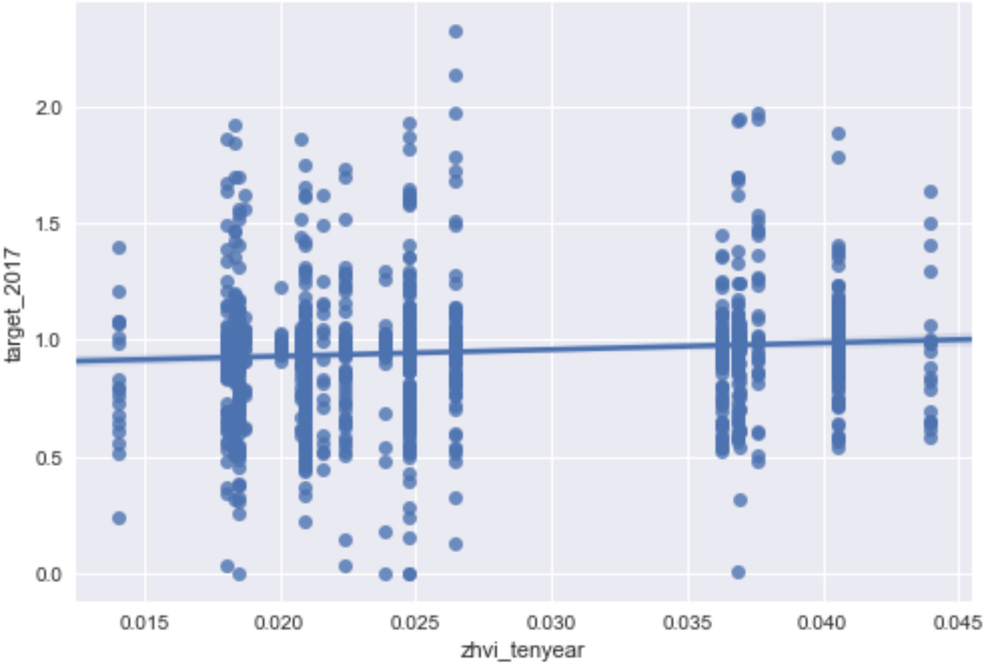

target_2017
0.5
1.0
1.5
2.0

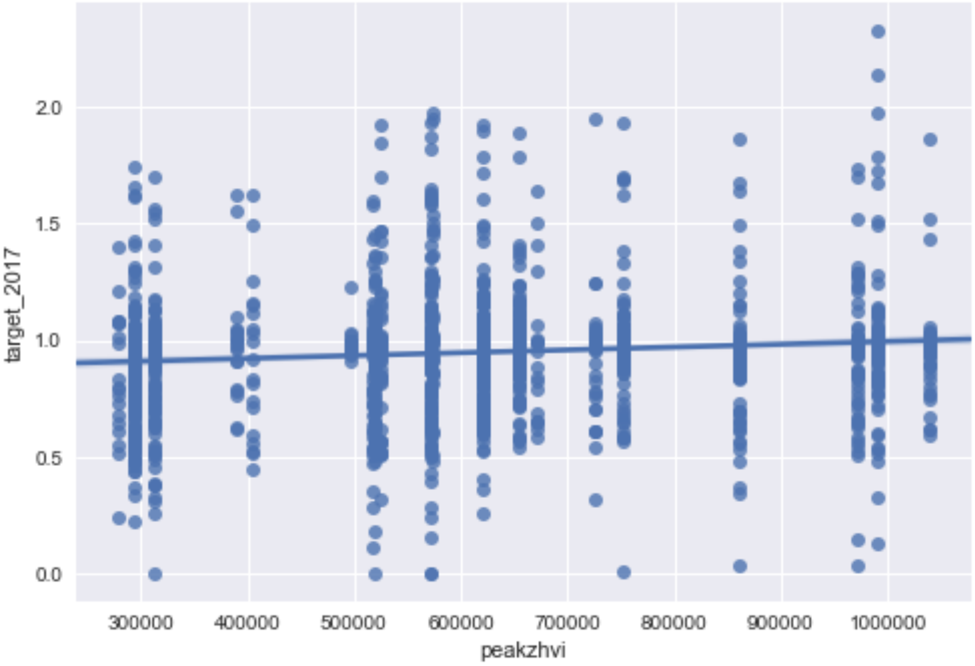

0.5
0.0
−0.02
−0.01
0.00
0.01
0.02
0.03
zhvi_qoq

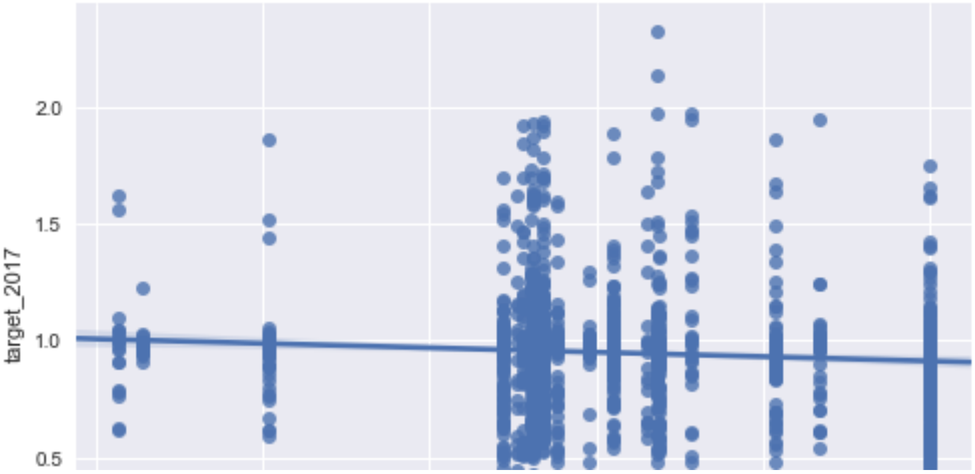

target_2017
2.0
1.5
1.0
0.5
0.0
0.04
0.06
0.08
0.10
0.12
0.14
zhvi_yoy

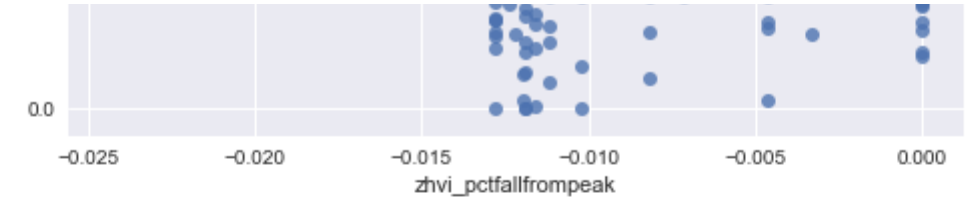

target_2017
2.0
1.5
1.0
0.5
0.0
0.04
0.06
0.08
0.10
0.12

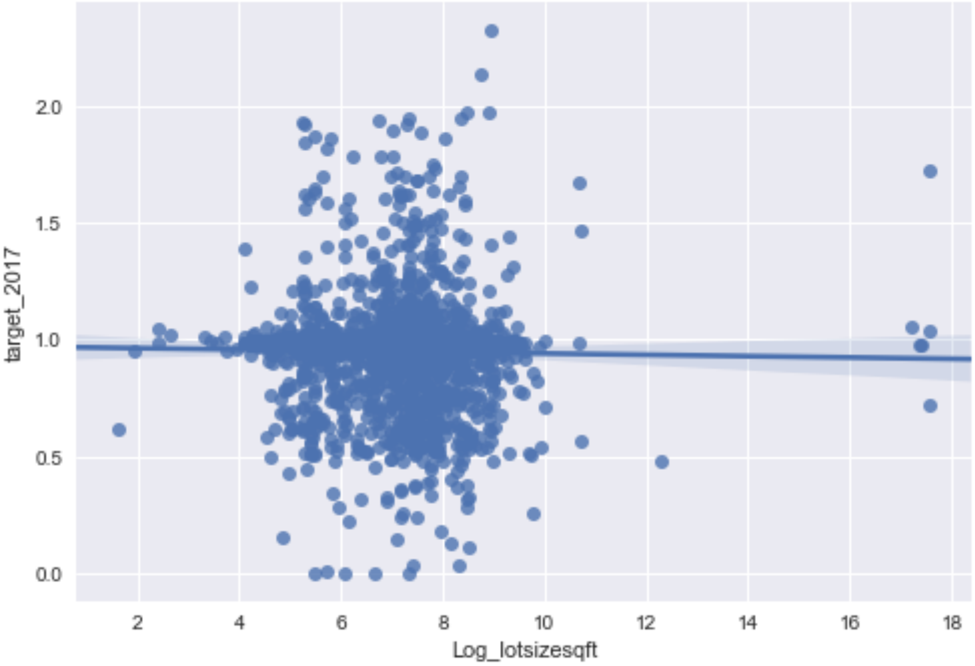
zhvi_fiveyear

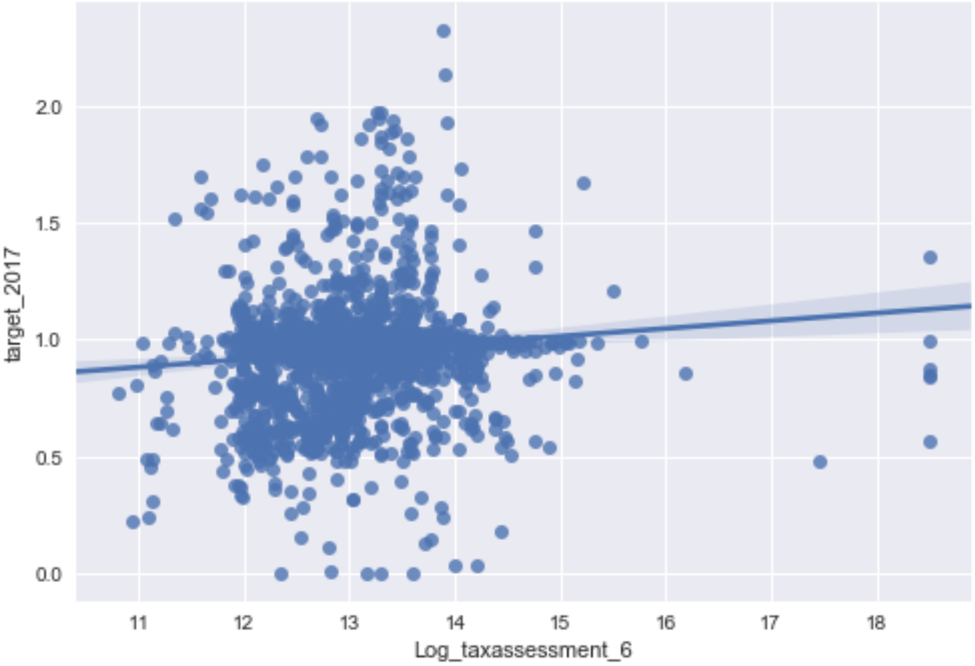


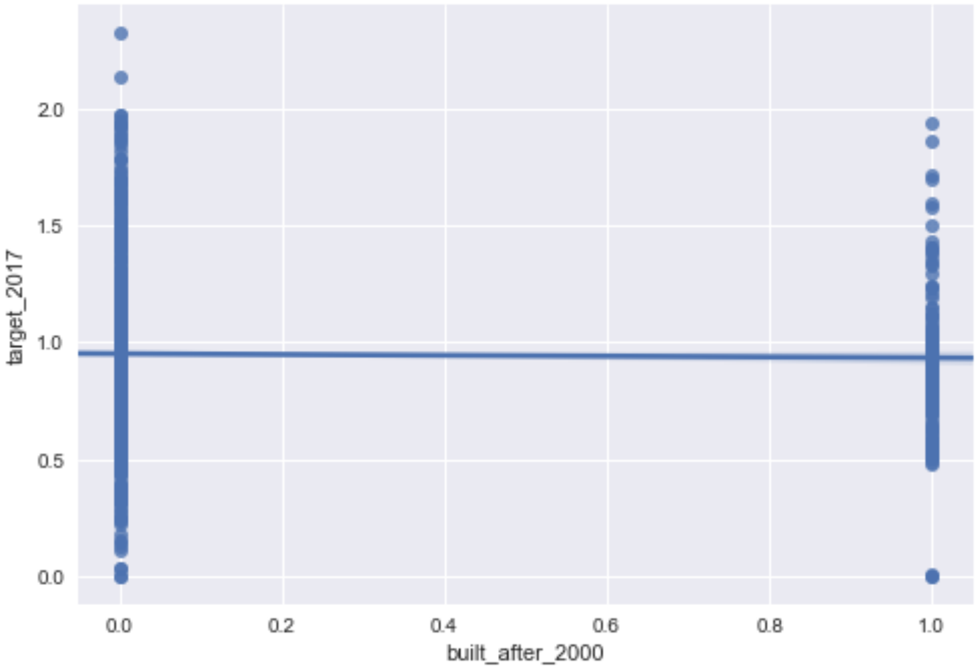


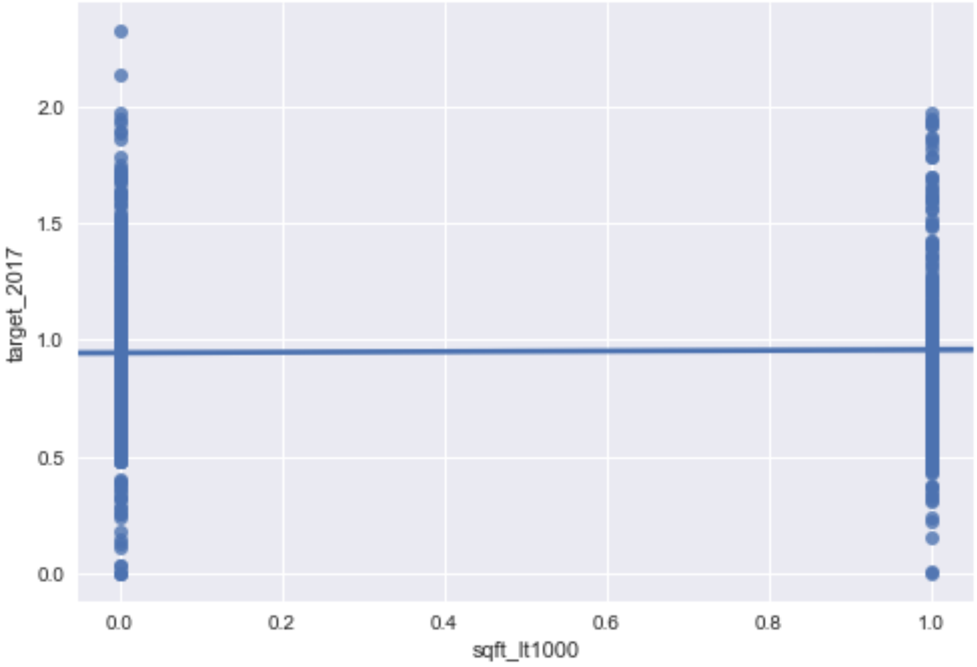


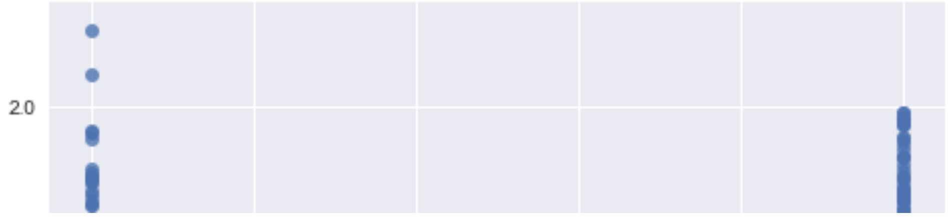

0.0
−0.025
−0.020
−0.015
−0.010
−0.005
0.000
zhvi_pctfallfrompeak

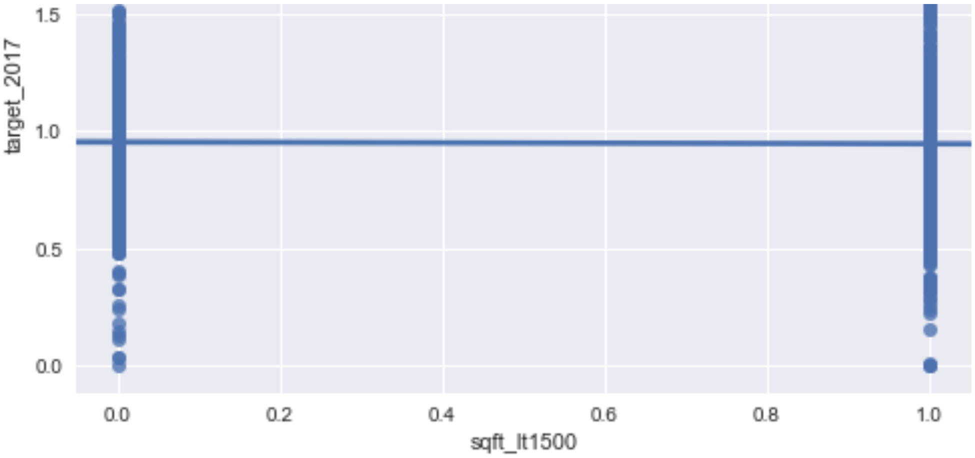

target_2017
2.0
1.5
1.0
0.5
0.0
2
4
6
8
10
12
14
16
18
Log_lotsizesqft

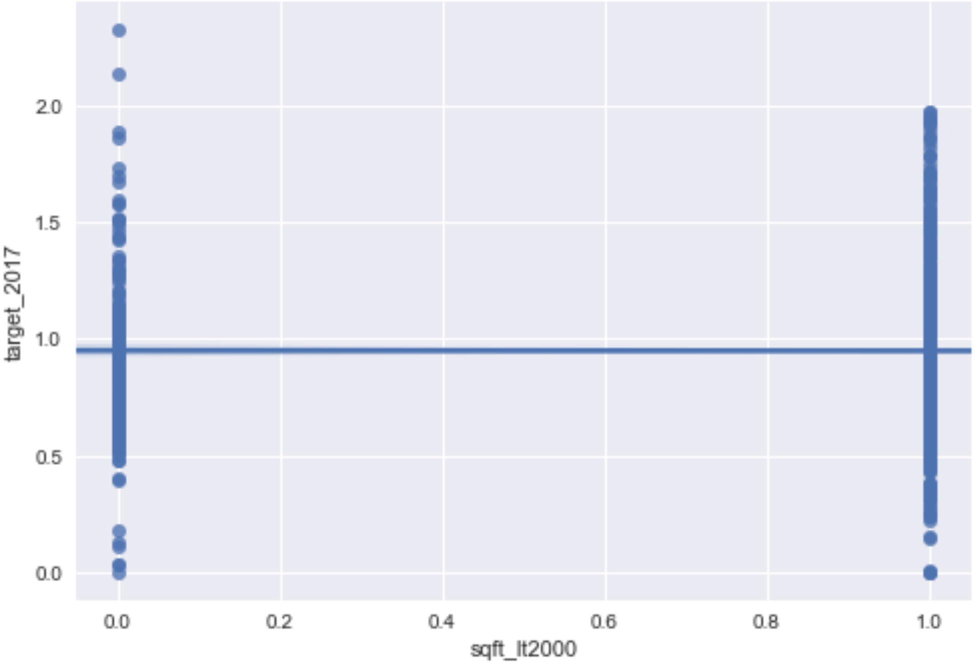

target_2017
2.0
1.5
1.0
0.5
0.0
11
12
13
14
15
16
17
18
Log_taxassessment_6

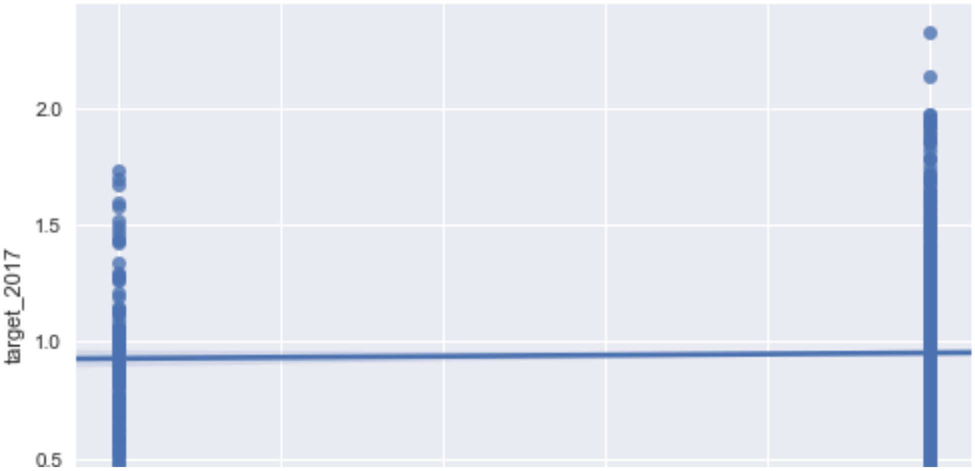

target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
built_after_2000

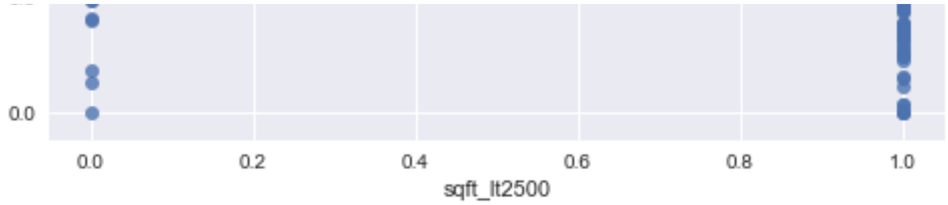

target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
sqft_lt1000

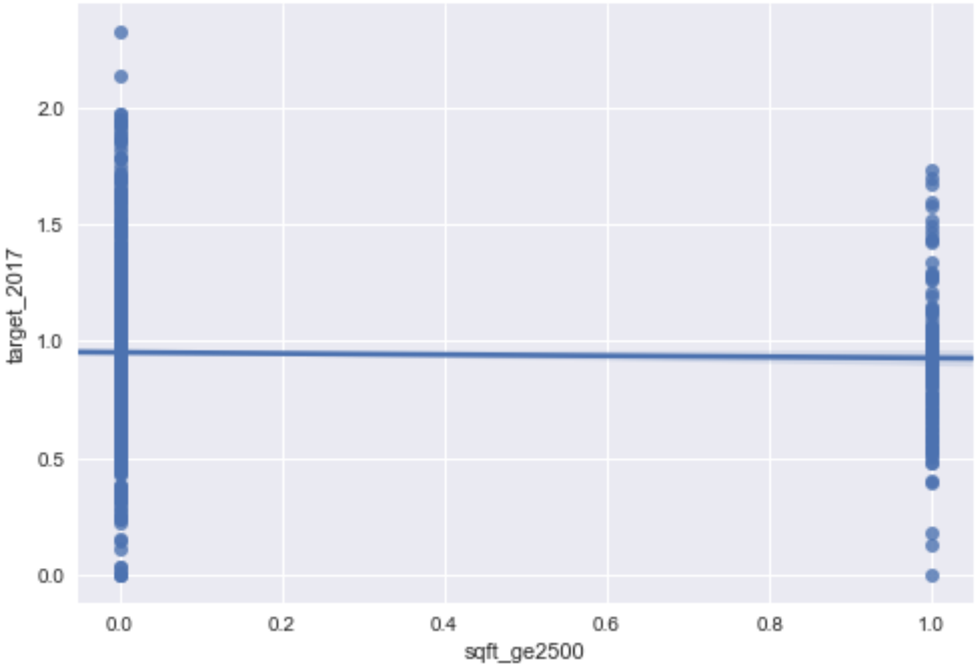

2.0

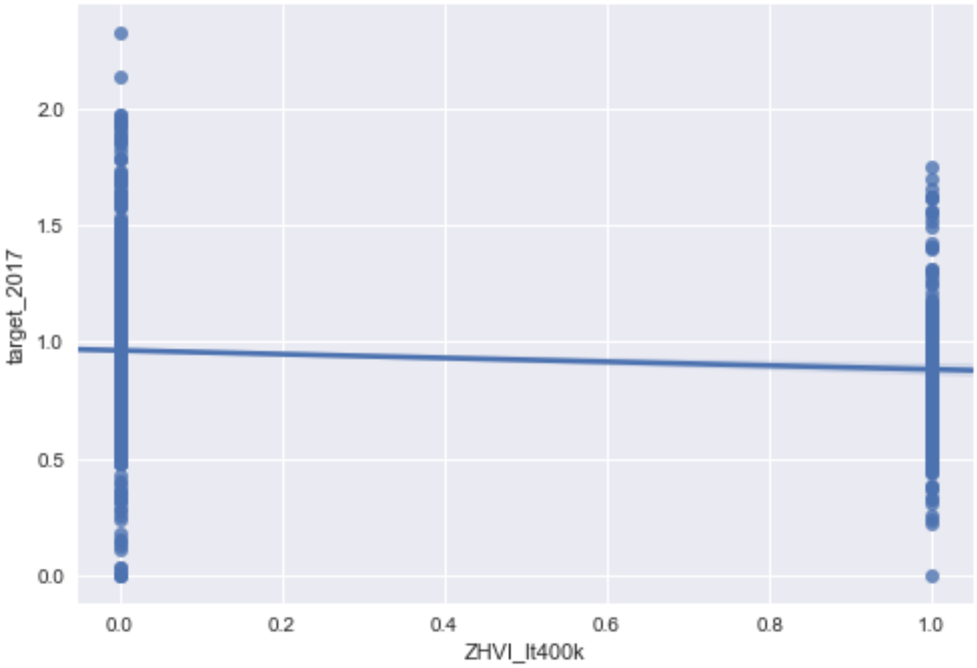

target_2017
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
sqft_lt1500

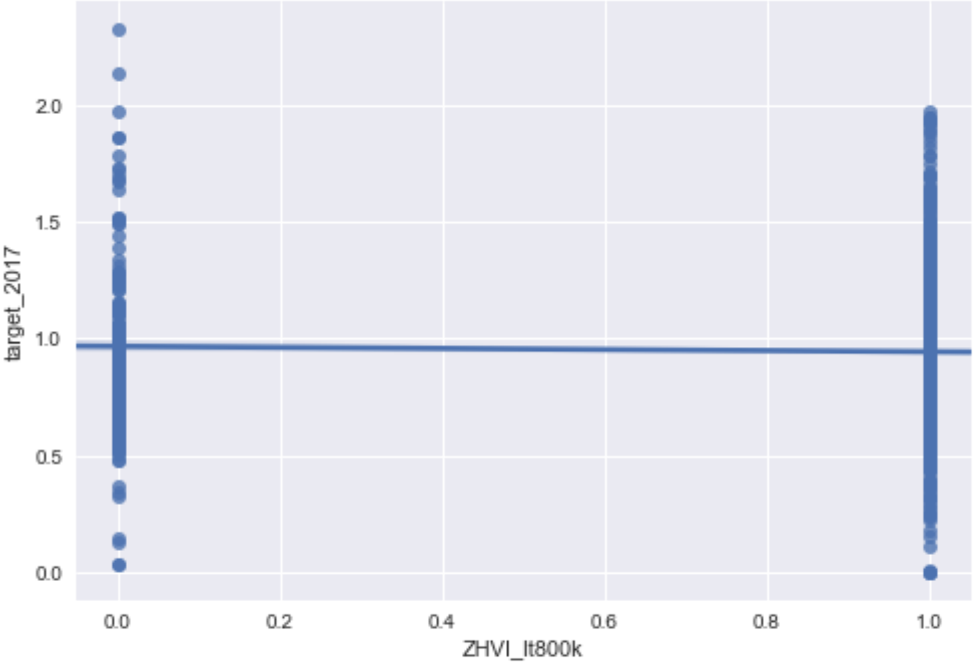

target_2017
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
sqft_lt2000

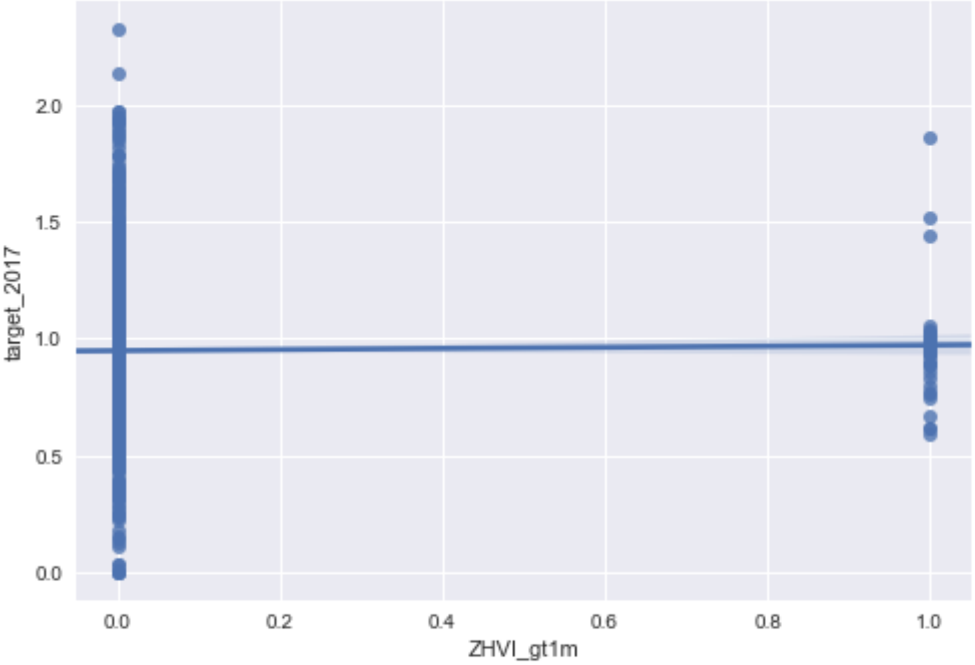

target_2017
2.0
1.5
1.0
0.5


0.0
0.0
0.2
0.4
0.6
0.8
1.0
sqft_lt2500


target_2017
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
sqft_ge2500


target_2017
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
ZHVI_lt400k


target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
ZHVI_lt800k


target_2017
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0
ZHVI_gt1m

```
In [16]: # Define feature and target lists and view correlation of features

         numerics = list(merged5.select_dtypes(include=[np.number]).columns.values)
         numerics.remove('zpid')

         numerics2 = merged5[numerics]

         f, ax = plt.subplots(figsize=(30, 30))
         corr = numerics2.corr()
         # sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverg
         #             square=True, ax=ax)
         # sns.plt.show()
         cmap = cmap=sns.diverging_palette(5, 250, as_cmap=True)

         def magnify():
             return [dict(selector="th",
                          props=[("font-size", "7pt")]),
                     dict(selector="td",
                          props=[('padding', "0em 0em")]),
                     dict(selector="th:hover",
                          props=[("font-size", "12pt")]),
                     dict(selector="tr:hover td:hover",
                          props=[('max-width', '200px'),
                                 ('font-size', '12pt')])
         ]

         corr.style.background_gradient(cmap, axis=1)\
             .set_properties(**{'max-width': '80px', 'font-size': '10pt'})\
             .set_caption("Hover to magify")\
             .set_precision(2)\
             .set_table_styles(magnify())
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/matplotlib/colo
rs.py:494: RuntimeWarning: invalid value encountered in less
  cbook._putmask(xa, xa < 0.0, -1)
```

Out[16]:

Hover to magify

	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft	for_sale	zipcode	ze
bedrooms	1	0.71	-0.18	-0.02	-0.0062	0.65	-0.053	-0.0074	0.4
bathrooms	0.71	1	-0.075	-0.032	0.017	0.7	-0.025	-0.03	0.5
yearbuilt	-0.18	-0.075	1	-0.016	0.035	-0.11	0.077	0.19	-0.
lotsizesqft	-0.02	-0.032	-0.016	1	-0.0031	-0.015	0.065	-0.0051	0.0
taxassessment_6	-0.0062	0.017	0.035	-0.0031	1	0.04	0.064	0.16	0.0
finishedsqft	0.65	0.7	-0.11	-0.015	0.04	1	0.023	-0.024	0.7
for_sale	-0.053	-0.025	0.077	0.065	0.064	0.023	1	0.19	0.0
zipcode	-0.0074	-0.03	0.19	-0.0051	0.16	-0.024	0.19	1	-0.
zestimate_1	0.46	0.59	-0.12	0.0021	0.092	0.77	0.045	-0.099	1

	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft	for_sale	zipcode	ze
zestimate_6	0.48	0.6	-0.14	0.0014	0.087	0.77	0.0031	-0.13	0.9
percentile_1	0.55	0.52	-0.2	0.0028	-0.051	0.42	-0.27	-0.041	0.2
percentile_6	0.56	0.51	-0.21	0.0016	-0.053	0.41	-0.26	-0.044	0.2
lastsoldprice_6	0.45	0.56	-0.14	0.0095	0.075	0.72	-0.0015	-0.14	0.9
bbl_pct_chg_1617	-0.06	-0.042	-0.098	-0.022	-0.0091	-0.02	0.027	-0.24	0.0
bbl_pct_chg_1217	-0.039	-0.0025	0.0076	0.0021	0.067	0.014	0.0016	0.18	0.0
barber_shop	-8.5e-18	1.1e-16	1.7e-15	1e-16	-6e-17	-4e-17	2e-16	nan	-3.
cigarette_retail	0.12	-0.037	0.13	0.016	-0.41	-0.053	0.038	0.44	-0.
delicatessen	0.15	0.086	0.22	-0.0013	-0.16	0.032	0.1	0.45	-0.
food_products	0.019	0.024	-0.19	-0.018	-0.069	-0.015	-0.088	-0.46	-0.
parking_facility_attendant	-0.077	-0.092	0.18	0.04	-0.13	-0.13	-0.14	-0.55	-0.
grocery_store	0.16	0.18	0.049	-0.021	-0.025	0.2	0.53	1	0.0
special_events	0.023	0.064	0.13	0.0034	0.11	0.16	-0.058	0.95	0.0
charitable_solicitation	-0.27	-0.21	0.087	-0.05	0.0062	-0.16	0.021	0.36	-0.
home_improvement_salesman	-0.23	-0.14	-0.095	0.0013	-0.017	-0.17	0.038	-0.4	0.0
secondhand_dealers_a	-1.1e-16	2.4e-16	-5.7e-16	-8.6e-17	-3.2e-17	9.6e-17	1.7e-16	nan	-1
motor_vehicle_salesman	0.1	-0.037	0.15	-0.043	-0.15	-0.13	0.07	0.55	-0.
beauty_shop	2.8e-17	0	-1.7e-15	-1.4e-16	0	0	nan	nan	1.8
parking_facility	-0.28	-0.28	0.44	-0.039	-0.24	-0.28	-0.15	-0.27	-0.
gen_contr_construction_mngr	0.13	0.13	-0.11	-0.035	0.061	0.17	0.12	0.32	0.
consumer_goods_auto_repair	-1.2e-16	-1.2e-16	2.6e-15	-1.3e-16	3.8e-17	1.5e-16	-3.8e-17	nan	1.2
apartment	-0.034	0.042	-0.13	-0.018	-0.04	0.059	0.022	-0.19	0.2
two_family_rental	0.22	0.17	0.043	0.062	-0.025	0.12	-0.073	-0.65	0.0
home_improvement_contractor	0.039	-0.089	0.0001	-0.032	-0.15	-0.051	0.041	0.29	-0.
general_business_licenses	-0.091	-0.068	-0.072	-0.0029	0.0059	-0.049	0.063	0.012	0.0
patent_medicine	-0.0031	0.037	0.098	0.03	0.034	0.051	0.35	0.39	0.0
restaurant	0.013	-0.043	-0.15	-0.033	-0.035	-0.11	0.0082	-0.25	-0.
one_family_rental	0.044	0.061	-0.084	-0.011	0.045	0.071	0.06	-0.085	0.0
sizerank	-0.0055	0.048	0.22	-0.0056	0.092	0.05	0.036	0.37	0.0
zhvi	0.089	0.19	-0.079	0.027	0.013	0.26	-0.11	-0.33	0.3
zhvi_mom	0.079	-0.013	0.043	-0.013	-0.041	-0.047	0.072	0.3	-0.

	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft	for_sale	zipcode	ze
zhvi_qoq	0.1	0.015	0.045	-0.018	-0.038	-0.018	0.053	0.29	-0.
zhvi_yoy	0.096	0.0015	0.13	-0.025	-0.028	-0.09	0.083	0.46	-0.
zhvi_fiveyear	0.18	0.039	-0.027	-0.02	-0.12	-0.088	0.065	0.068	-0.
zhvi_tenyear	0.031	0.069	-0.05	0.027	-0.066	-0.019	-0.022	-0.53	-0.
peakzhvi	0.088	0.19	-0.079	0.027	0.013	0.26	-0.11	-0.33	0.3
zhvi_pctfallfrompeak	0.11	0.037	0.028	-0.0078	-0.038	0.02	0.022	0.15	-0.
target_2017	-0.069	-0.047	-0.0056	0.032	-0.0097	-0.031	0.068	-0.078	-0.
Log_lotsizesqft	0.54	0.45	-0.2	0.43	-0.00063	0.47	0.038	0.17	0.2
Log_taxassessment_6	0.35	0.47	-0.1	0.0099	0.5	0.54	-0.026	-0.18	0.6
built_after_2000	-0.11	0.013	0.73	-0.023	0.0051	-0.032	0.066	-0.1	0.0
sqft_lt1000	-0.56	-0.55	0.14	-0.0057	-0.013	-0.52	0.023	0.0059	-0.
sqft_lt1500	-0.54	-0.6	0.15	0.019	-0.02	-0.67	0.018	0.034	-0.
sqft_lt2000	-0.51	-0.61	0.12	0.027	-0.023	-0.73	0.012	0.036	-0.
sqft_lt2500	-0.45	-0.52	0.089	0.019	-0.036	-0.75	-0.013	0.02	-0.
sqft_ge2500	0.45	0.52	-0.089	-0.019	0.036	0.75	0.013	-0.02	0.5
ZHVI_lt400k	0.019	-0.11	0.071	-0.026	-0.053	-0.12	0.092	0.55	-0.
ZHVI_lt800k	-0.12	-0.17	0.0054	-0.013	-0.0072	-0.28	0.079	-0.041	-0.
ZHVI_gt1m	0.11	0.1	0.011	-0.01	-0.0022	0.13	-0.041	0.083	0.0

In [17]:

```
# Remove the following features based on correlation and intuition
numerics = list(merged5.select_dtypes(include=[np.number]).columns.values)
numerics.remove('zpid')
numerics.remove('zestimate_1')
numerics.remove('zestimate_6')
numerics.remove('percentile_1')
numerics.remove('percentile_6')
numerics.remove('lastsoldprice_6')
numerics.remove('barber_shop')
numerics.remove('cigarette_retail')
numerics.remove('delicatessen')
numerics.remove('food_products')
numerics.remove('parking_facility_attendant')
numerics.remove('grocery_store')
numerics.remove('special_events')
numerics.remove('charitable_solicitation')
numerics.remove('home_improvement_salesman')
numerics.remove('secondhand_dealers_a')
numerics.remove('motor_vehicle_salesman')
numerics.remove('beauty_shop')
numerics.remove('parking_facility')
numerics.remove('gen_contr_construction_mngr')
numerics.remove('consumer_goods_auto_repair')
numerics.remove('apartment')
numerics.remove('two_family_rental')
numerics.remove('home_improvement_contractor')

numerics2 = merged5[numerics]

f, ax = plt.subplots(figsize=(30, 30))
corr = numerics2.corr()
# sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverg
#             square=True, ax=ax)
# sns.plt.show()
cmap = cmap=sns.diverging_palette(5, 250, as_cmap=True)

def magnify():
    return [dict(selector="th",
                 props=[("font-size", "7pt")]),
            dict(selector="td",
                 props=[('padding', "0em 0em")]),
            dict(selector="th:hover",
                 props=[("font-size", "12pt")]),
            dict(selector="tr:hover td:hover",
                 props=[('max-width', '200px'),
                        ('font-size', '12pt')])
]

corr.style.background_gradient(cmap, axis=1)\
    .set_properties(**{'max-width': '80px', 'font-size': '10pt'})\
    .set_caption("Hover to magify")\
    .set_precision(2)\
    .set_table_styles(magnify())
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/matplotlib/colo
rs.py:494: RuntimeWarning: invalid value encountered in less
```

```
    cbook._putmask(xa, xa < 0.0, -1)

Out[17]:
```

Hover to magify

	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft	for_sale	zipcode	bbl_pct_
bedrooms	1	0.71	-0.18	-0.02	-0.0062	0.65	-0.053	-0.0074	-0.06
bathrooms	0.71	1	-0.075	-0.032	0.017	0.7	-0.025	-0.03	-0.042
yearbuilt	-0.18	-0.075	1	-0.016	0.035	-0.11	0.077	0.19	-0.098
lotsizesqft	-0.02	-0.032	-0.016	1	-0.0031	-0.015	0.065	-0.0051	-0.022
taxassessment_6	-0.0062	0.017	0.035	-0.0031	1	0.04	0.064	0.16	-0.009
finishedsqft	0.65	0.7	-0.11	-0.015	0.04	1	0.023	-0.024	-0.02
for_sale	-0.053	-0.025	0.077	0.065	0.064	0.023	1	0.19	0.027
zipcode	-0.0074	-0.03	0.19	-0.0051	0.16	-0.024	0.19	1	-0.24
bbl_pct_chg_1617	-0.06	-0.042	-0.098	-0.022	-0.0091	-0.02	0.027	-0.24	1
bbl_pct_chg_1217	-0.039	-0.0025	0.0076	0.0021	0.067	0.014	0.0016	0.18	0.38
general_business_licenses	-0.091	-0.068	-0.072	-0.0029	0.0059	-0.049	0.063	0.012	0.53
patent_medicine	-0.0031	0.037	0.098	0.03	0.034	0.051	0.35	0.39	0.25
restaurant	0.013	-0.043	-0.15	-0.033	-0.035	-0.11	0.0082	-0.25	0.46
one_family_rental	0.044	0.061	-0.084	-0.011	0.045	0.071	0.06	-0.085	0.71
sizerank	-0.0055	0.048	0.22	-0.0056	0.092	0.05	0.036	0.37	0.066
zhvi	0.089	0.19	-0.079	0.027	0.013	0.26	-0.11	-0.33	0.042
zhvi_mom	0.079	-0.013	0.043	-0.013	-0.041	-0.047	0.072	0.3	-0.4
zhvi_qoq	0.1	0.015	0.045	-0.018	-0.038	-0.018	0.053	0.29	-0.47
zhvi_yoy	0.096	0.0015	0.13	-0.025	-0.028	-0.09	0.083	0.46	-0.31
zhvi_fiveyear	0.18	0.039	-0.027	-0.02	-0.12	-0.088	0.065	0.068	-0.27
zhvi_tenyear	0.031	0.069	-0.05	0.027	-0.066	-0.019	-0.022	-0.53	0.083
peakzhvi	0.088	0.19	-0.079	0.027	0.013	0.26	-0.11	-0.33	0.045
zhvi_pctfallfrompeak	0.11	0.037	0.028	-0.0078	-0.038	0.02	0.022	0.15	-0.4
target_2017	-0.069	-0.047	-0.0056	0.032	-0.0097	-0.031	0.068	-0.078	0.04
Log_lotsizesqft	0.54	0.45	-0.2	0.43	-0.00063	0.47	0.038	0.17	-0.17
Log_taxassessment_6	0.35	0.47	-0.1	0.0099	0.5	0.54	-0.026	-0.18	0.16
built_after_2000	-0.11	0.013	0.73	-0.023	0.0051	-0.032	0.066	-0.1	0.037
sqft_lt1000	-0.56	-0.55	0.14	-0.0057	-0.013	-0.52	0.023	0.0059	0.028
sqft_lt1500	-0.54	-0.6	0.15	0.019	-0.02	-0.67	0.018	0.034	0.021
sqft_lt2000	-0.51	-0.61	0.12	0.027	-0.023	-0.73	0.012	0.036	-0.021

	bedrooms	bathrooms	yearbuilt	lotsizesqft	taxassessment_6	finishedsqft	for_sale	zipcode	bbl_pct
sqft_lt2500	-0.45	-0.52	0.089	0.019	-0.036	-0.75	-0.013	0.02	-0.018
sqft_ge2500	0.45	0.52	-0.089	-0.019	0.036	0.75	0.013	-0.02	0.018
ZHVI_lt400k	0.019	-0.11	0.071	-0.026	-0.053	-0.12	0.092	0.55	-0.28
ZHVI_lt800k	-0.12	-0.17	0.0054	-0.013	-0.0072	-0.28	0.079	-0.041	0.24
ZHVI_gt1m	0.11	0.1	0.011	-0.01	-0.0022	0.13	-0.041	0.083	0.1

In [18]:

```
# Remove the following features based on correlation and intuition and to cr
numerics = list(merged5.select_dtypes(include=[np.number]).columns.values)
numerics.remove('zpid')
numerics.remove('zestimate_1')
numerics.remove('percentile_1')
numerics.remove('percentile_6')
numerics.remove('lastsoldprice_6')
numerics.remove('barber_shop')
numerics.remove('cigarette_retail')
numerics.remove('delicatessen')
numerics.remove('food_products')
numerics.remove('parking_facility_attendant')
numerics.remove('grocery_store')
numerics.remove('special_events')
numerics.remove('charitable_solicitation')
numerics.remove('home_improvement_salesman')
numerics.remove('secondhand_dealers_a')
numerics.remove('motor_vehicle_salesman')
numerics.remove('beauty_shop')
numerics.remove('parking_facility')
numerics.remove('gen_contr_construction_mngr')
numerics.remove('consumer_goods_auto_repair')
numerics.remove('apartment')
numerics.remove('two_family_rental')
numerics.remove('home_improvement_contractor')
numerics.remove('zhvi')
numerics.remove('zhvi_mom')
numerics.remove('zhvi_pctfallfrompeak')
numerics.remove('sqft_lt2500')
numerics.remove('ZHVI_lt800k')

# Create features and target data frames
features = merged5[numerics]
target = merged5[['target_2017']]
target
```

Out[18]:

	target_2017
lastsolddate_6	
2017-01-18	0.858865
2017-06-30	0.545293
2017-05-12	1.003960
2017-07-10	0.820322
2017-04-05	0.483660
2017-05-01	0.983266
2017-01-10	1.273225
2017-04-12	0.558435
2017-06-05	1.313315

In [19]:

```
# Create derivations and interactions

features.loc[:,'recent_for_sale'] = (features['for_sale'] * features['built_
features.loc[:,'zest_to_peakzhvi'] = (features['zestimate_6'] / features['ze

numerics = list(features.select_dtypes(include=[np.number]).columns.values)
numerics.remove('zestimate_6')
numerics.remove('target_2017')
numerics.remove('taxassessment_6')
numerics.remove('lotsizesqft')
numerics.remove('finishedsqft')
features2 = features[numerics]
list(features2)
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:297: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[key] = _infer_fill_value(value)
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/ind
exing.py:477: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self.obj[item] = s
```

Out[19]:

```
['bedrooms',
 'bathrooms',
 'yearbuilt',
 'for_sale',
 'zipcode',
 'bbl_pct_chg_1617',
 'bbl_pct_chg_1217',
 'general_business_licenses',
 'patent_medicine',
 'restaurant',
 'one_family_rental',
 'sizerank',
 'zhvi_qoq',
 'zhvi_yoy',
 'zhvi_fiveyear',
 'zhvi_tenyear',
 'peakzhvi',
 'Log_lotsizesqft',
 'Log_taxassessment_6',
 'built_after_2000',
 'sqft_lt1000',
 'sqft_lt1500',
 'sqft_lt2000',
 'sqft_ge2500',
 'ZHVI_lt400k',
 'ZHVI_gt1m',
 'recent_for_sale',
 'zest_to_peakzhvi']
```

In [20]:

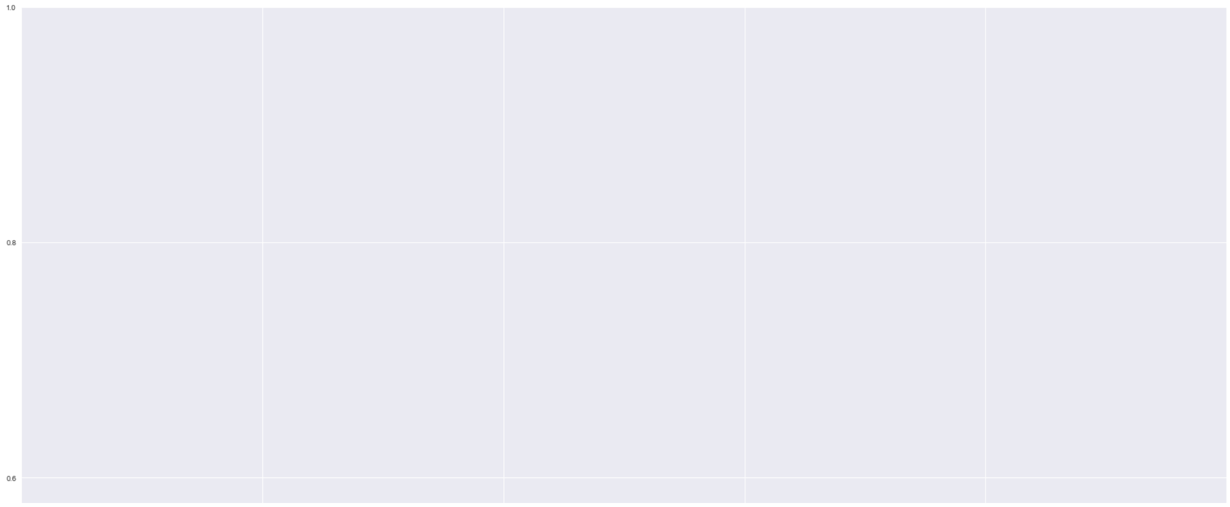
```
# Impute nulls to mean

med = np.mean(features2.bbl_pct_chg_1217)
features2.bbl_pct_chg_1217 = features2.bbl_pct_chg_1217.fillna(med)

med = np.mean(features2.bbl_pct_chg_1617)
features2.bbl_pct_chg_1617 = features2.bbl_pct_chg_1617.fillna(med)

med = np.mean(features2.general_business_licenses)
features2.general_business_licenses = features2.general_business_licenses.fi

med = np.mean(features2.patent_medicine)
features2.patent_medicine = features2.patent_medicine.fillna(med)

med = np.mean(features2.restaurant)
features2.restaurant = features2.restaurant.fillna(med)

med = np.mean(features2.one_family_rental)
features2.one_family_rental = features2.one_family_rental.fillna(med)

med = np.mean(features2.sizerank)
features2.sizerank = features2.sizerank.fillna(med)

med = np.mean(features2.zhvi_qoq)
features2.zhvi_qoq = features2.zhvi_qoq.fillna(med)

med = np.mean(features2.zhvi_fiveyear)
features2.zhvi_fiveyear = features2.zhvi_fiveyear.fillna(med)

med = np.mean(features2.zhvi_tenyear)
features2.zhvi_tenyear = features2.zhvi_tenyear.fillna(med)

med = np.mean(features2.zhvi_yoy)
features2.zhvi_yoy = features2.zhvi_yoy.fillna(med)

med = np.mean(features2.peakzhvi)
features2.peakzhvi = features2.peakzhvi.fillna(med)

med = np.mean(features2.zest_to_peakzhvi)
features2.zest_to_peakzhvi = features2.zest_to_peakzhvi.fillna(med)
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/pandas/core/gen
eric.py:2773: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.or
g/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  self[name] = value
```

In [21]:

```
# Plot target variable

g = sns.distplot(target.target_2017, rug=True, kde=True)
t = g.set_title("Distribution of Sale / Zestimate for 2017 Sales")
sns.plt.show()
```

```
/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/statsmodels/non
parametric/kdetools.py:20: VisibleDeprecationWarning: using a non-integer
number instead of an integer will result in an error in the future
  y = X[:m/2+1] + np.r_[0,X[m/2+1:],0]*1j
```



In [24]:

```
features2.describe()
```

Out[24]:

	bedrooms	bathrooms	yearbuilt	for_sale	zipcode	bbl_pct_chg_16
count	1899.000000	1899.000000	1899.000000	1899.000000	1899.000000	1899.000000
mean	2.531859	2.135887	1943.530806	0.286467	20011.234860	0.093709
std	1.489129	1.180729	37.126453	0.452229	8.471462	0.447923
min	0.000000	1.000000	1794.000000	0.000000	20001.000000	-0.604412
25%	2.000000	1.000000	1916.000000	0.000000	20003.000000	-0.092703
50%	2.000000	2.000000	1939.000000	0.000000	20009.000000	0.052114
75%	3.000000	3.000000	1964.000000	1.000000	20017.000000	0.419821
max	27.000000	10.000000	2017.000000	1.000000	20037.000000	1.523560

8 rows × 28 columns

In [25]:

```
np.any(np.isnan(features2))
```

Out[25]:

```
False
```

```
In [26]: model = RandomizedLasso(alpha=0.1)
         model.fit(features2, target["target_2017"])
         names = list(features2)

         print("Features sorted by their score:")
         print(sorted(zip(map(lambda x: round(x, 4), model.scores_),
                          names), reverse=True))
```

```
Features sorted by their score:
[(0.0, 'zipcode'), (0.0, 'zhvi_yoy'), (0.0, 'zhvi_tenyear'), (0.0, 'zhvi_
qoq'), (0.0, 'zhvi_fiveyear'), (0.0, 'zest_to_peakzhvi'), (0.0, 'yearbuil
t'), (0.0, 'sqft_lt2000'), (0.0, 'sqft_lt1500'), (0.0, 'sqft_lt1000'),
 (0.0, 'sqft_ge2500'), (0.0, 'sizerank'), (0.0, 'restaurant'), (0.0, 'rec
ent_for_sale'), (0.0, 'peakzhvi'), (0.0, 'patent_medicine'), (0.0, 'one_f
amily_rental'), (0.0, 'general_business_licenses'), (0.0, 'for_sale'),
 (0.0, 'built_after_2000'), (0.0, 'bedrooms'), (0.0, 'bbl_pct_chg_1617'),
(0.0, 'bbl_pct_chg_1217'), (0.0, 'bathrooms'), (0.0, 'ZHVI_lt400k'), (0.
0, 'ZHVI_gt1m'), (0.0, 'Log_taxassessment_6'), (0.0, 'Log_lotsizesqft')]
```

```
In [27]: splits = cv.train_test_split(features2, target, test_size=0.2)
         X_train, X_test, y_train, y_test = splits
         X_train.shape, y_train.shape

Out[27]: ((1519, 28), (1519, 1))
```

```
In [28]: X_test.shape, y_test.shape

Out[28]: ((380, 28), (380, 1))
```

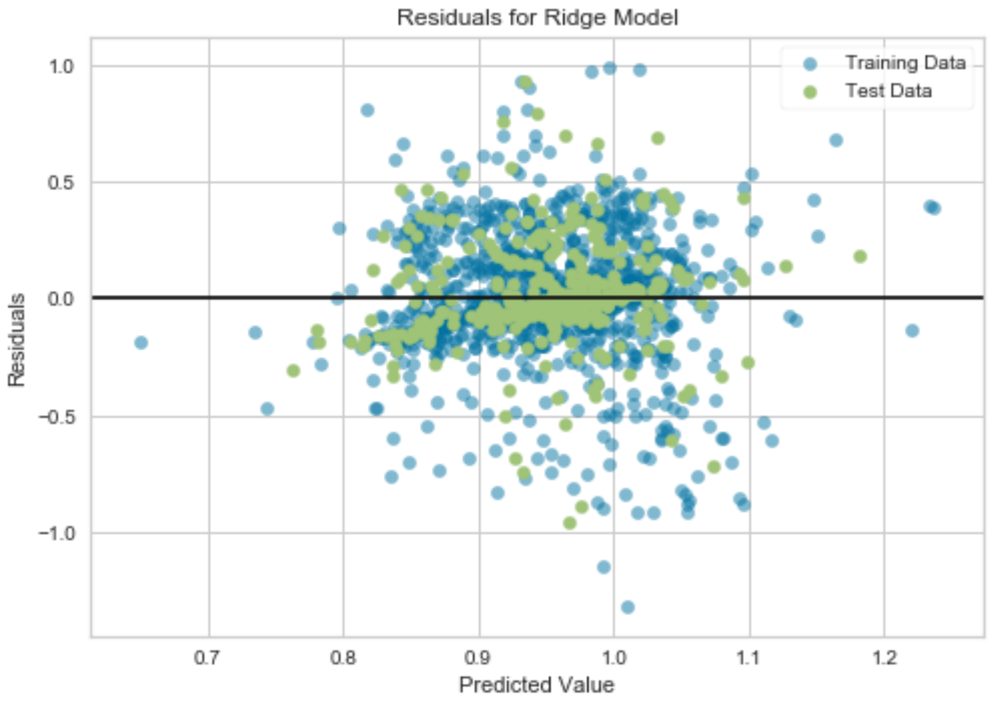
```
In [29]: ridge = Ridge().fit(X_train, y_train)

         # Predict on the test data: y_pred
         y_pred = ridge.predict(X_test)

         print('Ridge Model')
         print('Root Mean Squared Error: {:.3f}'.format(np.sqrt(mse(y_test, y_pred))
         print('Mean Squared Error: {:.3f}'.format(mse(y_test, y_pred)))
         print('Coefficient of Determination: {:.3f}'.format(r2_score(y_test, y_pred)
```

```
Ridge Model
Root Mean Squared Error: 0.223
Mean Squared Error: 0.050
Coefficient of Determination: 0.031
```

```
In [30]: print('ridge.coef_: {}'.format(ridge.coef_))
         print('ridge.intercept_: {}'.format(ridge.intercept_))
```

```
ridge.coef_: [[ -1.47611182e-02  -1.33512393e-02   6.62180936e-04   6.302
23673e-02
   -3.15699748e-03  -3.09326058e-02   2.24191101e-03  -2.80793758e-02
    8.84799274e-02   5.74814708e-03   2.91810187e-02  -2.38471850e-06
   -8.22145695e-02  -1.11627586e-01  -3.16529645e-02   1.49595918e-02
   -5.50135019e-09   5.61264293e-03   4.18550527e-02  -4.68716118e-02
    9.92562110e-03  -1.69357369e-02  -2.82697710e-02  -5.74225490e-02
   -2.17486879e-02   4.40706262e-02  -9.37333170e-02   0.00000000e+00]]
ridge.intercept_: [ 62.40538306]
```

In [31]:
```
from yellowbrick.regressor import ResidualsPlot

visualizer = ResidualsPlot(ridge)

fig = plt.figure()
visualizer.fit(X_train, y_train)
visualizer.score(X_test, y_test)
g = visualizer.poof()
```



In [32]:
```
model = LinearRegression()
model.fit(X_train, y_train)

expected = y_test
predicted = model.predict(X_test)

print("Linear Regression model")
print('Root Mean Squared Error: {:.3f}'.format(np.sqrt(mse(expected, predict
print("Coefficient of Determination: %0.3f" % r2_score(expected, predicted)
```

```
Linear Regression model
Root Mean Squared Error: 0.223
Coefficient of Determination: 0.030
```

```
In [33]: model = RandomForestRegressor()
         model.fit(X_train, y_train)

         expected = y_test
         predicted = model.predict(X_test)

         print("Random Forest model")
         print('Root Mean Squared Error: {:.3f}'.format(np.sqrt(mse(expected, predict
         print("R2 score = %0.3f" % r2_score(expected, predicted))
```

```
Random Forest model
Root Mean Squared Error: 0.236
R2 score = -0.085

/Users/uzairsiddiqui/anaconda/lib/python3.6/site-packages/ipykernel/__mai
n__.py:2: DataConversionWarning: A column-vector y was passed when a 1d a
rray was expected. Please change the shape of y to (n_samples,), for exam
ple using ravel().
  from ipykernel import kernelapp as app
```

Next Steps:

Further test hypothesis using additional:

- Data Sources:
  - Add Census data
  - Zillow Neighborhood Data
  - Other Geographies
- More granular analysis based on lat/long
- Refined time window
- Use Zillow's Kaggle Data Set for two counties in California
- Potential Real Estate Investor application

GitHub Code Repository:

https://github.com/IoTDevPro/RealEstateMoguls (https://github.com/IoTDevPro/RealEstateMoguls)

---- END ----