Analysis Evicitions

Renate van Kempen, April 2019

Julie Holzhauer stands among her family's possessions after being evicted from her home in Centennial, Colo., in 2011.

John Moore/Getty Images

Executive summary

Evictions in the USA, what is that about? According to the principle investigator at the Eviction Lab (Matthew Desmond), there are about 900,000 evictions each year, which equates to about an estimated 2.3 million people evicted, many of them children. That's about 6,300 people a day that are evicted, twice the number of people who die in car accidents every day in America. It's therefore a problem of enormous consequence in the country.

The goal of this analysis is to predict the number of evictions at county level in the USA, using different socioeconomic and demographic indicators. The data for this analysis was provided via the United States Department of Agriculture Economic Research Service and the Eviction Lab.

The socioeconomic indicators used in this analysis were:

- Education level
- Income
- Financial burden (in this case rent)
- Poverty rate
- Unemployment rate

The demographic indicators used were:

- Crude death rate - Annual number of deaths per 1,000 population.
- Crude birth rate - Annual number of births per 1,000 population.
- Population
- Information about the neighborhood

The steps taken in the process were:

1. Reading about the problem
2. Initial data exploration
3. Looking at correlations in the data
4. Cleaning the data
5. Modeling – trying different models to get the best

After trying different models, I've found the Random Forest model to work best as a predictor for evictions per county.

The key insights gained by performing this Capstone project about predicting evictions were:

- Before cleaning any features, try to use all features as is in the model to check what the results are. If needed, you can tweak, clean, drop, rename, recategorize or reshape all features if necessary.
- For so many features, a simple linear regression model might not be sufficient, therefore I will start with a Random Forrest for a future project with so many not directly correlated features.
- If you need a positive value as an outcome, you can not just use the linear regression model, as this gives you also negative results. Better to use either a Lasso Regression or a Random Forest model to get only positive results.
- Running a Random Forrest can give you different results after each time you run this. Therefore I've chosen to submit the one with the best R^2 result.
- All features have a (slight) influence on the model

***Personal note**: I'll try not to forget that an eviction is not a only number. It involves people and it is probably the worst thing that can happen to anyone, as one's home is taken away. It is more than just a place to sleep; it is a basic need, a place to feel save and welcome.*

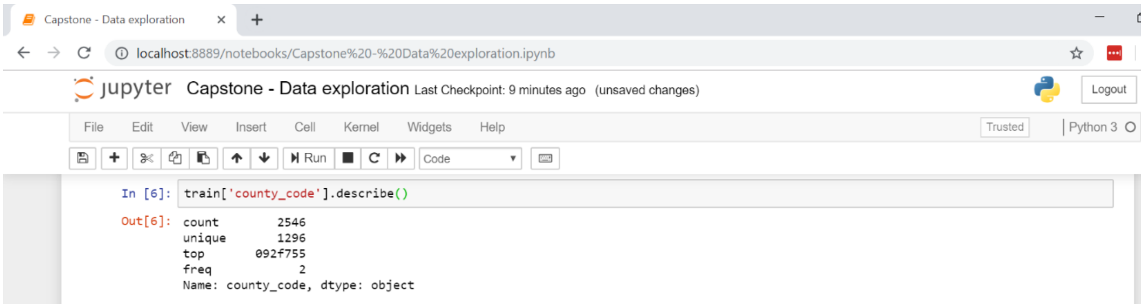
Description of the data

The data for this analysis consists of publicly available data and was provided via the United States Department of Agriculture Economic Research Service and the Eviction Lab.

The training data that was provided consisted of both values and labels, in order to be able to make a supervised machine learning model to predict the number of evictions at county level.

The original training values consisted of 48 columns with both socioeconomic and demographic indicators and 2.546 rows.

As we need to predict at county level, I've first explored how many different counties there were in this dataset using python. This resulted in 1.296 unique counties. The codes were encrypted and therefore not usable to be grouped. Therefore, I've not added this feature to the model for training.



For all 48 values, I've tried to get a first idea of the most important features to predict whether or not an eviction can be expected. The key features I've used to start with were:

Geographic indicators:

- county (in codes)

Socioeconomic indicators:

- median_household_income
- rent_burden
- poverty_rate
- pct_unemployment = percentage unemployment

Demographic indicators:

- population
- information about the different neighborhoods:
  - rucc = Rural-Urban Continuum Codes
  - *urban_influence → left out, as it seemed similar to rucc*
  - economic_typology = six mutually exclusive categories of economic dependence
- pct_female = percentage female
- pct_below_18_years_of_age = percentage children
- birth_rate_per_1k
- death_rate_per_1k

Explanation of the process

Allow me to explain the process that I've followed to get the best possible result.

1. Reading about the problem
2. Initial data exploration
3. Looking at correlations in the data
4. Cleaning the data
5. Modeling – trying different models to get the best

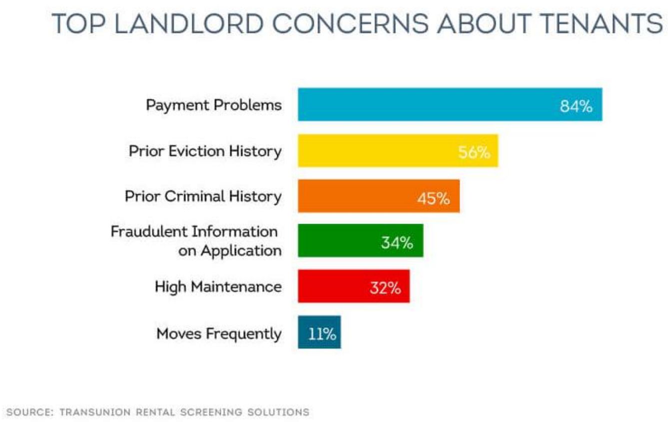
Reading about the problem

Before starting to work on the data, I've read all kinds of articles on the eviction-problem in the USA.

According to this article with the principle investigator at the Eviction Lab (Matthew Desmond), there are about 900,000 evictions each year, which equates to about an estimated 2.3 million people evicted, many of them children. That's about 6,300 people a day that are evicted, twice the number of people who die in car accidents every day in America. It's therefore a problem of enormous consequence in the country.

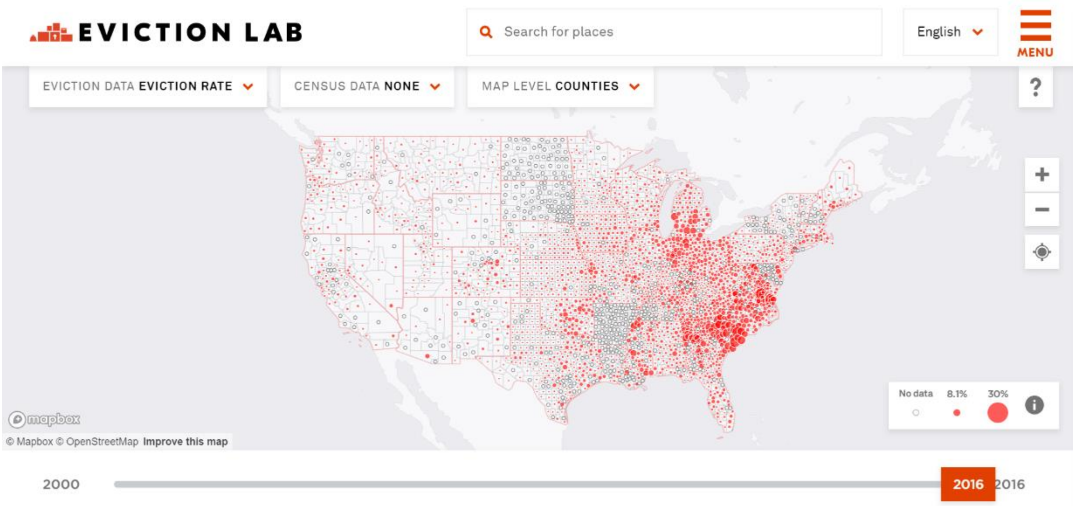
This article tries to explain why evictions are so high:

'*Incomes have remained flat for many Americans over the last two decades, but housing costs have soared. So between 1995 and today, median asking rents have increased by 70 percent, adjusting for inflation. So there's a shrinking gap between what families are bringing [in] and what they have to pay for basic shelter.*'



According to this information, it seems that the payment problems or financial burden (in this case housing costs / rent) is one of the main factors in predicting whether eviction can happen. Therefore, I've used this as one of the key features.

Finally, I've looked at the websites of the United States Department of Agriculture Economic Research Service and the Eviction Lab. In the latter, I've found this information about the evictions per county in 2016.

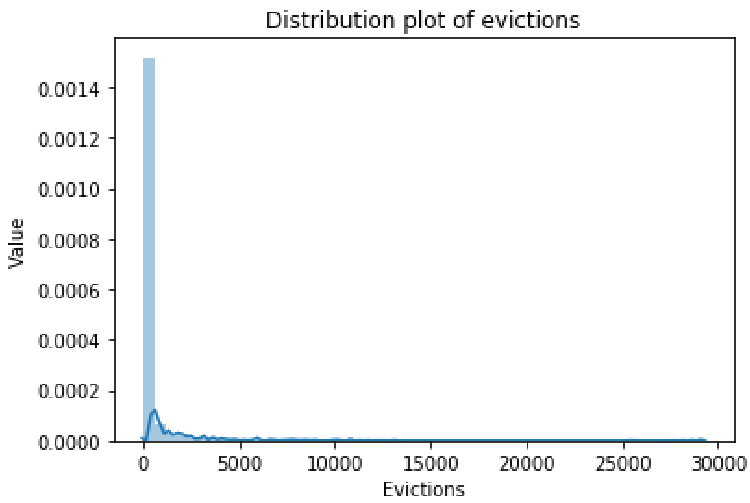


Initial Data Exploration

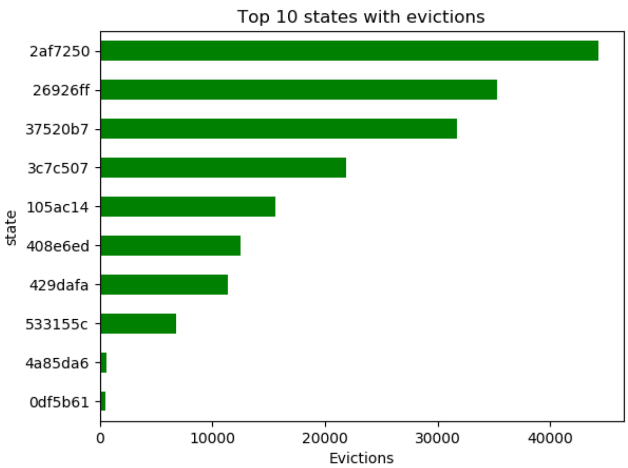
The initial exploration of the data began with some summary and descriptive statistics for the key numeric features.

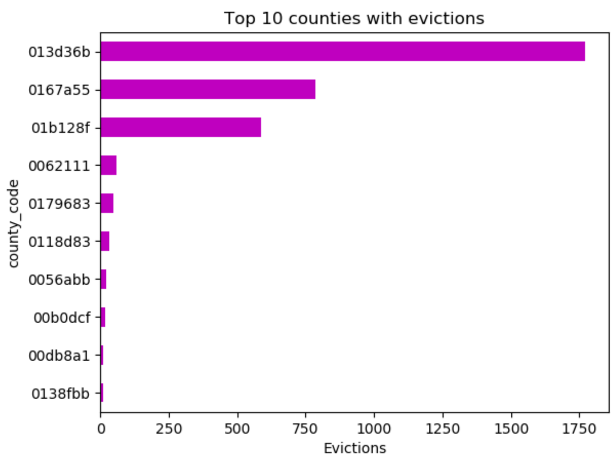
Feature	Mean	Std	Min	25%	Median	75%	Max
renter_occupied_households	15,008	53,334	14	1,052	2,581	8,099	882,101
population	106,246	332,852	116	10,294	23,863	67,969	5,279,852
median_household_income	46,051	11,586	19,328	38,496	44,480	51,526	123,452
rent_burden	28.5	4.5	10.0	26.0	28.8	31.2	49.5
poverty_rate	12.4	5.7	-	8.4	11.5	15.3	44.7
pct_unemployment	0.06	0.02	0.02	0.04	0.06	0.07	0.18
birth_rate_per_1k	11.5	2.6	3.6	9.9	11.3	12.8	28.9
death_rate_per_1k	10.4	2.7	-	8.6	10.4	12.1	27.4
evictions	378	1,405	-	4	29	161	29,251

Since evictions is the label, we try to predict here, I've also made a distribution plot for the values provided. You can clearly see the distribution being right-skewed here, meaning that most of the evictions are lower than 3.000.



As there were so many different counties, I've tried to look at both county and state level what the top 10 counties or states with total evictions were. As the data is encrypted, we don't know to which state or county the codes refer. Therefore, this information is not very useful, but in real life this could be very informative.



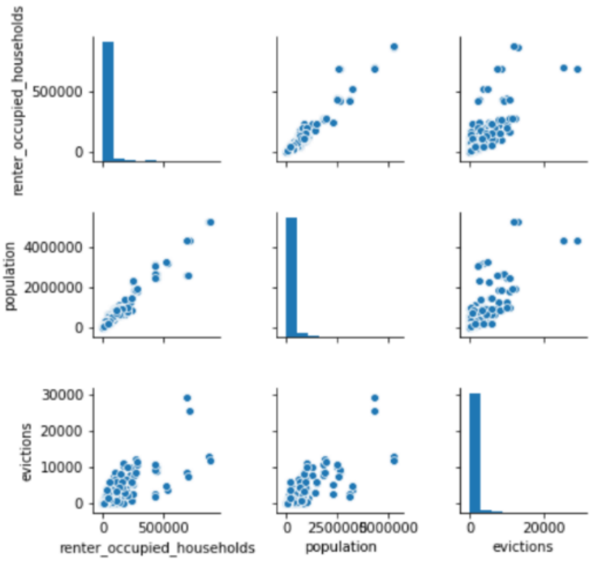


Correlation and Apparent Relationships

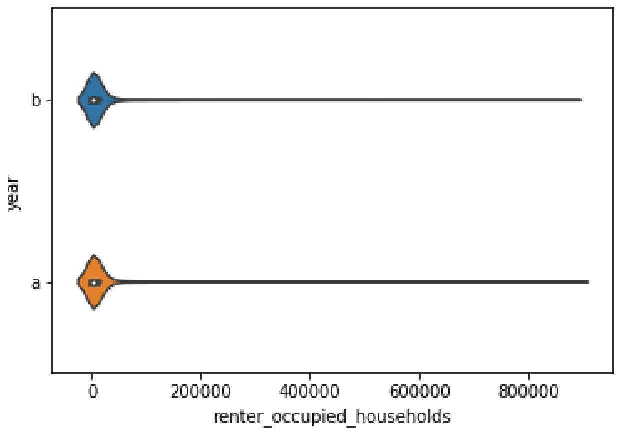
After the first data exploration, I've started to find correlations between the different numeric values and the label (evictions) via a correlation matrix in excel.

In both the matrix column for evictions (sorted) as wel as in the correlation matrix via Seaborn you can clearly see the high correlation between the label and population as well as renter occupied households. Therefore, I've decided to add renter_occupied_households to my key features in the model.

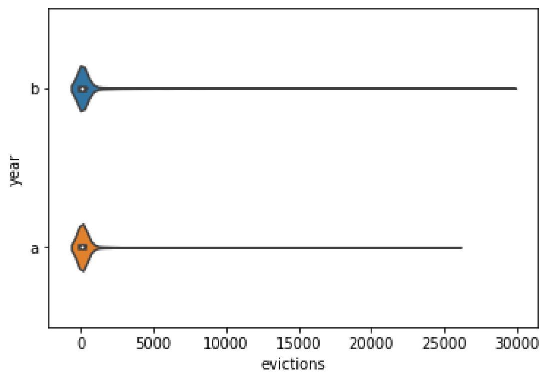
Correlation matrix via Seaborn



	evictions
evictions	1
population	0.808023612
renter_occupied_households	0.806801566
pct_renter_occupied	0.368407698
pct_asian	0.321427694
median_gross_rent	0.301743265
pct_adults_bachelors_or_higher	0.293026847
pct_af_am	0.195534715
pct_other	0.175722219
birth_rate_per_1k	0.174273451
median_property_value	0.174030866
homicides_per_100k	0.172053456
rent_burden	0.155837752
pct_hispanic	0.142144976
median_household_income	0.131494235
pct_female	0.131131574
pct_below_18_years_of_age	0.091834258
pct_civilian_labor	0.083338092
pct_multiple	0.068432416
pct_low_birthweight	0.060569069
pct_uninsured_adults	0.057476222
poverty_rate	0.025937151
pct_nh_pi	0.024022024
pct_excessive_drinking	0.000692803
pct_unemployment	-0.020083201
pct_adults_with_some_college	-0.029822684
air_pollution_particulate_matter_value	-0.034416082
pct_uninsured_children	-0.034523002
pct_am_ind	-0.038094141
pct_adults_less_than_a_high_school_diploma	-0.040633721
heart_disease_mortality_per_100k	-0.077512679
pct_diabetes	-0.114469184
pct_adult_obesity	-0.116674133
pop_per_primary_care_physician	-0.132156332
pct_adult_smoking	-0.145986274
pct_physical_inactivity	-0.17598342
pop_per_dentist	-0.176353624
death_rate_per_1k	-0.209736679
motor_vehicle_crash_deaths_per_100k	-0.256892928
pct_aged_65_years_and_older	-0.265322395
pct_white	-0.278550924
pct_adults_with_high_school_diploma	-0.299438281



As you can see from the violin plot, there are no real changes in the values of 'renter occupied households' over the two different years, same for evictions, only there is a difference between the highest outlier (in year a around 26.000, in year b around 30.000).



Cleaning the data

Before cleaning any data, I've explored to find out if there were any NaN's. There were NaN's in these columns

Column name	Actual number of NaN's	Action taken
median_household_income	2	Replaced with median
median_property_value	2	Replaced with mean
pct_adult_smoking	408	Removed column
pct_low_birthweight	126	Removed column
pct_excessive_drinking	810	Removed column
air_pollution_particulate_matter_value	1	Replaced with median
homicides_per_100k	1598	Removed column
motor_vehicle_crash_deaths_per_100k	308	Removed column
pop_per_dentist	190	Removed column
pop_per_primary_care_physician	175	Removed column

Only median_household_income was one of my key features. These had only 2 values missing. Therefore, I've concluded that replacing these values and dropping some of the other columns would not affect my model.

Second, I've looked at the categorical features needed for the model. As I wanted to include information about the neighborhood, I've decided to transform the rucc-feature from 9 categories to 4.

Original:

```
In [55]: train['rucc'].value_counts()

Out[55]: Nonmetro - Urban population of 2,500 to 19,999, adjacent to a metro area                466
         Nonmetro - Completely rural or less than 2,500 urban population, not adjacent to a metro area    370
         Metro - Counties in metro areas of 1 million population or more                         358
         Nonmetro - Urban population of 2,500 to 19,999, not adjacent to a metro area            337
         Metro - Counties in metro areas of 250,000 to 1 million population                      289
         Metro - Counties in metro areas of fewer than 250,000 population                        261
         Nonmetro - Completely rural or less than 2,500 urban population, adjacent to a metro area        210
         Nonmetro - Urban population of 20,000 or more, adjacent to a metro area                 170
         Nonmetro - Urban population of 20,000 or more, not adjacent to a metro area              85
         Name: rucc, dtype: int64
```

Transformed to:

```
In [56]: rucc_cat = {
                'Nonmetro - Urban population of 20,000 or more, adjacent to a metro area' : 'Urban',
                'Nonmetro - Urban population of 2,500 to 19,999, adjacent to a metro area': 'Urban',
                'Nonmetro - Completely rural or less than 2,500 urban population, adjacent to a metro area': 'Rural',
                'Metro - Counties in metro areas of 250,000 to 1 million population': 'Small',
                'Metro - Counties in metro areas of 1 million population or more': 'Large',
                'Nonmetro - Completely rural or less than 2,500 urban population, not adjacent to a metro area': 'Rural',
                'Nonmetro - Urban population of 2,500 to 19,999, not adjacent to a metro area': 'Urban',
                'Metro - Counties in metro areas of fewer than 250,000 population': 'Small',
                'Nonmetro - Urban population of 20,000 or more, not adjacent to a metro area': 'Urban'}
         train['rucc'] = [rucc_cat[x] for x in train['rucc']]
         train['rucc'].value_counts()

Out[56]: Urban    1058
         Rural     580
         Small     550
         Large     358
         Name: rucc, dtype: int64
```

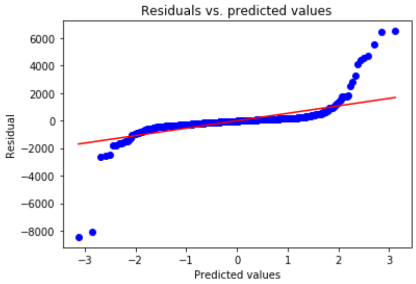
Modeling and testing

Model 1: Linear Regression

As we are trying to predict the number of evictions, which is an integer, I've started with a linear regression model.

The main statistics for this model were:

- Mean Square Error = 618354.8227635269
- Root Mean Square Error = 786.3554048669895
- Mean Absolute Error = 307.74185783057135
- Median Absolute Error = 137.31809594959918
- R^2 = 0.5485537553709087
- Adjusted R^2 = 0.5319925480271784



The R^2 is very low and as this model also predicted negative values (lower than zero), I've decided not to submit this, and first try another model.

Model 2: Lasso Regression

The second type of Machine Learning Model I've tried was a Lasso Regression model. I've done this after exploring on the internet which linear regression model could predict only positive values. In a

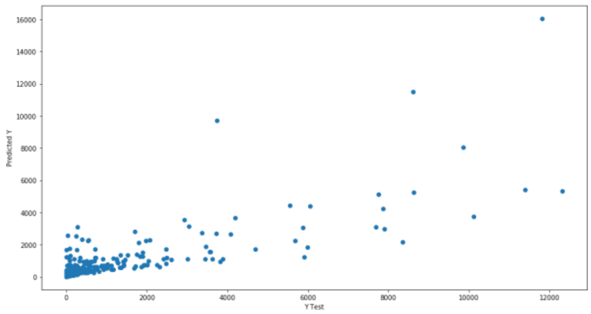
```
In [68]: from sklearn.linear_model import Lasso
         regressor = Lasso(alpha=10.0, max_iter=500, positive=True, fit_intercept=False, precompute=False, selection='random')
         regressor.fit(X_train,y_train)

Out[68]: Lasso(alpha=10.0, copy_X=True, fit_intercept=False, max_iter=500,
            normalize=False, positive=True, precompute=False, random_state=None,
            selection='random', tol=0.0001, warm_start=False)
```

Lasso Regression model you can set the parameter 'positive' to True, letting only positive results pass the model. This is the code I've used for this model:

The main statistics for this model were:

- Mean Square Error = 511164.8666554981
- Root Mean Square Error = 714.9579474734848
- Mean Absolute Error = 259.9891949961935
- Median Absolute Error = 81.24636320908778
- R^2 = 0.6611351904747027
- Adjusted R^2 = 0.6518934229421945



Unfortunately, while submitting this to the Capstone, the result was a R^2 of 0.5, so onwards to another type of model.

Model 3: Random Forest

The third type of model I've tried was a Random Forest Model with 75 trees. The first few times I've tried to submit, the model seemed to be overfitted, as the R^2 was over 0,9 and the result when submitted was around or below 0,5.

Eventually I've decided to leave out the 'county_code' and 'state' columns. There was a difference in unique counties and states between the train and the test data, therefore my model could not be fitted to the test data if I used them.

Therefore I had to go back and look at the input. It seemed that I've dropped way too many columns (all that had NaN-values), so I've decided to drop only 'pct_excessive_drinking' and 'homicides_per_100k' and fill the rest of the NaN-values in the remaining columns with mean-values.

In [5]:
```
#drop columns with too many NaN's in the train data
train.drop('pct_excessive_drinking', axis = 1, inplace = True)
train.drop('homicides_per_100k', axis = 1, inplace = True)

#drop columns that polute the model
train.drop('county_code', axis = 1, inplace = True)
train.drop('state', axis = 1, inplace = True)
```

In [6]:
```
#replace NaN values with mean values
train['median_property_value'] = train['median_property_value'].fillna(train['median_property_value'].mean())
train['median_household_income'] = train['median_household_income'].fillna(train['median_household_income'].mean())
train['pct_adult_smoking'] = train['pct_adult_smoking'].fillna(train['pct_adult_smoking'].mean())
train['pct_low_birthweight'] = train['pct_low_birthweight'].fillna(train['pct_low_birthweight'].mean())
train['motor_vehicle_crash_deaths_per_100k'] = train['motor_vehicle_crash_deaths_per_100k'].fillna(train['motor_vehicle_crash_dea
train['pop_per_dentist'] = train['pop_per_dentist'].fillna(train['pop_per_dentist'].mean())
train['pop_per_primary_care_physician'] = train['pop_per_primary_care_physician'].fillna(train['pop_per_primary_care_physician']
train['air_pollution_particulate_matter_value'] = train['air_pollution_particulate_matter_value'].fillna(train['air_pollution_pa


print(train.shape)
```

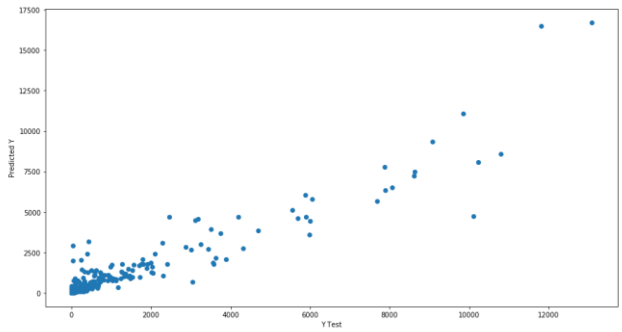
```
(2546, 44)
```

For the final submission, I've decided to keep in all other columns (except for the 4 dropped, as explained). I've also decided not to work with the changed 'rucc' column, so I've kept the original. I've not used the hot encoder nor the scaler, but I've used the 'pd.get_dummies' function.

The Random Forest model seemed to work best for this data, as there were no negative values for evictions predicted. Also the R^2 seemed to give a better result (0,867) than for the linear regression (0,548).

The main statistics for this model were:

- Mean Square Error = 174312.64009813542
- Root Mean Square Error = 417.50765274200114
- Mean Absolute Error = 121.28047105004907
- R^2 = 0.8929581516845616
- Mean R^2 cross validation = 0.8595062180714385



When submitted this final try, I've got a result of 0.7983.

I've tried to tweak the model for even better results afterwards, but it gave me worse results. Furthermore, I've also seen that if you run a Random Forest model several times, you get other results each time. So I've run the model till I got the best score. Therefore, this result was the final one.

Key insights

The key insights gained by performing this Capstone project about predicting evictions were:

1. Before cleaning any features, try to use all features as is in the model to check what the results are. If needed, you can tweak, clean, drop, rename, recategorize or reshape all features if necessary.
2. For so many features, a simple linear regression model might not be sufficient, therefore I will start with a Random Forrest for a future project with so many not directly correlated features.
3. If you need a positive value as an outcome, you can not just use the linear regression model, as this gives you also negative results. Better to use either a Lasso Regression or a Random Forest model to get only positive results.
4. Running a Random Forrest can give you different results after each time you run this. Therefore I've chosen to submit the one with the best R^2 result.
5. All features have a (slight) influence on the model, as you may see here in the overview of the numeric features and their influence:

Variable	**Importance**
renter_occupied_households	0.374551
population	0.339354
median_property_value	0.047879
pct_af_am	0.042619
pct_uninsured_children	0.028304
rent_burden	0.019744
poverty_rate	0.018750
pct_aged_65_years_and_older	0.016523
heart_disease_mortality_per_100k	0.008737
pop_per_dentist	0.008425
pct_asian	0.007898
pct_hispanic	0.006579
motor_vehicle_crash_deaths_per_100k	0.006067
row_id	0.005955
pct_below_18_years_of_age	0.005800
pct_uninsured_adults	0.005346
pct_adults_bachelors_or_higher	0.005192
pct_adults_with_some_college	0.005150
pct_multiple	0.004261
pct_renter_occupied	0.004089
median_household_income	0.004068
pct_civilian_labor	0.003752
pct_white	0.002797
birth_rate_per_1k	0.002598
pct_nh_pi	0.002558
pct_adult_obesity	0.002522
pct_female	0.002430
median_gross_rent	0.002151
pct_unemployment	0.001883
pct_other	0.001792
pct_am_ind	0.001727
air_pollution_particulate_matter_value	0.001471
pct_adult_smoking	0.001377
pct_low_birthweight	0.001143
pct_physical_inactivity	0.001099
death_rate_per_1k	0.001096
pct_diabetes	0.000845
pct_adults_less_than_a_high_school_diploma	0.000787
pop_per_primary_care_physician	0.000584