Predicting Housing Affordability with American Housing Survey Data

1. Introduction

Every other year the Census Bureau conducts the American Housing Survey (AHS). The American Housing Survey contains information about the types of homes in which people are now living and the characteristics of these homes, as well as the costs of running and maintaining them. The American Housing Survey data is used by the Department of Housing and Urban Development (HUD) to generate the Housing Affordability Data System (HADS). The HADS dataset measures the affordability of the housing units in the AHS.

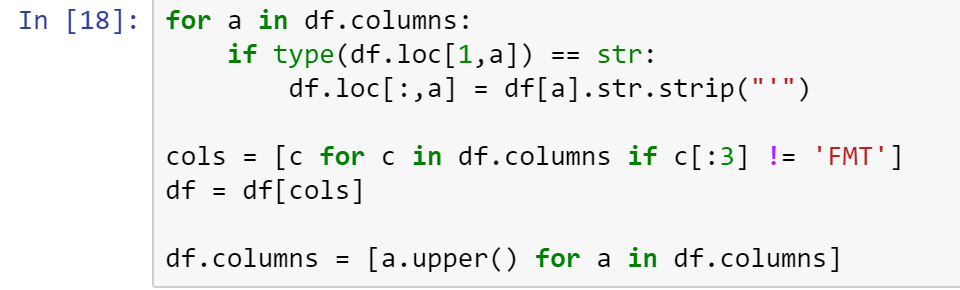
My goal was to combine the AHS and HADS and use this data to predict affordability using features of a housing unit. The purpose of creating such a predictive model is two-fold. Firstly, by examining how important each feature is in predicting affordability, we can learn which features effect affordability the most. Obviously the location and square-footage of a unit plays a major role in determining how affordable a unit is. However, there are many other features ought to be useful in predicting the affordability of a house, and their importance is not as obvious. For example, whether or not a unit has included parking, a garbage disposal, a fireplace, and a laundry room are all useful features in predicting the affordability of a unit, but it is not obvious which is most important.

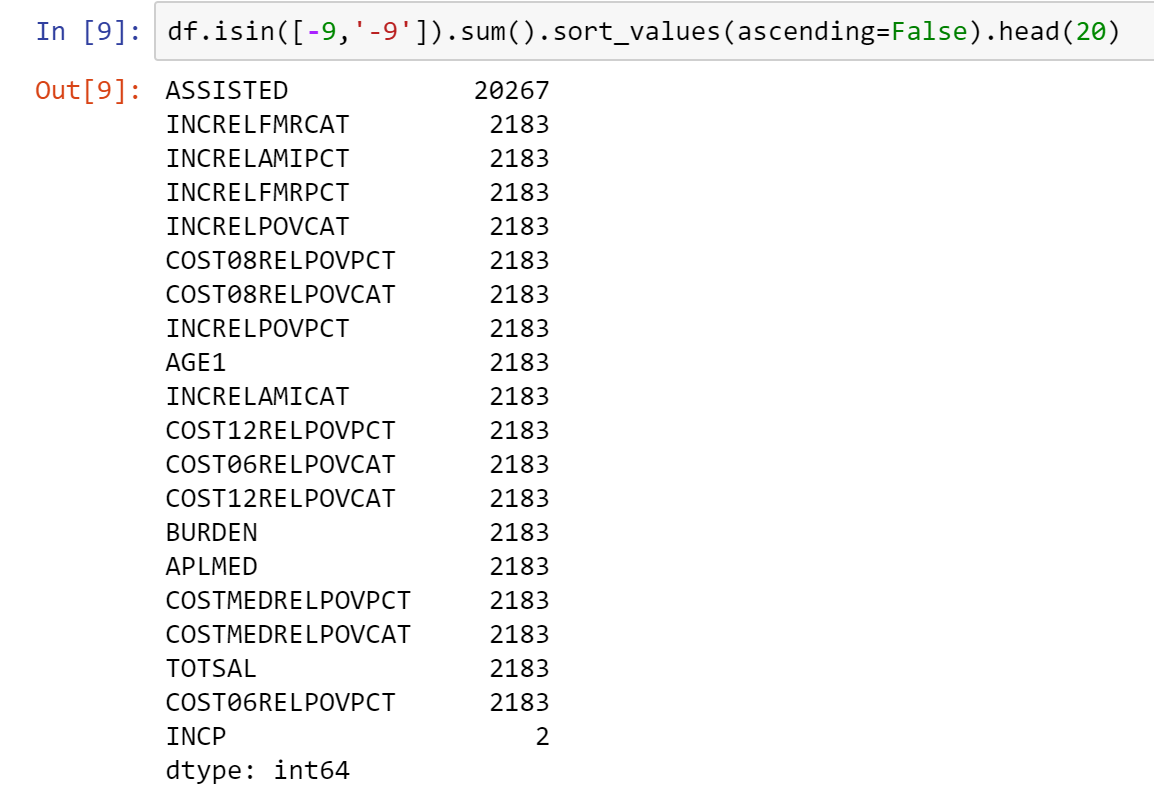
The second reason to build this model is because it can be useful in determining the feasibility of building or buying a certain type of housing unit in a given city. Unless a unit is targeted to the rich, it is very important that the inhabitants of a given city can actually afford the new units being build, or they will just sit vacant. Housing developers already spend lots of time researching to determine what will and will not work. Exploring this model is a great first step in this research, and use of the model could save lots of time. The model could also be useful to the HUD, which is responsible for several grant programs. These programs give grants for the construction, purchase, and upkeep of affordable housing, and this model provides useful information on how that grant money should best be allocated. In order to make the model user friendly, I used a program called dash to tie the model to a simple web application.

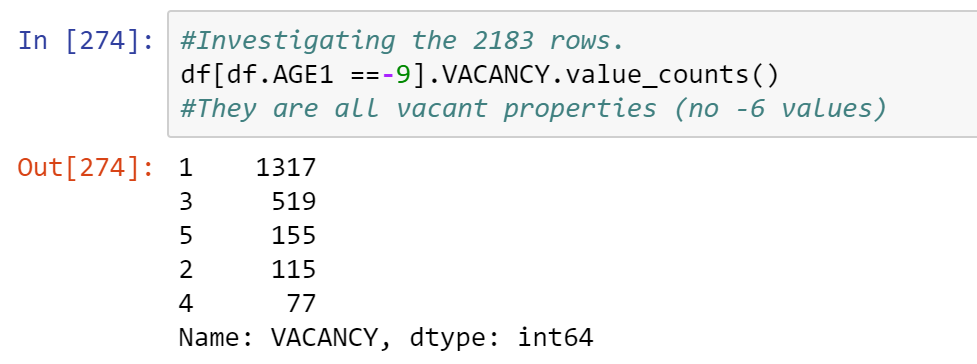
2. Data Wrangling

I started by downloading the AHS and HADS datasets from the HUD website and reading them into an ipython notebook using the pandas read\_csv function. The AHS records hundreds of features about each unit, so I started by looking through the AHS codebook and picking out the features that seemed most important for my goal. I originally intended to work with AHS and HUD data from several years, but unfortunately the variable names and their response categories change a lot between years. As my goal was to predict affordability for current day and the housing market changes a lot over time, I decided to just focus on 2013. AHS data has been released for 2015 but unfortunately, the latest HADS dataset is 2013, so I settled for this year. However, the model and application I have built could easily be updated for 2015 as soon as the data is released.

I merged the AHS on HADS datasets together using **“df = df.merge(d13,on = 'CONTROL').”** The “CONTROL” variable is a control number included specifically to merge the dataframes together. Next I ran this code:

Many of the columns have extra quotation mark characters around them. The first three lines of code remove these extra characters. The next two lines remove redundant columns from the data. The last line fixes inconsistencies in variable capitalization.

Beginning to explore the datasets, I found that values of -6 and -9 appeared in places that made no sense. For example, many housing unit were listed as having -6 units in their building or a market value of -9. Searching through the AHS codebook, I found out that weird values were the result of the AHS and HUD being creating using the program SAS. When converting from SAS to csv format, “not applicable” gets converted to -6, and “no response” get converted to -9. To find out where these -9 values were occurring, I ran this function:

 As you can see, most of the -9 values were in the “ASSISTED” variable, which is a categorical variable representing what type of government aid the people living in a given unit are receiving. The variable has nothing to do with my purpose so I decided to simply drop this column from the data. There are several variables that have exactly 2183 -9 values. It is probable that the -9 values occur in the same 2183 samples of data. Many of the variable here have something to do with the occupants of the unit; for example, AGE1 is a variable recording the age of the primary occupant of the unit. Perhaps these units are vacant? The VACANT variable has 6 categories. Categories 1-5 represent different statuses of a vacant property. The sixth and most common category is -6 or not applicable, meaning that a property is occupied. To investigate, I ran:

Clearly the -9 values are for vacant properties. Further investigating these 2183 row, I found that many other columns in these vacant rows show responses of -6. There is little information on these vacant rows and they make up only 7% of the data, so I decided to drop these rows

2. Methods

I decided to try and predict the variable COSTMEDRELAMICAT, which stands for Median Cost Relative to Area Median Income Category. This variable is calculated with two assumptions. The first being that housing cost is calculated for a person paying the median mortgage rate for their unit. The second assumption is that the most a household could afford to spend on housing costs is 30% of their total income. The variable answers the question, “what percentage of Area Median Income would a household need to make in order to afford monthly housing costs for this unit?” Originally the COSTMEDRELAMICAT variable was split into 7 different categories: 1: less than 30%, 2: 30-50%, 3: 50-60%, 4: 60-80%, 5: 80-100%, 6: 100-120%, and 7: over 120%. So if a housing unit has a COSTMEDRELAMICAT variable listed as 3, that would mean a person making between 50% and 60% of the Area Median Income.

There is a major trade-off in this project between the precision of the classes and the accuracy of the model. For example, the COSTMEDRELAMICAT variable has many classes, most only covering 20% difference. While this give more information, it is problematic because differences between a property and neighboring categories are likely very subtle and inconsistent. In fact, when I tried to build a model predicting all seven categories, accuracy scores were only 35%. Additionally, the additional precision in reporting from using seven categories is not all that informative because the measurements are made based on the two coarse approximations in the first place.

To remedy this, I assigned every unit a new variable, AFFORDABLE, based on whether the unit is or is not affordable at 100% of the Area Median Income. COSTMEDRELAMICAT categories 6 and 7 were assigned a 0, meaning “not affordable” and categories 1-5 were assigned a 1, meaning “affordable.” The AFFORDABLE variable is far easier to predict and it greatly simplifies COSTMEDRELAMICAT variable to make it easier for users to understand. Under the assumption that the average person in a given city makes the area median income and pays the median mortgage rate, the questions the AFFORDABLE variable answers simplifies to, “Is this property affordable for the average inhabitant of this city?” This simplification makes the model much easier to understand and increases the accuracy of the predictive model to about 83%.

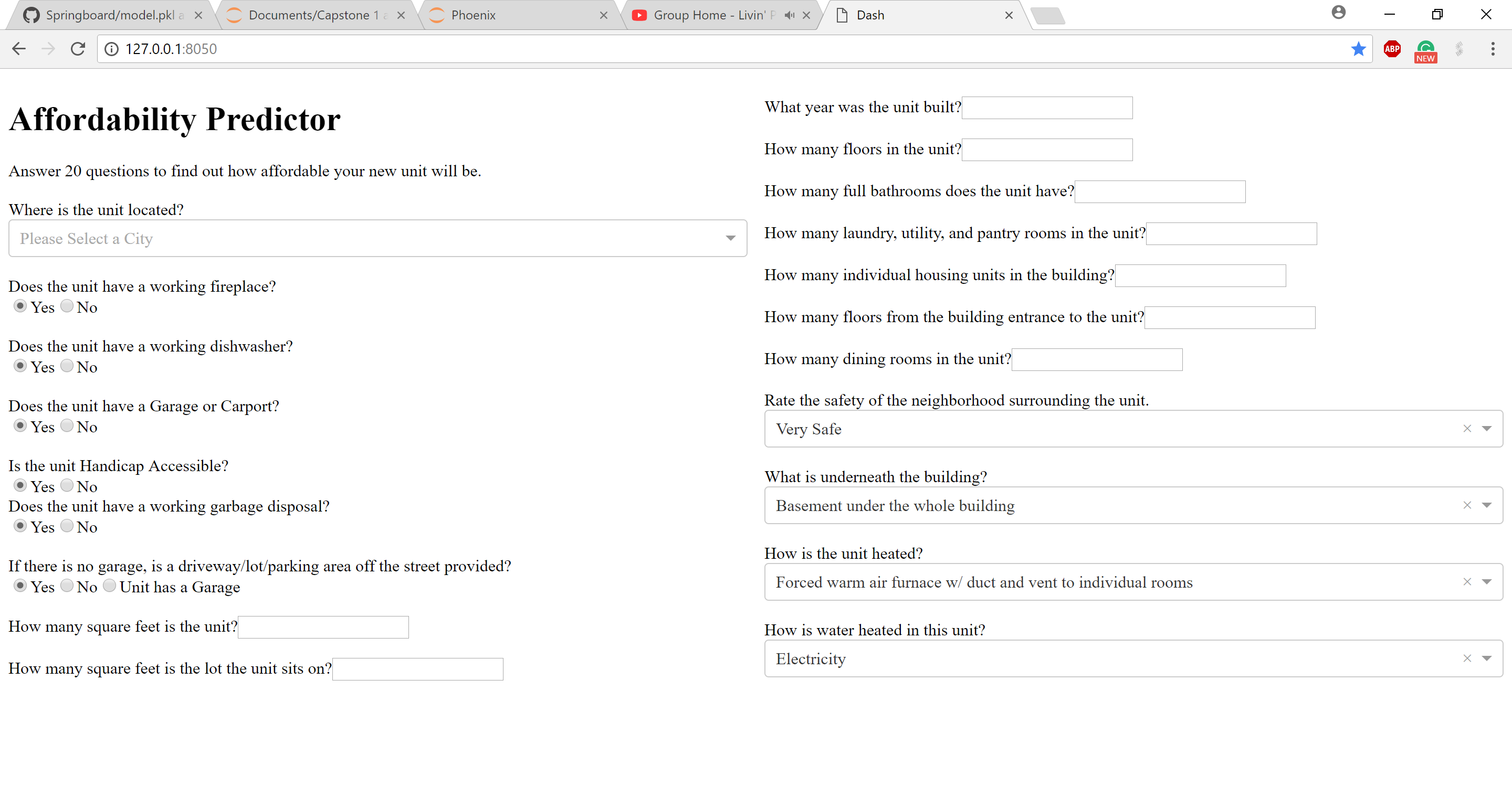
The location of a housing unit is recorded in the AHS under the SMSA (Standard Metropolitan Statistical Area) variable. The variable correspond to four digit codes which refer to metropolitan areas and can be found at <https://www.census.gov/population/estimates/metro-city/80mfips.txt>. Many of the units have codes in the range 9993-9999, which means that their SMSA code has been redacted for privacy reasons. The location of housing unit is important in predicting affordability, so I chose to focus on data from the metropolitan areas with the top 15 most AHS responses listed. The properties considered come from: New York, Los Angeles, San Diego, Philadelphia, Anaheim, Detroit, Chicago, Dallas, Houston, Seattle, Minneapolis-St. Paul, Oakland, Boston, Phoenix, and Washington D.C. There is a non-redacted version of the AHS which is available by permit to established researchers, and if I could obtain access I could make the model work nationwide.

3. The Web Application

I wanted to create an interface that allowed a person with no programming experience to utilize the model. In order to do this, I first created a program in my ipython notebook to prototype the application. The program uses the raw input function in python to ask users about features of the housing unit and then saves them to variables. The variables are then joined into a single sample, and the sample is predicted by the model. The original model predicted based on 59 features but asking a user 59 questions is impractical. Instead, I used the 20 features that were most important in the original model to build a new model, resulting in a loss of accuracy of about .5%.

The actual web application was built using Plotly Dash, a program python model that coverts python code into html. The simplified model was saved using Pickle and then loaded into a file used to generate the application. I added all of the questions to the dash application as either drop down menus or text entry. Once all questions on the web application are answer, the responses are feed into a function where they are joined into a sample and predicted by the model. The dash application then prints out a statement underneath the last question stating whether or unit is affordable for an average person in whichever city was listed.

The Dash application has several advantages over the prototype python program. It is better looking, removes ambiguity from several questions (i.e. should a person 1 or not applicable when indicated how many apartments are in the building), and live updates once all questions are answered. In order to use the web application, go to: https://github.com/thewho14/Springboard and download “model.pkl” and “app.py” and save them in the same folder. Then run the file called app.py and visit <http://127.0.0.1:8050/> in your web browser. It should look something like this.

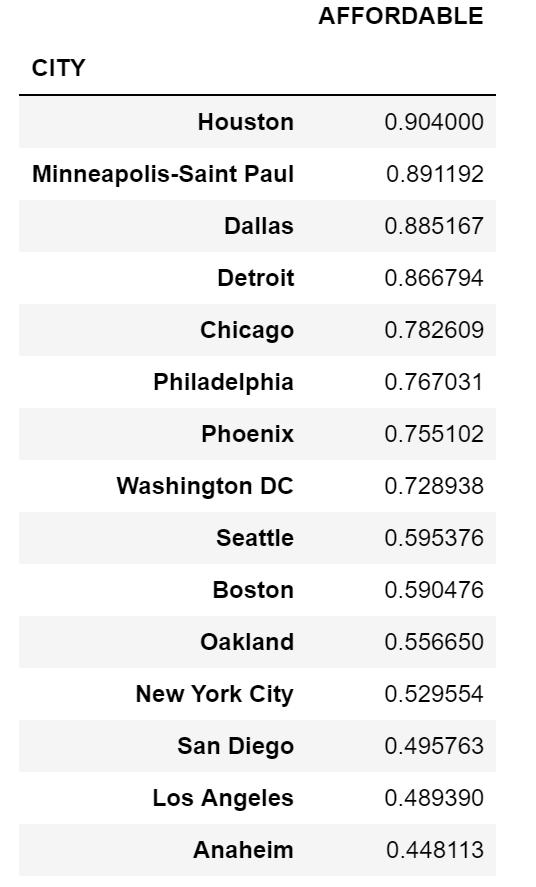


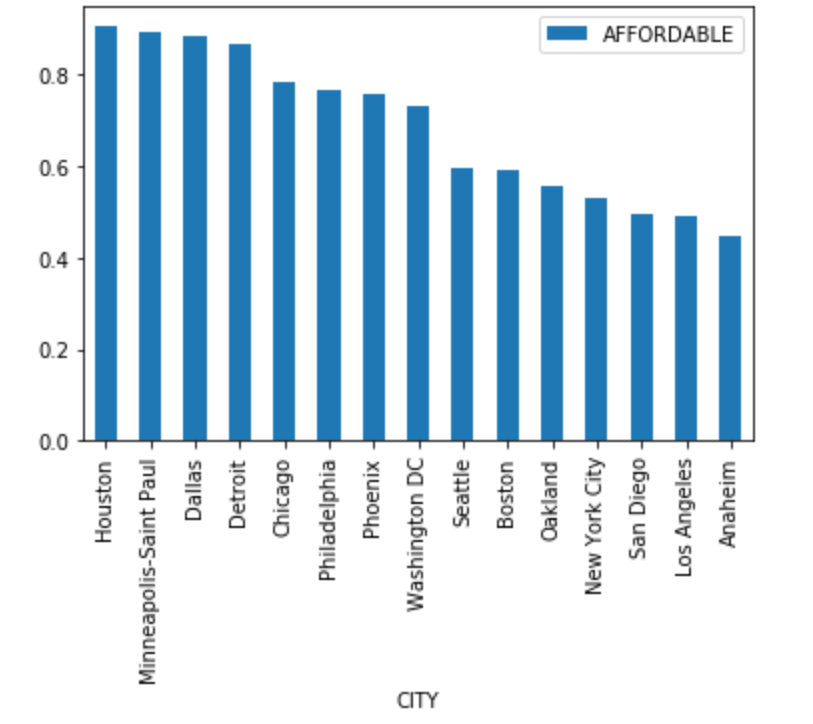
4. Data Exploration and Visualization

The binary nature of the AFFORDABLE variable has a very nice property. I examine the relationship by individual features and affordability by using the function:

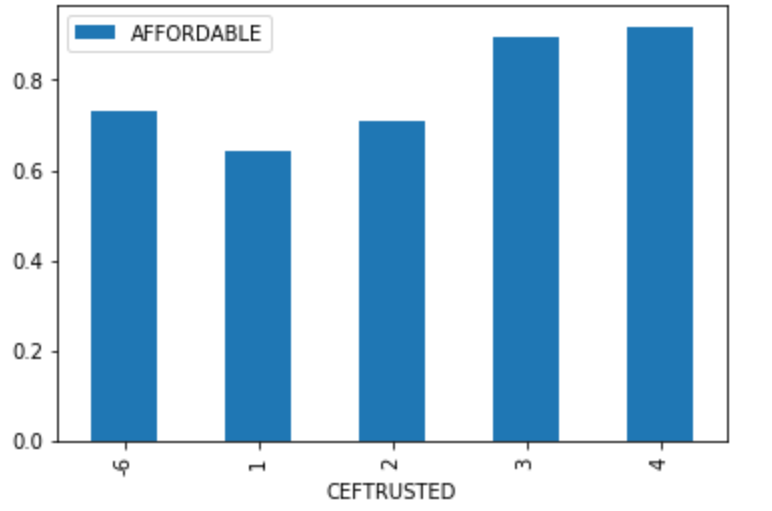
**dl[['AFFORDABLE','CITY']].groupby('CITY').mean().sort\_values('AFFORDABLE', ascending=False).**

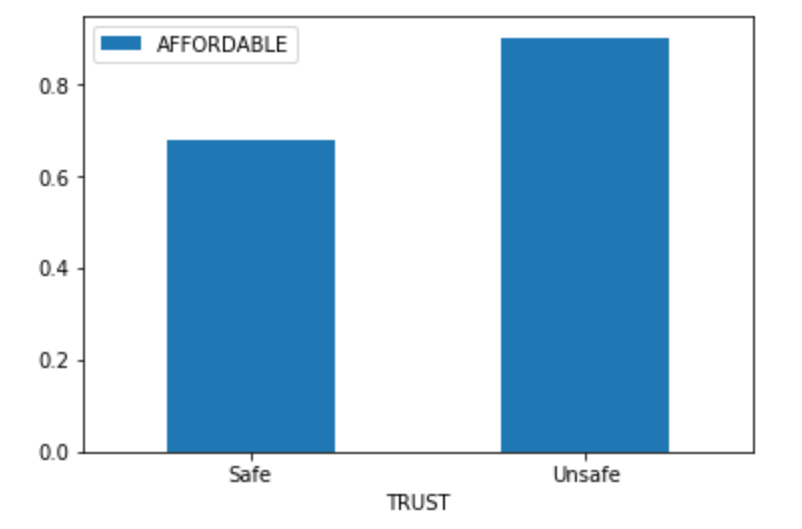
The function groups by the feature CITY and calculates the mean value of the AFFORDABLE variable for these groups. Since AFFORDABLE only takes values of 0 and 1, when the mean is calculated, the numerator is just the number of 1’s (affordable units) and the denominator is the total number of properties. Therefore, the mean is actually the proportion of affordable properties in each category. For example, 90.4% of housing units in Houston are affordable.





Unsurprisingly, California cities and NYC are the least affordable.

CEFTRUSTED is a variable is a rating of how safe the neighborhood surrounding a unit is, with 1 being safest and 4 being least safe. As one would expect, less safe neighborhoods tend to be more affordable and vice versa. Simplifying the problem by combining groups 1 and 2 into a “Safe” group and groups 3 and 4 into an “Unsafe” group, I find that housing units in unsafe areas are 22.56% more affordable than housing units in Safe areas. This result occurs under the null hypothesis with a probability of 5.15\*10^-52, so it is clearly significant.



We would also expect that more bathrooms means less expensive.

