Full Report of Homework 3

Bipul Islam
(SBU # 111578726)

Task 1. Create a Scoring Scheme for Listed Houses

Methodology

- We select Attributes that are mostly complete and generally describe a property holistically
    - Some inspection shows simple imputation is fallible to outlier irregularities,
- Lot of rows though non-empty likely has garbage values. eg: There are houses with structure tax 1,2,3 etc. similarly for lotsizes, finished square feet etc.
    - So it's a safe bet to reject all data in bottom and top 5%ile for each column.
    - This gets us clean data of 1.3million points.
- The distributions of attributes are mostly skewed, so
    - Get quantile scores for each columns, i.e Each value gets a score between 0-1.
    - The columns with descending desirability get scores of the form (1 - quant. score) i.e higher is worse.
- Now all columns are transformed to uniform(0,1) distributions.
    - We call these component scores of attributes for the listing
- Score for each listed property is chosen to be a linear combination of the component attribute scores.
- **I choose to combine component scores using Principal Component 1 coefficients**.
    - Finally we obtain a distribution of the scores for the data pool

I summarize Desirability as:

- More Bedrooms & bath rooms, more desirable- will get higher scores
- More finished square footing and more lot area means more desirable
- Buildings built later implies newer buildings, hence more desirable
- Lower tax value means higher desirability, Lower scores will mean better

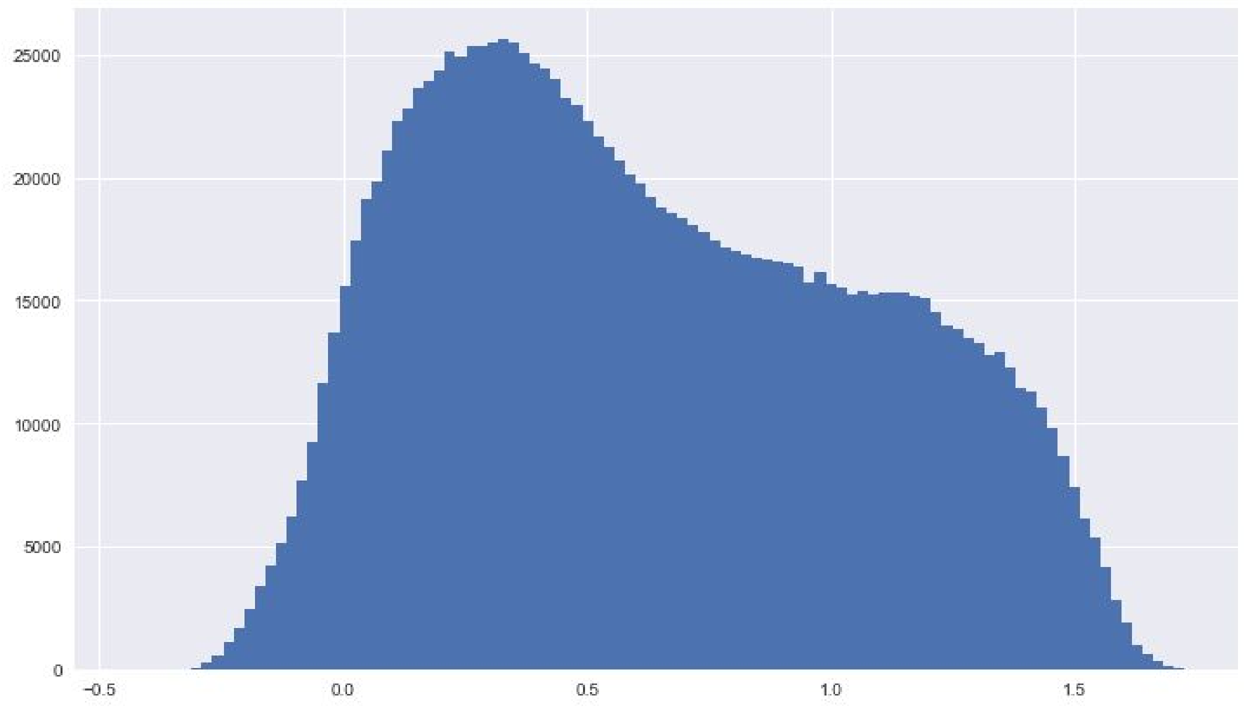
Combination coefficients obtained from PC1 are:
(-0.520797, 0.235099, 0.430328, 0.546572, 0.172876, 0.399611)

Combination formula:
desirability_score = **-0.520797***structuretaxvaluedollarcnt_score + **0.235099***bedroomcnt_score + **0.430328***bathroomcnt_score + **0.546572***calculatedfinishedsquarefeet_score + **0.172876***lotsizesquarefeet_score + **0.399611***yearbuilt_score

Desirability Score Range

- It gives **each house a score between -0.5 to 1.8**, then we sort by this score and assign ranks

Following is the histogram of the scores allocated to the houses by this scheme



Here along the x-axis we have the desirability scores assigned to houses by our scheme, to each house and on the y-axis we have the frequency of houses at various scores. We rank the houses by sorting them by the desirability score we have assigned to each house. One interesting observation here, there are significantly more houses with scores in the range 0-0.5, than there are in other buckets like 0.5-1.0 or 1.0-1.5.

Parcel ids of top 10 houses by this scheme (Scores in range: **1.73** to **1.71**)

14202712,17257315,14145142,17300511,14201457,14232873,17051508,14144920,17264250,17263523.

Parcel ids of the bottom 10 houses by this scheme (Scores in range **-0.35** to **-0.44**)

14132206,17066219,17097563,14404129,13878336,13893976,17070566,14118787,13839592,13951584

Task 2. Choosing a Distance metric

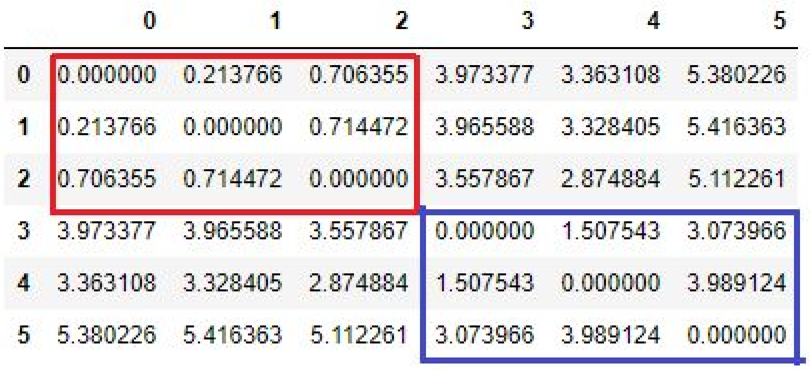
- This task requires us to figure out a distance metric between properties.
- The features being used:
    - property quantifying variables--
        - bathroom, bedroom counts,
    - finished living area,
    - year built, and
    - structure tax.
    - Geographic attributes:
        - Latitude and longitude

The Distance metric chosen Mahalanobis distance

- This is an improvement over the Euclidean distance, as Euclidean distance gives equal importance to all features.
- Mahalanobis distance uses the covariance of the feature data to normalise them
    - Mahalanobis distance is available in scikit learn package
    - It can be computed using **DistanceMetric** function of sklearn.neighbors
    - **DistanceMetric**.get_metric('mahalanobis', **cov_matrix_of_data**) creates the necessary object that can call pairwise distances

Evaluation of the distance metric

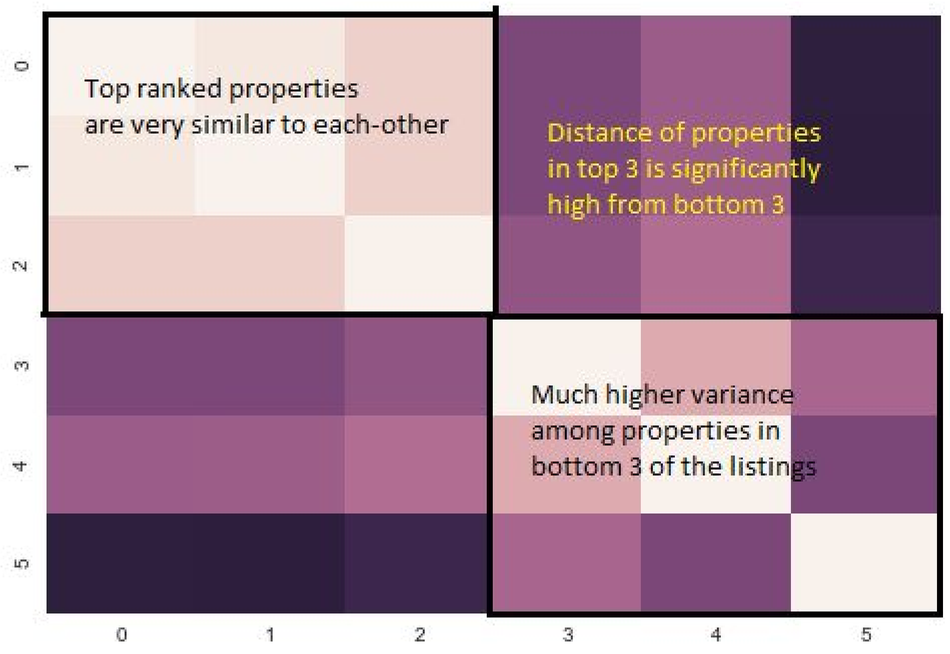
- We obtain the top 3 and bottom 3 houses obtained in the previous task.
- The pairwise distance among these houses by this distance metric looks like this:



	0	1	2	3	4	5
0	0.000000	0.213766	0.706355	3.973377	3.363108	5.380226
1	0.213766	0.000000	0.714472	3.965588	3.328405	5.416363
2	0.706355	0.714472	0.000000	3.557867	2.874884	5.112261
3	3.973377	3.965588	3.557867	0.000000	1.507543	3.073966
4	3.363108	3.328405	2.874884	1.507543	0.000000	3.989124
5	5.380226	5.416363	5.112261	3.073966	3.989124	0.000000

    - We can clearly see, houses 0,1,2 (in red) which are among the top 3 houses and similar in listing are at a distance of >= 3.4 from the bottom 3 listings (in blue).
    - One more interesting thing to observe here is that house to house distance among the top 3 houses are much lower than what we see among bottom three.
    - Intuitively this may mean properties ranked highly in our desirability score are much more similar to each other than than the properties which rank low in the desirability score.

- Following is an interesting representation of the pairwise distance among these 6 houses by using heatmap as a hack, that elucidates the point discussed above
- We know sns.heatmap ultimately accepts a square matrix where each cell has a number that gets mapped to a color intensity. So I used the matrix shown above and put it through heatmap call to get this.



Task 3. Clustering with the proposed distance function

Custom distance function is mostly non-pluggable in standard clustering algorithms of sklearn. However version 0.19 allows create and use custom distance metrics in DBSCAN clustering algorithm.

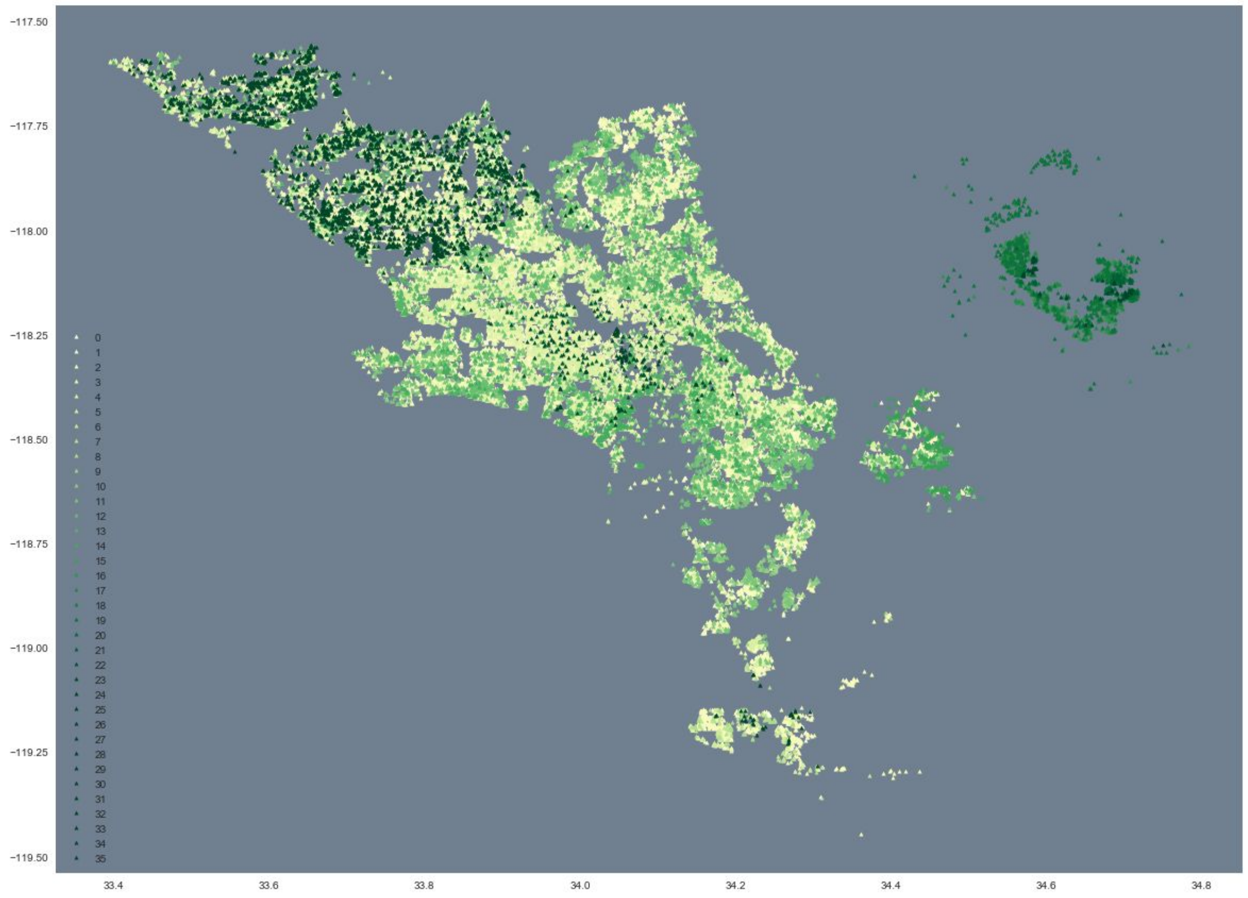
- Since the **Mahalanobis distance computation** takes care of the normalization, we don't need to separately normalize the features which would have been required in case of euclidean.
- With some trial and error, model parameters are set at
  - **epsilon = 0.5** (The maximum distance between two samples for them to be considered as in the same neighborhood)
  - **min_samples = 40** (The number of samples in a neighborhood for a point to be considered as a core point. This includes the point itself.)
  - **Algorithm = 'ball_tree'** (used by the NearestNeighbors module to compute pointwise distances and find nearest neighbors)
- We run the algorithm **on a sample of 60k points**

Clustering Output:

- After some playing around, we obtain an output that 37 clusters
    - 36 clusters are distinct similarity types
    - One cluster tagged with -1, are a listed a noisy samples that could not be clustered. We remove these cases from our dot-plot.

Dot plot of visualization of the points:

- Here we plot points which have cluster ids between 0-36
- We choose the YIGn color map
- Plot background is set to **slategrey** in order to make the clusters have more contrast.



Areas prominent in this plot is the Coastline that extends from Long beach (centre) to San Diego. The small disjointed location far from the coastline is probably the area around Palm Springs

Task 4. Select an external dataset to enhance the existing dataset

Two data sets have been augmented with the current dataset. Here are my external data sources, they are publicly available:

- Month on month inflation rates.
- Month on month changes in REIT Index

What is a 'Real Estate Investment Trust - REIT'

A REIT is a type of security that invests in real estate through property or mortgages and often trades on major exchanges like a stock. REITs provide investors with an extremely liquid stake in real estate. They receive special tax considerations and typically offer high dividend yields.

REITs, an investment vehicle for real estate that is comparable to a mutual fund, allowing both small and large investors to acquire ownership in real estate ventures, own and in some cases operate commercial properties such as apartment complexes, hospitals, office buildings, timber land, warehouses, hotels and shopping malls.

All REITs must have at least 100 shareholders, no five of whom can hold more than 50% of shares between them. At least 75% of a REIT's assets must be invested in real estate, cash or U.S. Treasuries; 75% of gross income must be derived from real estate.

REITs are required by law to maintain dividend payout ratios of at least 90%, making them a favorite for income-seeking investors. REITs can deduct these dividends and avoid most or all tax liabilities, though investors still pay income tax on the payouts they receive. Many REITs have dividend reinvestment plans (DRIPs), allowing returns to compound over time.

Source: Real Estate Investment Trust (REIT)

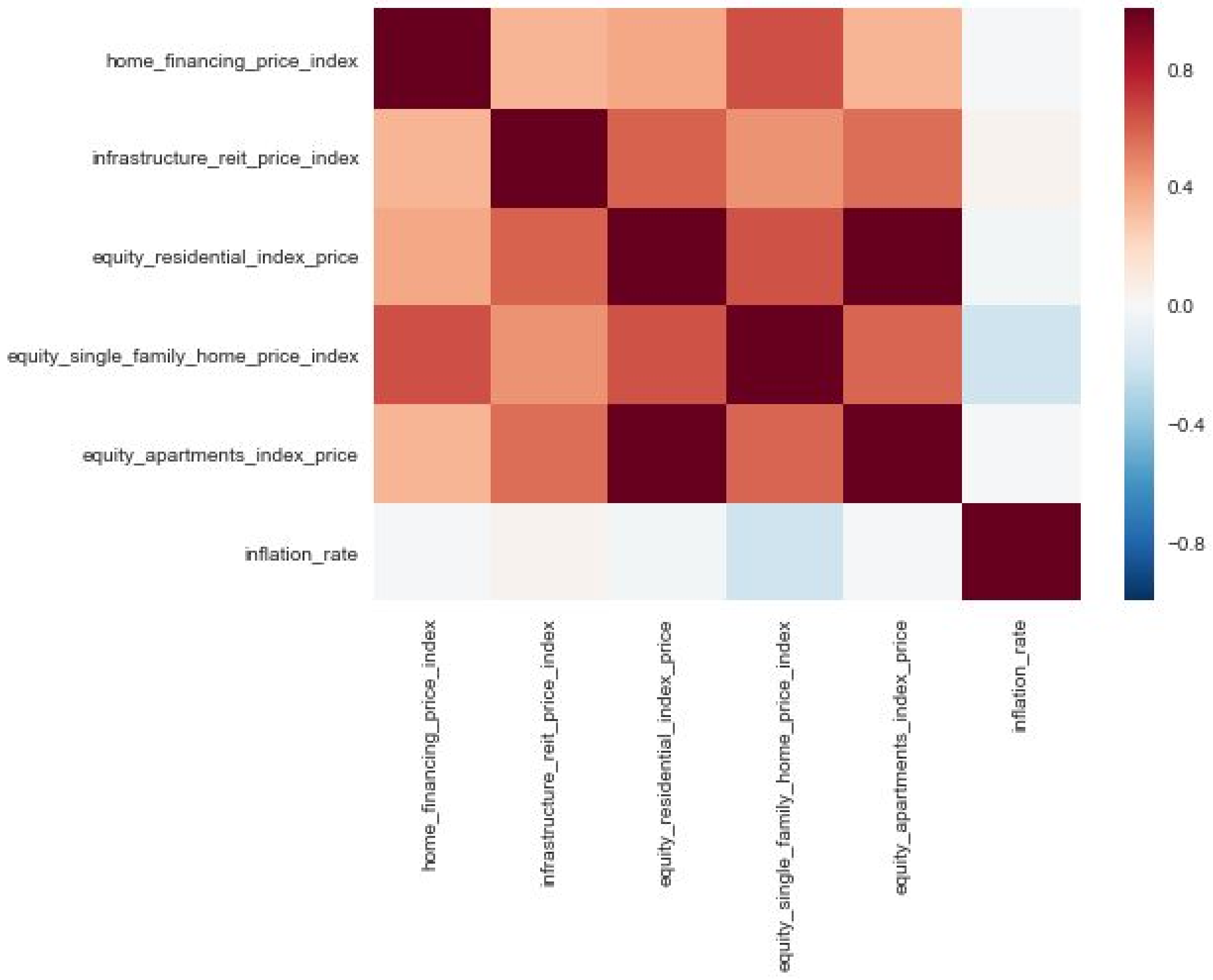
Out of various possible REIT indices following have been used for this task (month on month):

- Home financing price change
- Infrastructure price change
- Residential equity price change
- Single family home equity price changes
- Apartment equity price changes

Along with these we use month on month inflation rate change.

Relationship among the variables thus selected is explored in the following heatmap. It is interesting to see that there is some good correlation among the various index and pricing data that was obtained. However it is also interesting to note, inflation rate almost 0 to inversely correlated with other indices in the palette.

It is however interesting to see that only inflation rates have a very minor correlation with log-error values (0.02). May be inflation rate can be useful when used in presence of the structure tax with some adjustment as well ass other details of the house. I present the correlation matrix below.



Task 5. Building a Predictive model with existing as well as external data

Kaggle Score improved to 0.0650159 in HW3 (from 0.0650219 in HW2)

Made close to 5 submissions but this improvement happened in the first try subsequent tries didn't see any improvement

Here we use the external data set and learnings we obtained while performing previous tasks to create an improved model for the data set.

- First step is to train a model using the complete data we have for the two sets of zillow data.
- Second we use that to predict costs for the time points in the sample submission data set.

General Discussion about the methodology

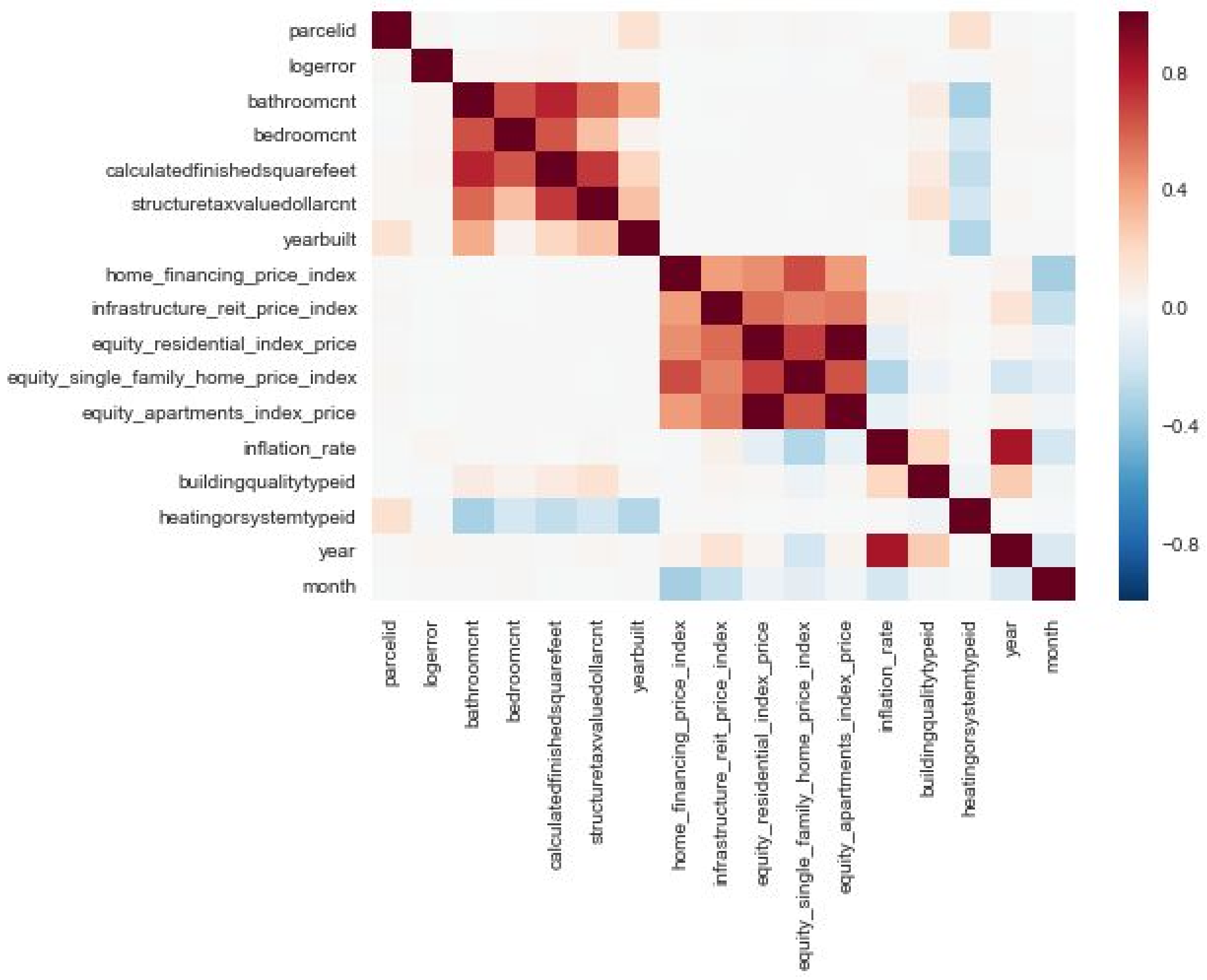
First we chose some attributes for our classification:

- I tried doing away with the host of property specific categorical variables I have used in my previous set of analysis in other homeworks, and adding a set of external data.
- The external data definitely boosted the performance of the algorithm a bit, however I think some more time could have been spent getting more appropriate data.
- **General data cleaning strategy** for me was:
    - Imputing the NULLs in numeric data, with median values and then normalising.
    - For time related categorical variable (month) I used a one-hot encoding to create the dummy columns.
- I did try out using quantile score based cleaning on the columns but I think it dropped off far too many rows that lost subtleties about the data, however fine tuning that may help towards the model performance.As there is definitely some columns with values quite different from normal

Creating the dataset

- Read: train_2016, train_2017, properties_2016, properties_2017.
- Join train and properties for 2016, and similarly for 2017 on **parcelid** columns.
- Append the two joined data sets.
- Create the year_month column for this data set.
- Join this data set with the indices_and_inflation_data data set to add external data
- We consider the following attributes in the data for my models:
    - Numeric data describing the property
      Bathroomcnt, **bedroomcnt**,
      Calculatedfinishedsquarefeet, **structuretaxvaluedollarcnt**
    - Numeric data from external dataset as mentioned in the previous task
    - Categorical variables:
      Buildingqualitytypeid, **heatingorsystemtypeid**, **year**, **month**
    - Oldness attribute: specifying how old the property is.

Heatmap of feature variables



Hence Analysis of the variables:

- The external economic indicators from Realtor indices seem to correlate amongst each other well. Variables in existing data correlate well.
- It is interesting to not month on month relative inflation rate is the sole independent variable that has a slight correlation with existing data.
- It is also interesting to note that inflation rate has negative correlation with some of Realtor market indices

Building Test and Training sets

- We use a 75%-25% split of train and test on our cleaned dataset

Models:

- Two models were used.
- First cut model was a **linear regression with no-intercept** (as one hot encoded variables were there)
    - r^2 was 0.003 with RMSE ~ 0.03 **(on test data)**
- Second model used was **Random forest**
    - Parameters: n_estimators= 50, max_depth=4
    - r^2 was 0.014 with RMSE ~ 0.026 **(on test data)**

The second model was used for the submission to Kaggle, returned score: **0.0650159**

Additional information:

- We had to submit data points for 201610,201611,201612,201710,201711,201712
- External data was available for only upto september of 2017, so artifical data values were used for external data columns using some approximation
    - **For 201710**
      I used the values for 2017-09 which was available.
    - **For 201711**
      I used average of values for 2017-09 & 2017-08
    - **For 201712**
      I used the average of values from 2017-07, 2017-08, 2017-09

Task 6. Permutation Test

In the 42k test set data points, I established the baseline score (M.a.d) at 0.068 using the original log error and predicted log error.

- Then I started running random shuffle/permutations of the 42k log errors and computed m.a.d score with the original log_error_prediction each time.
- The thousand scores thus obtained are then compared against the baseline.
- 0% of the time the scores were better than the model that has been build

Using permutation_test_score function from sklearn.cross_validation we obtain:

- p value of prediction over 10k permutations is: 0.019801980198