The Price is Always Right

Meir Joffe
324680461
meir.joffe@mail.huji.ac.il

***Abstract*—This paper contains a description of a study I performed concerning real estate in England and Wales. In particular, this study focuses on the prices of homes sold between 1999-2018. Real Estate prediction is an important problem with many real-world applications, including helping buyers and sellers with what is for many of them far and away their most valuable asset. Additionally, it directly affects the careers and livelihoods of real estate brokers and investors. It is a difficult and complex problem, as each market is very different and there are many factors that have the potential to affect prices. In this project I attempt to study some of these factors and apply Linear Regression, Random Forest Regression and Multi-Layer Perceptron models to learn and predict the real estate prices and compare the results. The large volume of data presents significant challenges when it comes to data processing and model training (for example Linear Regression cannot be applied directly, instead approximated via Stochastic Gradient Descent). Additionally, I perform a basic time series analysis to the home prices and attempt to predict the average price in the years to come.**

I. Background

The real estate vertical is a core component of any economy. Every year, millions of homes are sold across the world, including approximately 1 million per year in England and Wales (during the span of 1999-2018). This vertical sometimes has an outsize effect on the economy because, at least in highly developed nations, it directly affects large numbers of people who buy and sell their own homes, and the homes represent a significant portion of their overall assets.

The pricing of houses, though, is affected by a larger number of factors, some seemingly more obvious than others. Some of the more obvious factors include location, property type (for example apartment vs. house) and age (new vs. old). However, there are additional potential factors that are sometimes less obvious, including the effects of major political and economic events such as Brexit (the UK referendum on and subsequent exit of the EU), the financial crisis of 2008-2010 and more mundane things such as household income, quality of schools and even demographics.

In addition, there are factors that affect the pricing of a home not on a year-by-year basis, but rather over time. These factors include inflation and economic prosperity, where the former simply increases (or decreases) the price without increasing the value, and the latter increases/decreases both the price and the value.

Analyzing and learning real estate prices is an important problem to study because it has many practical implications:

1. It can help potential sellers gauge the value of their home before putting it on the market.
2. It can help potential buyers understand the market and help them find a home that both suits their needs as well as their budget.
3. It can help real estate brokers connect sellers and buyers.
4. It can help investors understand the trends in the market and empower them to make more informed investments.

Knowing which factors have the greatest effect on the overall house price is very useful as well. For example, if a district's median income has a significant positive effect on the home prices in that district, then an observed increase in this income can be used to help gauge the value of a home.

Or, if a district's smoking rate has a significant negative effect on home prices in the area, then a focused campaign by that district's government to help reduce smoking rates may, in addition to improved health, have the added benefit of increasing home prices. This would also provide an additional financial incentive to the district's residents, on top of the financial incentive of smoking cessation.

In this project I set out to explore some of the factors and features that affect the overall price. My project essentially has 4 core elements, each of which will be discussed in more detail:

1. Collection and processing of the data.
2. Analysis and visualization of the data.
3. Learning the data to predict home prices.
4. Time series analysis of the average home prices.

I attempt to learn the data both on a year-by-year basis and as a whole (i.e. learning the data from all of the years together), using the following models:

1. Linear Regression – implemented through Stochastic Gradient Descent.
2. Random Forest Regression.
3. Multi-Layer Perceptron.

II. Data Collection

The UK government collects and publicizes a yearly dataset containing all records of homes sold in England and Wales during the previous year. In order to be able to gain insights from trends in the real estate market that metastasize over time, I collected the house price data from 1999-2018, a span of 20 years (at the time I began this project, the data for 2019 had not yet been released) [1].

This data came in 34 files, with 2 parts for each of the years 1999-2012 and 1 part for the years 2013-2018. All told, the files took up approximately 2.84gb of space and contained over 20 million data points, each representing a single home sale. This averaged to approximately 1 million per year, though the exact number per year varies greatly.

During this period, the population of England grew every year from just over 49 million people in 1999 to just under 56 million in 2018, i.e. the population grew by just over 14%. Similarly, the population of Wales grew every year from 2.9 million in 1999 to just under 3.13 million in 2018, approximately a 7.8% population growth [7]. Over time, as

the population grew, more housing units were added and the number of people in the market buying and selling houses grew as well. Thus, the number of houses being sold goes up on a yearly basis. However, the financial crash of 2007-2009 devastated the real estate market, causing prices to drop, but its wider effects were felt by large swaths of the population, especially the middle-class, who tend to be homeowners. As a result, we see a steep drop in the number of homes sold between 2007 and 2008. As the economy started to recover, so did the housing market, with the number of homes sold increasing year after year. This upward trend holds until the effects of Brexit start to be felt across the economy in 2018, and, correspondingly, we observe what is essentially a freeze in the overall number of houses sold, when compared to the previous year.

The files containing the house price data all share the same features (columns), which include:

1. Transaction ID (unique)
2. Price
3. Date of Sale
4. Old/New
5. Type of home
6. Locality (neighborhood)
7. City
8. District

In addition to that, I was able to find data collected by the Office of National Statistics containing the mean and median income levels by district for each of the 4 countries that make up the UK: England, Scotland, Wales and Northern Ireland. I collected this data for each of the years I collected home prices for, i.e. 1999-2018. This data contained 2 columns I was interested in:

1. District
2. Mean/Median Income

Finally, while looking for additional economic data, I came across The UK Prosperity Index, procured by the Legatum Institute [3]. This is an organization that in 2016 released a report ranking each of the districts of the UK's 4 countries. The districts were ranked individually by 43 parameters, across the following 7 categories:

1. Economic Quality – examples: unemployment rate, child poverty rate, job satisfaction
2. Business Environment – examples: average broadband speed, entrepreneurship rate
3. Education – examples: population with no qualifications rate, education attainment at 16 rate, truancy rate
4. Health – examples: life expectancy at birth, obesity rate, infant mortality rate, smoking rate
5. Safety & Security – examples: violent crime rate, theft rate, feel unsafe rate
6. Social Capital – examples: voter turnout rate, rely on friends rate, volunteering rate
7. Natural Environment – examples: air pollution, protected areas, waste per head

III. Data Processing

After collecting the data from the sources described above, I set about processing the data. The different data processing procedures I performed can be classified into 3 general groupings:

1. Cleaning and Organizing
2. Aligning and Combining
3. Model Preparation

A. Cleaning and Organizing

The first thing I needed to do once I successfully gathered the data was to clean and organize it.

For the home price data, some of the files came in 2 parts for each year, while others came in just one part. I started by combining each of the years into a single file containing all of the records for that year. I then added headers and an index column to make the data easier to use in (pandas) dataframe format. I found that a few of the values in the district column were missing. To solve this, I looked at the city column, found all of the records of houses sold in the same city in the same year, and took the most common district value (mode) for that city. I then proceeded to drop all columns I deemed unnecessary – columns I considered to be of no value with respect to the data I was interested in. These included the 'tid' (the unique transaction id), 'paon' and 'saon' (Primary/Secondary Addressable Object Name) which are house numbers or names (and often, especially in the case of 'saon', empty), status and ppd_type (two columns whose values were identical for all rows).

For the income (mean and median) data, the first thing I did was to extract all of the relevant data from excel sheets (when relevant) and convert everything to standard csv format (all done in the code). I then went and split the files containing the mean and median incomes by year – there was a single file containing all the income information for the years 1999-2017 [2]. This was done to enable the data to be handled (and later combined) on a yearly basis.

B. Aligning and Combining

The next step in processing the data was to transform the data in such a way that the mean and median incomes, as well as the prosperity data, could be joined to the home price data to create a single dataset with all of the economic and other features alongside the home sales records.

It made the most sense to join (inner join) both the mean and median incomes, as well as the prosperity data, with the home price data on the district column, since each had a column for district. Therefore, I needed to ensure that all of the district values were accurate and identical across each of the files both within a specific year as well as across all years.

Periodically – for reasons of local government efficiency – the UK government makes changes to districts. These changes can essentially be one of 3 types:

1. Creation of a new district – generally by merging existing districts or by carving out areas from one or more adjacent districts.

2. Abolition of a district – generally by it being absorbed into one or more new or existing adjacent districts.
3. Renaming of a district.

During the period of concern (1999-2018), there was a major realignment in 2009, when no less than 37 districts in England were abolished (out of 317 today), many of them being merged to create new, larger districts. For example, the districts of Blyth Valley, Wansbeck, Castle Morpeth, Tynedale, Alnwick and Berwick-upon-Tweed were all combined to form a new district called Northumberland.

In addition to adjusting the districts to be accurate to the most recent makeup (even if at the time the data was collected the district was different), there were missing districts and incorrect districts (for example there were a few records where the value in the district was actually the city). I therefore set about rectifying this, as well as setting standard names for the different districts. This was an issue since there are a few districts that are congruent with a city and named for it. Setting a consistent naming scheme for districts such as these was important to ensure that, for example, 'Derby' and 'City of Derby' get correctly joined, as they are the same district.

In addition to all of this, I added a column called 'brexit' which was the number of days after Brexit that the home sale took place (negative for sales that took place before). The reference date (value 0) was set to be the day after the Brexit referendum (June 24, 2016), as this was the day the results were announced. The purpose of this was to see if and how the results of the referendum (as well as the campaigning in the leadup to it), and the subsequent economic volatility, affected the real estate market over time.

After making these adjustments, I then combined the different data files, joined on the district column. This gave me a file for each year that contained all of the relevant data for that year.

C. Model Preparation

Finally, after combining all of the data from the different sources, I needed to prepare it for usage in a model. For this I needed to convert all categorical columns to numeric ones. There were a couple columns that had only 2 values, for example a column called 'old-new' whose values were either 'Y' or 'N' (i.e. yes it is new or no it's not), thus a single binary column was an ideal solution. There were a number of other columns where there were multiple values and no clear relative order between them, for example the columns for districts and regions, thus I converted each of those into n 'dummy' binary columns.

There was, however, one column that I felt did have a relative order to the values. A column called 'property-type' had four possible values:

1. Flat ('F')
2. Terraced ('T')
3. Semi-Detached ('S')
4. Detached ('D')

Intuitively, in a given area, the average detached home would sell at a higher price than a semi-detached home, which would sell higher than a terraced home, which, in turn, would sell higher than a flat. With this in mind, I decided to try two different options:

1. Convert the column into 4 binary columns, similar to what was done for the districts and regions.
2. Convert the column into a discrete column, with numeric values for each of the four options. In this case, replace 'F' with 1, 'T' with 2, 'S' with 3 and 'D' with 4.

The end result of all of the preprocessing was 80 files:

- 20 files corresponding to each of the years (1 per year) containing all of the data for that particular year and having converted the property_type column to a discrete column. All told, each file had 468 columns, with the number of rows varying by the number of homes sold in that particular year.
- 20 files corresponding to each of the years, as above, but instead the property_type column was converted into 4 binary columns. Thus, each file had 471 columns (the number of rows is identical to those described above).
- 20 files corresponding to having pooled all of the data across all years together and divided it up randomly into 20 parts, each with a little over 1 million rows. These files also had the property_type column converted to a discrete column, thus had 468 columns in total.
- 20 files identical to those above, albeit with a different (also random) division of the homes sold across all years into 20 parts. These files too had a little over 1 million rows, and their property_type column was converted to 4 binary columns, giving them 471 columns in total.

IV. Models

After all of the preprocessing was complete, I was finally ready to begin training – and testing - models on the data. For this I chose to try 3 models:

1. Linear Regression – implemented through Stochastic Gradient Descent
2. Random Forest Regression
3. Multi-Layer Perceptron

For each year, before beginning to test the models, I randomly selected 20% of the data from each year and set it aside for testing. I then trained the models on the remaining 80% of the data (including bootstrapping the Gradient Descent model).

For the models that were trained on the data from all of the years together, when it came time to train the models, I randomly selected 4 out of the 20 parts to be test parts. I made sure these were consistent so that, for example, the bootstrapping models weren't trained on the test data. As this data was randomly divided to begin with, this is equivalent to sampling 20% of the data and setting it aside for testing.

The time it took to train each of the models varied, but generally took 2-6 hours per year, with the Random Forest models taking the longest time.

A. *Linear Regression*

Due to the large volume of data, I knew I didn't have the resources to solve the linear regression problem with the closed-form solution. I therefore implemented a custom mini-batch stochastic gradient descent algorithm (partly based on an implementation I found on GeeksforGeeks [4]). My implementation then trains each of the examples 10 times, randomly shuffling the rows in each batch between iterations.

I chose to score the test results based on the mean absolute error (L1), as I felt I was trying to predict the price of the house as close as possible, thus I the errors to reflect an actual price. Since I wanted the model to be trained on the same error it would be scored on, I also chose this to be the scoring function on which I trained the model.

I also fine-tuned the hyper-parameter learning rate with values that were (negative) powers of 10, until I settled on the value that was most successful at training. The value for this that I found provided the lowest error rate was 0.0000000001. This turned out to be the optimal value (power of 10) for all the years as well as for the data containing all of the years.

Additionally, I performed bootstrap for regression on the model, to give me a confidence interval for the values of theta, the learned weight vector. This was done by sampling from the training data, where the sample contained the exact same number of rows but with each row being individually and randomly selected to enable rows to be sampled more than once, while others perhaps not at all. For each such sample, I then trained a model identical to the one I would later use for training and compared each of the values of theta with the values in the other models that shared an index. In other words, I was comparing the weights of each feature with the weights that feature had in each of the bootstrap samples. I chose the threshold of 0.9 (for a confidence interval of 90%) and stored the values – a high and a low – from the bootstrap samples to be used to compare the values of the trained model.

B. *Random Forest Regression*

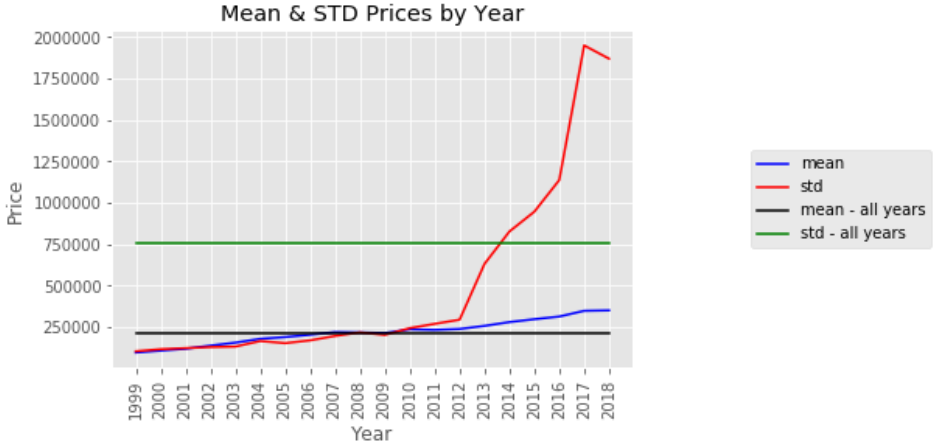
For this model, I made use of scikit-learn's RandomForestRegressor model [5], though here too I needed to make slight adjustments to the way in which I invoked the training (fit) and predict methods to train and predict in batches, due to the volume of data. To maintain consistency, here too I chose to train and score the model with the mean absolute error.

C. *Multi-Layer Perceptron*

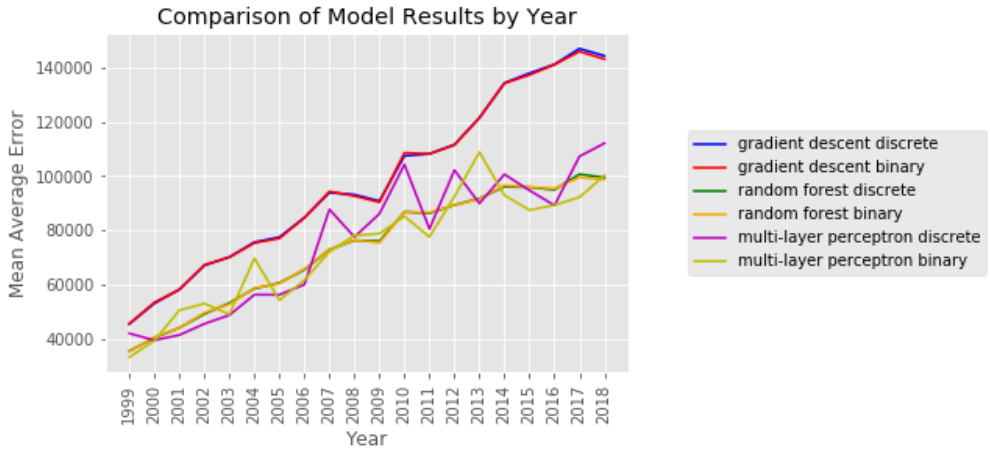
For this model I used scikit-learn's MLPRegressor model [5]. Surprisingly, the model was able to handle an entire year's worth of data in one single call to fit. Here too I used mean absolute error to train and score the model.

V. Results

Throughout this section, there are a number of graphs highlighting and comparing the results of the different models (graphs 2.1-2.18). These include graphs pertaining to the test error results of the models (graphs 2.2-2.14) as well as the



Graph 2.1 – Mean and Standard Deviation of prices by year and for all years combined.

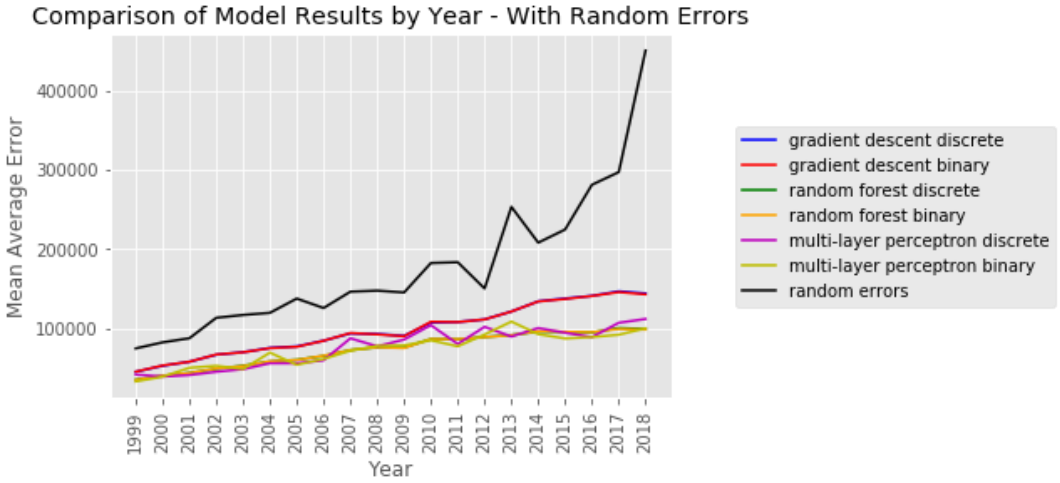


Graph 2.2 – Comparison of results of all 6 models by year.

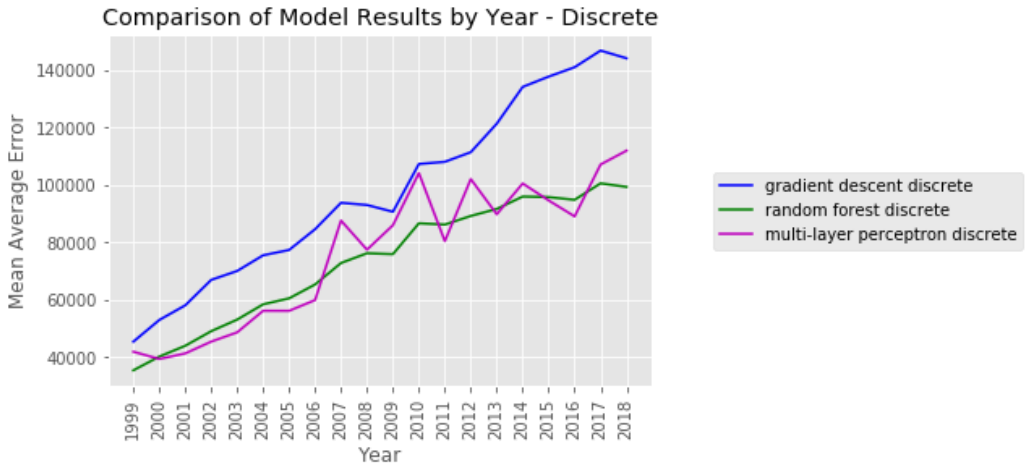
most important features for each year (for the Stochastic Gradient Descent and Random Forest Models).

The results for the different models can be found in Table 1.1 in the Additional Material section. Overall, the errors increased by year across all the models. This is in line with the increases in price mean and standard deviation (see graph 2.1).

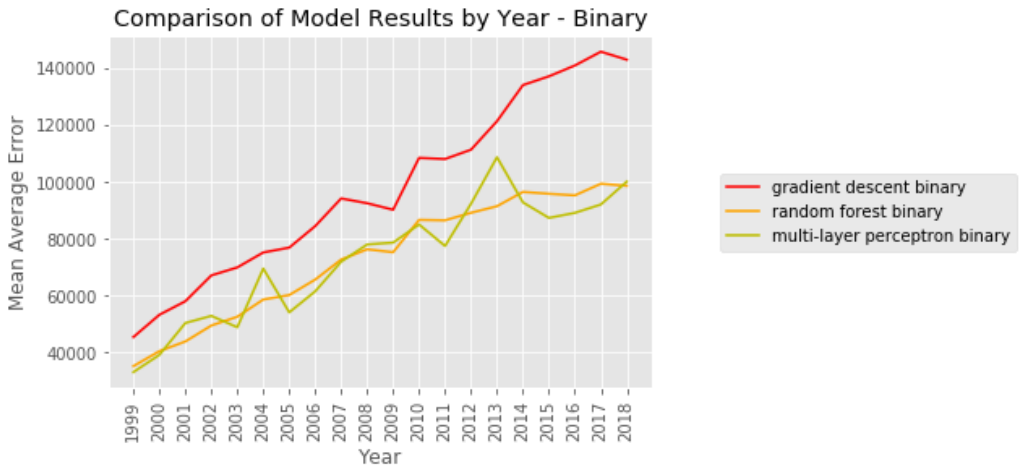
Overall, the results are not great, with both the Random Forest Regression models and the Multi-Layer Perceptron models significantly outperforming the Stochastic Gradient Descent models. While there isn't a clear 'winner' between the Random Forest and Perceptron models, there was significantly more variation in the Perceptron models' results. Additionally, it appears the binary models performed slightly better than their discrete counterparts. This is not unexpected, since if the true ratio between the prices were indeed 4:3:2:1 (detached to semi-detached to terraced to flat), the optimal weights learned for each of the binary features would reflect that as well. In other words, one would expect the binary models to be as good as, if not better than, the discrete ones.



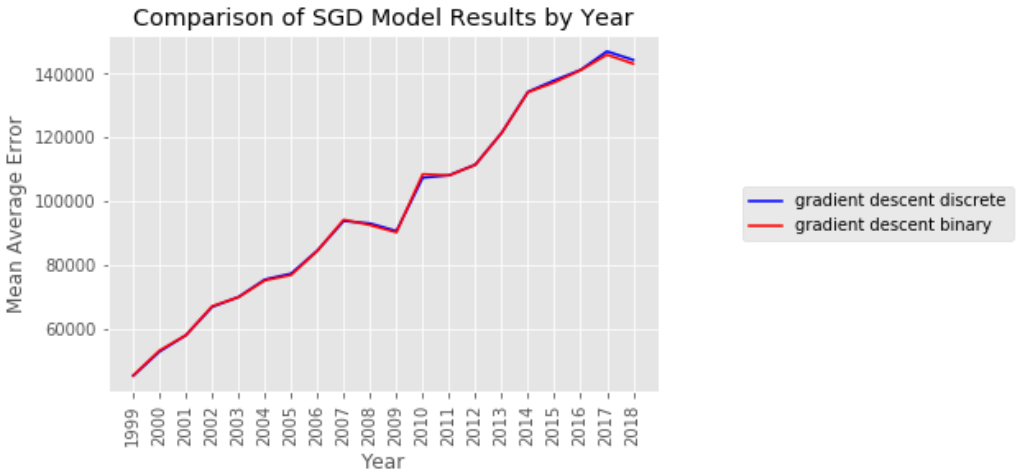
Graph 2.3 – Comparison of results of all 6 models by year and random errors.



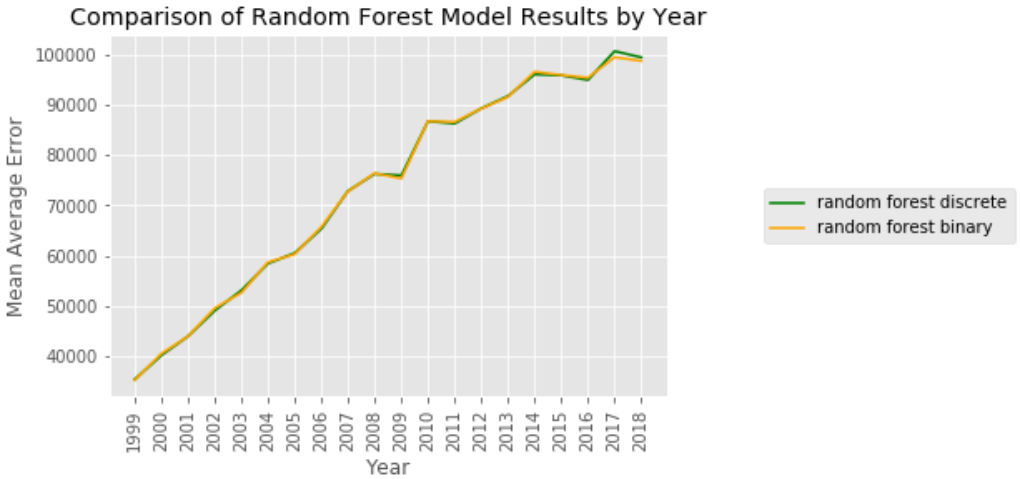
Graph 2.4 – Comparison of results of 3 Discrete (property type) models by year.



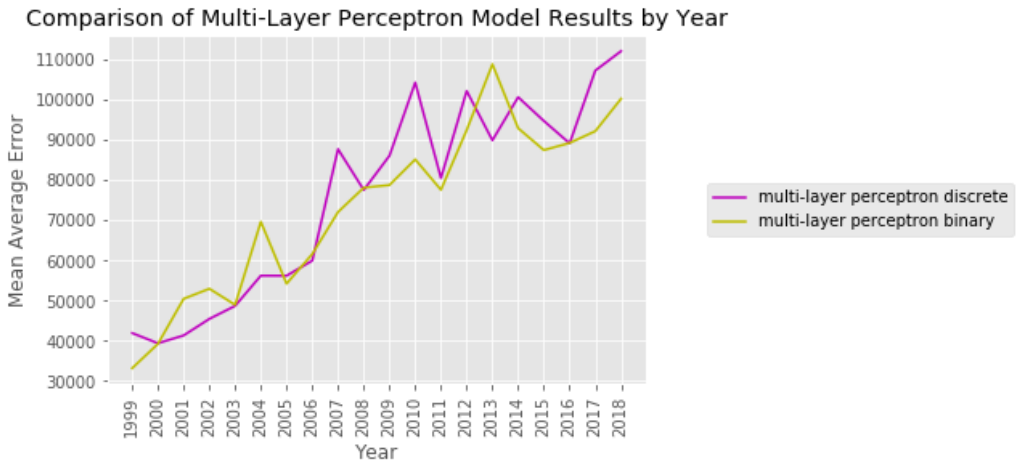
Graph 2.5 – Comparison of results of 3 Binary (property type) models by year.



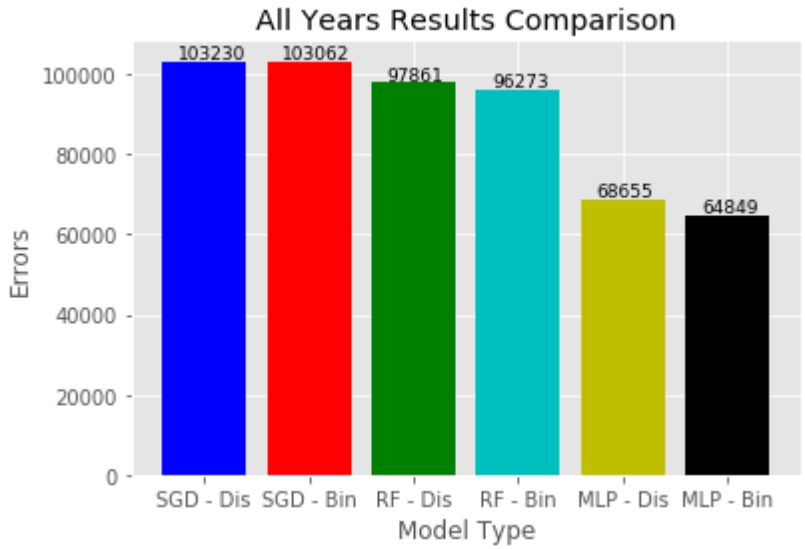
Graph 2.6 – Comparison of results of 2 Stochastic Gradient Descent (SGD) models by year.



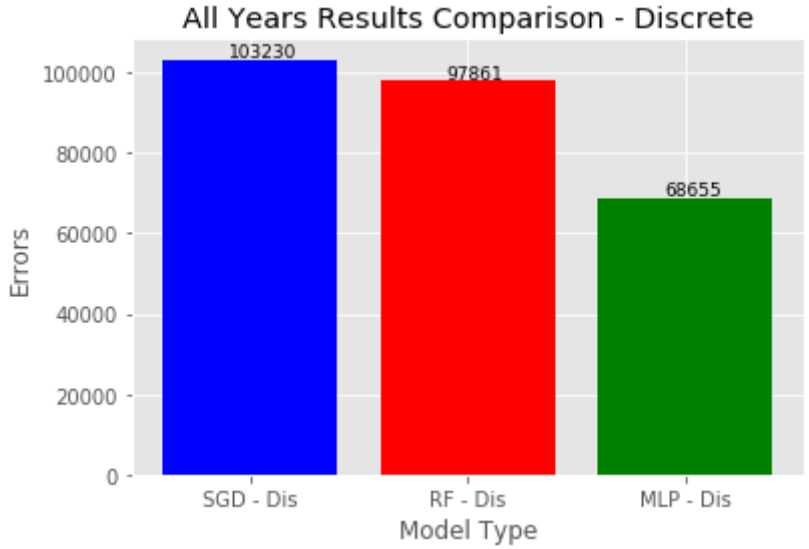
Graph 2.7 – Comparison of results of 2 Random Forest Regression models by year.



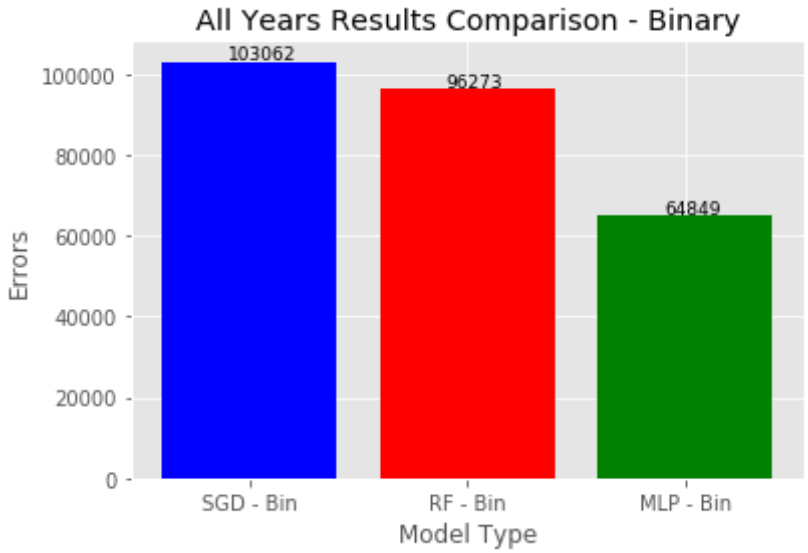
Graph 2.8 – Comparison of results of 2 Multi-Layer Perceptron models by year.



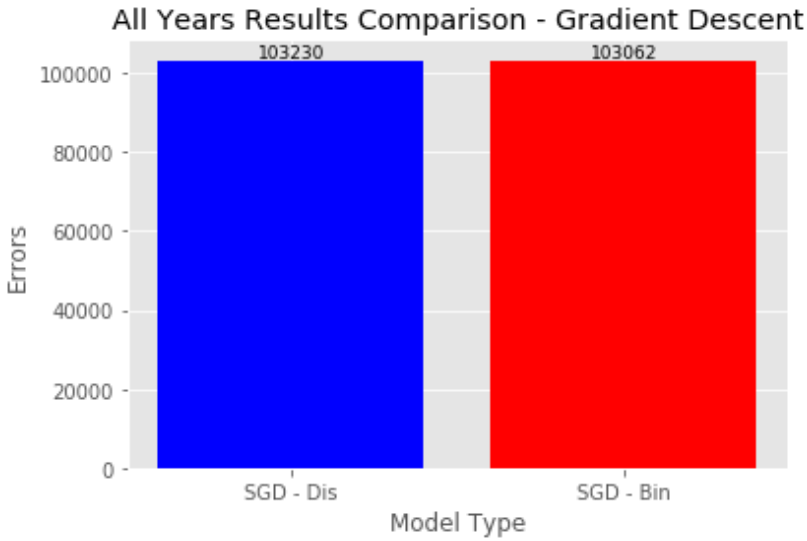
Graph 2.9 – Comparison of results of all 6 models for combined years.



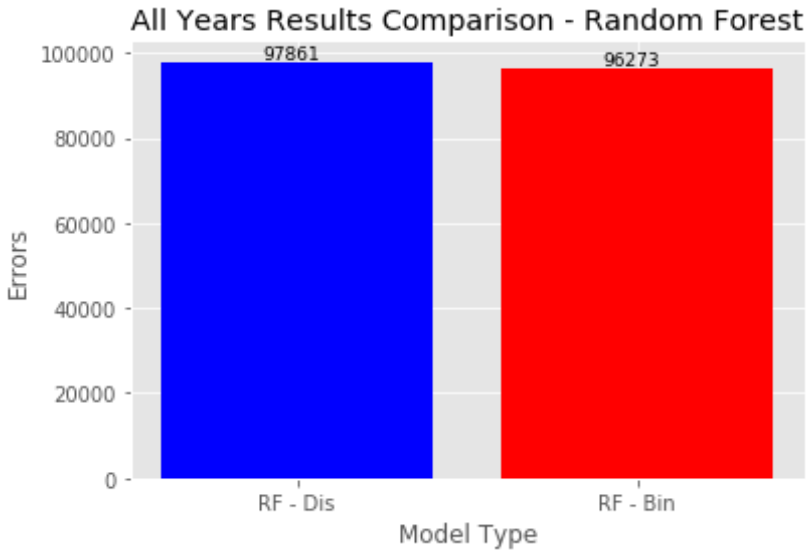
Graph 2.10 – Comparison of results of all 3 Discrete (property type) models for combined years.



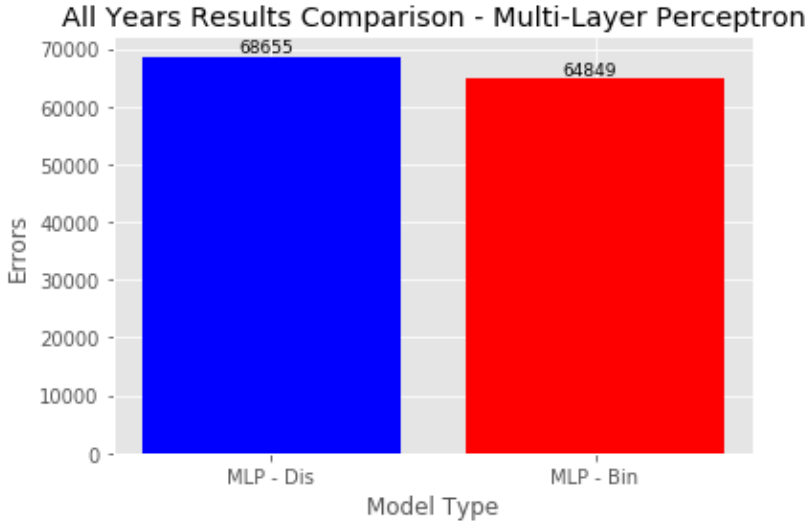
Graph 2.11 – Comparison of results of all 3 Binary (property type) models for combined years.



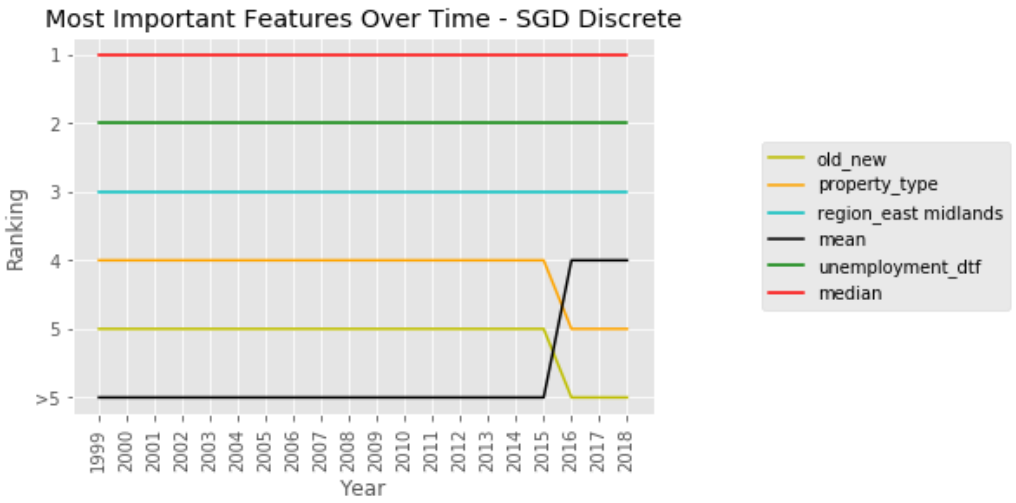
Graph 2.12 – Comparison of results of 2 Stochastic Gradient Descent (SGD) models for combined years.



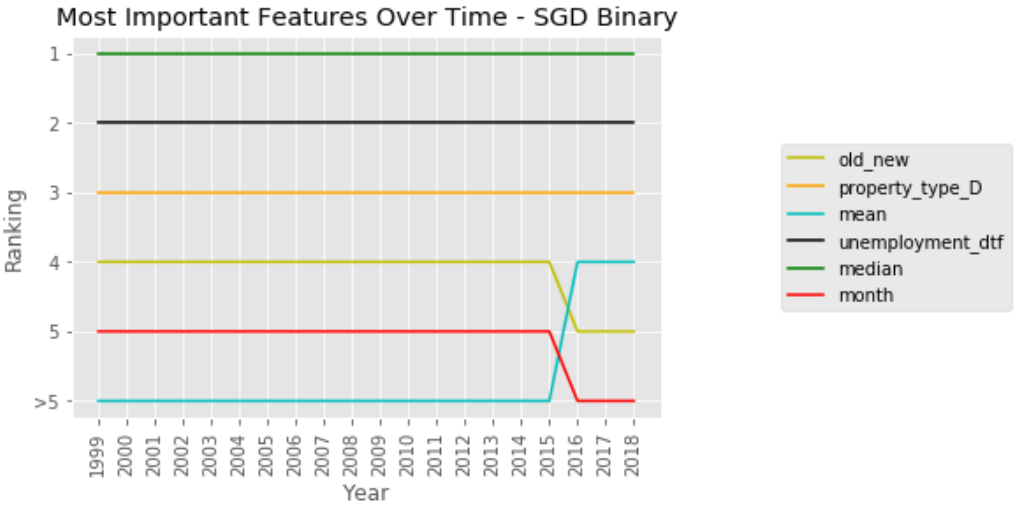
Graph 2.13 – Comparison of results of 2 Random Forest Regression models for combined years.



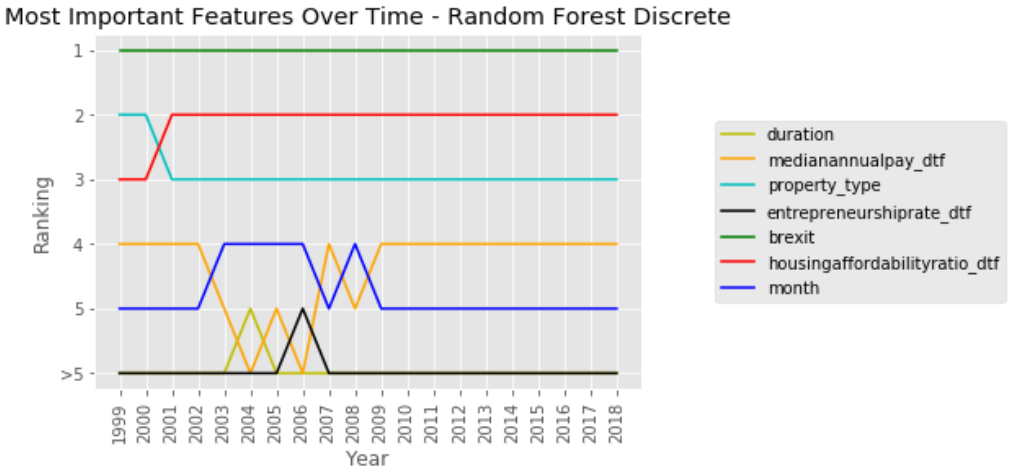
Graph 2.14 – Comparison of results of 2 Multi-Layer Perceptron models for combined years.



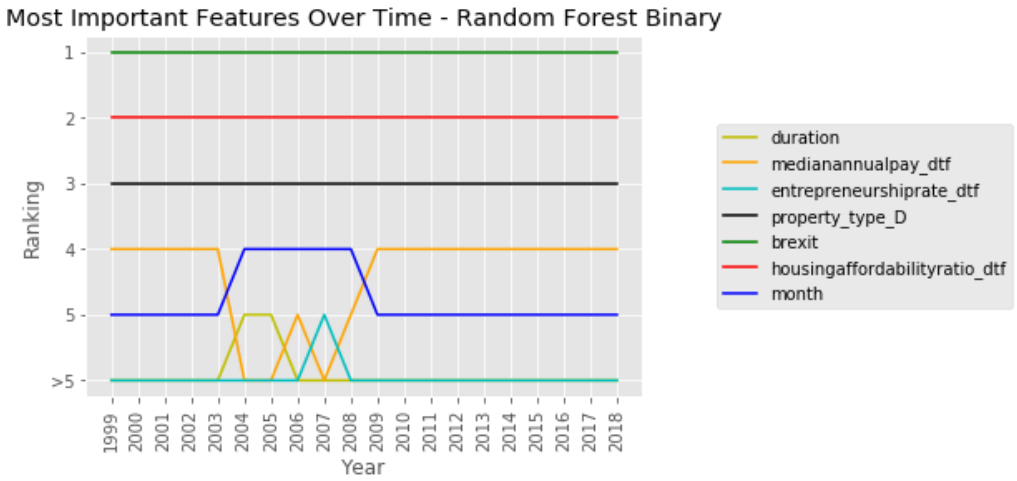
Graph 2.15 – Most important features for Stochastic Gradient Descent Discrete model (by year).



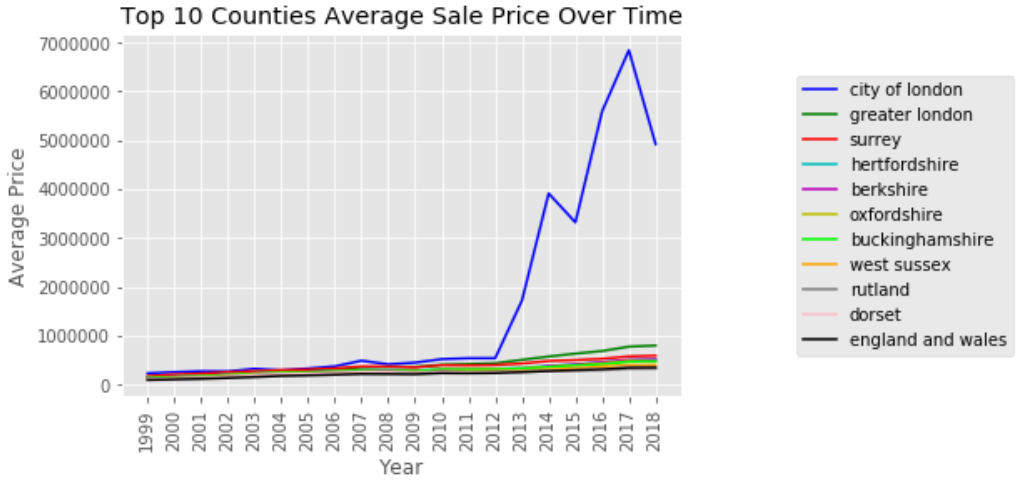
Graph 2.16 – Most important features for Stochastic Gradient Descent Binary model (by year).



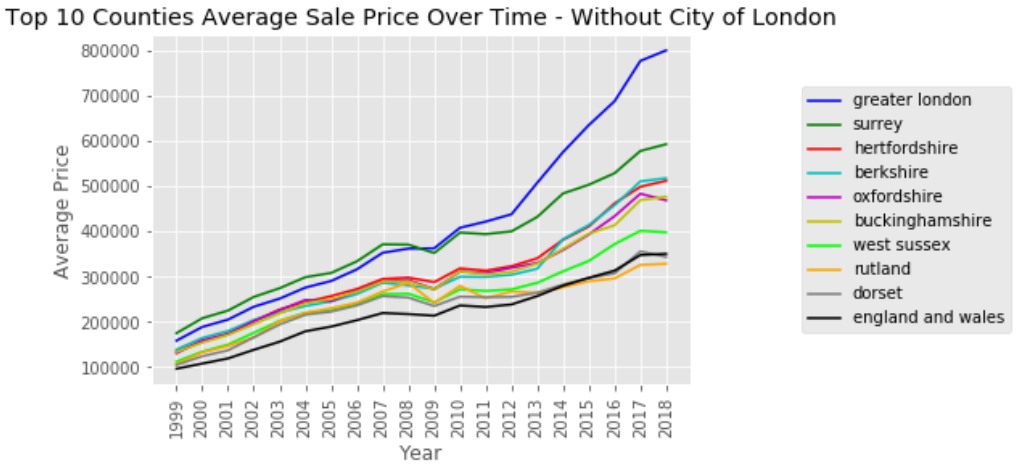
Graph 2.17 – Most important features for Random Forest Regression Discrete model (by year).



Graph 2.18 – Most important features for Random Forest Regression Binary model (by year).

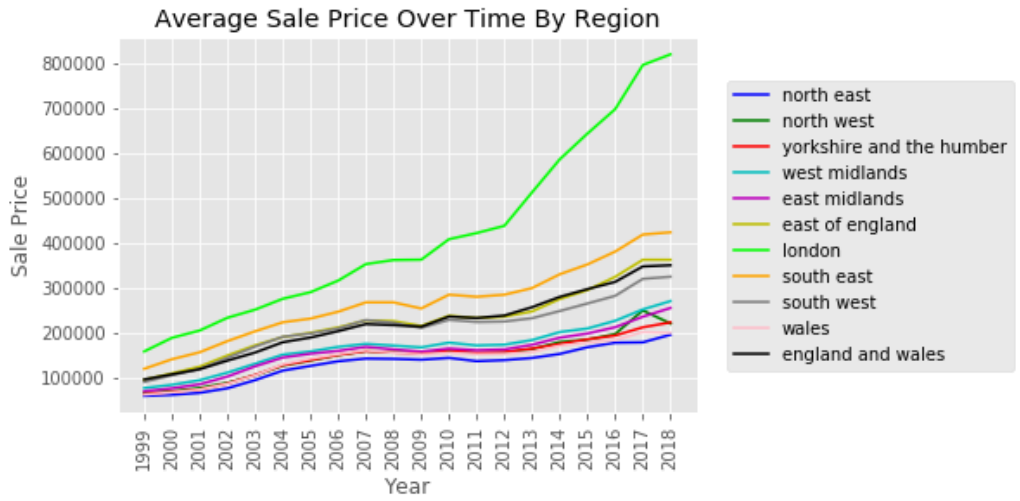


Graph 1.7 – 10 counties with highest average sale price. Notice how City of London has a tremendous spike in 2012-2013.

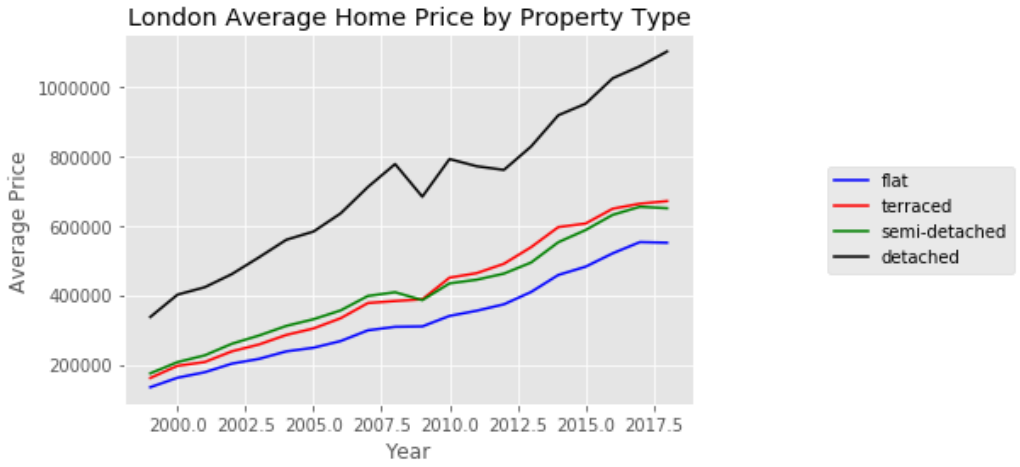


Graph 1.8 – 10 counties with highest average sale price, not including City of London. Notice how Greater London has a spike starting in 2012-2013, but not as pronounced as City of London (graph 1.7).

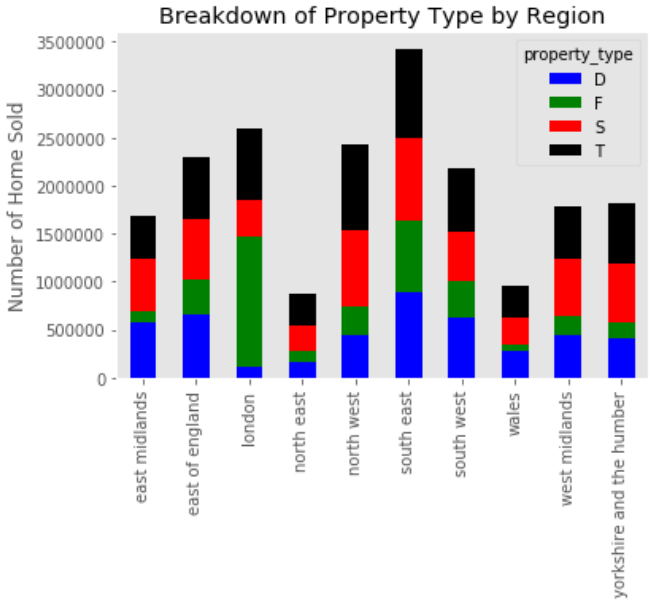
As can be seen from the graphs, the average error consistently increased over time across all of the models, with the average errors across all models being below 46,000 in 1999 compared to 2018 where all of the model's errors were higher than 98,000. These increases in errors seem to follow a similar trajectory to that of the standard deviation, which increased from 103,631.7 in 1999 to 1,868,498 in 2018. An interesting thing to note, and a possible factor for this increase, could be the increase in average home price for the City of London (not to be confused with London – the City of London is an independent city in central London on the site of the old Roman city of Londinium). We see both in graphs 1.7 and 1.8, containing the top 10 counties by average price through the years and in graph 2.1, containing the means and standard deviations of price throughout the years, the significant increase beginning in 2012-2013. This, despite the City of London only accounting for 0.02-0.05% of sales each year.



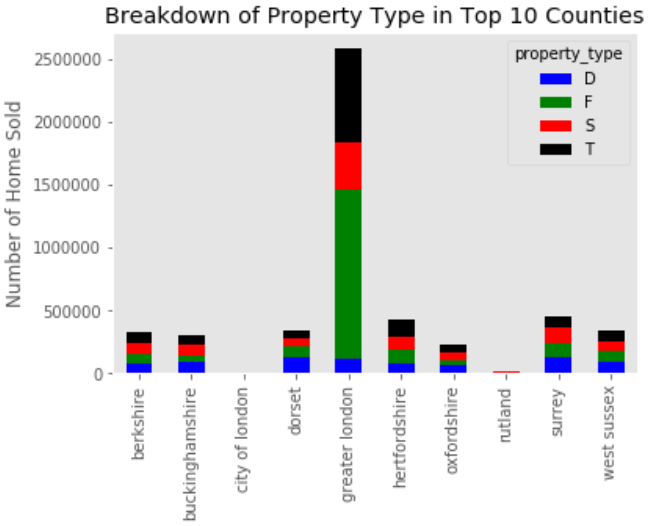
Graph 1.3 – Comparison of average sale prices by year per region. Notice how the gap between London and the rest of England and Wales grows dramatically starting in 2012-2013.



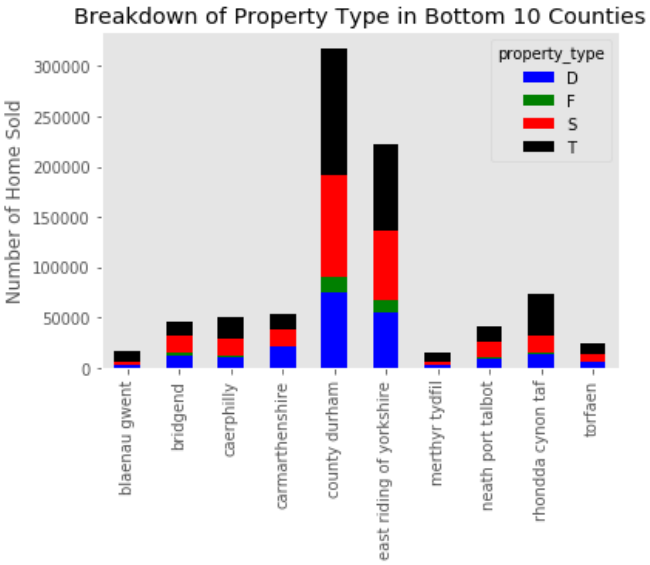
Graph 1.16 – Comparison of average home price in London by property type.



Graph 1.13 – Stacked bar plot comparing the breakdown by type of house by region (height is total number of sales).



Graph 1.14 – Stacked bar plot comparing the breakdown by type of house for the counties with the highest average sale price (height is total number of sales).

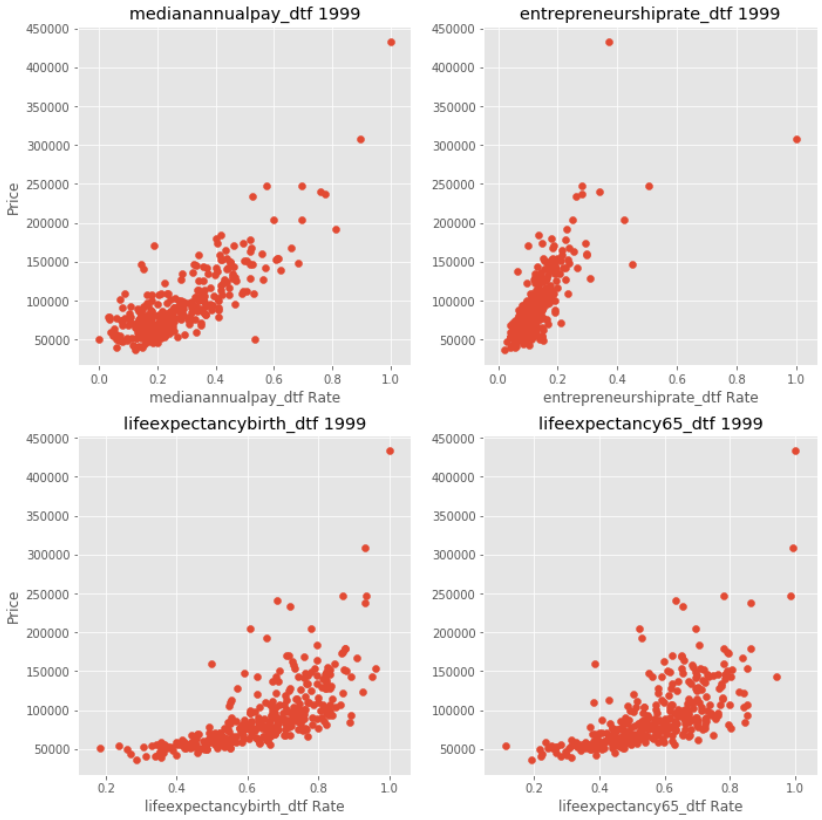


Graph 1.15 – Stacked bar plot comparing the breakdown by type of house for the counties with the lowest average sale price (height is total number of sales).

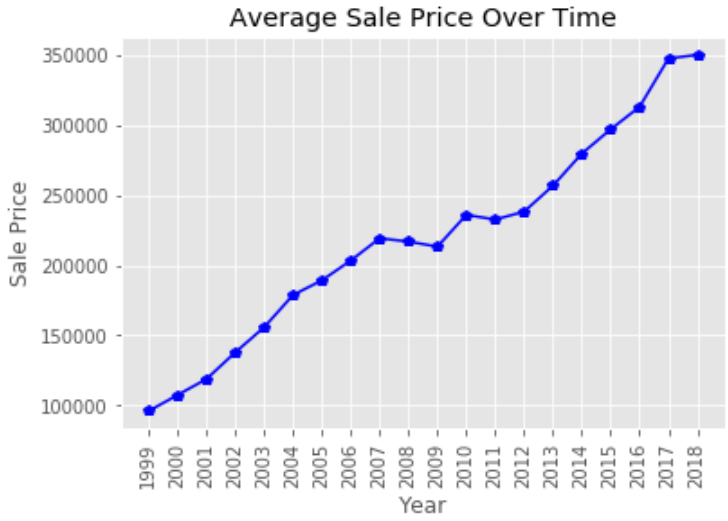
Paradoxically, while it is intuitive to think that the pricing for different home types would be – in order – detached, semi-detached, terraced, flat, the region with the highest number of flats sold is London (see graph 1.13), the region with the highest average price per home (as per graph 1.3). As such, it is possible that the initially expected discrepancies between the average price for a flat and other property types are significantly smaller. Additionally, many flats tend to be in cities, where average prices are higher than rural towns and villages, where homes tend to be detached or semi-detached. However, this in no way disputes the notion that the average price for a flat would be lower than that of a terraced, semi-detached or detached home, only that this needs to be examined at a more local level. In fact, for every single year (1999-2018), the average price for a flat in London was lower that the price for a terraced home, which was similar to the price for a semi-detached home and lower than the price for a detached home (see graph 1.16). Additionally, by looking at graph 1.15 we see that in the bottom 10 counties by average home price, the share of flats compared to other types of homes is very low, whereas in Greater London, one of the counties with the highest average price (see graphs 1.7 and 1.8), the share of flats is very high. (see graph 1.14).

Additionally, upon looking at pairwise correlations between price and socio-economic factors (graphs 1.17.1-1.17.80), some of the features that appear to have a fairly significant correlation are median annual pay, entrepreneurship rate and life expectancy at birth and at 65.

Other interesting insights gleaned from this data include a steep rise in home prices in the City of London compared to the rest of England and Wales starting in 2012 (graph 1.7), at the tail end of the financial crisis. Prior to and during the recession, the City of London was among the counties with the highest average home price, but not much different than other wealthy counties such as Greater London and Surrey. Afterwards, the average price shot up significantly. Although less pronounced, this can also be seen at the region level (see graph 1.3).



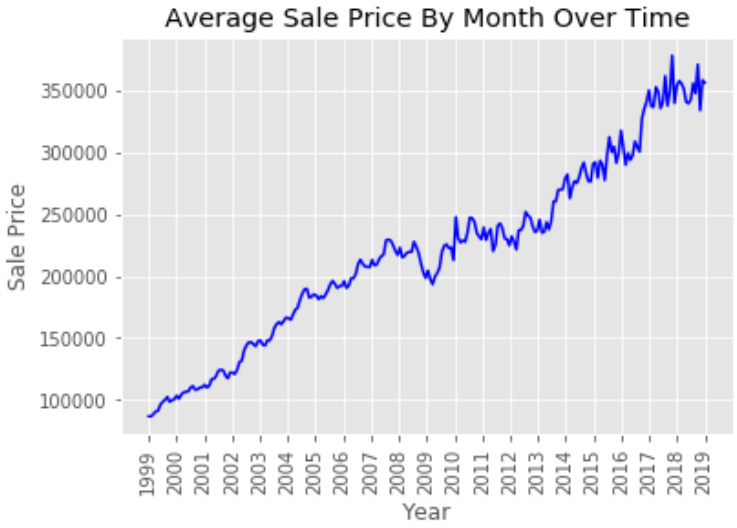
Graph 1.17.1-1.17.80 – Features that appear to have clear correlation with home price. Only 1999 shown here, but trend is similar for each of these features in each of the years (the rest can be viewed in the notebook).



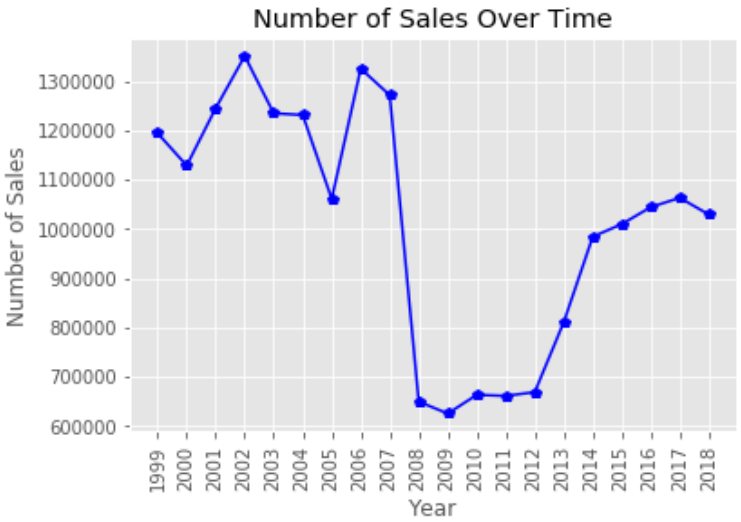
Graph 1.1 – Average sale price trend (yearly average).

We can clearly see the devastating effect of the 2007-2009 recession on the average price (graphs 1.1-1.3) and especially on the number of homes sold (graphs 1.4-1.6). Prior to the recession the average home price was on a clear and unbroken upward trajectory (starting at least in 1999) and dipped from 2007-2009, essentially not changing much until 2012 (a 5 year slump). This is even more pronounced when looking at the number of sales, which dropped significantly during the recession and even by 2018 hadn't returned to its pre-recession numbers. The lone exception to this can be seen in graph 1.5 in 2016 surrounding the Brexit referendum when the number of sales shot up to the highest numbers recorded (in the years 1999-2018) before immediately returning to its previous levels.

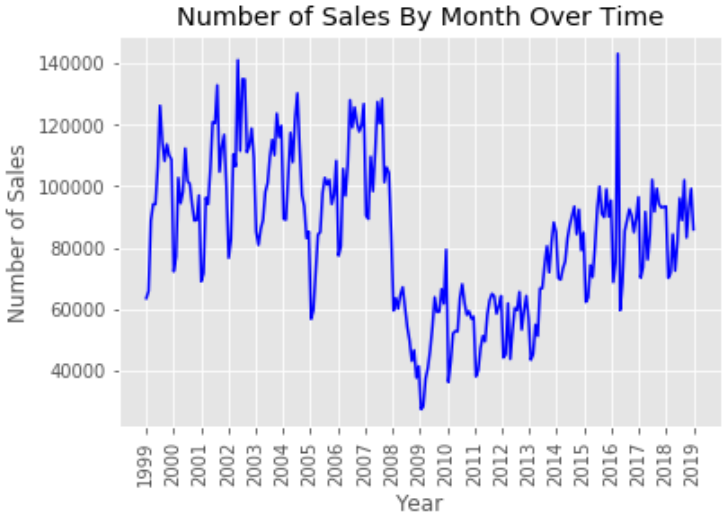
Despite the relatively poor results, the data processing performed is in fact valuable and did contribute positively to the overall results. When comparing the model results to a completely random selection of homes and random prices from the data – as if to 'correspond' to a prediction for those homes – the models outperformed the random samples by significant margins (see Graph 2.2).



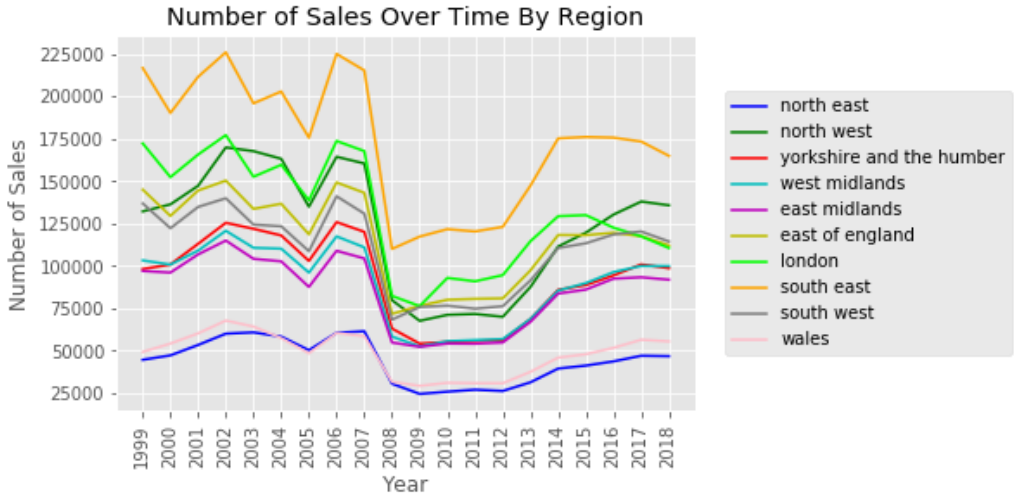
Graph 1.2 – Average sale price trend (monthly average).



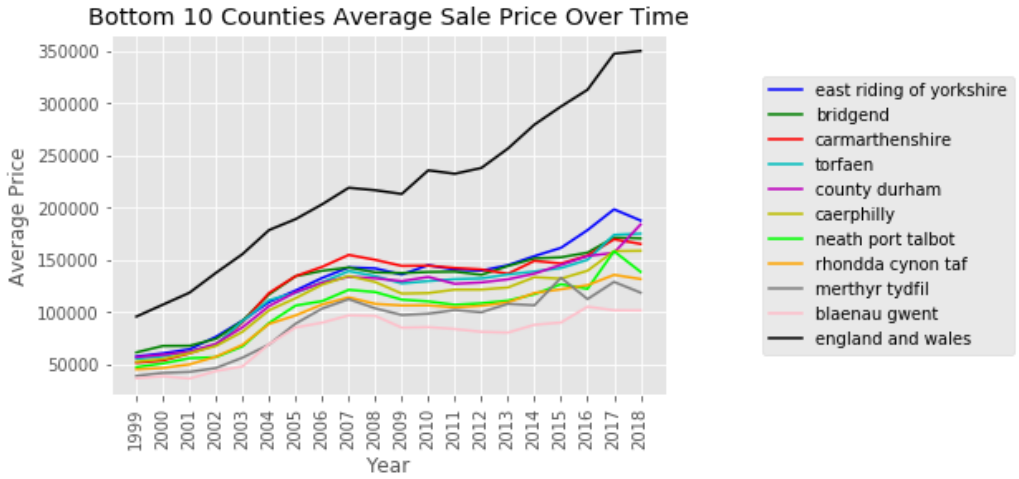
Graph 1.4 – Number of sales trend (yearly).



Graph 1.5 – Number of sales trend (monthly).

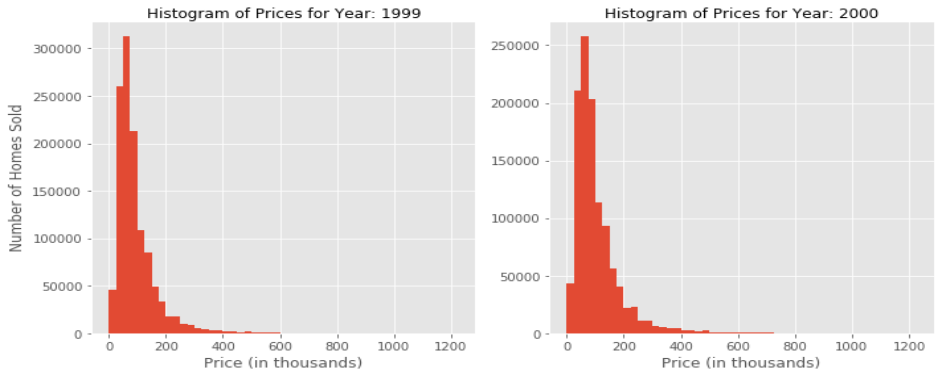


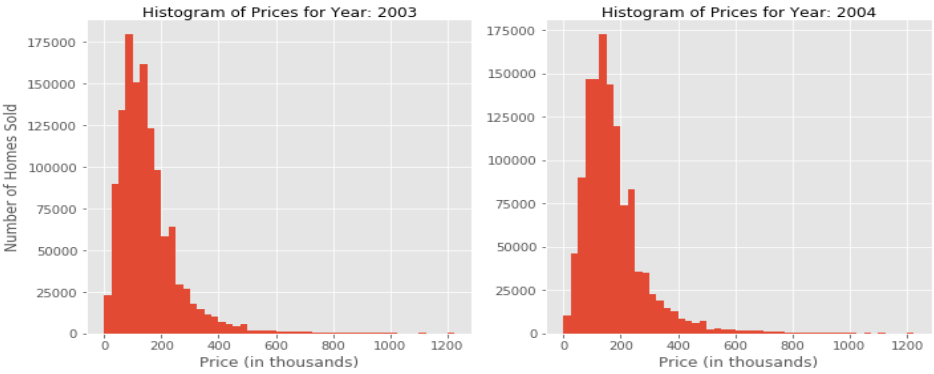
Graph 1.6 – Number of sales trend by region (yearly).



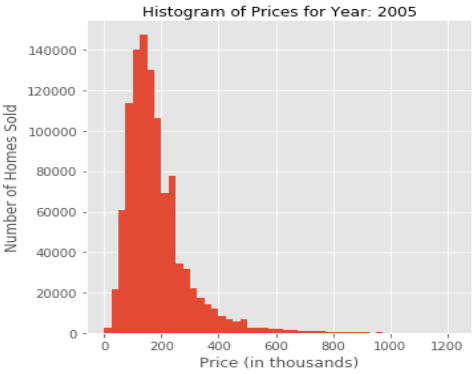
Graph 1.9 – 10 counties with lowest average sale price, compared to average of England and Wales.

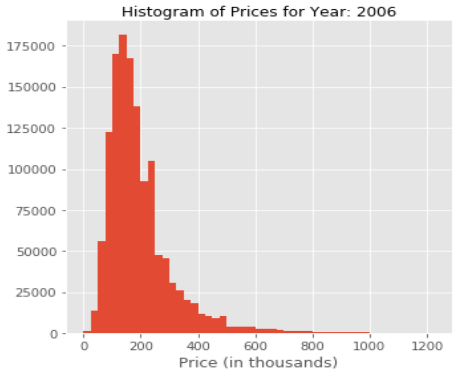
Another thing that can be seen is what appears to be a growing inequality between wealthier, more prosperous regions and those less well-off. In graph 1.9 we see that while the average home priced increased between 1999 and 2018 in the counties with the lowest average price per home, the rate at which the average price increased was much smaller than the average across England and Wales. This can also be seen in graphs 1.10.1-1.10.20 where many homes significantly increased in value, while many did not. This can be seen in the 'fattening' of the histograms over time. Were this to have been spread more equally, we would have seen the main mass of the histogram move to the right, as we do in fact see, but without the significant 'fattening' of the histogram.

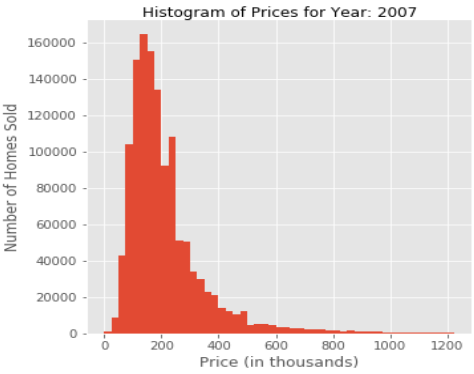


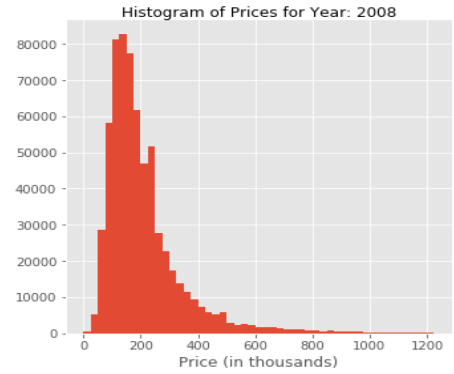

Histogram of Prices for Year: 2003
Number of Homes Sold
Price (in thousands)

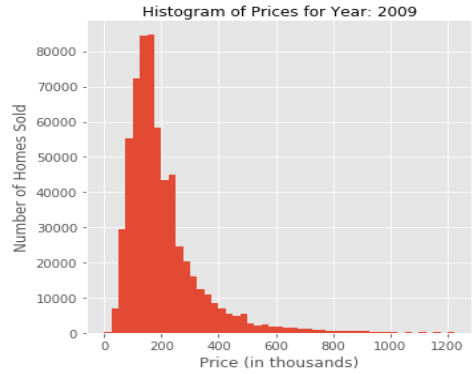
Histogram of Prices for Year: 2004
Price (in thousands)

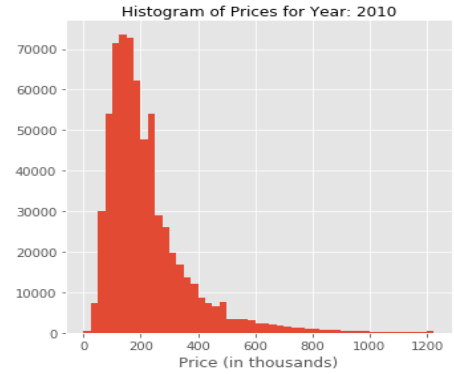

Histogram of Prices for Year: 2005
Number of Homes Sold
Price (in thousands)

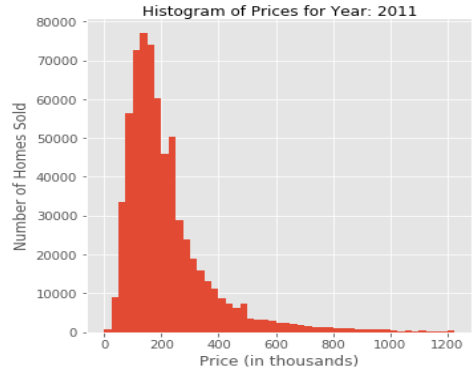

Histogram of Prices for Year: 2006
Price (in thousands)

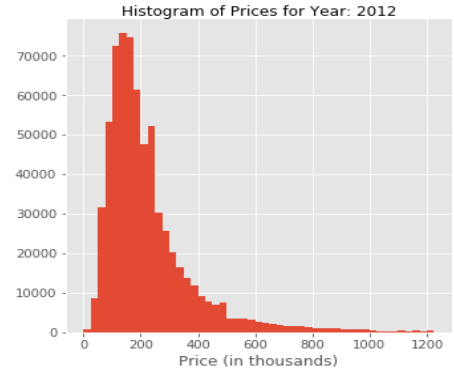

Histogram of Prices for Year: 2007
Number of Homes Sold
Price (in thousands)

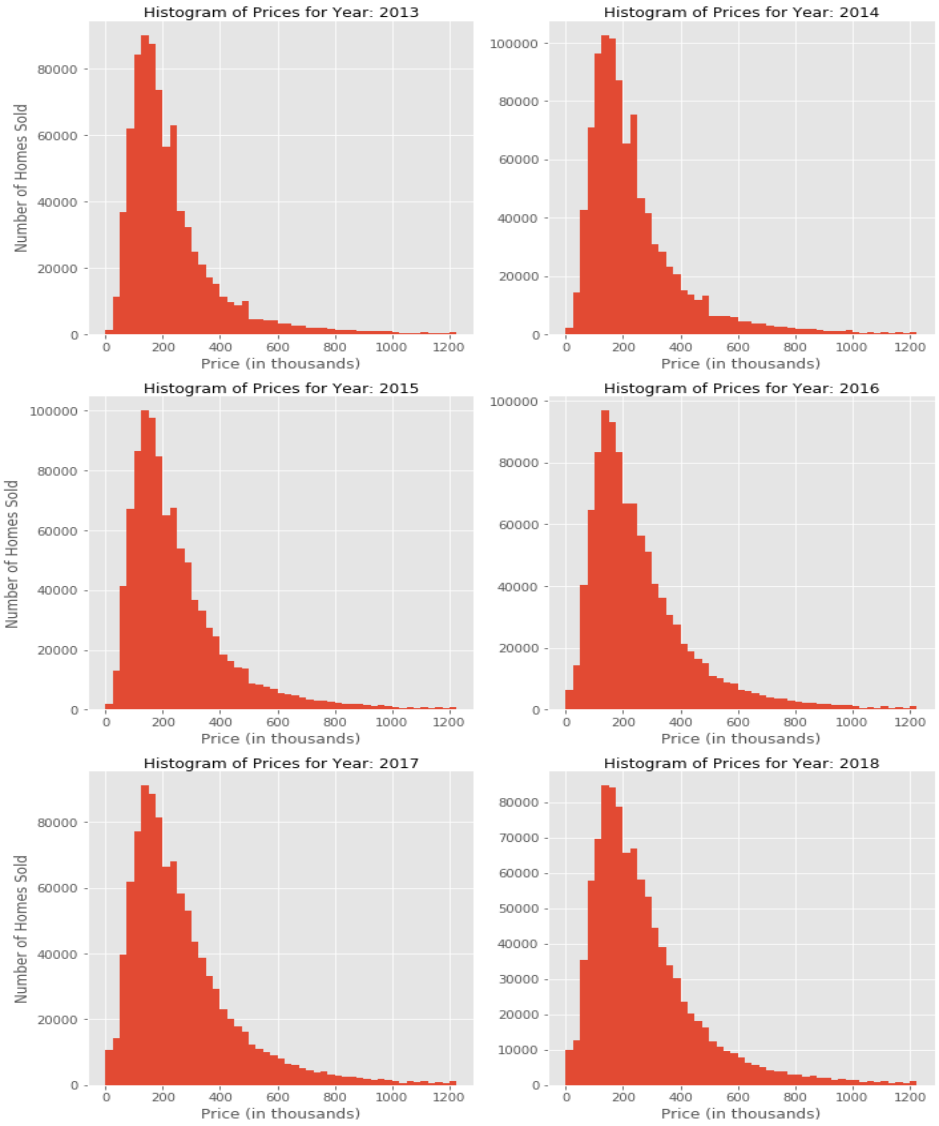

Histogram of Prices for Year: 2008
Price (in thousands)


Histogram of Prices for Year: 2009
Number of Homes Sold
Price (in thousands)


Histogram of Prices for Year: 2010
Price (in thousands)


Histogram of Prices for Year: 2011
Number of Homes Sold
Price (in thousands)


Histogram of Prices for Year: 2012
Price (in thousands)

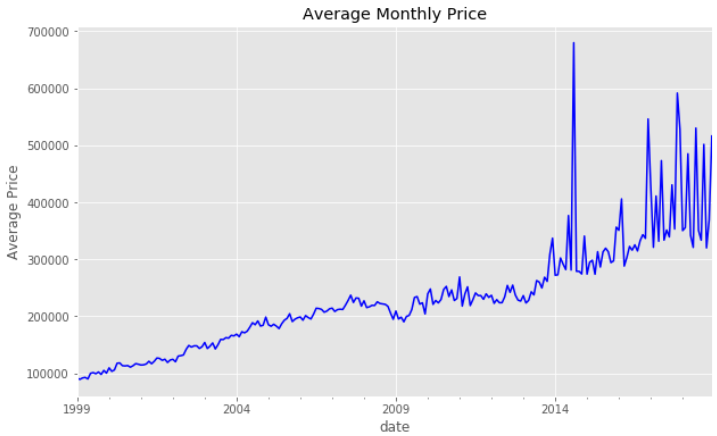


Graphs 1.10.1-1.10.20 – Histogram of prices by year. Notice that the histogram gets 'fatter' over time.

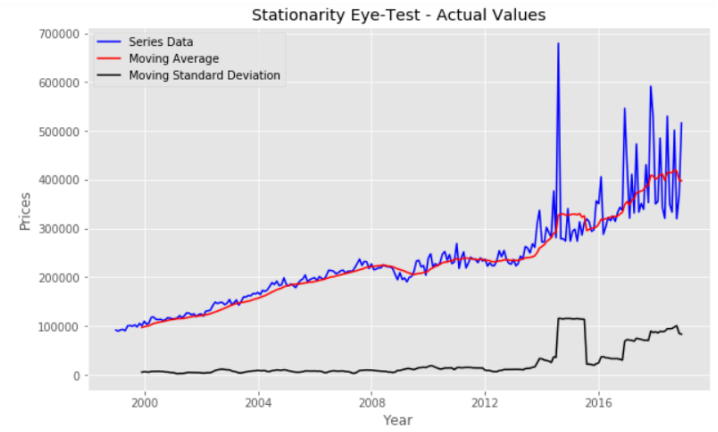
VI. Time-Series Analysis

We begin by getting familiar with average monthly price over time (graph 3.1). As we can see, there appears to be a generally upward trend in the average price over time. Additionally, the variations between months grow significantly larger starting in 2014.

One of the first things we would like to know is whether our data is stationary, a prerequisite for time series models such as ARIMA. In graph 3.2 we see the monthly average, along with the rolling average and rolling standard deviation (calculated at 12 months). From both the visual data as well as the Dickie-Fuller test results (the p-value is very large and the test statistic is larger than each of the critical values), we can

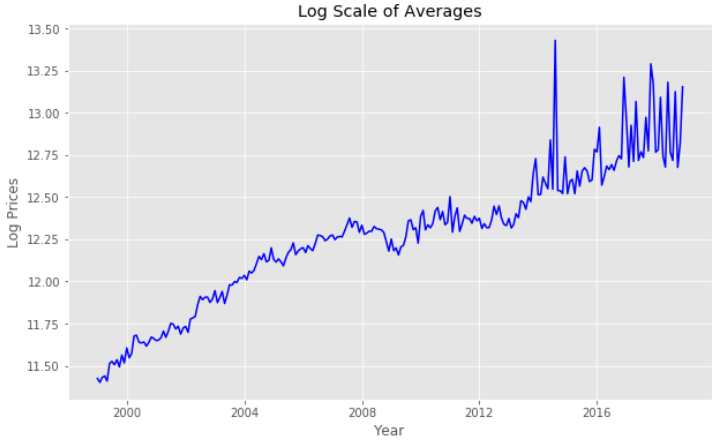


Graph 3.1 – Average sale price over time (monthly).




```
Dickie-Fuller Test Results:
Test Statistic             0.201899
p-Value                    0.972368
Num Lags                  11.000000
Num Observations         228.000000
Critical Values (1%)      -3.459361
Critical Values (5%)      -2.874302
Critical Values (10%)     -2.573571
dtype: float64
```

Graph 3.2 – Average sale price over time (monthly) alongside rolling average and rolling standard deviation and Dickie-Fuller test results showing the data is not stationary.

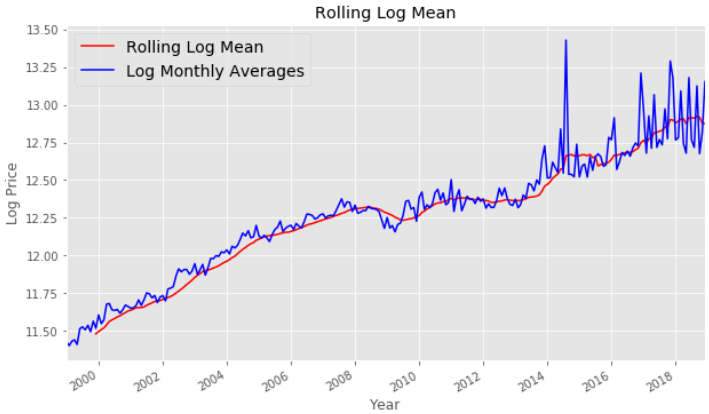


Graph 3.3 – Average log-scale sale price over time (monthly). Note the trend is about same as in graph 3.1.

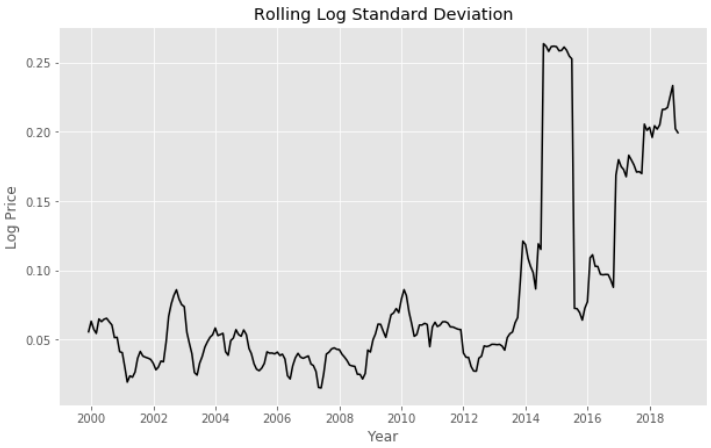
conclude that the data is not stationary (i.e. accept the null hypothesis that the data is not stationary).

In order to enable the usage of a time series model, we need to first transform the data. We start by taking the log of the prices and plotting the monthly average (graph 3.3) along with the rolling average (graph 3.4) and rolling standard deviation (graph 3.5). We see that although the values on the y axis have clearly shrunk, the overall trend remains about the same. Note that due to the larger variations in monthly average prices beginning in 2014 we see that the standard deviation increases as well.

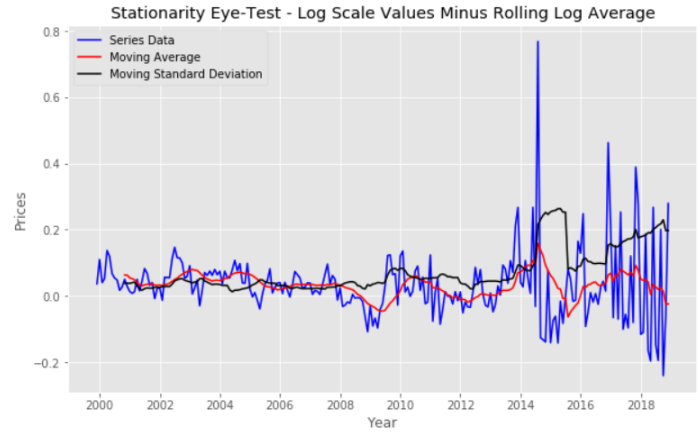
Next, we take the log scale data and subtract the rolling log average from it. In graph 3.6, both visually as well as from the Dickie-Fuller test, we can conclude that the data is now, finally, stationary. However, the rolling mean does have a fair amount of variation, especially in the later years.



Graph 3.4 – Average log-scale sale price over time (monthly) alongside rolling average.

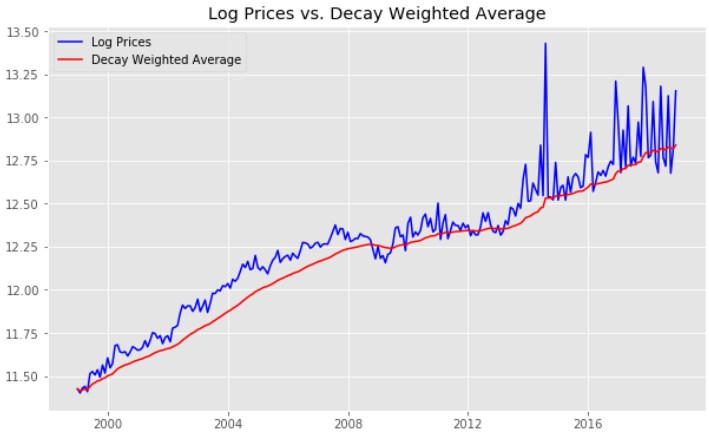


Graph 3.5 – Rolling log-scale standard deviation.



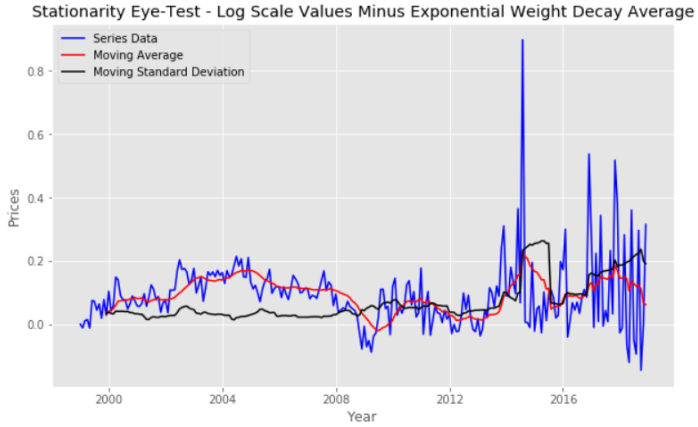

```
Dickie-Fuller Test Results:
Test Statistic            -5.514298
p-Value                    0.000002
Num Lags                   7.000000
Num Observations         221.000000
Critical Values (1%)      -3.460291
Critical Values (5%)      -2.874709
Critical Values (10%)     -2.573789
dtype: float64
```

Graph 3.6 – Average log-scale sale price over time (monthly) minus the rolling log average alongside its rolling average and rolling standard deviation and Dickie-Fuller stationarity test results showing the data is stationary.



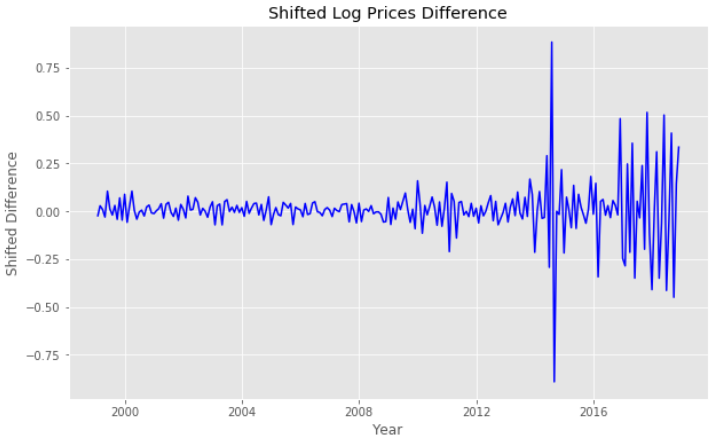
Graph 3.7 – Average log-scale prices alongside exponential weight decay average.

Another transformation we can do is to apply an exponential weighted decay function (to the log scale prices). Based on graphs 3.7 and 3.8, we see similar results, and most importantly, we see that after this the data appears to be stationary (both visually and by the Dickie-Fuller test), but again, the rolling mean has a fair amount of variation, especially in the later years.

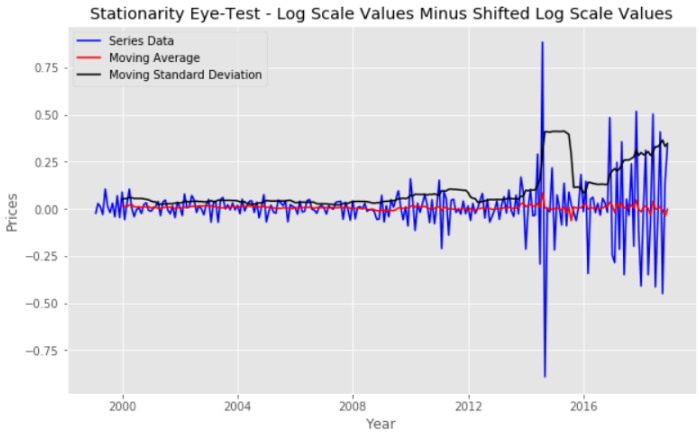



```
Dickie-Fuller Test Results:
Test Statistic            -4.080340
p-Value                    0.001044
Num Lags                   5.000000
Num Observations         234.000000
Critical Values (1%)      -3.458608
Critical Values (5%)      -2.873972
Critical Values (10%)     -2.573396
dtype: float64
```

Graph 3.8 – Average log-scale sale price over time (monthly) minus the exponential weight decay average alongside its rolling average and rolling standard deviation and Dickie-Fuller stationarity test results showing the data is stationary.

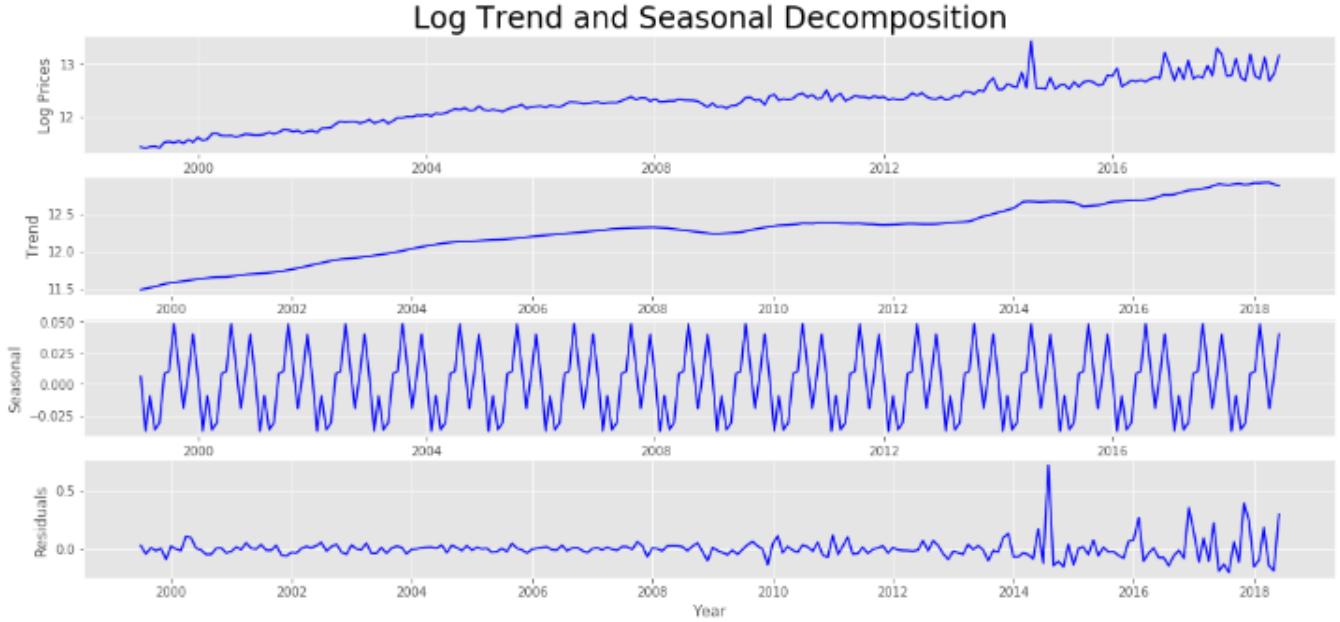


Graph 3.9 – Average log-scale sale price over time (monthly) minus itself shifted (i.e. the value for February 2005 for example is February 2005 minus January 2005).

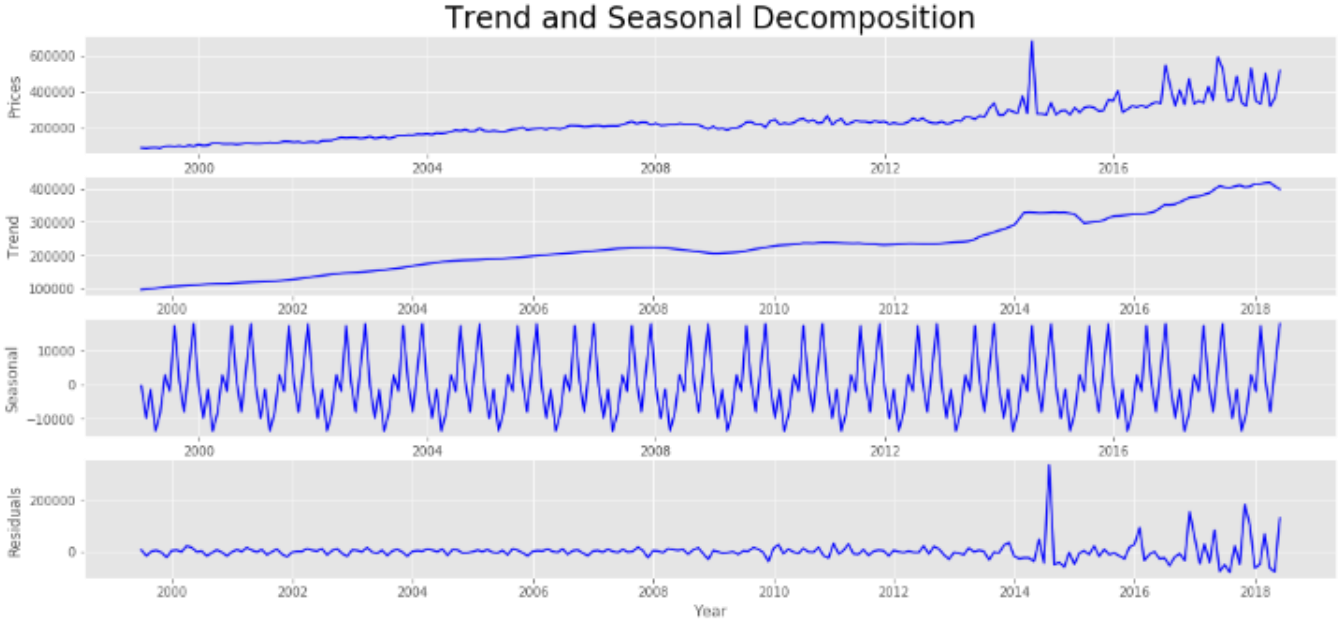



```
Dickie-Fuller Test Results:
Test Statistic          -7.467923e+00
p-Value                  5.145512e-11
Num Lags                 1.000000e+01
Num Observations         2.280000e+02
Critical Values (1%)    -3.459361e+00
Critical Values (5%)    -2.874302e+00
Critical Values (10%)   -2.573571e+00
dtype: float64
```

Graph 3.10 – Average log-scale sale price over time (monthly) minus itself shifted alongside its rolling average and rolling standard deviation and Dickie-Fuller stationarity test results showing the data is stationary.



Graphs 3.11.1-3.11.4 – Decomposition of log scale values into actual values, overall trend, seasonal effects and residuals.



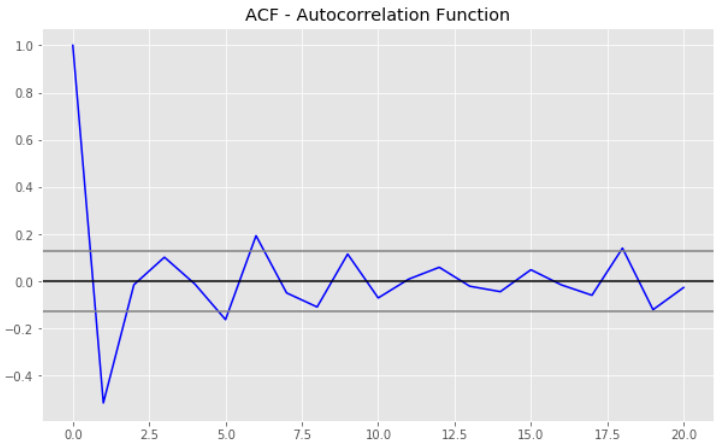
Graphs 3.12.1-3.12.4 – Decomposition of actual values into actual values, overall trend, seasonal effects and residuals. Notice how the graphs here are nearly identical (apart from the values on the y axis) to the log scale graphs (3.11.1-3.11.4).

Yet another transformation we can apply is to shift all of the monthly average values 1 month ahead and then subtract the shifted prices from the actual prices (graph 3.9). For example, we subtract the average price for January 2006 from the average price for February 2006, etc. Again, as per graph 3.10, we conclude that the data is stationary. However, it appears that this time, the rolling mean varies little.

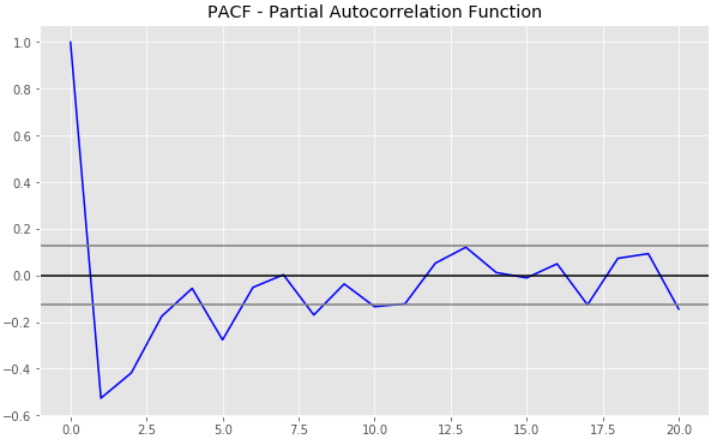
We now begin decomposing the (log) price data into trends and seasonal effects. As we can see in graphs 3.11.1-3.11.4, there is a clear upward trend, as we assumed. Additionally, there appears to be a clear seasonal effect. Finally, the residuals – what isn't explained by either the trend nor the seasonal effects – appears to hover around 0, with the variation increasing starting in 2014, as expected. As we can see in graphs 3.12.1-3.12.4, this breakdown appears to be the same with the original data (without the log transformation).

Now that we have stationary data, we need to find the optimal values for p and q, our autoregression parameters. The value of the parameter d is 1 since we subtracted past values one time. To find the value of p, we use an autocorrelated function (ACF) and determine at which value (nearest integer) the function crosses 0. As per graph 3.13, it appears to be 1. Similarly, to find the value of q, we use a partially autocorrelated function (PACF) and determine at which value the function crosses 0. Per graph 3.14, it too appears to be 1.

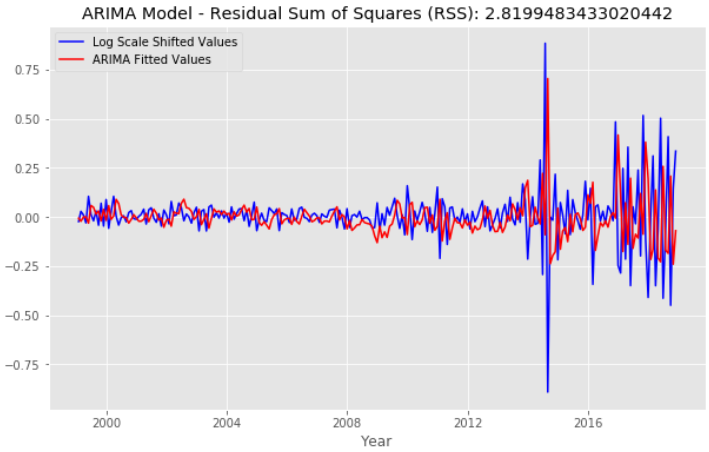
Finally, we can now apply the transformed data to an ARIMA model [6] (using the log scale prices and the log shifted data to ensure stationarity). After fitting the model, we plot the fitted values alongside the log scale shifted values (see graph 3.15) and calculate the residual sum of squares (RSS), which in this case is about 2.82. We essentially want this value to be minimized, as this means our model has optimally



Graph 3.13 – Autocorrelation Function (ACF) graph. Notice the first time the function appears to cross 0 on the y axis is around 1 on the x axis.



Graph 3.14 – Partial Autocorrelation Function (PACF) graph. Notice the first time the function appears to cross 0 on the y axis is around 1 on the x axis.

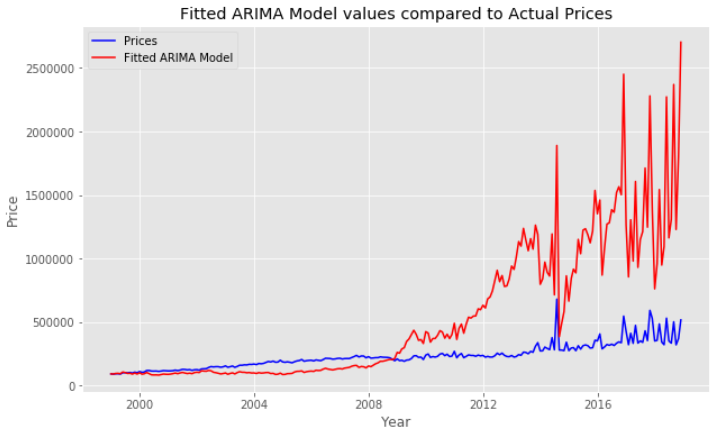


Graph 3.15 – The log scale shifted values alongside the fitted ARIMA values. Note the Residual Sum of Squares (RSS) is about 2.82.

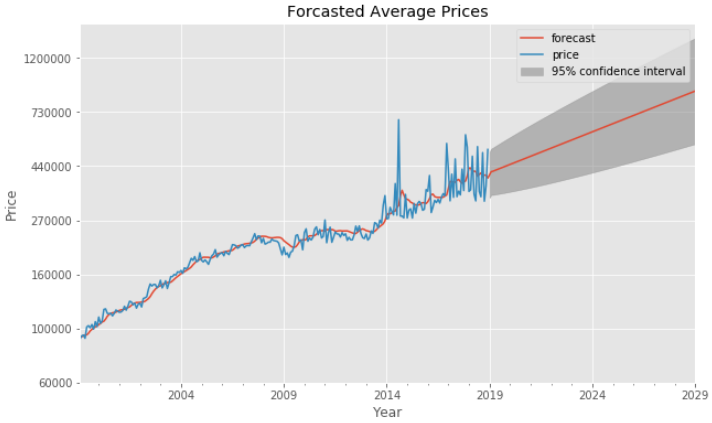
captured the trend and seasonal effects of the data, which will enable us to make more accurate predictions for the future.

Next, we take the exponent of the model-fitted data and compare it to the actual monthly average prices (not the log scale ones). In graph 3.16 we see that although the magnitude is not the same, the overall trend very much is, including many of the spikes and drops beginning in 2014.

At last, we are ready to predict. Using the ARIMA model we fitted, we plot the forecasted values alongside the actual values for the years 1999-2018, and then the forecasted average price for the next 10 years along with a 95% confidence interval range. As we can see (graph 3.17), the forecasted and actual prices line up fairly well, and it appears that the forecast for the coming years is for a continued, significant increase in average home prices (note the scale of the graph is exponential on the y axis).



Graph 3.16 – Comparison of fitted ARIMA values and actual prices. Note the magnitude/scale is off but the trend appears mostly correct.



Graph 3.17 – The forecasted values for 1999-2018 alongside the actual prices (a pretty good fit) and then a prediction for the average price through 2028 with a 95% confidence interval (the grey shaded area), based on the fitted ARIMA model.

VII. Discussion

A. *Project Goal*

The goal of this project was to study the residential real estate market in England and Wales over a 20 year period, attempting to glean insights as to which factors most affect the average price of a home.

To accomplish this, I gathered data the home price data as well as socio-economic data and studied different breakdowns and correlations between specific features as well as attempted to train different models on different versions of the data.

The models were trained both yearly as well as for all of the data combined and each was done twice. The first, having converted the property type column to 4 binary ones, and the second having converted it to a single discrete column.

While I was able to find projects on sites such as Kaggle in which the home price data was used, none of them attempted to combine socio-economic data to glean better insights. Additionally, none of them used more that a single year of data, and none attempted to apply machine learning models to predict prices.

B. *Results and Conclusions*

See sections V and VI.

C. *Pain Points*

I believe that one of the reasons for the relatively poor performance of the models lies in the data itself. The data I collected is relatively 'rich' in terms of factors that are – at least in part – related to geography. This is true for the features that are purely geographical, such as district, region, etc. as well as the socio-economic features, which are collected at the district level. Where I believe the data is lacking is in more minute details about the homes themselves. 2 examples:

1. Home Size – while there is a feature for property type, which helps distinguish between detached homes, semi-detached homes, terraced homes and flats, there is no feature for home size. I believe that features such as home size and property size (both in square meters or equivalent) and number of rooms in the home, could be useful in helping lower some of the variance in home prices. For example, given 2 flats in the exact same multi-story building, one being a 5 bedroom penthouse and the other a 2 bedroom on the second floor, one would intuitively expect the larger penthouse to sell for significantly more than the other.
2. Home Age – it is useful to know whether a home is new or old, but even more useful would be to know precisely how old (in years). For example, one would intuitively expect that given 2 otherwise equal homes, where one is 50 years old and the other is only 5 years

old, the price for the older one would probably be lower, yet neither of them would be classified as new.

Another thing that appears odd to me is that the most important feature for the Random Forest Regression models across all years is Brexit. As most of the years happened long before the referendum was even discussed as a possibility, I believe that it instead represents a measure of time, unrelated to Brexit. Perhaps a better measure would be to have the value as a constant 0 for all days prior to the announcement of results – or even the announcement of the referendum itself – and see if and how that affects the model.

D. Looking Ahead

While working on the project, and especially after seeing the results, I couldn't help but wonder if perhaps it would be more appropriate to look at the real estate market in a more segmented manner. England and Wales as a whole is not a homogeneous region, rather it is made up of different, distinct subdivisions. For example, the London market – a major world city which is highly urbanized – is likely very different from the markets in Wales and the North West of England, which are largely rural. Perhaps looking at the real estate market for each region separately would be more appropriate, or even just looking at London separately from England and Wales (minus London).

The errors of the models get worse by the year, but the price average and standard deviation increase too. As the standard deviation increases, we expect the errors to increase, but it is difficult to quantify precisely by how much, and if there is an improvement, a worsening or neither in the accuracy of the models from different years.

Another thing that would be valuable to this project would be to provide confidence intervals for each of the models. To be able to provide a confidence interval of 90% would require running each model at least 20 times which would take weeks.

Additionally, a more comprehensive time-series analysis might be able to provide more insights about the market and perhaps help identify anomalies that skew the overall results.

Finally, as suggested in the results section, an attempt to separate London from the rest of England and Wales, or even study each region separately, might yield more region-specific insights and may be more appropriate for studying the different markets in different parts of the country. There is enough data for each region to make this study possible.

REFERENCES

1. The housing price data was found on the UK Government's data website:

   https://data.gov.uk/dataset/4c9b7641-cf73-4fd9-869a-4bfeed6d440e/hm-land-registry-price-paid-data (2013-2018)

   https://data.gov.uk/dataset/314f77b3-e702-4545-8bcb-9ef8262ea0fd/archived-price-paid-information-residential-property-1995-to-2012 (1999-2012)

2. The mean and median income data was found on the website of UK Government's Office for National Statistics (ONS) in 2 files (1999-2017 and 2018): https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/earningsandworkinghours/adhocs/007997annualsurveyofhoursandearningsasheregionaltimeseries1997to2017 (1999-2017)

   https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/earningsandworkinghours/datasets/residencebasedtraveltoworkareaashetable12 (2018)

3. The prosperity data was found on the Legatum Institute's UK Prosperity Index website: http://uk.prosperity.com/

4. The implementation of the mini-batch Stochastic Gradient Descent algorithm was largely based on an implementation I found on GeeksforGeeks: https://www.geeksforgeeks.org/ml-mini-batch-gradient-descent-with-python/

5. The Random Forest Regression and Multi-Layer Perceptron models used scikit-learn's built-in models: https://scikit-learn.org/stable/

6. The time series analysis ARIMA model for prediction made use of statsmodels' built-in ARIMA model: https://www.statsmodels.org/stable/index.html

7. The population data for England and Wales came from Wikipedia:

   https://en.wikipedia.org/wiki/Demography_of_England

   https://en.wikipedia.org/wiki/Demography_of_Wales

Additionally, I derived inspiration from a number of other works in articles from TowardsDataScience.com, Kaggle.com and Medium.com.

ADDITIONAL MATERIAL

	SGD-D	SGD-B	RF-D	RF-B	MLP-D	MLP-B
1999	45426	45502	35434	35265	41914	33178
2000	52988	53330	40221	40544	39378	39226
2001	58096	58095	43992	43943	41322	50435
2002	66962	67180	49076	49551	45440	52939
2003	70074	69983	53115	52634	48670	48937
2004	75536	75245	58443	58695	56186	69579
2005	77394	76950	60519	60274	56168	54182
2006	84621	84518	65353	65704	59922	61570
2007	93843	94245	72842	72752	87640	72008
2008	93070	92560	76241	76366	77484	78058
2009	90709	90275	75931	75321	86018	78712
2010	107371	108497	86694	86711	104200	85093
2011	108121	108110	86240	86550	80471	77524
2012	111482	111393	89248	91508	102108	92385
2013	121416	121405	91689	96540	89841	108742
2014	134243	134120	96012	95888	100571	92852
2015	137833	137193	95815	95328	94653	87403
2016	141108	141007	94882	99421	89090	89175
2017	146908	145857	100617	98678	107188	92117
2018	144226	143045	99382	96273	112014	100180
All	103230	103062	97861	96273	68655	64849

Table 1.1: (results are rounded, ‘D’-property_type converted to discrete column, ‘B’-property_type converted to 4 binary columns, ‘SGD’-Stochastic Gradient Descent, ‘RF’-Random Forest Regression, ‘MLP’-Multi-Layer Perceptron)