

Predicting Days-on-Market for Residential Real Estate Sales

Sergey V. Ermolin

Department of Computer Science

Stanford University

sermolin@stanford.edu

Abstract

By analyzing historical residential sales data in San Francisco Bay Area, we predict how many days a house will stay on the market before a sale occurs. This method would enable owners, investors, and real-estate agents to price properties based on individual's personal circumstances and cost-of-money financial considerations

1 Introduction

1.1 Motivation

When a property is about to be put on the market, establishing a list price is a non-trivial exercise. The list price is usually arrived at by surveying comparative sales and taking into account the length of time the owner/investor is willing to wait to close the deal. Most of this process is based on the experience of a real-estate agent and how ambitious an owner/investor is. Analytics-based approach to establishing list price is usually constrained to multiplying the house's square footage by the average cost-per-sq.ft of comparable houses and then scaling it up or down based on the agent's subjective recommendations and particular financial and time constraints of the seller/investor. This publication attempts to provide a more rigorous approach to this problem based on predictive data-analytics.

1.2 Real life use-case

Since a List Price of a house is one of the parameters in our predictive model, a real estate agent would be able to utilize it to predict Days-on-Market as a function of the property's List Price once all other parameters have been entered into the model (see dataset section below). There already exists a significant body of work on predicting sales price of a house, both in academia and industry (eg. Zillow's Zestimate). Thus, the task of predicting the sales price lies outside of the scope of this paper and for the purposes of this analysis we assume that a true value of a property is available as either the final sales price (in a training/test data set) or as a Zestimate or an appraised value (for a house to be put up for sale).

2 Related work

While reviewing prior work using <https://scholar.google.com>, we did not find publications claiming to predict days on market. Some works (eg. [5]) show correlation between days on market and sales price, but none went as far as providing a prediction method.

The most popular (and the most obvious) approach described in the literature was the use of multi-variate linear regression analysis. Features with non-linear correlation are typically mapped to a linear form by using continuous functions such as logarithm, exponent, or a polynomial.

In [1] the authors maintain, however, that linear regression has a limited usefulness because of its inherent inability to account for "...nonlinear interactions between the numerical features and the price targets". While we also started our analysis with multivariate linear regression, we quickly came to the same conclusion. In the same paper, the authors concluded that decision-tree algorithms, such as RandomForest or K-Nearest Neighbor are better able to account for such non-linearities.

Based on the review of the related work and conversations with real-estate agents, a few assumptions about the real-estate market seem to be universally used:

1. In the SF Bay area, Days-on-Market below 1 week or above 2 months or sales/list price ratio outside [0.8 : 1.2] range indicate an outlier either in a sense of mispricing or some other deficiency (poor condition, liens, title problems, etc).
2. Should one use linear regression, it is advisable to add a normalizing term to our regression loss function.
3. The number of offers that arrive per week is well-modeled by a Poisson distribution. The time between offers is well-modeled by an exponential distribution.

Publication [2] contained a closed-form formula for DOM dependency on TrueHomeValue, SalesPrice, variation in purchase offers and overall market activity. The paper did not offer any information on testing this formula against real sales data. As part of future work, it

would be interesting to explore its applicability to our dataset. $E[P_{t+1}] = (P_t + \sqrt{3}\sigma_{P_t}) - \frac{2\sqrt{3}}{\lambda E[TOM_{t+1}]^\theta + 1} \sigma_{P_t}$

3 Dataset, Features, and pre-processing

Courtesy of MLS Listings, Inc, we obtained 3 consecutive months' worth of mid-2016 single family residential sales data in five local counties (Santa Clara, Santa Cruz, San Mateo, San Benito, and Monterey). The data in the form of a CSV file was extracted from MLS Listings Business Intelligence data warehouse and as such came well-regularized and validated for content correctness. Even so, some non-mandatory fields such as ElementarySchool and HighSchool were left blank by real estate agents and were dropped from our analysis. We also restricted the data to Days-on-Market within the range of [7 to 60] and Sales/List price ratio within the range of [0.8 to 1.2] (see discussion in "Related Work" section above). We further augmented the data by adding publicly available high school and elementary school ratings based on state-wide test scores. No such data was available for middle schools for some reason. In the end, our dataset shrunk from 4998 down to 1690 records, mostly due to lack of Elementary School and High School entries. To our surprise, Python's sklearn.tree and sklearn.ensemble libraries could not accept non-numerical values (such as "High School District"). To compensate for it, we had to add numerical columns for such categorical variables to make the dataset work with decision tree algorithms. This is perhaps an area to be explored in the future and maybe even improve built-in python libraries.

Table 1: Dataset format and Features

DOM	age	zip	Pool	Fire	PLace	city	ListPrice	SalesPrice	Bdrm	Bath	SqFt	ElemSchool	HighSchool	Elementary	HighSchool	Latitude	Longitude
32	76	95002	0	0	ALVISO		450000	470000	2	1	584	Santa Clara Unified	Santa Clara Unified	George Mayne	Adrian Wilcox High	37.42815	-121.973
43	59	95008	1	1	CAMPBELL		1329000	1329000	4	3	2925	Cambrian Elementary	Campbell Union High	Bagby Elementary	Branham High	37.27322	-121.939
36	33	95008	0	1	CAMPBELL		699000	703000	2	2	1346	Cambrian Elementary	Campbell Union High	Bagby Elementary	Branham High	37.28507	-121.936
9	38	95008	1	0	CAMPBELL		410000	425000	1	1	704	Elementary	Campbell Union High	Bagby Elementary	Branham High	37.28472	-121.936
44	58	95008	0	1	CAMPBELL		1285000	1230000	4	2	1997	Elementary	Campbell Union High	Bagby Elementary	Branham High	37.28293	-121.93

We performed an extensive dataset exploration and visualization to uncover meaningful features that affect DOM. Two issues became apparent relatively quickly:

1. The only numeric feature that significantly affected DOM was the ratio of Sales/List price (SPLP). See figure 1.
2. DOM ~ SPLP dependency became more pronounced (but still weak) as we reduced the set to geographically homogeneous entries (same School District, same school, etc). One can also see that just a linear regression by itself would not be able to produce reasonable prediction results. See figure 2

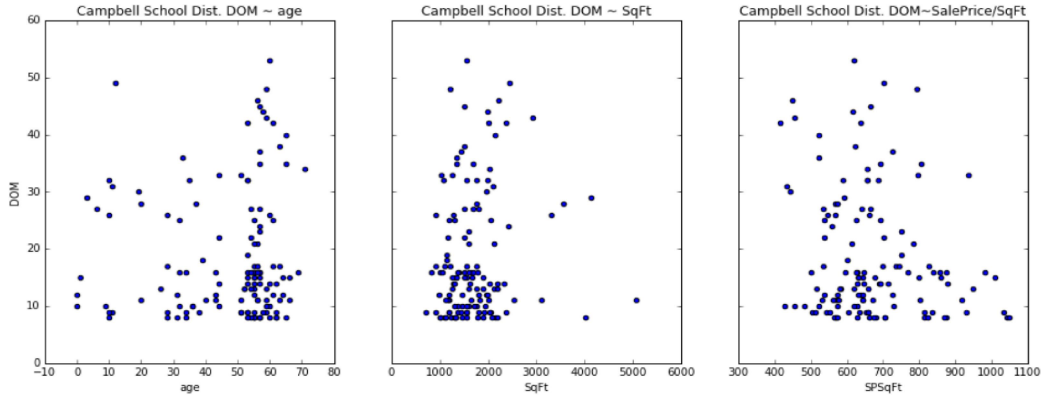


Figure 1 – Evidence of no clear dependency of DOM on age, house SqFt, or SalesPrice/SqFt.

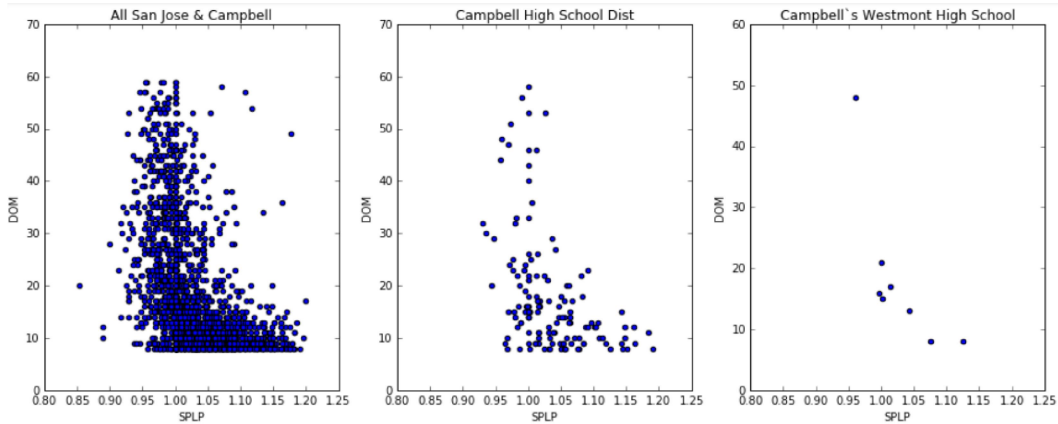


Figure 2 – Illustration of the effect of geographical localization on DOM vs. SPLP dependency.

To our surprise, utilizing geo-spatial location data (latitude/longitude) did not improve DOM ~ SPLP correlation. Our expectation was that by predicting DOM~SPLP dependency on N geographically closest houses, we would be able to produce a more homogeneous data subset than by clustering by zip/School District. However, the data did not bear out this concept. This is understandable given the jaggedness of zip codes and school district boundaries.

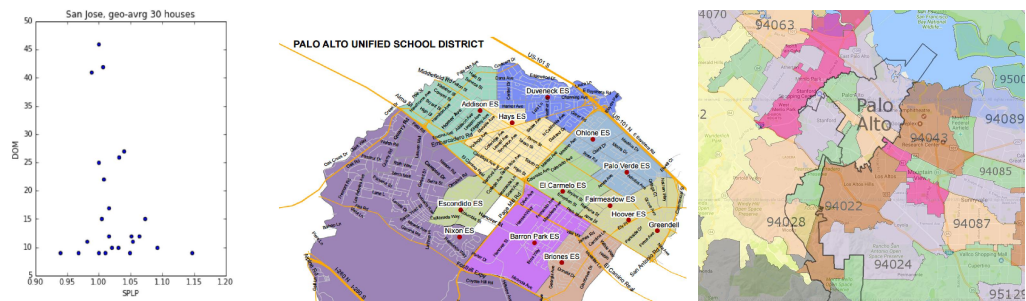


Figure 3 – Due to jagged zip code and school district boundaries, radial geo-spatial clustering does not always produce improved DOC ~ SPLP dependency.

4 Prediction Methods

Based on the results of data exploration and visualization, we abandoned the use of linear multivariate regression and concentrated instead on two decision-tree algorithms: basic decision tree regressor and Random Forest. For completeness, we also ran multivariate regression alongside. We used Python 3.0 for data processing, making extensive use of Panda data frames and built-in sklearn and scipy libraries.

Basic Regression Decision Tree:

- *Building/Training:* A decision tree is built/trained by a recursive algorithm. At each step we find the feature in the training set to be used for making the split decision. The feature choice is usually done by either maximizing information gain (for classification tree), or by the variance reduction (for regression tree) of the remaining data downstream of each split node. Prior to making the split decision, we evaluate whether we reached either maximum tree depth or if all data at the node is homogeneous or if we reached the minimum data points per leaf. For continuous data such as DOM, sometime we bin the data in to bins/ranges. Regression is then implemented at leaf nodes. Reference [11] describes this procedure the best, albeit in a somewhat embellishing way: “we... sub-divide, or partition, the (feature) space into smaller regions,... until finally we get to chunks of the space which are so tame that we can fit simple models to them”
- *Using/Testing:* A decision tree is used on test data as follows: the tree takes data records and analyses variables one-at-a-time while descending along the tree. Once it reaches the final leaf node, a regression is performed to fit the data.

RandomForest is a variant of a decision tree algorithm with the following enhancements:

- During tree construction/training at each split node candidate, the random forest algorithm selects a random set of features.
- Another important feature of Random Forest is ‘bagging’ (bootstrap aggregation) where we repeatedly take K data samples with replacement from the original training dataset and use each sample to build a separate decision tree. When it is time to use/test the decision tree algorithm against a test example, the method applies all created decision trees to the test example and averages out the results.

Both of the two Random Forest properties above help with data overfitting when several features are heavily correlated causing node decisions to be correlated as well. Since our data exploration analysis did not reveal any feature correlation, this should not be a concern and we would expect very similar results from Random Forest and a “plain vanilla” Decision Tree Regressor. The final data bears this out.

Multi-variate regression predicts response variables’ value to be a linear combination of input variables x_n : $response = a_0 + a_1 * x_1 + a_2 * x_2 + a_3 * x_3 + \dots$. As one can imagine, it would not do well to predict response’s dependency on categorical values such as ‘Fremont High School’ or ‘Leigh High School’.

5 Results/Discussion.

Below are the results of DOM actual vs DOM predicted obtained by Random Forest algorithm for the cities of San Jose and Campbell. We cross-validated the training algorithm using k-fold cross-validation using random sampling for k=10 and the plots in Figure 4 are done for one of the cross-validation “folds”. The final set of eight features selected for Decision Tree fitting was: [City, Sales/List_Price, Sq.Ft, Elementary_School_District, High_School_District, Elementary_School_District, Elementary_School_Rate, High_School_Rate]. Limiting max_depth parameter to 5 proved to provide the best results. Higher than that made prediction worse (too many branches with too few leaves on the decision tree). Lower than that shrunk the predicted DOM down to just a few values.

Table 2 summarizes DOM’s average absolute prediction errors obtained for each of the attempted methods. The errors are averaged using k-fold algorithm described above.

Algorithm	Random Forest	DecisionTree Regressor	Multi-var Linear Regression
Pred Error	7.5 days	6.8 days	13.5 days

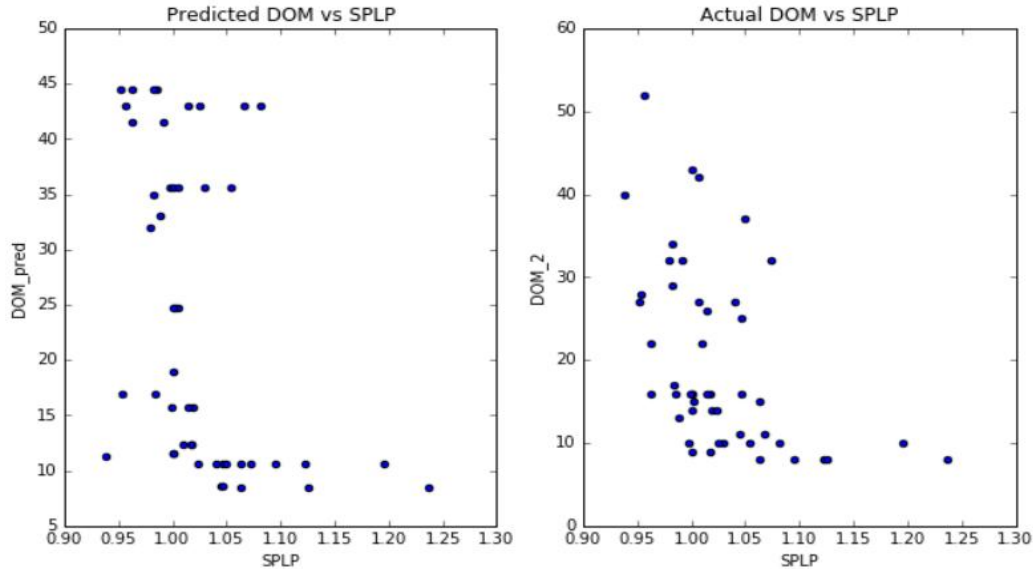


Figure 4 – predicted vs actual DOM obtained by Random Forest method.

6 Conclusion/Future Work

We succeeded in being able to predict property’s Days-on-Market to within 7 days. In real-life, any accuracy better than 1-week would be considered unreasonable due to the seasonality and weather-dependency of a real-estate market (eg. on a rainy or very hot week, a house may get only 30% of average traffic). Not surprisingly, linear regression performed poorly while decision-tree algorithms did better. Random Forest did not provide an advantage over a plain-vanilla Decision Tree Regressor because we did not have correlated features in our dataset. The fact that Random Forest performed a bit worse than Decision Tree Regressor is somewhat baffling and deserves further attention in future work. Perhaps there is room for improvement in tuning its parameters. Contrary to our expectations, but not surprising in retrospect, incorporating geo-spatial information in predictive process did not improve the results because of jagged school district and zip code boundaries.

Areas to continue working on in the future:

- Further refinement of feature selection and Decision Tree algorithm’s parameters.
- 2nd attempt of incorporating geo-spatial information in decision process.
- Based on the suggestion in ref [10], we may want to implement C4.5 decision tree since it is supposed to better handle non-linear dependencies between our features and target variable of DOM.
- Explore Convolutional Networks as another possible approach.
- Improve Python random tree libraries to allow inclusion of categorical variables.

Project material is considered for publication at the following venues:

- Journal of Real Estate Economics
- American Real Estate and Urban Economics Association conference (Washington, DC June 1-2, 2017).
- American Real Estate Standards Organization Conf. (Chicago, April 24-26, 2017)

Acknowledgments

- MLS Listings for providing several revisions of the dataset and advising us on data engineering and pre-processing
- Pavel Berkhin (Microsoft) for general advice on machine learning methods and feature selection.
- Hao Sheng (Stanford) for advice on the project’s topic and several reviews along the way.
- Gregory Shklover (Intel) for help and advice with Python

References

- [1]. Nissan Pow, Emil Janulewicz, Liu (Dave) Liu. Prediction of real estate property prices in Montr´eal
- [2]. Ping Cheng,* Zhenguo Lin** and Yingchun Liu***A Model of Time-on-Market and Real Estate Price Under Sequential Search with Recall
- [3]. Hengshu Zhu1,Hui Xiong2,Fangshuang Tang3,Qi Liu3,Yong Ge4,Enhong Chen3,Yanjie Fu5. Days on Market: Measuring Liquidity in Real Estate Markets
- [4]. Andreja Cirman, Marko Pahor, Miroslav Verbic. Determinants of Time on the Market in a Thin Real Estate Market
- [5]. John R. Knight. Listing Price, Time on Market, and Ultimate Selling Price: Causes and Effects of Listing Price Changes. *Journal of Real Estate Economics*, 2002
- [6]. https://en.wikipedia.org/wiki/Random_forest
- [7]. <http://scikit-learn.org/stable/modules/ensemble.html#forest>
- [8]. http://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html#sphx-glr-auto-examples-tree-plot-tree-regression-py
- [9]. Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, *JMLR* 12, pp. 2825-2830, 2011.
- [10]. www.greatschools.org
- [11]. http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm
- [12]. <http://www.simafore.com/blog/bid/62482/2-main-differences-between-classification-and-regression-trees>
- [13]. <http://www.stat.cmu.edu/~cshalizi/350-2006/lecture-10.pdf>

By improving Random Forest and Decision Tree parameters and reducing the number of features, we were able to achieve a (50%) improvement in prediction error compared to the data provided during the poster session.