Real Estate Price Prediction – Final Report

Jiawen Tong, Shiyun Qiu, Yiqi Xie, Chia Chi (Michelle) Ho

April 30, 2018

For detailed implementation and code, please see our Github Link.

1 Motivation and Problem Statement

Real estate purchases is one of the most substantial investment one can make in life, and the real estate market constitutes a significant part of the overall economy. Therefore, the ability to accurately predict real estate prices and trends is lucrative and valuable. For this project, we aim to build a predictive model that accurately forecasts the sold price and the number of days on market of real estate properties in Boston. The specific goals of our project are as follows:

- Curate historical real estate transaction data with an emphasis on property images
- Develop feature extraction methods and identify key factors that determine property prices
- Build predictive models to predict sold price and the number of days on market

2 Data Description

2.1 MLS Data (Provided)

There are 45 columns of fields in the provided .csv file. The key fields are:

- 'MLSNUM': A unique string that identifies the property
- 'STATUS': The status of the property (sold, pending,...etc.), categorical
- 'LISTDATE': List dates, datetime object
- 'SOLDPRICE', 'DOM': Sold price and days on market, all numeric
- 'ADDRESS', 'CITY', 'STATE', 'ZIP': Street address, city, state and zip, all categorical
- 'BEDS', 'BATHS': Number of beds and baths in the property, all numeric
- 'SQFT', 'AGE', 'LOTSIZE': Square feet, age and lot size, all numeric
- 'GARAGE': Number of parking spaces in the garage, numeric

- 'REMARKS': Description of the property in detail, string

2.2 Redfin Data (Externally Curated)

There are 33 columns of fields that we scraped. The key fields are:

- 'beds', 'baths': Number of beds and baths in the property, all numeric
- 'sqft_finished', 'sqft_unfinished': Finished and unfinished square footage, all numeric
- 'year_built', 'year_renovated': Year built and renovated, datetime object
- 'parking_space', 'garage_space': Number of parking spaces in and out of the garage, all numeric
- 'school_ratings', 'school_distances': Top 5 closest school ratings and distances, all numeric
- 'walk_score', 'transit_score', 'bike_score': Convenience scores, all numeric
- 'num_photo': Number of photos posted on Redfin, numeric
- photos posted on Redfin

3 Exploratory Data Analysis

We explored variables given and features scraped from Redfin. Based on the plots below, we found:

- Both response variables, soldprice and DOM, are very right skewed. A single log transformation makes the price distribution much more symmetric while DOM requires 2 sequential log transformation. We will use log(price) and $\log(\log(\text{DOM} + 5))$ as our response variables.
- The relationships between the log sold price and most of the explanatory variables are quite linear. Note that the features scraped from Redfin and extracted from remarks also provide strong explanation to the variance of the sold price.
- The log-log DOM the trends are much less pronounced, which implies that model predictions on DOM using these features could be weak.
- The distributions and the relationships are different across different property types, suggesting that separate models for different types of properties are justified and warranted.

3.1 Exploring Price

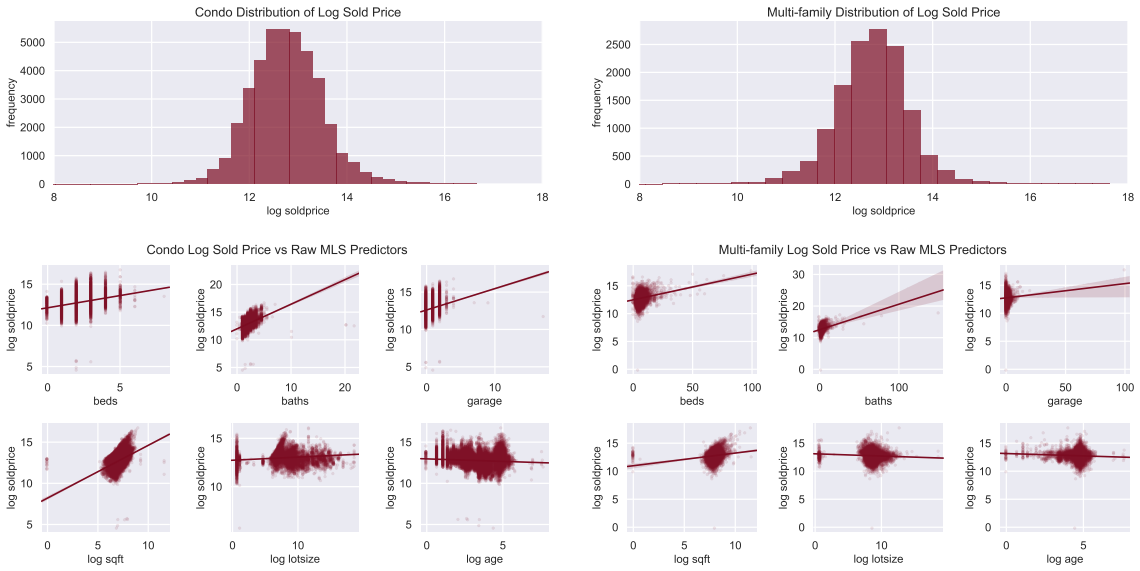

Figure 1: Price exploration on the provided MLS dataset. Note that the original sold price distribution is very skewed. We have taken logarithm to mitigate.

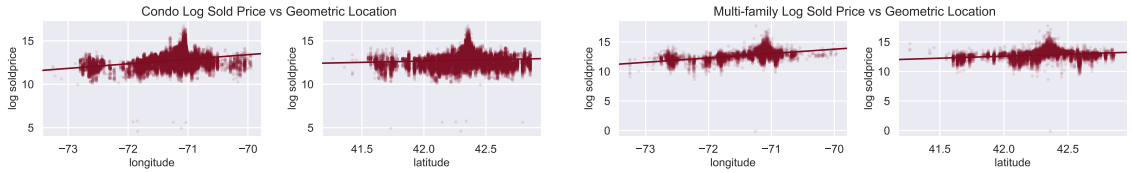

Figure 2: Price vs location. The geometric coordinates are scraped from Google map using zipcodes.

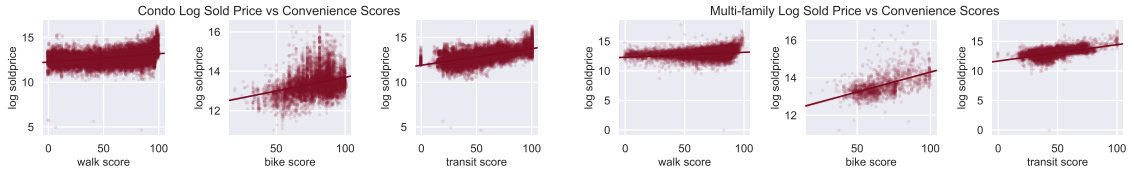

Figure 3: Price vs convenient scores. The convenient scores are scraped from Redfin webpages.

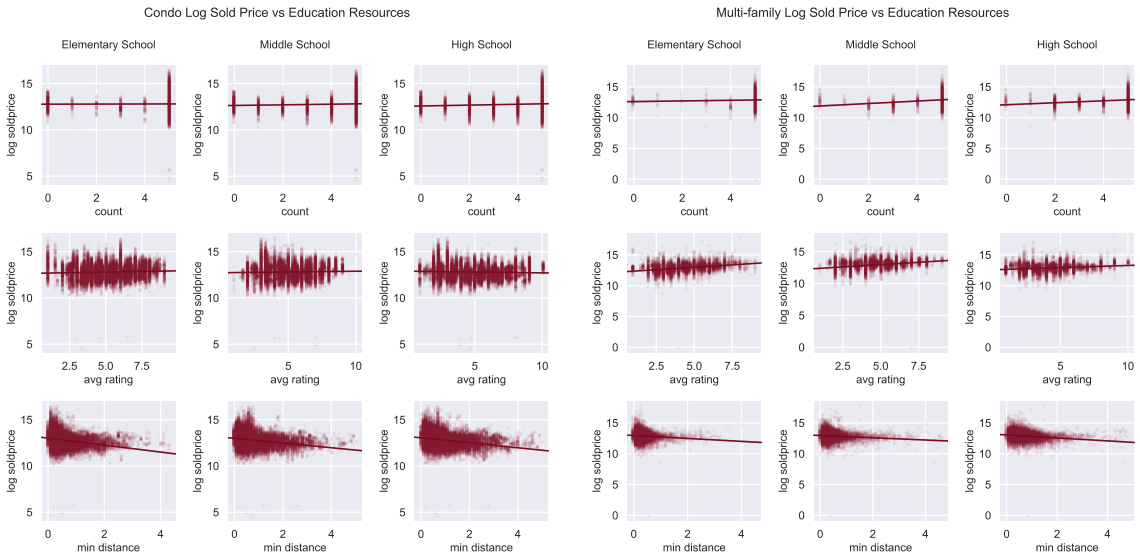

Figure 4: Price vs nearby education resources. The school information is scraped from Redfin webpages. The "count" is the number of schools within 5 miles. Redfin only shows the nearest 5 schools, so the count number is no greater than 5.

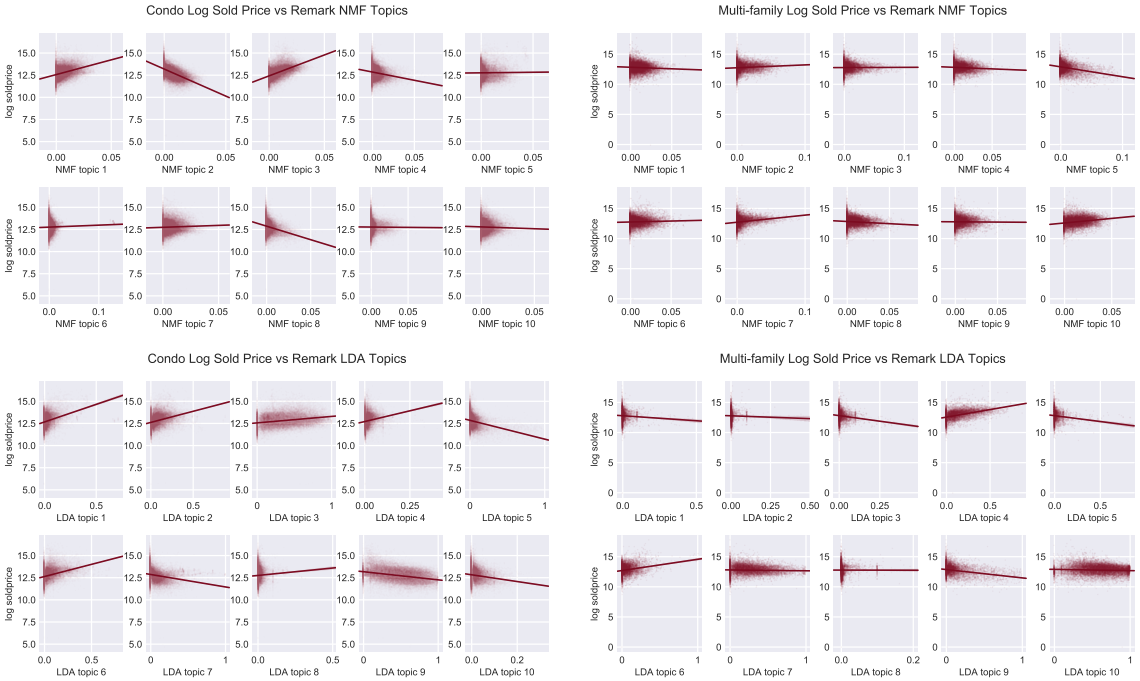

Figure 5: Price vs remarks in the provided MLS dataset. We applied both NMF and LDA to extract the topics. Note that the order of the topics is arbitrary, and the topic sets generated through NMF and LDA could be very different.

3.2 Exploring DOM

Similar but less obvious trends are observed with DOM.

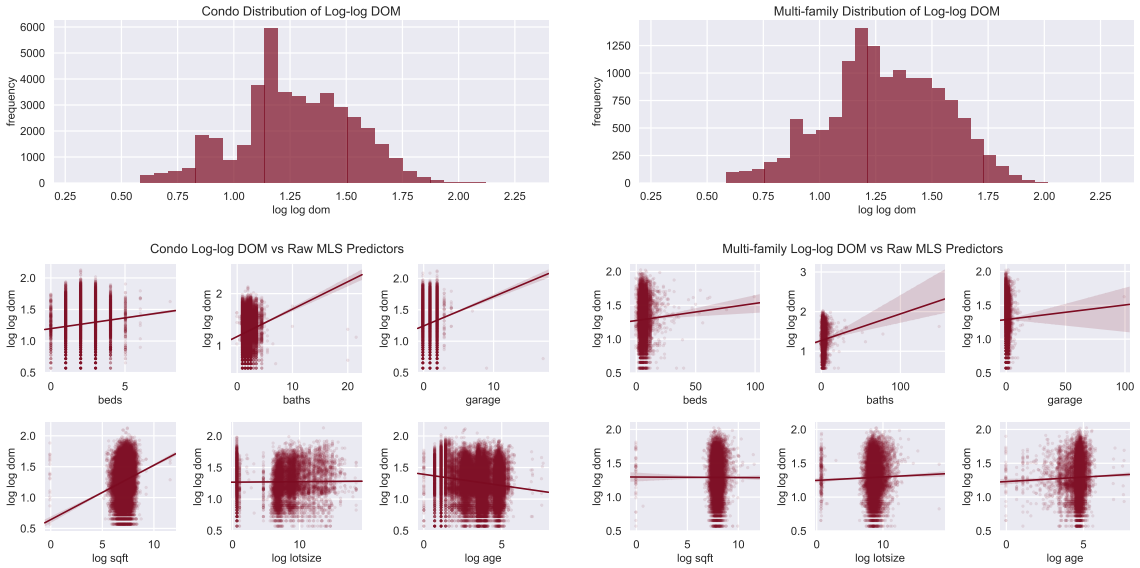

Figure 6: DOM exploration on the provided MLS dataset. Note that the original dom distribution is very skewed. We have taken $\log(\log(\text{DOM} + 5))$ to mitigate.

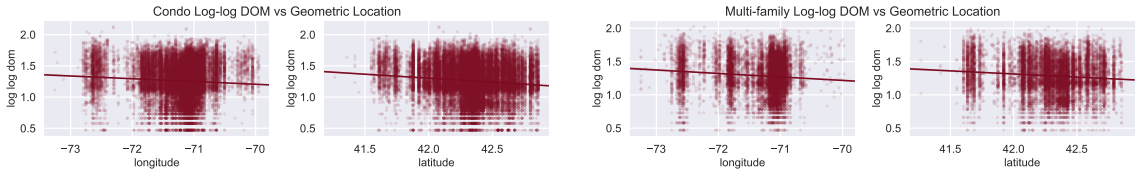

Figure 7: DOM vs location. The geometric coordinates are scraped from Google map using zip-codes.

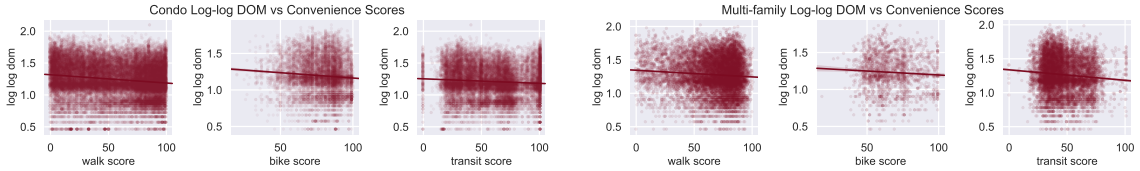

Figure 8: DOM vs convenient scores. The convenient scores are scraped from Redfin webpages.

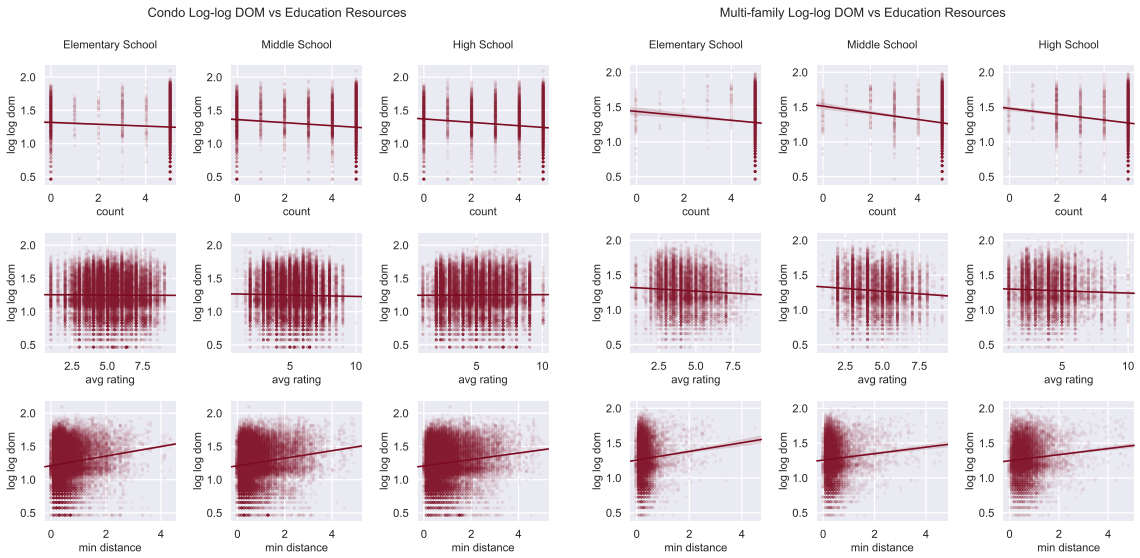

Figure 9: DOM vs nearby education resources. The school information is scraped from Redfin pages. The "count" is the number of schools within 5 miles. Redfin only shows the nearest 5 schools, so the count number is no greater than 5.

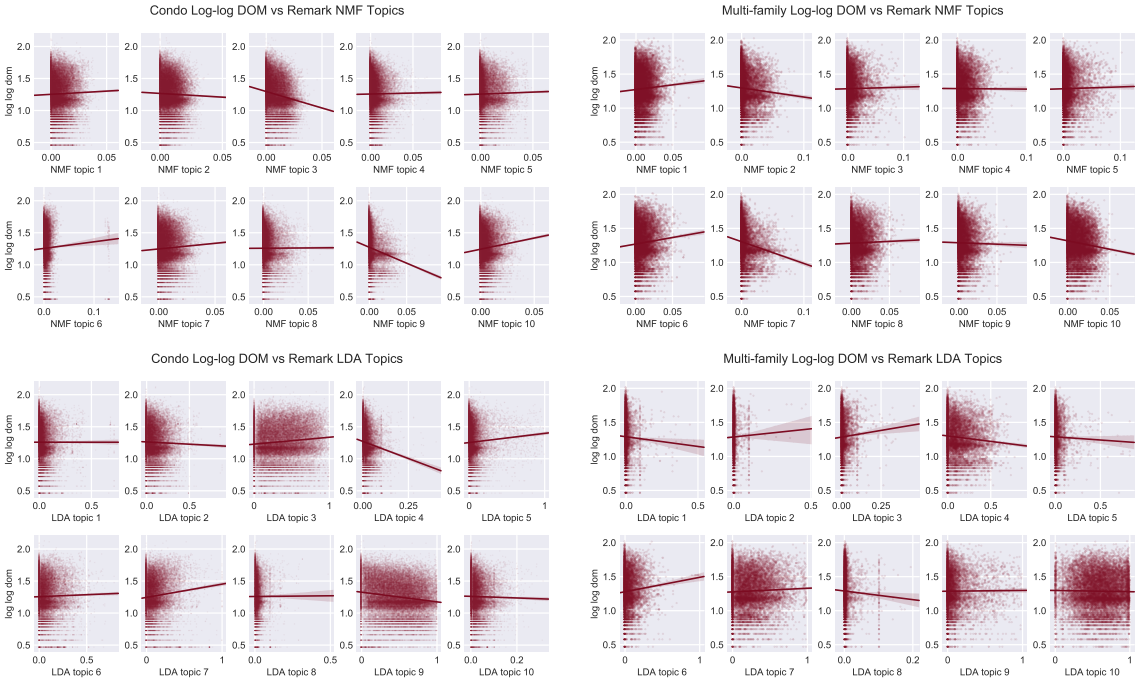

Figure 10: DOM vs remarks in the provided MLS dataset. We applied both NMF and LDA to extract the topics. Note that the order of the topics is arbitrary, and the topic sets generated through NMF and LDA could be very different.

4 Literature Review

You et al. proposed a novel framework for real estate price estimation by 1) quantifying the impact of visual features using deep Convolutional Neural Networks (CNNs); 2) employing random walks to generate a geographically neighboring houses sequence and using Recurrent Neural Networks (RNNs) to predict the housing prices in this sequence. The proposed framework was proved to be effective and flexible at predicting housing prices in two selected cities [1]. Specifically to our interest was their approach to extract visual features from real estate images: They used the output of the last average pooling layer of Googlenet to represent each image as a 1024-dimensional vector. The final visual representation for each house then is the average of all of the images belonging to that house. We used a similar approach as discussed below.

5 Approach

Our overall approach is summarized in Figure 11.

Figure 11: Overall Approach

5.1 Data Preprocessing/Feature Extraction

Our data preprocessing and feature extraction methods are summarized in Figure 12.

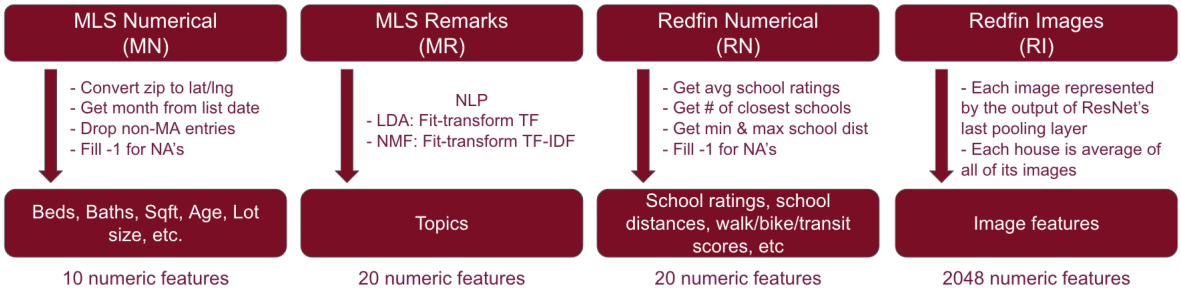

Figure 12: Feature Extraction

The full list of features generated using our approach are shown below.

1. **MLS numerical**: AGE, LOTSIZE, GARAGES, LISTMONTH, LAT, LNG, baths, beds, sqft_unfinished, sqft_finished

2. **MLS remarks**: lda_0 to lda_9, nmf_0 to nmf_9
3. **Redfin numerical**: num_photos, parking_space, garage_space, year_built, year_renovated, school_ratings_elementary_avg, school_number_elementary, school_distances_elementary_max, school_distances_elementary_min, school_ratings_middle_avg, school_number_middle, school_distances_middle_max, school_distances_middle_min, school_ratings_high_avg, school_number_high, school_distances_high_max, school_distances_high_min, walk_score, transit_score, bike_score
4. **Redfin images**: resnet_1 to resnet_2048

Finally, the steps to generate these features are as follows:

1. Convert (city, state, zip code) combinations into (longitude, latitude) via the Google Map Geoencoding API – This produces 2 features.
2. Transform remark text into topic features by Non-negative Matrix Factorization (NMF) and Latent Dirichlet Allocation (LDA) – This produces 20 features.
    - Fit and transform the tfidf-Vectorizer to get `tf-idf` (term-frequency-inverse-document-frequency) matrix and count-Vectorizer to get `tf` (term-frequency) matrix on the training data
    - Transform the remarks in the test data with the fitted tfidf-Vectorizer and count-Vectorizer
    - Fit NMF on the transformed training `tf-idf` and LDA on the transformed training `tf`
    - Apply the fitted NMF and LDA model to the transformed remark matrices to get the remark topic features
3. Extract features from images
    - Utilize keras.ResNet50 and pretrained imagenet weights as our model.
    - For each image, extract a 2048-dimensional feature vector by obtaining the output of the last layer (`flatten_1`) prior to the final prediction layer in the model.
    - For each house, take the average features of all of its images as the final image features used to predict sold price.

    This produces 2048 features.
4. Extract features from top 5 closest school ratings/distances
    - Record average of school ratings and distances
    - Record number of closest schools
    - Record the minimum and maximum distances from the house to the schools

    This produces 5 features.
5. Extract month from the list date
6. Merge MLS/Redfin data and drop listings that are not in Massachusetts

7. Take the Redfin features if they overlap with the MLS ones (e.g., BEDS, BATHS, SQFT)
8. Replace all unreasonable values (e.g. AGE = -8000) and/or missing data with -1

5.2 Feature Sets

From the provided data set and additionally curated Redfin data, we divided the features into 4 categories as shown in Figure 11. Subsequently, 6 feature sets were formed by combinations of these categories as shown in Table 1.

Type	**Source**	**Feature Set**					
		Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
Non-image	MLS numerical	X	X	X	X	X	X
	MLS remarks		X		X		X
	Redfin numerical			X	X	X	X
Image	Redfin images					X	X
Total # of features		10	30	30	50	2078	2098

Table 1: Feature Sets

5.3 Modeling Approach

Based on our EDA results, we chose to model different property types independently. For this project, we focused on condos (CON) and multi-family (MF) properties. We divided the whole training data of MF and CON (Apr 2016 to Dec 2017) into 90% training set and 10% validation set. We used Ridge, XGBoost (eXtreme Gradient Boosting) and LightGBM as our base models with which we then used a stacking approach to make the final predictions on the test set (Jan 2018 to March 2018). Our modeling approach is summarized in Figure 13, and the detailed steps are described following the figure.

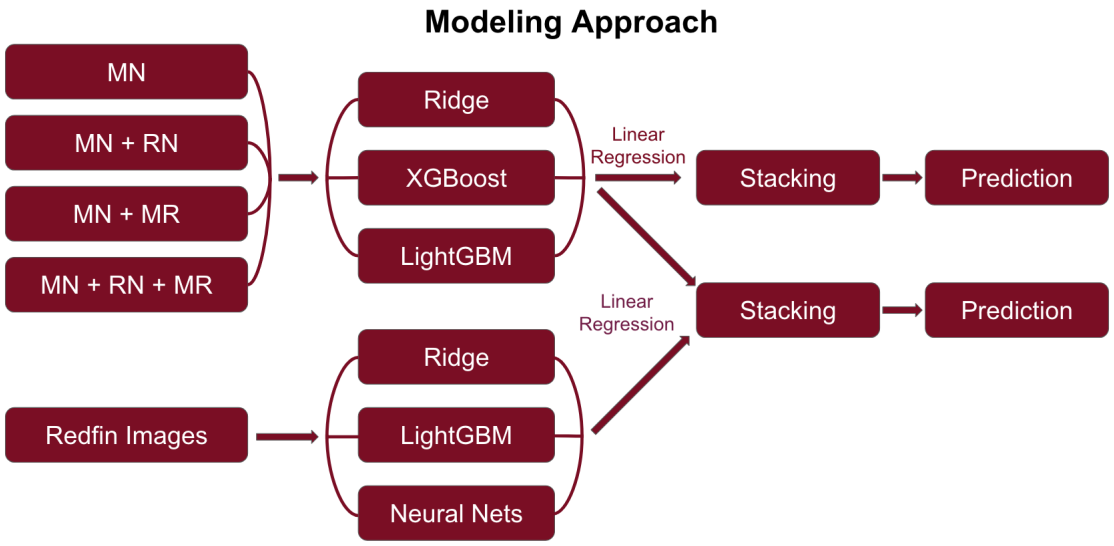

Figure 13: Modeling Approach. MN = MLS numerical, RN = Redfin numerical, MR = MLS remarks features.

The models were built in 2 sequential steps.

- **Step 1**: For each feature set from Set 1 to Set 4, we built 3 predictive models using Ridge, XGBoost and LightGBM. In addition, we built 3 predictive models using LightGBM, Ridge and multi-layer feed-forward neural network. In this step, we built 15 models for a single response variable (e.g. condo SOLDPRICE, multi-family SOLDPRICE). See Table 2 below.

Model form	**Set 1**	**Set 2**	**Set 3**	**Set 4**	**Image**
XGBoost	Model 1	Model 4	Model 7	Model 10	
Light GBM	Model 2	Model 5	Model 8	Model 11	Model 13
Ridge	Model 3	Model 6	Model 9	Model 12	Model 14
NN					Model 15

Table 2: Model numeric designations for model form/feature set combinations

- **Step 2**: In this step, we stacked/combined models using linear regression to get final predictions. See Table 3 below.

Models Used	Feature Set					
	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
Model 1	X					
Model 2	X					
Model 3	X					
Model 4		X				
Model 5		X				
Model 6		X				
Model 7			X		X	
Model 8			X		X	
Model 9			X		X	
Model 10				X		X
Model 11				X		X
Model 12				X		X
Model 13					X	X
Model 14					X	X
Model 15					X	X

Table 3: Feature Sets and the models used for stacking and final predictions. For instance, for feature set 5, the predictions from models 7, 8, 9, 13, 14, and 15 were used as features in linear regression to predict the response variable.

Using this approach, we built predictive models for the following response variables:

- Condos - SOLDPRICE
- Condos - DOM
- Multi-Family - SOLDPRICE
- Multi-Family - DOM

For single-family (SF) houses, we applied a simplified approach only with features extracted from the given MLS data as we didn't have enough time to scrape Redfin data and images for all SF properties. We divided the whole SF training data into 70% training set and 30% validation set and fitted 3 models, Ridge, XGBoost and LightGBM. This amounts to 87 models in total.

6 Results and Discussion

Here, we focus our discussion on modeling SOLDPRICE, and the results for DOM are included in the appendix. Our results for modeling SOLDPRICE are summarized in the tables below, and our main findings are:

1. The **best models** with highest test performance for predicting SOLDPRICE are:
    - Condos — built using all of the features (Feature set 6: MLS + Redfin + Remarks + Images) with a $R^2_{validation} = 0.948$ and $R^2_{test} = 0.911$

- Multi-family — built using all non-image features (Feature set 4: MLS + Redfin + Remarks) with a $R^2_{validation} = 0.849$ and $R^2_{test} = 0.803$

2. **Model generalizability**: Using training performance as the reference, we observed more significant drop in test performance than validation performance with all model/feature set/response combinations. We suspect that this is because the training and validation sets come from the same time period (Apr 2016 to Dec 2017) whereas the test set is from a later time period (Jan 2018 to March 2018). In this way, the validation set is an interpolation in time whereas the test set is an extrapolation in time. Since our data features do not in general reflect overall economy and market temperature which closely relate to time, it is not surprising that the test performance dropped. The models for predicting condo prices show comparable training/validation scores, suggesting good generalizability within the training time frame for condo. However, the validation scores are significantly lower than training scores for multi-family properties. This may be due to greater inherent variance and smaller sample size ($\sim 1/4$ of the condos sample size) of multi-family data, which make it more prone to overfitting. Please see next section for possible future directions to improve validation and test performances.

3. **Efficacy of remarks features**: Recall that we used NMF and LDA to extract topic features from remarks. Adding these remarks features consistently improved model performance. For instance, models using MLS + Remarks features performed better on the training, validation and test sets than those using MLS features only. This suggests that our methodology was effective at extracting predictive and generalizable language remarks features.

4. **Efficacy of Redfin features**: Adding numeric features scraped from Redfin (e.g. average school ratings, average school distances and convenience scores such as walk and transit scores) improved training, validation and test scores on both condos and multi-family properties compared to using base features (MLS numeric features only). Such improvement suggests the numeric features extracted from Redfin data are effective.

5. **Efficacy of image features**: Adding image features (2048 ResNet50 final layer output) only marginally improved model performance compared to MLS + Redfin or MLS + Redfin + Remarks feature combinations. One possible explanation is that our ensemble models are flexible enough so that the additional feature dimensions without additional observations increase the propensity of overfitting. Another explanation is that our visual features are sub-optimal for price prediction. The ResNet outputs contain features for ImageNet classification which may not be as effective to capture real-estate image semantics. We discuss some future directions to potentially better incorporate this wealth of data in the next section.

6. **Single-family properties**: Using feature set 2 (MLS numerical and remarks), LightGBM performed the best with a $R^2_{validation} = 0.904$ and $R^2_{test} = 0.904$.

Condo: Price R^2 (Ensemble Model)			
	Training	Validation	Test
MLS	0.977	0.936	0.884
MLS + Remarks	0.990	0.942	0.899
MLS + Redfin	0.987	0.947	0.893
MLS + Redfin + Remarks	0.989	0.947	0.907
MLS + Redfin + Images	0.988	0.949	0.898
MLS + Redfin + Remarks + Images	0.990	0.948	0.911

Multi-family: Price R^2 (Ensemble Model)			
	Training	Validation	Test
MLS	0.903	0.782	0.718
MLS + Remarks	0.956	0.841	0.801
MLS + Redfin	0.930	0.807	0.736
MLS + Redfin + Remarks	0.961	0.849	0.803
MLS + Redfin + Images	0.947	0.777	0.724
MLS + Redfin + Remarks + Images	0.967	0.837	0.800

Single-family: Price R^2 (Single Model)			
	Training	Validation	Test
(Ridge) MLS + Remarks	0.689	0.701	0.648
(XGBoost) MLS + Remarks	0.960	0.901	0.902
(LightGBM) MLS + Remarks	0.964	0.904	0.904

7 Conclusions and Possible Future Work

In summary, we have:

- Developed a method using NMF and LDA to extract topic features from remarks
- Developed a method to scrape property data and images from Redfin
- Developed a method to extract visual features from property images (the average 2048-dimensional ResNet final average pooling layer output)
- Found that both transformed remark topic features and information from Redfin (e.g. average school ratings and distances to the property as well as convenience scores) are useful features for predicting the sold price
- Found that our current method of extracting images is likely sub-optimal since it only marginally and inconsistently improved model performance

Possible future work include:

- Curate more multi-family observations to reduce overfitting and improve model generalizability.

- Develop a better method of incorporating image features. We envision building a new Convolutional Neural Network from the ground up, where images are the input and sold price is the response.
- Curating additional features from external sources to try to capture market temperature and the overall economy. We envision that doing so will improve model performance on the test set, which typically would be outside of the time period from which the training data was gathered.

8 Reference

[1]. Q.You, et al. Image-Based Appraisal of Real Estate Properties. IEEE Transactions on Multimedia, vol. 19, no. 12, pp. 2751-2759, 2017.

9 Appendix

9.1 Modeling Results for DOM

Our results for modeling DOM are summarized in the following tables. Consistent with the skewness and messy trends observed in EDA on DOM, we got fairly poor results in any of our model/feature set combinations predicting DOM. We strongly suspect our current features do not actually capture the variance in DOM. Therefore, the models are simply fitted to noise. To improve prediction on DOM, we need to find dataset that can produce more effective features.

Condo: Days on Market R^2 (Ensemble Model)			
	Training	Validation	Test
MLS	0.340	0.125	-0.029
MLS + Remarks	0.507	0.127	0.020
MLS + Redfin	0.529	0.108	-0.014
MLS + Redfin + Remarks	0.586	0.165	0.064
MLS + Redfin + Images	0.748	0.106	0.031
MLS + Redfin + Remarks + Images	0.763	0.144	0.061

Multi-family: Days on Market R^2 (Ensemble Model)			
	Training	Validation	Test
MLS	0.134	0.082	-0.053
MLS + Remarks	0.219	0.079	-0.087
MLS + Redfin	0.206	0.053	-0.131
MLS + Redfin + Remarks	0.356	0.014	-0.184
MLS + Redfin + Images	0.753	-0.031	-0.133
MLS + Redfin + Remarks + Images	0.767	0.005	-0.114

Single-family: Days on Market R^2 (Single Model)			
	Training	Validation	Test
(Ridge) MLS + Remarks	0.076	0.085	0.092
(XGBoost) MLS + Remarks	0.270	0.162	0.102
(LightGBM) MLS + Remarks	0.351	0.166	0.098