

# Predicting House Prices in King County, Washington

Eryn Blagg *eblagg@iasate.edu*  
Yiqun Jiang *yiqunj@iasate.edu*

May 2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>The Data</b>	<b>2</b>
<b>3</b>	<b>Initial Model</b>	<b>3</b>
<b>4</b>	<b>Pre-Processing</b>	<b>4</b>
<b>5</b>	<b>Model Trials</b>	<b>5</b>
<b>6</b>	<b>Final Model</b>	<b>5</b>
<b>7</b>	<b>Discussion/Further Thoughts</b>	<b>6</b>
<b>8</b>	<b>Conclusion</b>	<b>7</b>
<b>9</b>	<b>Appendices</b>	<b>7</b>

## 1 Introduction

Housing prices are often an indication of the economy and growth in an area. King County is a county located in the U.S. state of Washington, a country that contains Seattle as the largest city. The data obtained from Kaggle, contains 21,613 home sales made in the years 2014 and 2015. This data contains information about house specifications, locations and the sale price. Based on this information, our goal is to find the best prediction model for the houses' price, given the house specification.

In order to measure the accuracy of each model performed, RMSLE (root mean squared logarithmic error<sup>1</sup>) is computed and compared. Using a variety of different types of models, we worked toward finding a good prediction of house prices in King County based off a training and test set. After comparison of these models, and some complications, we find that location, number of bathrooms, square feet of living space, lot space, water view and conditions of the house are all important features predicting houses' price. It fits our common knowledge that normally house's price is relevant to the location, condition and space, and our final model reflects that knowledge. In the end, we found that the best way to predict housing prices is through a log transformed linear model, with these aspects included.

## 2 The Data

In order to find the models, we will be using a training set made up of a subset of 10000 houses from the whole 21,613 data set. Then these models were tested on a testing set made up of the remaining 11,613 cases. Throughout the models we will be using the following parameters, or subset of parameters.

id	description
property	unique identifier for a house
date	date house was sold (YYYYMMDD)
price	selling price
bedrooms	number of bedrooms
bathrooms	number of bathrooms
sqft living	square footage of house
sqft lot	lot area
floors	total floors/levels
waterfront	indicator view to a waterfront
condition	an overall condition rating
grade	overall grade per a King County grading system
sqft above	square footage apart from basement
sqft basement	square footage of basement
yr built	year of initial construction
yr renovated	year when house was renovated
zipcode	US Postal Service zipcode
lat	latitude
long	longitude
sqft living15	square footage of house in 2015
sqft lot15	lot size in 2015

There are a few things to point out here in our data. Zip code was used as a factorial variable,

---

<sup>1</sup>mathematically, the RMSLE can be found by  $\epsilon = \sqrt{\frac{1}{N} \sum_{i=1}^N (\log(\text{prediction}_i + 1) - \log(\text{actual}_i + 1))^2}$

where originally it was coded as numeric. Later we will go over some processing that we tried later, but here are the initial variables in our data set that we used in our model.

### 3 Initial Model

When deciding on how to start modeling, we decided to start with the most basic model, a linear regression. Linear regression is a widely used model in continuous variable prediction. Running a model with all of the variables other than the property id and the date the house was sold, we set up a linear model, using cross validation with 5 folds and 10 runs. Due to limited computing power, we could not more cross validation folds as we wished. After computing this initial cross validation, and applying the results, we noticed that some of our predicted prices were negative, which as prices of house go, is impossible. In order to correct for this issue, we took the absolute value of the prices to convert our analysis into a real world solution. Applying this to our test set in order to come up with an RMSLE score. This simple yet effective analysis left us with and RMSLE score of 0.33990. This was a promising start, so we tried to stay on this pathway at first.

After trying a linear model, we wanted to explore the possibility we could improve a linear model itself. We then tried a smoothing method on the linear regression first. Using "gamSpline" with a cross validation of 10 folds and 10 replications, we attempted to smooth any of the terms that may have caused issues with our first linear model. This model did not work at all, and left us with an RMSLE score of .60, which was the opposite direction in which we wanted to go. With a regression spline not working, then we decided to try other models but not confine ourselves to linear regression.

Our fist step in trying new models was to start with the list of different models laid out through the process of 502. Starting with kNN, we applied a 10 fold cross validation. Using all the variables other than "property" and "date", this cross validation resulted in a error rate of 0.501. This was higher than our simple linear model, so we continued to move down the list of possible models. Next using the same variables as kNN, we tried a decision tree. Using 10 fold cross validation with 10 replications, without setting tuning grid, which means it would gives pick-the-winner error. This model also preformed poorly, and our most accurate model continued to be the linear regression.

While these were better than our spline model, they did not beat our regression describe above. This was frustrating, but never the less we persisted. After these models, we tried other models that were introduced in the course of class. These included principal component analysis, MARS and Cubist predictions using caret, and 10 fold cross validation, with the fit control of tuning on a grid. Similar to the kNN, and the decision tree, these models also resulted in higher RMSLE scores. We did not expect that the results of these different models would be a lot worse with more complicated and advanced modeling. This was very frustrating.

After trying different models that we listed above, we thought two ways to improve the results: to separate some of the variables into similar groups, spectrs of the house, location, etc. and finding models based on each of these subgroups and trying to stack the models to see if that helps our process and our predictions. The first step was to see which of these subgroups could be important in a model. One thought was to try just using location as the explanatory variables. An important aspect of buying a house is location, location, location. With this in mind, we tried kNN on only the latitude and longitude of the house. By doing this, we hoped to see the effect of location on an analysis. Running a 10- fold cross validation with 10 replications, we got a RMSLE of .30, showing that in our modeling the location is important and a factor that wee need to keep in our final model. It was interesting to see that this model with only location performed better than the more complicated models of kNN regression spline, etc. and as well as our linear model, however we had a long way to go, in predicting.

Another method we tried, in order to get our error lower, was to boost our best result so far, with random forest and other methods. Our best model up to this point was linear regression. We fitted a random forest model for the residuals from linear regression model, then chose different  $\lambda$ , from -0.05 to 0.05, to combine residuals and original results using  $lmFitValues + \lambda * ResidualFitValues$ . While we thought that this would take our best models so far and improve them, all the outcomes were slightly worse than original predictions. With this occurrence, we needed to think of a different angle on tackling the problem. So a new approach was tried beginning with selecting, and transpose some of the variables.

model	cv error rate
linear regression	0.33990
random forest	0.38951
partial least squares	0.42182
kNN	0.5005948
decision tree	0.4034631
linear regression splines	0.6038
lat and long kNN	0.30171

The models listed above with the corresponding error rates.

## 4 Pre-Processing

After our first set of models, it seemed some variables needed to be transformed, to reduce our prediction errors. In order to do this, we pre-processed some of the variables. In this pre-processing phase, we deleted the 'property' variable since it is only used as a id number, and not useful to predict a houses' price. After we deleted this variable, we then transformed the date to be a numeric variable. Our next step in this pre-processing phase was to delete latitude and longitude since they have strong correlation with zip code. Later we used latitude and longitude rather than zip code to see if it changed our results, but for now we just used zip code. We removed these variable because having both sets of variables in our model caused severe co-linearity, and caused our prediction errors to increase.

Our last step in the pre-processing the explanatory variables in the training set was to transform the age of the house. For built year of the house and renovated year, we did some input engineering. If a house had been renovated, then the age of the house would be a lot newer than the original built date, and our knowledge of renovations suggested that if a house had been renovated in a given year, the style of that renovation would be very similar to the houses built that year, as styles are highly correlated with the times. In order to apply this thought mathematically, we created a column called house 'age' to replace the columns of `yr_built` and `yr_renovated`. This new column then represented the period from last modification of the house to 2020. This was accomplished using the formula:

$$2020 - \max(yr_{built}, yr_{renovated})$$

Among the remaining features, we treated view, zip code, waterfront as factor variables, while the rest remained numeric as the original data set suggested.

Our last step as part of the pre-processing was to transform the price of the house. As mentioned above in our initial models, some of the predictions for price were in fact negative, which is impossible. To correct for that we used the absolute value of our prices. While this fixed the problem, it is not a very good process in this model. In order to combat this issue, we used  $\log_2(price + 1)$ . We added the 1 in order to make sure our price was not NA, and is important in noticing our model. After our models were produced, we then back transformed our predictions to find the errors with the models we will go into more depth in the following sections. Once we applied these pre-processing steps to our data, we continued to look and hope for better models.

## 5 Model Trials

Once we made these changes to our variables, we decided to re-run our models in order to see if these changes had any effects on lowering our RMSLE. Just like our initial models, after our pre-processing, we applied kNN, random forest, a decision tree, etc. to our new variables. Like before, we used 10 fold cross validation, we ran these models only to get RMSLE scores of 0.18-0.34.

model	cv error rate
linear regression	0.185
linear regression with lat and long	0.186
random forest	0.286
boosted linear regression	0.1883854
Neural Network with 1 hidden layer	0.24
Neural Network with 3 hidden layers	0.202

The models listed above with the corresponding error rates.

The first step of this was to try our log transformed linear model. Applying our linear model with our price being  $\log_2(\text{price} + 1)$ , we run a 10 fold cross validation, repeated 5 times and get an RMSLE of 0.185. While this was better than our initial linear model. We first tried this model with zip code as our location variable and got an error of 0.1853, then to test the difference, we used latitude and longitude as our location variables. Switching out zip code for lat and long, under the same conditions of the log transformed linear model of a 10 fold repeated 5 times, we got a new RMSLE of 0.1869, This RMSLE value is only slightly higher than the zip code model. For this reason, we continued to use zip code for the location variable through the rest of the process of testing different models.

These log transformed linear models were a positive move in our modeling, however we wanted to try the models we did before. Since linear regression produced a better result after input engineering, so we re-ran random forest with cross validation. The pick-the-winner error rate decreases but like before, the final error rate is still larger than linear regression. For example after, we tried a random forest with our new model, with a maximum tree depth of 30, and we found that using 10 fold cross validation, we were only able to get a error score of 0.286, which is 0.1 higher than our log transformed linear model. This was the same case with neural networks. Using a neural network with 1 hidden layer, we only bettered our error slightly, getting an error of 0.24. This was only slightly improved upon by increasing the hidden layers to 3, leaving us with a error rate of 0.202. In order to implement these neural networks, we used the function "neuralnet" with a logistic smoothing function. This is only one way to apply a neural network, and may have effected our results. With more time this is one aspect of modeling that we could go into more depth about.

It seemed single models, even more complex ones were not very good at out-perform the current linear regression model, so we tried to boosting our best result. We fitted a random forest model for the residuals. With  $\lambda$  equals -0.1, -0.05, 0.05, 0.1, the prediction errors of  $lmFitValues + \lambda * ResidualFitValues$  are 0.1914708, 0.1961755, 0.1883854, 0.1907908 respectively, which is higher than original error rates. We expanded the range of  $\lambda$ , but still, no results get better than original ones. By this time, it was the end of the allotted time, so we were unable to try more models.

## 6 Final Model

Our final model is linear regression with input engineering described in part 4 pre-processing (using the log transformed price and the age of a house, with the zipcode). The cross validation

error rate is 0.1816125 while the test set error rate 0.18564. And the significance of the parameters showed that except 6 levels of zip code and years, square footage of house in 2015, lot size in 2015, all variables are important, which fits our expectations-the most important factors of house price is location, square footage of living space and the house condition.

## 7 Discussion/Further Thoughts

Throughout this modeling processes we tried a lot of different models without much success. Even after transformations and running a wide variety of cross validations with different models. However, there was still not a lot of progress made after our linear regression model. We found this very frustrating.

Due to our limited computer computing, we had difficulty running some models. So while we had ideas to do other models, due to their immensely long computing times, we were not able to compute some of them. For example, we attempted to do a Gaussian Kernel SVGM, however, it took over 48 hours to run, after which our computer was tired and quit. This happened with other models as well, and was quite frustrating to not get results. Furthermore, there were some ideas that resulted in computing errors. We tried to stack a neural network on location, using latitude and longitude and a neural network using the specs of a house. However, the most effective package for this is no longer is compatible with the newest R. This made stacking these neural networks very difficult, and with the computer that we used, were not able to accomplish.

What also made this process difficult was Kaggle itself. During the process of this competition, while both of us had accounts, we were not able to merge our accounts. This caused us to have to log into one account on two different computers. Kaggle apparently had difficulties with this. After our first 4 submissions, we got locked out of Kaggle for cheating. This was just another set back in our process. It took a week of on hold with Kaggle to fix the situation, and get the account to be unlocked. This was made further difficult working at a distance. We had a difficult time working together on the models, and tried to share each others codes. In order to do this, we created a git repository, but it was still difficult not to communicate coding ideas. We used Facetime to communicate weekly, tho it was still a difficult and a new challenge for both of us.

As we mentioned above, throughout this modeling process, we were not able to beat our linear regression. This was beyond frustrating. While many of our models got close, it was disheartening to see the other groups surpass us on our error rate. Even after trying many different models, the fact that a simple model was unbeatable was surprising. We expected at the beginning of this process that we would need to transform many models and combined a lot of models to make the best prediction, but in the end the best model was one of our simplest. It was a annoying fact that discouraged us when looking at model. We tried all these complicated coding measures and different combinations of model ideas that were taught through out 502, only to get higher error in our results over and over again. In some aspect this makes sense, as housing price is relatively structured, in a similar location, but it definitely surprised us on how a simple model was so accurate.

Although we came up with a decently good model at the end, we feel like there is still some different models that could be implemented to better our final score. While we looked at some boosting and stacking, there is a lot more in this area that can be explored. We were only able to find one boosting model that was anywhere close to our log transformed linear model. However, we ran out of time to try everything, and there is still a lot of stacked models that we were not able to get to do to either errors in computing that were not fixed, large computing times our laptops could not handle, or just running out of time in the semester.

Furthermore, there was a lot of aspects of the pre-processing that we did not explore. There is a lot of variables that we could apply different functions to, to better explain house prices. While we transformed the age of the house, we did not spend a lot of time looking at other possible ways of combining and transforming our other variables. While we did look into the co-linearity of location variables, we did not explore the other variables in depth, which may have weakened our final model predictions. With further analysis, this is another area that we would have liked to explore more of. For example, having a waterfront view is highly dependent on the location of a house, however we never looked into that correlation. Both of us are decently competitive people, so we found hitting a plateau in the error values, and getting surpassed by our classmates annoying to say the least, and wish we were able to implement more of these ideas, to maybe lower our value.

## 8 Conclusion

Through this process we tried a lot of different models to a wide range of results. We had some success with a few models, while very bad results with others. Throughout this process, of pre-processing, modeling and testing we learned a lot. For one, based on error estimation comparison standard, we may need to modify our predictors a little bit to fit the requirement. With this modification, input engineering is indispensable and real data analysis may require background knowledge to pre-process the data, like deleting useless features and combining co-linear features. Finally, our best result is obtained from linear regression, almost the most basic model for prediction. It seems that sometimes less is more. When it comes to house prices in King county, Washington, house prices are a combination of location, number of bathrooms, square feet of living space, lot space, water view and conditions of the house. While this seems like basic knowledge, it was reinforced by our log transformed linear model. House prices are an ever changing phenomenon that includes a lot of different variables, but with a little pre-processing and some computing power, it is possible to predict a house price, well, at least with an error rate of 0.185.

## 9 Appendices

## Stat 502 Class Project Weekly Updates

**Competition:** King County real estate price prediction competition

**Team Kaggle Name (if Relevant):** \_\_\_\_\_ Y \_\_\_\_\_

**Team Members:** \_\_\_\_\_ Eryn Blagg, \_\_ Yiqun Jiang \_\_\_\_\_

---

### **Week 10 (March 23-27) Summary of Activity and Progress**

Preprocessing: we transformed waterfront and zip code to be factors and delete date and property.

Model: This week we tried to use a simple linear regression model to predict house price but one problem we meet is with this method, we may get negative predictions. It's commonsense that house price should be positive so we modify our model a little bit to be absolute linear regression model. We forced the prediction to be positive to get more reasonable result.

Future plan: one thing we can do to improve our results I think is input engineering. Besides we would try more models.

### **Week 11 (March 30-April 3) Summary of Activity and Progress**

Now we tried using glam to use as a smoothing to some of the variables. Our goal with this is to try and get rid of the negative values that we have in our first attempt at a model. This makes sense as it would be the next logical step in keeping our original model to be a linear model. For our results, we no longer have any negative prices on our houses, but we still have a relatively high error rate, but its nice to see that we no longer have negative prices in our models prediction.

### **Week 12 (April 6-10) Summary of Activity and Progress**

We used  $\log(\text{price}+1)$  to be our prediction object this week and for the preprocessing part, besides what we did before, we also deleted yr\_built and yr\_renovated columns and derived a



new column from  $2020 - \max(\text{yr\_built}, \text{yr\_renovated})$  representing age of the house. And we applied linear regression on the modified data set. After that, we applied random forest and kNN on it too but get larger errors. So our next step may be try some other methods and try to combine some results together to see whether there are some improvements.

### **Week 13 (April 13-17) Summary of Activity and Progress**

At the beginning of this week we got kicked off of kaggle, so a lot of time has been spent on hold with them. But that has now been resolved.

This week we seemed to have hit a plateau on our error. After we combined the age of the house last week, we re-ran our models of knn, random forest, elastic net, etc. to compare to our linear model that we have as our current lowest estimate on our predictions on Kaggle. These models are coming up with estimates are in the range of  $[0.22, 0.3]$ , which is much higher than our 0.18. So, now we are taking a new approach. At the end of this week we have been starting to separate some of the variables into similar groups: specs of the house, location, etc. and finding models based on each of these subgroups and trying to stack the models to see if that helps our process and our predictions.

### **Week 14 (April 20-24) Summary of Activity and Progress**

we tried to boosting our best result. We fitted a random forest model for the residuals. With  $\lambda$  equals -0.1, -0.05, 0.05, 0.1, the prediction errors of  $\text{lm Fit Values} + \lambda * \text{Residual Fit Values}$  are 0.1914708, 0.1961755, 0.1883854, 0.1907908 respectively, which is higher than original error rates. We expanded the range of  $\lambda$ , but still, no results get better than original ones.

### **Week 15 (April 27-May 1) Summary of Activity and Progress**

This week we tried to add date column to the regression model to see whether it helps to improve the results and we found the result increased a little bit from 0.18681 to 0.18583. And we also tried to put longitude and latitude back but error rate increases. It seemed that current model is the best we could do for linear regression. The stack model failed due to limited computing power. We kind of got stuck at this point.

### **Week 16 (May 4-May 8) (Finals Week) Summary of Activity and Progress**

This week we summarized all results we have and wrote the report based on that. We made a PowerPoint and recorded a video presentation.

## Stat 502 Class Project Kaggle Scores Log

Team Kaggle Name: \_\_\_\_\_Y\_\_\_\_\_

Team Members: \_\_\_\_\_Eryn Blagg, Yiqun Jiang\_\_\_\_\_

---

---

### Week 10 (March 23-27)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.35071\_\_\_\_\_

Date and Time: \_\_\_\_\_March 30\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.36238\_\_\_\_\_

Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

1	Y		0.35071	1	3d
---	---	--	---------	---	----

Date and Time of the Screenshot: \_\_\_\_\_March 30\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_3\_\_\_\_\_

### Week 11 (March 30-April 3)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.18681\_\_\_\_\_

Date and Time: \_\_\_\_\_April 6\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.1881555\_\_\_\_\_

Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

3	Y		0.18681	5	3d
---	---	---	---------	---	----

Date and Time of the Screenshot: \_\_\_\_\_April 6\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_8\_\_\_\_\_

(this week has been missing since our account is blocked. We'll make up later.)

### Week 12 (April 6-10)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.18681\_\_\_\_\_

Date and Time: \_\_\_\_\_April 6\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.1881555\_\_\_\_\_

Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

3	Y		0.18681	5	3d
---	---	---	---------	---	----

Date and Time of the Screenshot: \_\_\_\_\_April 9\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_8\_\_\_\_\_


### Week 13 (April 13-17)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.18681\_\_\_\_\_

Date and Time: \_\_\_\_\_04/16/2020 at 13:48\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.1881555\_\_\_\_\_

Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

4	Y		0.18681	5	10d
---	---	---	---------	---	-----

Date and Time of the Screenshot: 04/16/2020 at 13:48\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_8\_\_\_\_\_


#### Week 14 (April 20-24)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.18681\_\_\_\_\_

Date and Time: \_\_\_\_\_April 6\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.1881555\_\_\_\_\_

#### Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

6	Y		0.18681	6	3h
---	---	---	---------	---	----

Date and Time of the Screenshot: \_\_\_\_\_April 24\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_9\_\_\_\_\_


#### Week 15 (April 27-May 1)

Team's Best Result/Public Score Thus Far: \_\_\_\_\_0.18681\_\_\_\_\_

Date and Time: \_\_\_\_\_April 6\_\_\_\_\_

Team's CV Result for the Submission: \_\_\_\_\_0.1881555\_\_\_\_\_

#### Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)

6	Y		0.18681	6	3h
---	---	---	---------	---	----

Date and Time of the Screenshot: \_\_\_\_\_April 30\_\_\_\_\_

Number of Teams Registered at the Time of the Screenshot: \_\_\_\_\_9\_\_\_\_\_


**Week 16 (May 4-May 8) (Finals Week)**

**Team's Best Result/Public Score Thus Far:** \_\_\_\_\_ 0.18564 \_\_\_\_\_

**Date and Time:** \_\_\_\_\_ May 3 \_\_\_\_\_

**Team's CV Result for the Submission:** \_\_\_\_\_ 0.1816125 \_\_\_\_\_

**Screenshot of the Team's Best Score of the Week (Include Team Ranking and Score)**

6	Y		0.18564	11	3d
---	---	--	---------	----	----

**Date and Time of the Screenshot:** \_\_\_\_\_ May\_6 \_\_\_\_\_

**Number of Teams Registered at the Time of the Screenshot:** \_\_\_\_\_ 8 \_\_\_\_\_