Prediction of House Prices In King County, Washington Using Regression Models

Omar Choudhry

oachoud2

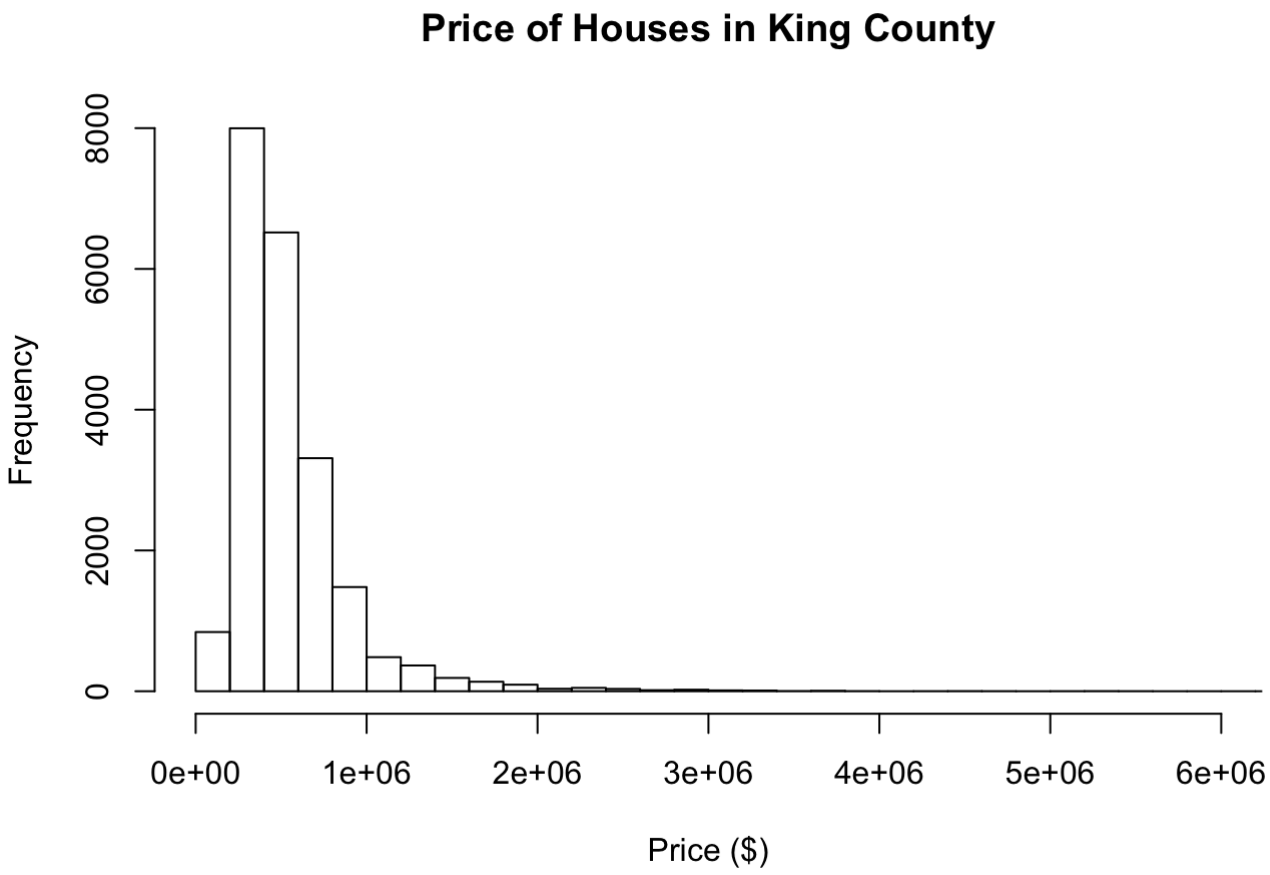
Abstract

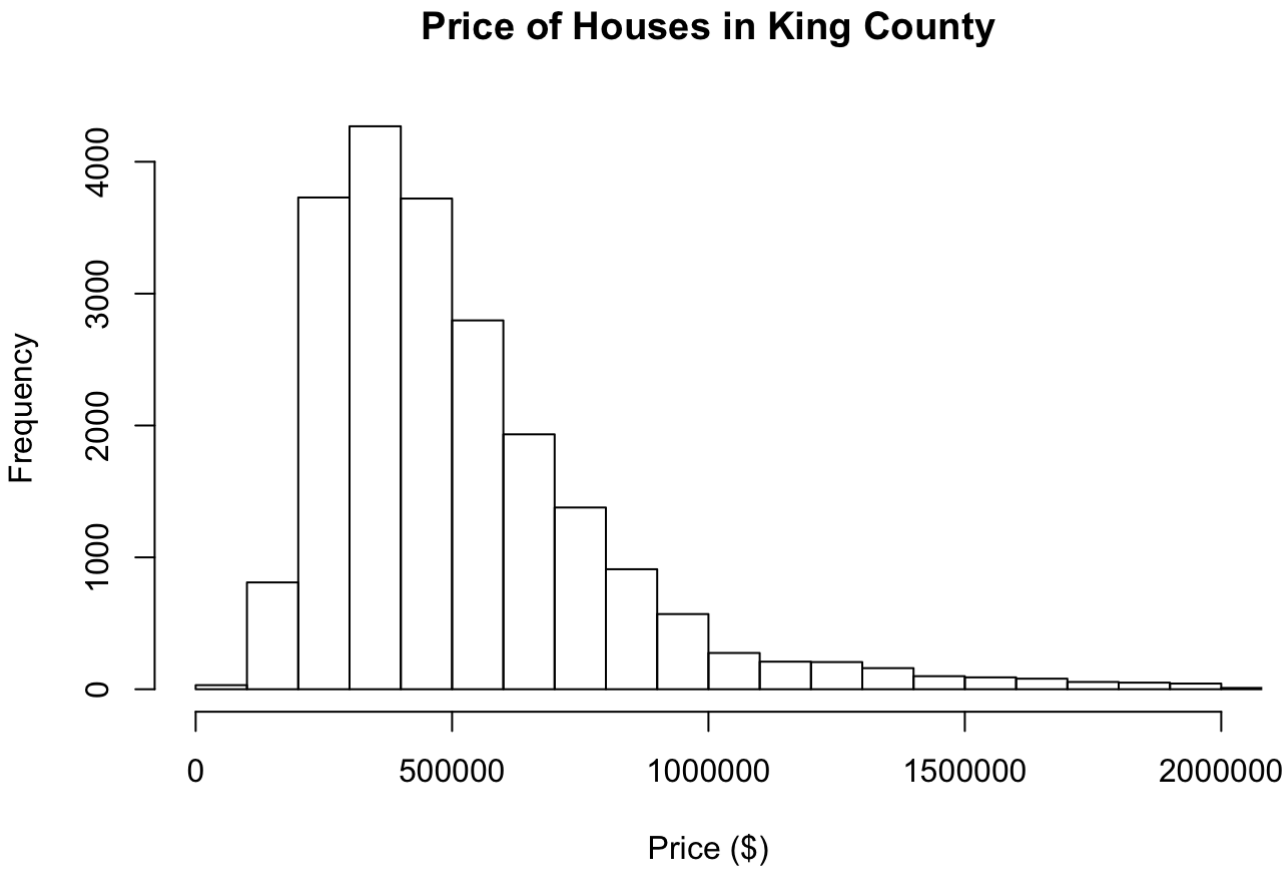
In this project I will be using a dataset that describes houses in King County, Washington. I am attempting to predict the price of the house using various attributes of the house such as square footage, condition, and age. This is useful because it will be able to determine wether or not a house is listed at an appropriate price. I fit numerous Linear and KNN Regression models to find the model that best predicts price. Using the Test RMSE I found that a linear interaction model did the best job in predicting the price of a house in King County. This gave me a good model to apply to my problem, being able to list and buy houses at fair prices, so that both parties are sure they are getting a good deal.

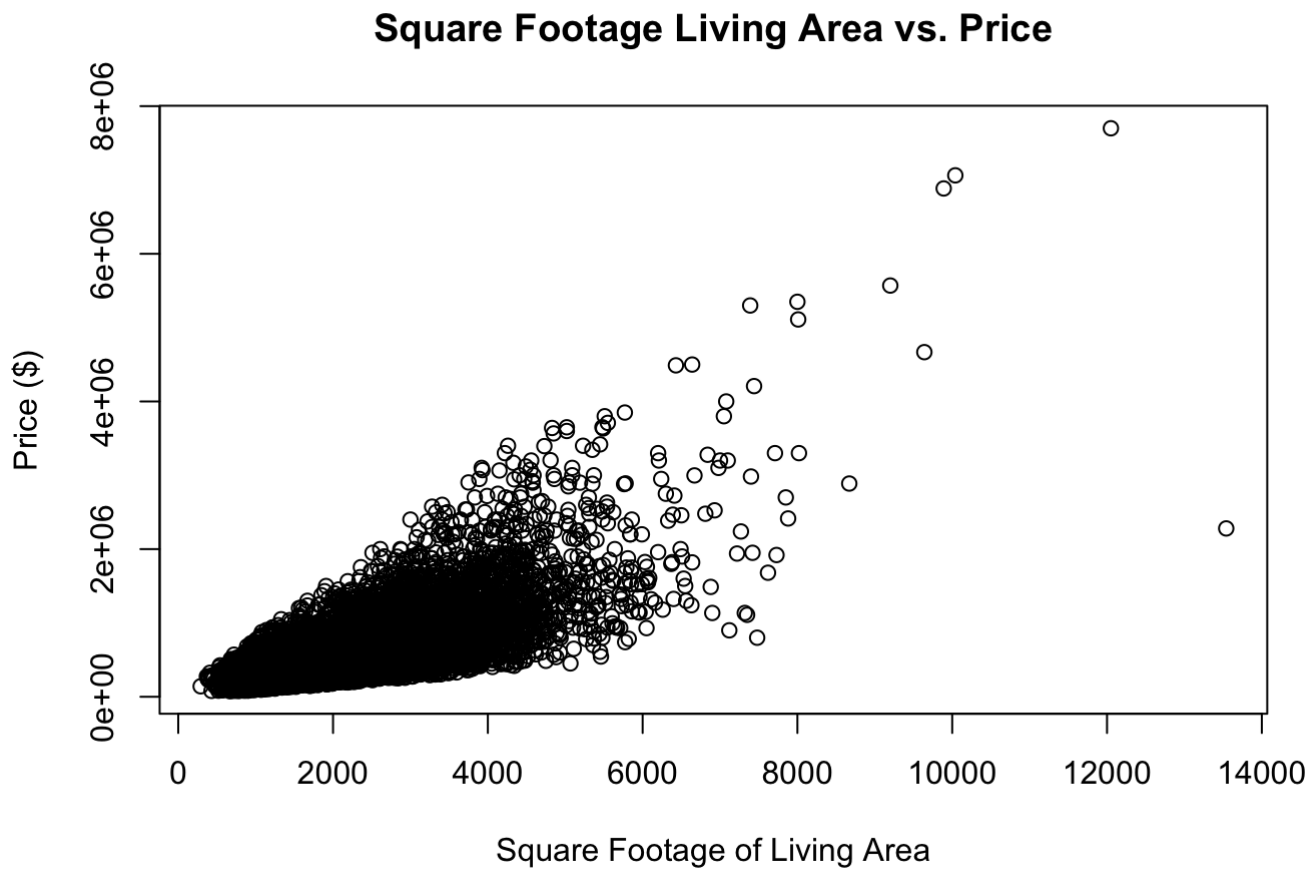
Introduction

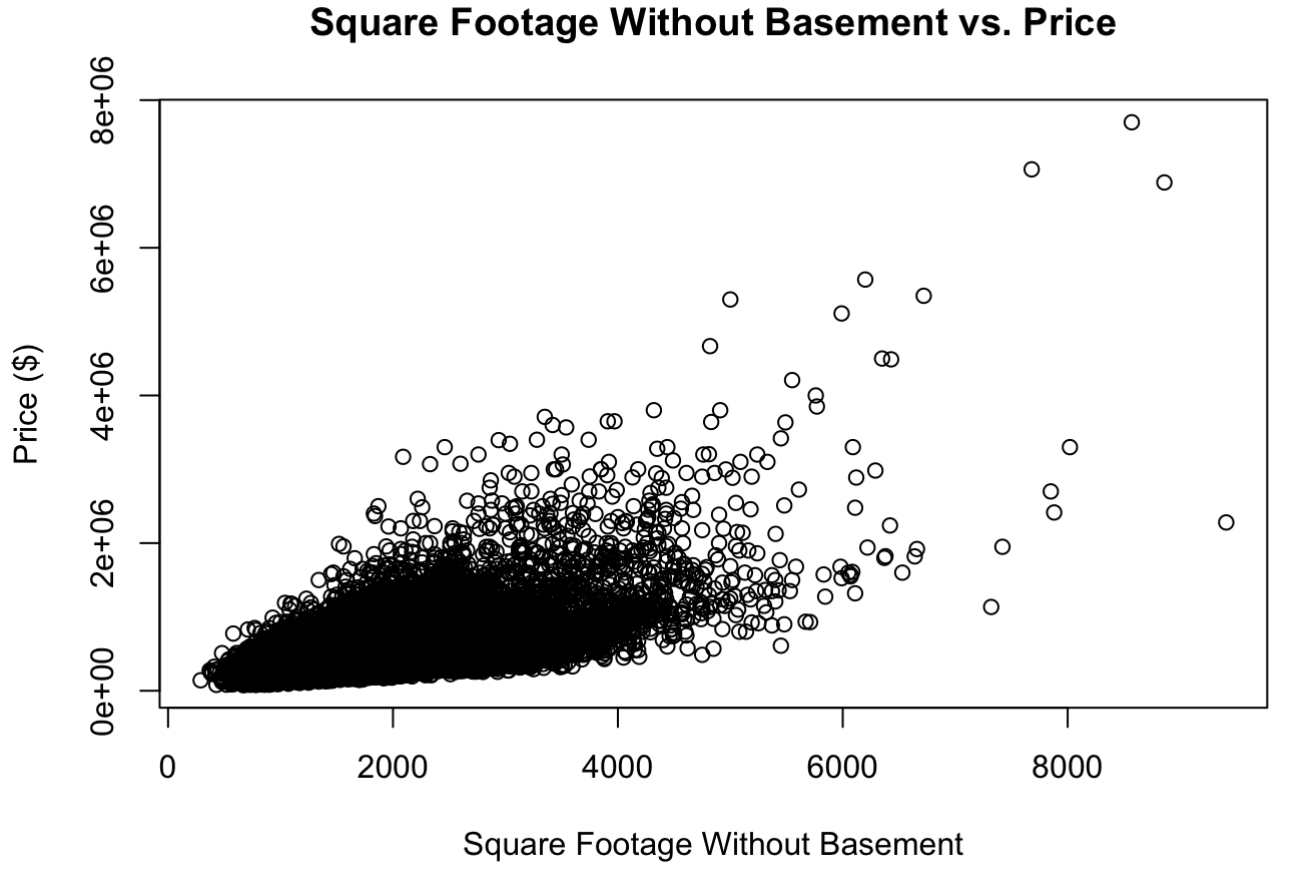
The dataset I worked with for this project described house sales in King County, Washington. King County is a relatively large county in the state of Washington, and it contains Seattle, the state's largest city. The purpose of my analysis is to predict the price of houses in this county based on attributes such as square footage, condition, year built, and zip code to name a few. If I am able to create a model that does a good job with predicting price I will be able to use this model to find a good price for listing a house on the market, as well as determine wether or not a house is a good deal for the price it is being offered at. The data I am using has 19 features and 21k observations. In other words, we have 21k houses in our dataset, each one having 19 attributes that describe it. I have provided a full description of these 19 attributes in a data dictionary in the Appendix section.

Below is some exploratory analysis of my data set.


Price of Houses in King County
Frequency
0
2000
4000
6000
8000
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
Price ($)


Price of Houses in King County
Frequency
0
1000
2000
3000
4000
0
500000
1000000
1500000
2000000
Price ($)





Explanation

In the first histogram we see how skewed the distribution of prices is. We should be mindful of how expensive houses (outliers) will affect our analysis. In the 2nd histogram we see that most houses are between 300k-400k, which will help us gauge plausibility when making predictions. The 3rd and 4th plots show us that we may need to be aware of interaction between square footage variables.

Methods

In order to find the model that best predicts price based on the house attributes I trained both linear regression, and K-Nearest Neighbor Models. This involved many steps that I have broken down below. The goal of training multiple models of each type was to find the model with the lowest test Root Mean Squared Error. This model will do the best job at predicting the price of a house in King County based on the attributes we included.

Data

For purposed of my analysis I decided to drop both the ID and Date variables, as they do not directly relate to how much a house costs. There were a few other variables in question, so I did some testing when training models to see if I should alter or drop them. The variables in question included zip code, latitude, longitude, and square footage variables. Specifically, I wanted to check if zip code was better as a categorical variable, if latitude and longitude are significant at all, and if the square footage variables have any interaction that would skew the model.

```
#Reading Data
data = read.csv("https://daviddalpiaz.github.io/stat432sp18/projects/kc_house_data.csv")
#drop ID and Date Variables
data = data[-1:-2]
```

Variable	Issue	Test	Solution
Zipcode	Not Categorical	Refactor and check RMSE	Refactor in Dataset
Lat/Long	May not be Significant	Check Significance via Summary	Do Not Remove
Square Footage	Possible Interactions	Check RMSE with Interactions	Fit With Interaction

After finalizing the dataset I will be using to train the models, I split the data into a training set and a testing set. I used 80 percent of the data to train my models, and 20 percent to test them.

```
#Test-Train Split
set.seed(1)
dataidx  = sample(nrow(data), size = trunc(0.80 * nrow(data)))
trn = data[dataidx, ]
tst = data[-dataidx, ]
```

Models

I built 21 models in total 5 of them being linear, and 16 being K nearest-neighbor regression models. I will provide a detailed list of these in the following 2 sections. For every model I will obtain the test RMSE and use that to evaluate the accuracy.

Linear

```
#Linear Models
#solutions for zipcode
lmod1 = lm(price ~ .,  data = trn)
lmod2 = lm(price ~ .-zipcode + as.factor(zipcode),  data = trn)
lmod3 = lm(price ~ .-zipcode,  data = trn)
```

```
lmod4 = lm(price ~ . + (sqft_living*sqft_lot) + (sqft_living*sqft_above)+ (sqft_living*sqft_base
ment)+ (sqft_living*sqft_living15)+ (sqft_living*sqft_lot15)+ (sqft_lot15*sqft_lot) + (sqft_lot*
sqft_above) + (sqft_lot*sqft_basement),  data = trn)
null=lm(price~1, data=trn)
lmod5 = step(null, scope = list(upper=lmod1), data=trn, direction="both")
```

Models Fitted Above:

1. Full Additive Model
2. Full Additive Model with Zipcode Factored
3. Full Additive Model -Zipcode
4. Square Foot Interaction Model
5. Stepwise Model

K-Nearest Neighbors

```
unscaledKNN = train(
  price ~ .,
  data = trn,
  trControl = trainControl(method = "cv", number = 5),
  method = "knn",
  tuneGrid = expand.grid(k = c(1, 5, 10, 15, 20, 25, 30, 35))
)

scaledKNN = train(
  price ~ .,
  data = trn,
  trControl = trainControl(method = "cv", number = 5),
  preProcess = c("center", "scale"),
  method = "knn",
  tuneGrid = expand.grid(k = c(1, 5, 10, 15, 20, 25, 30, 35))
)
```

For K-Nearest Neighbors I utilized 2 types of models:

1. KNN with no pre-processing
2. KNN where variables are centered and scaled (Mean=0 and Variance=1)

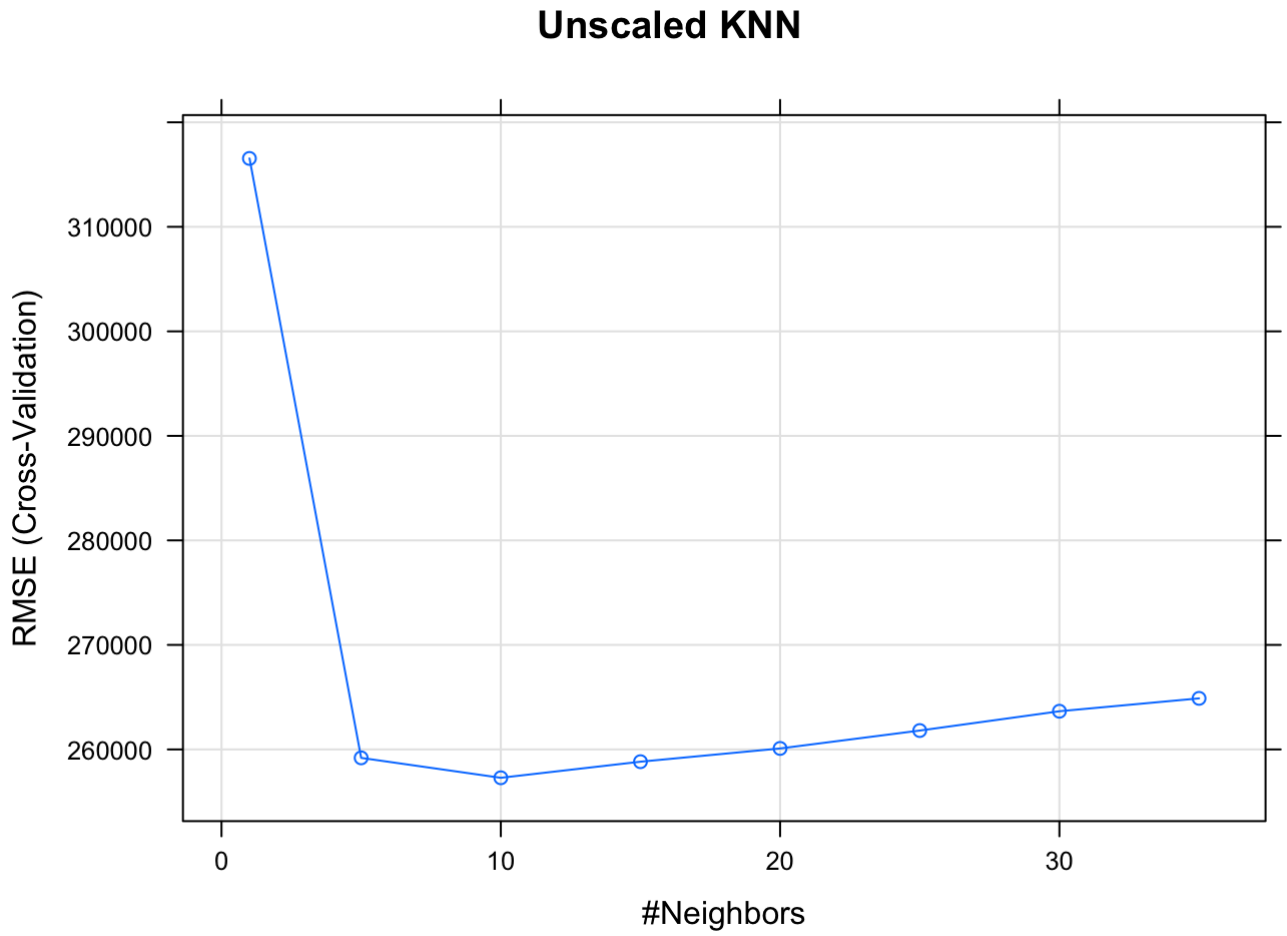
For both methods I used 8 values for the k parameter (1, 5, 10, 15, 20, 25, 30, 35). This resulted in 16 KNN Models.

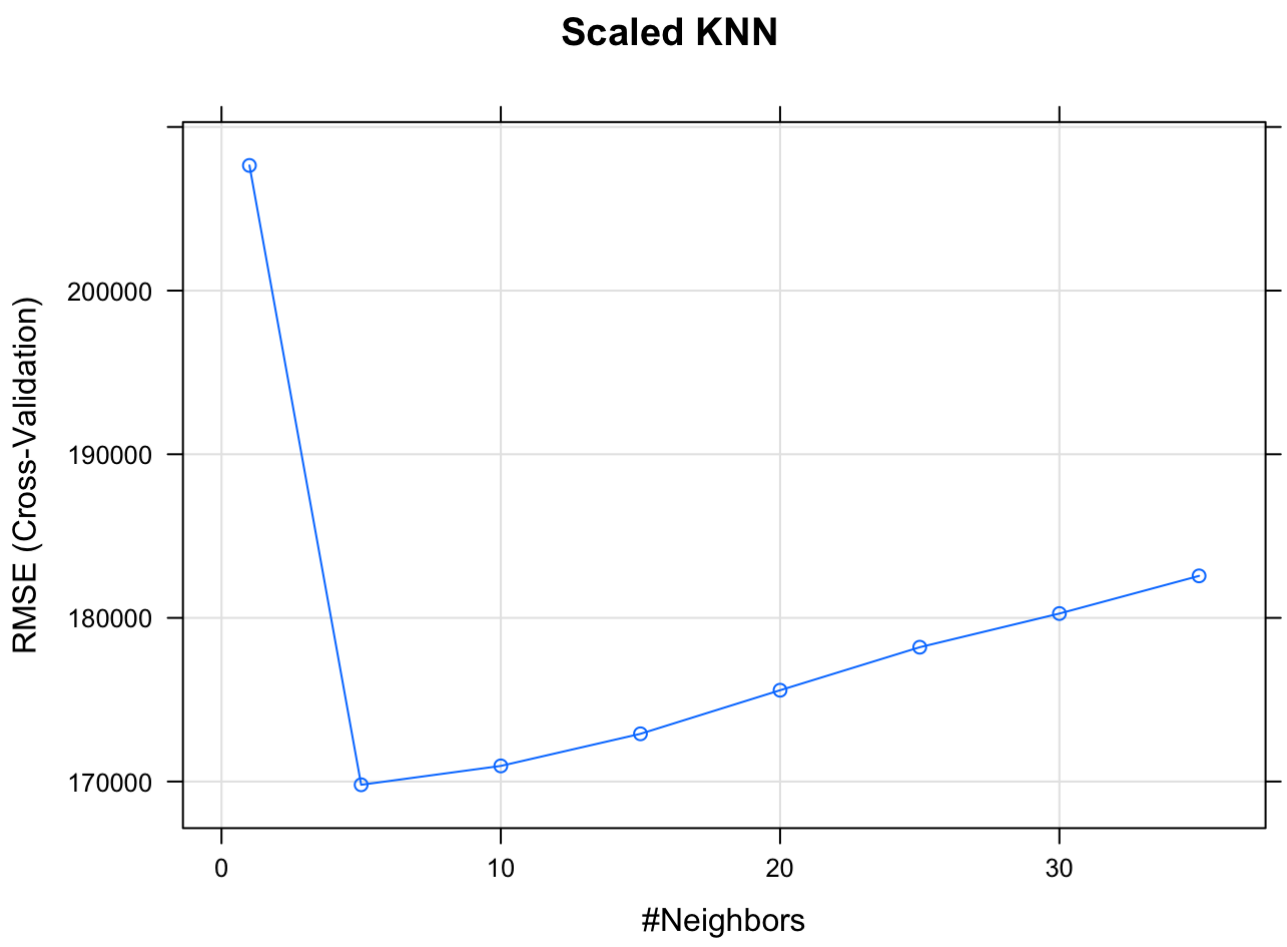
Results

After determining the Test RMSE of the 21 models, and comparing them, we are able to narrow it down to a single best model for predicting house price from the included attributes.

Model	Train RMSE	Test RMSE
Full Additive Model	200793	203096
Full Additive Model w Zipcode Factored	159749	165437

Model	Train RMSE	Test RMSE
Full Additive Model -Zipcode	202152	204890
Square Foot Interaction Model	146459	153684
Stepwise Model	159749	165437


Unscaled KNN
RMSE (Cross-Validation)
310000
300000
290000
280000
270000
260000
0
10
20
30
#Neighbors


Scaled KNN
RMSE (Cross-Validation)
200000
190000
180000
170000
0
10
20
30
#Neighbors

Method	CV RMSE	Test RMSE
Unscaled KNN	257287	259768
Scaled KNN	169801	164444

By looking at the tables we see that the Square Foot Interaction Linear model has the lowest Test RMSE. This means it does the best job at predicting House Price from the 16 predictor attributes we leveraged.

Discussion

Because we have determined that the Square Foot Interaction Linear Model is best for predicting house prices in King County, Washington, we can deduce a few things.

1. The Zipcode that the house is located in should be treated as a category, and not as a numeric value
2. The square footage of the living space as well as the square footage of the lot and basement are very correlated, and accounting for their interactions makes our model much more accurate.

Final Model:

```
lmod4
```

```
##
## Call:
## lm(formula = price ~ . + (sqft_living * sqft_lot) + (sqft_living *
##     sqft_above) + (sqft_living * sqft_basement) + (sqft_living *
##     sqft_living15) + (sqft_living * sqft_lot15) + (sqft_lot15 *
##     sqft_lot) + (sqft_lot * sqft_above) + (sqft_lot * sqft_basement),
##     data = trn)
##
## Coefficients:
##               (Intercept)                   bedrooms
##                -2.944e+07                  -7.299e+03
##                 bathrooms                sqft_living
##                 2.784e+04                  -7.677e+01
##                  sqft_lot                      floors
##                 6.297e-01                  -3.059e+04
##                waterfront                        view
##                 5.905e+05                   5.160e+04
##                 condition                       grade
##                 3.216e+04                   6.271e+04
##                sqft_above             sqft_basement
##                 4.841e+01                         NA
##                  yr_built              yr_renovated
##                -5.512e+02                   2.171e+01
##             zipcode98002              zipcode98003
##                 2.703e+04                  -2.859e+04
##             zipcode98004              zipcode98005
##                 7.046e+05                   2.782e+05
##             zipcode98006              zipcode98007
##                 2.224e+05                   2.218e+05
##             zipcode98008              zipcode98010
##                 2.241e+05                   1.050e+05
##             zipcode98011              zipcode98014
##                 7.807e+04                   1.307e+05
##             zipcode98019              zipcode98022
##                 8.506e+04                   2.888e+04
##             zipcode98023              zipcode98024
##                -5.616e+04                   1.920e+05
##             zipcode98027              zipcode98028
##                 1.763e+05                   6.472e+04
##             zipcode98029              zipcode98030
##                 2.293e+05                   1.627e+04
##             zipcode98031              zipcode98032
##                 1.643e+04                  -2.249e+04
##             zipcode98033              zipcode98034
##                 3.187e+05                   1.467e+05
##             zipcode98038              zipcode98039
##                 7.677e+04                   1.156e+06
##             zipcode98040              zipcode98042
##                 4.501e+05                   2.402e+04
##             zipcode98045              zipcode98052
##                 1.655e+05                   2.120e+05
##             zipcode98053              zipcode98055
##                 1.927e+05                   3.603e+04
##             zipcode98056              zipcode98058
##                 7.958e+04                   3.116e+04
##             zipcode98059              zipcode98065
##                 7.685e+04                   1.347e+05
```

```
##              zipcode98070                zipcode98072
##                -7.950e+04                   1.183e+05
##              zipcode98074                zipcode98075
##                 1.716e+05                   1.700e+05
##              zipcode98077                zipcode98092
##                 9.572e+04                  -1.165e+04
##              zipcode98102                zipcode98103
##                 4.214e+05                   2.489e+05
##              zipcode98105                zipcode98106
##                 4.012e+05                   7.207e+04
##              zipcode98107                zipcode98108
##                 2.502e+05                   8.055e+04
##              zipcode98109                zipcode98112
##                 4.275e+05                   5.450e+05
##              zipcode98115                zipcode98116
##                 2.646e+05                   2.242e+05
##              zipcode98117                zipcode98118
##                 2.309e+05                   1.246e+05
##              zipcode98119                zipcode98122
##                 4.060e+05                   2.766e+05
##              zipcode98125                zipcode98126
##                 1.333e+05                   1.224e+05
##              zipcode98133                zipcode98136
##                 7.779e+04                   1.819e+05
##              zipcode98144                zipcode98146
##                 2.208e+05                   5.756e+04
##              zipcode98148                zipcode98155
##                 3.419e+04                   6.292e+04
##              zipcode98166                zipcode98168
##                 2.084e+04                   3.047e+04
##              zipcode98177                zipcode98178
##                 1.420e+05                   1.182e+04
##              zipcode98188                zipcode98198
##                 4.321e+03                  -1.939e+04
##              zipcode98199                         lat
##                 3.166e+05                   1.469e+05
##                      long                sqft_living15
##                -1.897e+05                   -8.393e+00
##                sqft_lot15       sqft_living:sqft_lot
##                 1.019e+00                   -6.840e-04
##    sqft_living:sqft_above  sqft_living:sqft_basement
##                 3.204e-02                   3.274e-02
## sqft_living:sqft_living15     sqft_living:sqft_lot15
##                 1.402e-02                   -4.417e-04
##       sqft_lot:sqft_lot15        sqft_lot:sqft_above
##                 1.309e-06                   5.353e-04
##    sqft_lot:sqft_basement
##                        NA
```

Each of the coefficients listed above describes how much an attribute of a house should be multiplied by when making price predictions. For example the coefficient for sqft_lot is 0.6297. This means that the square footage of the lot of the given house we are making a prediction for will be multiplied by this value, and then added to the sum of the rest of the coefficients times their corresponding attributes. This summation of (coefficients * attributes) is what determines the price of the house.

From this model I could take a new house that was listed on the market in King County, gather the attributes that are apart of my model, plug them in, and return a predicted price for that house. If I were in the market to buy a house, this would be very helpful in deciding whether or not I am getting a good deal. If I am a realtor in the market, this model would be helpful in picking starting price for the house.

Appendix

Additional Code

```
library(caret)
library(randomForest)
library(tidyverse)
library(knitr)
library(kableExtra)

calc_rmse = function(actual, predicted) {
  sqrt(mean((actual - predicted) ^ 2))
}

get_best_result = function(caret_fit) {
  best = which(rownames(caret_fit$results) == rownames(caret_fit$bestTune))
  best_result = caret_fit$results[best, ]
  rownames(best_result) = NULL
  best_result
}

modlist1 = list(lmod1,lmod2,lmod3)
trnpred1 = lapply(modlist1, predict, newdata = trn)
tstpred1 = lapply(modlist1, predict, newdata = tst)

# get RMSEs
trnrmse1 = sapply(trnpred1, calc_rmse, actual = trn$price)
tstrmse1 = sapply(tstpred1, calc_rmse, actual = tst$price)

modlist2 = list(lmod4,lmod5)
trnpred2 = lapply(modlist2, predict, newdata = trn)
tstpred2 = lapply(modlist2, predict, newdata = tst)

# get RMSEs
trnrsme2 = sapply(trnpred2, calc_rmse, actual = trn$price)
tstrmse2 = sapply(tstpred2, calc_rmse, actual = tst$price)
```

Zipcode Refactoring

Model	Train RMSE	Test RMSE
Full Model	200793	203096
factor Zipcode	159749	165437
Without Zipcode	202152	204890

We see that factoring zipcode is very useful in lowering RMSE.

Lat and Long are Significant

```
summary(lmod1)
```

```
##
## Call:
## lm(formula = price ~ ., data = trn)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1243444   -98461    -9726    77608  4376540
##
## Coefficients: (1 not defined because of singularities)
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.873e+06  3.283e+06   1.484  0.13778
## bedrooms      -3.455e+04  2.090e+03 -16.530  < 2e-16 ***
## bathrooms      3.833e+04  3.620e+03  10.591  < 2e-16 ***
## sqft_living    1.457e+02  4.903e+00  29.722  < 2e-16 ***
## sqft_lot       1.748e-01  5.463e-02   3.200  0.00138 **
## floors         3.596e+03  4.013e+03   0.896  0.37022
## waterfront     5.327e+05  1.970e+04  27.047  < 2e-16 ***
## view           5.621e+04  2.391e+03  23.508  < 2e-16 ***
## condition      2.756e+04  2.626e+03  10.496  < 2e-16 ***
## grade          9.437e+04  2.399e+03  39.346  < 2e-16 ***
## sqft_above     3.308e+01  4.871e+00   6.792 1.14e-11 ***
## sqft_basement         NA         NA      NA       NA
## yr_built      -2.522e+03  8.110e+01 -31.094  < 2e-16 ***
## yr_renovated   1.990e+01  4.075e+00   4.883 1.05e-06 ***
## zipcode       -5.646e+02  3.685e+01 -15.320  < 2e-16 ***
## lat            6.016e+05  1.198e+04  50.218  < 2e-16 ***
## long          -2.142e+05  1.478e+04 -14.493  < 2e-16 ***
## sqft_living15  2.522e+01  3.862e+00   6.531 6.73e-11 ***
## sqft_lot15    -3.928e-01  8.232e-02  -4.772 1.84e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200900 on 17272 degrees of freedom
## Multiple R-squared:  0.6955, Adjusted R-squared:  0.6952
## F-statistic:  2321 on 17 and 17272 DF,  p-value: < 2.2e-16
```

We see that latitude and longitude are significant variables.