# Data Analysis Project

Miranda Yang

2025-10-24

**Project - Due Friday, Oct. 24th by midnight to Gradescope**

## Practicing Academic Integrity

If you worked with others or used resources outside of provided course material, such as Generative AI (anything besides our textbook, course materials in the repo, labs, R help menu) to help complete this assignment, please acknowledge them below using a bulleted list. Remember that according to our class framework, if you engage with Generative AI tools, the tool and date of use must be listed here, and then a complete record of the chat history must be appended to the assignment.

*I acknowledge the following individuals with whom I worked on this assignment:*

Name(s) and how they assisted you

- Chelsea Wang - code error on train function and confusion matrix

*I used the following sources to help complete this assignment:*

Source(s) and where used in your submission

- https://www.kaggle.com/datasets/swathiachath/kc-housesales-data
- shape file: https://ofm.wa.gov/washington-data-research/population-demographics/gis-data
- https://gis-kingcounty.opendata.arcgis.com/datasets/kingcounty::district-codes-dstcode-area/about
- plotting multiple ggplot in one plot: https://stackoverflow.com/questions/49183067/trying-to-make-a-list-of-ggplot-objects-in-a-for-loop-all-items-in-list-are-wri
- converting data frame col from numeric to categorical: https://stackoverflow.com/questions/9251326/conv data-frame-column-format-from-character-to-factor
- https://stackoverflow.com/questions/2288485/how-to-convert-a-data-frame-column-to-numeric-type

- https://stackoverflow.com/questions/33180058/coerce-multiple-columns-to-factors-at-once
- diagrams: https://bookdown.org/yihui/rmarkdown-cookbook/diagrams.html
- not include caption under figures: https://stackoverflow.com/questions/38514954/removing-figure-text-in-rmarkdown, https://tex.stackexchange.com/questions/733949/figures-without-captions
- tables: https://rfortherestofus.com/2019/11/how-to-make-beautiful-tables-in-r, https://r-graph-gallery.com/package/gtextras.html, https://cran.r-project.org/web/packages/kableExtra/ https://www.littlemissdata.com/blog/prettytables
- removing x labels and tick marks: https://stackoverflow.com/questions/35090883/remove-all-of-x-axis-labels-in-ggplot
- degree symbol in R: https://stackoverflow.com/questions/75848494/add-degree-symbol-to-axis-labels-in-ggplot2
- increase size of points in legend but not on graph: https://stackoverflow.com/questions/20415963/how-to-increase-the-size-of-points-in-legend-of-ggplot2

## Instructions for your submission

Your submission should include the following sections (along with all code necessary to reproduce your results):

- Introduction and Exploratory Data Analysis - could be two separate sections
- Your final GLM and relevant details
- Your final Tree and relevant details (can be before or after your GLM)
- Your Model Comparison (can be woven in sections above)
- Your Conclusion

Descriptions of the purpose for each section follow. Do not use this as a set of check boxes for the assignment. Instead, the idea here is to explain why you'd write each section, and you can work out what needs to be in each in order to fulfill that purpose.

- Introduction - The real estate developer has interests, but will you be able to address them? How do you understand the tasks you are presented with? It's important to state what you will be doing in the analysis, so the reader can make sure their understanding lines up with what you will be doing. The reader also needs an introduction to your data, either in this section, or below. They need enough detail to be able to determine if your actions later in the analysis are reasonable.

- Exploratory Data Analysis - You are the analyst here. That means you can use variable transformations, subsets of observations, and employ other modeling/analytic practices (e.g. training/testing data sets) at your discretion to aid in your analysis, so long as you explain your rationale for them. The reader needs to know what you found when looking at the data and what you decided to do about it, because this has impacts on the rest of your analysis. Remember our lessons from the first classes - be sure you look at the data! Here's your chance to share what you found based on how it impacts your analysis.

- GLM and Tree sections - These are new techniques. By having each in their own section, you can focus on your explanation for them one at a time. They can be in any order. The developer does not know what these are - you need to explain them at least enough they can follow your discussion. You have options that you need to pick for the different techniques (various tuning parameters) - for example, are you using a classification or a regression tree? What tree stopping options are being used? What input variables are you using and why? What type of GLM are you using? How will you be measuring performance? Be sure you discuss what options you are selecting and what made you choose those values. Helping explain this shows your understanding of the techniques. (Adding this level of detail is a feature of this assignment that makes it a little different from the example data analysis articles you looked at.)

You also want these sections to show off your ability to build models. It is not appropriate to only fit kitchen sink models or accept all default settings without exploring to see if that is really what is best for the task at hand. Your write-up should make it clear what you explored, but you don't have to show every model you considered. You can do a lot in sentences discussing your process. How you got to your final model should be clear.

Finally, remember to check out your "best" model for each technique. Did you check reasonable diagnostics? Does the model make sense? Are variables behaving as you expect? If you focus solely on model performance, that's missing the point. For example, you might find a model has very little error because a variant of the response was accidentally included as a predictor. You are responsible for checking over the model before reporting it. Your write-up should convey that you've done this (the reader can't read your mind to know you did it).

- Model Comparison - There are several ways to compare the models - performance, interpretability, variables involved, etc. Remember you are trying to address the real estate developer's questions of interest. You may do that with a focus on just one model, or maybe you found a way to combine results from a "best" GLM and a "best" tree. The idea for this section is that you need to convey the process used to pick a final model(s) to use to answer those questions, with justification for your choices. This section can be woven in the tree/GLM sections so it may not be a separate section.
- Conclusion - This section should be a stand alone summary of your analysis. It is where you get to state your final model and a quick summary of the process used to get there. You also need to address the developer's questions (if not done before), and this is your last place to do that! Remember to flush out the details here. If your final sentence is "Model 5 is the model I choose and it answers your questions", does it? Are you doing your job as the analyst to leave things at that? For that matter, what is Model 5?
- Printing Trees - It is fine to print your trees out to separate .pdfs if you want to for some reason (just be sure to include them in your submission!). Please do show your code though (so no echo = FALSE) and remember that your work should be reproducible.
- Length - Your submission should be as long as it needs to be, but not unnecessarily long. Plots, etc. are encouraged to help illustrate points, but don't include output that isn't needed. For example, a histogram or densityplot and favstats for each variable is something good to explore, but you shouldn't put that in your submission for every variable. You might need plots to point out outliers you are removing, or maybe you show one plot and then say others were similar. I encourage you to do any additional work in a separate .Qmd from your submission.
- Audience - For purposes of this submission, remember that the audience is the real estate developer, so you'll probably need to include some explanations that you'd leave out of another assignment. For example, the developer isn't going to know what a GLM is, or what type you are using, or why you'd use it. You should think about where that information should go, and do your best to explain what you need in order to report your findings. Similarly, assume that you found this data to assist the developer and they need basic details about it to understand the variables. They didn't hand it over to you.

You may remove these instructions if desired for your submission, but must leave the page on Academic Integrity above. The template with sections for the report is below.

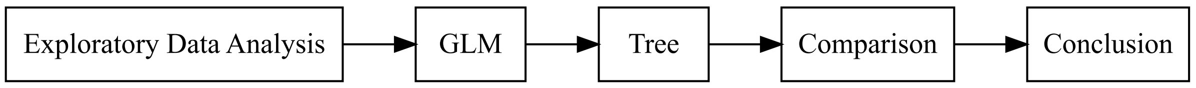
## Introduction

To predict whether a house sells for more than half a million dollars and determine the most significant variables that influence this decision in King County, Washington, we utilized a dataset found on Kaggle (a website that hosts various datasets), which contains relevant variables to help us answer the question. It lists all the properties sold from May 2014 to May 2015, along with their various characteristics. We built two types of models–generalized linear models and classification trees–that let us answer the question of which variables are the most relevant to predicting whether a house will sell for more than $500K.

Our process for analysis is outlined below in the diagram:

```
DiagrammeR::grViz("
  digraph graph2 {

  # left to right
  graph [rankdir = LR]

  # box shaped
  node [shape = box]

  # labels for boxes
  a [label = 'Exploratory Data Analysis']
  b [label = 'GLM']
  c [label = 'Tree']
  d [label = 'Comparison']
  e [label = 'Conclusion']

  a -> b -> c -> d -> e
  }

  ")
```

```
file:///C:\Users\miran\AppData\Local\Temp\RtmpmCLPAh\file72602fcb4561\widget726011be6afd.htm
```

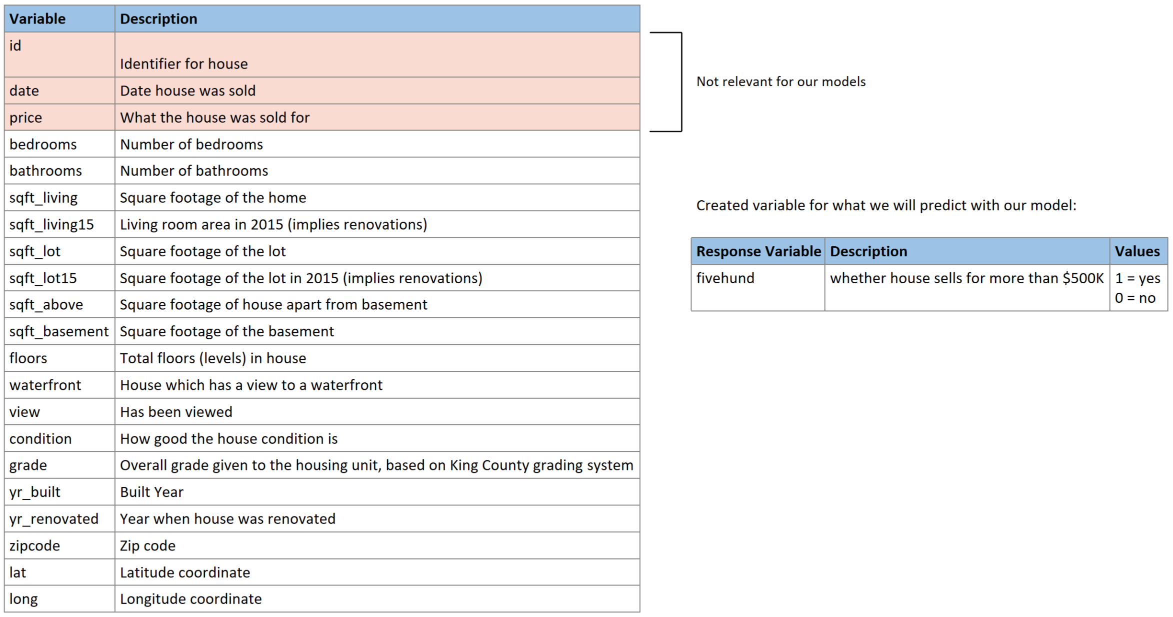


First, we will examine the dataset and explore the variables that seem to be useful indicators of house selling for more than $500K, then we will describe and apply two different models–generalized linear model (GLM) and classification tree–and compare them. Finally, we will summarize our findings on the final model and how it can be used to answer the question of interest.

## Exploratory Data Analysis (EDA)

### Dataset and Variables

We began by examining our dataset from Kaggle. The variables in the dataset are below.



| Variable | Description |
|---|---|
| id | Identifier for house |
| date | Date house was sold |
| price | What the house was sold for |
| bedrooms | Number of bedrooms |
| bathrooms | Number of bathrooms |
| sqft_living | Square footage of the home |
| sqft_living15 | Living room area in 2015 (implies renovations) |
| sqft_lot | Square footage of the lot |
| sqft_lot15 | Square footage of the lot in 2015 (implies renovations) |
| sqft_above | Square footage of house apart from basement |
| sqft_basement | Square footage of the basement |
| floors | Total floors (levels) in house |
| waterfront | House which has a view to a waterfront |
| view | Has been viewed |
| condition | How good the house condition is |
| grade | Overall grade given to the housing unit, based on King County grading system |
| yr_built | Built Year |
| yr_renovated | Year when house was renovated |
| zipcode | Zip code |
| lat | Latitude coordinate |
| long | Longitude coordinate |

Not relevant for our models

Created variable for what we will predict with our model:

| Response Variable | Description | Values |
|---|---|---|
| fivehund | whether house sells for more than $500K | 1 = yes<br>0 = no |

Using the *price* variable, we created a variable–*fivehund*–to indicate whether a house sells for more than $500K. *fivehund* will take on the value of "1" if the house sells for more than $500K and "0" if it doesn't. This is the variable we will predict.

```r
# reading in data set
kchouse <- read.csv("https://awagaman.people.amherst.edu/stat495/kc_house_data.csv",
                    header = T)

# create indicator var on whether house sells for more than $500K
kchouse1 <- kchouse |>
  mutate(fivehund = ifelse(price > 500000, 1, 0)) |>
  # remove outlier
  filter(bedrooms < 33)

# turn indicator var from numeric into categorical
kchouse1$fivehund = as.factor(kchouse1$fivehund)
```

We left *id*, *date*, and *price* out from our models since they are not helpful in predicting whether future houses will sell for more than $500K.

```r
# don't select id, date, price
kchouse2 <- kchouse1 |>
  select(-id, -date, -price)
```

Next, we looked at the remaining individual variables and their relationship with *fivehund*, whether a house will sell for more than $500K.

We kept variables as quantitative (if they're on a numerical scale like *bedrooms*) or categorical (if they're more like categories like *condition*) as appropriate to be used in our models.

```r
# variables that will be converted to categorical
cols <- c("waterfront", "condition", "grade", "zipcode")

kchouse2[cols] <- lapply(kchouse2[cols], factor)
```

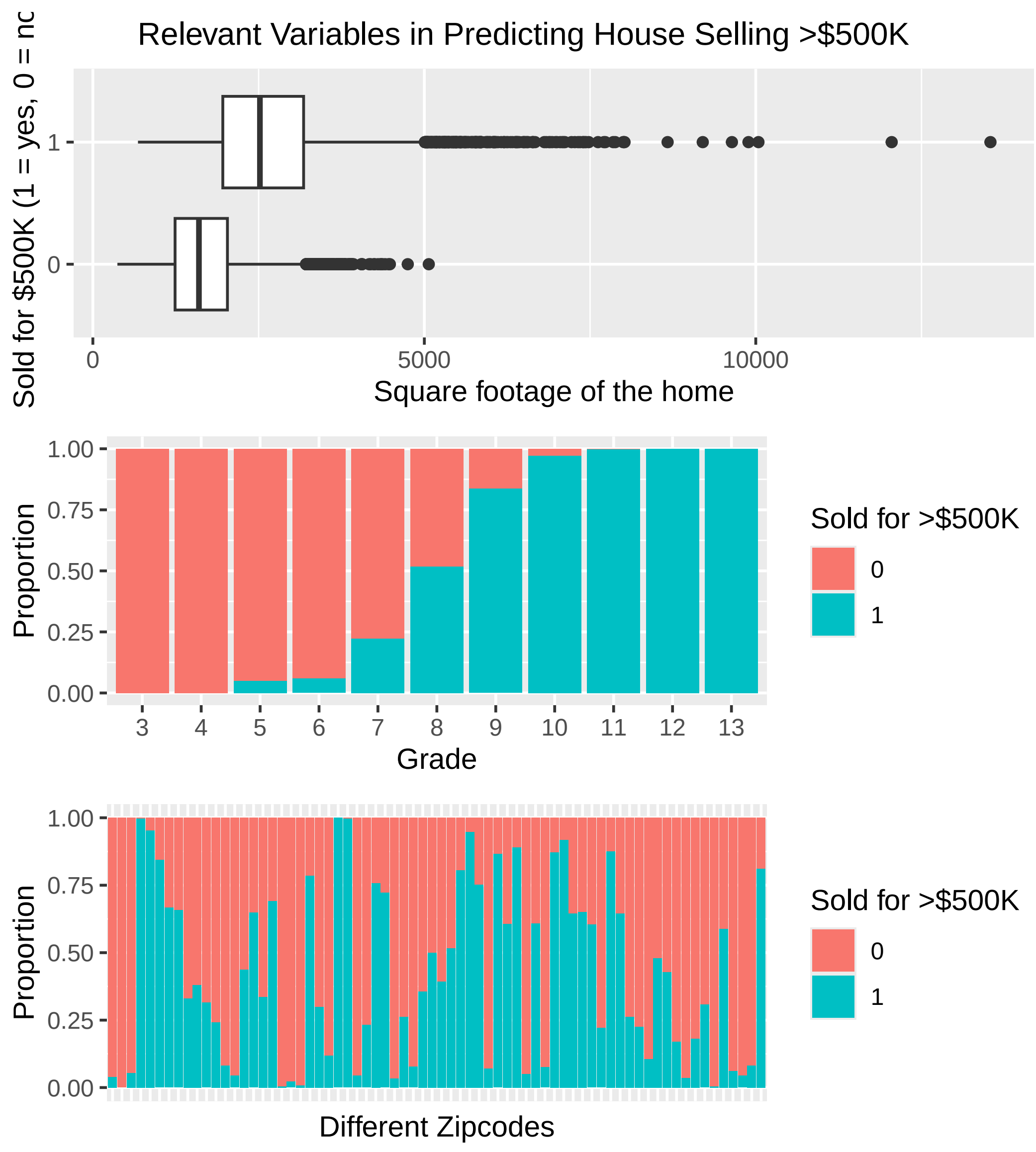
## Variables' Relationship with House Price >$500K

We explored the relationship between all predictors and the proportion of houses that were sold for more than $500K. We did this by plotting bar graph and box plots and found that *sqft_living*, *grade*, *zipcode*, *lat*, *long* seemed to be relevant indicators. For these variables, we saw striking distinctions between the proportion in houses that sold for more than $500K. In the box plot below, we saw that houses that sold for more than $500K tend to have more square footage than those that did not. On the bar graph for *grade*, we saw that houses with a *grade* larger than 9 had very high proportions of houses that sold for more than $500K while those with a *grade* smaller than an 8 had under 25% of houses selling for more than $500K. The proportion of houses sold for $500K also varied drastically for different zip codes.[1]

```r
# box plot for sqft_liv
sqft_liv <- ggplot(kchouse2, aes(x = sqft_living, y = fivehund)) +
  geom_boxplot() +
  labs(x = "Square footage of the home",
       y = "Sold for $500K (1 = yes, 0 = no)")

# bar graph for grade
grade <- ggplot(kchouse2, aes(x = grade, fill = fivehund)) +
```

[1]The bar graph for zip code does not display the exact zip code since, for now, our concern is not seeing the exact zip codes, but rather noticing that zip codes seem impactful on proportions of houses selling for $500K.

```r
  geom_bar(position = "fill") +
  labs(fill = "Sold for >$500K",
       x = "Grade",
       y = "Proportion")

# bar graph for zip code
zip <- ggplot(data = kchouse2, aes(x = zipcode, fill = fivehund)) +
  geom_bar(position = "fill") +
  theme(axis.text.x=element_blank(),
        axis.ticks.x=element_blank())  +
  labs(fill = "Sold for >$500K",
       x = "Different Zipcodes",
       y = "Proportion")

# arranging graphs together
grid.arrange(sqft_liv, grade, zip,
             top = "Relevant Variables in Predicting House Selling >$500K")
```

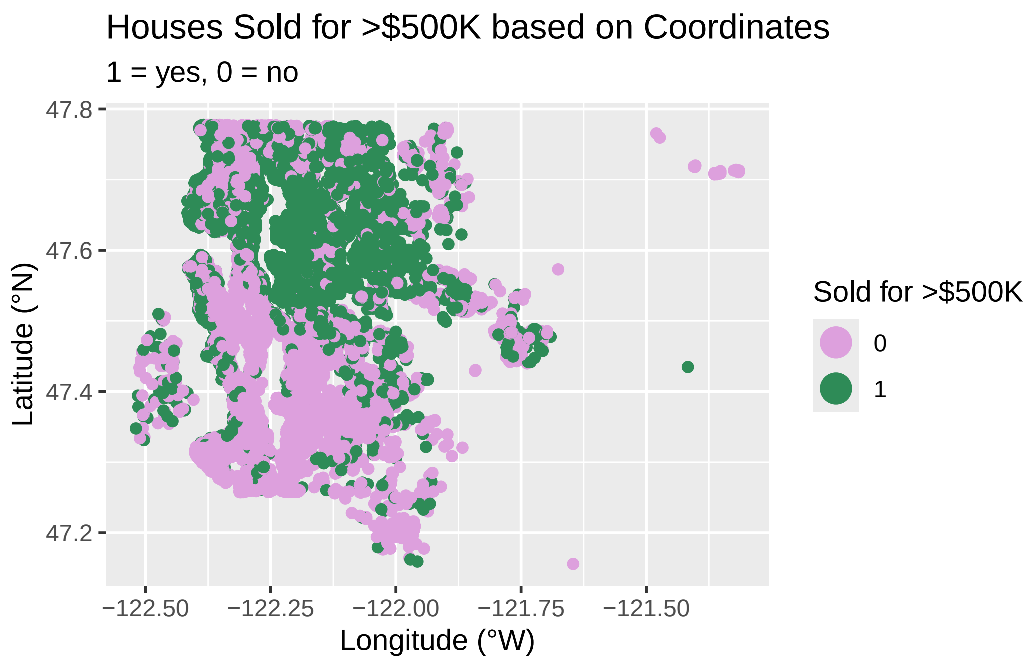


There were other variables that have relationship with whether house sells for more than $500K, but the difference in the proportions of houses selling more than $500K are more obvious in the variables selected (e.g., there is greater differences between the proportion selling for $500K as *grade* increases than *condition* increases).

Additionally, there are variables related to selected variables (e.g., *sqft_lot* similar to *sqft_living*), but we do not want to include variables that correlate too much with each other since we will introduce issues of multicollinearity where the added variable does not add much to improve the performance of the model and could rather harm it by overfitting (overly matching to the data we have, making it apply poorly in future applications).

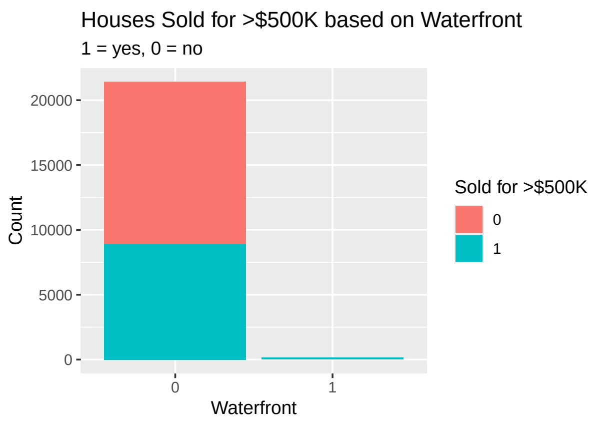
Latitude and longitude also seemed to be important variables: when we plotted latitude and longitude with colors indicating whether a house sells for more than \$500K, we saw clusters of \$500K houses around 47.6 °N and -122.25 °W.

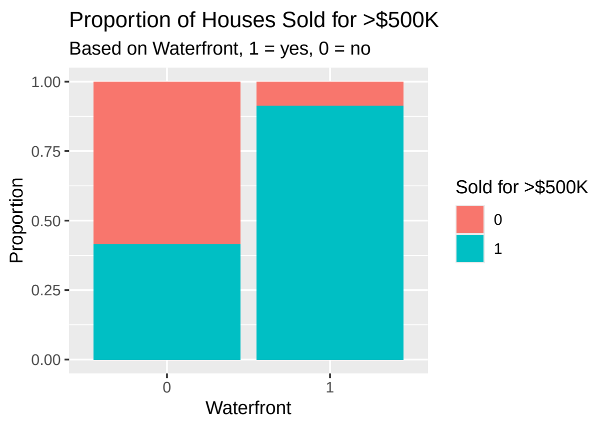
```
# plot lat and long
ggplot(data = kchouse2, aes(x = long, y = lat,
                         color = fivehund)) +
  geom_point() +
  scale_colour_manual(values = c("plum", "seagreen")) +
  labs(y = "Latitude (°N)",
       x = "Longitude (°W)",
      title = "Houses Sold for >$500K based on Coordinates",
       subtitle = "1 = yes, 0 = no",
      color = "Sold for >$500K") +
  guides(color = guide_legend(override.aes = list(size=5)))
```



Waterfront also seemed to be a great indicator of whether a house sells for more than \$500K. More than 80% of houses with a waterfront sell for more than \$500K, while less than 50% of houses without a waterfront sell for more than \$500K. However, we will not include this variable in our models, since there were not many houses with waterfront to begin with, so we might end up overfitting the model.

```r
# bar graph on waterfront
g1 <- ggplot(data = kchouse2,
       aes(x = waterfront, fill = fivehund)) +
  geom_bar() +
  labs(fill = "Sold for >$500K",
       x = "Waterfront",
       y = "Count",
       title = "Houses Sold for >$500K based on Waterfront",
       subtitle = "1 = yes, 0 = no")

# proportion bar graph on waterfront
g2 <- ggplot(data = kchouse2,
       aes(x = waterfront, fill = fivehund)) +
  geom_bar(position = "fill") +
  labs(fill = "Sold for >$500K",
       x = "Waterfront",
       y = "Proportion",
       title = "Proportion of Houses Sold for >$500K",
       subtitle = "Based on Waterfront, 1 = yes, 0 = no")

# arranging graphs together
grid.arrange(g1, g2, ncol = 2)
```





## Plans for Modeling

Since we want to ensure our model will perform well in actual applications, we split our data into a training and test set and used a standard procedure called cross-validation (CV).[2] The training set is the portion of data used to build our model, and the test set allows us to see the performance of the model in real scenarios. We divided 80% of the entire data set to be the training data set and 20% to be the test set. We will apply the two models–GLM and classification tree–to the training data set first to evaluate performance. Then, we will fit the model to the test set to see how well the model will perform in actual applications.

```r
set.seed(495)

# split into training and test
split_data <- initial_split(kchouse2, prop = 0.8)
train_data <- training(split_data)
test_data <- testing(split_data)

# set up cv framework
model_control <- trainControl(method = "repeatedcv", number = 10, repeats = 5)
```

[2] Cross-validation is when you split up the training set further into separate portions, where each portion will take a turn in being the training set (instead of having a single training and test split). This helps build a more accurate model.

## GLM

### Explanation

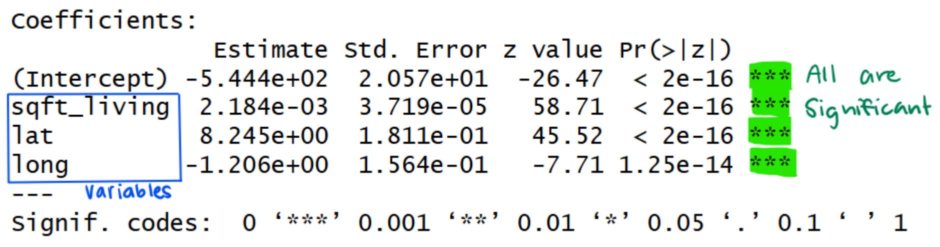
A generalized linear model (GLM) is an equation involving different variables to make predictions. In our case, we are predicting *fivehund*, whether the price sells for more than $500K. Since *fivehund* has binary response values–either 1 or 0–we will use a type of GLM called logistic regression. It allows predictions to take on a value of 1 or 0 instead of a range of numerical values, as in the case of usual models.

### Process and Results

First, we tried to create a baseline to compare future models to by fitting a model comprised of all variables except for *id*, *date*, and *price*. However, the model did not converge, which tells us that we are likely overfitting the data. This meant that narrowing down the variables is necessary.

Using the variables that seemed the most relevant in EDA, we created a model consisting of *sqft_living*, *grade*, *lat*, *long*. However, this model and separate model we tried that included *zipcode* instead of *grade* also did not converge.

Removing *grade* and *zipcode* from our model, we arrived at a model that did perform well. All the predictors in the final GLM, *sqft_living*, *lat*, and *long* are statistically significant, meaning that each variable is useful in predicting whether the house will sell for more than $500K.



```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.444e+02  2.057e+01  -26.47  < 2e-16 ***
sqft_living  2.184e-03  3.719e-05   58.71  < 2e-16 ***
lat          8.245e+00  1.811e-01   45.52  < 2e-16 ***
long        -1.206e+00  1.564e-01   -7.71 1.25e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

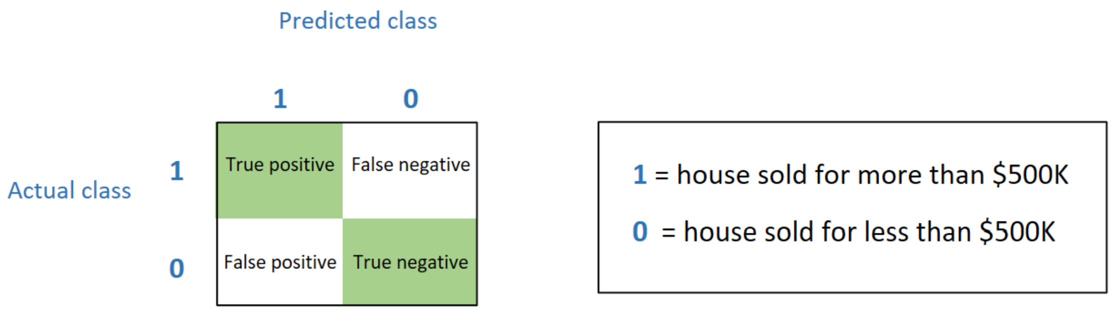
The code to generate the model is below:

```
# final glm
finalreg <-  caret::train(
  fivehund ~ sqft_living + lat + long,
  data = train_data,
  method = "glm",
  family = "binomial",
  trControl = model_control
```

```r
)

summary(finalreg)
```

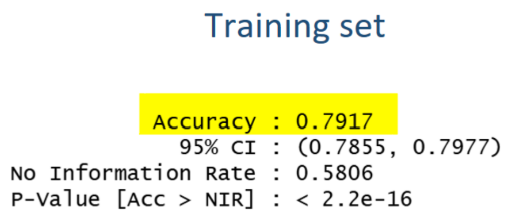
```
Call:
NULL

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -5.444e+02  2.057e+01  -26.47  < 2e-16 ***
sqft_living  2.184e-03  3.719e-05   58.71  < 2e-16 ***
lat          8.245e+00  1.811e-01   45.52  < 2e-16 ***
long        -1.206e+00  1.564e-01   -7.71 1.25e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 23498  on 17275  degrees of freedom
Residual deviance: 14686  on 17272  degrees of freedom
AIC: 14694

Number of Fisher Scoring iterations: 5
```

We also explored other combinations of models, such as replacing *long* with *floors* or *bedrooms*, but these offered no significant improvement in our models. The performance metric we used to evaluate and compare models is accuracy, which is the proportion of correct classifications.



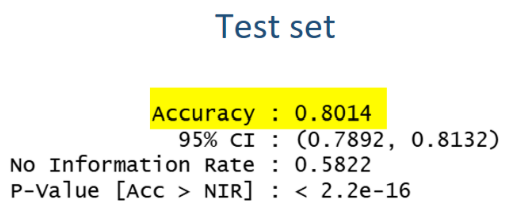
$$\text{Accuracy} = \frac{\text{number of true positives and true negatives}}{\text{total classifications}}$$

We computed the accuracy of our models on the training and test sets. For our final model, the accuracy was 0.7917 on the training set and 0.8014 on the test set. A value of 0.8014 on the test set means that for future performances, the model works quite well as it correctly predicts whether a house will sell for more than $500K around 80% of the time.



Training set

```
              Accuracy : 0.7917
                95% CI : (0.7855, 0.7977)
   No Information Rate : 0.5806
   P-Value [Acc > NIR] : < 2.2e-16
```



Test set

```
              Accuracy : 0.8014
                95% CI : (0.7892, 0.8132)
   No Information Rate : 0.5822
   P-Value [Acc > NIR] : < 2.2e-16
```

```
# training set accuracy
finalreg_trainpred <- predict(finalreg, train_data)
confusionMatrix(finalreg_trainpred, as.factor(train_data$fivehund))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 8558 2126
         1 1473 5119

               Accuracy : 0.7917
                 95% CI : (0.7855, 0.7977)
    No Information Rate : 0.5806
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5668

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8532
            Specificity : 0.7066
         Pos Pred Value : 0.8010
         Neg Pred Value : 0.7765
```

```
             Prevalence : 0.5806
         Detection Rate : 0.4954
   Detection Prevalence : 0.6184
      Balanced Accuracy : 0.7799

       'Positive' Class : 0
```

```r
# test set accuracy
finalreg_testpred <- predict(finalreg, test_data)
confusionMatrix(finalreg_testpred, as.factor(test_data$fivehund))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2173  516
         1  342 1289

               Accuracy : 0.8014
                 95% CI : (0.7892, 0.8132)
    No Information Rate : 0.5822
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5861

 Mcnemar's Test P-Value : 3.503e-09

            Sensitivity : 0.8640
            Specificity : 0.7141
         Pos Pred Value : 0.8081
         Neg Pred Value : 0.7903
             Prevalence : 0.5822
         Detection Rate : 0.5030
   Detection Prevalence : 0.6225
      Balanced Accuracy : 0.7891

       'Positive' Class : 0
```

## Classification Tree

### Explanation

Classification trees are similar to flow charts that lead you to a predicted result. The difference is that at each split, there are always two diverging paths based on a certain variable. We provide the tree with a list of variables, and the tree automatically decides when to split and the variable to base the split on.

We can limit the growth of the tree if it becomes too large with too many splits (which hinders the performance) by using what's called the "complexity parameter" (CP).[3] CP sets a threshold for how much the split has to improve the performance of the model by in order to proceed with the split. If it's set to 0, the tree could grow without limits.

### Process and Results

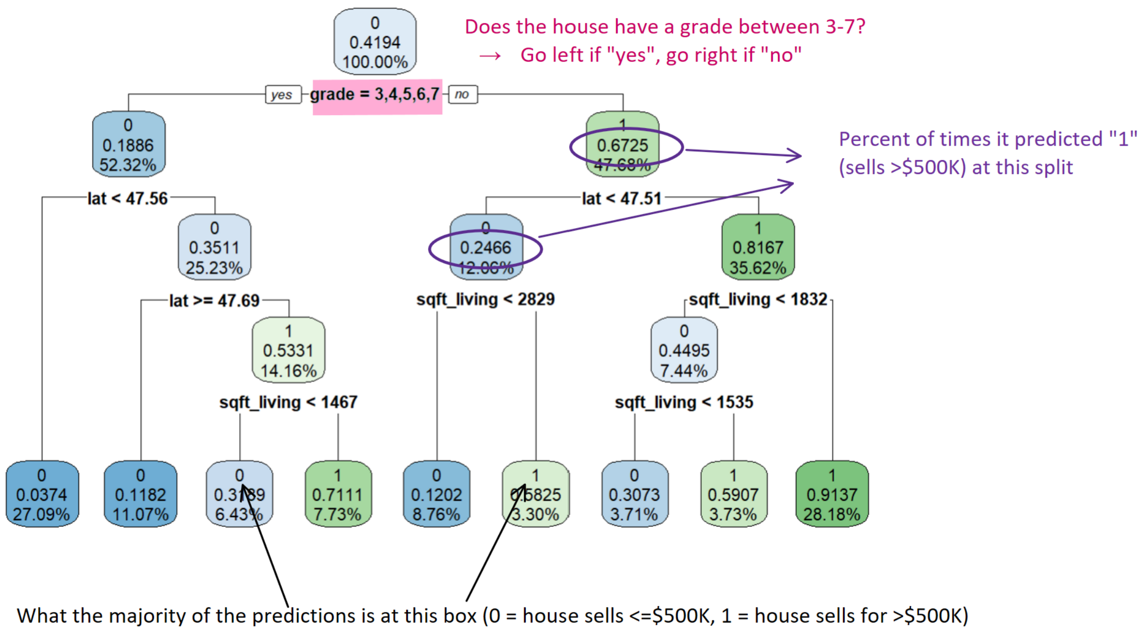
Given a list of all possible variables and the default CP value, the baseline model tree only chose *zipcode* and *sqft_living* to decide the splits. The accuracy is quite good, at around 86%, but it is quite difficult to tell the zip code values in each split because of the long list of zip codes involved.

When we removed *zipcode* from our list and instead limited the variables to *sqft_living*, *waterfront*[4], *view*, *lat*, and *long* (which we identified as relevant from our exploratory data analysis), we obtained a tree that used *grade*, *sqft_living*, and *lat* for its splits. We explored different CP values–0, 0.001 (both made the tree too complex to follow), 0.05, 0.1 (both made the tree too simple with very few splits)–but the default CP value of 0.01 seemed to strike a balance between being too simple or too complex and gave the best accuracy.

How the tree predicts whether a house sells for more than $500K is by seeing at each split, is something true for this aspect of the house (e.g., is the latitude of the house <47.56? If yes, go left, else go right) until we reach the final box. The number on top of the rectangular box is its final prediction. If it's a "1", then it mostly predicted that the house will sell for more than $500K, else it mostly predicted that it did not.

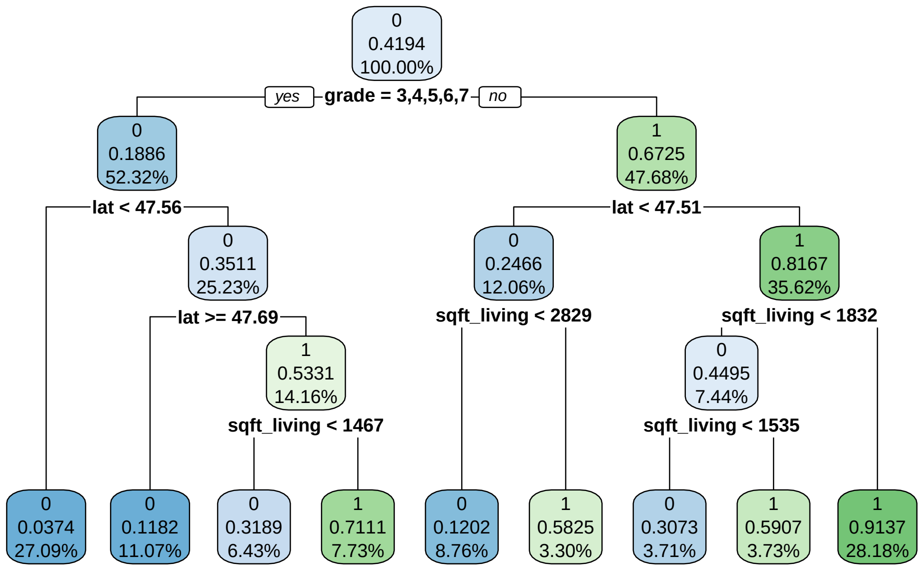
[3]There are also other ways to "tune" the tree, using the "minbucket" and "minsplit" options, which determine the number of observations that result in each branch after a split and the number of observations before a split. This can prevent the tree from splitting when there are very few observations (might end up overfitting the model). However, since the number of observations in the data set is more than 10,000, even when the tree splits off 1% of the observation, it is still hundreds of observations, so we do not have to worry about these parameters.

[4]We included waterfront in the tree model even though we stated concerns about it in the EDA section to see if the tree would select it. The tree did not end up including the variable.



The code for the tree is below:

```r
set.seed(495)

# set tuning parameters
house_control <- rpart.control(cp = 0.01)

# create tree with tuning param
finaltree <- rpart(data = train_data,
                   fivehund ~ sqft_living +
                 waterfront + view + grade + lat + long,
                   control = house_control,
                   method = "class")

rpart.plot(finaltree, digits = 4)
```



```
# summary(finaltree)
```

In general, we want our model to be better than the baseline model (full tree with all the variables in this case), but the accuracy for our model with *grade*, *sqft_living*, and *lat* is not far off–85.86% for on the training set and 85.9% on the test set. This means that the tree model will correctly predict whether a house will sell for more than $500K around 85.9% of the time. We favored this model over the baseline model since the tree is clearer to see without a long list of zip codes at the splits.

The code we used to obtain accuracy on the training set and test set is below.

```
set.seed(495)

# fitting tree to training set
train_finaltree <- mutate(train_data, pred = predict(finaltree, type="class"))

tally(pred ~ fivehund, data = train_finaltree)
```

```
    fivehund
pred    0    1
   0 8723 1134
   1 1308 6111
```

```
# training accuracy = 0.8586478
(8723 + 6111)/(8723 + 6111 + 1134 + 1308 )
```

```
[1] 0.8586478
```

```
# fitting tree to test set
test_finaltree <- mutate(test_data,
                     pred = predict(finaltree,
                                    type="class",
                                    newdata = test_data))

tally(pred ~ fivehund, data = test_finaltree)
```

```
    fivehund
pred    0    1
   0 2184  278
   1  331 1527
```

```
# test accuracy = 0.8590278
(2184 + 1527)/(2184 + 1527 + 331 + 278)
```

```
[1] 0.8590278
```

## Model Comparison

To pick the final model, we considered the benefits and downsides of each model and compared the accuracy between the final GLM and the final classification tree model.

GLM

| Benefits | Downsides |
|---|---|
| ▶ Easy to use - similar to plugging in values for an equation | ▶ Variables' impact on predictions depend on other variables present in the model |
| ▶ Interpretable - can see how variables affect predicted values (through coefficients) | ▶ Have to manually decide which variables to include in the model |

Tree

| Benefits | Downsides |
|---|---|
| ▶ Easy to follow like a flow chart | ▶ "Greedy" - might not choose the best combination overall |
| ▶ Automated selection of most useful variables given a list a variables | ▶ Can't see how variables directly impacts the predicted value |
| ▶ Good for nonlinear relationship between predictors and response | |

GLMs will include all the variables we provide, and we can distinguish whether each variable is significant in the model and whether it increases or decreases the response values (based on coefficients). However, the direction of influence (whether it increases or decreases) is always dependent on other variables present in the model. For instance, if "condition" and "bedrooms" are both in a model and "condition" has a negative coefficient while "bedrooms" has a positive coefficient, it does not mean that better conditions will negatively impact the price of a house. The manual aspect of selecting variables for a GLM can be both a positive and a negative aspect. On the positive side, you can have control over what variables you want to include in the model (as opposed to the tree that might not include all the variables you give it). On the other hand, it is practically difficult to try out all the different combinations of predictors, even with automated selection methods, to see which one is the most optimal.

For trees, given a list of variables, it will pick the ones most relevant to make predictions. However, trees are "greedy" in that they will pick what they think is the best variable to split on early on. However, this might not result in the best combination of splits overall since every resulting split will depend on your initial split. Also, trees are not interpretable in the same way as GLMs–it does not tell us "one unit change in x changes this amount of odds in y." Even though we are able to see the variables involved, we do not see directly how the variables impact house selling for more than $500K. However, trees are simple to use, will pick the most useful variables, and will not forcibly include all of them.

Considering these aspects and the accuracy between the models, we decided on the classification tree for predicting whether houses will sell for more than $500K. The final GLM that involved the variables *sqft_living*, *lat*, and *long* had 80.1% accuracy on the test set while the final classification tree with variables *grade*, *sqft_living*, and *lat* had 85.9% accuracy. Even though trees can be "greedy" and might not choose the best combination of variables, the tree still had 5% higher accuracy than the GLM, and it is easy to follow and see which variables the tree chose as the most useful.

## Conclusion

To summarize, after a process of analyzing and picking relevant models to include in our models, comparing the performances of GLM and classification tree, we propose the final model to predict whether houses will sell for more than \$500K as a classification tree with the tuning parameter CP = 0.01 and variables *grade*, *sqft_living*, and *lat*. This model has 85.9% accuracy when applied to the test set, which indicates that it will perform well in future performances.

The tree decided that these three variables are the only ones necessary out of the other variables identified as potentially relevant from the EDA process (*waterfront*, *view*, and *long*) to make predictions on a house selling for more than \$500K. In terms of how they impact the model, these variables are the ones that the tree chooses to decide the splits, which impact the final classification of whether the house sells for more than \$500K.

However, the variables included in the tree are not necessarily the most important or the only variables that matter in predicting house prices. For example, waterfront was not included in our model because there were so few houses with waterfront, and including them could potentially overfit our models, but it is intuitive that houses near water will sell for higher prices because they are prized locations and only so many can be near water. Another limitation of the model is that it is only applicable to houses in King County, MA, and will not generalize well to other areas since we only trained the model based on the data in that specific county. There is no definite, objectively "best" model that captures all the possible variables' impact on house price while also being accurate, but the final classification tree, as stated above, could perform quite well in predicting whether a house will sell for more than \$500K in King County, WA, with around 85.9% accuracy.