KC House Pricing

```
#Neccessary libraries
library(tidyverse) # to be able to use data visual.tools
library(viridis) #to make plots easier to read
library(ggplot2) # to make plots
library(ggrepel) #ggrepel provides geoms for ggplot2
library(gridExtra) #notably to arrange multiple grid-based plots on a page
```

Estimating with linear modell

Consider the data set *kc_house_data.csv* providing the prices and specifications of houses sold between May 2014 to May 2015 in King County (Seattle, USA). For our analysis we consider the following variables: **price:** Price,**bedrooms:** Number of bedrooms, **bathrooms:** Number of bathrooms per bedroom, **sqft_living:** Square footage of the home, **floors:** Total floors in house, **view:** Has been viewed (1 for viewed, 0 for not viewed / has to be converted), **condition:** How good is the condition (from 1 to 5), **grade:** Grade given to the housing unit (from 1 to 13), **yr_built:** Year the house was built.

First of all, we read the dataset, where we will extract only variables from above:

```
# read the data with specified columns
kc.house <- read.csv("kc_house_data.csv")[ ,c('price', 'bedrooms', 'bathrooms',
                                              'sqft_living', 'floors', 'view',
                                              'condition', 'grade', 'yr_built')]
head(kc.house,2) #quick look at the dataset with 2 rows
```

```
##    price bedrooms bathrooms sqft_living floors view condition grade yr_built
## 1 221900        3      1.00        1180      1    0         3     7     1955
## 2 538000        3      2.25        2570      2    0         3     7     1951
```

We have total 21613 observations of 9 variables.

We will format **view** into binary format, where "1" is for viewed and "0" otherwise:

```
#we convert column values into boolean type,
#if we have views more than 0, so it will take true value
#but the true value we will write in num format
kc.house <- kc.house %>% mutate(view = as.numeric(view > 0))
```

I'm assuming that **bathrooms** column includes the number of bathrooms per house. As was requested in the exercise description **bathrooms** is for number of bathrooms per bedroom. So, we should convert this column in proportion of $\frac{\textbf{bathrooms}}{\textbf{bedrooms}}$.

```
#change the bathrooms column through ratio
kc.house <- kc.house %>% mutate(bathrooms = as.numeric(bathrooms/bedrooms))
```

```
#quick review of variable types in df
filter(kc.house, bedrooms == 0)
```

```
##      price bedrooms bathrooms sqft_living floors view condition grade yr_built
## 1  1095000        0       NaN        3064    3.5    1         3     7     1990
## 2   380000        0       NaN        1470    3.0    1         3     8     2006
## 3   288000        0       Inf        1430    3.0    0         3     7     1999
## 4   228000        0       Inf         390    1.0    0         2     4     1953
## 5  1295650        0       NaN        4810    2.0    0         3    12     1990
## 6   339950        0       Inf        2290    2.0    0         3     8     1985
## 7   240000        0       Inf        1810    2.0    0         3     7     2003
## 8   355000        0       NaN        2460    2.0    0         3     8     1990
## 9   235000        0       NaN        1470    2.0    0         3     7     1996
## 10  320000        0       Inf        1490    2.0    0         3     7     1999
## 11  139950        0       NaN         844    1.0    0         4     7     1913
## 12  265000        0       Inf         384    1.0    0         3     4     2003
## 13  142000        0       NaN         290    1.0    0         1     1     1963
```

We have interesting situation, that some of houses have 0 bedrooms(what can be occured by report mistake or that a house is some kind of warehouse). Because of that we get for our variable **bathrooms** either "NaN" or "Inf" values. I think it is better to omit the rows with 0 bedrooms, to better estimate the model:

```
kc.house <- kc.house %>% filter(bedrooms != 0)#delete rows without bedrooms
str(kc.house)#quick review on the dataset
```

```
## 'data.frame':    21600 obs. of  9 variables:
##  $ price      : num  221900 538000 180000 604000 510000 ...
##  $ bedrooms   : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms  : num  0.333 0.75 0.5 0.75 0.667 ...
##  $ sqft_living: int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ floors     : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ view       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition  : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade      : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ yr_built   : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
```

We have now 21600 observations, so now we can estimate our linear modell with the response variable **price** and all remaining variables as covariates. For this we use *lm()* to estimate the coefficients of the model.

```
#Estimate lin model with the response variable price
mod <- lm(price ~ bedrooms + bathrooms + sqft_living + floors + view
          + condition + grade + yr_built, data = kc.house)
summary(mod)
```

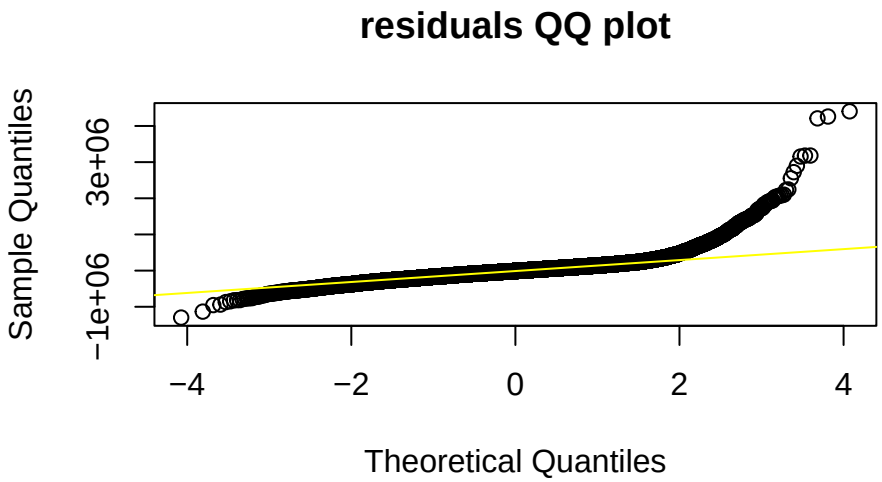
```
##
## Call:
## lm(formula = price ~ bedrooms + bathrooms + sqft_living + floors +
##     view + condition + grade + yr_built, data = kc.house)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1298105  -113877   -10571    90562  4404719
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6200334.69  131211.84  47.254  < 2e-16 ***
## bedrooms     -24257.45    2482.82  -9.770  < 2e-16 ***
## bathrooms     95137.46   10255.47   9.277  < 2e-16 ***
## sqft_living     183.74       3.19  57.595  < 2e-16 ***
## floors        31195.74    3555.45   8.774  < 2e-16 ***
## view         144042.41    5469.50  26.336  < 2e-16 ***
## condition     19603.83    2550.68   7.686 1.59e-14 ***
## grade        125775.28    2218.66  56.690  < 2e-16 ***
## yr_built      -3608.45      68.38 -52.770  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 224200 on 21591 degrees of freedom
## Multiple R-squared:  0.627,  Adjusted R-squared:  0.6269
## F-statistic:  4537 on 8 and 21591 DF,  p-value: < 2.2e-16
```

We can see that all variables are significant. Surprisingly though, it seems that a higher number of bedrooms and a higher number for the variable **yr_built** (meaning a newer house) leads to a lower price for the house. R^2 is approximately 0.63, which means that our model explains about 63% of the variance in price.

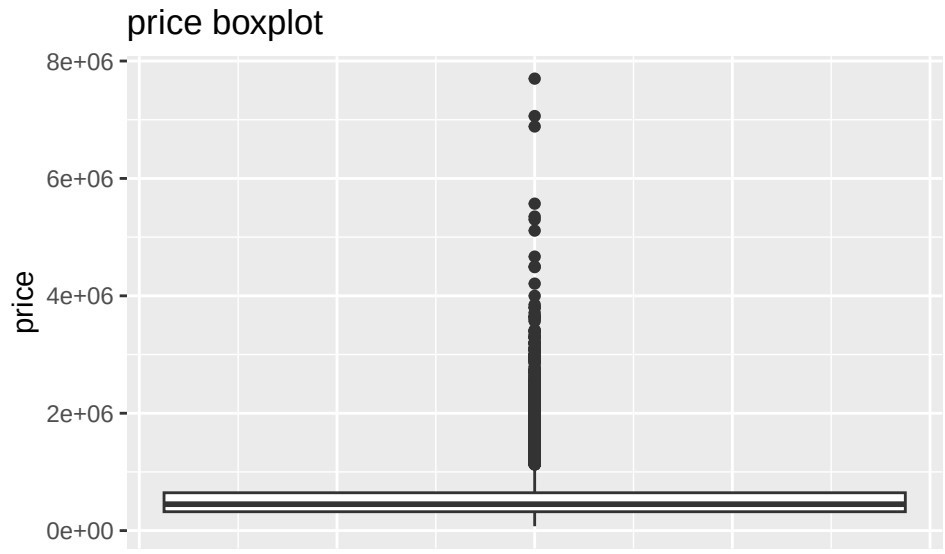
Now we shall take a look at the residuals:

```
qqnorm(mod$residuals, main = "residuals QQ plot")
qqline(mod$residuals, col = "yellow")
```



We notice that our residuals are not normally distributed, the problem lies in the response variable price, because it has many large outliers, we can check it with boxplot:

```
#plotting the distribution of price
ggplot(kc.house, aes(y = price)) +
  geom_boxplot() +
  theme(axis.text.x=element_blank(), axis.ticks.x = element_blank()) +
  labs(title = "price boxplot")
```



This violates the assumption of normally distributed residuals.

distribution of price vs. log(price)

Now we want to produce a histogram and a QQ-plot of the response variable price, as well as of its log-transform log(price) and compare both distributions to the normal one. First of all, we create a new variable **logprice** in our dataframe:

```
#generating new variable log(price)
kc.house <- kc.house %>% mutate(lprice = log(price))
str(kc.house) # quick review
```

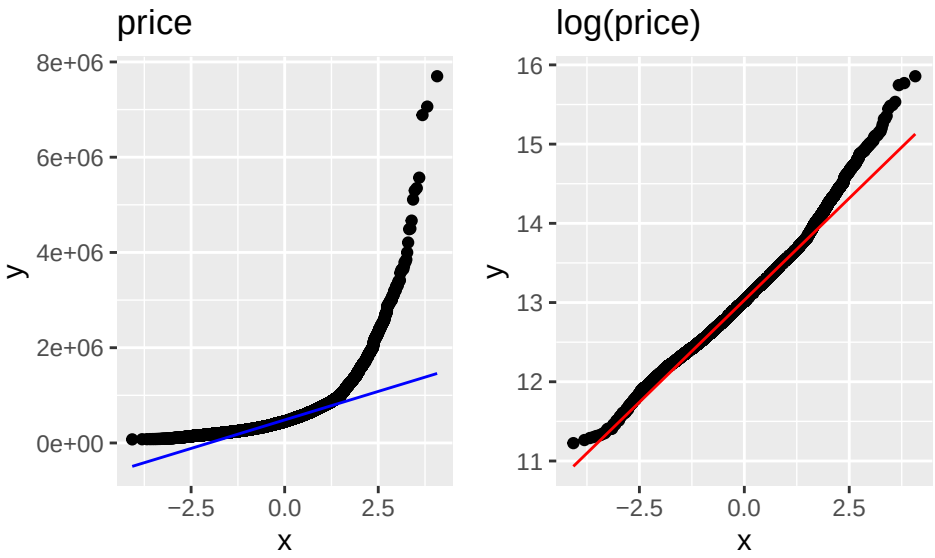
```
## 'data.frame':    21600 obs. of  10 variables:
##  $ price      : num  221900 538000 180000 604000 510000 ...
##  $ bedrooms   : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms  : num  0.333 0.75 0.5 0.75 0.667 ...
##  $ sqft_living: int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ floors     : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ view       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition  : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade      : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ yr_built   : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
##  $ lprice     : num  12.3 13.2 12.1 13.3 13.1 ...
```

Now QQ-plots:

```r
#qqplot of price
q.price <- ggplot(kc.house, aes(sample=price)) +
  geom_qq()+
  geom_qq_line(col = "blue")+
  labs(title = "price")

#qqplot of log(price)
q.lprice <- ggplot(kc.house, aes(sample = lprice))+
  geom_qq()+
  geom_qq_line(col = "red")+
  labs(title = "log(price)" )

#plots side by side
grid.arrange(q.price, q.lprice, ncol = 2)
```

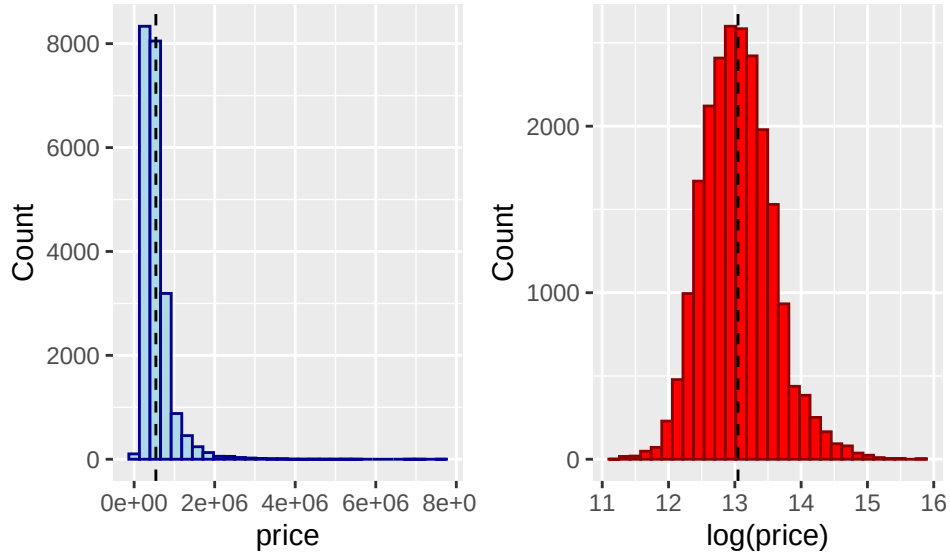


Let's have a look at histograms:

```r
#histogram of price
hist.price <- ggplot(kc.house, aes(x = price)) +
    geom_histogram(color="darkblue", fill="lightblue") +
    geom_vline(aes(xintercept=mean(price)), linetype="dashed")+
    labs(x = "price",y = "Count")

#histogram of log(price)
hist.lprice <- ggplot(kc.house, aes(x = lprice)) +
   geom_histogram(color="darkred", fill="red") +
   geom_vline(aes(xintercept=mean(lprice)), linetype="dashed") +
    labs(x = "log(price)",y = "Count")

#plots side by side
grid.arrange(hist.price, hist.lprice, ncol = 2)
```



We can clearly see that **log(price)** is much more normally distributed as compared to **price**. This is also confirmed by taking a closer look at their distributions.

Now we will fit a new model with the response variable **log(price)**:

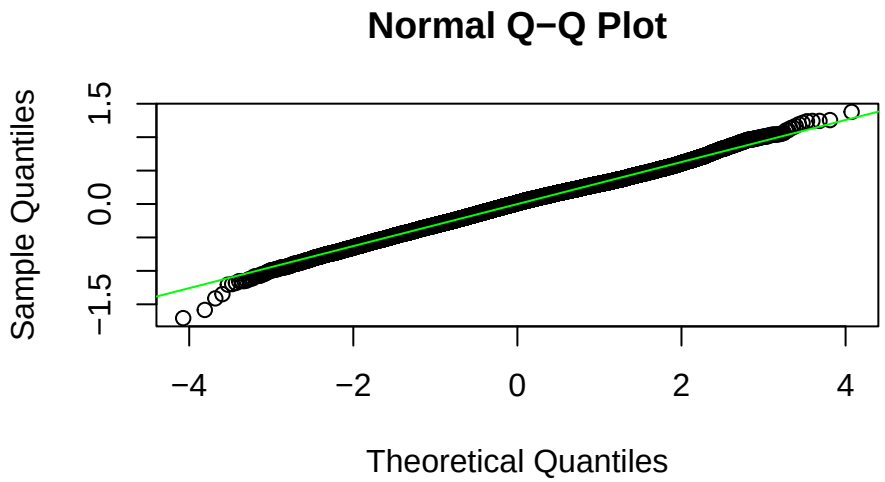
```
#estimating new model using log(price) as dependent variable
lmod <- lm(lprice  ~ bedrooms + bathrooms + sqft_living + floors + view
           + condition + grade + yr_built, data = kc.house)
summary(lmod)
```

```
##
## Call:
## lm(formula = lprice ~ bedrooms + bathrooms + sqft_living + floors +
##     view + condition + grade + yr_built, data = kc.house)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7070 -0.2126  0.0157  0.2115  1.3778
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.133e+01  1.842e-01 115.798  < 2e-16 ***
## bedrooms     2.092e-02  3.486e-03   6.001 1.99e-09 ***
## bathrooms    2.639e-01  1.440e-02  18.323  < 2e-16 ***
## sqft_living  1.767e-04  4.479e-06  39.450  < 2e-16 ***
## floors       8.329e-02  4.992e-03  16.683  < 2e-16 ***
## view         1.592e-01  7.680e-03  20.735  < 2e-16 ***
## condition    4.272e-02  3.581e-03  11.929  < 2e-16 ***
## grade        2.230e-01  3.115e-03  71.584  < 2e-16 ***
## yr_built    -5.524e-03  9.602e-05 -57.528  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3149 on 21591 degrees of freedom
```

```
## Multiple R-squared:  0.6425, Adjusted R-squared:  0.6424
## F-statistic:  4851 on 8 and 21591 DF,  p-value: < 2.2e-16
```

Once again, all parameters are highly significant. However, the estimate for the variable **bedrooms** is now positive, meaning that more bedrooms lead to a higher house price. This makes much more sense than in the previous model from a). It still appears that a newer house house has a lower price than an older one. R^2 is now somewhat higher as well, we are able to explain 64 of the variance of our response variable using this model.

```
#qqplot of residuals
qqnorm(lmod$residuals)
qqline(lmod$residuals, col = "green")
```



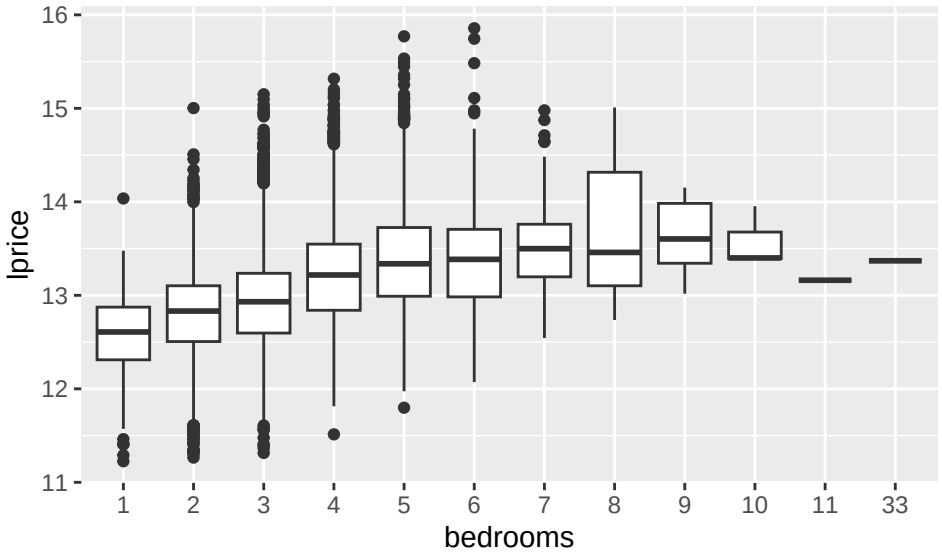
In this model, the residuals are much more normally distributed, which makes this model more adequate than the first one.

Interpretation of covariates

Now we want to interpret in the model from b) interpret the effect of each covariate on the response. Where we will plot each covariate against **log(price)**:

Bedrooms:

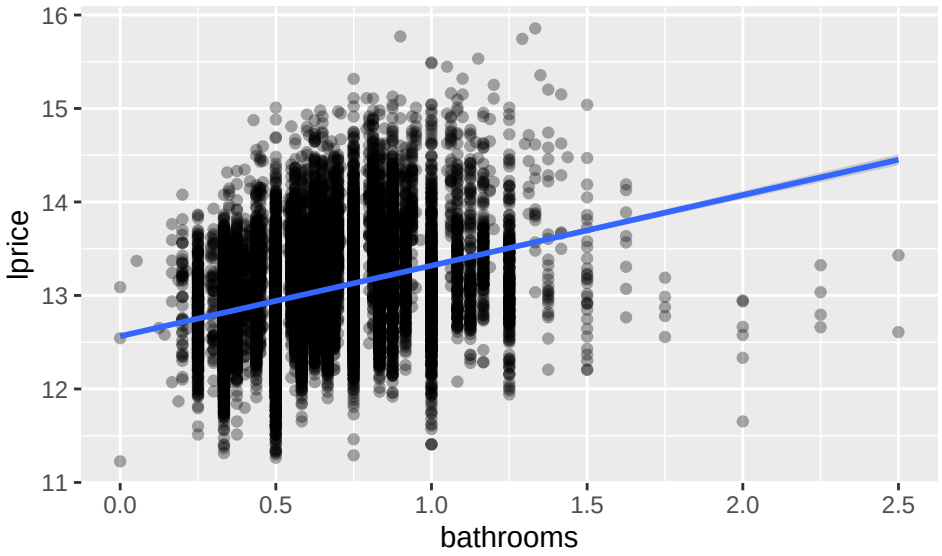
```
ggplot(kc.house, aes(x = as.factor(bedrooms), y = lprice)) +
  geom_boxplot() +
  xlab("bedrooms")
```



On average, a higher number of bedrooms means a higher price for the house. However, this changes again for higher numbers of bedrooms. We must be careful of taking this trend too seriously though, because we only have very few observations with a very high number of bedrooms. And for **bedrooms** $= \{11, 33\}$ there is only one observation each.

Bathrooms:

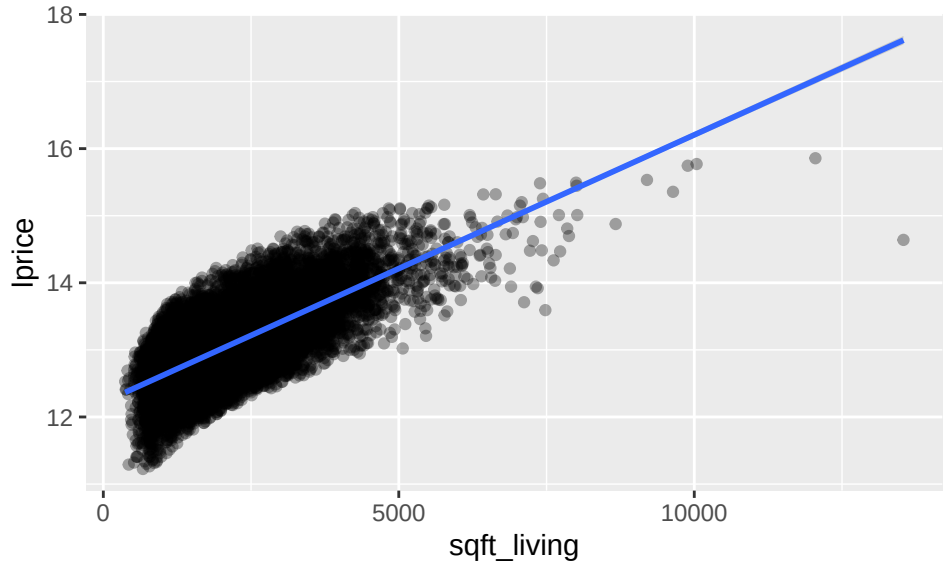
```
ggplot(kc.house, aes(x = bathrooms, y = lprice)) +
  geom_point(alpha = 1/3) +
  geom_smooth(method = "lm")
```



A higher bathroom to bedroom ratio leads to a higher price, although it again doesn't seem to hold for very high values of **bathrooms**. Perhaps this is not really a linear relationship, but we do not have enough values larger than 1.5 to really know.

Square footage of the home(sqft_living):

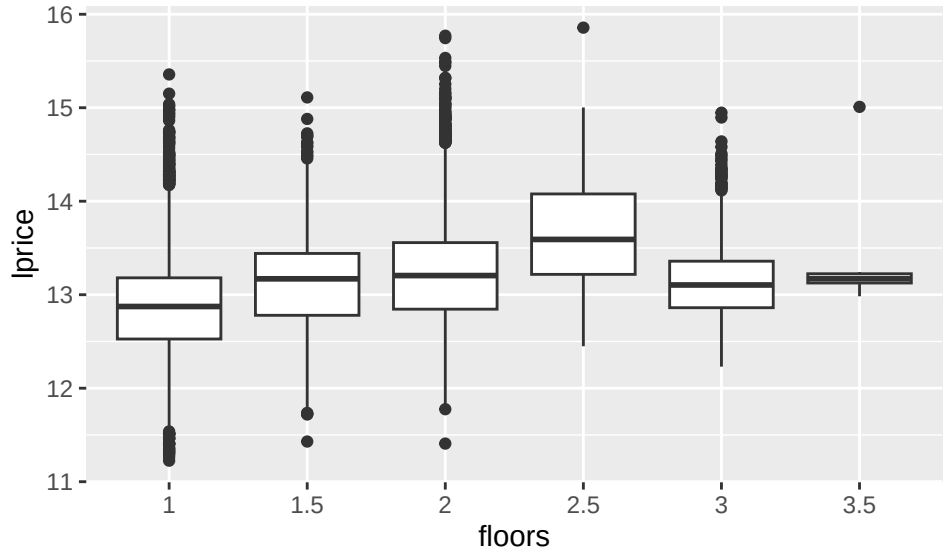
```
ggplot(kc.house, aes(x = sqft_living, y = lprice)) +
  geom_point(alpha = 1/3) +
  geom_smooth(method = "lm")
```



Here we see a very clear positive relationship between **sqft_living** and **lprice**. As before, we should be careful of interpreting this result for very high values of **sqft_living**, as there aren't many observations there.

Floors:

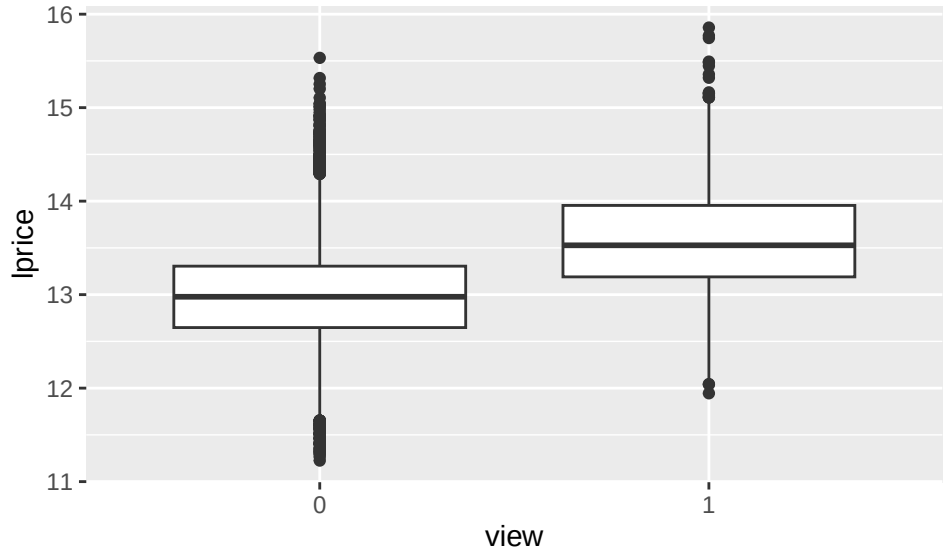
```
ggplot(kc.house, aes(x = as.factor(floors), y = lprice)) +
  geom_boxplot() +
  xlab("floors")
```



For **floors** and **lprice**, the relationship seems to be positive at first, but then it turns negative for higher values. A linear dependence may not be the most accurate in this case, although (once again) we can not be too sure because observations with higher values are more rare.

View:

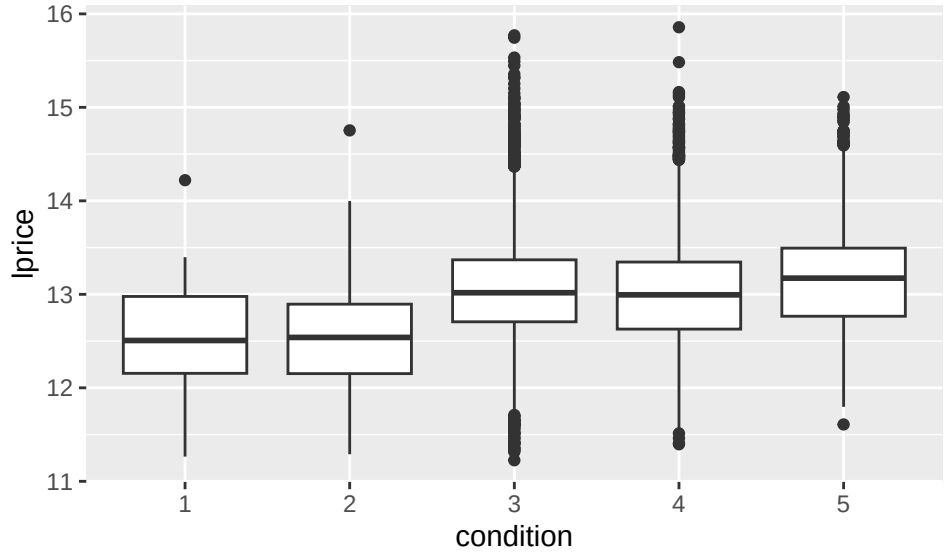
```
ggplot(kc.house, aes(x = as.factor(view), y = lprice)) +
  geom_boxplot() +
  xlab("view")
```



Houses that have been viewed have a higher price than houses that haven't been viewed.

Condition:

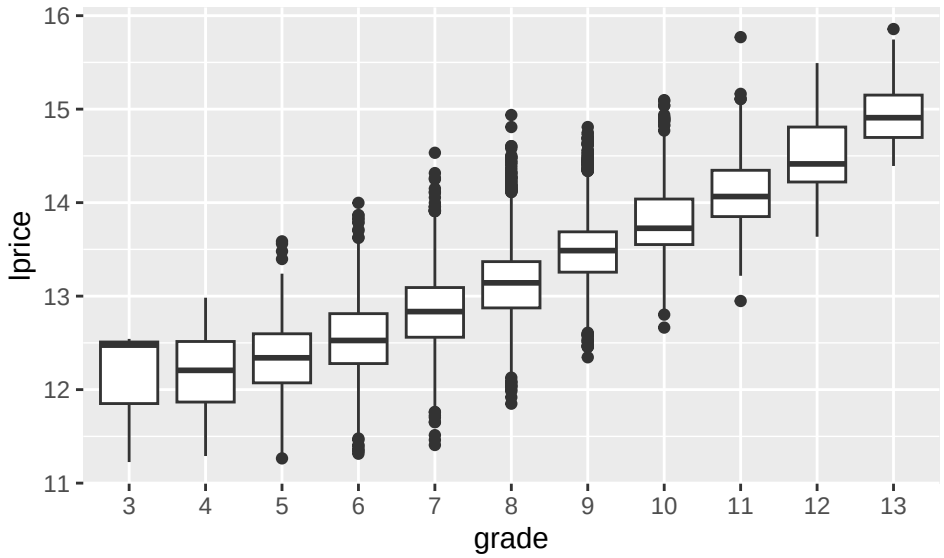
```
ggplot(kc.house, aes(x = as.factor(condition), y = lprice)) +
  geom_boxplot() +
  xlab("condition")
```



This looks like a mostly linear relationship: Better condition means higher price.

Grade:

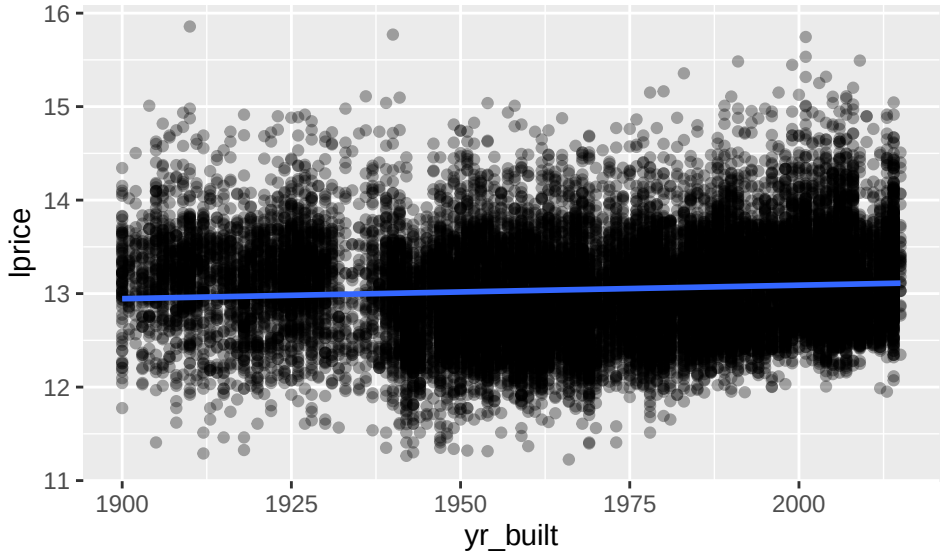
```
ggplot(kc.house, aes(x = as.factor(grade), y = lprice)) +
  geom_boxplot() +
  xlab("grade")
```



The same as with the variable **condition**: Higher grade, higher price.

Year the house was built (yr_built):

```
ggplot(kc.house, aes(x = yr_built, y = lprice)) +
  geom_point(alpha = 1/3) +
  geom_smooth(method = "lm")
```



There seems to be a rather weak positive relationship between **yr_built** and **lprice** as well.

Now, let's look at what happens when we add squared **yr_built** and **sqft_living**:

```
#adding new variables to the dataframe
#I add yr and sqvrt to df too,they will help to extend the 3rd model
#Further explanition of this choice will be below
#Where we will extend the third model
kc.house <- kc.house %>% mutate(yr_built2 = yr_built^2,
                                sqft_living2 = sqft_living^2,
                                sqvrt = log(sqft_living^2),
                                yr = log(yr_built^2))

#calculating new model
lmod2 <- lm(lprice ~ bedrooms + bathrooms + sqft_living + floors
            + view + condition + grade + yr_built
            + yr_built2 + sqft_living2, data = kc.house)
summary(lmod2)
```

```
##
## Call:
## lm(formula = lprice ~ bedrooms + bathrooms + sqft_living + floors +
##     view + condition + grade + yr_built + yr_built2 + sqft_living2,
##     data = kc.house)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3323 -0.2117  0.0128  0.2106  1.4044
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.656e+02  1.084e+01  15.277   <2e-16 ***
## bedrooms      8.305e-03  3.583e-03   2.318   0.0205 *
## bathrooms     2.188e-01  1.449e-02  15.100   <2e-16 ***
## sqft_living   2.775e-04  8.692e-06  31.927   <2e-16 ***
## floors        4.674e-02  5.638e-03   8.290   <2e-16 ***
## view          1.686e-01  7.645e-03  22.057   <2e-16 ***
## condition     4.734e-02  3.596e-03  13.165   <2e-16 ***
## grade         2.196e-01  3.102e-03  70.794   <2e-16 ***
## yr_built     -1.528e-01  1.107e-02 -13.804   <2e-16 ***
## yr_built2     3.759e-05  2.827e-06  13.297   <2e-16 ***
## sqft_living2 -1.590e-08  1.178e-09 -13.488   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3124 on 21589 degrees of freedom
## Multiple R-squared:  0.6482, Adjusted R-squared:  0.648
## F-statistic:  3978 on 10 and 21589 DF,  p-value: < 2.2e-16
```

Both of these new terms are highly significant. Interestingly, the variable **bedrooms** is a lot less significant in this model than in the previous ones, but still significant for $\alpha = 0.05$, which is the most common significance level of choice.

And what about R^2?

```
## [1] "lmod: r_squared"
```

```
## [1] 0.6423887
```

```
## [1] "lmod2: r_squared"
```

```
## [1] 0.6480413
```

The value for R^2 in this new model *lmod2* is somewhat higher than in the model before.

Prediction

Now we would like to compare how well models from b) and c) make prediction. For this we divide the dataset into a training and a test set, where we sample randomly 10 800 rows to include into the training set and test set (because we removed some observations in a), we have a few observations less and will therefore include 10800 observations each in the training and test dataset):

```
#randomly selecting the 10800 indices from the dataset to be part of the training dataset
train_ind <- sample(seq_len(nrow(kc.house)), size = 0.5*nrow(kc.house))

train <- kc.house[train_ind, ]#fill the train set
test <- kc.house[-train_ind, ]#fill the test set
```

We fit both models on the training set and make predictions on the test set:

```
#training model from b)
train_lmod <- lm(lprice ~ bedrooms + bathrooms + sqft_living
                 + floors + view + condition + grade
                 + yr_built, data = train)

#predicting model from b)
pred_lmod <- predict(train_lmod, test) #model from b)
pred_lmod[1:10]#quick review
```

```
##        1       10       13       14       15       16       17       19
## 12.66470 12.73844 12.95433 12.69175 13.17900 13.67087 12.74508 12.92639
##       20       21
## 12.65035 12.88432
```

```
#training model from c)
train_lmod2 <- lm(lprice ~ bedrooms + bathrooms + sqft_living
                  + floors + view + condition + grade
                  + yr_built + yr_built2
                  + sqft_living2, data = train)

#predicting model from c)
pred_lmod2 <- predict(train_lmod2, test)
pred_lmod2[1:10]#quick review
```

```
##        1       10       13       14       15       16       17       19
## 12.63770 12.75418 12.96248 12.67882 13.26008 13.64964 12.74263 12.95952
##       20       21
## 12.63403 12.87110
```

Now we calculate the mean squared difference between predicted values and values of log(price) from the test set for each model:

```
#MSE of the model from b)
mean((test$lprice - pred_lmod)^2)
```

```
## [1] 0.09879812
```

```
#MSE of the model from c)
mean((test$lprice - pred_lmod2)^2)
```

```
## [1] 0.09689574
```

We notice that the prediction error of the model from c) is smaller, what implies better prediction.

We can try to extend the model to improve the prediction. More better prediction would be if we would add another variable. As we have seen before, **yr_built2** and **sqft_living2** very good correlate with **log(price)**, so it makes sence to extend our model using these variables. We will add variables **yr** and **sqvrt**, which have log values of **yr_built2** and **sqft_living2** :

```
#we look at the new model
lmod3 <- lm(lprice ~ bedrooms + bathrooms  + yr_built
                  + floors + view + grade + condition
                   + yr_built2 + sqft_living2 + sqft_living +
                  yr+sqvrt, data = kc.house)
summary(lmod3)
```

```
##
## Call:
## lm(formula = lprice ~ bedrooms + bathrooms + yr_built + floors +
##     view + grade + condition + yr_built2 + sqft_living2 + sqft_living +
##     yr + sqvrt, data = kc.house)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.34729 -0.20822  0.01371  0.20743  1.37451
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.533e+05  1.161e+04 -21.818  < 2e-16 ***
## bedrooms      9.034e-03  3.627e-03   2.490 0.012766 *
## bathrooms     2.325e-01  1.447e-02  16.067  < 2e-16 ***
## yr_built     -4.268e+01  1.948e+00 -21.912  < 2e-16 ***
## floors        4.055e-02  5.584e-03   7.262 3.96e-13 ***
## view          1.590e-01  7.581e-03  20.977  < 2e-16 ***
```

```
## grade         2.257e-01  3.080e-03  73.281  < 2e-16 ***
## condition     4.730e-02  3.560e-03  13.284  < 2e-16 ***
## yr_built2     5.460e-03  2.483e-04  21.985  < 2e-16 ***
## sqft_living2 -8.210e-09  2.148e-09  -3.823 0.000132 ***
## sqft_living   1.617e-04  2.560e-05   6.317 2.71e-10 ***
## yr            2.084e+04  9.547e+02  21.832  < 2e-16 ***
## sqvrt         8.461e-02  1.666e-02   5.077 3.86e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3089 on 21587 degrees of freedom
## Multiple R-squared:  0.6559, Adjusted R-squared:  0.6557
## F-statistic:  3429 on 12 and 21587 DF,  p-value: < 2.2e-16
```

Both of these new terms are highly significant. $Adj.R^2 = 0.6557$, what is bigger than in previous models.

```
train_lmod3 <- lm(lprice ~ bedrooms + bathrooms + yr_built
                  + floors + view + grade + condition
                   + yr_built2 + sqft_living2 + sqft_living +
                  yr+sqvrt, data = train)

pred_lmod3 <- predict(train_lmod3, test)

#MSE of the model
mean((test$lprice - pred_lmod3)^2)
```

```
## [1] 0.09505739
```

We notice that the prediction error of this model is smaller than in c). The best prediction for the third model for me was 0.09375756.