STOR 455 Homework #5

40 points - Due Wednesday 3/20 at 5:00pm

Directions: For parts 7 and 10 you should work together, but these parts must be **submitted individually** by each group member. For parts 8 and 9, you must have only **one submission per group**. There will be separate places on Gradescope to submit the individual vs group work.

Situation: Can we predict the selling price of a house in Ames, Iowa based on recorded features of the house? That is your task for this assignment. Each team will get a dataset with information on forty potential predictors and the selling price (in $1,000's) for a sample of homes. The data sets for your group are AmesTrain??.csv and AmesTest??.csv (where ?? corresponds to your group number) A separate file identifies the variables in the Ames Housing data and explains some of the coding.

Part 7. Cross-validation: In some situations, a model might fit the peculiarities of a specific sample of data well, but not reflect structure that is really present in the population. A good test for how your model might work on "real" house prices can be simulated by seeing how well your fitted model does at predicting prices that were NOT in your original sample. This is why we reserved an additional 200 cases as a holdout sample in AmesTest??.csv. Use the group number and AmesTest??.csv corresponding to your group number for homework #3. Import your holdout test data and

- Compute the predicted Price for each of the cases in the holdout test sample, using your model resulting from the initial fit and residual analysis in parts 1 through 3 of Homework #3.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
AmesData <- read.csv('AmesTrain10.csv')

AmesData1 <- select(AmesData, is.numeric)
```

```
## Warning: Use of bare predicate functions was deprecated in tidyselect 1.1.0.
## i Please use wrap predicates in `where()` instead.
##   # Was:
##   data %>% select(is.numeric)
##
##   # Now:
##   data %>% select(where(is.numeric))
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
holdout <- read.csv('AmesTest10.csv')

reducedmodel1 <- lm(formula = Price ~ Quality + GroundSF + BasementFinSF + BasementSF +
    LotArea + YearBuilt + GarageSF + YearRemodel + Bedroom +
    LotFrontage + FullBath + Condition, data = AmesData1)

test_data <- predict(reducedmodel1, newdata = holdout)
```

- Compute the residuals for the 200 holdout cases.

```r
test_resid <- holdout$Price - test_data
```

- Compute the mean and standard deviation of these residuals. Are they close to what you expect from the training model?

```r
mean(test_resid)
```

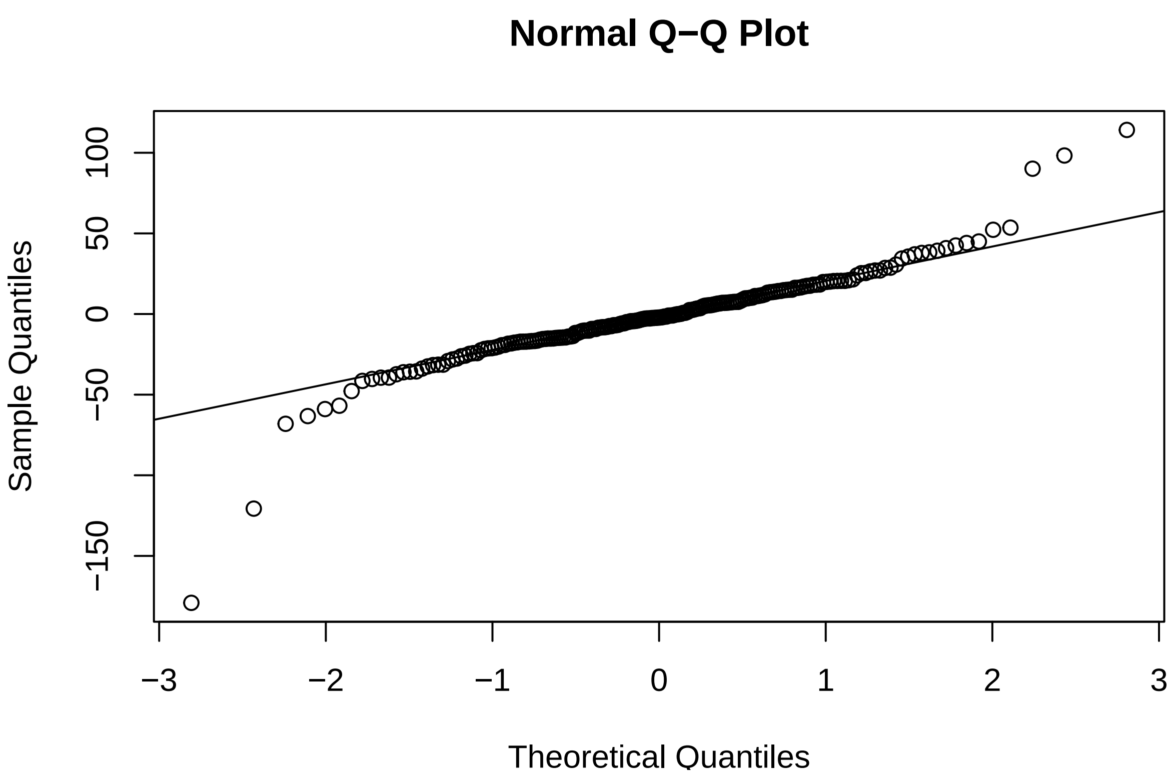
```
## [1] -1.631249
```

```r
sd(test_resid)
```

```
## [1] 28.98615
```

- Construct a plot of the residuals to determine if they are normally distributed. Is this plot what you expect to see considering the training model?

```r
qqnorm(test_resid)
qqline(test_resid)
```



- Are any holdout cases especially poorly predicted by the training model? If so, identify by the row number(s) in the holdout data. Why might these cases be poorly predicted?

```r
sort(abs(test_resid), decreasing = TRUE)[1:5]
```

```
##       118        48        57       175       186
```

```
## 179.10581 120.69174 114.15403  98.29246  90.12243
```

- Compute the correlation between the predicted values and actual prices for the holdout sample. This is known as the cross-validation correlation. We don't expect the training model to do better at predicting values different from those that were used to build it (as reflected in the original R^2), but an effective model shouldn't do a lot worse at predicting the holdout values. Square the cross-validation correlation to get an R^2 value and subtract it from the original multiple R^2 of the training sample. This is known as the shrinkage. We won't have specific rules about how little the shrinkage should be, but give an opinion on whether the shrinkage looks OK to you or too large in your situation.

```r
summary(reducedmodel1)$r.squared - cor(holdout$Price,test_data)^2
```

```
## [1] 0.0331965
```

Part 8. Find a "fancy model": Again using AmesTrain??.csv, where ?? corresponds to your new group number in homework #5, to build a regression model to predict Price. In addition to the quantitative predictors, you may now consider models with

```r
NewAmesTrain <- read.csv('AmesTest10.csv', stringsAsFactors = TRUE)
NewAmesTrain <- na.omit(NewAmesTrain)
NewAmesTrain <- NewAmesTrain[-1]
age_built <- 2010 - NewAmesTrain$YearBuilt
age_remodel <- 2010 - NewAmesTrain$YearRemodel

full <- lm(Price ~ ., data=NewAmesTrain)
#summary(full)
```

- Categorical variables - Just put these in the model and let R take care of making the indicator predictors (and picking one category to leave out). Use factor( ) to treat a numeric variable as categorical. You'll see the coefficients for each indicator when you look at the summary( ) and they will be grouped together in the ANOVA. Be careful, since adding a single categorical variable with a lot of categories might actually be adding a lot of new indicator terms.

```r
NewAmesTrain_wfactor <- NewAmesTrain
NewAmesTrain_wfactor$Quality <- factor(NewAmesTrain_wfactor$Quality)
NewAmesTrain_wfactor$Condition <- factor(NewAmesTrain_wfactor$Condition)

full_wfactor <- lm(Price ~ ., data=NewAmesTrain_wfactor)
summary(full_wfactor)
#plot(Price ~ ., data=NewAmesTrain_wfactor)
```

- Transformations of predictors - You can include functions of quantitative predictors. Probably best to use the I( ) notation so you don't need to create new columns when you run the predictions for the test data.
- Transformations of the response - You might address curvature or skewness in residual plots by transforming the response prices with a function like log(Price), sqrt(Price), Price^2, etc.. These should generally not need the I( ) notation to make these adjustments. IMPORTANT: If you transform Price, be sure to reverse the transformation when making final predictions!
- Combinations of variables - This might include interactions or other combinations. You do not need the I( ) notation when making an interaction using a categorical predictor (e.g. GroundSF*CentralAir).

Keep general track of the approaches you try and explain what guides your decisions as you select a new set of predictors (but again you don't need to give full details of every model you consider). Along the way you should consider some residual analysis.

Notes/Tips:

- WARNING: When using a categorical predictor with multiple categories in regsubsets( ), R will create indicators and treat them as separate predictors when deciding which to put into a model. So you might get a model with quantitative predictors like LotArea and GroundSF along with specific indicators like GarageQTA and HouseStyle1Story. This may not be very useful, since we should generally use all indicators for a categorical predictor if we include one in the model. On the other hand, when using the step( ) function, R will generally keep the multiple indicators for different categories of the same variable together as a unit.

- In some cases the indicators created for different categorical variables will have identical values. For example, if you include both GarageC and GarageQ in a model, R will produce values for each of the indicators. The indicators for GarageQNone and GarageCNone (equal to one only for houses that don't have a garage) will be identical. This may be handled differently in R depending on the procedure. regsubsets( ) may give a "warning" about variables being linearly dependent. You can still use the results, just be aware that some variables are completely dependent. lm( ) might give output with coefficients (and tests) of some predictors listed as NA. This is not a problem, R is just automatically deleting one of the redundant variables. If you are predicting for a house with no garage you might have a coefficient to use for GarageQNone but then you don't need to worry about having one for GarageCNone.
- If your residual analysis from homework #3 or an early model here suggest you might want to do a transformation for the response variable (Price), do so *before* fitting a lot more models. No sense fine tuning a set of predictors for Price, then deciding you should be predicting log(Price) or Price^2. So make that decision fairly early, but don't get too picky and expect to get perfect plot of residuals versus fits or an exact normal quantile plot.
- Similarly, if you decide that some data cases should be dropped from the training set, don't wait until late in the process to do so. For example, if you spot a *very* large residual you should look at the characteristics for that house to see if it should be deleted. Don't forget about the value of simple plots (like a scatterplot of Price vs. LotArea) for helping to see what is going on and recognize extreme cases. Be sure to document any adjustments you make in the final report.
- Comparing C_p from different predictor pools - While Mallow's C_p is a useful tool for comparing models from the same pool of predictors. You should not use it to compare models based on different predictor pools. For example, if you add a bunch of categorical variables to all the quantitative predictors from homework #3 to make a new "full" model, then find C_p from a model that you fit in homework #3, it will be worse than it was before. If you look at the formula for calculating C_p, you will see that all that has changed is MSE for the full model after adding the new batch of predictors.
- I should be able to follow the steps you use when selecting a model. I certainly don't need to see every bit of output, but it might help to include more of the R commands you use. For example, saying you used backward elimination is not very helpful when I don't know what you start with for the full model or pool of predictors (e.g. did you include Condition and Quality as numeric predictors? or did you decide to eliminate one of GroundSF, FirstSF, or SecondSF due to redundancy?). The easiest way to convey this in many cases is to show the R command you used. It is fine to abbreviate the output (for example, delete many steps in a stepwise procedure using trace=FALSE), but it would be helpful if you identified the parts you do include. For example, a sentence like "After 12 steps of the stepwise procedure, we have the output below for the fitted model." Similarly, I don't need to see 600 residuals, using head and sort can show the important ones.
- Once you have settled on a response, made adjustments to the data (if needed), and chosen a set of predictors, be sure to include the summary( )for your "fancy" model at this stage.

```r
DoublenewNewAmesTrain <- NewAmesTrain_wfactor %>%
  mutate(Price = log(Price),
         LotArea = log(LotArea))

first_model <- lm(log(Price) ~ . + I(age_built^2) + I(GarageCars^2)+ I(GarageSF^2)+ TotalRooms * Bedroom

summary(first_model)
```

```
##
## Call:
```

```
## lm(formula = log(Price) ~ . + I(age_built^2) + I(GarageCars^2) +
##     I(GarageSF^2) + TotalRooms * Bedroom + LotArea * LotFrontage +
##     GarageSF * GarageCars, data = DoublenewNewAmesTrain)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.037757 -0.009151  0.000000  0.008070  0.056877
##
## Coefficients: (7 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.549e-01  8.815e-01   0.289 0.773003
## LotFrontage         -1.189e-03  1.201e-03  -0.990 0.324216
## LotArea              1.609e-02  5.878e-03   2.737 0.007257 **
## LotConfigCulDSac     1.185e-03  8.151e-03   0.145 0.884657
## LotConfigFR2        -1.565e-02  1.044e-02  -1.498 0.136932
## LotConfigFR3        -2.926e-02  2.940e-02  -0.995 0.321838
## LotConfigInside      8.673e-04  4.667e-03   0.186 0.852929
## HouseStyle1Story     5.682e-03  9.855e-03   0.577 0.565436
## HouseStyle2.5Unf    -1.600e-02  2.215e-02  -0.723 0.471502
## HouseStyle2Story    -1.219e-02  8.393e-03  -1.453 0.149182
## HouseStyleSFoyer    -1.448e-03  1.246e-02  -0.116 0.907722
## HouseStyleSLvl       1.322e-03  1.214e-02   0.109 0.913512
## Quality4             2.856e-02  1.679e-02   1.701 0.091848 .
## Quality5             5.478e-02  1.680e-02   3.261 0.001488 **
## Quality6             6.157e-02  1.716e-02   3.588 0.000503 ***
## Quality7             7.508e-02  1.791e-02   4.191 5.70e-05 ***
## Quality8             8.514e-02  1.926e-02   4.421 2.35e-05 ***
## Quality9             1.304e-01  2.503e-02   5.211 9.09e-07 ***
## Condition4           1.619e-01  3.331e-02   4.862 3.98e-06 ***
## Condition5           1.724e-01  3.265e-02   5.281 6.72e-07 ***
## Condition6           1.795e-01  3.225e-02   5.567 1.91e-07 ***
## Condition7           1.908e-01  3.279e-02   5.820 6.11e-08 ***
## Condition8           1.992e-01  3.324e-02   5.994 2.75e-08 ***
## Condition9           2.333e-01  3.759e-02   6.205 1.03e-08 ***
## YearBuilt            4.640e-04  5.139e-04   0.903 0.368554
## YearRemodel         -2.584e-05  1.691e-04  -0.153 0.878799
## ExteriorQFa         -3.655e-03  4.731e-02  -0.077 0.938555
## ExteriorQGd          2.803e-02  1.883e-02   1.489 0.139498
## ExteriorQTA          3.429e-02  1.987e-02   1.725 0.087303 .
## ExteriorCFa         -6.090e-03  3.290e-02  -0.185 0.853521
## ExteriorCGd         -3.451e-02  2.639e-02  -1.308 0.193737
## ExteriorCTA         -1.788e-02  2.612e-02  -0.685 0.495029
## FoundationCBlock     5.478e-03  9.237e-03   0.593 0.554422
## FoundationPConc      1.517e-02  9.726e-03   1.559 0.121815
## FoundationSlab       4.545e-02  2.680e-02   1.696 0.092755 .
## FoundationStone      5.671e-02  2.557e-02   2.218 0.028635 *
## FoundationWood      -3.488e-02  2.698e-02  -1.292 0.198978
## BasementHtEx         7.754e-03  1.131e-02   0.686 0.494455
## BasementHtFa         1.041e-02  1.507e-02   0.691 0.491268
## BasementHtGd         6.147e-03  6.808e-03   0.903 0.368536
## BasementHtNone      -8.456e-03  2.771e-02  -0.305 0.760830
## BasementHtTA                NA         NA      NA       NA
## BasementCFa         -7.417e-03  1.285e-02  -0.577 0.565105
## BasementCGd         -5.372e-04  6.765e-03  -0.079 0.936851
## BasementCNone               NA         NA      NA       NA
```

```
## BasementCTA                   NA         NA      NA       NA
## BasementFinALQ         8.061e-04  8.288e-03   0.097 0.922700
## BasementFinBLQ        -1.177e-03  8.922e-03  -0.132 0.895325
## BasementFinGLQ         6.090e-03  7.536e-03   0.808 0.420740
## BasementFinLwQ        -8.512e-05  1.024e-02  -0.008 0.993381
## BasementFinNone               NA         NA      NA       NA
## BasementFinRec        -3.509e-03  1.047e-02  -0.335 0.738189
## BasementFinUnf                NA         NA      NA       NA
## BasementFinSF          7.867e-06  1.530e-05   0.514 0.608187
## BasementUnFinSF       -3.323e-06  1.565e-05  -0.212 0.832293
## BasementSF             4.842e-05  2.231e-05   2.170 0.032190 *
## HeatingGasW            3.306e-03  1.512e-02   0.219 0.827378
## HeatingQCFa           -5.978e-03  1.341e-02  -0.446 0.656746
## HeatingQCGd            3.582e-03  5.696e-03   0.629 0.530717
## HeatingQCTA           -7.864e-04  6.049e-03  -0.130 0.896799
## CentralAirY            9.262e-04  1.207e-02   0.077 0.938995
## FirstSF                7.436e-05  5.611e-05   1.325 0.187946
## SecondSF               1.130e-04  5.833e-05   1.938 0.055250 .
## GroundSF              -4.273e-05  5.421e-05  -0.788 0.432299
## BasementFBath         -7.958e-05  5.255e-03  -0.015 0.987946
## BasementHBath         -2.250e-03  1.304e-02  -0.173 0.863333
## FullBath               1.524e-03  6.154e-03   0.248 0.804863
## HalfBath              -5.233e-03  6.119e-03  -0.855 0.394329
## Bedroom               -7.439e-03  1.112e-02  -0.669 0.504933
## KitchenQFa            -7.219e-02  2.499e-02  -2.889 0.004677 **
## KitchenQGd            -3.507e-03  9.821e-03  -0.357 0.721743
## KitchenQTA            -1.161e-02  1.109e-02  -1.047 0.297615
## TotalRooms             4.379e-03  5.170e-03   0.847 0.398837
## Fireplaces             5.128e-03  3.725e-03   1.377 0.171477
## GarageTypeAttchd      -1.615e-02  3.011e-02  -0.537 0.592707
## GarageTypeBasment      2.585e-04  3.488e-02   0.007 0.994100
## GarageTypeBuiltIn     -2.466e-02  3.154e-02  -0.782 0.436131
## GarageTypeCarPort     -6.584e-02  3.503e-02  -1.879 0.062896 .
## GarageTypeDetchd      -7.214e-03  2.858e-02  -0.252 0.801211
## GarageTypeNone        -5.605e-02  5.011e-02  -1.119 0.265760
## GarageCars             5.521e-03  1.993e-02   0.277 0.782308
## GarageSF              -1.228e-05  7.075e-05  -0.174 0.862473
## GarageQGd             -6.876e-03  1.972e-02  -0.349 0.728045
## GarageQNone                   NA         NA      NA       NA
## GarageQTA             -2.032e-02  1.276e-02  -1.593 0.114182
## GarageCFa              3.552e-02  3.176e-02   1.118 0.265914
## GarageCGd             -4.147e-03  3.806e-02  -0.109 0.913443
## GarageCNone                   NA         NA      NA       NA
## GarageCTA              2.467e-02  2.843e-02   0.868 0.387468
## WoodDeckSF            -1.288e-06  1.650e-05  -0.078 0.937917
## OpenPorchSF            7.510e-07  2.967e-05   0.025 0.979849
## EnclosedPorchSF       -5.379e-06  3.137e-05  -0.171 0.864172
## ScreenPorchSF         -1.704e-05  4.245e-05  -0.401 0.688952
## I(age_built^2)        -2.133e-06  4.228e-06  -0.504 0.615020
## I(GarageCars^2)       -1.267e-03  1.018e-02  -0.124 0.901231
## I(GarageSF^2)          1.638e-08  1.353e-07   0.121 0.903868
## Bedroom:TotalRooms     9.633e-05  1.513e-03   0.064 0.949345
## LotFrontage:LotArea    1.230e-04  1.278e-04   0.963 0.337778
## GarageCars:GarageSF   -1.532e-06  6.449e-05  -0.024 0.981094
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01959 on 108 degrees of freedom
## Multiple R-squared:  0.9602, Adjusted R-squared:  0.9267
## F-statistic: 28.64 on 91 and 108 DF,  p-value: < 2.2e-16
```

#Firstly, we opted to transform the price variable by taking its logarithm. This decision was based on its ability to mitigate the influence of extreme values, thus enhancing the fit of our model, particularly with respect to its curvature.

#Similarly, we applied a logarithmic transformation to the LotArea variable. Our rationale behind this choice stems from observing a curving trend within the data, which logarithmic adjustment tends to alleviate effectively.

#For variables such as age built and garage cars, which exhibited polynomial trends according to our visual inspection of scatter plots, we decided to employ polynomial transformations. This method allowed us to capture the non-linear relationships more accurately, thereby improving the model's performance.

#Additionally, recognizing the high correlation between total rooms and total bedrooms, we deemed it essential to include an interaction term between these two variables. By doing so, we aimed to capture any nuanced interplay between these features, thus enriching the model's predictive capacity.

#Similarly, we identified a significant correlation between lot area and lot frontage, prompting us to incorporate an interaction term for these variables as well. This approach enables us to account for their mutual influence more effectively, enhancing the model's robustness.

#Finally, given the strong correlation between garage cars and garage square footage, we decided to handle them in a similar manner, introducing an interaction term to capture their combined effect accurately.

```r
MSE2 <- (summary(first_model)$sigma)^2

AmesModNone2 <- lm(Price ~ 1, NewAmesTrain_wfactor)

#simple_model<-step(first_model, scale = MSE2)
#step(AmesModNone2, scope = list(upper=first_model), scale=MSE2, direction='forward')
simple_model <- step(AmesModNone2, scale = MSE2, scope = list(upper = first_model), trace = FALSE)

summary(simple_model)
```

```
##
## Call:
## lm(formula = Price ~ Quality + GroundSF + Condition + YearBuilt +
##     BasementFinSF + ExteriorQ + HouseStyle + GarageSF + Foundation +
##     LotArea + LotConfig + GarageType + Bedroom + TotalRooms +
##     ExteriorC + BasementHBath + I(GarageSF^2) + GarageC + HeatingQC +
##     BasementSF + BasementC + BasementHt + KitchenQ + GarageQ +
##     ScreenPorchSF + Fireplaces + WoodDeckSF + FullBath + HalfBath +
##     SecondSF + FirstSF + BasementFin + BasementUnFinSF + BasementFBath +
##     YearRemodel + Heating + EnclosedPorchSF + LotFrontage + I(age_built^2) +
##     GarageCars + I(GarageCars^2) + CentralAir + OpenPorchSF +
##     Bedroom:TotalRooms + LotArea:LotFrontage + GarageSF:GarageCars,
##     data = NewAmesTrain_wfactor)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.069  -9.990   0.000   8.393  74.343
##
## Coefficients: (7 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.767e+03  9.761e+02  -1.810 0.073036 .
```

```
## Quality4            -3.349e+00  1.854e+01  -0.181 0.856997
## Quality5            -2.475e+00  1.851e+01  -0.134 0.893861
## Quality6             7.226e+00  1.892e+01   0.382 0.703224
## Quality7             1.704e+01  1.977e+01   0.862 0.390433
## Quality8             3.056e+01  2.102e+01   1.454 0.148874
## Quality9             5.414e+01  2.717e+01   1.993 0.048805 *
## GroundSF            -1.442e-01  5.930e-02  -2.431 0.016699 *
## Condition4           1.672e+02  3.597e+01   4.648 9.54e-06 ***
## Condition5           1.787e+02  3.531e+01   5.060 1.73e-06 ***
## Condition6           1.820e+02  3.475e+01   5.237 8.11e-07 ***
## Condition7           1.956e+02  3.559e+01   5.496 2.62e-07 ***
## Condition8           2.063e+02  3.611e+01   5.712 9.95e-08 ***
## Condition9           2.394e+02  4.139e+01   5.784 7.20e-08 ***
## YearBuilt            1.009e+00  5.681e-01   1.775 0.078652 .
## BasementFinSF        1.211e-02  1.697e-02   0.714 0.476768
## ExteriorQFa         -1.006e+02  5.207e+01  -1.931 0.056096 .
## ExteriorQGd         -3.216e+01  2.130e+01  -1.510 0.133973
## ExteriorQTA         -3.232e+01  2.239e+01  -1.443 0.151900
## HouseStyle1Story     4.551e+00  1.143e+01   0.398 0.691239
## HouseStyle2.5Unf    -1.797e+00  2.444e+01  -0.074 0.941530
## HouseStyle2Story    -1.110e+01  9.119e+00  -1.217 0.226263
## HouseStyleSFoyer    -5.681e+00  1.414e+01  -0.402 0.688709
## HouseStyleSLvl      -3.976e+00  1.389e+01  -0.286 0.775163
## GarageSF            -2.664e-02  7.833e-02  -0.340 0.734419
## FoundationCBlock     4.698e+00  1.008e+01   0.466 0.642071
## FoundationPConc      1.302e+01  1.074e+01   1.213 0.227961
## FoundationSlab       1.687e+01  2.941e+01   0.574 0.567498
## FoundationStone      2.450e+01  2.806e+01   0.873 0.384471
## FoundationWood      -4.161e+01  2.966e+01  -1.403 0.163481
## LotArea              1.681e-04  1.836e-04   0.915 0.362050
## LotConfigCulDSac     7.015e+00  9.241e+00   0.759 0.449452
## LotConfigFR2        -2.541e+01  1.139e+01  -2.231 0.027745 *
## LotConfigFR3        -2.713e+01  3.240e+01  -0.837 0.404311
## LotConfigInside     -4.574e-01  5.062e+00  -0.090 0.928169
## GarageTypeAttchd    -2.888e+01  3.331e+01  -0.867 0.387897
## GarageTypeBasment   -2.999e+01  3.840e+01  -0.781 0.436457
## GarageTypeBuiltIn   -2.774e+01  3.479e+01  -0.798 0.426872
## GarageTypeCarPort   -7.833e+01  3.884e+01  -2.017 0.046218 *
## GarageTypeDetchd    -2.258e+01  3.166e+01  -0.713 0.477369
## GarageTypeNone      -6.769e+01  5.545e+01  -1.221 0.224847
## Bedroom             -2.395e+01  1.214e+01  -1.972 0.051110 .
## TotalRooms           1.263e+00  5.616e+00   0.225 0.822567
## ExteriorCFa         -2.703e+01  3.631e+01  -0.744 0.458204
## ExteriorCGd         -1.109e+01  2.904e+01  -0.382 0.703360
## ExteriorCTA         -8.596e-01  2.876e+01  -0.030 0.976209
## BasementHBath       -1.699e+01  1.434e+01  -1.185 0.238749
## I(GarageSF^2)       -6.440e-05  1.493e-04  -0.431 0.666981
## GarageCFa            8.627e+00  3.483e+01   0.248 0.804817
## GarageCGd           -3.961e+01  4.208e+01  -0.941 0.348680
## GarageCNone                 NA         NA      NA       NA
## GarageCTA            7.820e+00  3.140e+01   0.249 0.803827
## HeatingQCFa          8.185e+00  1.460e+01   0.561 0.576229
## HeatingQCGd          1.279e+01  6.303e+00   2.029 0.044966 *
## HeatingQCTA          4.616e+00  6.657e+00   0.693 0.489569
## BasementSF           3.224e-02  2.461e-02   1.310 0.192923
```

```
## BasementCFa         -1.187e+01  1.411e+01  -0.841 0.402300
## BasementCGd         -7.052e+00  7.456e+00  -0.946 0.346333
## BasementCNone       -4.743e+00  3.041e+01  -0.156 0.876369
## BasementCTA                 NA         NA      NA       NA
## BasementHtEx         9.074e+00  1.244e+01   0.729 0.467506
## BasementHtFa         1.048e+01  1.654e+01   0.633 0.527864
## BasementHtGd         6.876e+00  7.472e+00   0.920 0.359471
## BasementHtNone              NA         NA      NA       NA
## BasementHtTA                NA         NA      NA       NA
## KitchenQFa          -3.605e+01  2.741e+01  -1.315 0.191239
## KitchenQGd          -6.624e+00  1.081e+01  -0.613 0.541368
## KitchenQTA          -8.142e+00  1.218e+01  -0.669 0.505132
## GarageQGd           -8.199e+00  2.151e+01  -0.381 0.703840
## GarageQNone                 NA         NA      NA       NA
## GarageQTA           -1.128e+01  1.375e+01  -0.820 0.413753
## ScreenPorchSF       -3.934e-02  4.675e-02  -0.842 0.401831
## Fireplaces           4.572e+00  4.099e+00   1.115 0.267191
## WoodDeckSF           3.515e-03  1.848e-02   0.190 0.849472
## FullBath            -6.706e+00  6.741e+00  -0.995 0.322085
## HalfBath            -1.375e+01  6.750e+00  -2.037 0.044100 *
## SecondSF             2.216e-01  6.379e-02   3.474 0.000739 ***
## FirstSF              2.075e-01  6.105e-02   3.398 0.000951 ***
## BasementFinALQ      -1.421e+00  8.953e+00  -0.159 0.874163
## BasementFinBLQ      -4.280e+00  9.850e+00  -0.435 0.664770
## BasementFinGLQ      -3.534e+00  8.305e+00  -0.425 0.671338
## BasementFinLwQ      -6.426e+00  1.131e+01  -0.568 0.571240
## BasementFinNone             NA         NA      NA       NA
## BasementFinRec      -9.143e-01  1.155e+01  -0.079 0.937041
## BasementFinUnf              NA         NA      NA       NA
## BasementUnFinSF     -1.819e-02  1.725e-02  -1.055 0.293942
## BasementFBath       -3.908e+00  5.878e+00  -0.665 0.507549
## YearRemodel         -1.235e-01  1.869e-01  -0.661 0.510279
## HeatingGasW         -1.779e+01  1.674e+01  -1.063 0.290187
## EnclosedPorchSF     -2.222e-02  3.462e-02  -0.642 0.522453
## LotFrontage         -2.930e-01  1.399e-01  -2.094 0.038602 *
## I(age_built^2)       2.779e-03  4.658e-03   0.597 0.552070
## GarageCars          -1.699e+01  2.196e+01  -0.774 0.440912
## I(GarageCars^2)     -6.113e+00  1.124e+01  -0.544 0.587617
## CentralAirY          2.857e+00  1.324e+01   0.216 0.829516
## OpenPorchSF         -3.777e-03  3.253e-02  -0.116 0.907781
## Bedroom:TotalRooms   1.649e+00  1.654e+00   0.997 0.320916
## LotArea:LotFrontage  2.398e-05  8.872e-06   2.703 0.007987 **
## GarageSF:GarageCars  6.718e-02  7.113e-02   0.944 0.347033
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.63 on 108 degrees of freedom
## Multiple R-squared:  0.9454, Adjusted R-squared:  0.8994
## F-statistic: 20.56 on 91 and 108 DF,  p-value: < 2.2e-16
```

```r
ames10_holdout <- read.csv('AmesTest10.csv')
ames10_holdout$Quality <- factor(ames10_holdout$Quality)
ames10_holdout$Condition <- factor(ames10_holdout$Condition)
```

Part 9: Cross-validation for your "fancy" model We will compute the predicted Price for each of the cases in the holdout test sample, using our new model.

```
predictions <- predict(simple_model, newdata = ames10_holdout)
head(predictions)
```

```
##        1        2        3        4        5        6
## 238.1032 138.1705 185.0101 186.9975 172.7794 131.7513
```

As in part 7 we will compute the residuals for the 200 holdout cases.

```
holdout_resid <- log(ames10_holdout$Price) - predictions
```

And find the mean and standard deviation of these residuals.

```
mean(holdout_resid)
```

```
## [1] -169.0262
```

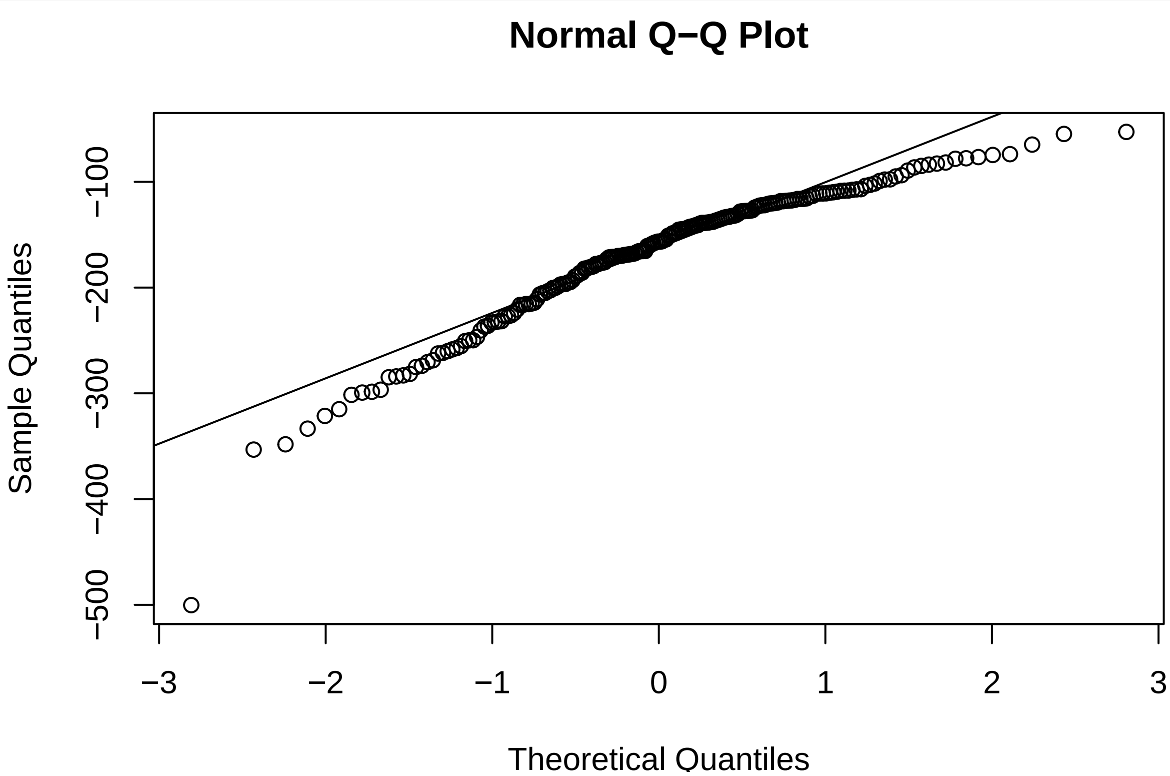
```
sd(holdout_resid)
```

```
## [1] 65.96771
```

The mean value of our residuals is incredibly close to zero which is ideal. The standard deviation of .0766 seems very low but this is in terms of log(Price) which is much smaller than price.

We will construct a plot to determine if they are normally distributed.

```
qqnorm(holdout_resid)
qqline(holdout_resid)
```



The residuals appear much more normally distributed than in previous models. There are potentially one or two extreme points.

Compute the correlation between the predicted values and actual prices for the holdout sample. This is known as the cross-validation correlation.

```r
cor(predictions, ames10_holdout$Price)
```

```
## [1] 0.9723245
```

```r
cor(predictions, log(ames10_holdout$Price))
```

```
## [1] 0.947947
```

Square the cross-validation correlation to get an R^2 value.

```r
cor(predictions, ames10_holdout$Price)^2
```

```
## [1] 0.9454149
```

Now subtract it from the original multiple R^2 of the training sample to find the shrinkage.

```r
summary(simple_model)$r.squared - cor(predictions, ames10_holdout$Price)^2
```

```
## [1] 1.110223e-16
```

The shrinkage is very small so it appears we are not over fitting our training data.

Part 10. Final Model Again, you may choose to make some additional adjustments to your model after considering the final residual analysis. If you do so, please explain what (and why) you did and provide the summary() for your new final model.

```r
Price ~ Quality + GroundSF + Condition + YearBuilt +
    BasementFinSF + ExteriorQ + HouseStyle + GarageSF + Foundation +
    LotArea + LotConfig + GarageType + Bedroom + TotalRooms +
    ExteriorC + BasementHBath + I(GarageSF^2) + GarageC + HeatingQC +
    BasementSF + BasementC + BasementHt + KitchenQ + GarageQ +
    ScreenPorchSF + Fireplaces + WoodDeckSF + FullBath + HalfBath +
    SecondSF + FirstSF + BasementFin + BasementUnFinSF + BasementFBath +
    YearRemodel + Heating + EnclosedPorchSF + LotFrontage + I(age_built^2) +
    GarageCars + I(GarageCars^2) + CentralAir + OpenPorchSF +
    Bedroom:TotalRooms + LotArea:LotFrontage + GarageSF:GarageCars,
    data = NewAmesTrain_wfactor
```

Suppose that you are interested in a house in Ames that has the characteristics listed below. Construct a 95% confidence interval for the mean price of such houses.

A 2 story 11 room home, built in 1987 and remodeled in 1999 on a 21540 sq. ft. lot with 328 feet of road frontage. Overall quality is good (7) and condition is average (5). The quality and condition of the exterior are both good (Gd) and it has a poured concrete foundation. There is an 757 sq. foot basement that has excellent height, but is completely unfinished and has no bath facilities. Heating comes from a gas air furnace that is in excellent condition and there is central air conditioning. The house has 2432 sq. ft. of living space above ground, 1485 on the first floor and 947 on the second, with 4 bedrooms, 2 full and one half baths, and 1 fireplace. The 2 car, built-in garage has 588 sq. ft. of space and is average (TA) for both quality and construction. The only porches or decks is a 205 sq. ft. open porch in the front.

```r
new_house_ex <- data.frame(
  Order = 0,
  Price = 0,
  LotFrontage = 328,
  LotArea = 21540,
  LotConfig = NA,
  HouseStyle = "2Story",
  Quality = 7,
  Condition = 5,
  YearBuilt = 1987,
```

```r
  YearRemodel = 1999,
  age_built = 2010 - 1987,
  age_remodel = 2010 - 1999,
  ExteriorQ = "Gd",
  ExteriorC = "Gd",
  Foundation = "PConc",
  BasementHt = "Ex",
  BasementC = "None",
  BasementFin = "Unf",
  BasementFinSF = 0,
  BasementUnFinSF = 757,
  BasementSF = 757,
  Heating = "GasA",
  HeatingQC = "Ex",
  CentralAir = "Y",
  FirstSF = 1485,
  SecondSF = 947,
  GroundSF = 2432,
  BasementFBath = 0,
  BasementHBath = 0,
  FullBath = 2,
  HalfBath = 1,
  Bedroom = 4,
  KitchenQ = NA,
  TotalRooms = 11,
  Fireplaces = 1,
  GarageType = "BuiltIn",
  GarageCars = 2,
  GarageSF = 588,
  GarageQ = "TA",
  GarageC = "TA",
  WoodDeckSF = 0,
  OpenPorchSF = 205,
  EnclosedPorchSF = 0,
  ScreenPorchSF = 0
)

house_ex <- data.frame(new_house_ex, stringsAsFactors = TRUE)

predict.lm(simple_model, new_data = house_ex, interval = 'confidence', level = 0.95)
```

```
##          fit       lwr       upr
## 1   238.10315 215.00708 261.19923
## 2   138.17050 108.84123 167.49977
## 3   185.01011 169.69790 200.32232
## 4   186.99751 156.51768 217.47733
## 5   172.77941 149.18516 196.37365
## 6   131.75126 105.46419 158.03832
## 7   267.92839 245.93027 289.92651
## 8   185.78926 164.16522 207.41330
## 9   136.46716 116.12777 156.80654
## 10   98.30000  55.42534 141.17466
## 11  125.23664  90.40281 160.07047
## 12  147.35103 120.39318 174.30887
## 13  178.30778 149.65838 206.95718
```

```
## 14  245.89066 224.05794 267.72338
## 15  161.74613 139.86511 183.62715
## 16  289.64840 264.73465 314.56214
## 17  171.67475 148.68351 194.66599
## 18  288.66402 254.44704 322.88099
## 19  176.44757 144.99253 207.90262
## 20  175.19712 144.61307 205.78117
## 21  127.00000  84.12534 169.87466
## 22  290.52981 272.64251 308.41711
## 23  123.12901 101.79972 144.45831
## 24   99.63442  71.19392 128.07492
## 25  174.50203 144.47845 204.52561
## 26  237.53508 212.37463 262.69553
## 27  143.45083 112.81180 174.08986
## 28  221.04683 184.95011 257.14354
## 29  143.71220 114.51210 172.91230
## 30  262.79920 242.66594 282.93246
## 31  115.79434  85.23153 146.35714
## 32  231.88143 214.77944 248.98342
## 33  174.97169 157.13611 192.80727
## 34  219.50773 198.95500 240.06046
## 35  181.64714 152.29397 211.00032
## 36  170.73879 153.92887 187.54872
## 37  279.60950 258.11133 301.10768
## 38  255.23001 224.24678 286.21324
## 39  238.59140 211.73016 265.45264
## 40   82.70727  47.18729 118.22724
## 41  200.55850 163.00418 238.11282
## 42  137.33961  97.12178 177.55743
## 43  122.29682 101.75646 142.83719
## 44  120.92081  94.22509 147.61652
## 45   93.83956  57.86959 129.80952
## 46  113.20946  79.97635 146.44258
## 47  170.66772 149.01395 192.32149
## 48  237.18055 194.70398 279.65712
## 49  164.11061 136.00500 192.21622
## 50  155.50000 112.62534 198.37466
## 51  161.32528 133.01085 189.63972
## 52  138.68579 105.85408 171.51750
## 53  211.88021 188.72945 235.03097
## 54  201.89697 180.50738 223.28656
## 55   78.34235  41.03928 115.64541
## 56   89.58576  61.85766 117.31385
## 57  506.54447 469.31666 543.77227
## 58  139.71838 111.51721 167.91954
## 59  165.38898 136.59389 194.18406
## 60  174.21000 148.15985 200.26016
## 61  159.40273 133.36888 185.43657
## 62  242.30761 210.07120 274.54402
## 63  137.04318 107.43360 166.65276
## 64  159.93771 122.42612 197.44929
## 65  113.01572  84.40633 141.62510
## 66  115.71523  79.58168 151.84878
## 67  260.91020 240.78844 281.03195
## 68  127.45100  90.93414 163.96786
```

```
## 69    79.00000  36.12534 121.87466
## 70   208.80509 190.21256 227.39762
## 71   359.04044 328.72297 389.35791
## 72   125.10054 101.41702 148.78406
## 73   339.16003 305.74476 372.57530
## 74   162.30618 128.03218 196.58018
## 75    90.94697  57.78198 124.11197
## 76    88.10635  61.39890 114.81379
## 77   216.07663 197.80619 234.34707
## 78   173.58959 147.90190 199.27728
## 79   155.96449 126.49987 185.42912
## 80   128.46393  98.96381 157.96406
## 81    57.02896  19.92681  94.13110
## 82   165.87406 132.47397 199.27414
## 83   207.74932 192.85578 222.64287
## 84   143.82068 114.55738 173.08399
## 85   126.09480  88.24414 163.94545
## 86   146.87714 118.14440 175.60989
## 87   132.11582 106.44094 157.79070
## 88   132.48303 105.81233 159.15373
## 89   307.12254 267.28419 346.96088
## 90   232.66774 197.84586 267.48961
## 91   186.11105 164.09492 208.12718
## 92   143.00000 100.12534 185.87466
## 93   115.95621  90.85780 141.05463
## 94   191.39027 168.84989 213.93065
## 95   150.00000 107.12534 192.87466
## 96   287.29898 264.29597 310.30200
## 97   170.63702 152.32823 188.94580
## 98   173.82363 149.01786 198.62940
## 99    69.03653  32.40560 105.66746
## 100  175.30520 137.45455 213.15586
## 101  140.39733 116.37178 164.42287
## 102   58.69935  22.89808  94.50061
## 103  124.97266  89.13526 160.81006
## 104  135.23935 110.64270 159.83601
## 105  143.50310 116.04000 170.96620
## 106  111.82342  82.71786 140.92897
## 107  176.06885 154.67186 197.46584
## 108  115.89032  84.09813 147.68251
## 109  221.06229 183.55071 258.57388
## 110  118.00000  75.12534 160.87466
## 111  121.00434  96.58282 145.42586
## 112  274.47618 254.59886 294.35350
## 113  280.92580 248.19324 313.65836
## 114  149.47338 126.58167 172.36509
## 115  114.24810  86.11735 142.37884
## 116  210.64391 183.03491 238.25291
## 117  202.01792 180.48975 223.54610
## 118  150.00000 107.12534 192.87466
## 119  182.92783 147.20250 218.65316
## 120  229.50095 210.28857 248.71332
## 121  145.91211 112.64748 179.17675
## 122  112.24649  89.71910 134.77388
## 123   86.27907  47.78298 124.77517
```

```
## 124 204.02500 182.34187 225.70813
## 125 124.36529  94.29659 154.43400
## 126 122.97922  96.93491 149.02353
## 127 302.33863 285.31049 319.36676
## 128 264.02608 229.10699 298.94517
## 129 142.87081 114.80173 170.93989
## 130 210.43468 192.31363 228.55574
## 131 176.13338 152.14195 200.12481
## 132 183.02615 148.74727 217.30503
## 133 121.78383 100.01661 143.55105
## 134 133.03600 107.84304 158.22896
## 135 108.70928  81.82543 135.59313
## 136 141.16601 122.47950 159.85252
## 137 144.47702 112.92526 176.02879
## 138 147.95485 123.12325 172.78646
## 139 198.07295 173.54576 222.60014
## 140 132.71210 105.63879 159.78540
## 141 148.78594 118.24532 179.32655
## 142 107.66039  67.44257 147.87822
## 143 118.89412  91.03051 146.75774
## 144 161.65908 137.06946 186.24869
## 145 267.39340 236.58881 298.19799
## 146 304.16678 281.84029 326.49327
## 147 120.35575  88.05964 152.65185
## 148 193.69526 174.61211 212.77842
## 149 232.46084 208.26068 256.66099
## 150 162.93672 141.37113 184.50230
## 151 106.51556  76.76622 136.26490
## 152 205.67280 165.29701 246.04859
## 153 265.59900 232.50168 298.69632
## 154 187.28477 151.15122 223.41832
## 155 256.07065 239.47614 272.66517
## 156 103.98360  79.36134 128.60586
## 157 153.08467 119.80161 186.36773
## 158 115.18937  81.93441 148.44432
## 159 141.73564 111.94988 171.52139
## 160 126.95715  98.60502 155.30928
## 161  87.23441  49.94263 124.52618
## 162 129.37746  89.53912 169.21581
## 163 200.14524 178.51683 221.77365
## 164 114.47924  79.49701 149.46146
## 165 202.39671 185.51189 219.28153
## 166 121.03298  94.36598 147.69998
## 167 102.35584  68.15629 136.55539
## 168 153.32743 126.91272 179.74213
## 169 138.29502 105.34546 171.24459
## 170  81.00000  38.12534 123.87466
## 171 327.10850 294.22044 359.99657
## 172 102.52387  72.20970 132.83804
## 173 111.47386  77.49648 145.45124
## 174 122.58171 100.37452 144.78889
## 175 320.98777 295.41440 346.56114
## 176 113.24099  90.37787 136.10412
## 177 122.86582  95.27722 150.45442
## 178 194.78582 170.76077 218.81087
```

```
## 179 276.15107 244.83255 307.46958
## 180 191.17408 174.86039 207.48777
## 181 181.13843 163.37857 198.89829
## 182 252.18771 222.21042 282.16501
## 183 205.35158 190.52677 220.17640
## 184 170.73296 142.96992 198.49599
## 185 241.66030 214.43150 268.88909
## 186 255.65746 229.24799 282.06694
## 187 354.01984 328.21049 379.82918
## 188 182.03595 156.60388 207.46802
## 189 150.36668 131.44000 169.29337
## 190  82.10568  48.92358 115.28778
## 191 221.72897 202.48049 240.97744
## 192 173.18291 151.65725 194.70858
## 193 153.72387 122.74630 184.70144
## 194 176.08603 148.19146 203.98060
## 195 226.09690 201.77304 250.42077
## 196 146.09421 116.27690 175.91153
## 197 132.49186 107.27921 157.70451
## 198 305.05227 280.20864 329.89591
## 199 221.77385 199.70072 243.84697
## 200 220.52526 194.87758 246.17294
```