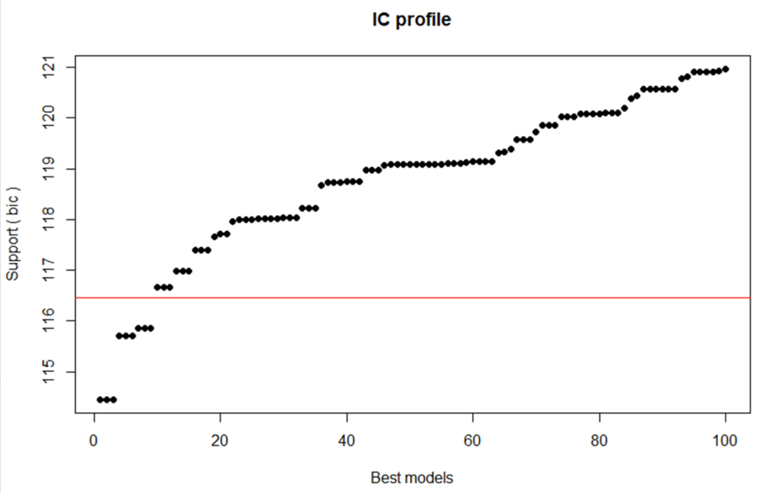
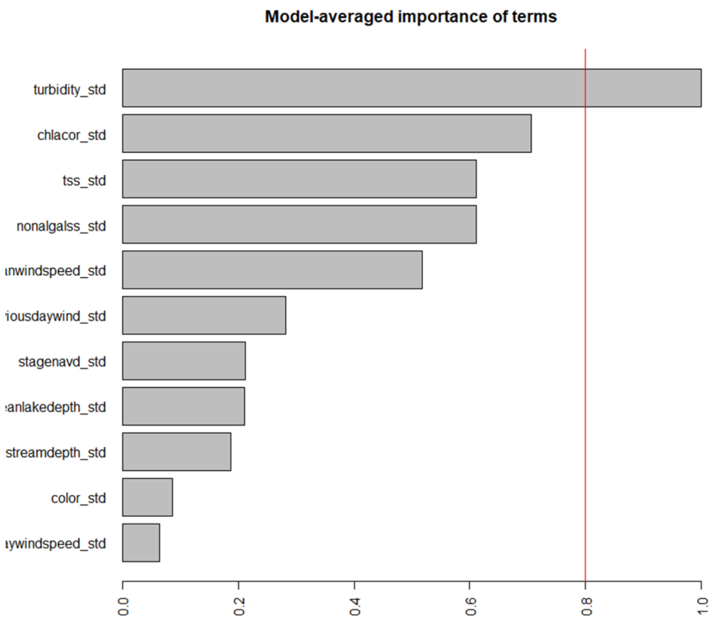
Advanced R: Statistical Machine Learning

Dan Schmutz, MS
Chief Environmental Scientist

Zoom Workshop for SJRWMD
September 24, 2020





Linear Regression

Loading libraries

```
# working on linear models with ames

# Helper packages
library(tidyverse)
library(dplyr)    # for data manipulation
library(ggplot2)  # for awesome graphics

# Modeling packages
library(caret)    # for crossvalidation, etc.

# Model interpretability packages
library(vip)      # variable importance
library(visreg) # visualizing partial residual plots
```

GPI

Simple (i.e., one predictor) linear regression

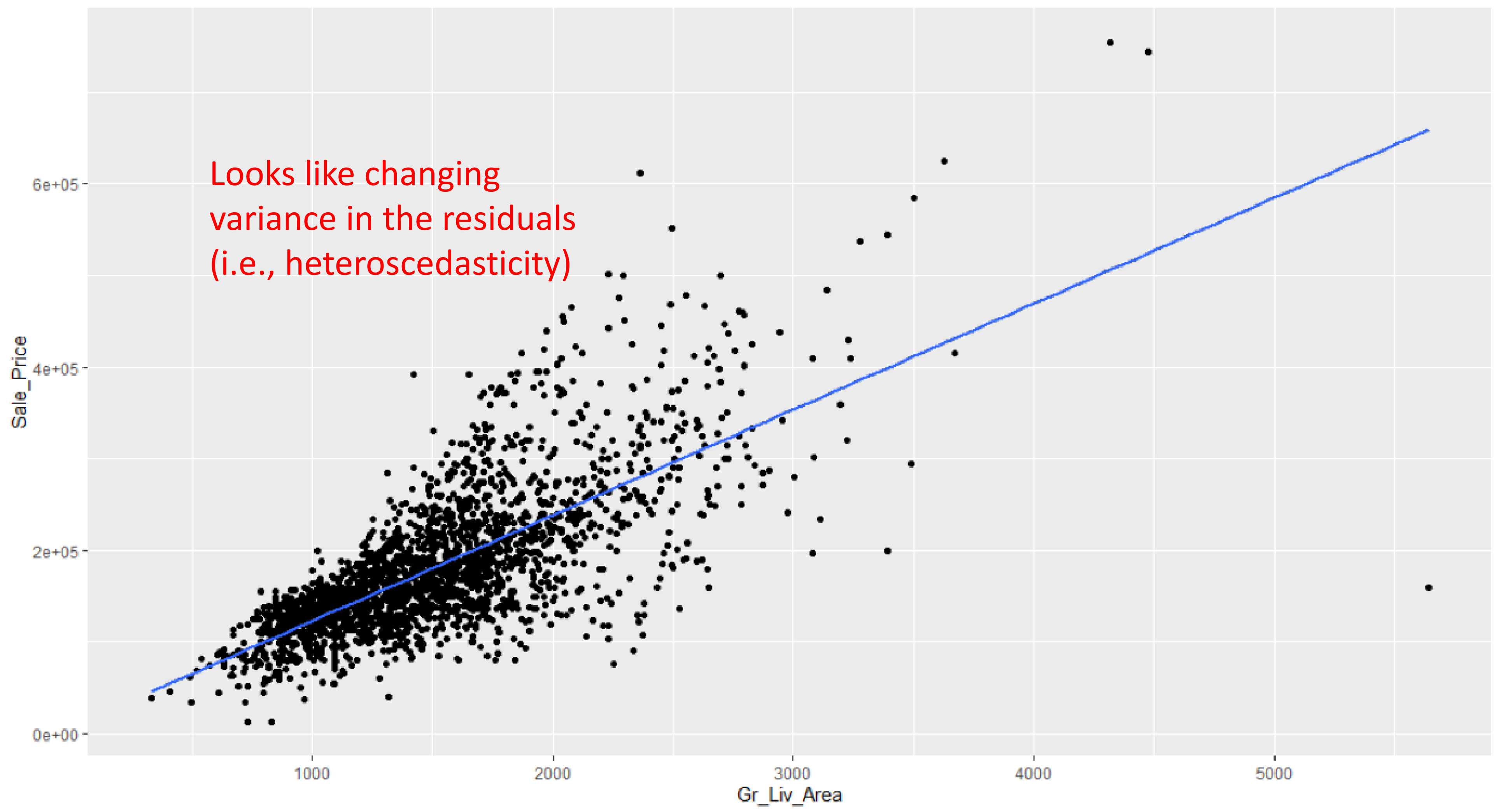
```
# basic linear regression
lm1 <- lm(Sale_Price ~ Gr_Liv_Area, data = train_3)
summary(lm1)

ggplot(train_3, aes(x= Gr_Liv_Area, y=Sale_Price))+
  geom_point()+stat_smooth(method="lm", se=F)
```

```
> summary(lm1)

Call:
lm(formula = Sale_Price ~ Gr_Liv_Area, data = train_3)

Residuals:
    Min      1Q  Median      3Q     Max
-498916  -30162   -1797   23268  331104

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7689.038   3515.735   2.187   0.0288 *
Gr_Liv_Area  115.425      2.229  51.786   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 53990 on 2344 degrees of freedom
Multiple R-squared:  0.5336,	Adjusted R-squared:  0.5334
F-statistic:  2682 on 1 and 2344 DF,  p-value: < 2.2e-16
```

$53,990 is a pretty big average error on median $160,000 home

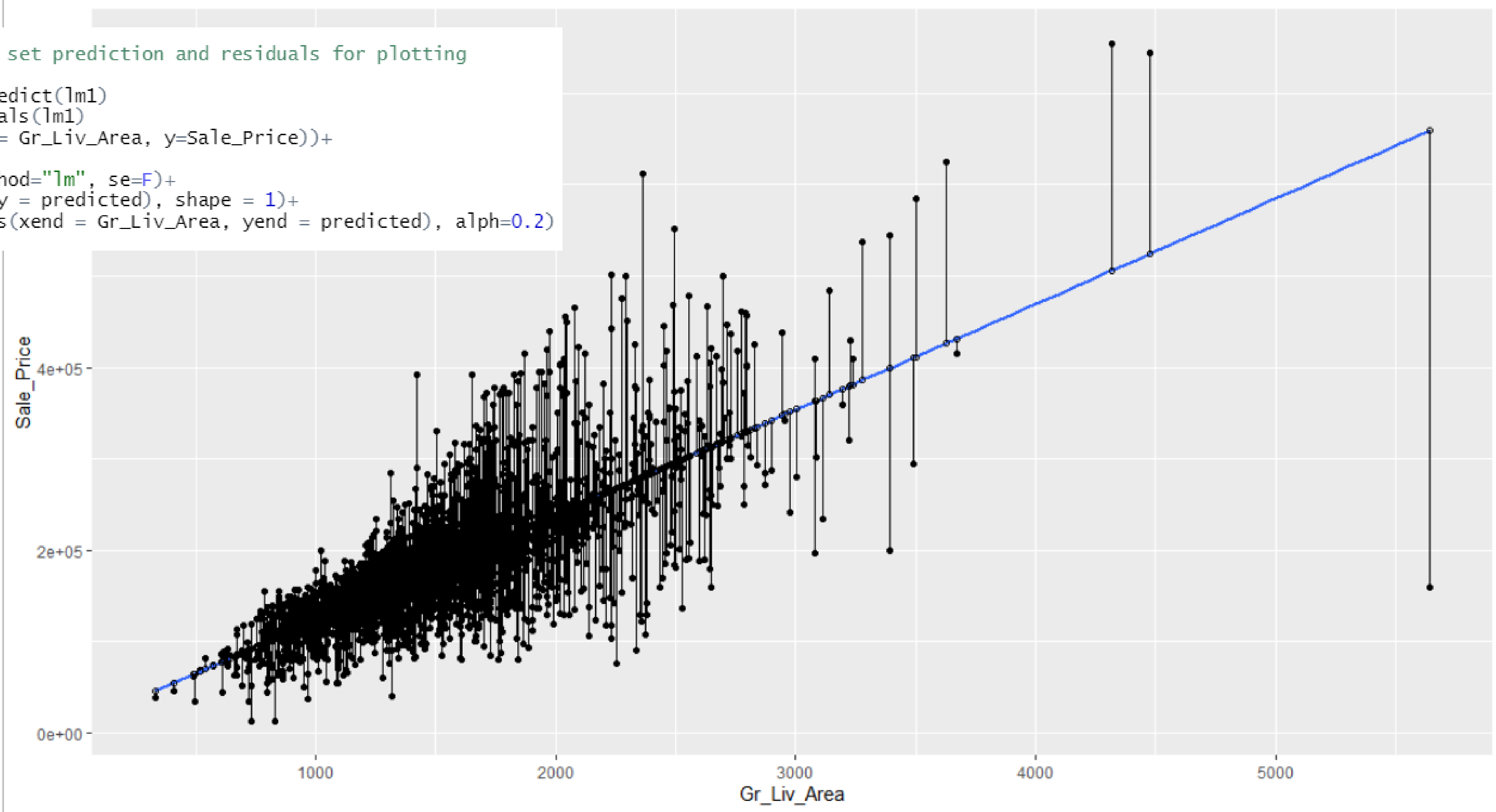
R^2 = 53%, so around half variation in dependent explained by the predictor

Sale Price predicted by Aboveground Living Area



Showing the residuals

```r
# saving training set prediction and residuals for plotting
t3a<-train_3
t3a$predicted<-predict(lm1)
t3a$resid<-residuals(lm1)
ggplot(t3a, aes(x= Gr_Liv_Area, y=Sale_Price))+
  geom_point()+
  stat_smooth(method="lm", se=F)+
  geom_point(aes(y = predicted), shape = 1)+
  geom_segment(aes(xend = Gr_Liv_Area, yend = predicted), alph=0.2)
```



GPI

RMSE and confidence intervals for the coefficients

```
sigma(lm1)    # RMSE same as residual standard error in output (with rounding)
## [1] 53992.84
sigma(lm1)^2  # MSE
## [1] 2915226792

confint(lm1, level = 0.95)
##                2.5 %     97.5 %
##(Intercept)  794.7637 14583.3121
## Gr_Liv_Area 111.0540   119.7955
```

Multiple Linear Regression (MLR)

```
# mutliple linear regression, two variables
lm2 <- lm(Sale_Price ~ Gr_Liv_Area + Year_Built, data = train_3)
summary(lm2)
coef(lm2)
sigma(lm2)

visreg2d(lm2, "Gr_Liv_Area", "Year_Built")
```

GPI

Multiple Linear Regression summary

```
> summary(lm2)

Call:
lm(formula = Sale_Price ~ Gr_Liv_Area + Year_Built, data = train_

Residuals:
    Min      1Q  Median      3Q     Max 
-472607  -26128   -1924   18241  304508 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -2.059e+06  6.072e+04  -33.90   <2e-16 ***
Gr_Liv_Area  9.961e+01  1.881e+00   52.95   <2e-16 ***
Year_Built   1.060e+03  3.112e+01   34.07   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 44160 on 2343 degrees of freedom
Multiple R-squared:  0.6881,	Adjusted R-squared:  0.6878 
F-statistic:  2585 on 2 and 2343 DF,  p-value: < 2.2e-16

> coef(lm2)
  (Intercept)   Gr_Liv_Area    Year_Built 
-2.058515e+06  9.960896e+01  1.060323e+03 
> sigma(lm2)
[1] 44162.19
> 
```

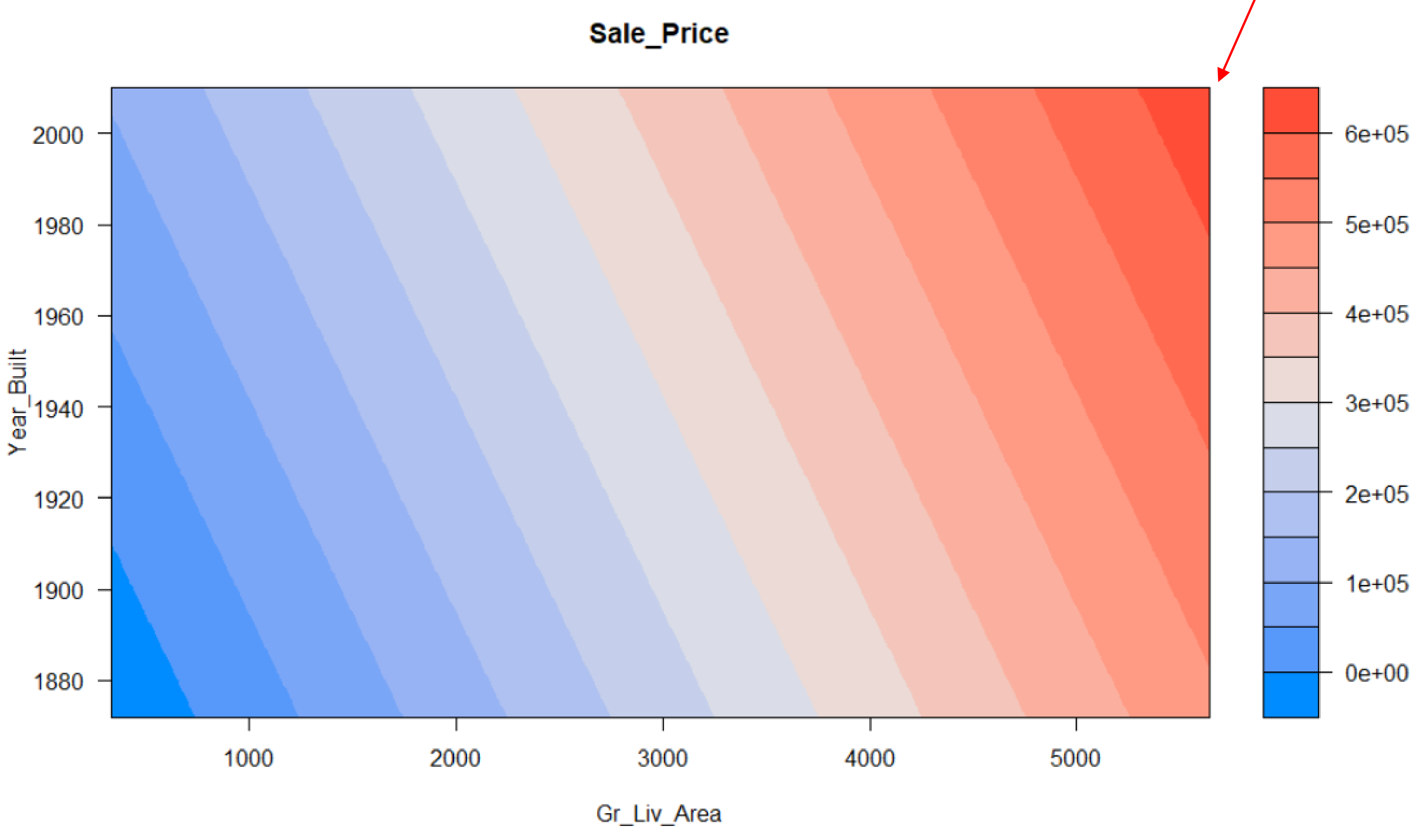
Some improvement in RMSE ($44,160)

R^2 = 69%, improved here too.

Visualizing partial residuals (effects while holding other factors constant at their medians)

Big and new houses most expensive

visreg2d(lm2, "Gr_Liv_Area", "Year_Built")



MLR modeling an interaction term

```
# mutliple linear regression, allowing interaction
lm2b <- lm(Sale_Price ~ Gr_Liv_Area + Year_Built + Gr_Liv_Area:Year_Built, data = train_3)
summary(lm2b)
coef(lm2b)
sigma(lm2b)
visreg2d(lm2b, "Gr_Liv_Area", "Year_Built")
```

GPI

MFL with interaction summary

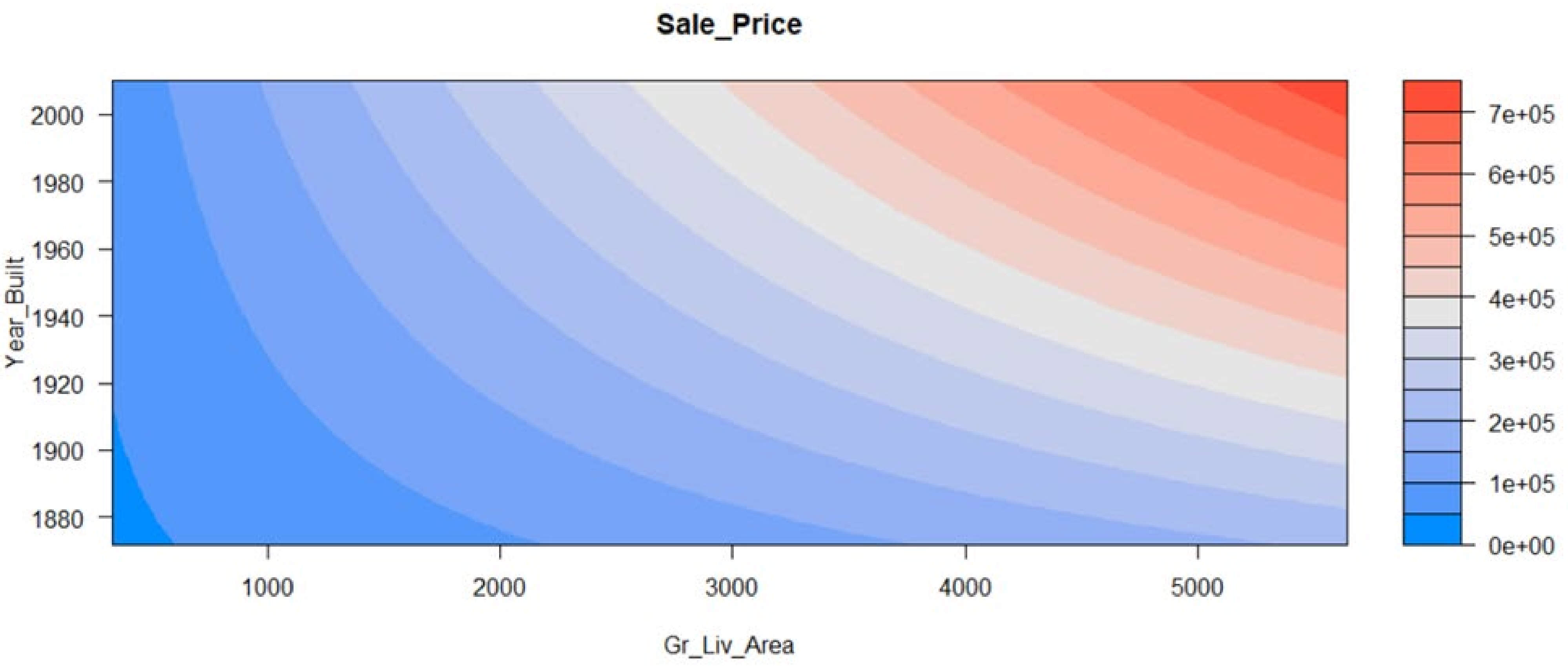
```
> summary(lm2b)

Call:
lm(formula = Sale_Price ~ Gr_Liv_Area + Year_Built + Gr_Liv_Area:Year_Built,
    data = train_3)

Residuals:
    Min      1Q  Median      3Q     Max
-574038  -23928   -1328   17867  286357

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)             7.525e+04  1.875e+05   0.401    0.688
Gr_Liv_Area            -1.257e+03  1.131e+02 -11.106   <2e-16 ***
Year_Built             -2.365e+01  9.534e+01  -0.248    0.804
Gr_Liv_Area:Year_Built  6.881e-01  5.740e-02  11.988   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 42880 on 2342 degrees of freedom
Multiple R-squared:  0.7061,	Adjusted R-squared:  0.7058
F-statistic:  1876 on 3 and 2342 DF,  p-value: < 2.2e-16

> coef(lm2b)
           (Intercept)            Gr_Liv_Area             Year_Built Gr_Liv_Area:Year_Built
         75251.8086542          -1256.5258459            -23.6461432              0.6881007
> sigma(lm2b)
[1] 42875.76
```

Slight improvement in RMSE ($44,160)

R^2 = 72%, slightly improved here too.

Visualizing partial residuals (effects while holding other factors constant at their medians)

```
visreg2d(lm2b, "Gr_Liv_Area", "Year_Built")
```



GPI

Let's use all 80 predictors with convenient dot (.) notation

```
lm3 <- lm(Sale_Price ~ ., data = train_3)
sigma(lm3)
summary(lm3)
```

```
> sigma(lm3)
[1] 18914.88
> summary(lm3)

Call:
lm(formula = Sale_Price ~ ., data = train_3)

Residuals:
    Min      1Q  Median      3Q     Max 
-142067   -8722       0    8724  144081 

Coefficients: (9 not defined because of singularities)
                                              Estimate Std. Error t value Pr(>|t|)
(Intercept)                                 -1.369e+07  9.368e+06  -1.462 0.144030
MS_SubClassOne_Story_1945_and_Older          5.099e+03  3.007e+03   1.696 0.090068
MS_SubClassOne_Story_with_Finished_Attic_All_Ages   1.127e+04  9.171e+03   1.229 0.219251
MS_SubClassOne_and_Half_Story_Unfinished_All_Ages   1.616e+04  1.190e+04   1.358 0.174514
MS_SubClassOne_and_Half_Story_Finished_All_Ages     6.494e+03  5.355e+03   1.213 0.225340
MS_SubClassTwo_Story_1946_and_Newer         -4.324e+03  4.807e+03  -0.900 0.368443
MS_SubClassTwo_Story_1945_and_Older          6.601e+03  5.303e+03   1.245 0.213403
MS_SubClassTwo_and_Half_Story_All_Ages      -1.524e+04  9.349e+03  -1.630 0.103156
```

It goes on and on…

Let's use all 80 predictors with convenient dot (.) notation

```
lm3 <- lm(Sale_Price ~ ., data = train_3)
sigma(lm3)
summary(lm3)
```

```
> sigma(lm3)
[1] 18914.88
> summary(lm3)

Call:
lm(formula = Sale_Price ~ ., data = train_3)

Residuals:
    Min      1Q  Median      3Q     Max 
-142067   -8722       0    8724  144081 

Coefficients: (9 not defined because of singularities)
                                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                                     -1.369e+07  9.368e+06  -1.462 0.144030
MS_SubClassOne_Story_1945_and_Older              5.099e+03  3.007e+03   1.696 0.090068
MS_SubClassOne_Story_with_Finished_Attic_All_Ages  1.127e+04  9.171e+03   1.229 0.219251
MS_SubClassOne_and_Half_Story_Unfinished_All_Ages  1.616e+04  1.190e+04   1.358 0.174514
MS_SubClassOne_and_Half_Story_Finished_All_Ages  6.494e+03  5.355e+03   1.213 0.225340
MS_SubClassTwo_Story_1946_and_Newer             -4.324e+03  4.807e+03  -0.900 0.368443
MS_SubClassTwo_Story_1945_and_Older              6.601e+03  5.303e+03   1.245 0.213403
MS_SubClassTwo_and_Half_Story_All_Ages          -1.524e+04  9.349e+03  -1.630 0.103156
```

It goes on and on…

80 predictor MLR

```
Residual standard error: 18910 on 2052 degrees of freedom
Multiple R-squared:  0.9499,	Adjusted R-squared:  0.9427
F-statistic: 132.8 on 293 and 2052 DF,  p-value: < 2.2e-16
```

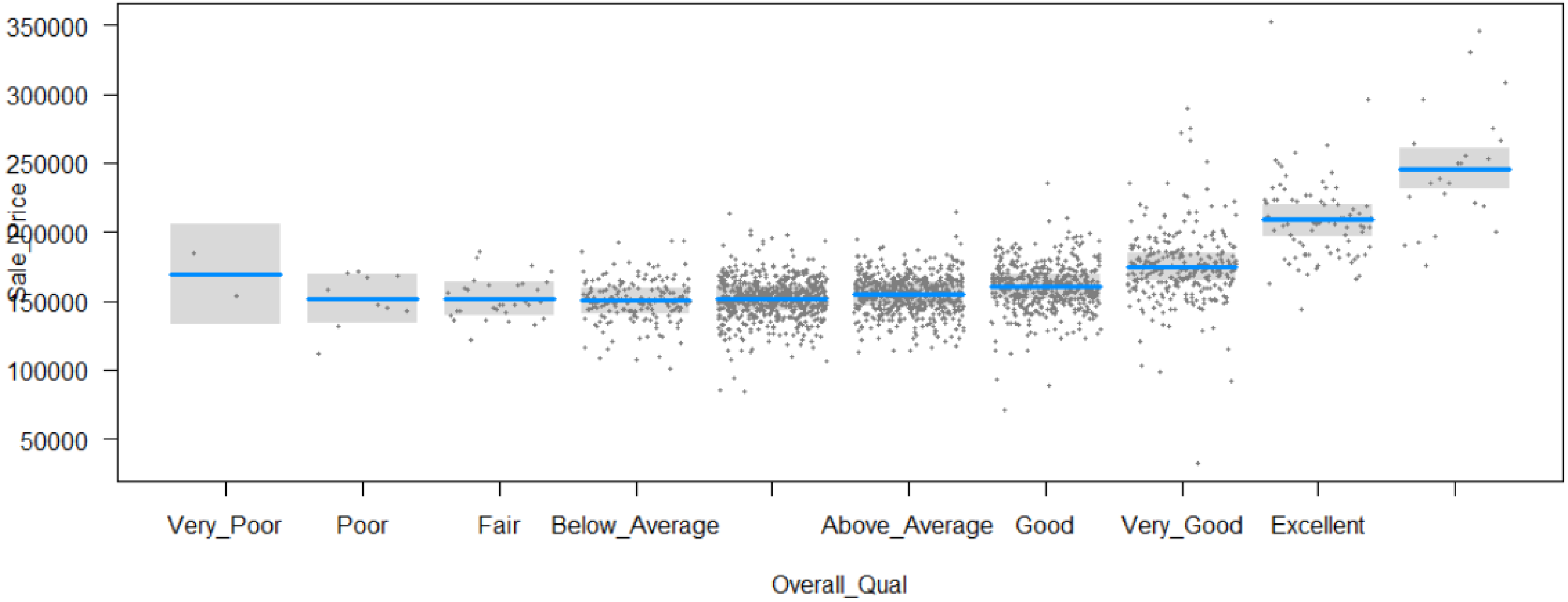
Substantial improvement in RMSE ($18,910)

R^2 = 94%, highly improved.

With so many variables, would the model do well on future datasets? Is it overfit?

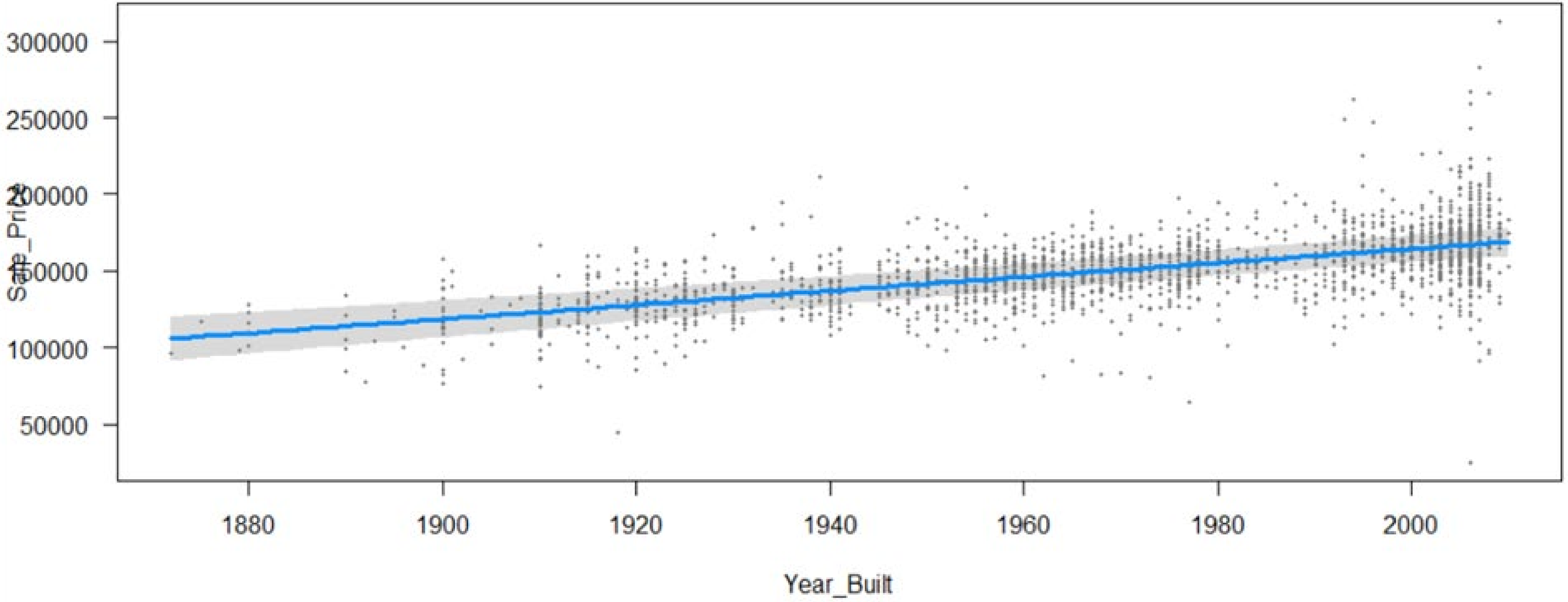
Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



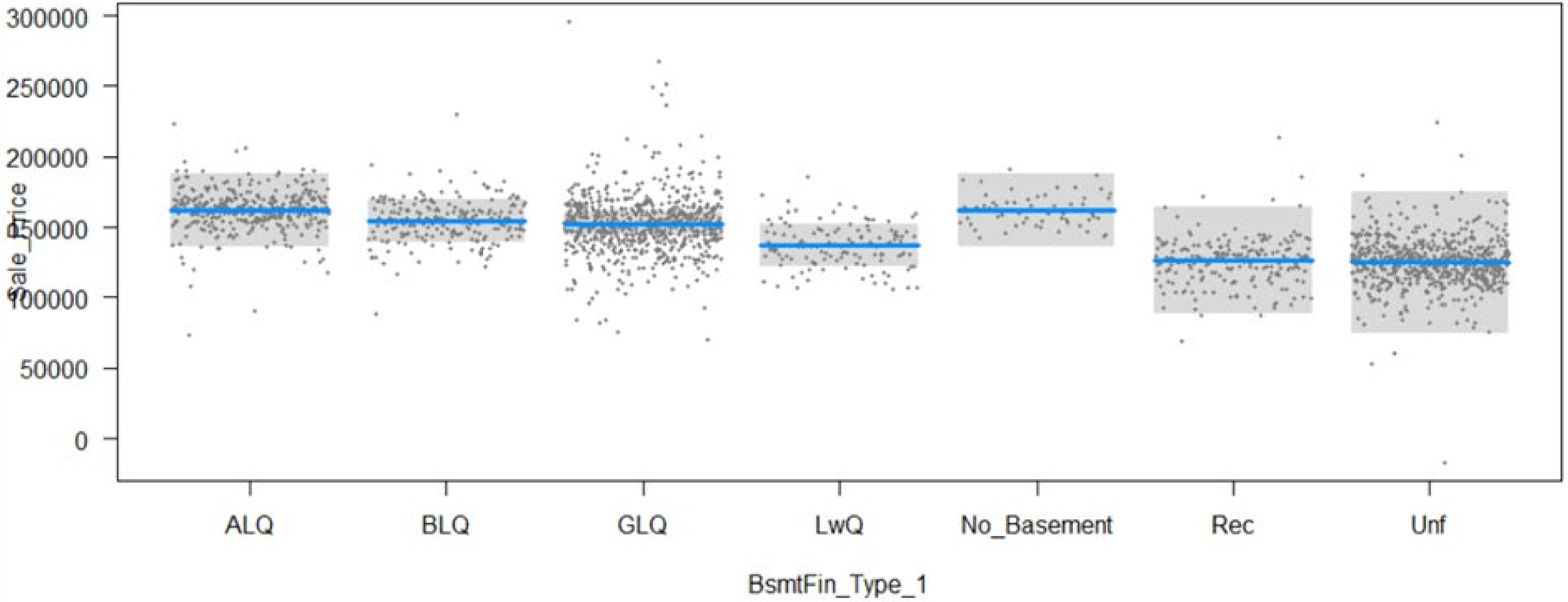
Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



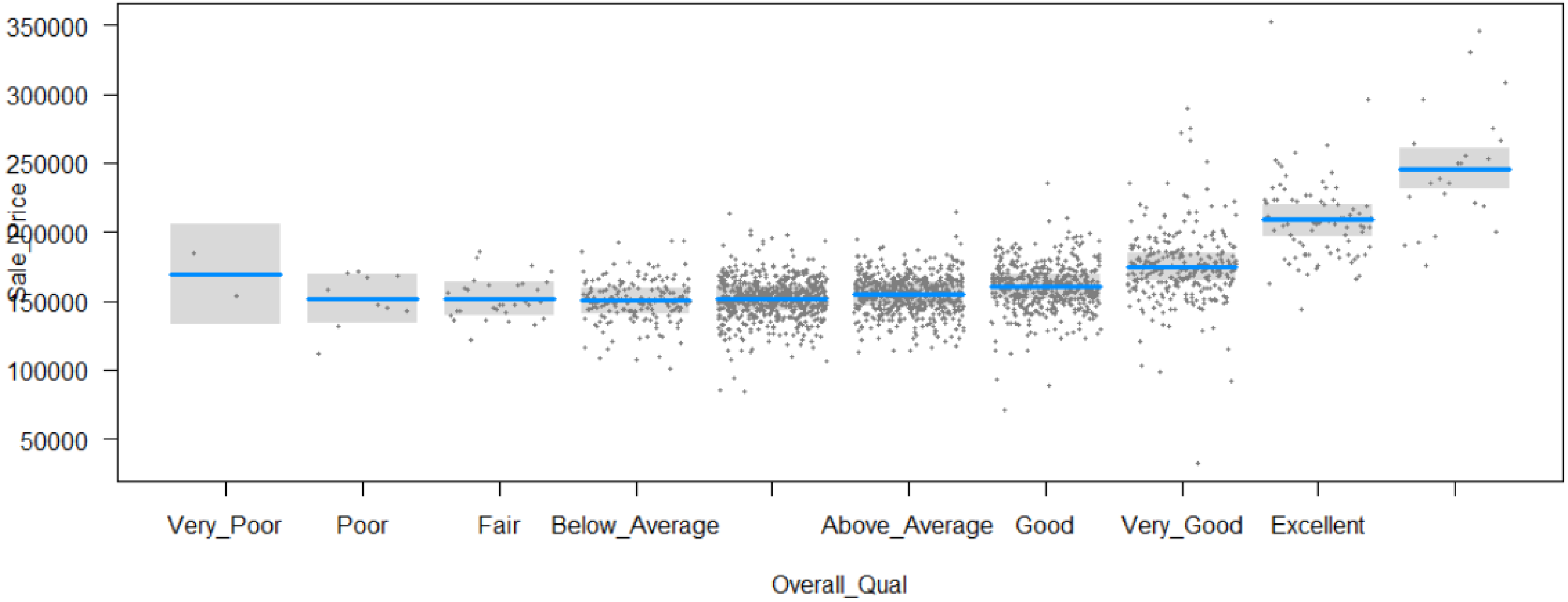
Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



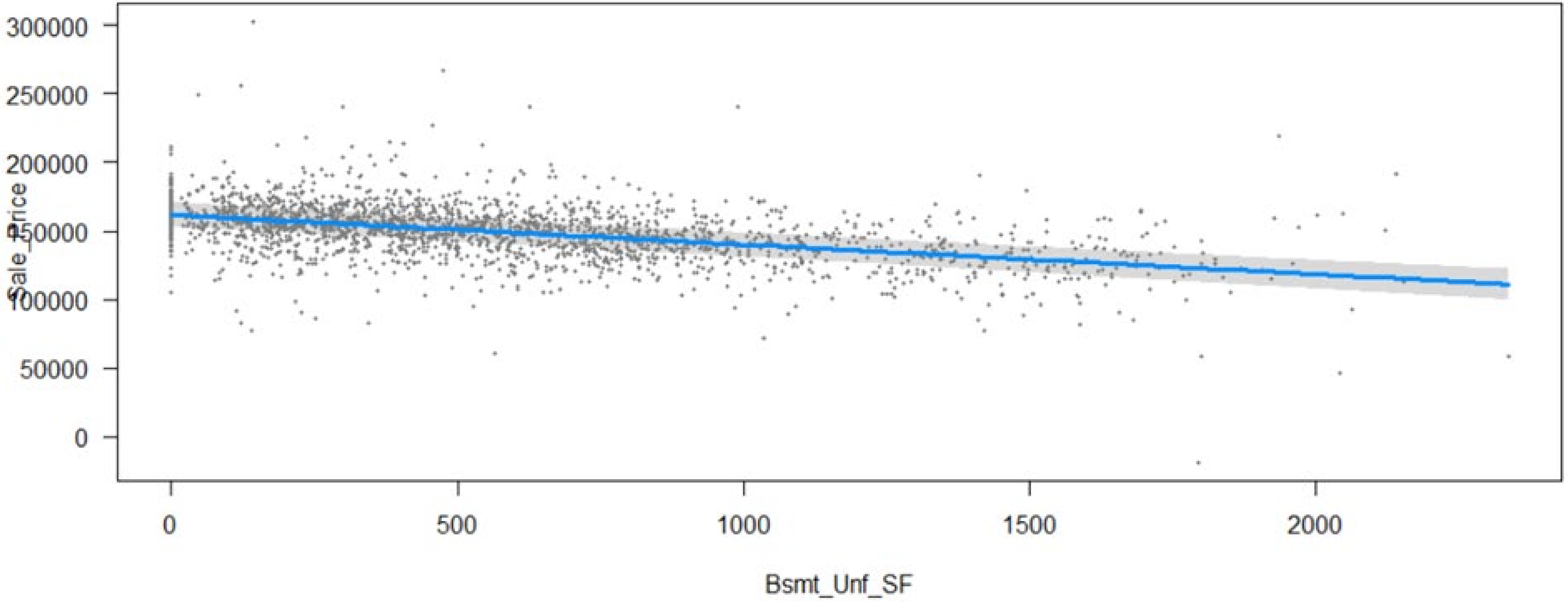
Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



Partial residuals for 80-variable MLR

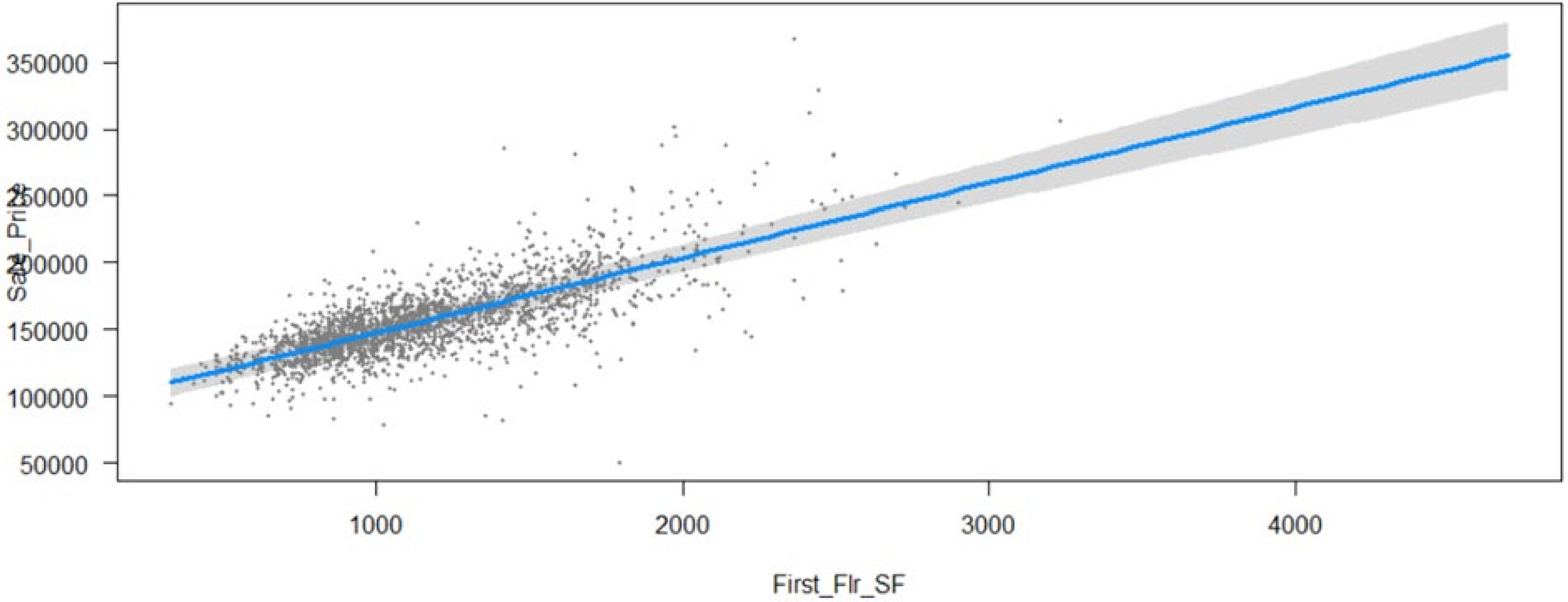
visreg(lm3) # let's visualize a few of the variable partial residuals



GPI

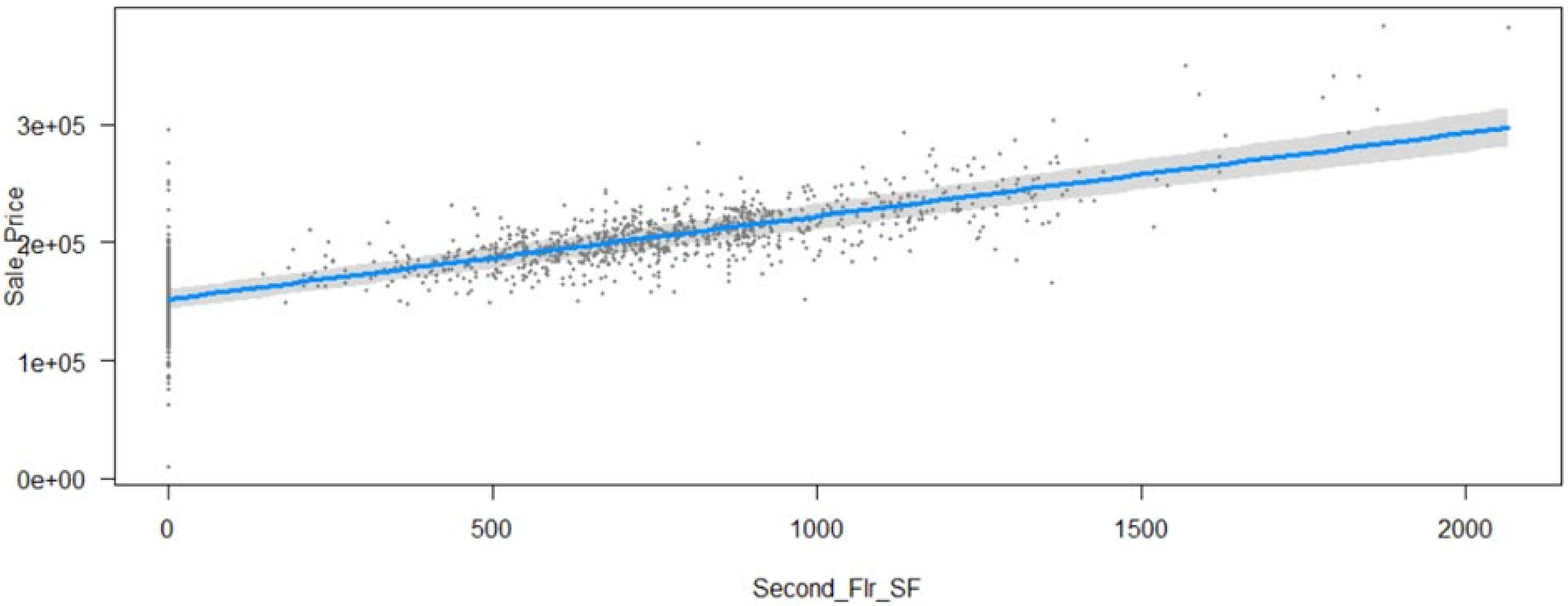
Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



Partial residuals for 80-variable MLR

visreg(lm3) # let's visualize a few of the variable partial residuals



Comparing models using 10-fold cv in caret

- Must use same random seed to allow comparison of exact same folds

```r
# using caret's cv to compare models using 10-fold cv
# Train model using 10-fold cross-validation
# model 1 CV
set.seed(42)  # for reproducibility
cv_model1 <- train(
  form = Sale_Price ~ Gr_Liv_Area,
  data = train_3,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)
# model 2 CV
set.seed(42)
cv_model2 <- train(
  Sale_Price ~ Gr_Liv_Area + Year_Built,
  data =train_3,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)

# model 3 CV
set.seed(42)
cv_model3 <- train(
  Sale_Price ~ .,
  data = train_3,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)

# Extract out of sample performance measures
sum123<-summary(resamples(list(
  model1 = cv_model1,
  model2 = cv_model2,
  model3 = cv_model3
)))
sum123
```

Yes, r uses a function to obtain “random” numbers, from the Mersenne-Twister algorithm.

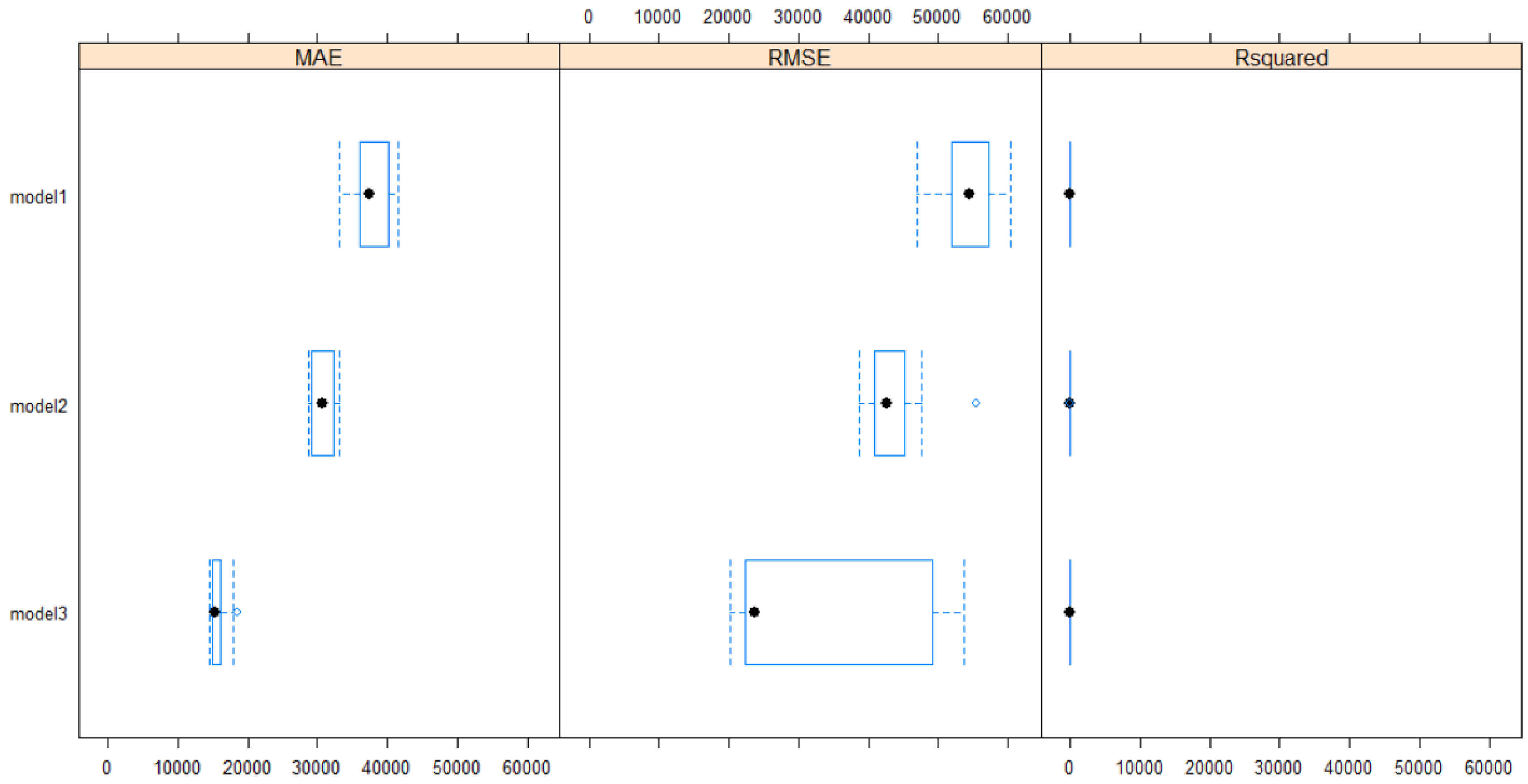
GPI

Comparing models using 10-fold cv in caret

```
> sum123

Call:
summary.resamples(object = resamples(list(model1 = cv_model1, model2 = cv_model2, model3
 = cv_model3)))

Models: model1, model2, model3
Number of resamples: 10

MAE
           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
model1 33102.82 36340.54 37418.57 37646.29 39825.20 41498.91    0
model2 28807.12 29265.42 30784.17 30743.23 32113.27 33085.22    0
model3 14506.81 14907.17 15355.27 15877.03 16203.89 18539.76    0

RMSE
           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
model1 46985.33 51934.20 54376.35 53872.26 56970.20 60433.74    0
model2 38694.23 41190.29 42691.08 43946.59 45195.99 55536.29    0
model3 20152.65 22407.26 23697.61 31374.85 43190.61 53722.75    0

Rsquared
            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
model1 0.4472120 0.5012196 0.5404753 0.5369577 0.5587922 0.6471227    0
model2 0.6196496 0.6816929 0.6979881 0.6918525 0.6997295 0.7586587    0
model3 0.6274540 0.7612039 0.9037655 0.8439449 0.9256921 0.9374916    0
```

Comparing models using 10-fold cv in caret

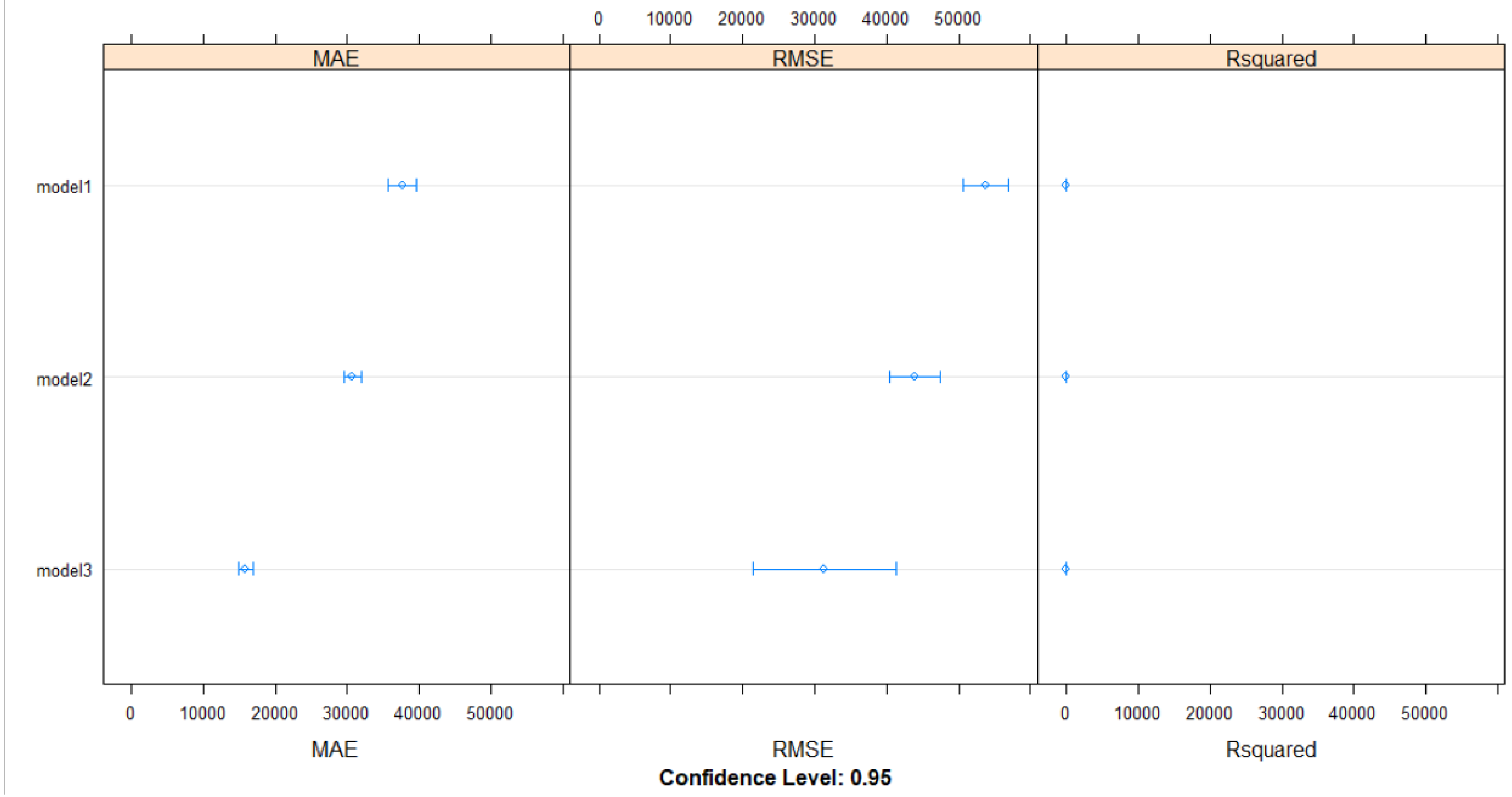
bwplot(results123)
dotplot(results123)



Comparing models using 10-fold cv in caret

```
bwplot(results123)
dotplot(results123)
```

Outliers might account for difference between MAE and RMSE



Can use broom:augment to add regression results on to the training dataframe

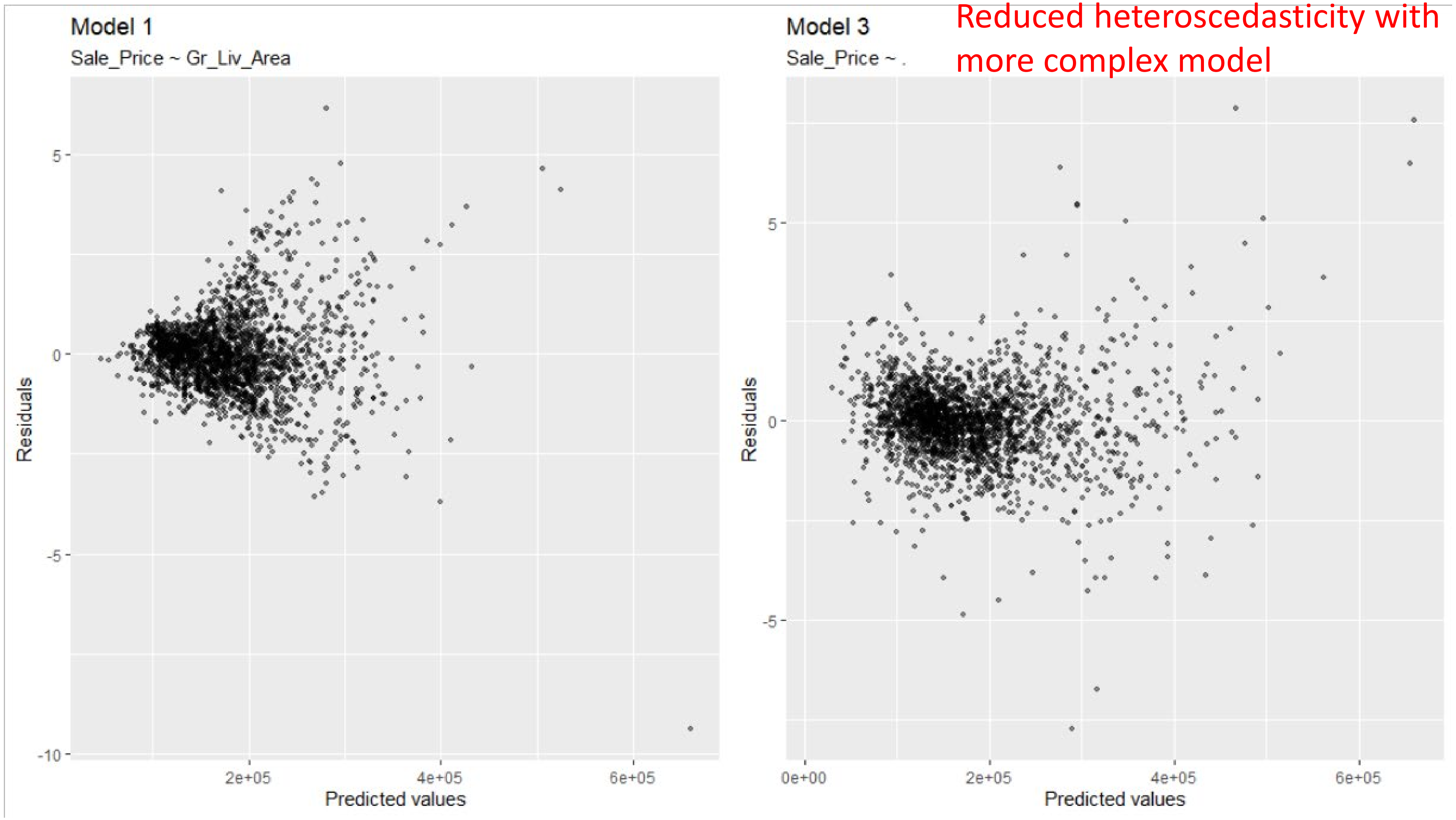
```
dflm1 <- broom::augment(cv_model1$finalModel, data = train_3)
str(dflm1)
glimpse(dflm1)

dflm3 <- broom::augment(cv_model3$finalModel, data = train_3)
```

```
$ Longitude           <dbl> -93.61975, -93.61976, -93.61939, -93.61...
$ Latitude            <dbl> 42.05403, 42.05301, 42.05266, 42.05125,...
$ .fitted             <dbl> 198832.5, 111109.6, 161088.6, 251235.3,...
$ .std.resid          <dbl> 0.29950893, -0.11321526, 0.20213836, -0...
$ .hat                <dbl> 0.0004698942, 0.0010396878, 0.000473770...
$ .sigma              <dbl> 54003.33, 54004.21, 54003.89, 54004.15,...
$ .cooksd             <dbl> 2.108598e-05, 6.670135e-06, 9.683702e-0...
$ id                  <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, ...
```

GPI

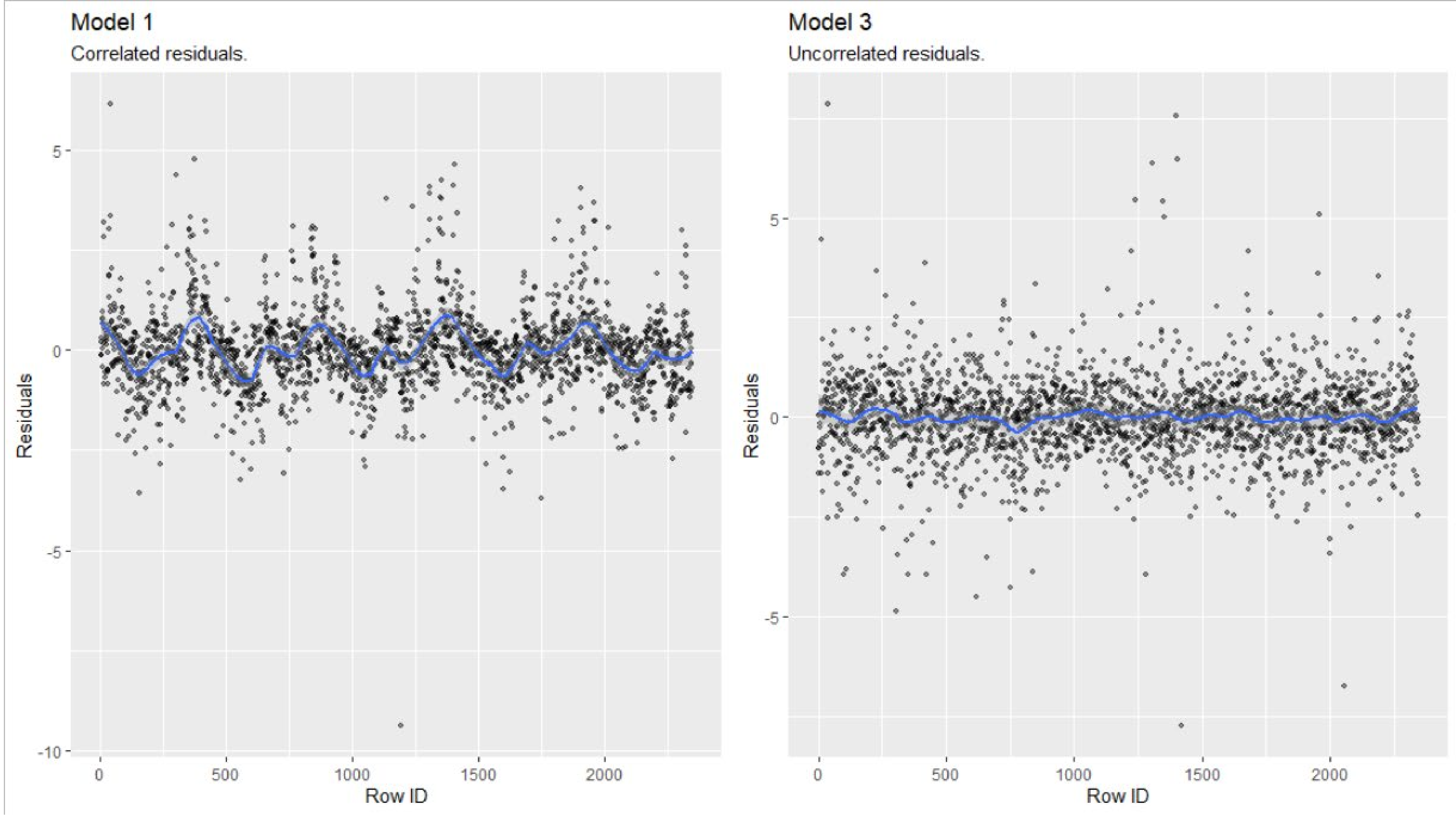
Residual plots

Reduced heteroscedasticity with more complex model

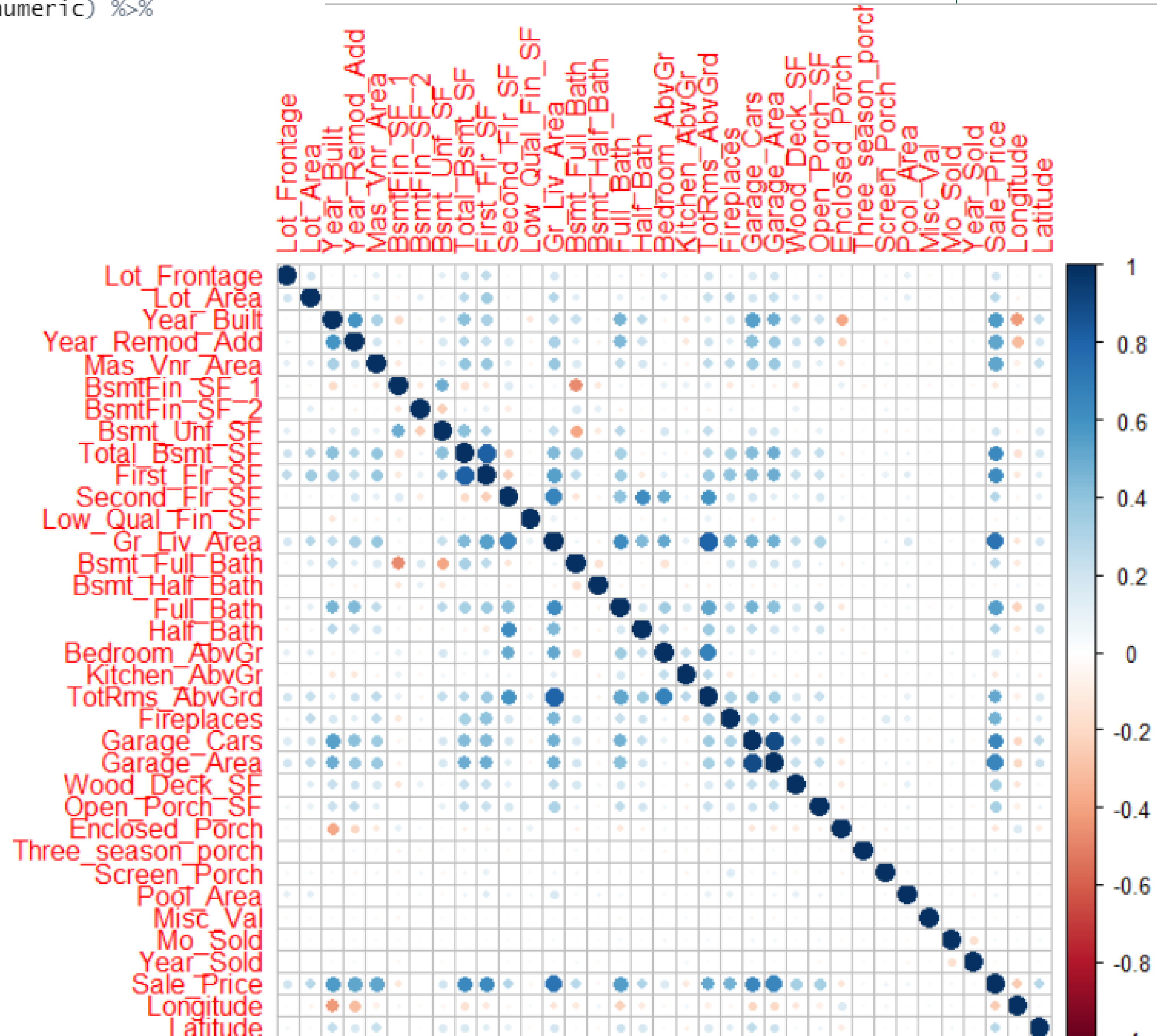


Residual plots

The simple model showed spatial autocorrelation due to houses in same neighborhood occurring in nearby row numbers



```r
library(corrplot)
# look at correlations among the 35 numeric variables
train_3 %>% dplyr::select_if(is.numeric) %>%
  cor() %>% corrplot()
```



Evaluating lm1 and lm2 on test data

```
# how do we do on prediction with the 3 models, evaluated on the test data?
pred_lm1<-predict(lm1,newdata=test_3)
pred_lm2<-predict(lm2,newdata=test_3)
pred_lm3<-predict(lm3,newdata=test_3) # problem with new level in roof mater

test_3aug<-test_3
test_3aug$pred_lm1<-pred_lm1
test_3aug$pred_lm2<-pred_lm2

ggplot(test_3aug,aes(x=pred_lm1,y=Sale_Price))+
  geom_point()+stat_smooth(method=lm)+
  geom_abline(slope=1, intercept=0, col='red')
ggplot(test_3aug,aes(x=pred_lm2,y=Sale_Price))+
  geom_point()+stat_smooth(method=lm)+
  geom_abline(slope=1, intercept=0, col='red')

cor(test_3aug$pred_lm1,test_3aug$Sale_Price)^2
cor(test_3aug$pred_lm2,test_3aug$Sale_Price)^2

res_lm1<-test_3aug$Sale_Price-test_3aug$pred_lm1
res_lm2<-test_3aug$Sale_Price-test_3aug$pred_lm2

(mean((res_lm1)^2))^0.5
(mean((res_lm2)^2))^0.5
```

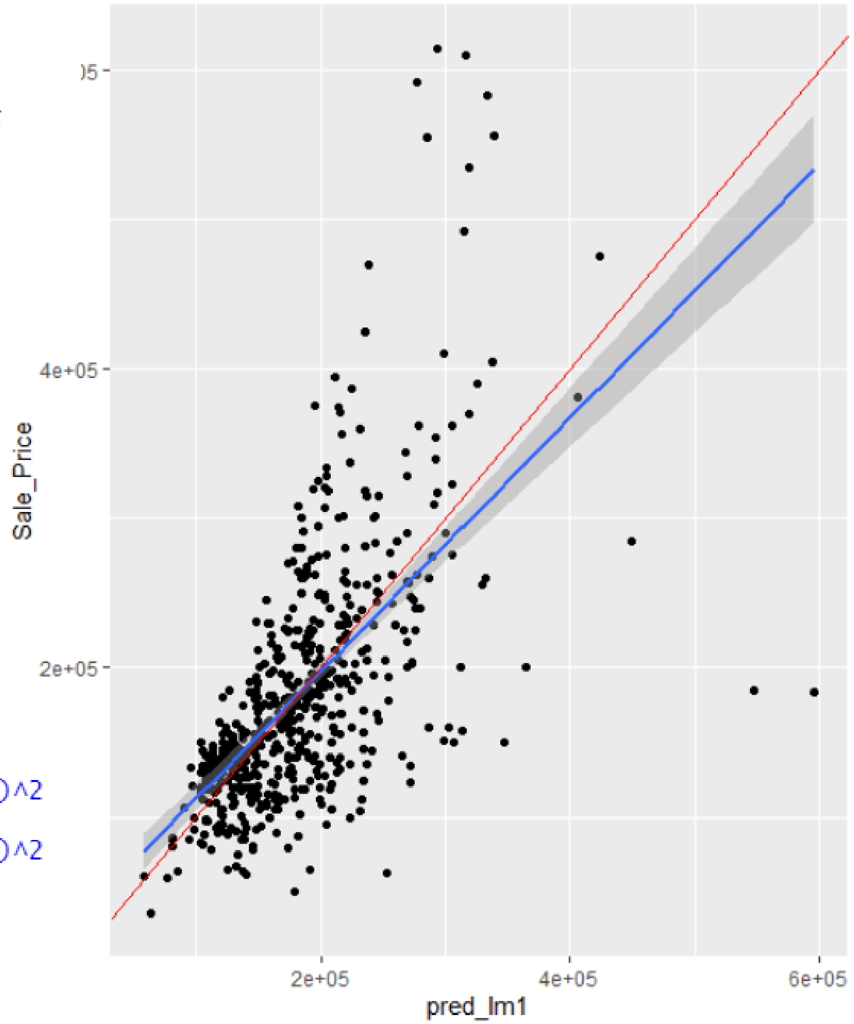
GPI

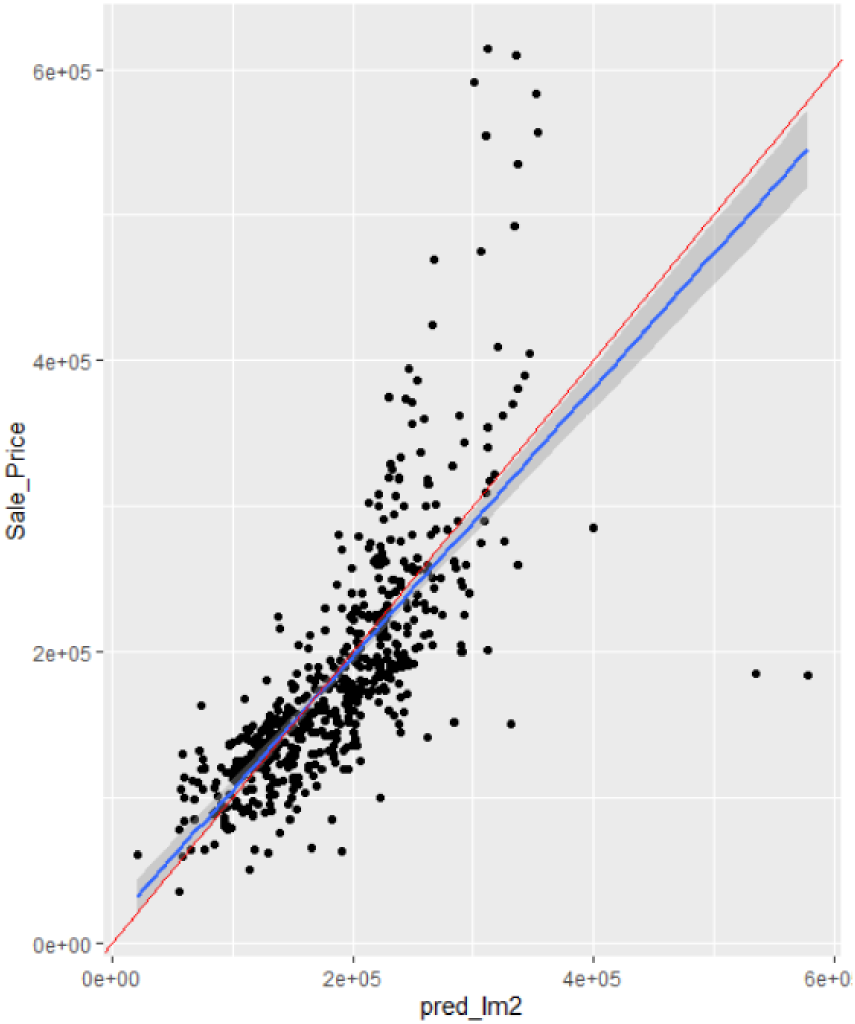
lm2 appears better on observed vs expected plot

r^2 = 39%, RMSE = 65845

r^2 = 56%, RMSE = 55721

```
cor(test_3aug$pred_lm1,test_3aug$Sale_Price)^2
cor(test_3aug$pred_lm2,test_3aug$Sale_Price)^2

res_lm1<-test_3aug$Sale_Price-test_3aug$pred_lm1
res_lm2<-test_3aug$Sale_Price-test_3aug$pred_lm2

(mean((res_lm1)^2))^0.5
(mean((res_lm2)^2))^0.5
```






```
> cor(test_3aug$pred_lm1,test_3aug$Sale_Price)^2
[1] 0.3850424
> cor(test_3aug$pred_lm2,test_3aug$Sale_Price)^2
[1] 0.5554152
> (mean((res_lm1)^2))^0.5
[1] 65845.83
> (mean((res_lm2)^2))^0.5
[1] 55721.26
```

Introducing glmulti, a wrapper for all subsets regression

- glmulti supports search for best subset of variables, where best is defined as the most likely model given the data with a penalty for complexity (e.g., Bayesian Information Criterion, BIC).
- An exhaustive search can be made for smaller datasets.
- Up to 30 variables (without interactions) can be investigated using genetic algorithm method.

GPI

Creating smaller dataset for glmulti

- Created a numeric only dataset and then dropped variables with low correlation with Sales_Price

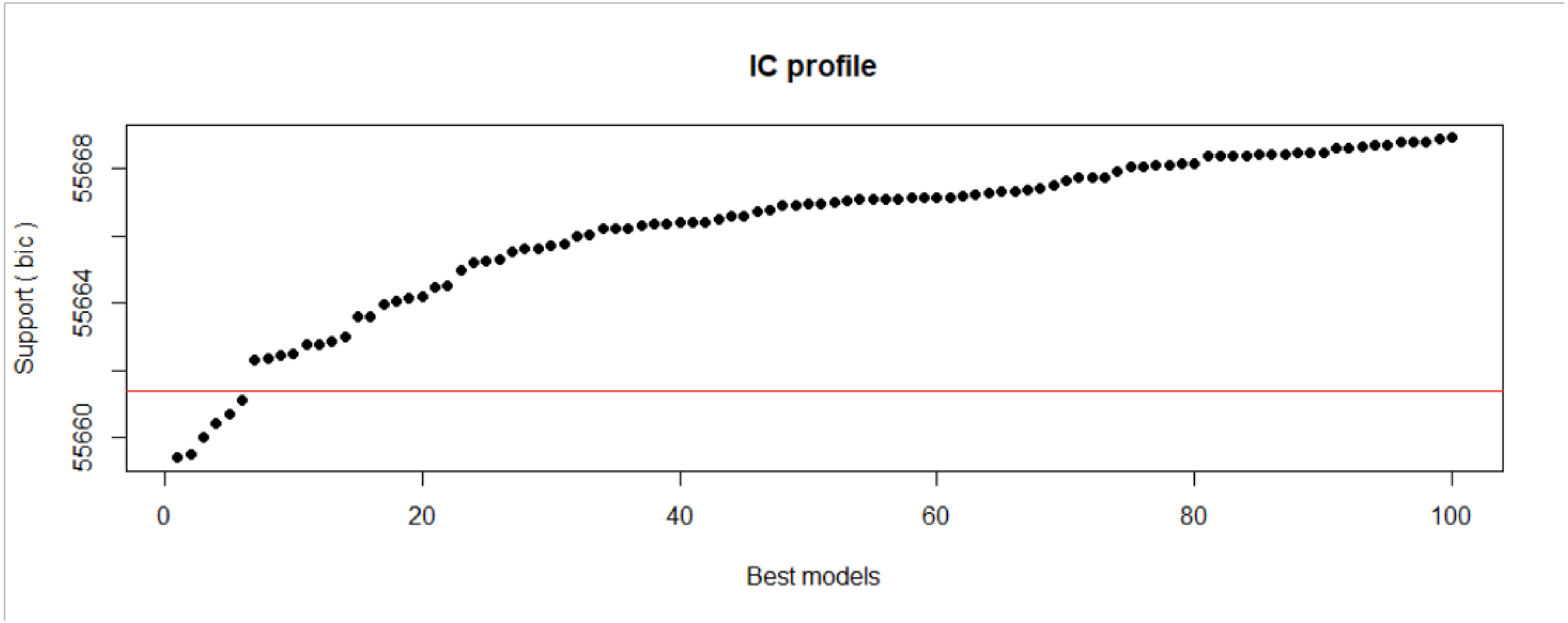
```
> print(train_3numcor,n=nrow(train_3numcor))
# A tibble: 35 x 2
   rowname            Sale_Price
   <chr>                   <dbl>
 1 Longitude            -0.255
 2 Enclosed_Porch       -0.131
 3 BsmtFin_SF_1         -0.129
 4 Kitchen_AbvGr        -0.112
 5 Low_Qual_Fin_SF      -0.0509
 6 Bsmt_Half_Bath       -0.0433
 7 Year_Sold            -0.0279
 8 Misc_Val             -0.0247
 9 BsmtFin_SF_2          0.00281
10 Three_season_porch    0.0342
11 Mo_Sold               0.0418
12 Pool_Area             0.0759
13 Screen_Porch          0.0848
14 Bedroom_AbvGr         0.150
15 Bsmt_Unf_SF           0.190
16 Lot_Frontage          0.203
17 Bsmt_Full_Bath        0.279
18 Lot_Area              0.285
```

glmulti

```
train_3num_sm<-train_3num %>% select(-Misc_Val,-Year_Sold,-BsmtFin_SF_2,-Three_season_porch,-Mo_Sold)
glm1<-glmulti(Sale_Price~.,data=train_3num_sm,crit="bic",level=1,method="d")
glm2<-glmulti(Sale_Price~ .,data=train_3num_sm,method="g",crit="bic",level=1,popsize=5,mutrate=0.05,sexrate=0.7,imm=0.2,deltaM=0.5,deltaB=0.1,conseq=6) #
glm3<-glmulti(Sale_Price~ .,data=train_3num_sm,method="g",crit="bic",level=1,popsize=100,mutrate=0.05,sexrate=0.7,imm=0.2,deltaM=0.5,deltaB=0.1,conseq=6)
glm4<-glmulti(Sale_Price~ .,data=train_3num_sm,method="g",crit="bic",level=1,popsize=1000,mutrate=0.05,sexrate=0.7,imm=0.2,deltaM=0.5,deltaB=0.1,conseq=6)

set.seed(42)  # for reproducibility
cv_modelglm4 <- train(
  form = Sale_Price ~ Lot_Frontage+Lot_Area+Year_Built+Year_Remod_Add+Mas_Vnr_Area+Bsmt_Unf_SF+Total_Bsmt_SF+First_Flr_SF+Second_Flr_SF+Bsmt_Full_Bath+Bed
  data = train_3num_sm,
  method = "lm",
  trControl = trainControl(method = "cv", number = 10)
)

pred_lmglm4<-predict(cv_modelglm4$finalModel,newdata=test_3)


test_3aug$pred_lmglm4<-pred_lmglm4
ggplot(test_3aug,aes(x=pred_lmglm4,y=Sale_Price))+
  geom_point()+stat_smooth(method=lm)+geom_abline(slope=1, intercept=0, col='red')
cor(test_3aug$pred_lmglm4,test_3aug$Sale_Price)^2
res_lmglm4<-test_3aug$Sale_Price-test_3aug$pred_lmglm4
(mean((res_lmglm4)^2))^0.5


summary(glm4)
print(glm4)
glm4@formulas[1:6]
tmp<-weightable(glm4)
tmp <- tmp[tmp$bic <= min(tmp$bic) + 2,]
tmp
plot(glm4, type="r")
plot(glm4, type="s")

pred_lmglm46best<-predict(glm4,select=6,newdata=test_3)

test_3aug$pred_lmglm46best<-as.vector(pred_lmglm46best$averages)
ggplot(test_3aug,aes(x=pred_lmglm46best,y=Sale_Price))+
  geom_point()+stat_smooth(method=lm)+geom_abline(slope=1, intercept=0, col='red')
cor(test_3aug$pred_lmglm46best,test_3aug$Sale_Price)^2
res_lmglm46best<-test_3aug$Sale_Price-test_3aug$pred_lmglm46best
(mean((res_lmglm46best)^2))^0.5

ggplot(test_3aug,aes(x=pred_lmglm46best,y=pred_lmglm4))+
  geom_point()+stat_smooth(method=lm)+geom_abline(slope=1, intercept=0, col='red')

ggplot(test_3aug,aes(x=(pred_lmglm46best-pred_lmglm4)))+geom_histogram(col='white')
```

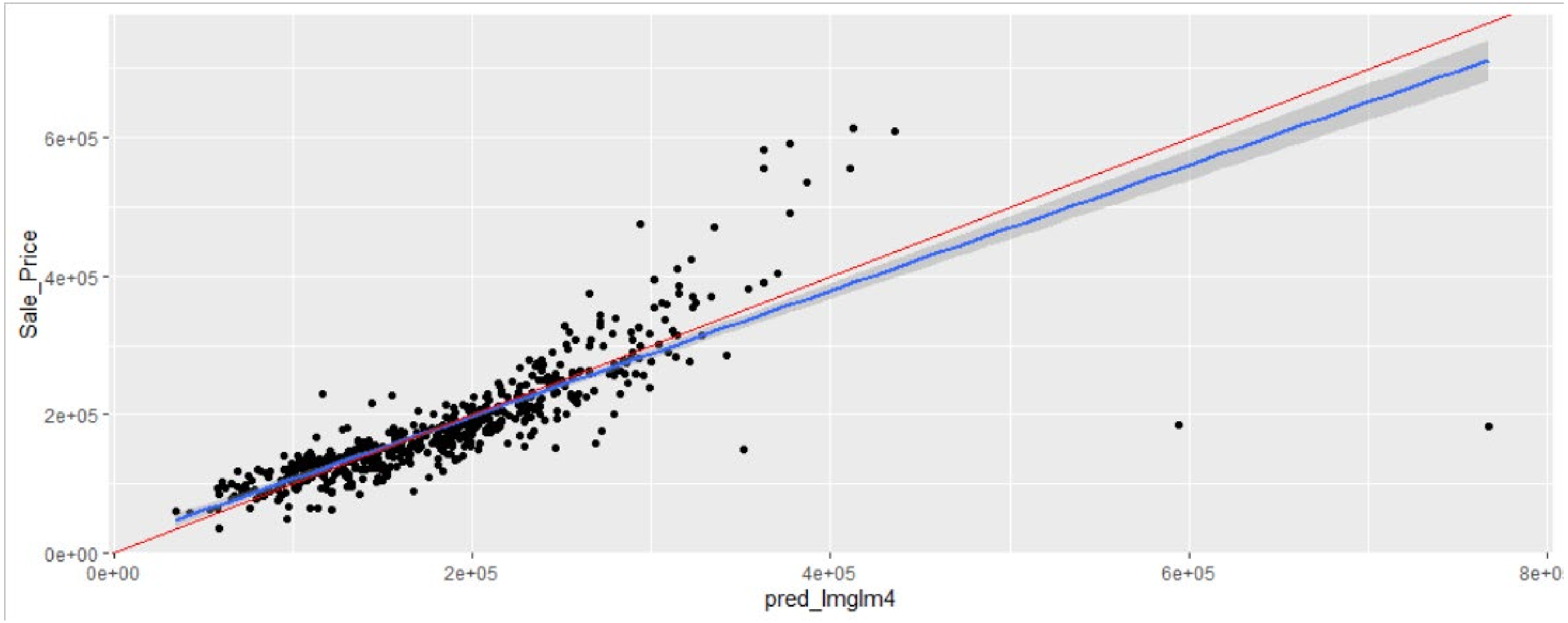
GPI

IC profile shows 6 best models are within 2 IC units, suggesting model-averaging

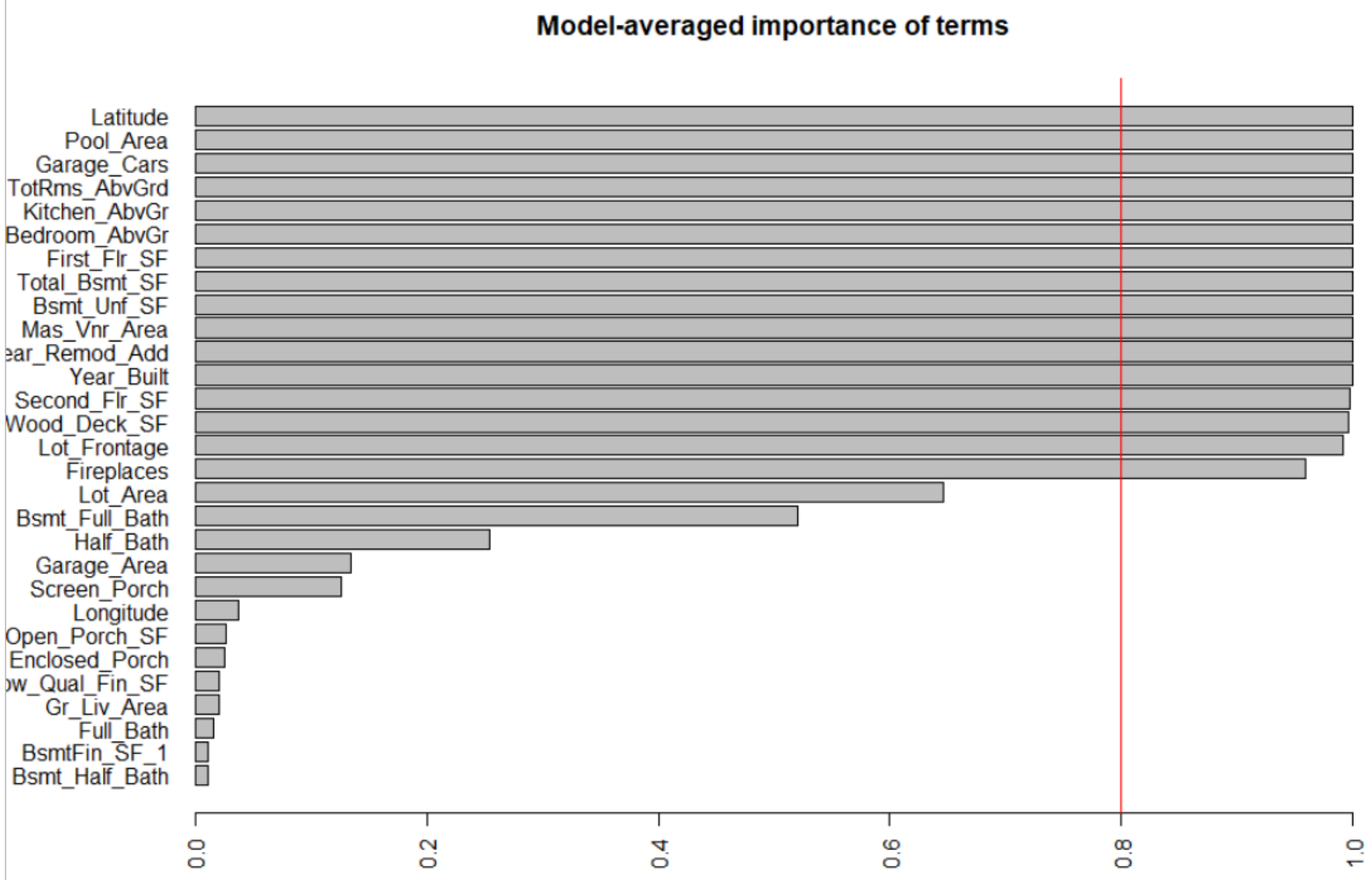


GPI

glmulti predictions on test data

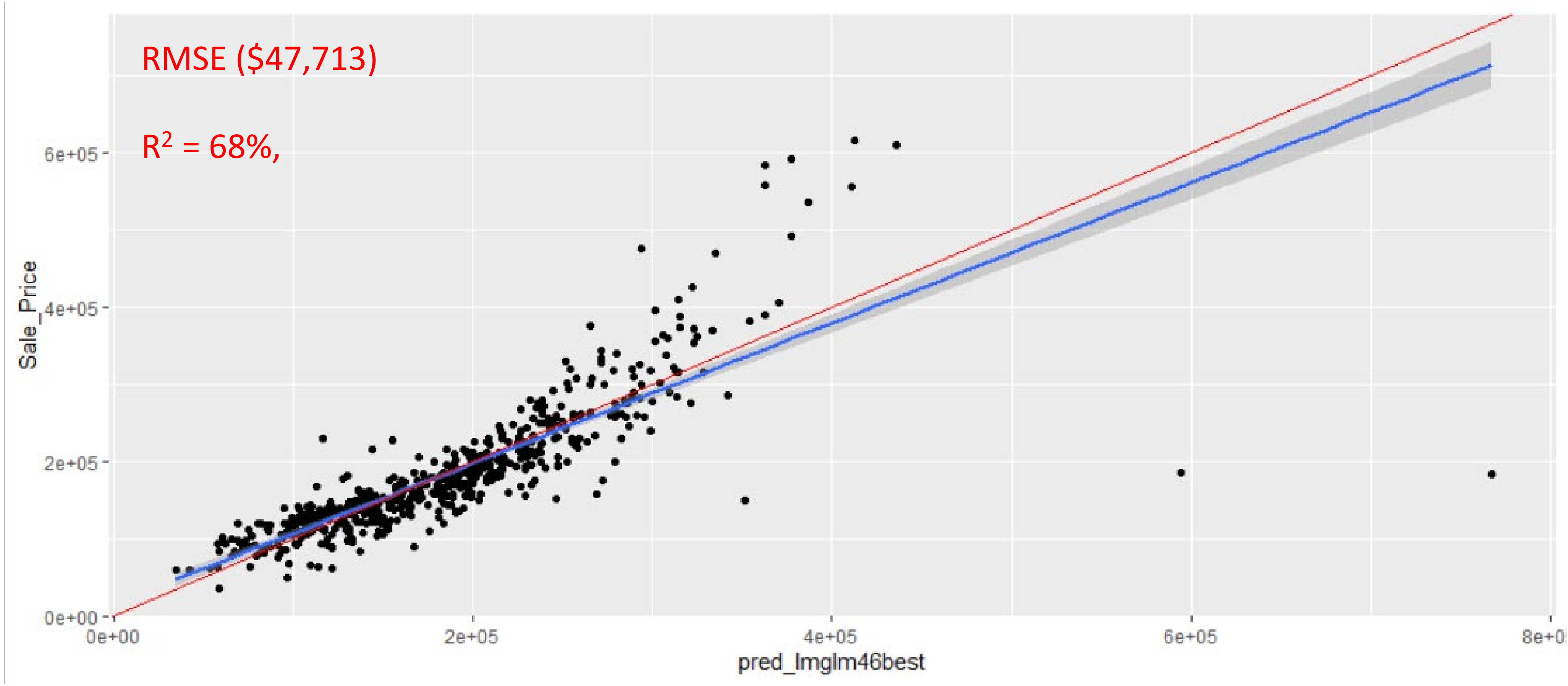


Importance based on weights/probabilities of variables in the models



GPI

Results on test essentially the same when use average of the 6 best models



Training RMSE was 33340, so some overfitting in this large number of models search.

GPI