

STAT 516 Lec 04

Multiple linear regression (part 2/2)

Karl Gregory

2026-09-23

Overall F test:

$$H_0: \beta_1 = \dots = \beta_p = 0$$

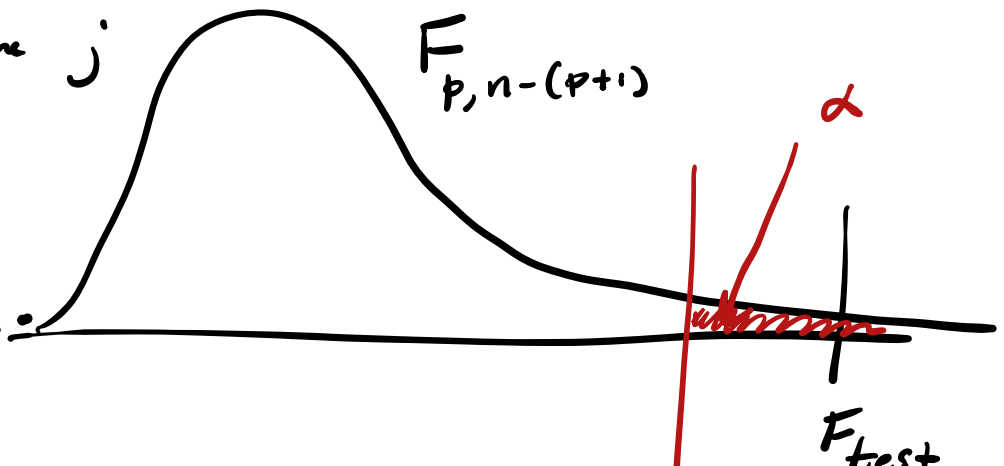
$$H_1: \beta_j \neq 0 \text{ for at least one } j$$

Reject H_0 if $F_{\text{test}} > F_{p, n-(p+1), \alpha}$

$$F_{\text{test}} = \frac{MS_{\text{Reg}}}{MS_{\text{Error}}}$$

$df = p$

$df = n - (p+1)$



SC apartments example

$F_{p, n-(p+1), \alpha}$

As in part 1/2, consider these data from “Apartment for Rent Classified” (2019)

```
link <- url("https://gregorkb.github.io/data/scapts.csv")
scapts <- read.csv(link)
head(scapts)
```

	price	nbath	nbed	sqft	city	pets
1	1199	2	3	1200	Columbia	yes
2	1199	2	3	1450	Columbia	yes
3	1299	2	3	1950	Columbia	yes
4	1175	2	3	1312	Columbia	yes
5	1349	2	4	2015	Columbia	yes
6	1225	2	3	1336	Columbia	yes

```
n <- nrow(scapts)
```

There are $n = 214$ observations.

Setup

Consider data $(Y_1, \mathbf{x}_1), \dots, (Y_n, \mathbf{x}_n)$, with each $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$.

The multiple linear regression model is

$$Y_i = \beta_0 + x_{i1}\beta_1 + \dots + x_{ip}\beta_p + \varepsilon_i, \quad i = 1, \dots, n,$$

where

- ▶ $\mathbf{x}_1, \dots, \mathbf{x}_n$ are vectors in $\mathbb{R}^p$ of covariate or predictor values.
- ▶ $Y_1, \dots, Y_n$ are the response values
- ▶ $\beta_0, \beta_1, \dots, \beta_p$ are the regression coefficients.
- ▶ $\varepsilon_1, \dots, \varepsilon_n$ are iid $\text{Normal}(0, \sigma^2)$ error terms.
- ▶ σ^2 is the error term variance.

Goals in multiple linear regression

In part 1/2, we addressed these goals:

1. Estimate the regression coefficients β_0 and $\beta_1, \dots, \beta_p$.
2. Estimate the error term variance σ^2 .
3. Perform inference on $\beta_1, \dots, \beta_p$.
4. Include interaction effects (this we did not do in SLR).
5. Build a CI for $\beta_0 + \beta_1 x_{\text{new},1} + \dots + \beta_p x_{\text{new},p}$ at any $\mathbf{x}_{\text{new}}$.
6. Build a prediction interval for Y at any $\mathbf{x}_{\text{new}}$.
7. Decompose the variation in Y into (sums of) sums of squares.
8. Check whether the model assumptions are satisfied.
9. Identify outliers and understand their effects.

In part 2/2 we focus on these:

10. Test for significance of a subset of covariates.
11. Understand how correlations among the covariates affect inferences.
12. Do variable selection.

Review of F distributions

For $W_1 \sim \chi_{\nu_1}^2(\phi)$, $W_2 \sim \chi_{\nu_2}^2$ independent, $R = \frac{W_1/\nu_1}{W_2/\nu_2} \sim F_{\nu_1, \nu_2}(\phi)$.

$F_{\nu_1, \nu_2}(\phi)$ denotes the F distribution with

- ▶ numerator degrees of freedom ν_1
- ▶ denominator degrees of freedom ν_2
- ▶ noncentrality parameter $\phi \geq 0$

If $\phi > 0$ the distribution is a non-central F distribution.

When $\phi = 0$ we just write F_{ν_1, ν_2} to denote the “central” F distribution.

We will encounter ratios of sums of squares which have F distributions.

Plot of some F distribution pdfs

```
nu1 <- c(1,2,3,5,5,5,50,50)
nu2 <- c(3,3,3,10,10,10,50,50)
phi <- c(0,0,0,0,4,8,0,4)
f <- seq(.01,4,length=200)
dfmat <- matrix(0,length(f),200)
for(j in 1:length(nu1)){

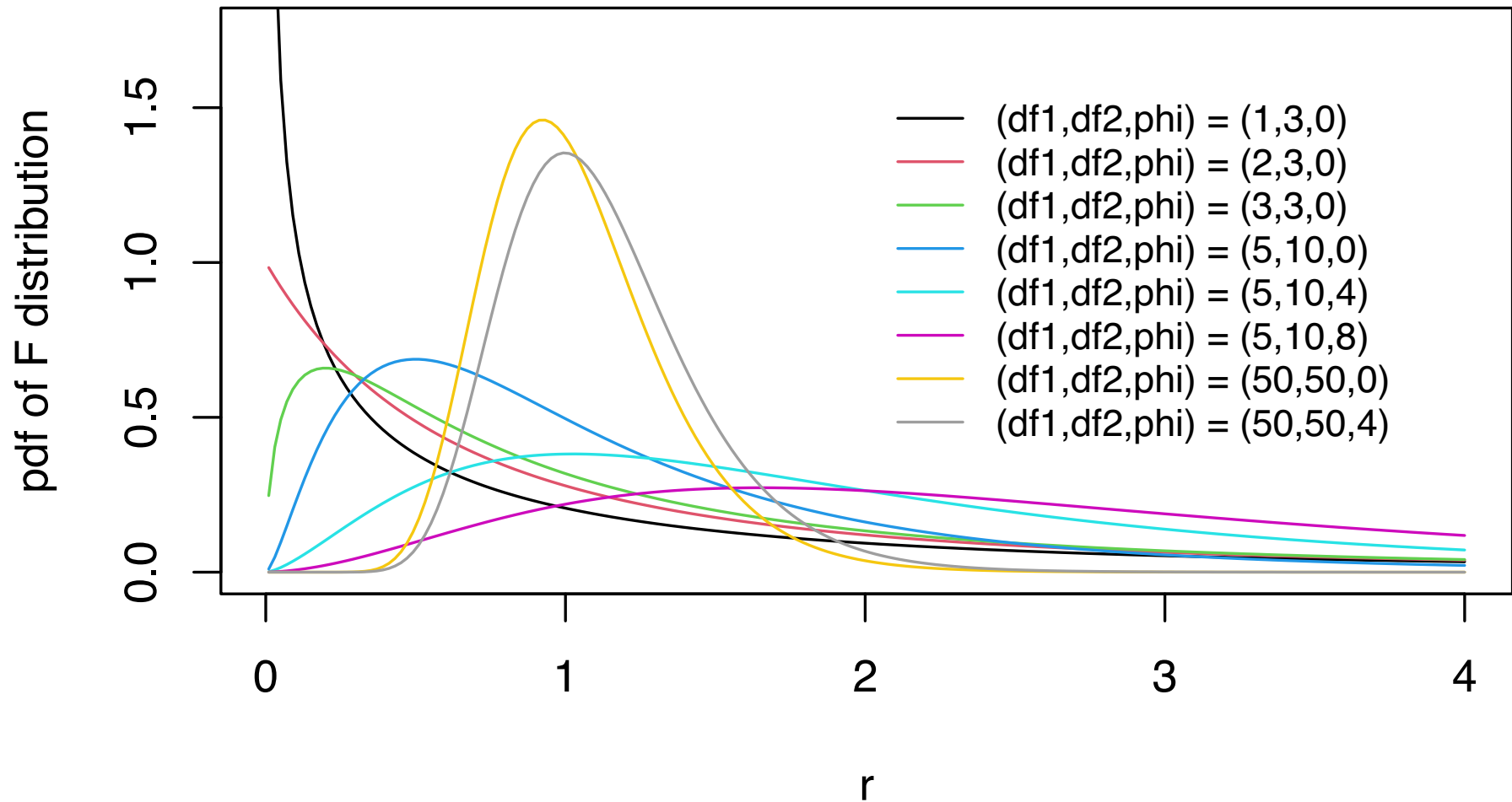
  dfmat[j,] <- df(f,df1 = nu1[j],df2=nu2[j],ncp=phi[j])

}
lab <- paste("(df1,df2,phi) = (",
  apply(cbind(nu1,nu2,phi),1,paste,collapse = ","),
  ")",sep="")
```

```

plot(NA,xlim = range(f),ylim = c(0,1.2*max(dfmat[-1,])),
     xlab = "r",
     ylab = "pdf of F distribution")
for(j in 1:length(nu1)) lines(dfmat[j,]~f, col = j)
legend(x = .5*max(f),y = 1.1*max(dfmat[-1,]),legend = lab,
      col = 1:length(nu1), lty = 1,bty = "n", cex = .8)

```



The overall F-test

We may wish to test whether *any* covariates are important, that is

$$H_0: \beta_j = 0 \text{ for all } j = 1, \dots, p.$$

The overall F-test of significance is carried out as follows:

1. Fit the model with all the covariates and obtain the value

$$F_{\text{test}} = \frac{\text{MS}_{\text{Reg}}}{\text{MS}_{\text{Error}}} \left(= \frac{\text{SS}_{\text{Reg}} / p}{\text{SS}_{\text{Error}} / (n - (p + 1))} \right)$$

2. Reject H_0 at α if $F_{\text{test}} > F_{p, n-(p+1), \alpha}$.
3. Obtain p-value is $P(F > F_{\text{test}})$, where $F \sim F_{p, n-(p+1)}$.

This test statistic and p-value are reported by `summary()` on `lm()`.

Exercise: Show that the test statistic of the overall F test can be written

$$F_{\text{test}} = \frac{\text{MS}_{\text{Reg}}}{\text{MS}_{\text{Error}}} = \frac{(n - (p + 1))}{p} \frac{R^2}{1 - R^2},$$

where R^2 is the coefficient of determination.

SC apartments example (cont)

```
lm_out <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets +  
             log(sqft):city, data = scapts)  
summary(lm_out)
```

Call:

```
lm(formula = log(price) ~ city + nbath + nbed + log(sqft) + pets +  
    log(sqft):city, data = scapts)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.58204	-0.13289	-0.00398	0.09305	0.68414

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	2.88590	0.68527	4.211	3.81e-05	***
•cityColumbia	1.52500	0.75905	2.009	0.04585	*
•cityGreenville	0.87340	0.88640	0.985	0.32563	
•cityRock Hill	0.33480	1.36863	0.245	0.80700	
•nbath	-0.13511	0.04508	-2.997	0.00306	**
•nbed	0.07974	0.03470	2.298	0.02260	*
•log(sqft)	0.62871	0.10617	5.922	1.34e-08	***
•petsyes	-0.00804	0.03089	-0.260	0.79490	
•cityColumbia:log(sqft)	-0.25861	0.11113	-2.327	0.02094	*
•cityGreenville:log(sqft)	-0.14616	0.13059	-1.119	0.26436	
•cityRock Hill:log(sqft)	-0.07234	0.19835	-0.365	0.71572	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2069 on 203 degrees of freedom

Multiple R-squared: 0.43, Adjusted R-squared: 0.4019

F-statistic: 15.31 on 10 and 203 DF, p-value: < 2.2e-16

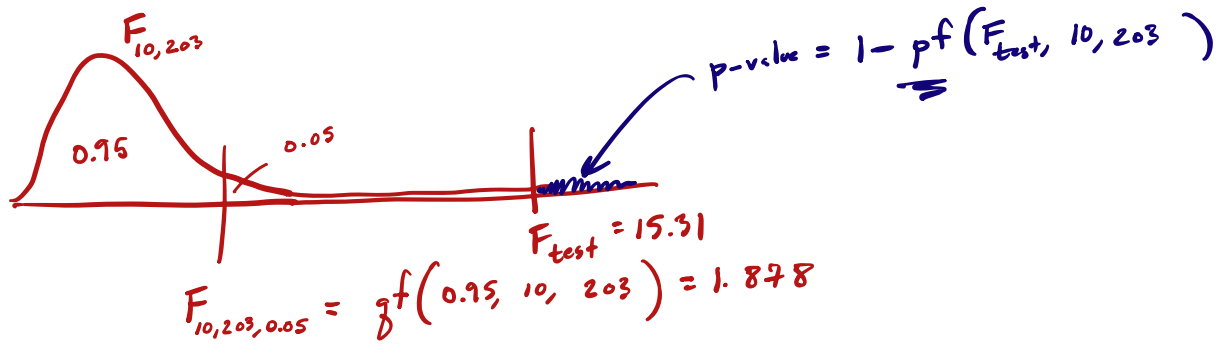
$n = 214$

$F_{test} = 15.31$

$\alpha = 0.05$

$= 2.2 \times 10^{-16}$

$n - (p+1) = 214 - (10+1) = 203$



Exercise: Suppose you fit a regression model with $p = 10$ predictors on a data set with 214 observations, and you obtain $\hat{\sigma} = 0.207$ and $R^2 = 0.430$. Use this information to fill in the entire ANOVA table:

Source	Df	SS	MS	F value	p-value
Reg	p	SS_{Reg}	MS_{Reg}	F_{test}	$P(F > F_{\text{test}})$
Error	$n - (p + 1)$	SS_{Error}	MS_{Error}		
Total	$n - 1$	SS_{Tot}			

$$SS_{Error} = \sum_{i=1}^n \hat{\epsilon}_i^2$$

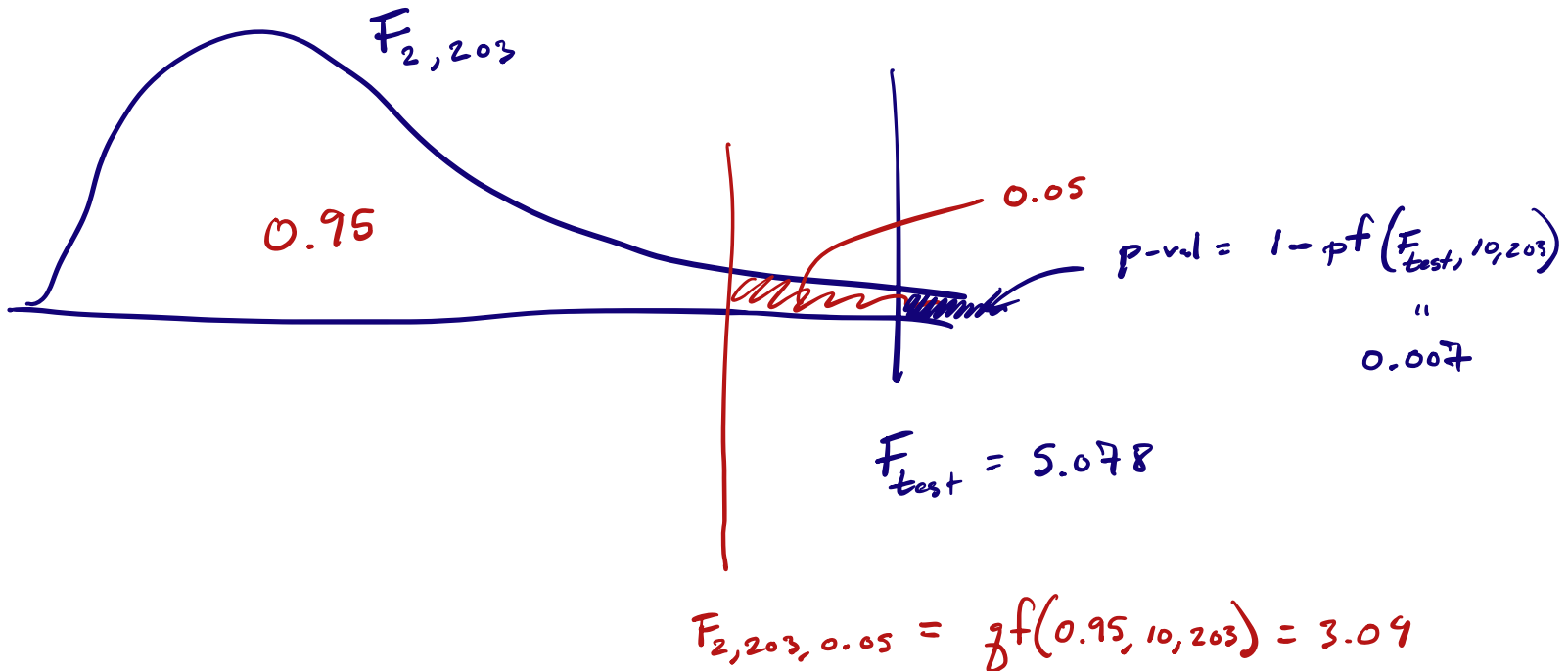
↑
residuals

$$H_0: \beta_{nbhd} = \beta_{nbath} = 0$$

$$H_1: \text{not true}$$

$$F_{test} = \frac{(SS_{Error}(\text{Reduced}) - SS_{Error}(\text{Full})) / 2}{SS_{Error}(\text{Full}) / \underbrace{(n - (p+1))}_{203}}$$

$$= 5.078$$



SC apartments example (cont)

Check whether the number of bedrooms or bathrooms contributes significantly to the rent.

That is test $H_0: \beta_{\text{nbed}} = \beta_{\text{nbath}} = 0$.

```
lm_red <- lm(log(price) ~ city + log(sqft) + pets + log(sqft):city, data = scapts)
lm_full <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets
              + log(sqft):city, data = scapts)
SSE_red <- sum(lm_red$residuals^2)
SSE_full <- sum(lm_full$residuals^2)
s <- 2 # significance of two covariates being tested
Ftest <- (SSE_red - SSE_full)/s / ( SSE_full / (n - (p + 1)))
alpha <- 0.05
F_crit <- qf(1 - alpha, s, n - (p + 1))
pval <- 1 - pf(Ftest, s, n - (p + 1))
```

We obtain $F_{\text{test}} = 5.078$ and $F_{s, n-(p+1), 0.05} = 3.04$, and the p-value is 0.007.

Q: Does city make any difference
in the model?

SC apartments example (cont)

Check whether the city makes any difference in the price.

```
lm_red <- lm(log(price) ~ log(sqft) + pets, data = scapts)
lm_full <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets
              + log(sqft):city, data = scapts)
SSE_red <- sum(lm_red$residuals^2)
SSE_full <- sum(lm_full$residuals^2)
s <- 6 # significance of this many columns of X being tested
Ftest <- (SSE_red - SSE_full)/s / ( SSE_full / (n - (p + 1)))
alpha <- 0.05
F_crit <- qf(1 - alpha, s, n - (p + 1))
pval <- 1 - pf(Ftest, s, n - (p + 1))
```

We obtain $F_{\text{test}} = 8.566$ and $F_{s, n-(p+1), 0.05} = 2.143$, and the p-value is 0.

Full-reduced model F test for a single covariate

If we test $H_0: \beta_j = 0$ for a single covariate using the full-reduced model F test, the test statistic F_{test} will be equal to the square of the test statistic T_{test} for testing $H_0: \beta_j = 0$ in the full model.

```
lm_red <- lm(log(price) ~ city + nbath + nbed + log(sqft) +  
             log(sqft):city, data = scapts)  
lm_full <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets  
             + log(sqft):city, data = scapts)  
SSE_red <- sum(lm_red$residuals^2)  
SSE_full <- sum(lm_full$residuals^2)  
s <- 1 # significance of a single covariate being tested  
Ftest <- (SSE_red - SSE_full)/s / ( SSE_full / (n - (p + 1)))  
pval <- 1-pf(Ftest,s,n-(p+1))
```

We obtain $\sqrt{F} = 0.2602945 = |T_{\text{test}}|$. The p value is 0.7949004.
Compare this to that given in the `summary()` output.

Effect of correlations among the covariates

From before $\text{Var } \hat{\beta}_j = \sigma^2 \Omega_{jj}/n$. An alternate expression gives

$$\text{Var } \hat{\beta}_j = \frac{1}{1 - R_j^2} \sigma^2 \sum_{i=1}^n (x_{ji} - \bar{x}_j)^2$$

V.I.F.

spread of values of covariate j

where R_j^2 is the R^2 from regressing x_j on the other covariates.

So multicollinearity of x_j with the other covariates “inflates” $\text{Var } \hat{\beta}_j$:

- Makes confidence intervals for β_j wider.
- Makes tests of $H_0: \beta_j = 0$ vs $H_1: \beta_j \neq 0$ less powerful.

Call $\frac{1}{1 - R_j^2}$ the variance inflation factor (VIF) for x_j , $j = 1, \dots, p$.

For “nbed” $R_{\text{nbed}}^2 = 0.7482$. So $\text{VIF}_{\text{nbed}} = \frac{1}{1 - 0.7482} \approx 4$

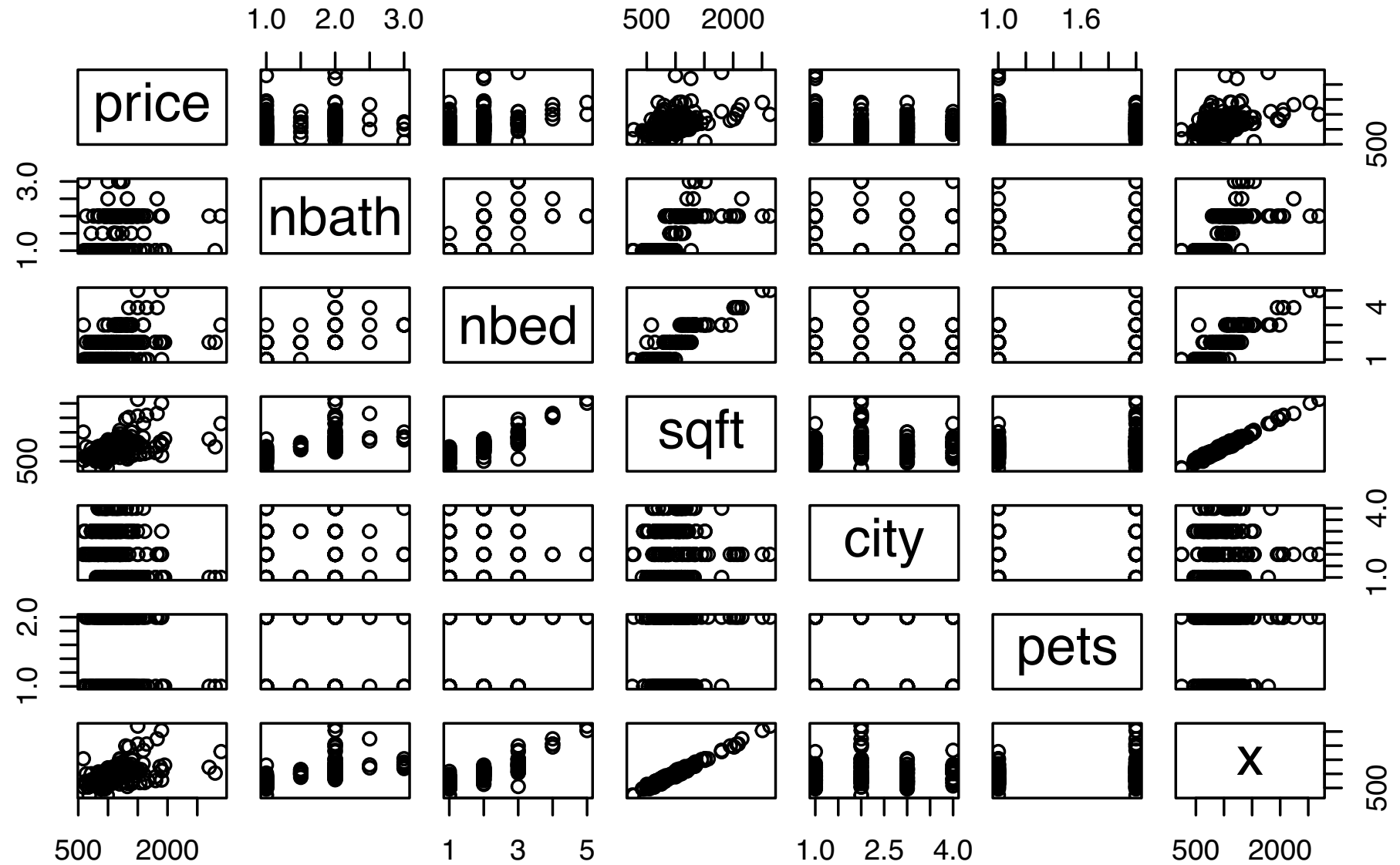
VIFs in SC apartment prices example

Add to the data set a spurious predictor highly correlated with the square footage.

Check the effect of this on our inferences for $\beta_{\log(\text{sqft})}$.

```
# make new x correlated with square feet  
x <- scapts$sqft + rnorm(n,0,50)  
scaptsx <- cbind(scapts,x)
```

```
plot(scptsx)
```



```
lm_out <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets
             + log(sqft):city, data = scapts)
confint(lm_out)
```

	2.5 %	97.5 %
(Intercept)	1.53473861	4.23706727
cityColumbia	0.02837274	3.02163477
cityGreenville	-0.87432396	2.62112302
cityRock Hill	-2.36376193	3.03336081
nbath	-0.22398601	-0.04622684
nbed	0.01131394	0.14816125
log(sqft)	0.41937140	0.83804586
petsyes	-0.06894052	0.05286101
cityColumbia:log(sqft)	-0.47771265	-0.03949987
cityGreenville:log(sqft)	-0.40364746	0.11132373
cityRock Hill:log(sqft)	-0.46342341	0.31874865

$\beta_{\log(\text{sqft})}$

```
lmx_out <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets
              + log(sqft):city + log(x), data = scapts)
confint(lmx_out)
```

	2.5 %	97.5 %
(Intercept)	1.51345088	4.23595308
cityColumbia	0.02384058	3.06897310
cityGreenville	-0.87710201	2.62707219
cityRock Hill	-2.36463428	3.04834343
nbath	-0.22481008	-0.04629188
nbed	0.01132651	0.14863282
log(sqft)	0.07333335	1.10626003
petsyes	-0.06951176	0.05281545
log(x)	-0.45236315	0.53366767
cityColumbia:log(sqft)	-0.48452610	-0.03887673
cityGreenville:log(sqft)	-0.40468141	0.11162162
cityRock Hill:log(sqft)	-0.46569972	0.31882982

95% C.I. for $\beta_{\log(sqft)}$

The width of the CI for $\beta_{\log(sqft)}$ was 0.419.

With the new covariate the width of the CI for $\beta_{\log(sqft)}$ becomes 1.033.

So including $\log(x)$ in the model makes our estimation of $\beta_{\log(sqft)}$ less accurate.

```
summary(lm_out)
```

Call:

```
lm(formula = log(price) ~ city + nbath + nbed + log(sqft) + pets +  
    log(sqft):city, data = scapts)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-0.58204	-0.13289	-0.00398	0.09305	0.68414

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	2.88590	0.68527	4.211	3.81e-05	***
cityColumbia	1.52500	0.75905	2.009	0.04585	*
cityGreenville	0.87340	0.88640	0.985	0.32563	
cityRock Hill	0.33480	1.36863	0.245	0.80700	
nbath	-0.13511	0.04508	-2.997	0.00306	**
nbed	0.07974	0.03470	2.298	0.02260	*
log(sqft)	0.62871	0.10617	5.922	1.34e-08	***
petsyes	-0.00804	0.03089	-0.260	0.79490	
cityColumbia:log(sqft)	-0.25861	0.11113	-2.327	0.02094	*
cityGreenville:log(sqft)	-0.14616	0.13059	-1.119	0.26436	
cityRock Hill:log(sqft)	-0.07234	0.19835	-0.365	0.71572	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2069 on 203 degrees of freedom

Multiple R-squared: 0.43, Adjusted R-squared: 0.4019

F-statistic: 15.31 on 10 and 203 DF, p-value: < 2.2e-16

The p-value for log(sqft) is very small.

```
summary(lmx_out)
```

Call:

```
lm(formula = log(price) ~ city + nbath + nbed + log(sqft) + pets +  
    log(sqft):city + log(x), data = scapts)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.58243	-0.13326	-0.00447	0.09537	0.68220

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	2.874702	0.690368	4.164	4.63e-05	***
cityColumbia	1.546407	0.772180	2.003	0.04655	*
cityGreenville	0.874985	0.888583	0.985	0.32595	
cityRock Hill	0.341855	1.372614	0.249	0.80357	
nbath	-0.135551	0.045268	-2.994	0.00309	**
nbed	0.079980	0.034818	2.297	0.02264	*
log(sqft)	0.589797	0.261928	2.252	0.02541	*
petsyes	-0.008348	0.031020	-0.269	0.78811	
log(x)	0.040652	0.250036	0.163	0.87101	
cityColumbia:log(sqft)	-0.261701	0.113007	-2.316	0.02157	*
cityGreenville:log(sqft)	-0.146530	0.130923	-1.119	0.26438	
cityRock Hill:log(sqft)	-0.073435	0.198940	-0.369	0.71242	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2074 on 202 degrees of freedom

Multiple R-squared: 0.43, Adjusted R-squared: 0.399

F-statistic: 13.85 on 11 and 202 DF, p-value: < 2.2e-16

The p-value for log(sqft) is not nearly as small when log(x) is included!

Getting VIFs with `vif()` from the `car` package

We can use the R package `car` from Fox and Weisberg (2019).

First time must install the package with `install.package("car")`.

```
library(car)
vif(lm_out)
```

	GVIF	Df	$GVIF^{(1/(2*Df))}$
city	1.894814e+08	3	23.965938
nbath	2.972991e+00	1	1.724236
nbed	3.970734e+00	1	1.992670
log(sqft)	6.181892e+00	1	2.486341
pets	1.191902e+00	1	1.091743
city:log(sqft)	1.969984e+08	3	24.121841

```
vif(lmx_out)
```

	GVIF	Df	$GVIF^{(1/(2*Df))}$
city	1.966477e+08	3	24.114678
nbath	2.983877e+00	1	1.727390
nbed	3.978008e+00	1	1.994494
log(sqft)	3.744500e+01	1	6.119232
pets	1.196376e+00	1	1.093790
log(x)	3.495488e+01	1	5.912265
city:log(sqft)	2.040551e+08	3	24.263750

Note the change in VIF for `log(sqft)` due to including `x` in the model!

Sometimes the number of potentially important predictors is quite large.

Large p tends to increase the VIFs, leading to low power.

So we may wish to discard some predictors. We briefly discuss:

1. Best subset selection with Mallow's $C(p)$ ← optional
2. Forward and backward stepwise selection with AIC
3. LASSO selection ← optional

And most importantly:

- The dangers of naïve post-selection inference!!

Best subset selection with Mallows's C_p

Given q available covariates, there are 2^q possible subset models (why?).

Mallows's C_p can be used to compare subset models: Let

$$C_p = (n - (p + 1)) \left[\frac{\text{MS}_{\text{Error}}(\text{subset})}{\text{MS}_{\text{Error}}(\text{all})} - 1 \right] + (p + 1),$$

where

- ▶ p is the number of predictors *in the subset model*.
- ▶ $\text{MS}_{\text{Error}}(\text{subset})$ is the MS_{Error} of the subset model.
- ▶ $\text{MS}_{\text{Error}}(\text{all})$ is the MS_{Error} of the model with all the covariates.

If the subset model is adequate, $\text{MS}_{\text{Error}}(\text{subset})$ estimates the same target as $\text{MS}_{\text{Error}}(\text{all})$, so the first term should be small and $C_p \approx p + 1$.

Can look at C_p values for all subset models of each size $p = 0, 1, 2, \dots, q$

Want smallest model such that $C_p \approx p + 1$.

Mallow's C_p on the rental properties data

Compute Mallow's C_p for a single subset model:

```
lm_all <- lm(log(price) ~ city + nbath + nbed + log(sqft) + pets + log(
MSE_all <- sum(lm_all$residuals^2) / (n - (p+1))
```

```
lm_sub <- lm(log(price) ~ city + nbath + nbed, data = scapts)
```

```
p_sub <- 5
```

```
MSE_sub <- sum(lm_sub$residuals^2) / (n - (p_sub+1))
```

```
C_sub <- (n - (p_sub+1))*(MSE_sub / MSE_all - 1) + (p_sub+1)
```

```
C_sub
```

```
[1] 44.4997
```

This value is too large; the subset is not a good one.

The regsubsets() function from the R package leaps

```
library(leaps) # first time run install.packages("leaps")
regsubsets_out <- regsubsets(log(price) ~ city + nbath + nbed + log(sqft) + pets + log(sqft):city, data = scapts)
summary(regsubsets_out)
```

Subset selection object

Call: regsubsets.formula(log(price) ~ city + nbath + nbed + log(sqft) +
pets + log(sqft):city, data = scapts)

10 Variables (and intercept)

	Forced in	Forced out
cityColumbia	FALSE	FALSE
cityGreenville	FALSE	FALSE
cityRock Hill	FALSE	FALSE
nbath	FALSE	FALSE
nbed	FALSE	FALSE
log(sqft)	FALSE	FALSE
petsyes	FALSE	FALSE
cityColumbia:log(sqft)	FALSE	FALSE
cityGreenville:log(sqft)	FALSE	FALSE
cityRock Hill:log(sqft)	FALSE	FALSE

1 subsets of each size up to 8

Selection Algorithm: exhaustive

	cityColumbia	cityGreenville	cityRock Hill	nbath	nbed	log(sqft)	petsyes
1 (1)	" "	" "	" "	" "	" "	"*	" "
2 (1)	" "	" "	" "	" "	" "	"*	" "
3 (1)	" "	" "	"*	" "	" "	"*	" "
4 (1)	" "	" "	"*	" "	" "	"*	" "
5 (1)	" "	" "	"*	"*	" "	"*	" "
6 (1)	" "	" "	"*	"*	"*	"*	" "
7 (1)	"*	" "	" "	"*	"*	"*	" "
8 (1)	"*	"*	" "	"*	"*	"*	" "

	cityColumbia:log(sqft)	cityGreenville:log(sqft)
1 (1)	" "	" "
2 (1)	"*	" "
3 (1)	"*	" "
4 (1)	"*	"*
5 (1)	"*	"*
6 (1)	"*	"*
7 (1)	"*	"*
8 (1)	"*	"*

cityRock Hill:log(sqft)

1 (1) " "

FIFA data

Wages and stats of male FIFA players in 2022 from Pedersen (2022).

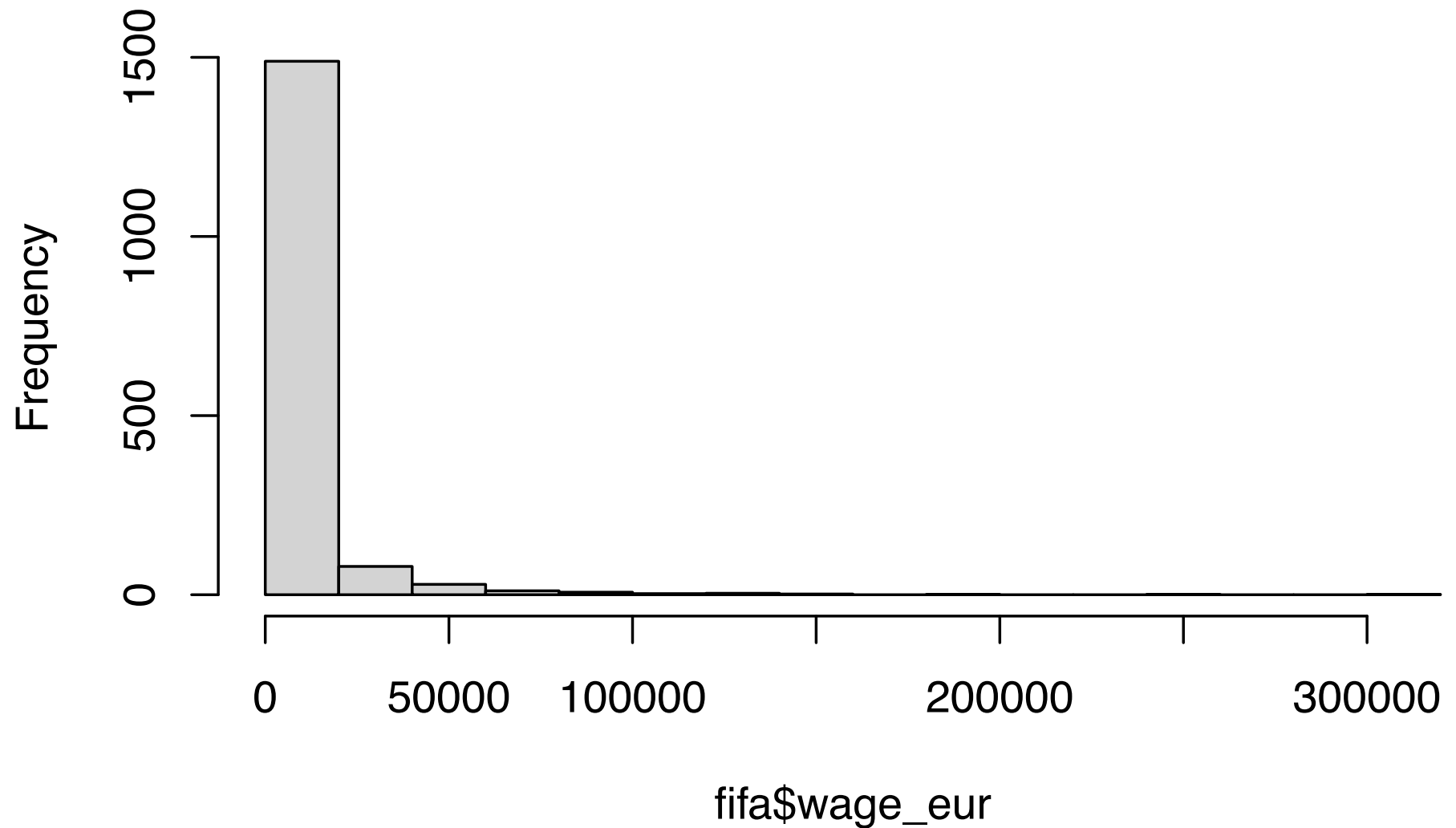
```
link <- url("https://people.stat.sc.edu/gregorkb/data/fifa_usge.csv")
fifa <- read.csv(link)
colnames(fifa)
```

[1] "wage_eur"	"age"
[3] "height_cm"	"weight_kg"
[5] "nationality_name"	"overall"
[7] "potential"	"attacking_crossing"
[9] "attacking_finishing"	"attacking_heading_accuracy"
[11] "attacking_short_passing"	"attacking_volleys"
[13] "skill_dribbling"	"skill_curve"
[15] "skill_fk_accuracy"	"skill_long_passing"
[17] "skill_ball_control"	"movement_acceleration"
[19] "movement_sprint_speed"	"movement_agility"
[21] "movement_reactions"	"movement_balance"
[23] "defending_standing_tackle"	"defending_sliding_tackle"
[25] "goalkeeping_diving"	"goalkeeping_handling"
[27] "goalkeeping_kicking"	"goalkeeping_positioning"
[29] "goalkeeping_reflexes"	

Predict wage from 28 covariates? Too many sub-models to consider!

```
hist(fifa$wage_eur)
```

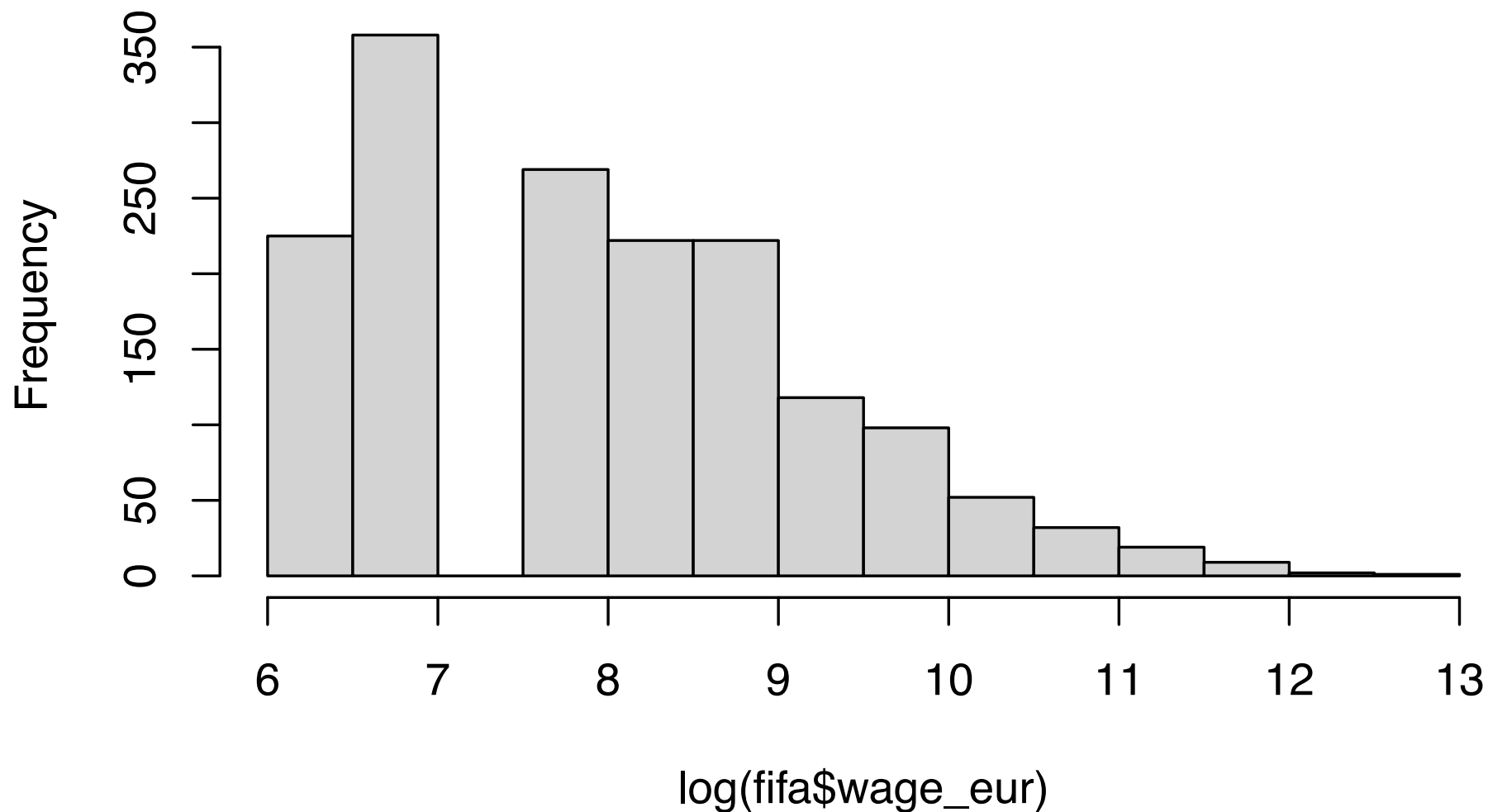
Histogram of fifa\$wage_eur



The wage distribution has some high outlying observations.

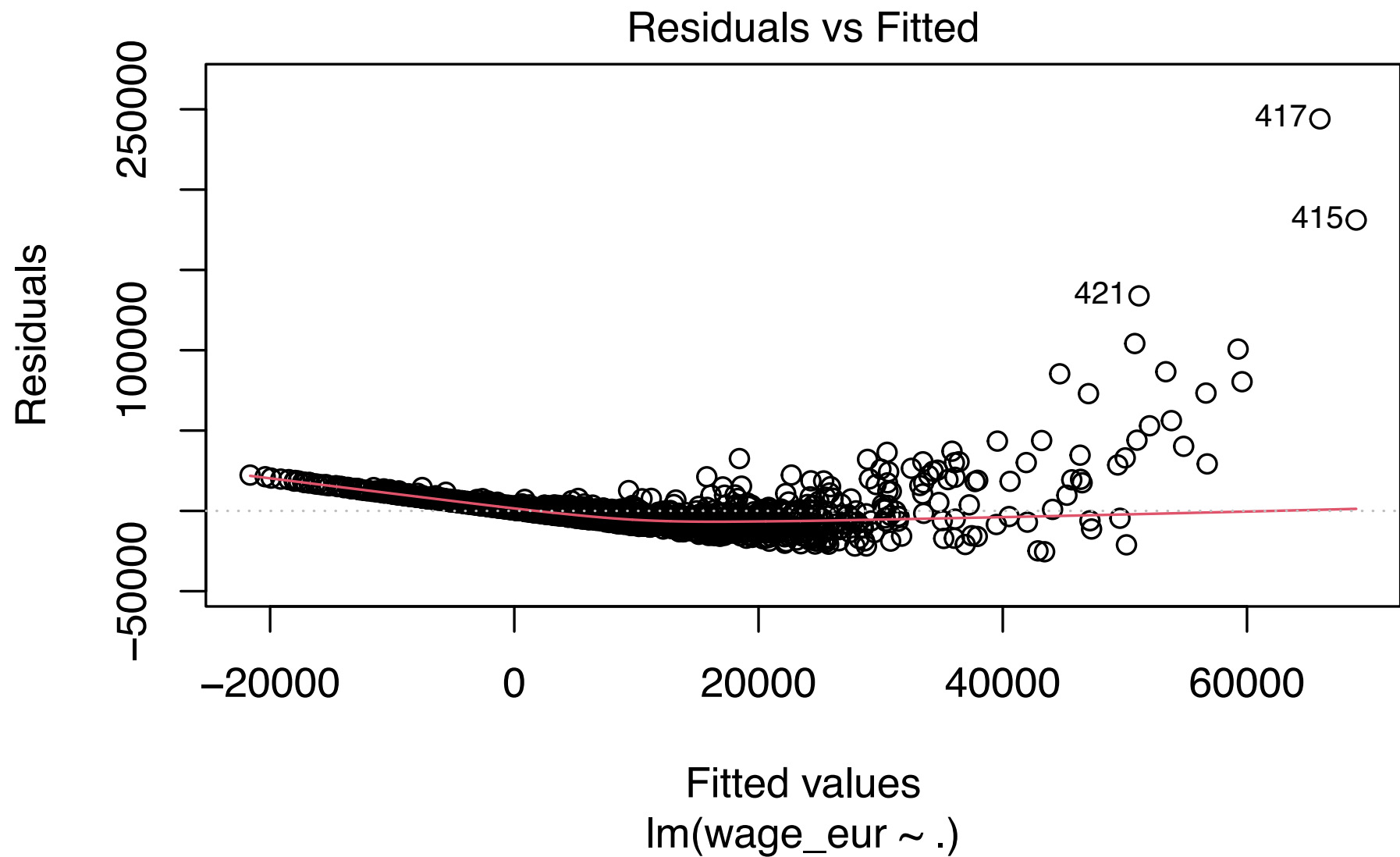
```
hist(log(fifa$wage_eur))
```

Histogram of log(fifa\$wage_eur)

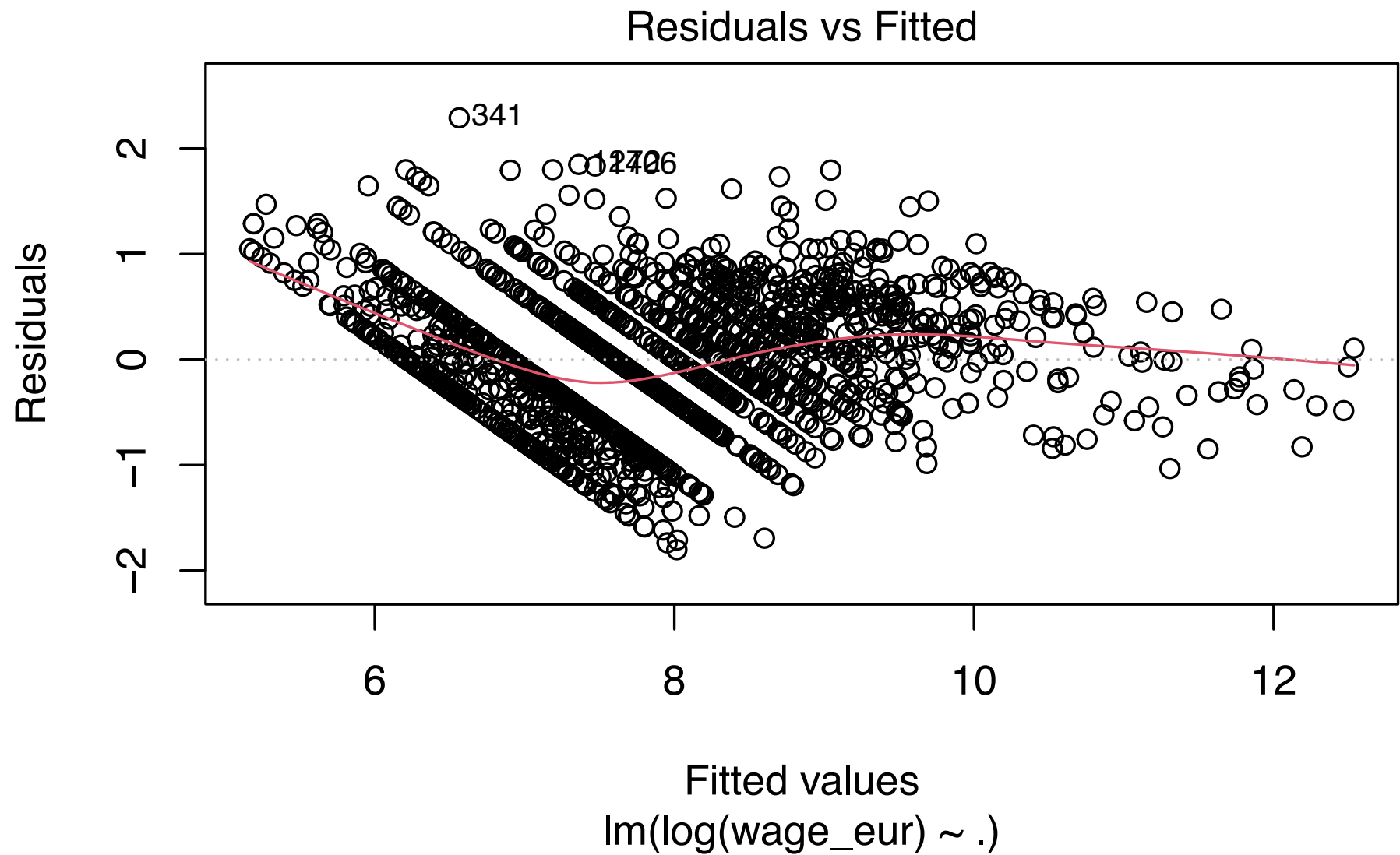


Perhaps better to consider the log of the wage.

```
lm_wage <- lm(wage_eur ~ ., data = fifa)
plot(lm_wage, which = 1)
```



```
lm_logwage <- lm(log(wage_eur) ~ ., data = fifa)
plot(lm_logwage, which = 1)
```



赤池 弘次 (あかいけひろつぐ)



Introduced Akaike's Information Criterion (AIC).

Akaike's Information Criterion (AIC) for comparing models

For a given model, i.e. set of covariates, AIC is defined as

$$\boxed{\text{AIC}} = \boxed{2(p + 1)} - 2 \underbrace{\ell(\hat{\sigma}^2, \hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_p)}_{\text{log-likelihood}}.$$

covariates

larger values mean better prediction

The log-likelihood is the log of the joint pdf of the data (STAT 512).

AIC can be used to compare several models for the same data.

The “best” model is the one which minimizes AIC.

The extractAIC() function

The extractAIC function in R returns a modified version of AIC:

$$AIC^* = 2(p + 1) + n \log(SS_{\text{Error}} / n)$$

```
lm_out <- lm(log(wage_eur) ~ age + potential, data = fifa)
extractAIC(lm_out) # gives value p + 1 as well as AIC value
```

```
[1]      3.0000 -860.2624
```

```
# compute it "manually"
n <- nrow(fifa)
p <- 2
2*(p+1) + n * log(sum(lm_out$residuals^2)/n)
```

```
[1] -860.2624
```

Comparing models using AIC

Compare two models for the FIFA data with AIC:

```
lm1 <- lm(log(wage_eur) ~ age + potential + height_cm, data = fifa)
extractAIC(lm1)
```

```
[1]      4.0000 -858.5413
```

```
lm2 <- lm(log(wage_eur) ~ height_cm + overall, data = fifa)
extractAIC(lm2)
```

```
[1]      3.000 -1377.386
```

The second model has a smaller value of AIC, so it is better according to this criterion.

Stepwise selection based on AIC

Stepwise selection:

- ▶ Backward: Begin with all the predictors and remove one at a time.
- ▶ Forward: Begin with no predictors and add one at a time.

In each step remove/add predictor to get largest decrease in AIC.

If a decrease in AIC is not possible, stop.

Stepwise selection with fifa data

Use the `step()` function for backward selection:

```
lm_intercept <- lm(log(wage_eur) ~ 1, data = fifa)
lm_all <- lm(log(wage_eur) ~ ., data = fifa)

# backward selection
step_back <- step(lm_all,
                  direction = "backward",
                  scope = formula(lm_all),
                  trace = 0) # suppress printed output
```

Call:

```
lm(formula = log(wage_eur) ~ age + height_cm + nationality_name +
    overall + potential + attacking_crossing + attacking_finishing +
    attacking_heading_accuracy + attacking_volleys + skill_dribbling +
    skill_fk_accuracy + skill_ball_control + movement_sprint_speed +
    movement_agility + movement_reactions + movement_balance +
    defending_sliding_tackle, data = fifa)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.81770	-0.42583	-0.00742	0.44183	2.26000

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-1.285472	1.056816	-1.216	0.22403
age	-0.018033	0.008311	-2.170	0.03018 *
height_cm	-0.012306	0.005090	-2.418	0.01574 *
nationality_nameusa	-0.102428	0.038424	-2.666	0.00776 **
overall	0.153692	0.007624	20.159	< 2e-16 ***
potential	0.025805	0.006486	3.978	7.25e-05 ***
attacking_crossing	0.004266	0.002123	2.009	0.04472 *
attacking_finishing	-0.003433	0.002427	-1.415	0.15740
attacking_heading_accuracy	0.004529	0.001829	2.476	0.01337 *
attacking_volleys	0.005065	0.002464	2.056	0.03995 *
skill_dribbling	0.006814	0.003818	1.785	0.07450 .
skill_fk_accuracy	0.003723	0.001845	2.018	0.04372 *
skill_ball_control	-0.005958	0.004151	-1.435	0.15141
movement_sprint_speed	-0.002731	0.001744	-1.566	0.11754
movement_agility	-0.008042	0.002766	-2.907	0.00370 **
movement_reactions	0.011193	0.003620	3.092	0.00202 **
movement_balance	-0.005163	0.002830	-1.825	0.06824 .
defending_sliding_tackle	-0.004565	0.001446	-3.156	0.00163 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6215 on 1609 degrees of freedom

Multiple R-squared: 0.7725, Adjusted R-squared: 0.7701

F-statistic: 321.4 on 17 and 1609 DF, p-value: < 2.2e-16

Use the `step()` function for forward selection:

```
# forward selection
step_forw <- step(lm_intercept,
                  direction = "forward",
                  scope = formula(lm_all),
                  trace = 0) # suppress printed output
```

```
summary(step_forw)
```

Call:

```
lm(formula = log(wage_eur) ~ overall + potential + attacking_volleys +  
    movement_agility + skill_fk_accuracy + nationality_name +  
    movement_reactions + defending_sliding_tackle + attacking_crossing +  
    age, data = fifa)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.93782	-0.42630	-0.00072	0.44823	2.29004

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-3.7485849	0.3310448	-11.323	< 2e-16	***
overall	0.1510192	0.0074671	20.225	< 2e-16	***
potential	0.0259892	0.0064344	4.039	5.62e-05	***
attacking_volleys	0.0042858	0.0016009	2.677	0.00750	**
movement_agility	-0.0102281	0.0016175	-6.324	3.30e-10	***
skill_fk_accuracy	0.0038679	0.0017634	2.193	0.02841	*
nationality_nameusa	-0.0759454	0.0366090	-2.075	0.03819	*
movement_reactions	0.0113347	0.0036018	3.147	0.00168	**
defending_sliding_tackle	-0.0026760	0.0009261	-2.889	0.00391	**
attacking_crossing	0.0042171	0.0018421	2.289	0.02219	*
age	-0.0163448	0.0080810	-2.023	0.04328	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6231 on 1616 degrees of freedom

Multiple R-squared: 0.7703, Adjusted R-squared: 0.7689

F-statistic: 542 on 10 and 1616 DF, p-value: < 2.2e-16

Forward and backward stepwise selection may give different models!

LASSO selection

The LASSO estimators $\hat{\beta}_0^L, \hat{\beta}_1^L, \dots, \hat{\beta}_p^L$ are obtained by minimizing

$$Q_\lambda(b_0, b_1, \dots, b_p) = \sum_{i=1}^n (Y_i - (b_0 + b_1 x_{i1} + \dots + b_p x_{ip}))^2 + \lambda \sum_{j=1}^p |b_j|,$$

where $\lambda > 0$ is a tuning parameter.

- ▶ The penalty term $\lambda \sum_{j=1}^p |b_j|$ can cause $\hat{\beta}_j^L = 0$ for some j .
- ▶ For λ large enough, all the $\hat{\beta}_j^L$ will be equal to zero.
- ▶ So LASSO performs variable selection and estimation simultaneously.
- ▶ Drawback: Hard to build CIs based on $\hat{\beta}_0^L, \hat{\beta}_1^L, \dots, \hat{\beta}_p^L$.

Effect of LASSO penalty on the objective function

```
# simulate some data with centered X and centered y (eliminates intercept)
n <- 500; p <- 2
X <- scale(matrix(rnorm(n*p),n,p)); b <- c(2,1/4); e <- rnorm(n)
y <- drop(X %*% b) + e - mean(e)

# define least squares and LASSO objective functions
Q <- function(b,X,y) mean((y - X %*% b)^2)
Qlambda <- function(b,X,y,lambda) Q(b,X,y) + lambda * sum(abs(b))

# set LASSO penalty parameter
lambda <- 1

# evaluate Q and Qlambda over a grid of b1 and b2 values
b1seq <- seq(b[1]-2,b[1]+2,length=200)
b2seq <- seq(b[2]-2,b[2]+2,length=200)
Q_vals <- Qlambda_vals <- matrix(0,length(b1seq),length(b2seq))
for(i in 1:length(b1seq))
  for(j in 1:length(b2seq)){

    Q_vals[i,j] <- Q(b=c(b1seq[i],b2seq[j]),X,y)
    Qlambda_vals[i,j] <- Qlambda(b=c(b1seq[i],b2seq[j]),X,y,lambda)

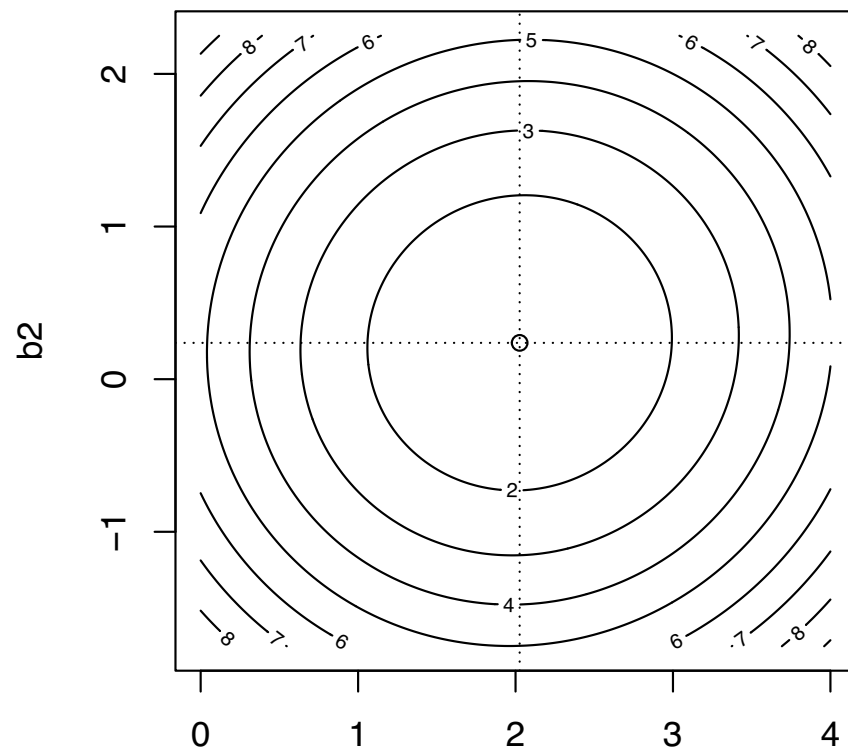
  }
}
```

```
# compute least squares and lasso estimator
bhat <- coef(lm(y~X-1))
bhat_lambda <- optim(par = c(0,0),fn = Qlambda,X = X, y = y,lambda = lambda)$par

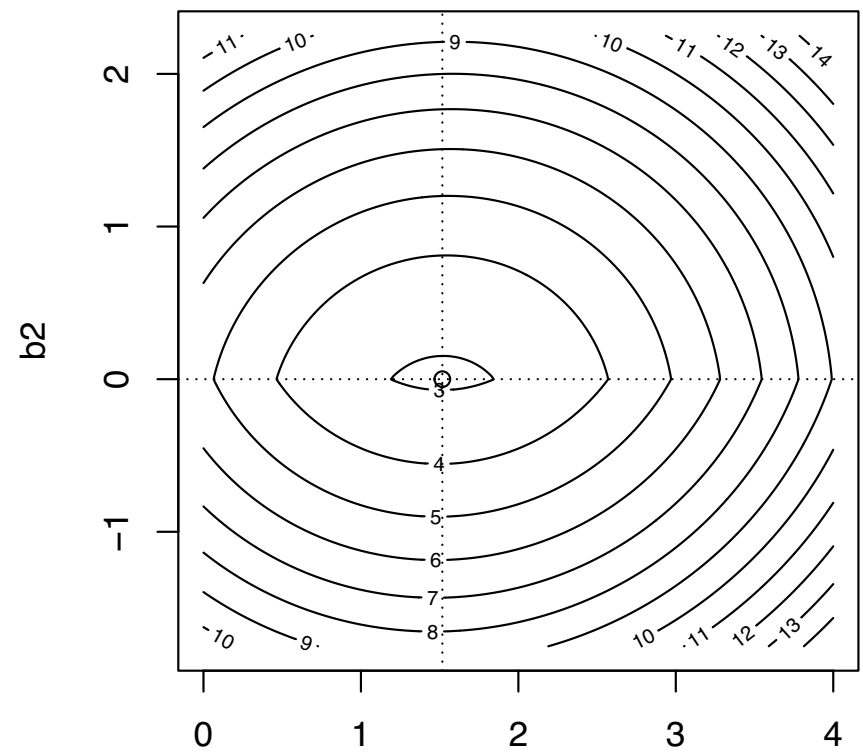
# make contour plots of least-squares and LASSO objective functions
par(mfrow=c(1,2))
contour(z = Q_vals, x = b1seq, y = b2seq, main = "lambda = 0",xlab = "b1", ylab = "b2")
points(x = bhat[1],y = bhat[2]);abline(v = bhat[1], lty = 3);abline(h = bhat[2], lty = 3)

contour(z = Qlambda_vals, x = b1seq, y = b2seq, main = paste( "lambda =",lambda), xlab = "b1", ylab = "b2")
points(x = bhat_lambda[1],y = bhat_lambda[2]); abline(v = bhat_lambda[1],lty = 3);abline(h = bhat_lambda[2],lty = 3)
```

lambda = 0



lambda = 1



LASSO on the FIFA data

Use `cv.ncvreg()` function from R package `ncvreg`.

Runs crossvalidation to choose the best value of λ .

```
library(ncvreg) # first time run install.packages("ncvreg")

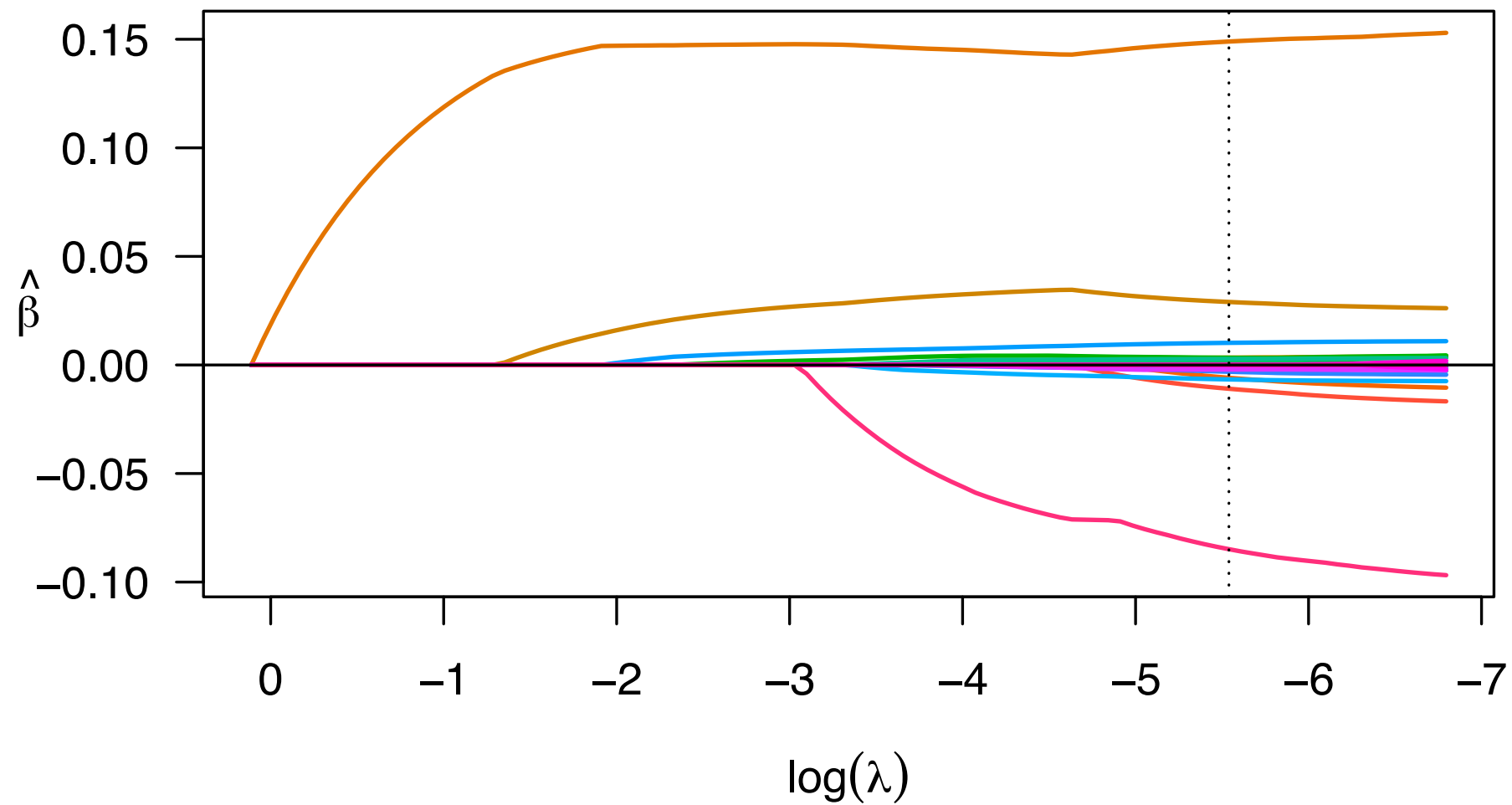
# prepare response vector and design matrix
y <- log(fifa$wage_eur)
X <- fifa[,-c(1,5)]
X$nationality <- ifelse(fifa$nationality_name == "usa",1,0)

# crossvalidation to choose lambda
lasso <- cv.ncvreg(X,y,penalty = "lasso")
```

```
lasso$fit$beta[,lasso$min] # estimates under the "best" lambda
```

(Intercept)	age
-2.7141745677	-0.0109595020
height_cm	weight_kg
-0.0059229907	0.0000000000
overall	potential
0.1489539112	0.0290051658
attacking_crossing	attacking_finishing
0.0033871138	0.0000000000
attacking_heading_accuracy	attacking_short_passing
0.0019742827	0.0000000000
attacking_volleys	skill_dribbling
0.0033051630	0.0005030114
skill_curve	skill_fk_accuracy
0.0008486285	0.0027692643
skill_long_passing	skill_ball_control
0.0000000000	0.0000000000
movement_acceleration	movement_sprint_speed
-0.0010281516	-0.0012251654
movement_agility	movement_reactions
-0.0066529040	0.0101870798
movement_balance	defending_standing_tackle
-0.0031334884	-0.0008652520
defending_sliding_tackle	goalkeeping_diving
-0.0020922056	0.0000000000
goalkeeping_handling	goalkeeping_kicking
0.0000000000	0.0000000000
goalkeeping_positioning	goalkeeping_reflexes
0.0000000000	0.0000000000
nationality	
-0.0848418250	

```
plot(lasso$fit, log.l = TRUE)  
abline(v = log(lasso$fit$lambda[lasso$min]), lty = 3)
```



The dangers of post-selection inference

It is dangerous to:

1. Ask the data what hypotheses to test (what model to build).
2. Use afterwards the same data to perform inference (get p values).

Illustration:

Add 50 spurious predictors to the SC apartment prices data.

See how many we find to be significant.

```
n <- nrow(scpts)
X <- matrix(rnorm(n*50),n,50)
colnames(X) <- paste("x",1:50,sep="")
scptsX <- cbind(scpts,X)

lmX_out <- lm(log(price) ~ ., data = scptsX)
```

Call:

lm(formula = log(price) ~ ., data = scaptsX)

Residuals:

	Min	1Q	Median	3Q	Max
	-0.40691	-0.10593	-0.00984	0.09176	0.59829

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	6.723e+00	5.726e-02	117.407	< 2e-16	***
nbath	-7.572e-02	4.374e-02	-1.731	0.085410	.
nbed	3.523e-02	4.373e-02	0.806	0.421727	
sqft	4.615e-04	9.509e-05	4.853	2.93e-06	***
cityColumbia	-2.700e-01	4.415e-02	-6.116	7.40e-09	***
cityGreenville	-1.455e-01	3.991e-02	-3.646	0.000363	***
cityRock Hill	-1.156e-01	5.213e-02	-2.218	0.028009	*
petsyes	-1.274e-04	3.296e-02	-0.004	0.996921	
x1	-4.256e-02	1.630e-02	-2.612	0.009881	**
x2	-6.735e-03	1.737e-02	-0.388	0.698724	
x3	-1.336e-02	1.673e-02	-0.798	0.426021	
x4	1.800e-02	1.633e-02	1.103	0.271885	
x5	3.440e-02	1.567e-02	2.194	0.029681	*
x6	-4.503e-02	1.547e-02	-2.912	0.004123	**
x7	-5.928e-03	1.656e-02	-0.358	0.720860	
x8	2.528e-03	1.683e-02	0.150	0.880780	
x9	-2.727e-02	1.556e-02	-1.752	0.081674	.
x10	8.986e-03	1.705e-02	0.527	0.599015	
x11	2.285e-02	1.512e-02	1.511	0.132906	
x12	1.499e-03	1.667e-02	0.090	0.928478	
x13	2.631e-03	1.525e-02	0.172	0.863300	
x14	1.517e-02	1.801e-02	0.842	0.400914	
x15	2.069e-02	1.586e-02	1.304	0.194158	
x16	1.618e-02	1.624e-02	0.996	0.320773	
x17	-2.213e-02	1.657e-02	-1.335	0.183679	
x18	7.988e-04	1.571e-02	0.051	0.959508	
x19	2.092e-02	1.749e-02	1.196	0.233336	
x20	-8.961e-03	1.562e-02	-0.574	0.567008	
x21	-2.818e-02	1.598e-02	-1.763	0.079846	.
x22	6.120e-03	1.683e-02	0.364	0.716675	
x23	9.889e-03	1.531e-02	0.646	0.519226	
x24	1.779e-03	1.567e-02	0.114	0.909774	

We reject $H_0: \beta_j = 0$ at $\alpha = 0.05$ for 4 of the spurious predictors.

So the Type I error rate was $4/50 = 0.08$.

Now do backwards stepwise selection to throw some variables away.

Then see how many of the spurious predictors we find “significant”.

```
stepX_out <- step(lmX_out, data = scaptsX, trace = 0)
summary(stepX_out)
```

Call:

```
lm(formula = log(price) ~ nbath + sqft + city + x1 + x5 + x6 +
    x9 + x11 + x17 + x21 + x27 + x29 + x31 + x33 + x34 + x38 +
    x45 + x48, data = scaptsX)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.43594	-0.10202	-0.01113	0.09349	0.69316

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	6.685e+00	4.659e-02	143.474	< 2e-16	***
nbath	-6.782e-02	3.487e-02	-1.945	0.05321	.
sqft	5.520e-04	5.574e-05	9.902	< 2e-16	***
cityColumbia	-2.781e-01	3.883e-02	-7.161	1.64e-11	***
cityGreenville	-1.554e-01	3.432e-02	-4.529	1.04e-05	***
cityRock Hill	-1.236e-01	4.442e-02	-2.782	0.00593	**
x1	-3.432e-02	1.407e-02	-2.440	0.01561	*
x5	3.751e-02	1.342e-02	2.795	0.00572	**
x6	-3.670e-02	1.353e-02	-2.713	0.00726	**
x9	-2.676e-02	1.322e-02	-2.024	0.04433	*
x11	2.907e-02	1.306e-02	2.226	0.02719	*
x17	-2.680e-02	1.411e-02	-1.899	0.05907	.
x21	-2.796e-02	1.384e-02	-2.020	0.04473	*
x27	-2.400e-02	1.390e-02	-1.726	0.08592	.
x29	2.656e-02	1.376e-02	1.929	0.05516	.
x31	-2.025e-02	1.361e-02	-1.488	0.13842	
x33	-2.042e-02	1.315e-02	-1.553	0.12197	
x34	2.317e-02	1.396e-02	1.660	0.09848	.
x38	-2.221e-02	1.384e-02	-1.604	0.11027	
x45	2.033e-02	1.369e-02	1.485	0.13910	
x48	-3.285e-02	1.421e-02	-2.312	0.02181	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.19 on 193 degrees of freedom

Multiple R-squared: 0.5429, Adjusted R-squared: 0.4955

F-statistic: 11.46 on 20 and 193 DF, p-value: < 2.2e-16

Backwards stepwise selection keeps 15 of the 50 spurious predictors.

Among these 15, we reject $H_0: \beta_j = 0$ at $\alpha = 0.05$ for 7 of them.

So the post-selection Type I error rate was $7/15 = 0.467$ 🤯.

WARNING: Selecting variables and then getting p-values in the selected model often leads to astonishingly inflated Type I error rates.

References

“Apartment for Rent Classified.” 2019. UCI Machine Learning Repository.

Fox, John, and Sanford Weisberg. 2019. *An R Companion to Applied Regression*. Third. Thousand Oaks CA: Sage.

<https://socialsciences.mcmaster.ca/jfox/Books/Companion/>.

Pedersen, Ulrik Thyge. 2022. “FIFA Players.” kaggle.

<https://www.kaggle.com/datasets/ulrikthygepedersen/fifa-players>.