Bruce Campell ST 503 HW 1

Problems 1, 4, 7 Chapter 2 Faraway, Julian J. Linear Models with R, Second Edition. CRC Press.

Bruce Campbell

03 September, 2017

Problem 2.1

The dataset teengamb concerns a study of teenage gambling in Britain. Fit a regression model with the expenditure on gambling as the response and the sex, status, income and verbal score as predictors. Present the output.

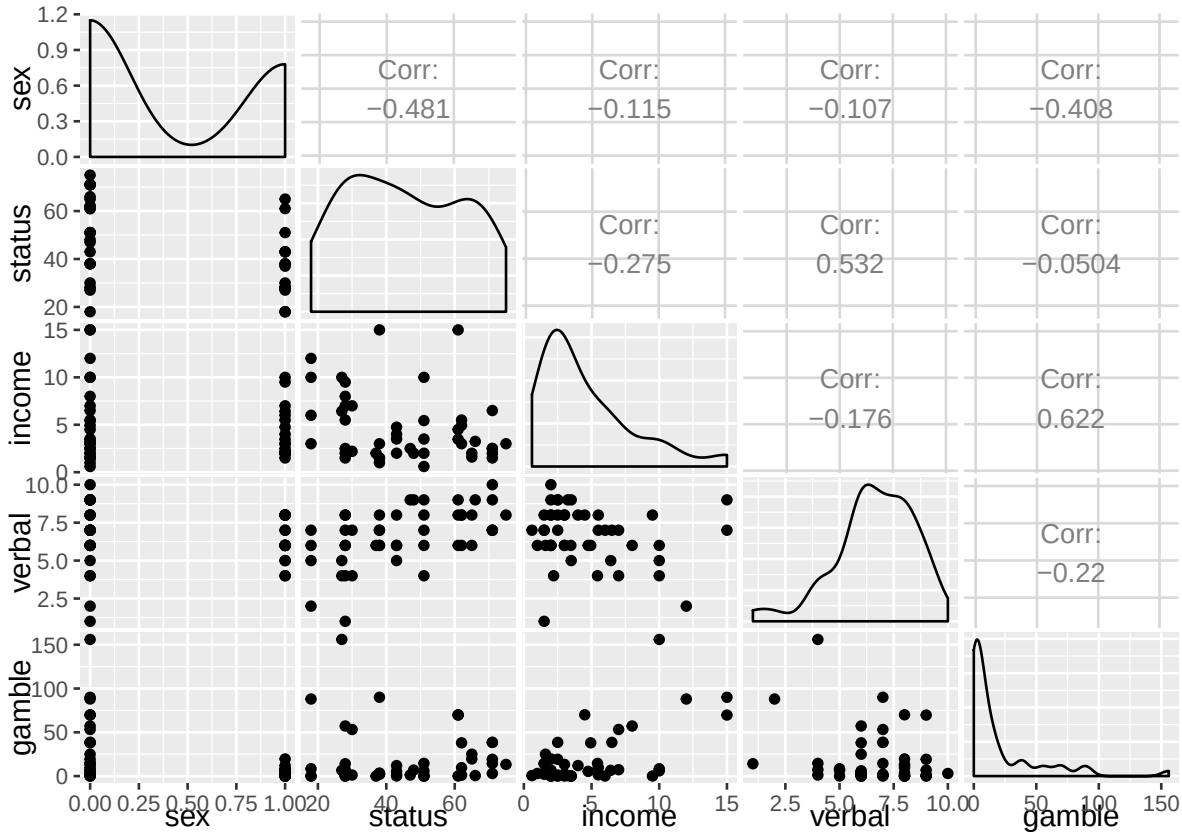
- (a) What percentage of variation in the response is explained by these predictors?
- (b) Which observation has the largest (positive) residual? Give the case number.
- (c) Compute the mean and median of the residuals.
- (d) Compute the correlation of the residuals with the fitted values.
- (e) Compute the correlation of the residuals with the income.
- (f) For all other predictors held constant, what would be the difference in predicted expenditure on gambling for a male compared to a female?

```
if (!require(faraway)) {
    install.packages("faraway")
    library(faraway)
}
```

```
library(pander)
library(ggplot2)
library(GGally)
data(teengamb, package = "faraway")
head(teengamb)
```

```
##   sex status income verbal gamble
## 1   1     51   2.00      8    0.0
## 2   1     28   2.50      8    0.0
## 3   1     37   2.00      6    0.0
## 4   1     28   7.00      4    7.3
## 5   1     65   2.00      8   19.6
## 6   1     61   3.47      6    0.1
```

```
ggpairs(teengamb)
```




```
lm.fit <- lm(gamble ~ sex + status + income + verbal, data = teengamb)

summary(lm.fit)
```

```
##
## Call:
## lm(formula = gamble ~ sex + status + income + verbal, data = teengamb)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.082 -11.320  -1.451   9.452  94.252
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.55565   17.19680   1.312   0.1968
## sex         -22.11833    8.21111  -2.694   0.0101 *
## status        0.05223    0.28111   0.186   0.8535
## income        4.96198    1.02539   4.839 1.79e-05 ***
## verbal       -2.95949    2.17215  -1.362   0.1803
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 22.69 on 42 degrees of freedom
## Multiple R-squared:  0.5267, Adjusted R-squared:  0.4816
## F-statistic: 11.69 on 4 and 42 DF,  p-value: 1.815e-06
```

```
# uncomment for diagnostic plots plot(lm.fit)
```

(a) What percentage of variation in the response is explained by these predictors?

Here we calculate the proportion of explained and unexplained variance in the response that is given by the predictors in the mode we fit.

```
var.explained.proportion <- summary(lm.fit)$r.squared
var.unexplaned.proportion <- 1 - summary(lm.fit)$r.squared

pander(data.frame(var.explained.proportion = var.explained.proportion), caption = "Explained")
```

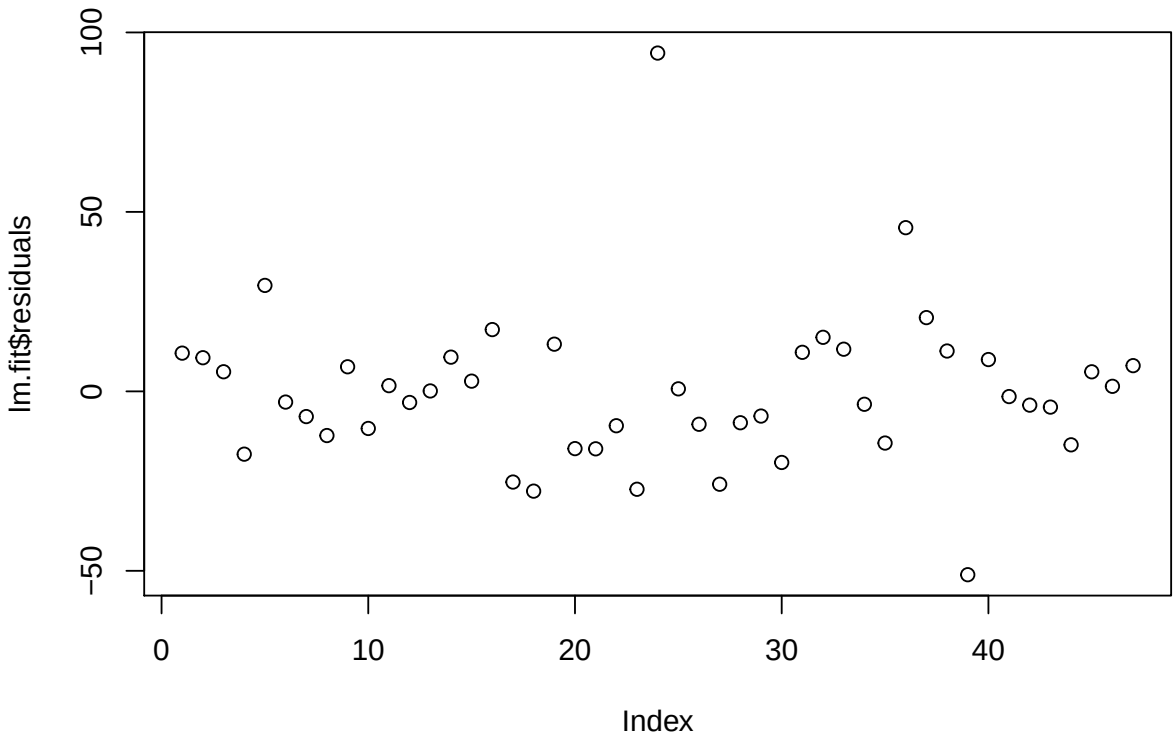
Table 1: Explained

var.explained.proportion
0.5267

(b) Which observation has the largest (positive) residual? Give the case number.

We're not sure if the question seeks the larges residual in absolute value or the largest of the positive residuals. We suspect that we're looking for the largest residual in absolute values since this may be an outlier that needs investigation, but we'll report both.

```
plot(lm.fit$residuals)
```




```
index.largest.pos.residual <- which.max(lm.fit$residuals)

index.largest.abs.residual <- which.max(abs(lm.fit$residuals))
```

The largest residual occurs at index 24 of the dataframe. This is the associated cases data.

```
pander(teengamb[24, ], caption = "Potential outlier.")
```

Table 2: Potential outlier.

	sex	status	income	verbal	gamble
24	0	27	10	4	156

(c) Compute the mean and median of the residuals.

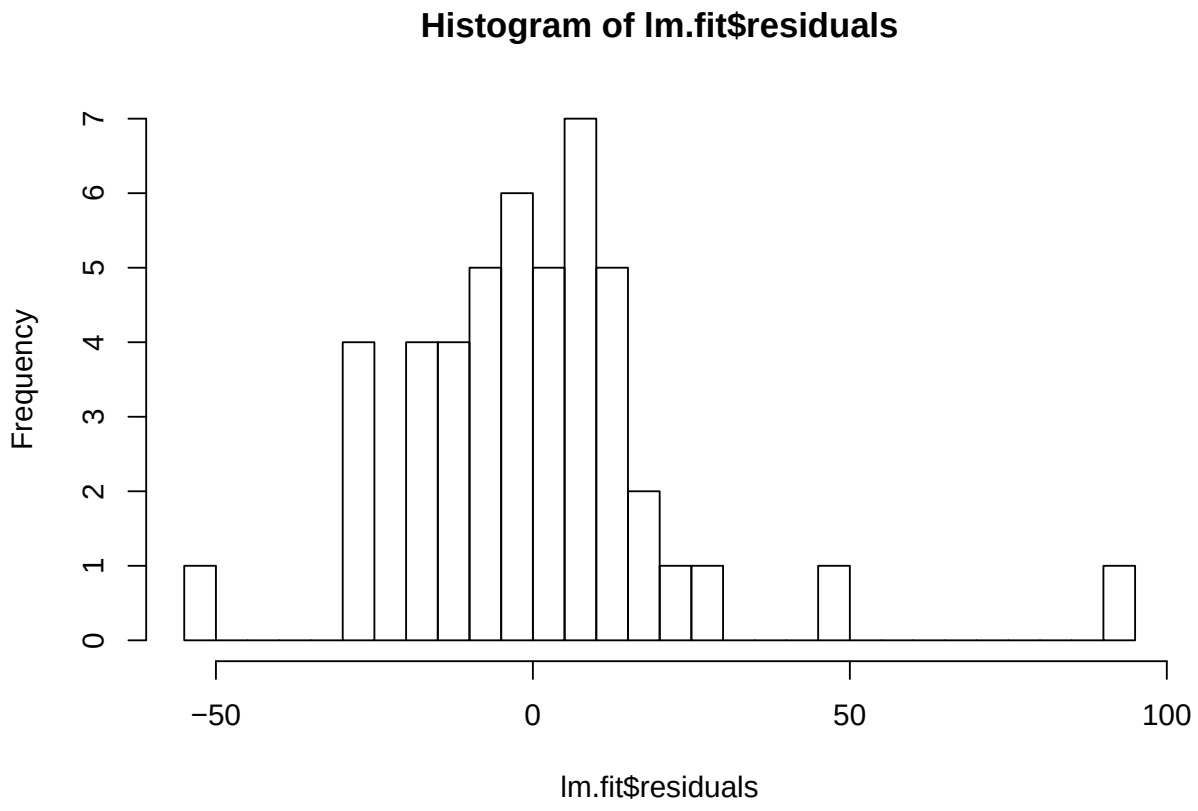
```
residuals.mean <- mean(lm.fit$residuals)

residuals.median <- median(lm.fit$residuals)

pander(data.frame(residuals.mean = residuals.mean, residuals.median = residuals.median))
```

residuals.mean	residuals.median
-3.065e-17	-1.451

```
hist(lm.fit$residuals, 30)
```



The mean residual is a very small number! We'd need to think through the implications of this - possibly it is an artifact of data that was generated.

(d) Compute the correlation of the residuals with the fitted values.

```
corr.residuals.vs.fitted <- cor(lm.fit$residuals, lm.fit$fitted.values)
pander(data.frame(corr.residuals.vs.fitted = corr.residuals.vs.fitted))
```

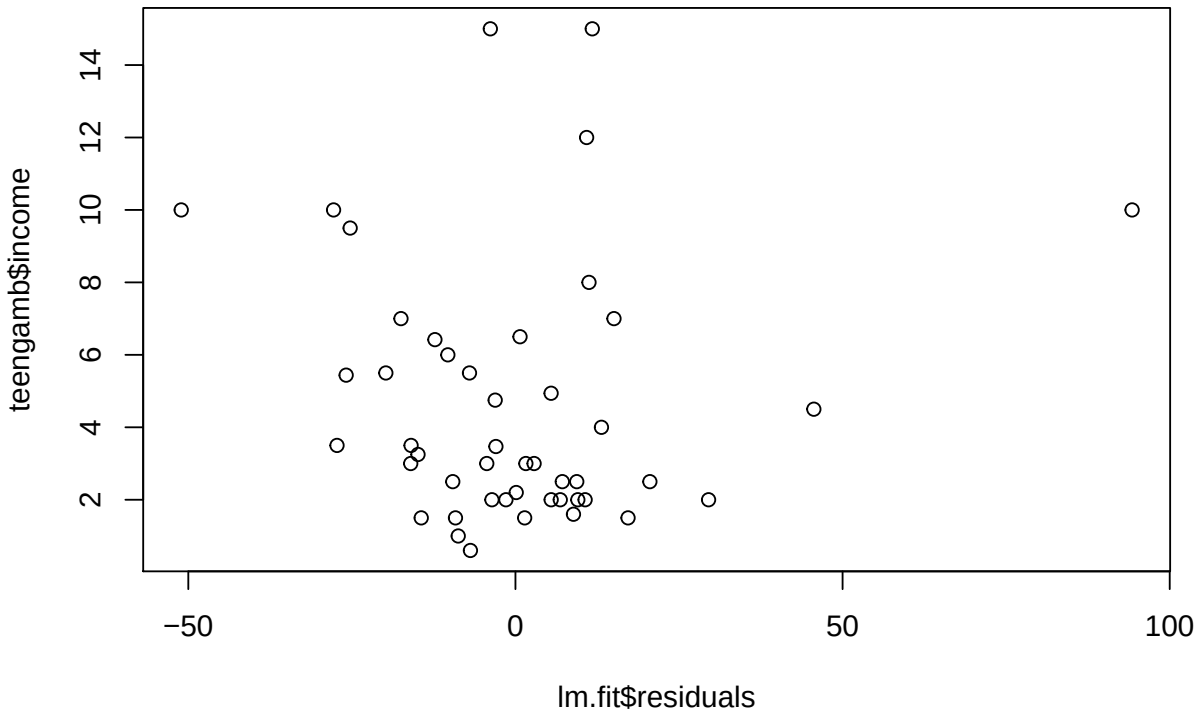
corr.residuals.vs.fitted
-1.071e-16

(e) Compute the correlation of the residuals with the income.

```
corr.residuals.income <- cor(lm.fit$residuals, teengamb$income)
pander(data.frame(corr.residuals.income = corr.residuals.income))
```

corr.residuals.income
-7.242e-17

```
plot(lm.fit$residuals, teengamb$income)
```



(f) For all other predictors held constant, what would be the difference in predicted expenditure on gambling for a male compared to a female?

This should be the value of the coefficient for gender. We need to be careful about the encoding and understanding whether this was treated as a factor in the regression. Querying the data `?teengamb` tells us that sex is encoded as so `0=male, 1=female`. Looking at the data frame teengamb we see that the class of the variable is integer and not a factor so we can now interpret the coefficient properly.

```r
gender.coefficient <- lm.fit$coefficients["sex"]

pander(data.frame(gender.coefficient = gender.coefficient))
```

	gender.coefficient
sex	-22.12

This value represents the change in the response when there is a unit change in the predictor. In this case since female is encoded as `1` we can say that females have that much less gamble response (less because the coefficient is negative).

We can apply the model by hand to a element of the data set to see this in practice.

```r
data.sample <- sample(nrow(teengamb), 1)
data.element <- teengamb[data.sample, ]
data.element$gamble <- NULL

data.element <- as.matrix(cbind(intercept = 1, data.element))
beta.hat <- as.matrix(lm.fit$coefficients)

pander(data.frame(data.element), caption = "Data sample")
```

Table 7: Data sample

	intercept	sex	status	income	verbal
42	1	0	61	15	9

```r
response.orig <- (data.element) %*% beta.hat

# change the gender of our data element
data.element[1, 2] <- ifelse(data.element[1, 2] == 1, 0, 1)


pander(data.frame(data.element), caption = "Data sample with gender modified")
```

Table 8: Data sample with gender modified

	intercept	sex	status	income	verbal
42	1	1	61	15	9

```r
response.gendermod <- (data.element) %*% beta.hat


pander(data.frame(response.difference = (response.orig - response.gendermod)))
```

	response.difference
42	22.12

Problem 2.4

The dataset prostate comes from a study on 97 men with prostate cancer who were due to receive a radical prostatectomy. Fit a model with lpsa as the response and lcavol as the predictor. Record the residual standard error and the R^2. Now add lweight, svi, lbph, age, lcp, pgg45 and gleason to the model one at a time. For each model record the residual standard error and the R^2. Plot the trends in these two statistics.

Load data and fit the models

Fit lpsa ~ lcavol +lweight

```
rm(list = ls())
# This is a library from the 'tidyverse' - we use it here to display the
# models neatly
library(broom)

data(prostate, package = "faraway")

# Make a data frame to hold the results

model.stats <- data.frame(num.predictors = integer(), r.squared = numeric(),
    residual.se = numeric(), model.string = character())

lm.fit <- lm(lpsa ~ lcavol, data = prostate)

# Dispaly both summaries for the first model
summary(lm.fit)
```

```
##
## Call:
## lm(formula = lpsa ~ lcavol, data = prostate)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.67625 -0.41648  0.09859  0.50709  1.89673
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.50730    0.12194   12.36   <2e-16 ***
## lcavol       0.71932    0.06819   10.55   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7875 on 95 degrees of freedom
## Multiple R-squared:  0.5394, Adjusted R-squared:  0.5346
## F-statistic: 111.3 on 1 and 95 DF,  p-value: < 2.2e-16
```

```
tidy(lm.fit)
```

```
##          term  estimate  std.error statistic      p.value
## 1 (Intercept) 1.5072979 0.12193682  12.36130 1.722234e-21
## 2      lcavol 0.7193201 0.06819288  10.54832 1.118616e-17
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol"

model.stats <- rbind(list(num.predictors = 1, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)

# This is annoying the step above to add the element to the data frame
# converts the model.string to a factor even though we've specified that
# it's character when we created the dataframe.
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight

```
lm.fit <- lm(lpsa ~ lcavol + lweight, data = prostate)

tidy(lm.fit)
```

```
##          term   estimate  std.error  statistic      p.value
## 1 (Intercept) -0.3026179 0.56904195 -0.5318024 5.961175e-01
## 2      lcavol  0.6775253 0.06626223 10.2249086 6.120248e-17
## 3     lweight  0.5109495 0.15725697  3.2491371 1.606370e-03
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight"

model.stats <- rbind(list(num.predictors = 2, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi, data = prostate)

tidy(lm.fit)
```

```
##          term   estimate  std.error  statistic      p.value
## 1 (Intercept) -0.2680926 0.54349952 -0.4932711 6.229839e-01
## 2      lcavol  0.5516380 0.07466789  7.3878878 6.303883e-11
## 3     lweight  0.5085413 0.15017008  3.3864358 1.039028e-03
## 4         svi  0.6661584 0.20977694  3.1755557 2.029012e-03
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi"

model.stats <- rbind(list(num.predictors = 3, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi + lbph

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi + lbph, data = prostate)

tidy(lm.fit)
```

```
##          term  estimate  std.error statistic      p.value
## 1 (Intercept) 0.1455407 0.59747312 0.2435938 8.080878e-01
## 2      lcavol 0.5496031 0.07405520 7.4215337 5.644522e-11
## 3     lweight 0.3908759 0.16600277 2.3546348 2.066733e-02
## 4         svi 0.7117370 0.20995670 3.3899226 1.031341e-03
## 5        lbph 0.0900933 0.05616596 1.6040553 1.121295e-01
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi + lbph"

model.stats <- rbind(list(num.predictors = 4, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi + lbph + age

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi + lbph + age, data = prostate)

tidy(lm.fit)
```

```
##          term    estimate  std.error statistic      p.value
## 1 (Intercept)  0.95099742 0.83174633  1.143374 2.558822e-01
## 2      lcavol  0.56560801 0.07458967  7.582926 2.772519e-11
## 3     lweight  0.42369200 0.16687265  2.539014 1.281437e-02
## 4         svi  0.72095499 0.20902367  3.449155 8.539487e-04
## 5        lbph  0.11183992 0.05805266  1.926525 5.715950e-02
## 6         age -0.01489225 0.01075481 -1.384706 1.695282e-01
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi + lbph + age"

model.stats <- rbind(list(num.predictors = 5, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi + lbph + age + lcp

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp, data = prostate)

tidy(lm.fit)
```

```
##          term    estimate  std.error  statistic      p.value
## 1 (Intercept)  0.93486843 0.83577360  1.1185666 2.663018e-01
## 2      lcavol  0.58764668 0.08662936  6.7834589 1.201212e-09
## 3     lweight  0.41808376 0.16792458  2.4897116 1.462162e-02
## 4         svi  0.78256448 0.24261409  3.2255525 1.753263e-03
## 5        lbph  0.11381222 0.05842139  1.9481258 5.451588e-02
## 6         age -0.01511242 0.01080779 -1.3982902 1.654622e-01
## 7         lcp -0.04118380 0.08135166 -0.5062441 6.139231e-01
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi + lbph + age + lcp"

model.stats <- rbind(list(num.predictors = 6, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
```

```
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45, data = prostate)

tidy(lm.fit)
```

```
##          term     estimate   std.error statistic      p.value
## 1 (Intercept)  0.953926041 0.829439251  1.150085 2.531896e-01
## 2      lcavol  0.591614545 0.086001482  6.879120 8.069406e-10
## 3     lweight  0.448292433 0.167770596  2.672056 8.965360e-03
## 4         svi  0.757733506 0.241281796  3.140450 2.289875e-03
## 5        lbph  0.107671072 0.058107614  1.852960 6.720199e-02
## 6         age -0.019336452 0.011065868 -1.747396 8.401791e-02
## 7         lcp -0.104482266 0.090477516 -1.154787 2.512688e-01
## 8       pgg45  0.005317704 0.003432566  1.549192 1.248843e-01
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45"

model.stats <- rbind(list(num.predictors = 7, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Fit lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45+ gleason

```
lm.fit <- lm(lpsa ~ lcavol + lweight + svi + lbph + age + lcp + pgg45 + gleason,
    data = prostate)

tidy(lm.fit)
```

```
##          term     estimate   std.error  statistic      p.value
## 1 (Intercept)  0.669336698 1.296387471  0.5163091 6.069335e-01
## 2      lcavol  0.587021826 0.087920303  6.6767493 2.110698e-09
## 3     lweight  0.454467424 0.170012435  2.6731423 8.955363e-03
## 4         svi  0.766157326 0.244309148  3.1360157 2.328749e-03
## 5        lbph  0.107054031 0.058449214  1.8315735 7.039846e-02
## 6         age -0.019637176 0.011172725 -1.7575995 8.229321e-02
## 7         lcp -0.105474263 0.091013487 -1.1588861 2.496377e-01
## 8       pgg45  0.004525231 0.004421179  1.0235350 3.088604e-01
## 9     gleason  0.045141598 0.157464523  0.2866779 7.750328e-01
```

```
model.summary <- summary(lm.fit)

r.squared <- model.summary$r.squared
residual.se <- model.summary$sigma
model.string <- "lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45+ gleason"

model.stats <- rbind(list(num.predictors = 8, r.squared = r.squared, residual.se = residual.se
    model.string = model.string), model.stats)
model.stats$model.string <- as.character(model.stats$model.string)
```

Present the model stats

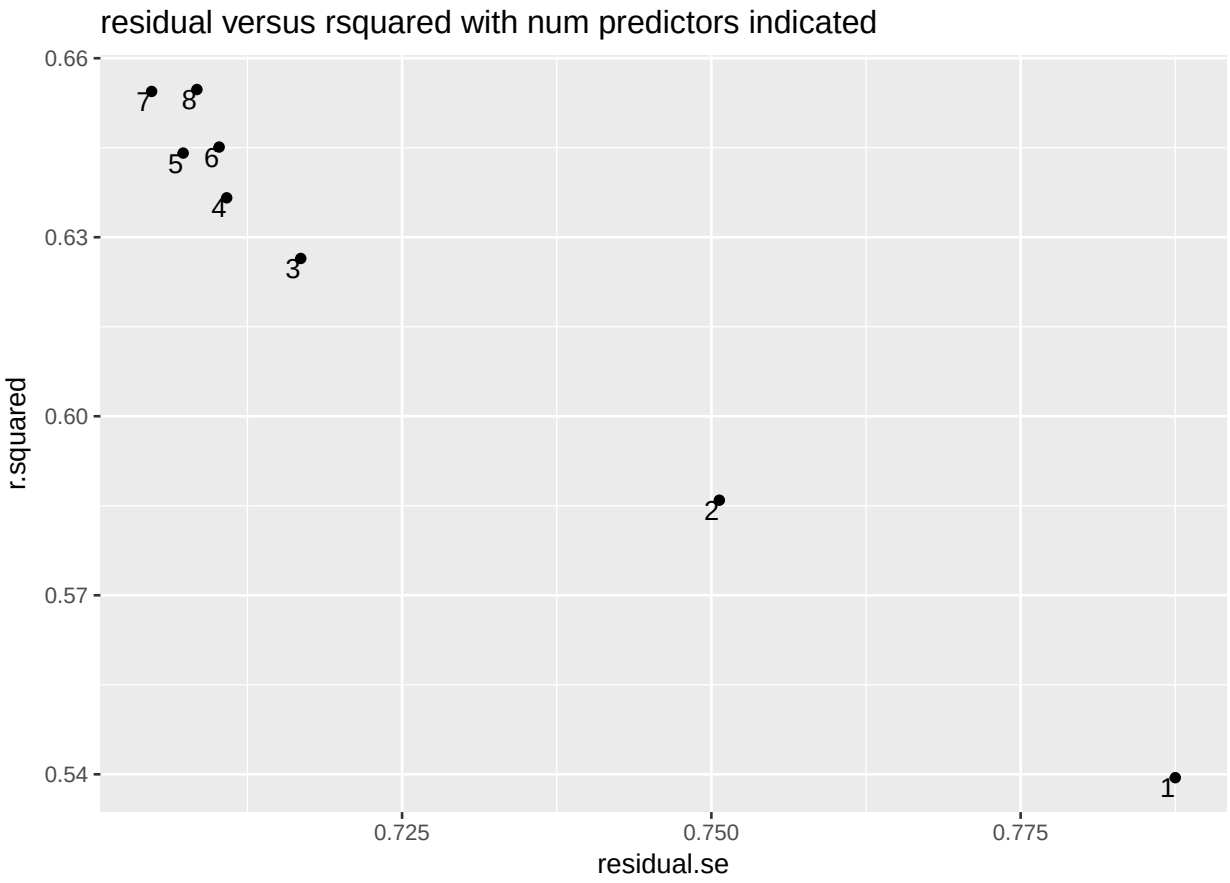
```
rownames(model.stats) <- NULL
pander(model.stats, caption = "model statistics")
```

Table 10: model statistics

num.predictors	r.squared	residual.se	model.string
8	0.6548	0.7084	lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45+ gleason
7	0.6544	0.7048	lpsa ~ lcavol +lweight + svi + lbph + age + lcp + pgg45
6	0.6451	0.7102	lpsa ~ lcavol +lweight + svi + lbph + age + lcp
5	0.6441	0.7073	lpsa ~ lcavol +lweight + svi + lbph + age
4	0.6366	0.7108	lpsa ~ lcavol +lweight + svi + lbph
3	0.6264	0.7168	lpsa ~ lcavol +lweight + svi
2	0.5859	0.7506	lpsa ~ lcavol +lweight
1	0.5394	0.7875	lpsa ~ lcavol

Plot SE versus R^2

```
p <- ggplot(model.stats, aes(x = residual.se, y = r.squared)) + geom_point()
p <- p + geom_text(aes(label = num.predictors, num.predictors = "red", hjust = 1,
    vjust = 1))
p + ggtitle("residual versus rsquared with num predictors indicated")
```



We see that generally the proportion of variance explained by the model increases and the residual standard error decreases as the dimension of the model increases. The effect becomes less pronounced as we get to 6+ predictors. One could argue that inclusion of gleason to the model does not add much explanatory power. This may make empirical sense since the gleason score is assigned by a pathologist based on a stained tissue slide. It could be the case that this feature summaries or is a weak proxy for the biochemical variables.

Problem 2.7

An experiment was conducted to determine the effect of four factors on the resistivity of a semiconductor wafer. The data is found in wafer where each of the four factors is coded as - or + depending on whether the low or the high setting for that factor was used.

Fit the linear model $resist\ x1 + x2 + x3 + x4$

```
rm(list = ls())
data(wafer, package = "faraway")

print("Inspect Data")
```

```
## [1] "Inspect Data"
```

```
head(wafer)
```

```
##   x1 x2 x3 x4 resist
## 1  -  -  -  -  193.4
## 2  +  -  -  -  247.6
## 3  -  +  -  -  168.2
## 4  +  +  -  -  205.0
## 5  -  -  +  -  303.4
## 6  +  -  +  -  339.9
```

```
print("check the class of the columns")
```

```
## [1] "check the class of the columns"
```

```
lapply(wafer, class)
```

```
## $x1
## [1] "factor"
##
## $x2
## [1] "factor"
##
## $x3
## [1] "factor"
##
## $x4
## [1] "factor"
##
## $resist
## [1] "numeric"
```

```
print("Fi the model")
```

```
## [1] "Fi the model"
```

```
lm.fit <- lm(resist ~ x1 + x2 + x3 + x4, data = wafer)
```

(a) Extract the X matrix using the model.matrix function. Examine this to determine how the low and high levels have been coded in the model.

```
model.matrix(lm.fit)
```

```
##    (Intercept) x1+ x2+ x3+ x4+
## 1            1   0   0   0   0
## 2            1   1   0   0   0
## 3            1   0   1   0   0
## 4            1   1   1   0   0
## 5            1   0   0   1   0
```

```
## 6            1   1   0   1   0
## 7            1   0   1   1   0
## 8            1   1   1   1   0
## 9            1   0   0   0   1
## 10           1   1   0   0   1
## 11           1   0   1   0   1
## 12           1   1   1   0   1
## 13           1   0   0   1   1
## 14           1   1   0   1   1
## 15           1   0   1   1   1
## 16           1   1   1   1   1
## attr(,"assign")
## [1] 0 1 2 3 4
## attr(,"contrasts")
## attr(,"contrasts")$x1
## [1] "contr.treatment"
##
## attr(,"contrasts")$x2
## [1] "contr.treatment"
##
## attr(,"contrasts")$x3
## [1] "contr.treatment"
##
## attr(,"contrasts")$x4
## [1] "contr.treatment"
```

Now let's look at the data matrix to see how the factors are coded

```
pander(wafer)
```

x1	x2	x3	x4	resist
-	-	-	-	193.4
+	-	-	-	247.6
-	+	-	-	168.2
+	+	-	-	205
-	-	+	-	303.4
+	-	+	-	339.9
-	+	+	-	226.3
+	+	+	-	208.3
-	-	-	+	220
+	-	-	+	256.4
-	+	-	+	165.7
+	+	-	+	203.5
-	-	+	+	285
+	-	+	+	268
-	+	+	+	169.1
+	+	+	+	208.5

Comparing the model matrix to the original dataframe we see that low level $- \longrightarrow 0$ and high level $+ \longrightarrow 1$

(b) Compute the correlation in the X matrix. Why are there some missing values in the matrix?

```
pander(data.frame(cor(model.matrix(lm.fit))), caption = "Correlation of X")
```

Table 12: Correlation of X

	X.Intercept.	x1.	x2.	x3.	x4.
(Intercept)	1	NA	NA	NA	NA
x1+	NA	1	0	0	0
x2+	NA	0	1	0	0
x3+	NA	0	0	1	0
x4+	NA	0	0	0	1

The correlation in the X matrix is the pairwise values of the column correlations. The correlation is the covariance divided by the sqaure root of the product of the two variances.

If X and Y are jointly distributed random variables and the variances and covariances of both X and Y exist and the variances are nonzero, then the correlation of X and Y , denoted by , is $\rho = Cov(X,Y)/\sqrt{Var(X)Var(Y)}$

In this case we're dealing with samples and calculating the sample correlations.

There are NaN's in the due to the intercept. The variance of this vector is 0 and when R attempts to divide by 0 in the calculation of $\rho_{i,j}$ the results is a NaN. Note that R sets the diagonal of the correlation to one - it does not calculate the value - that's why we see $\rho_{1,1} = 1$.

We noted that the $i =\neq j$ terms for $i, j > 1$ were zero - this cause concern and we wrote some test code to validate the entries. The values were verified to be correct.

```
MM <- model.matrix(lm.fit)
# colMeans <- colSums(MM)/nrow(MM) corr_ij <- function(i,j,MM){return(
# t(MM[,i] - mean(MM[,i])) %*% (MM[,j]-mean(MM[,j]))
# /sqrt(var(MM[,i]*var(MM[,j]))) ) } corr_ij(2,3,MM) corr_ij(3,4,MM)
# corr_ij(2,4,MM)
```

(d) Refit the model without x4 and examine the regression coefficients and standard errors? What stayed the same as the original fit and what changed?

```
lm.fit.reduced <- lm(resist ~ x1 + x2 + x3, data = wafer)

summary(lm.fit.reduced)
```

```
##
```

```
## Call:
## lm(formula = resist ~ x1 + x2 + x3, data = wafer)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.137 -20.550   3.575  18.462  41.013
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   229.54      13.32  17.231 7.88e-10 ***
## x1+            25.76      13.32   1.934 0.077047 .
## x2+           -69.89      13.32  -5.246 0.000206 ***
## x3+            43.59      13.32   3.272 0.006677 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.64 on 12 degrees of freedom
## Multiple R-squared:  0.7777, Adjusted R-squared:  0.7221
## F-statistic: 13.99 on 3 and 12 DF,  p-value: 0.0003187
```

```
print("Now print the summary for the full model")
```

```
## [1] "Now print the summary for the full model"
```

```
summary(lm.fit)
```

```
##
## Call:
## lm(formula = resist ~ x1 + x2 + x3 + x4, data = wafer)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.381 -17.119   4.825  16.644  33.769
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   236.78      14.77  16.032 5.65e-09 ***
## x1+            25.76      13.21   1.950 0.077085 .
## x2+           -69.89      13.21  -5.291 0.000256 ***
## x3+            43.59      13.21   3.300 0.007083 **
## x4+           -14.49      13.21  -1.097 0.296193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.42 on 11 degrees of freedom
## Multiple R-squared:  0.7996, Adjusted R-squared:  0.7267
## F-statistic: 10.97 on 4 and 11 DF,  p-value: 0.0007815
```

We note that the p value for X4 was not significant in the first model and that removing it resulted in a model where the explained variance is not significantly changes. We could do a LRT on the

two models to further understand if adding X4 enhances the model.

(e) Explain how the change in the regression coefficients is related to the correlation matrix of X.

This took me a while to sort out. When the model matrix is orthogonal the covariance matrix of the sampling distribution of the regression parameters will be diagonal - when the error are iid $N(0, \sigma)$.

$$\hat{\beta} \sim N(\beta, \sigma(\mathbf{X}^\intercal \mathbf{X})^{-1})$$

This means the regression parameters are independent. That's why we did not see a change in the estimates of the coefficients for $X1\,X2$, $X3$ when we removed $X4$ from the model.

We can verify

$$(\mathbf{X}^\intercal \mathbf{X})^{-1}(\mathbf{X}^\intercal \mathbf{X}) = I$$

for our model matrix

```
solve(t(MM) %*% MM) %*% t(MM) %*% MM
```

```
##             (Intercept) x1+ x2+ x3+ x4+
## (Intercept)           1   0   0   0   0
## x1+                   0   1   0   0   0
## x2+                   0   0   1   0   0
## x3+                   0   0   0   1   0
## x4+                   0   0   0   0   1
```