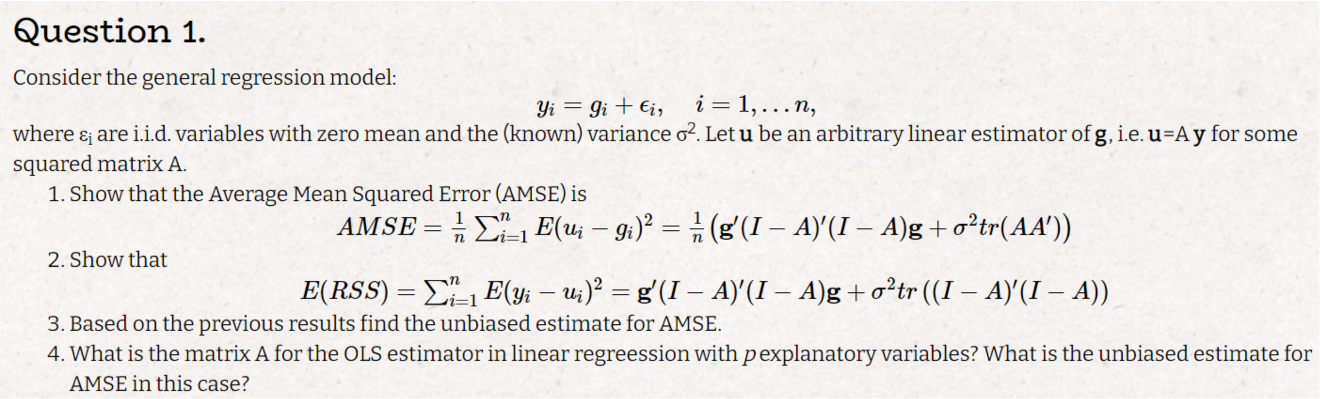
EX 3

roi hezkiyahu

23 3 2022

Q.1



Question 1.

Consider the general regression model:

$$y_i = g_i + \epsilon_i, \quad i = 1, \dots n,$$

where ε_i are i.i.d. variables with zero mean and the (known) variance σ^2. Let **u** be an arbitrary linear estimator of **g**, i.e. **u**=A **y** for some squared matrix A.

1. Show that the Average Mean Squared Error (AMSE) is

$$AMSE = \frac{1}{n}\sum_{i=1}^{n} E(u_i - g_i)^2 = \frac{1}{n}\left(\mathbf{g}'(I-A)'(I-A)\mathbf{g} + \sigma^2 tr(AA')\right)$$

2. Show that

$$E(RSS) = \sum_{i=1}^{n} E(y_i - u_i)^2 = \mathbf{g}'(I-A)'(I-A)\mathbf{g} + \sigma^2 tr\left((I-A)'(I-A)\right)$$

3. Based on the previous results find the unbiased estimate for AMSE.
4. What is the matrix A for the OLS estimator in linear regreession with p explanatory variables? What is the unbiased estimate for AMSE in this case?

a

$$\text{it is sufficent to show:}(1)\sum_{i=1}^{n} E(u_i - g_i)^2 = g^t(I-A)^t(I-A)g + \sigma^2 tr(AA^t)$$

$$(2)E(u_i - g_i)^2 = V(u_i - g_i) + [E(u_i - g_i)]^2$$

$$(3)V(u_i - g_i) = V(u_i) = V((Ay)_i) = V((A(g+\varepsilon))_i) = V((A\varepsilon)_i) = A_i^2\sigma^2$$

$$(4)[E(u_i - g_i)]^2 = [E(A(g+\varepsilon)_i - g_i)]^2 = ((Ag)_i - g_i)^2 = ((A-I)g)_i^2$$

$$\text{plug (3),(4) back to (1) and we get:} \sum_{i=1}^{n} E(u_i - g_i)^2 = g^t(I-A)^t(I-A)g + \sigma^2 tr(AA^t)$$

b

$$(5)\sum_{i=1}^{n} E(y_i - u_i)^2 = \sum_{i=1}^{n} V(y_i - u_i) + [E(y_i - u_i)]^2$$

$$(6)\ V(y_i - u_i) = V[((I-A)y)_i] = \sigma^2(I-A)_i^2$$

$$(7)\ [E(y_i - u_i)]^2 = [E((I-A)y)_i]^2 = [E((I-A)(g+\varepsilon))_i]^2 = [((I-A)g)_i]^2$$

$$\text{plug (7),(8) back to (5) and we get:} \sum_{i=1}^{n} E(y_i - u_i)^2 = g^t(I-A)^t(I-A)g + \sigma^2 tr((I-A)^t(I-A))$$

c

$$\text{our only paramater is } \sigma^2 \text{ so in order to get an unbaised estimator for AMSE we can use } \hat{\sigma}^2$$

$$\text{thus we get: } g^t(I-A)^t(I-A)g + \hat{\sigma}^2 tr(AA^t) \text{ is an unbaised estimator for AMSE}$$

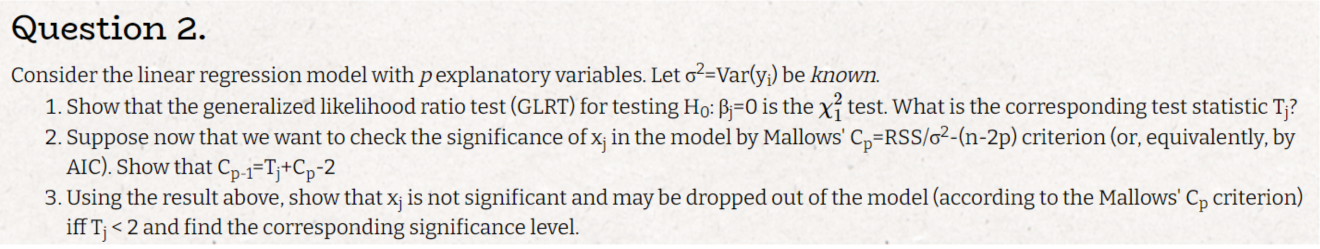
d

$$\text{the matrix A for the OLS estimator in linear regreession is H therefor the unbiased estimate for AMSE is:}$$

$$AM\hat{S}E = \frac{1}{n}[y(I-H)y + \hat{\sigma}^2 p] = \frac{1}{n}[X^t\hat{\beta}^t(I-H)X\hat{\beta} + (n-p)\hat{\sigma}^2 + p\hat{\sigma}^2] =$$

$$= \frac{1}{n}[X^t\hat{\beta}^t(I-H)X\hat{\beta} + n\hat{\sigma}^2]$$

Q.2



Question 2.

Consider the linear regression model with *p* explanatory variables. Let σ^2=Var(y_i) be *known*.

1. Show that the generalized likelihood ratio test (GLRT) for testing H_0: β_j=0 is the χ^2_1 test. What is the corresponding test statistic T_j?
2. Suppose now that we want to check the significance of x_j in the model by Mallows' C_p=RSS/σ^2-(n-2p) criterion (or, equivalently, by AIC). Show that C_{p-1}=T_j+C_p-2
3. Using the result above, show that x_j is not significant and may be dropped out of the model (according to the Mallows' C_p criterion) iff T_j < 2 and find the corresponding significance level.

a

$$(1)\ \lambda_{LR} = -2(l(\beta_0) - l(\beta_{MLE})) = 2(l(\beta_{MLE}) - l(\beta_0))$$

$$(2)\ ln(f_\beta(y)) = C_\sigma - \frac{1}{2\sigma^2}(y - X\beta)^t(y - X\beta) = C_\sigma - \frac{1}{2\sigma^2}RSS_\beta$$

$$\text{plug (2) into (1) using: } \hat{\beta}_0, \hat{\beta}_{MLE} = \frac{RSS_{\hat{\beta}_{MLE}} - RSS_{\hat{\beta}_0}}{\sigma^2} \sim \chi^2_1$$

$$\text{the corresponding test statistic } T_j \text{ is: } \sqrt{\frac{RSS_{\hat{\beta}_{MLE}} - RSS_{\hat{\beta}_0}}{\sigma^2}}$$

b

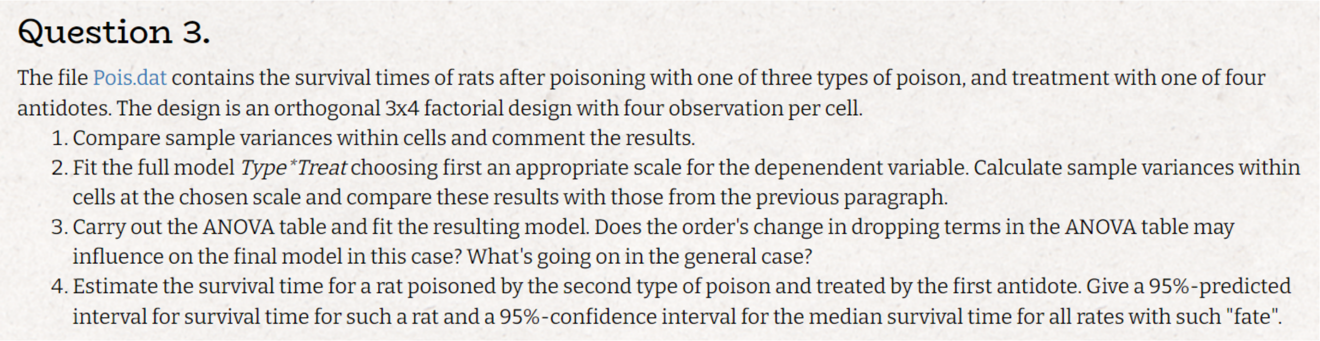
$$(3)\ C_{p-1} - C_P = \frac{RSS_{p-1} - RSS_p}{\sigma^2} - (n - 2p + 2) + n - 2p = \frac{RSS_{p-1} - RSS_p}{\sigma^2} - 2 = T_j^2 - 2$$

c

$$T_j^2 < 2 \iff C_{p-1} - C_P > 0 \iff x_j \text{ is not significant and may be dropped out of the model (according to the Mallows' Cp criteri}$$

$$\text{the corresponding significance level is } P(\chi^2_1 \geq 2) \approx 0.05$$

Q.3



Question 3.

The file Pois.dat contains the survival times of rats after poisoning with one of three types of poison, and treatment with one of four antidotes. The design is an orthogonal 3x4 factorial design with four observation per cell.

1. Compare sample variances within cells and comment the results.
2. Fit the full model *Type***Treat* choosing first an appropriate scale for the depenendent variable. Calculate sample variances within cells at the chosen scale and compare these results with those from the previous paragraph.
3. Carry out the ANOVA table and fit the resulting model. Does the order's change in dropping terms in the ANOVA table may influence on the final model in this case? What's going on in the general case?
4. Estimate the survival time for a rat poisoned by the second type of poison and treated by the first antidote. Give a 95%-predicted interval for survival time for such a rat and a 95%-confidence interval for the median survival time for all rates with such "fate".

a

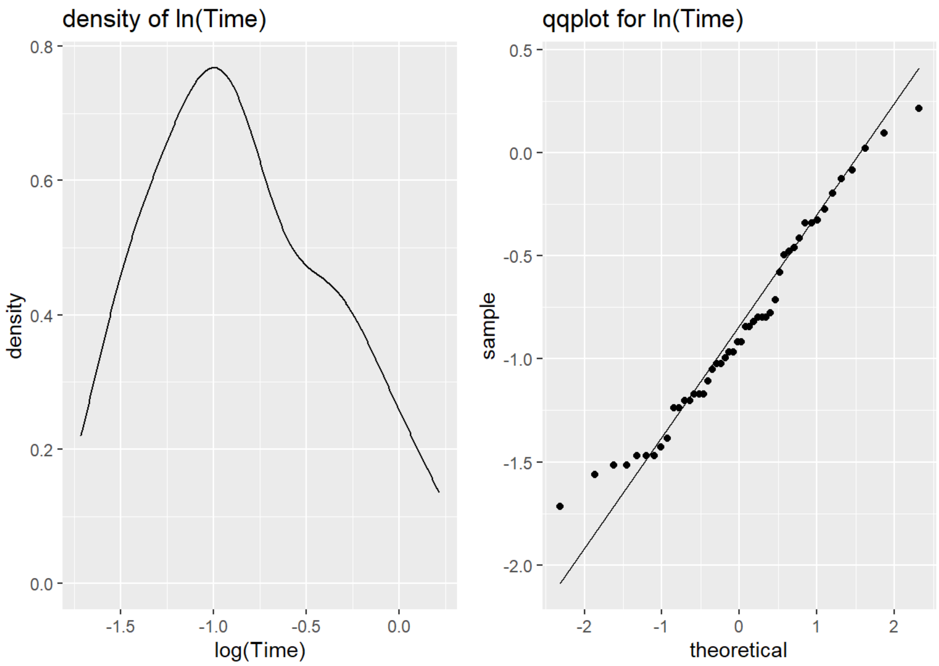
```
tbl <- as_tibble(read.table("Pois.dat",header = T))%>%
  mutate(across(c(Type,Treat),as.factor))
tbl %>%
  group_by(Type,Treat) %>%
  summarise(varinace_Time = var(Time)) %>%
  arrange(-varinace_Time)
```

```
## `summarise()` regrouping output by 'Type' (override with `.groups` argument)
## # A tibble: 12 x 3
## # Groups:   Type [3]
##    Type  Treat varinace_Time
##    <fct> <fct>         <dbl>
##  1 2     2          0.113
##  2 2     4          0.0734
##  3 1     2          0.0259
##  4 1     3          0.0246
##  5 1     4          0.0127
##  6 2     1          0.00567
##  7 1     1          0.00483
##  8 2     3          0.00323
##  9 3     2          0.00217
## 10 3     4          0.0007
## 11 3     1          0.000467
## 12 3     3          0.000167
```

we can see a large difference in variance between types and that for each type treat 2 has the largest variance

b

```
# chosing relevant transformation
g1 <- tbl %>%
  ggplot()+
  geom_density(aes(x = log(Time)))+
  ggtitle("density of ln(Time)")
g2 <- tbl %>%
  ggplot()+
  geom_qq(aes(sample = log(Time)))+
  geom_qq_line(aes(sample = log(Time)))+
  ggtitle("qqplot for ln(Time)")
g1 + g2
```



```
#model
logtbl <- tbl %>%
  mutate(across(Time,log)) %>%
  rename(log_Time = Time)

model <- lm(log_Time~ Type+Treat+Type*Treat,data = logtbl)

#in group var
logtbl %>%
  group_by(Type,Treat) %>%
  summarise(varinace_Time = var(log_Time)) %>%
  arrange(-varinace_Time)
```

```
## `summarise()` regrouping output by 'Type' (override with `.groups` argument)
## # A tibble: 12 x 3
## # Groups:   Type [3]
##    Type  Treat varinace_Time
##    <fct> <fct>         <dbl>
##  1 2     2           0.172
##  2 2     4           0.172
##  3 1     3           0.0746
##  4 2     1           0.0602
##  5 1     4           0.0404
##  6 1     1           0.0341
##  7 1     2           0.0315
##  8 2     3           0.0235
##  9 3     2           0.0195
## 10 3     1           0.0114
## 11 3     4           0.00644
## 12 3     3           0.00303
```

variance does not seem to have changed much

c

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: log_Time
##            Df Sum Sq Mean Sq F value    Pr(>F)
## Type        2 5.2375 2.61874 48.4324 6.195e-11 ***
## Treat       3 3.5572 1.18572 21.9295 2.987e-08 ***
## Type:Treat  6 0.3957 0.06596  1.2199    0.3189
## Residuals  36 1.9465 0.05407
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
res_model <-  lm(log_Time~ Type+Treat,data = logtbl)
anova(res_model,model)
```

```
## Analysis of Variance Table
##
## Model 1: log_Time ~ Type + Treat
## Model 2: log_Time ~ Type + Treat + Type * Treat
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     42 2.3423
## 2     36 1.9465  6   0.39575 1.2199 0.3189
```

the resulting model indicates that the interaction does not contribute in predicting Time, in general each Treatment and each type have their own influence on survival time

d

```
pred_tibl <- tibble(Type = 2, Treat = 1)%>% mutate(across(c(Type,Treat),factor))
pred_ci <- exp(predict.lm(res_model,pred_tibl,interval = "prediction"))
med_ci <- exp(predict.lm(res_model,pred_tibl,interval = "confidence"))
glue("the estimated survival time is: {round(pred_ci[1],3)}\n",
     "the prediction interval is: ({round(pred_ci[2],3)},{round(pred_ci[3],3)})\n",
     "the interval for median survival time is: ({round(med_ci[2],3)},{round(med_ci[3],3)})")
```

```
## the estimated survival time is: 0.342
## the prediction interval is: (0.206,0.566)
## the interval for median survival time is: (0.289,0.404)
```

Q.4

Question 4.

The prostate cancer data in the file Prostate.dat come from a study that examined the correlation between the level of prostate specific antigen (PSA) and the following clinical measurements in 97 men who were about to receive a radical prostatectomy:

lcavol	-	log(cancer volume)
lweight	-	log(prostate weight)
age	-	age
lbph	-	log(benign prostatic hyperplasia amount)
svi	-	seminal vesicle invasion
lcp	-	log(capsular penetration)
gleason	-	Gleason score
pgg45	-	percentage Gleason scores 4 or 5

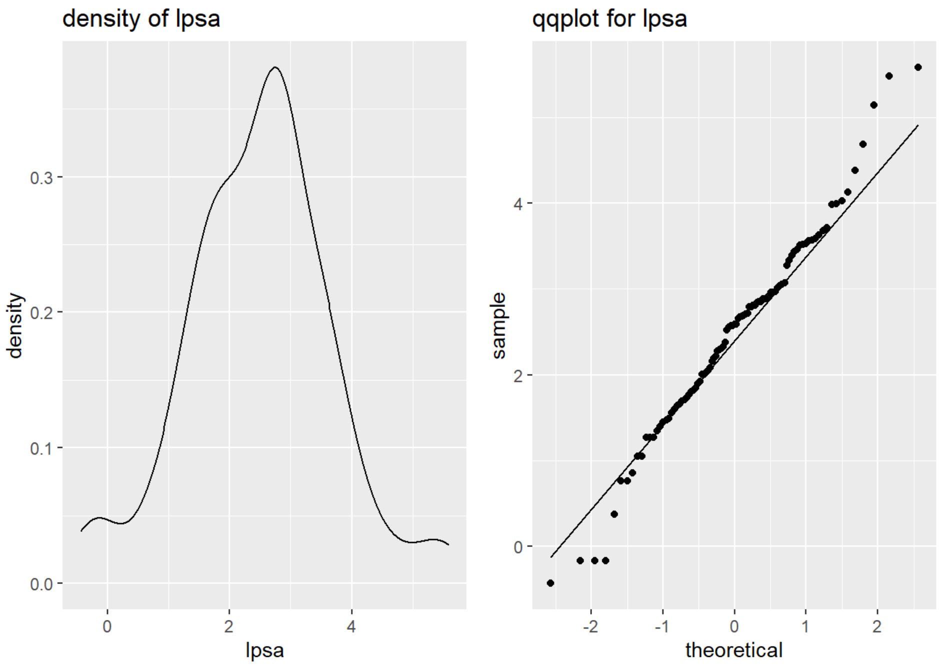
The goal is to predict the log of PCA (*lpsa*) from these measurements.

1. Analyse the data to get some first impression. Make some preliminary comments.
2. Check the presence of multicollinearity among the explanatory variables. What methodological and computational problems it might cause?
3. Split *randomly* (why?) the data into a training and test sets of 75 and 22 patiens respectively. Put a test set meanwhile aside and consider a training set:
    1. Start from the main effects model, verify its adequacy.
    2. Select the `best' model by adding/removing variables and their interactions w.r.t. several model selection criteria. Compare the resulting models (also with the main effects), comment the results.
4. Apply LASSO to the data and comment the results.
5. Try to reduce deminsionality by principle component regression and partial least squares. Comment the results. Are there any "conceptual" problems in using these methods for this data?
6. How would you test and compare the goodness-of-fit of different models from the previous paragraph on the test set? Apply your ideas and comment the results.
7. Split *randomly* again the initial data into training and test sets of the same sizes and repeat the steps 3-5. Compare and comment the results for two different splits. Are they surprising? (explain, answers like "Nothing can surprise me in this world anymore" won't be accepted!)
8. Make final conclusions and point out on the measurements relevant for predicting the prostate specific antigen.

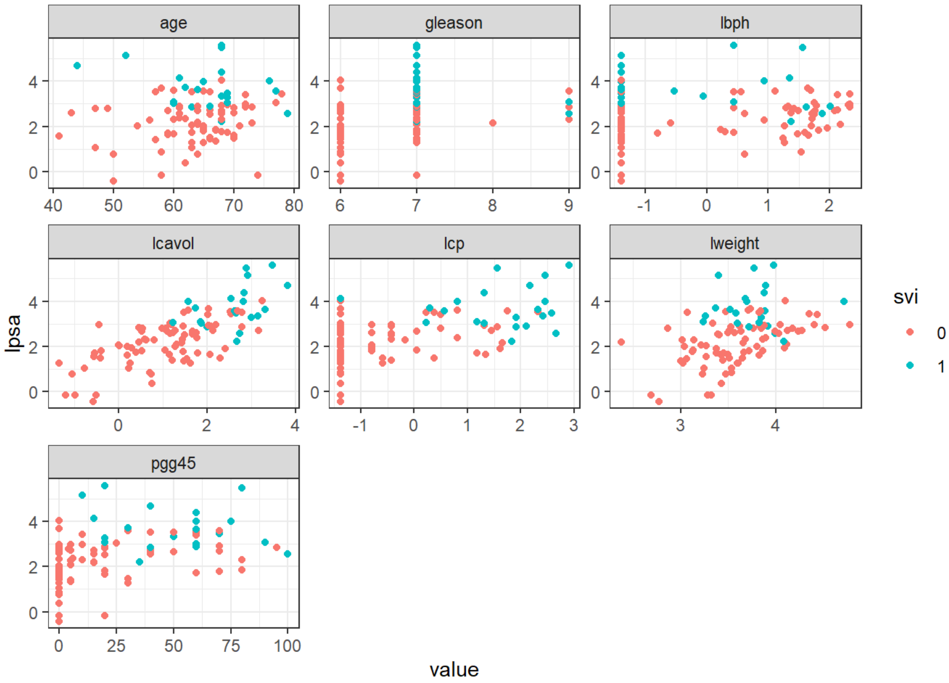
a

```
Prostate <- as_tibble(read.table("Prostate.dat",header = T)) %>% mutate(across(svi,factor))
g3 <- Prostate %>%
  ggplot()+
  geom_density(aes(x = lpsa))+
  ggtitle("density of lpsa")

g4 <- Prostate %>%
  ggplot()+
  geom_qq(aes(sample = lpsa))+
  geom_qq_line(aes(sample = lpsa))+
  ggtitle("qqplot for lpsa")
g3 + g4
```



```
Prostate %>%
  gather(-lpsa,-svi,key = "var", value = "value") %>%
  ggplot(aes(x = value, y = lpsa, color = svi)) +
    geom_point() +
    facet_wrap(~ var, scales = "free") +
    theme_bw()
```



we can see that lpsa is rather distributed normally

we can also see that svi would probably have a large influnce on the model given that svi =0 and svi =1 have different relationship with lpsa

lpsa~lcavol looks rather linear

b

```
Prostate %>%
  select(-lpsa)%>%
  mutate(across(svi,as.numeric))%>%
  cor()
```

```
##            lcavol   lweight       age         lbph         svi          lcp
## lcavol  1.0000000 0.28052138 0.2249999  0.027349703  0.53884500  0.675310484
## lweight 0.2805214 1.00000000 0.3479691  0.442264399  0.15538490  0.164537142
## age     0.2249999 0.34796911 1.0000000  0.350185896  0.11765804  0.127667752
## lbph    0.0273497 0.44226440 0.3501859  1.000000000 -0.08584324 -0.006999431
## svi     0.5388450 0.15538490 0.1176580 -0.085843238  1.00000000  0.673111185
## lcp     0.6753105 0.16453714 0.1276678 -0.006999431  0.67311118  1.000000000
## gleason 0.4324171 0.05688209 0.2688916  0.077820447  0.32041222  0.514830063
## pgg45   0.4336522 0.10735379 0.2761124  0.078460018  0.45764762  0.631528246
##            gleason      pgg45
## lcavol  0.43241706 0.43365225
## lweight 0.05688209 0.10735379
## age     0.26889160 0.27611245
## lbph    0.07782045 0.07846002
## svi     0.32041222 0.45764762
## lcp     0.51483006 0.63152825
## gleason 1.00000000 0.75190451
## pgg45   0.75190451 1.00000000
```

in multicollinearity the coefficient estimates of the multiple regression may change erratically in response to small changes in the model or the data, if we have a full multicollinearity X is not of full rank and this $X^T X$ cannot be inverted

the highest correlation is between pgg45 and gleason with r = 0.75, which is high but seems ok

c

```
set.seed(5)
split_obj <- Prostate%>%
  initial_split(prop = 74.5/97)
Prostate_tr <- training(split_obj)
Prostate_te <- testing(split_obj)
model_main <- lm(lpsa~.,data = Prostate_tr)
summary(model)
```

```
##
## Call:
## lm(formula = log_Time ~ Type + Treat + Type * Treat, data = logtbl)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.50006 -0.11846  0.01995  0.12202  0.48733
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.89755    0.11626  -7.720 3.82e-09 ***
## Type2        -0.26383    0.16442  -1.605 0.117330
## Type3        -0.66727    0.16442  -4.058 0.000254 ***
## Treat2        0.75768    0.16442   4.608 4.94e-05 ***
## Treat3        0.30281    0.16442   1.842 0.073777 .
## Treat4        0.38891    0.16442   2.365 0.023523 *
## Type2:Treat2  0.13472    0.23253   0.579 0.565960
## Type3:Treat2 -0.29379    0.23253  -1.263 0.214553
## Type2:Treat3 -0.13101    0.23253  -0.563 0.576659
## Type3:Treat3 -0.18730    0.23253  -0.805 0.425823
## Type2:Treat4  0.30494    0.23253   1.311 0.198023
## Type3:Treat4  0.04953    0.23253   0.213 0.832508
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2325 on 36 degrees of freedom
## Multiple R-squared:  0.8252, Adjusted R-squared:  0.7718
## F-statistic: 15.45 on 11 and 36 DF,  p-value: 1.643e-10
```

```
model_with_inter<- lm(lpsa~.+svi*lcavol+svi*lweight+svi * age + svi*lbph + svi*lcp+svi*gleason+svi*pgg45,data = Prostate_tr)
AIC <- stats::step(model_with_inter,direction  = "both",trace = 0)
BIC <- stats::step(model_with_inter,direction  = "both",trace = 0, k = log(75))
RIC <- stats::step(model_with_inter,direction  = "both",trace = 0, k = 2*log(15))
summary(AIC)
```

```
##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + pgg45 + lcavol:svi +
##     lweight:svi, data = Prostate_tr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.52634 -0.32833 -0.03406  0.40643  1.62179
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.099159   0.767200  -2.736  0.00792 **
## lcavol        0.457985   0.083917   5.458 7.33e-07 ***
## lweight       0.999057   0.212658   4.698 1.32e-05 ***
## svi1          6.713122   2.284193   2.939  0.00449 **
## pgg45         0.006341   0.003625   1.749  0.08476 .
## lcavol:svi1   0.452290   0.308567   1.466  0.14732
## lweight:svi1 -1.860230   0.613995  -3.030  0.00346 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6676 on 68 degrees of freedom
## Multiple R-squared:  0.7314, Adjusted R-squared:  0.7077
## F-statistic: 30.86 on 6 and 68 DF,  p-value: < 2.2e-16
```

```
summary(BIC)
```

```
##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + lweight:svi, data = Prostate_tr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65100 -0.38770  0.01339  0.39394  1.72681
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.95812    0.77656  -2.522  0.01396 *
## lcavol        0.53693    0.07613   7.053 1.00e-09 ***
## lweight       0.96378    0.21550   4.472 2.92e-05 ***
## svi1          6.81579    2.30363   2.959  0.00421 **
## lweight:svi1 -1.55658    0.60292  -2.582  0.01193 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6787 on 70 degrees of freedom
## Multiple R-squared:  0.7142, Adjusted R-squared:  0.6979
## F-statistic: 43.74 on 4 and 70 DF,  p-value: < 2.2e-16
```

```
summary(RIC)
```

```
##
## Call:
## lm(formula = lpsa ~ lcavol + lweight + svi + lweight:svi, data = Prostate_tr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65100 -0.38770  0.01339  0.39394  1.72681
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.95812    0.77656  -2.522  0.01396 *
## lcavol        0.53693    0.07613   7.053 1.00e-09 ***
## lweight       0.96378    0.21550   4.472 2.92e-05 ***
## svi1          6.81579    2.30363   2.959  0.00421 **
## lweight:svi1 -1.55658    0.60292  -2.582  0.01193 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6787 on 70 degrees of freedom
## Multiple R-squared:  0.7142, Adjusted R-squared:  0.6979
## F-statistic: 43.74 on 4 and 70 DF,  p-value: < 2.2e-16
```

```
anova(BIC,AIC)
```

```
## Analysis of Variance Table
##
## Model 1: lpsa ~ lcavol + lweight + svi + lweight:svi
## Model 2: lpsa ~ lcavol + lweight + svi + pgg45 + lcavol:svi + lweight:svi
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     70 32.247
## 2     68 30.310  2    1.9376 2.1735 0.1216
```

we can see that both RIC and BIC lead to the same model

from the anova result we can conclude that AIC model is no better then BIC model

d

```
lambda_cv <- cv.glmnet(model.matrix(model_with_inter),Prostate_tr$lpsa,alpha=1)$lambda.min
lasso_model <- glmnet(model.matrix(model_with_inter),Prostate_tr$lpsa,alpha=1,lambda = lambda_cv)
print(round(predict(lasso_model,s=lambda_cv,type="coefficients"),5))
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                     1
## (Intercept)  -0.44515
## (Intercept)   .
## lcavol        0.47619
## lweight       0.56060
## age           .
## lbph          0.04227
## svi1          0.12316
## lcp           .
## gleason       .
## pgg45         0.00275
## lcavol:svi1   0.24712
## lweight:svi1  .
## age:svi1      .
## lbph:svi1     .
## svi1:lcp      .
## svi1:gleason  .
## svi1:pgg45    .
```

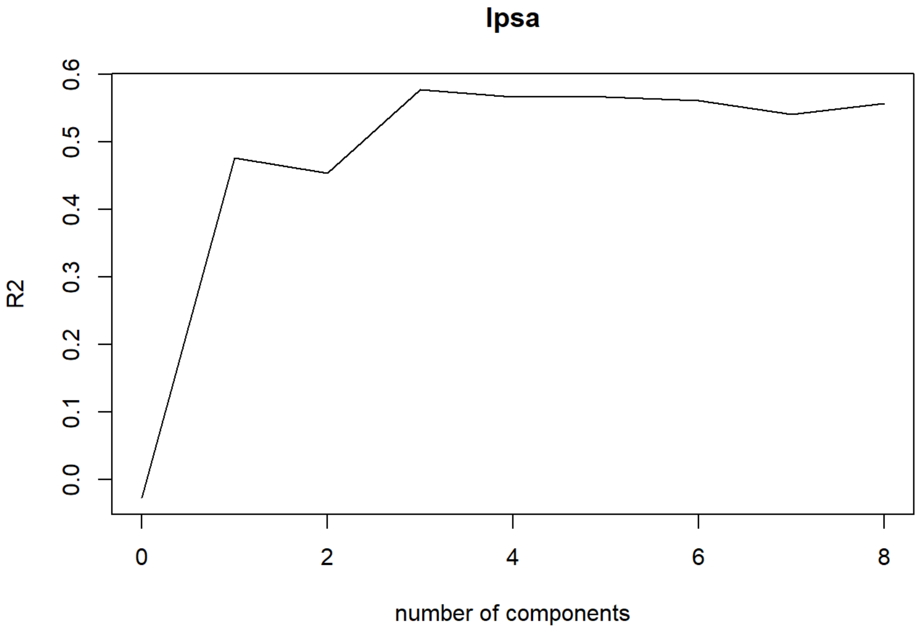
we can see that the lasso model is different from the AIC,BIC models, but the lcavol,lweight and svi are still importat, interactions have changed and we added lbph

e

```
#pcr
pcr_model <- pcr(lpsa~.,data = Prostate_tr,scale=T,validation="CV")
summary(pcr_model)
```

```
## Data:    X dimension: 75 8
##  Y dimension: 75 1
## Fit method: svdpc
## Number of components considered: 8
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.243   0.8877   0.9067   0.7979   0.8072   0.8076   0.8125
## adjCV        1.243   0.8839   0.9023   0.7934   0.8034   0.8040   0.8074
##        7 comps  8 comps
## CV      0.8316   0.8170
## adjCV   0.8255   0.8085
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       41.88    63.64    75.14    82.41    88.79    93.90    97.22   100.00
## lpsa    54.05    54.34    63.82    64.38    64.83    67.29    68.23    70.45
```

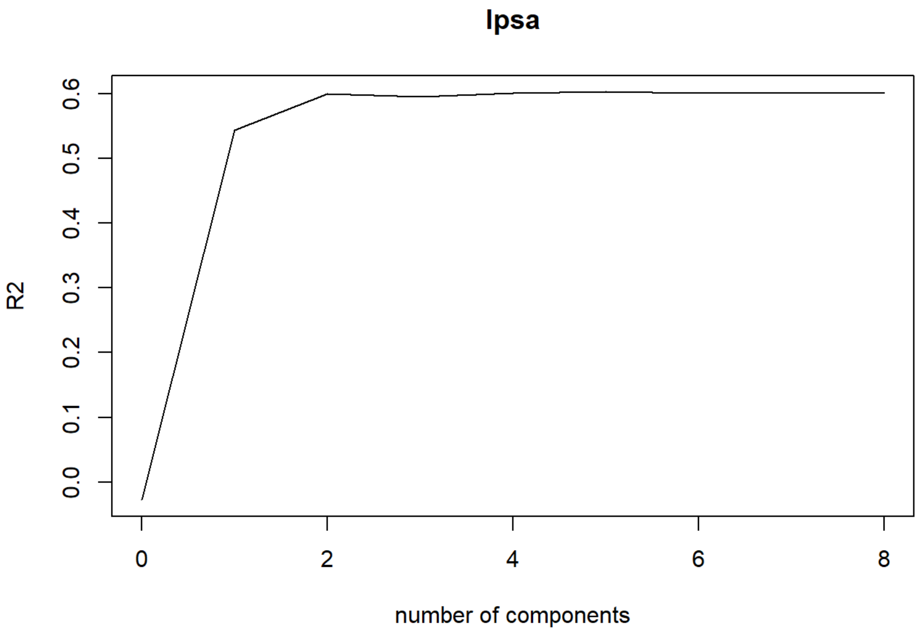
```
validationplot(pcr_model, val.type = "R2")
```



```
#pls
pls_model <- plsr(lpsa~.,data = Prostate_tr,scale=T,validation="CV")
summary(pls_model)
```

```
## Data:    X dimension: 75 8
##  Y dimension: 75 1
## Fit method: kernelpls
## Number of components considered: 8
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.243   0.8293   0.7766   0.7803   0.7754   0.7734   0.7748
## adjCV        1.243   0.8267   0.7723   0.7751   0.7697   0.7679   0.7692
##        7 comps  8 comps
## CV      0.7752   0.7752
## adjCV   0.7695   0.7696
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       41.41    52.94    65.75    79.20    84.26    90.14    94.59   100.00
## lpsa    60.18    68.36    69.74    70.31    70.44    70.45    70.45    70.45
```

```
validationplot(pls_model, val.type = "R2")
```



a problem that might arise is that some of the features are correlated and multicollinearity might impar these models

we can see that for the pcr model 3 components are sufficent, adding more components does not incrase model preformance very much

for ths pls model 3 components reach the best R^2

f

we can use the test set mse in order to compare the models

```
models <- list(model_main,
model_with_inter,
AIC,
BIC,
RIC,
lasso_model,
pls_model,
pcr_model)
model_names <- c("model_main",
"model_with_inter",
"AIC",
"BIC",
"RIC",
"lasso_model",
"pls_model",
"pcr_model")
mse = c()
for (i in 1:length(models)){
  if (i <= 5){y_pred <- predict(models[[i]],Prostate_te)}
  if (i==6) {y_pred <- predict(models[[i]],model.matrix(lpsa~.+svi*lcavol+svi*lweight+svi * age + svi*lbph + svi*lcp+svi*gle
ason+svi*pgg45,Prostate_te))}
  if (i >6){y_pred <- predict(models[[i]],Prostate_te,ncomp = 3)}
  mse[i] <- mean((y_pred - Prostate_te$lpsa)^2)
}
tibble(model = model_names,"mse" = mse) %>% arrange(mse)
```

```
## # A tibble: 8 x 2
##   model              mse
##   <chr>            <dbl>
## 1 lasso_model      0.498
## 2 pls_model        0.563
## 3 model_main       0.578
## 4 BIC              0.715
## 5 RIC              0.715
## 6 pcr_model        0.716
## 7 model_with_inter 0.784
## 8 AIC              0.815
```

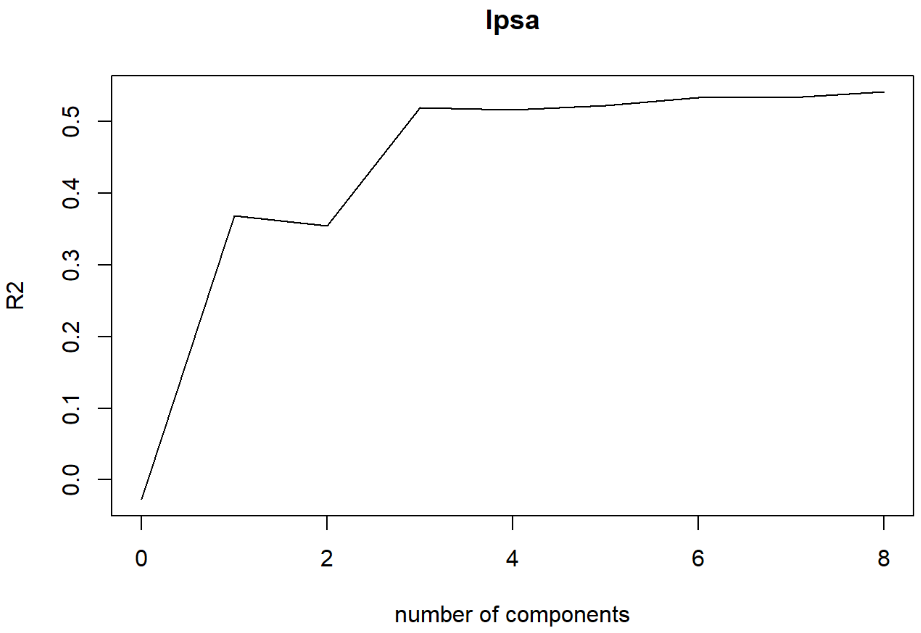
we can see that our best choise would be the lasso model

g

```
set.seed(100)
split_obj <- Prostate%>%
  initial_split(prop = 74.5/97)
Prostate_tr <- training(split_obj)
Prostate_te <- testing(split_obj)
model_main <- lm(lpsa~.,data = Prostate_tr)
model_with_inter<- lm(lpsa~.+svi*lcavol+svi*lweight+svi * age + svi*lbph + svi*lcp+svi*gleason+svi*pgg45,data = Prostate_tr)
AIC <- stats::step(model_with_inter,direction  = "both",trace = 0)
BIC <- stats::step(model_with_inter,direction  = "both",trace = 0, k = log(75))
RIC <- stats::step(model_with_inter,direction  = "both",trace = 0, k = 2*log(15))
lambda_cv <- cv.glmnet(model.matrix(model_with_inter),Prostate_tr$lpsa,alpha=1)$lambda.min
lasso_model <- glmnet(model.matrix(model_with_inter),Prostate_tr$lpsa,alpha=1,lambda = lambda_cv)
#pcr
pcr_model <- pcr(lpsa~.,data = Prostate_tr,scale=T,validation="CV")
summary(pcr_model)
```

```
## Data:    X dimension: 75 8
##  Y dimension: 75 1
## Fit method: svdpc
## Number of components considered: 8
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.129   0.8849   0.8944   0.7717   0.7744   0.7694   0.7599
## adjCV        1.129   0.8831   0.8923   0.7684   0.7715   0.7661   0.7558
##        7 comps  8 comps
## CV      0.7600   0.7539
## adjCV   0.7567   0.7488
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       43.03    61.70    75.23    83.71    89.57    94.84    97.72   100.00
## lpsa    42.13    42.14    57.25    58.35    60.94    62.28    62.50    65.19
```

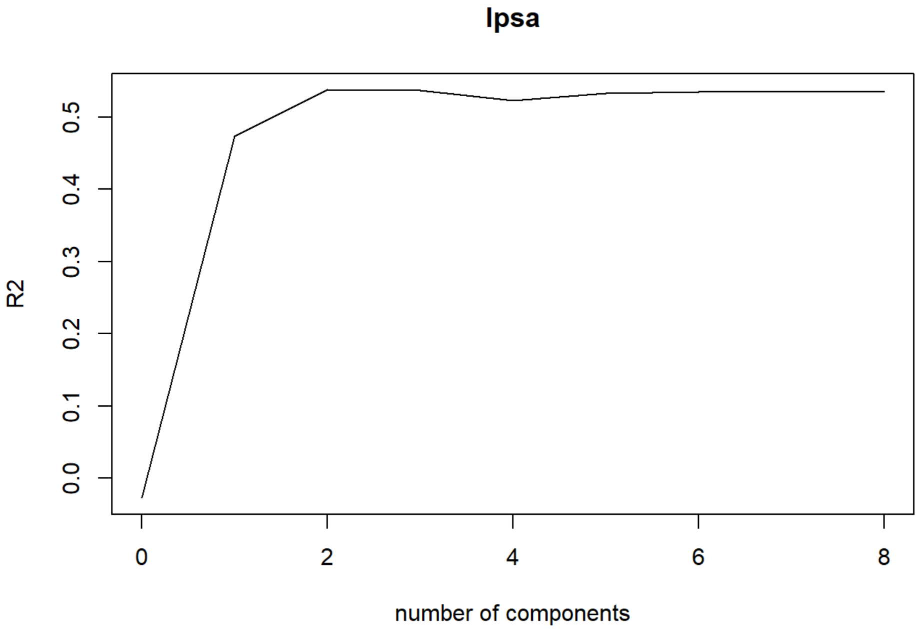
```
validationplot(pcr_model, val.type = "R2")
```



```
#pls
pls_model <- plsr(lpsa~.,data = Prostate_tr,scale=T,validation="CV")
summary(pls_model)
```

```
## Data:    X dimension: 75 8 
##  Y dimension: 75 1
## Fit method: kernelpls
## Number of components considered: 8
## 
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.129   0.8083   0.7574   0.7577   0.7693   0.7609   0.7597
## adjCV        1.129   0.8067   0.7535   0.7519   0.7623   0.7554   0.7543
##        7 comps  8 comps
## CV      0.7595   0.7596
## adjCV   0.7541   0.7542
## 
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       41.92    55.76    61.99    69.13    83.19    91.59    96.30   100.00
## lpsa    52.18    62.50    64.62    65.06    65.16    65.19    65.19    65.19
```

```
validationplot(pls_model, val.type = "R2")
```



```
mse_2 = c()
for (i in 1:length(models)){
  if (i <= 5){y_pred <- predict(models[[i]],Prostate_te)}
  if (i==6) {y_pred <- predict(models[[i]],model.matrix(lpsa~.+svi*lcavol+svi*lweight+svi * age + svi*lbph + svi*lcp+svi*gleason+svi*pgg45,Prostate_te))}
  if (i >6){y_pred <- predict(models[[i]],Prostate_te,ncomp = 3)}
  mse_2[i] <- mean((y_pred - Prostate_te$lpsa)^2)
}
tibble(model = model_names,"mse 1st split" = mse,"mse 2nd split" = mse_2) %>% arrange(mse)
```

```
## # A tibble: 8 x 3
##   model            `mse 1st split` `mse 2nd split`
##   <chr>                      <dbl>           <dbl>
## 1 lasso_model                0.498           0.488
## 2 pls_model                  0.563           0.456
## 3 model_main                 0.578           0.415
## 4 BIC                        0.715           0.548
## 5 RIC                        0.715           0.548
## 6 pcr_model                  0.716           0.533
## 7 model_with_inter           0.784           0.492
## 8 AIC                        0.815           0.559
```

We can see a difference in all of the models and even in best model (mse wise)

this is not surprising as Nothing can surprise me in this world anymore!

but seriously it is not surprising because different training set result in different models, and different testing sets results in different mse assessment, the test mse is an estimation to the expected error, but this estimation has a variance and it is not a constant

h

our 2 main relevant measurements are lcavol and lweight, as they are a part of every model and has a rather large coefficients. also svi looks important and its interaction with these 2 predictors