Vaja 6: Izbira modela

`R` paketi, ki jih bomo uporabili na vajah:

```
library(vtable) # summary table
library(kableExtra) # creates nice latex tables
library(corrplot) # correlation plot
library(car) # regression
library(reshape2) # reshape data sets for ggplot (melt)
library(ggplot2) # nice plots (ggplot)
library(knitr)  # for markdown
library(leaps) # best subset
library(glmnet) # lasso
library(mgcv) # gam
#library(summarytools) # summary table
```

Kako pristopati k modeliranju je v prvi meri odvisno od namena modeliranja. Kadar želimo na primer zgraditi nek pojasnjevalni model, to je, kadar želimo kvantificirati vzročni vpliv nekega proučevanega dejavnika na odzivno spremenljivko, si bomo prizadevali predvsem, da bo dobljena ocena danega vpliva nepristranska. Za to je pomembno dobro razumevanje vzročne povezanosti med setom neodvisnih spremenljivk ter odzivno spremenljivko, torej dobro predhodno poznavanje problema. Za gradnjo takega modela bo potrebno na primer poznati vse moteče (*confounding*) spremenljivke, ki imajo vpliv tako na proučevani dejavnik kot tudi na odzivno spremenljivko.

Kadar je cilj modeliranja napovedovanje, pa nas zanima predvsem, kako natančno lahko nek model napove vrednost odzivne spremenljivke za neke nove enote. Ovrednotenje kakovosti modela je izjemno pomemben korak modeliranja, saj nas vodi pri izbiri učne metode. Cilj tovrstnega modeliranja je najti kompromis med pristranskostjo in varianco modela (*bias-variance trade off*): želimo torej najti model, ki minimira pričakovano napako napovedi. Izkaže se, da ocena napake napovedi, ki jo dobimo na učnih podatkih, na katerih smo naredili model, ne da dobre ocene pričakovane napake napovedi. S kompleksnostjo modela se ocena na učnih podatkih lahko prilega zelo zapletenim zvezam med odzivno in napovednimi spremenljivkami. Posledično se pristranskost napovedi zmanjšuje, vendar se povečuje varianca. Učna napaka se tako s kompleksnostjo modela dosledno zmanjšuje in običajno pade na nič, če kompleksnost modela dovolj povečamo. Vendar pa se model, kjer je napaka napovedi na učnih podatkih enaka nič, preprilega in bo navadno dal slabe napovedi za nove enote.

V praksi tako modeliranje za namen napovedi poteka v dveh korakih:

- izbira modela: različne modele primerjamo med sabo, da izberemo končni model;
- ovrednotenje modela: ko smo izbrali končni model, ocenimo napako napovedi na testnih podatkih.

Če imamo na voljo veliko podatkov, lahko podatke delimo po principu slučajnosti na tri dele: učni niz, validacijski niz in testni niz. Učni niz se uporablja za gradnjo modela, validacijski niz za oceno napake napovedi pri izbiri modela, testni niz pa za oceno kakovosti (napake napovedi) končnega izbranega modela. Pomembno je, da do ovrednotenja modela testni niz ostane skrit in nedotaknjen. Če namesto tega testni niz uporabljamo večkrat in izberemo model z najmanjšo napako napovedi na testnem nizu, bo ocenjena testna napaka končnega izbranega modela podcenila pravo testno napako.

Slika 1: Razdelitev podatkov na tri dele, ko imamo na voljo zadosti podatkov.

Kadar podatkov ni dovolj, da bi jih razdelili na 3 dele, lahko korak validacije aproksimiramo bodisi analitično bodisi z metodami ponovnega vzorčenja. Na današnji vaji bomo obravnavali nekaj takih metod.

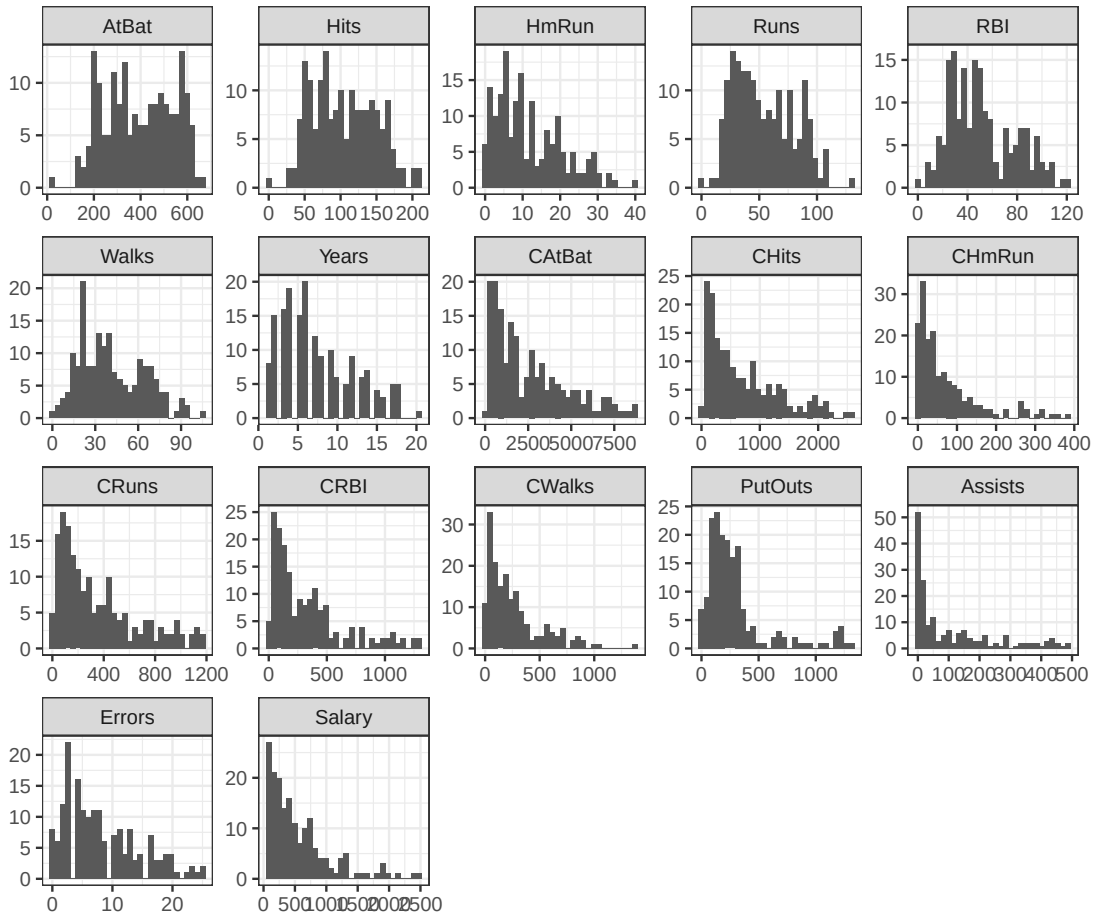
V podatkovnem okviru `bejzbol_train` so podatki o 175 igralcev bejzbola v prvi ligi, ki vključujejo 20 spremenljivk. Cilj modeliranja bo zgraditi napovedni model, ki bo kar se da natančno napovedoval plače (`Salary`) igralcev bejzbola v novi sezoni (leto 1987).

```
'data.frame':   175 obs. of  20 variables:
 $ AtBat    : int  288 196 216 151 528 213 573 492 491 327 ...
 $ Hits     : int  76 43 56 41 122 61 144 136 141 85 ...
 $ HmRun    : int  7 7 4 4 1 4 9 5 11 3 ...
 $ Runs     : int  34 29 22 26 67 17 85 76 77 30 ...
 $ RBI      : int  37 27 18 21 45 22 60 50 47 44 ...
 $ Walks    : int  15 30 15 19 51 3 78 94 37 20 ...
 $ Years    : int  4 13 12 2 4 17 8 12 15 8 ...
 $ CAtBat   : int  1644 3231 2796 288 1716 4061 3198 5511 4291 2140 ...
 $ CHits    : int  408 825 665 68 403 1145 857 1511 1240 568 ...
 $ CHmRun   : int  16 36 43 9 12 83 97 39 84 16 ...
 $ CRuns    : int  198 376 266 45 211 488 470 897 615 216 ...
 $ CRBI     : int  120 290 304 39 146 491 420 451 430 208 ...
 $ CWalks   : int  113 238 198 35 155 244 332 875 340 93 ...
 $ League   : chr  "N" "N" "A" "A" ...
 $ Division : chr  "W" "E" "E" "W" ...
 $ PutOuts  : int  203 80 391 28 209 178 1314 313 239 91 ...
 $ Assists  : int  3 45 44 56 372 45 131 381 8 185 ...
 $ Errors   : int  3 8 4 2 17 4 12 20 2 12 ...
 $ Salary   : num  240 240 250 95 350 ...
 $ NewLeague: chr  "N" "N" "A" "A" ...
```

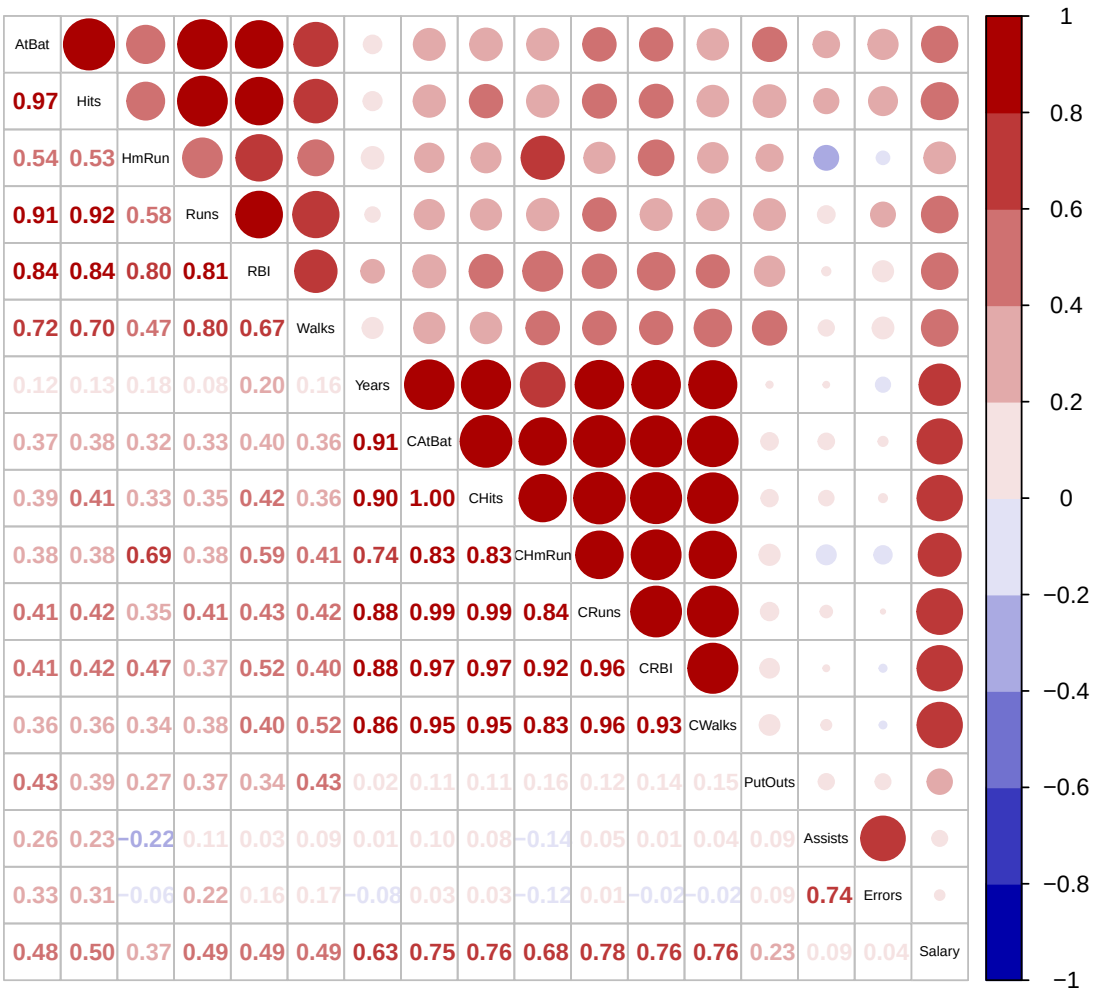
Poglejmo si najprej univariatne porazdelitve spremenljivk v podatkih ter korelacije med spremenljivkami.

Table 1: Opisne statistike za spremenljivke v podatkovnem okviru `bejzbol_train`.

Variable	N	Mean	Std. Dev.	Min	Pctl. 25	Pctl. 50	Pctl. 75	Max
AtBat	175	395	146	20	278	403	522	663
Hits	175	105	44	1	70	102	140	210
HmRun	175	12	9	0	5	9	18	40
Runs	175	54	26	0	32	50	76	130
RBI	175	51	26	0	30	46	72	121
Walks	175	42	22	0	22	38	61	105
Years	175	7	5	1	4	6	11	20
CAtBat	175	2621	2177	41	818	1924	3934	8759
CHits	175	706	608	9	208	491	1072	2583
CHmRun	175	70	80	0	16	40	97	384
CRuns	175	354	308	6	107	247	503	1175
CRBI	175	329	318	7	102	208	448	1299
CWalks	175	263	252	4	74	180	336	1380
League	175							
... A	97	55%						
... N	78	45%						
Division	175							
... E	82	47%						
... W	93	53%						
PutOuts	175	301	293	0	121	222	325	1314
Assists	175	110	138	0	7	43	172	487
Errors	175	8	6	0	3	7	12	25
Salary	175	546	487	68	193	400	750	2460
NewLeague	175							
... A	100	57%						
... N	75	43%						



Slika 2: Univariatne porazdelitve spremenljivk v podatkovnem okviru `bejzbol_train`.

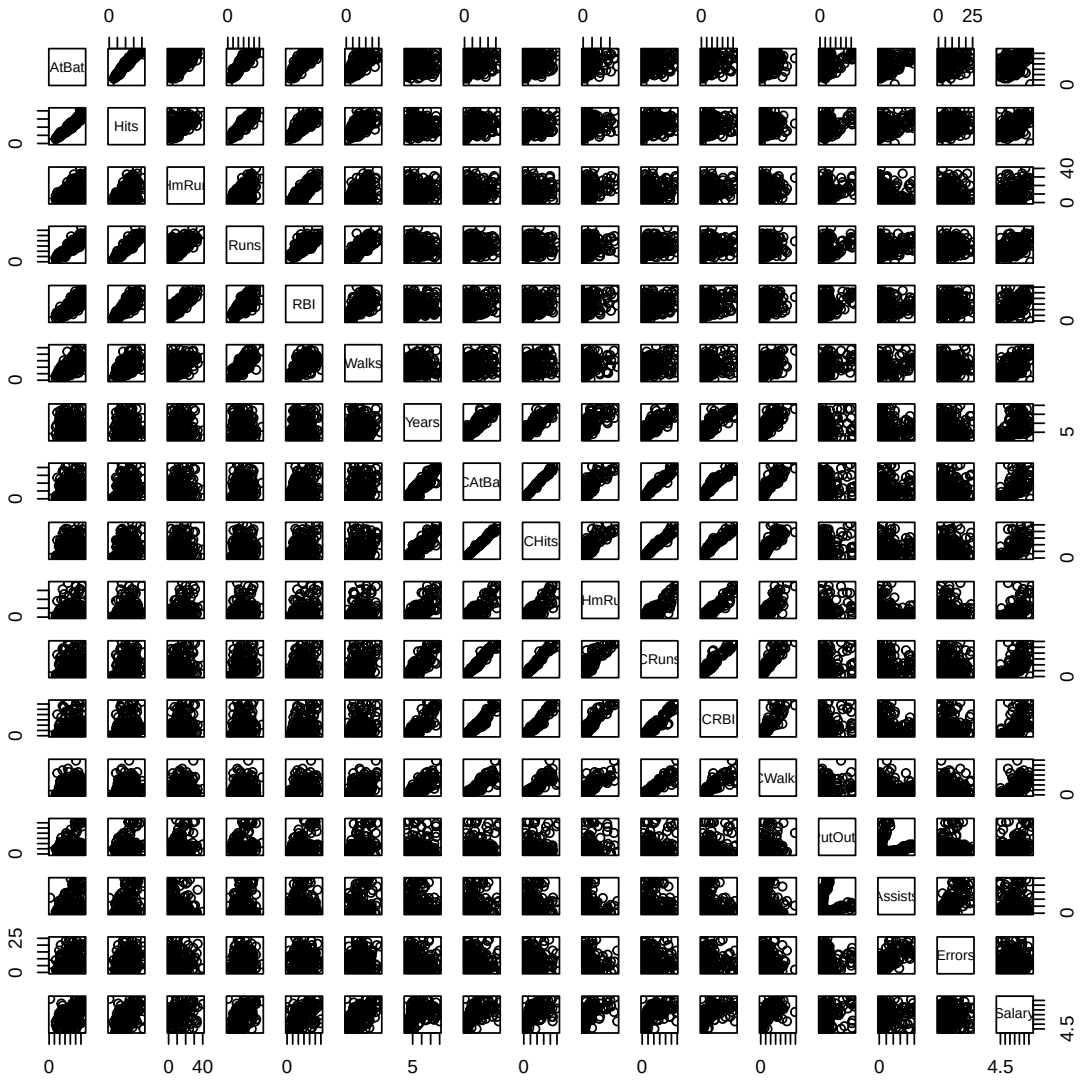


Slika 3: Spearmanovi koeficienti korelacije med napovednimi spremenljivkami v podatkovnem okviru `bejzbol_train`.

Porazdelitev odzivne spremenljivke `Salary` je asimetrična v desno, zato bomo njene vrednosti logaritmirali.

```
train_set$Salary <- log(train_set$Salary)
```

Poglejmo si še matriko razsevnih grafikonov za številske spremenljivke v podatkih.

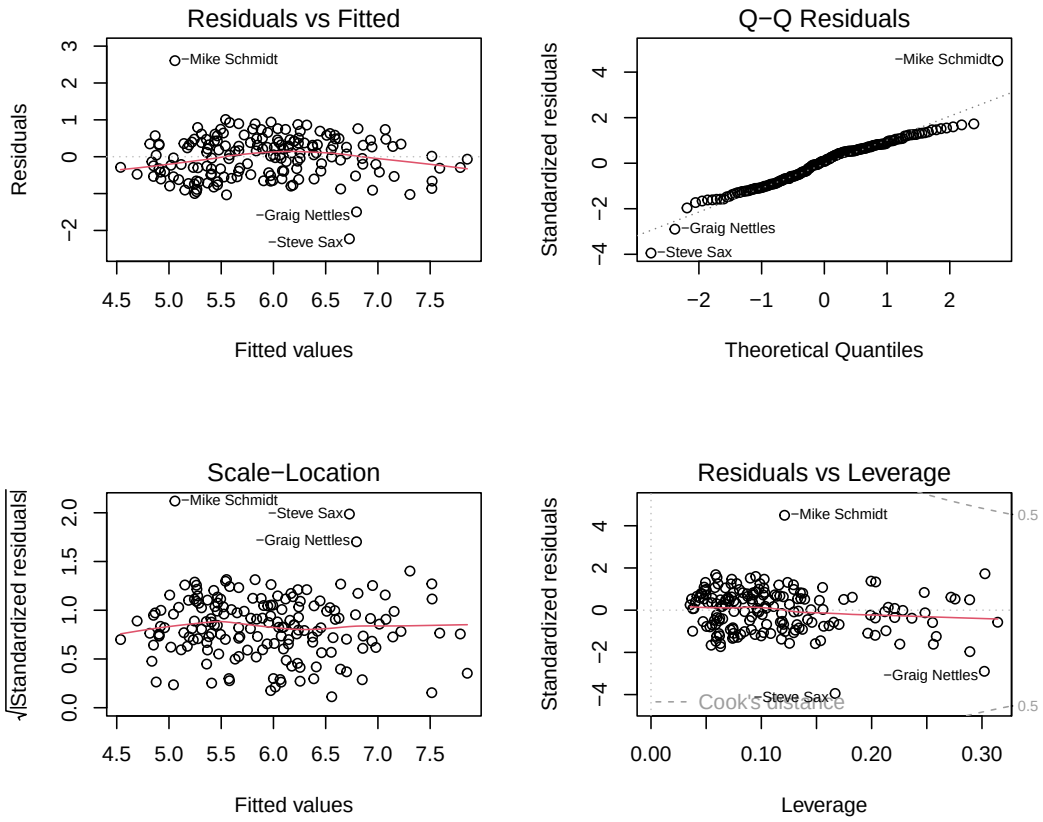


Slika 4: Matrika razsevnih grafikonov za številske spremenljivke v podatkovnem okviru `bejzbol_train`.

Vidimo, da nekatere spremenljivke močno korelirajo.

Najprej naredimo polni model, ki vključuje vseh $k = 19$ napovednih spremenljivk.

```
m0 <- lm(Salary~., data=train_set)

par(mfrow=c(2,2))
plot(m0)
```



Slika 5: Ostanki za polni model `m0`.

Vidimo, da slike ostankov kažejo na nelinearnost v modelu.

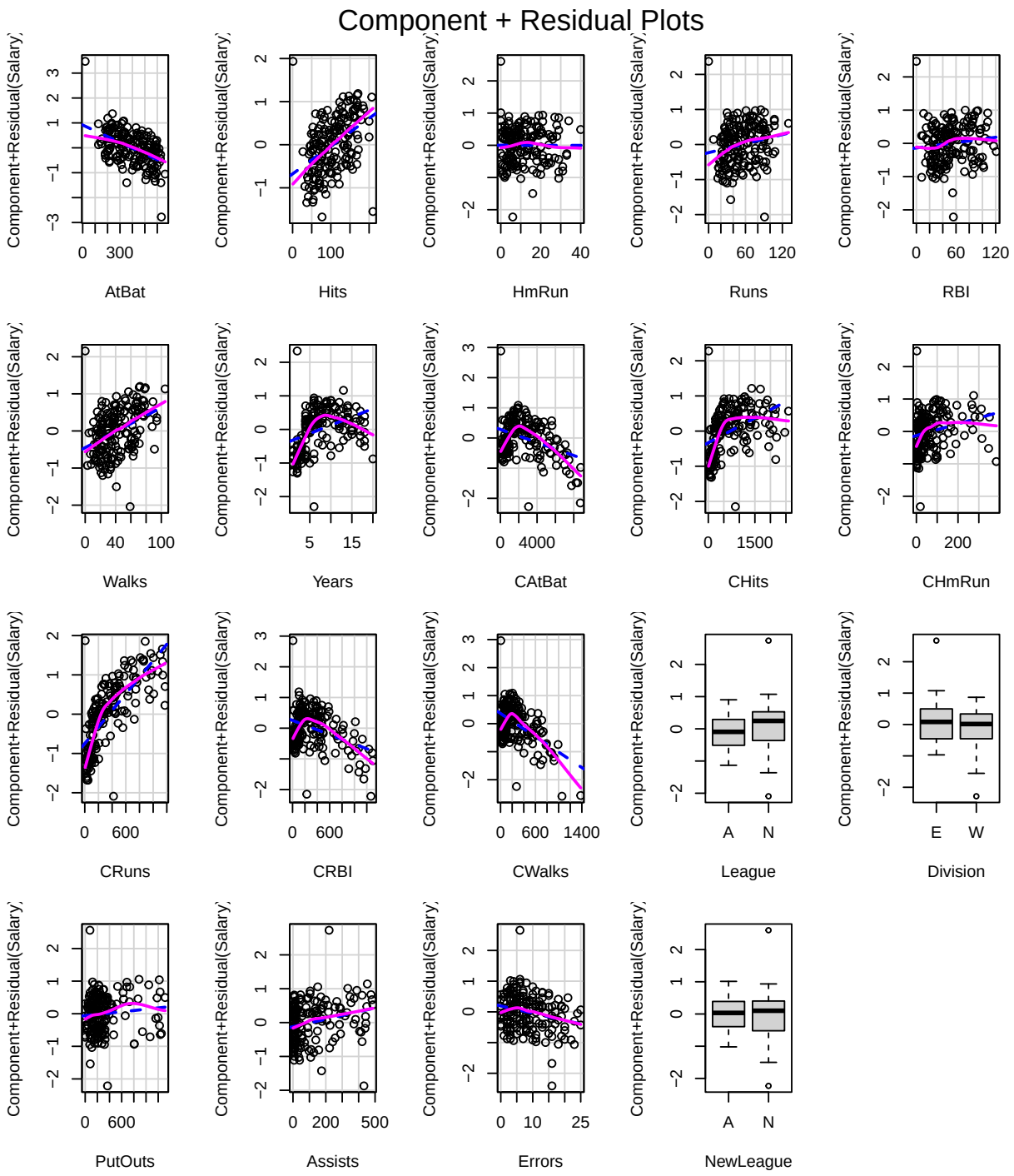
```
vif(m0)
```

```
     AtBat       Hits      HmRun       Runs        RBI      Walks      Years
 25.608718  36.251314   7.991434  19.671897  13.413189   5.749232   9.388015
    CAtBat      CHits     CHmRun      CRuns       CRBI     CWalks     League
249.212622 512.729710  50.599931 173.220856 167.457748  21.334631   5.491511
  Division    PutOuts     Assists     Errors  NewLeague
  1.138661   1.350831   3.053024   2.422279   5.461352
```

V modelu imamo prisotno kolinearnost. Za bolj natančne napovedi ima smisel nekatere spremenljivke izločiti.

Poglejmo si grafe parcialnih ostankov za model `m0`.

```
crPlots(m0, layout = c(4, 5))
```

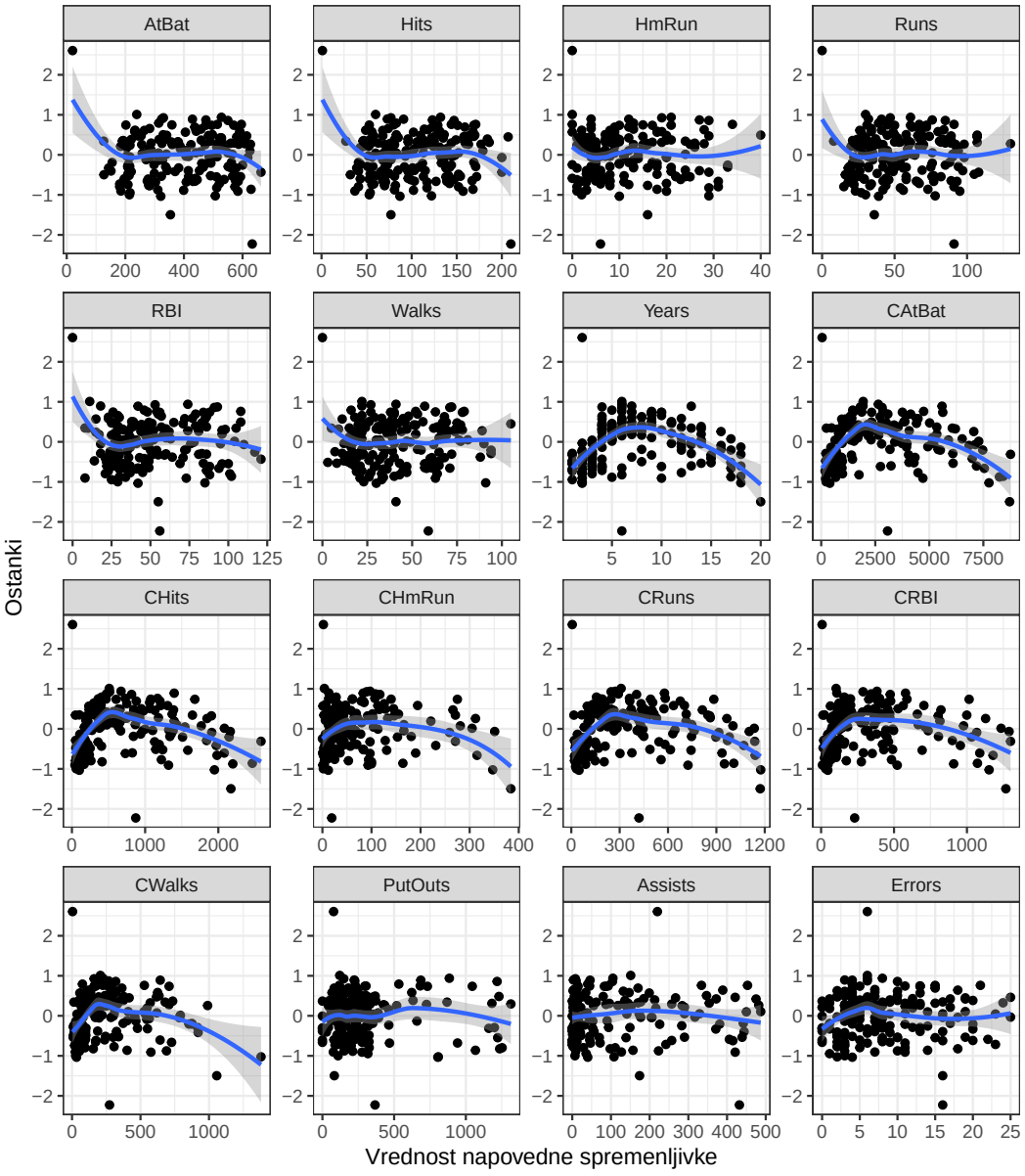


Slika 6: Parcialni ostanki za model `m0` v odvisnosti od napovednih spremenljivk.

Za vizualizacijo napake napovedi za posamezno napovedno spremenljivko lahko prikažemo tudi ostanke v odvisnosti od napovednih spremenljivk:

```
train_resid_long <- cbind(train_long[train_long[,"variable"]!="Salary",],
"resid" = rep(m0$residuals, length(unique(train_long$variable))-1))
```

```
ggplot(data = train_resid_long, aes(x = value, y = resid)) +
  geom_point() +
  geom_smooth(se = T) +
  facet_wrap( ~ variable, scale = "free") +
  xlab("Vrednost napovedne spremenljivke") +
  ylab("Ostanki") +
  theme_bw()
```



Slika 7: Ostanki za model `m0` v odvisnosti od napovednih spremenljivk.

Slika ostankov v odvisnosti od vrednosti napovednih spremenljivk nakazuje na nelinearnost pri nekaterih spremenljivkah, npr. `Years`, `CAtBat`.

V naslednjem koraku bomo poskusili v naš model izbrati spremenljivke, ki so pomembne za napovedovanje

`log(Salary)`, moteče spremenljivke (*noise*) pa odstraniti iz modela.

Izbira najboljše podmnožice spremenljivk

Najprej bomo poskusili z izbiro najboljše podmnožice spremenljivk (*best subset selection*) na podlagi statistike C_p ter prilagojene vrednosti R_a^2. Splošni algoritem za izbiro najboljše množice spremenljivk je sledeči:

1. Naj bo $\mathcal{M}_0$ ničelni model, ki vsebuje le presečišče brez napovednih spremenljivk. Tak model bo dal za vsako enoto napoved, ki bo enaka povprečni vrednosti odzivne spremenljivke.
2. Za $j = 1, 2, \ldots, k$:
   (a) Naredi $\binom{j}{k}$ modelov, ki vsebujejo natanko j napovednih spremenljivk.
   (b) Izmed vseh $\binom{j}{k}$ modelov izberi najboljši model $\mathcal{M}_j$ na podlagi najmanjše RSS oz. največjega R^2.
3. Izmed $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_k$ izberi najboljši končni model na podlagi nekega kriterija za izbiro modela (npr., AIC, prilagojeni R^2).

Pri izbiri najboljše podmnožice spremenljivk si lahko pomagamo s funkcijo `regsubsets` iz paketa `leaps`, argument `nvmax` določa maksimalno število napovednih spremenljivk v modelu. Funkcija na podlagi RSS oz. R^2 izbere `nvmax` najboljših modelov z $1, \ldots,$ `nvmax` napovednimi spremenljivkami.

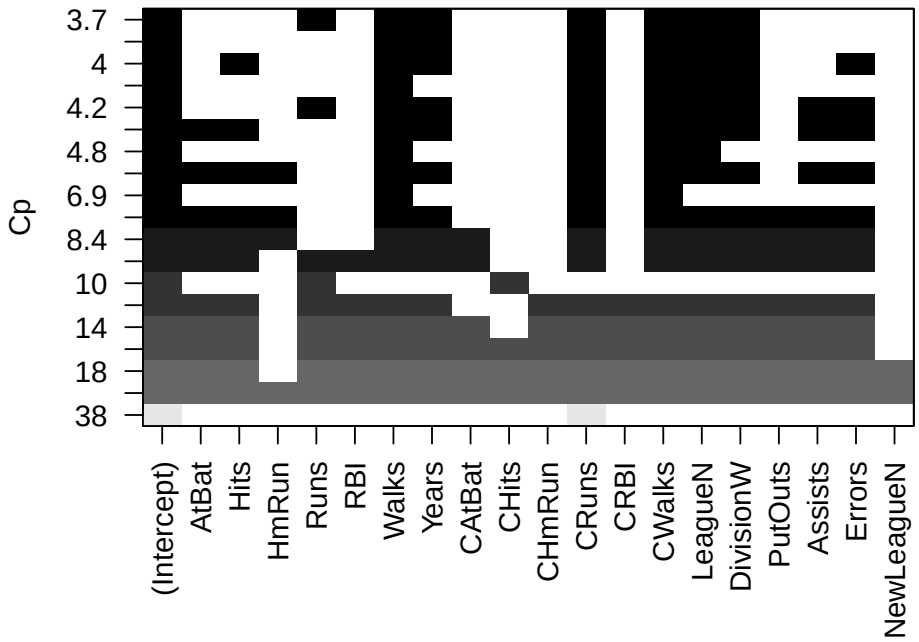
```
# nvmax nastavimo na število vseh napovednih spremenljivk
best_subset = regsubsets(Salary ~. , data = train_set, nvmax = 19)
summary(best_subset)
```

```
Subset selection object
Call: regsubsets.formula(Salary ~ ., data = train_set, nvmax = 19)
19 Variables  (and intercept)
           Forced in Forced out
AtBat          FALSE      FALSE
Hits           FALSE      FALSE
HmRun          FALSE      FALSE
Runs           FALSE      FALSE
RBI            FALSE      FALSE
Walks          FALSE      FALSE
Years          FALSE      FALSE
CAtBat         FALSE      FALSE
CHits          FALSE      FALSE
CHmRun         FALSE      FALSE
CRuns          FALSE      FALSE
CRBI           FALSE      FALSE
CWalks         FALSE      FALSE
LeagueN        FALSE      FALSE
DivisionW      FALSE      FALSE
PutOuts        FALSE      FALSE
Assists        FALSE      FALSE
Errors         FALSE      FALSE
NewLeagueN     FALSE      FALSE
1 subsets of each size up to 19
Selection Algorithm: exhaustive
          AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns CRBI
1  ( 1 )  " "   " "  " "   " "  " " " "   " "   " "    " "   " "    "*"   " "
2  ( 1 )  " "   " "  " "   "*"  " " " "   " "   " "    "*"   " "    " "   " "
3  ( 1 )  " "   " "  " "   " "  " " " "*"  " "   " "    " "   " "    "*"   " "
4  ( 1 )  " "   " "  " "   " "  " " " "*"  " "   " "    " "   " "    "*"   " "
5  ( 1 )  " "   " "  " "   " "  " " " "*"  " "   " "    " "   " "    "*"   " "
```

```
6  ( 1 )  " "   " "  " "   " "  " " "*"   "*"   " "    " "   " "    "*"   " "
7  ( 1 )  " "   " "  " "   "*"  " " "*"   "*"   " "    " "   " "    "*"   " "
8  ( 1 )  " "   "*"  " "   " "  " " "*"   "*"   " "    " "   " "    "*"   " "
9  ( 1 )  " "   " "  " "   "*"  " " "*"   "*"   " "    " "   " "    "*"   " "
10  ( 1 ) "*"   "*"  " "   " "  " " "*"   "*"   " "    " "   " "    "*"   " "
11  ( 1 ) "*"   "*"  "*"   " "  " " "*"   "*"   " "    " "   " "    "*"   " "
12  ( 1 ) "*"   "*"  "*"   " "  " " "*"   "*"   " "    " "   " "    "*"   " "
13  ( 1 ) "*"   "*"  "*"   " "  " " "*"   "*"   "*"    " "   " "    "*"   " "
14  ( 1 ) "*"   "*"  " "   "*"  "*" "*"   "*"   "*"    " "   " "    "*"   " "
15  ( 1 ) "*"   "*"  " "   "*"  "*" "*"   "*"   " "    " "   "*"    "*"   "*"
16  ( 1 ) "*"   "*"  " "   "*"  "*" "*"   "*"   "*"    " "   "*"    "*"   "*"
17  ( 1 ) "*"   "*"  " "   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
18  ( 1 ) "*"   "*"  " "   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
19  ( 1 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
          CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
1  ( 1 )  " "    " "     " "       " "     " "     " "    " "
2  ( 1 )  " "    " "     " "       " "     " "     " "    " "
3  ( 1 )  "*"    " "     " "       " "     " "     " "    " "
4  ( 1 )  "*"    "*"     " "       " "     " "     " "    " "
5  ( 1 )  "*"    "*"     "*"       " "     " "     " "    " "
6  ( 1 )  "*"    "*"     "*"       " "     " "     " "    " "
7  ( 1 )  "*"    "*"     "*"       " "     " "     " "    " "
8  ( 1 )  "*"    "*"     "*"       " "     " "     "*"    " "
9  ( 1 )  "*"    "*"     "*"       " "     "*"     "*"    " "
10  ( 1 ) "*"    "*"     "*"       " "     "*"     "*"    " "
11  ( 1 ) "*"    "*"     "*"       " "     "*"     "*"    " "
12  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
13  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
14  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
15  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
16  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
17  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    " "
18  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
19  ( 1 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

Funkcija `plot.regsubsets` prikaže izbiro najboljše podmožice za vsak model z $1, \ldots,$ `nvmax` napovednimi spremenljivkami.

```
plot(best_subset, scale = "Cp")
```



Slika 8: Izbrane spremenljivke za najboljši model po Cp-statistiki glede na število napovednih spremenljivk v modelu.

Med 19 najboljšimi modeli lahko izbiramo na podlagi različnih kriterijev, npr. C_p ali Adj R^2.

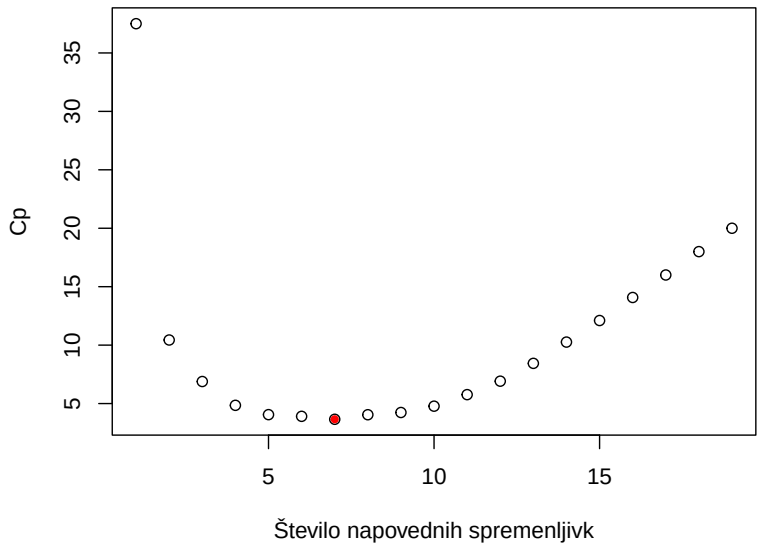
```
best_subset_summary = summary(best_subset)
names(best_subset_summary)
```

```
[1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
which.min(best_subset_summary$cp)
```

```
[1] 7
```

```
plot(best_subset_summary$cp, xlab = "Število napovednih spremenljivk", ylab = "Cp")
points(which.min(best_subset_summary$cp),
       best_subset_summary$cp[which.min(best_subset_summary$cp)],
       pch = 20, col = "red")
```



Slika 9: Vrednosti *Cp*-statistike v odvisnosti od števila napovednih spremenljivk v modelu.

Glede na kriterij *Cp* je najboljši model z 7 spremenljivkami. Poglejmo si ocene parametrov v izbranem modelu:

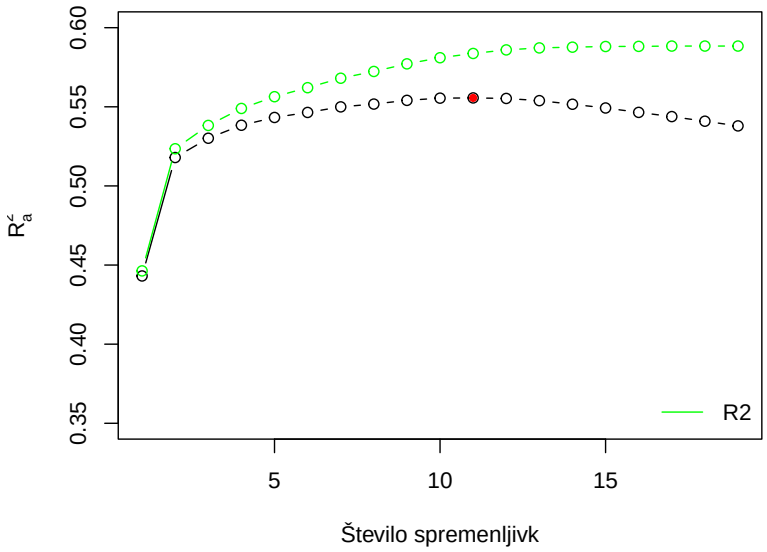
```
coef(best_subset, which.min(best_subset_summary$cp))
```

```
 (Intercept)         Runs        Walks        Years        CRuns       CWalks
 4.501696682  0.005629444  0.010285639  0.042742598  0.001989852 -0.001315601
    LeagueN    DivisionW
 0.215125115 -0.161028870
```

```
which.max(best_subset_summary$adjr2)
```

```
[1] 11
```

```
plot(best_subset_summary$adjr2,
     xlab = "Število spremenljivk", ylab = expression(R[a]^2),
     ylim=c(0.35, 0.6), type="b")
points(which.max(best_subset_summary$adjr2),
       best_subset_summary$adjr2[which.max(best_subset_summary$adjr2)],
       pch = 20, col = "red")
points(best_subset_summary$rsq, col = "green", type="b")
legend("bottomright", c("R2"),  col="green", bty="n", lty = 1)
```



Slika 10: Vrednosti R_a^2 (ter R^2) v odvisnosti od števila napovednih spremenljivk v modelu.

Glede na prilagojeno vrednost R^2 je najboljši model z 11 napovednima spremenljivkama.

```
coef(best_subset, which.max(best_subset_summary$adjr2))
```

```
  (Intercept)         AtBat          Hits         HmRun         Walks
 4.6642467016 -0.0021425566  0.0082527712  0.0076003573  0.0133939328
        Years         CRuns        CWalks       LeagueN     DivisionW
 0.0390139756  0.0021632827 -0.0015558088  0.2288020223 -0.1452144205
      Assists        Errors
 0.0009387473 -0.0226070587
```

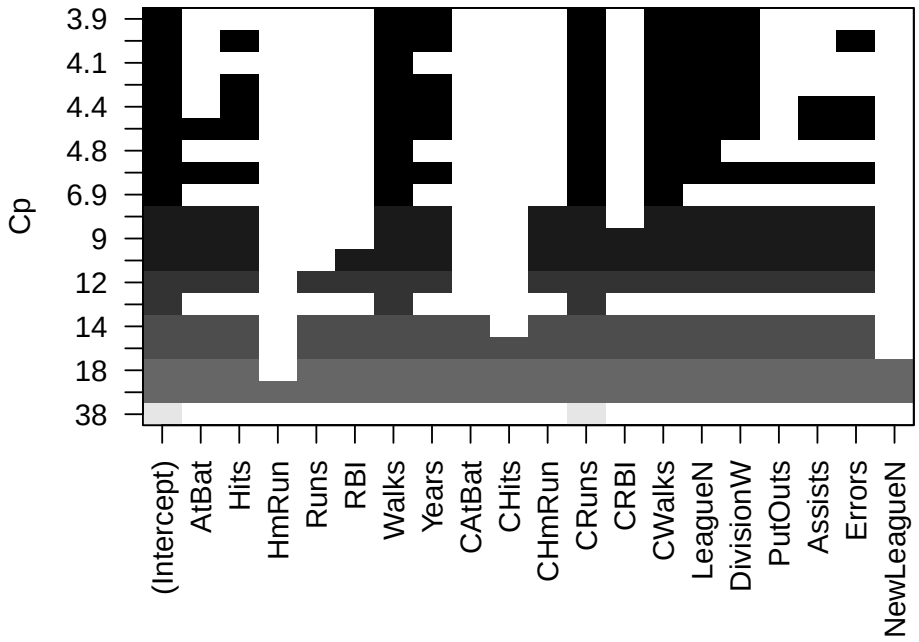
Izbira po korakih

V praksi izbira najboljše podmnožice spremenljivk pogosto ni mogoča. Alternative vključujejo izbiro nazaj, izbiro naprej ter hibridne pristope. Kadar imamo na voljo manj napovednih spremenljivk kot je enot, je v praksi izbira nazaj bolj zaželjena, še posebej v primeru kolinearnosti. Po drugi strani izbire nazaj ne moremo uporabiti na visokorazsežnih podatkih, medtem ko izbiro naprej lahko.

Paket `leaps` omogoča tako izbiro modela z izbiro nazaj kot z izbiro naprej. Splošni algoritem za izbiro nazaj je sledeči:

1. Naj bo $\mathcal{M}_k$ polni model, ki vsebuje vse napovedne spremenljivke.
2. Za $j = k, k-1, \ldots, 1$:
   (a) Naredi j modelov, ki vsebujejo vse razen ene napovedne spremenljivke modela $\mathcal{M}_j$, skupaj torej $j-1$ napovednih spremenljivk.
   (b) Izmed teh j modelov izberi najboljši model $\mathcal{M}_{j-1}$ na podlagi najmanjše RSS oz. največjega R^2.
3. Izmed $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_k$ izberi najboljši končni model na podlagi nekega kriterija za izbiro modela (npr., AIC, prilagojeni R^2).

```
bwd_sel = regsubsets(Salary ~., data = train_set, nvmax = 19, method = "backward")
bwd_sel_summary <- summary(bwd_sel)
plot(bwd_sel, scale = "Cp")
```

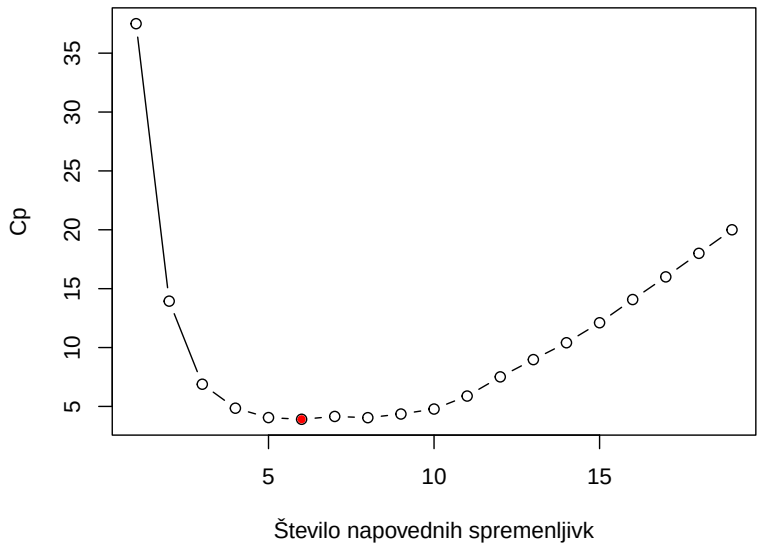


Slika 11: Izbrane spremenljivke za najboljši model po izbiri nazaj (*Cp*-statistika) glede na različno število napovednih spremenljivk v modelu.

```
which.min(bwd_sel_summary$cp)
```

```
[1] 6
```

```
plot(bwd_sel_summary$cp,
     xlab = "Število napovednih spremenljivk", ylab = "Cp", type="b")
points(which.min(bwd_sel_summary$cp),
       bwd_sel_summary$cp[which.min(bwd_sel_summary$cp)],
       pch = 20, col = "red")
```



Slika 12: Vrednosti Cp-statistike pri izbiri nazaj v odvisnosti od števila napovednih spremenljivk v modelu.

Izbira nazaj po Cp-kriteriju izbere model z 6 spremenljivkami. Vidimo, da izbira nazaj ne vodi do enake izbire napovednih spremenljivk kot pa izbira najboljše podmnožice.

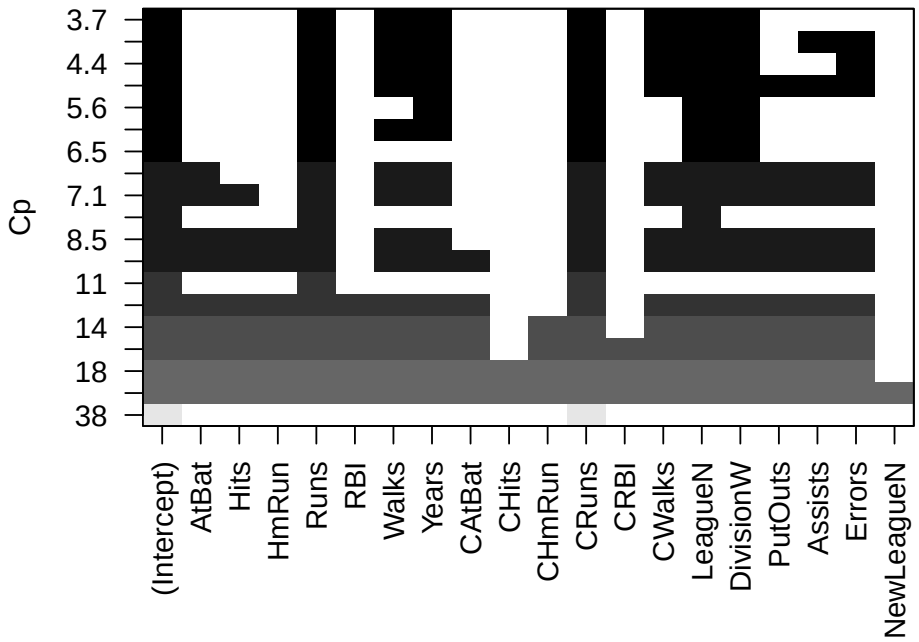
```
coef(bwd_sel, which.min(bwd_sel_summary$cp))
```

```
 (Intercept)        Walks        Years        CRuns       CWalks      LeagueN
 4.619936805  0.015627000  0.033170883  0.002506946 -0.001824196  0.186624456
   DivisionW
-0.164085287
```

Po podobnem principu deluje tudi izbira naprej, le da začnemo z ničelnim modelom in postopoma dodajamo spremenljivke. Splošni algoritem za izbiro naprej je sledeči:

1. Naj bo $\mathcal{M}_0$ ničelni model, ki ne vsebuje nobene napovedne spremenljivke.
2. Za $j = 0, \ldots, k-1$:
   (a) Naredi vseh $k - j$ modelov, ki v modelu $\mathcal{M}_j$ dodajo eno napovedno spremenljivko.
   (b) Izmed teh $k - j$ modelov izberi najboljši model $\mathcal{M}_{j+1}$ na podlagi najmanjše RSS oz. največjega R^2.
3. Izmed $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_k$ izberi najboljši končni model na podlagi nekega kriterija za izbiro modela (npr., AIC, prilagojeni R^2).

```
fwd_sel = regsubsets(Salary ~., data = train_set, nvmax = 19, method = "forward")
fwd_sel_summary <- summary(fwd_sel)
plot(fwd_sel, scale = "Cp")
```

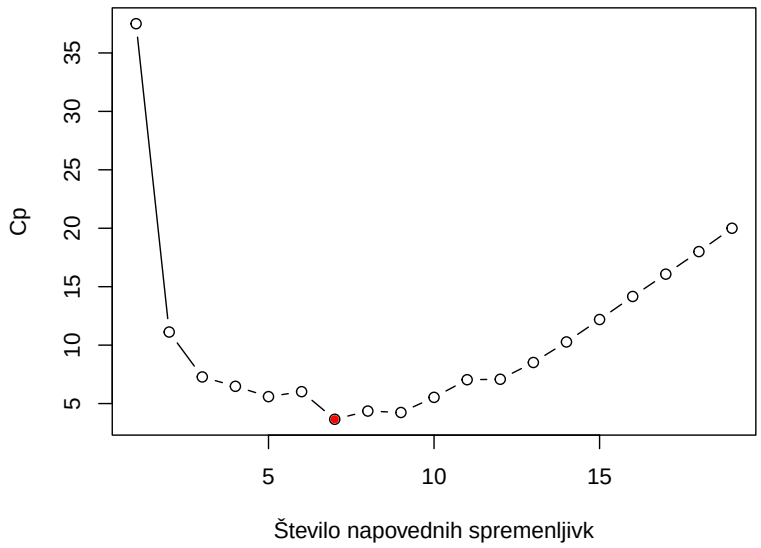


Slika 13: Izbrane spremenljivke za najboljši model po izbiri naprej (*Cp*-statistika) glede na različno število spremenljivk v modelu.

```
which.min(fwd_sel_summary$cp)
```

```
[1] 7
```

```
plot(fwd_sel_summary$cp,
     xlab = "Število napovednih spremenljivk", ylab = "Cp", type="b")
points(which.min(fwd_sel_summary$cp),
       fwd_sel_summary$cp[which.min(fwd_sel_summary$cp)],
       pch = 20, col = "red")
```



Slika 14: Vrednosti *Cp*-statistike pri izbiri nazaj v odvisnosti od števila spremenljivk v modelu.

Izbira nazaj po C_p-kriteriju izbere model s 7 spremenljivkami.

```
coef(fwd_sel, which.min(fwd_sel_summary$cp))
```

```
 (Intercept)         Runs        Walks        Years        CRuns       CWalks
 4.501696682  0.005629444  0.010285639  0.042742598  0.001989852 -0.001315601
    LeagueN    DivisionW
 0.215125115 -0.161028870
```

K-kratno navzkrižno preverjanje

Statistiki Cp ali Adj R^2 kakovost modela ocenjujeta posredno, tako da vrednost RSS prilagodita glede na število napovednih spremenljivk v modelu. Pri K-kratnem navzkrižnem preverjanju kakovost modela, ki smo ga dobili na učnem vzorcu, ocenjujemo neposredno na validacijskem vzorcu tako, da podatke razdelimo na K enako velikih delov; za npr. $K = 5$ dobimo:

Slika 15: Razdelitev podatkov pri K-kratnem navzkrižnem preverjanju.

Na vsakem od $j = 1, \ldots, K$ korakov uporabimo $(K-1)/K$ podatkov, da zgradimo model, j-ti del podatkov pa uporabimo, da izračunamo napako napovedi zgrajenega modela pri napovedovanju j-tega dela podatkov. To ponovimo K-krat in povprečimo ocene napake napovedi:

$$CV_{(j)} = \frac{1}{K}\sum_{j=1}^{K} MSE_j.$$

Pomembno pri K-kratnem navzkrižnem preverjanju je to, da v posameznem j-tem koraku ponovimo vse korake modeliranja!

Da lahko v nadaljevanju izvedemo K-kratno navzkrižno preverjanje, bomo definirali funkcijo, ki vrne napovedi za vsak najboljši model na $K-1$-delu podatkov z $1,\ldots,$`nvmax` napovednimi spremenljivkami.

```
predict.regsubsets <- function(object, newdata, id){
  form = as.formula(object$call[[2]])
  mat = model.matrix(form, newdata)
  coefi = coef(object, id=id)
  xvars = names(coefi)
  mat[,xvars]%*%coefi
}
```

Pri 5-kratnem navzkrižnem preverjanju moramo postopek izbire najboljše podmnožice spremenljivk ponoviti petkrat. V vsakem koraku učni vzorec sestavlja 4/5 podatkov. Za vsakega od 19 najboljših modelov izračunamo povprečno kvadratno napako napovedi (MSE), ki je definirana kot povprečna kvadrirana razdalja med dejanskimi vrednostmi odzivne spremenljivke na validacijskem vzorcu, ki ga sestavlja 1/4 podatkov, ter napovedmi na podlagi ocen izbranega modela na učnem vzorcu. Na koncu napake povprečimo po vseh 5 ponovitvah.

Navzkrižno preverjanje nam da optimalno število regresorjev v modelu ne pa tudi, kateri regresorji so izbrani v model.

V kodi v nadaljevanju bo `R` samodejno uporabil funkcijo `predict.regsubsets()`, ko kličemo `predict()`, saj je objekt `best.fit` razreda `regsubsets`.

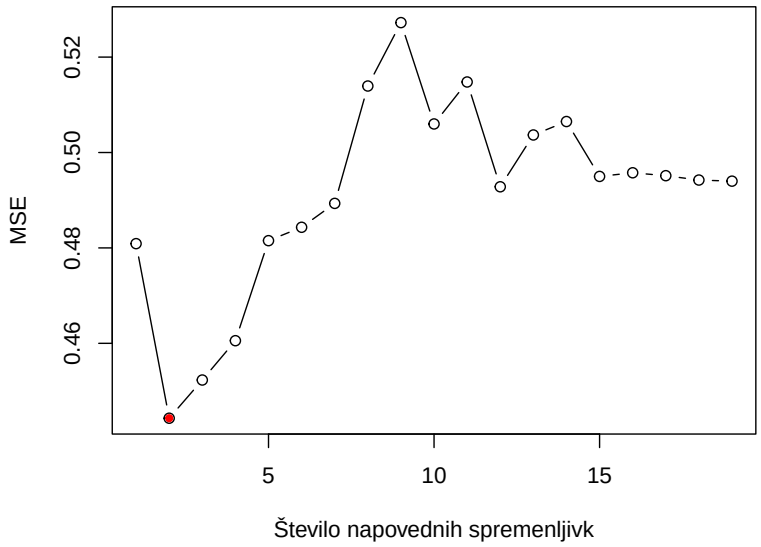
```
K=5
set.seed(1)
folds <- sample(1:K, nrow(train_set), replace=TRUE)
cv.errors <- matrix(NA, K, 19, dimnames=list(NULL, paste(1:19)))

for(j in 1:K){
  best.fit=regsubsets(Salary~., data=train_set[folds!=j,], nvmax=19)
  for(i in 1:19){
    pred = predict(best.fit, train_set[folds==j,], id=i)
    cv.errors[j, i] = mean((train_set$Salary[folds==j] - pred)^2)
  }
}

mean.cv.errors=apply(cv.errors, 2, mean)
mean.cv.errors
```

```
        1         2         3         4         5         6         7         8
0.4808965 0.4443018 0.4523086 0.4605514 0.4815202 0.4843258 0.4893338 0.5139379
        9        10        11        12        13        14        15        16
0.5272271 0.5059845 0.5147840 0.4928150 0.5036741 0.5064932 0.4949885 0.4957509
       17        18        19
0.4951395 0.4942262 0.4939968
```

```
which.min(mean.cv.errors)
```

```
2
2
```

```
plot(mean.cv.errors, xlab = "Število napovednih spremenljivk", ylab = "MSE", type="b")
points(which.min(mean.cv.errors),
       mean.cv.errors[which.min(mean.cv.errors)],
       pch = 20, col = "red")
```



Slika 16: Vrednosti povprečne kvadratne napake napovedi v odvisnosti od števila napovednih spremenljivk v modelu.

Navzkrižno preverjanje izbere model z 2 napovednima spremenljivkama.

```
coef(best_subset, which.min(mean.cv.errors))
```

```
 (Intercept)         Runs        CHits
4.7005162153 0.0115057455 0.0008495368
```

Domača naloga: Izbira modela

Za podatke `bejzbol_train.txt` naredite svoj napovedni model z uporabo zlepkov, interakcij, metodami izbire modela ... Posamezne korake analize komentirajte. Svoj končni model shranite z uporabo funkcije `saveRDS(model_koncni, "PRIIMEK.rds")` in ga skupaj s poročilom naložite v spletno učilnico.