w2ex2

Paul Hosek

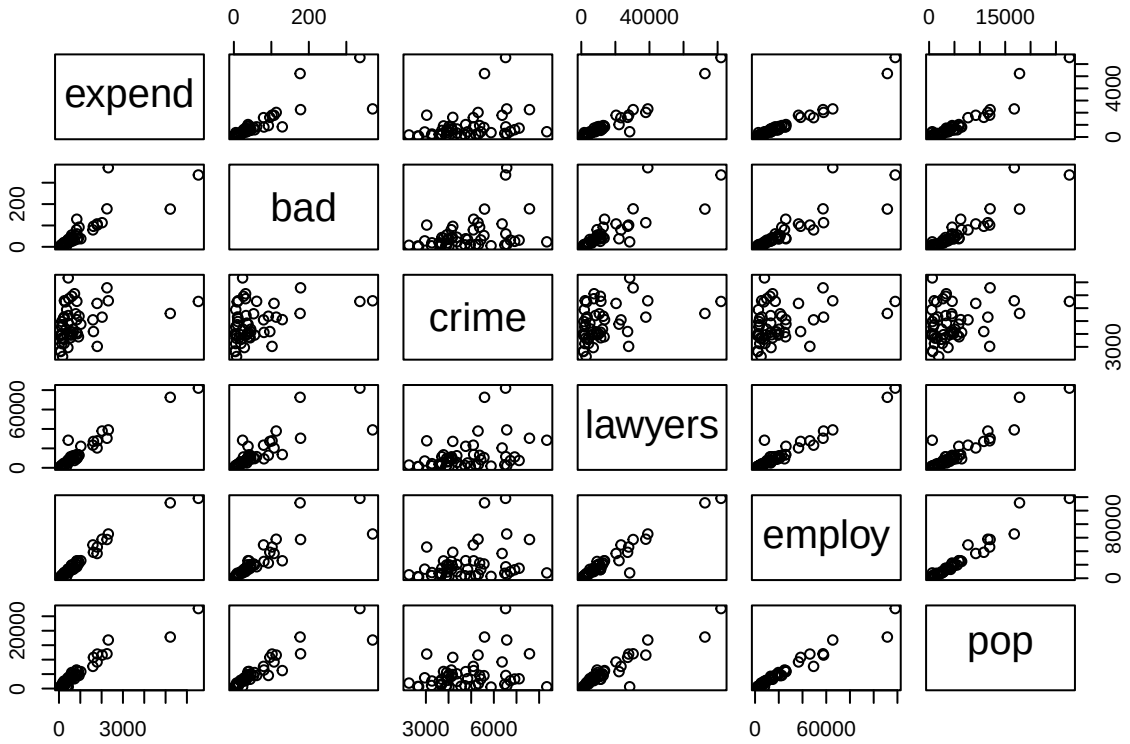
2023-03-03

Ex 2

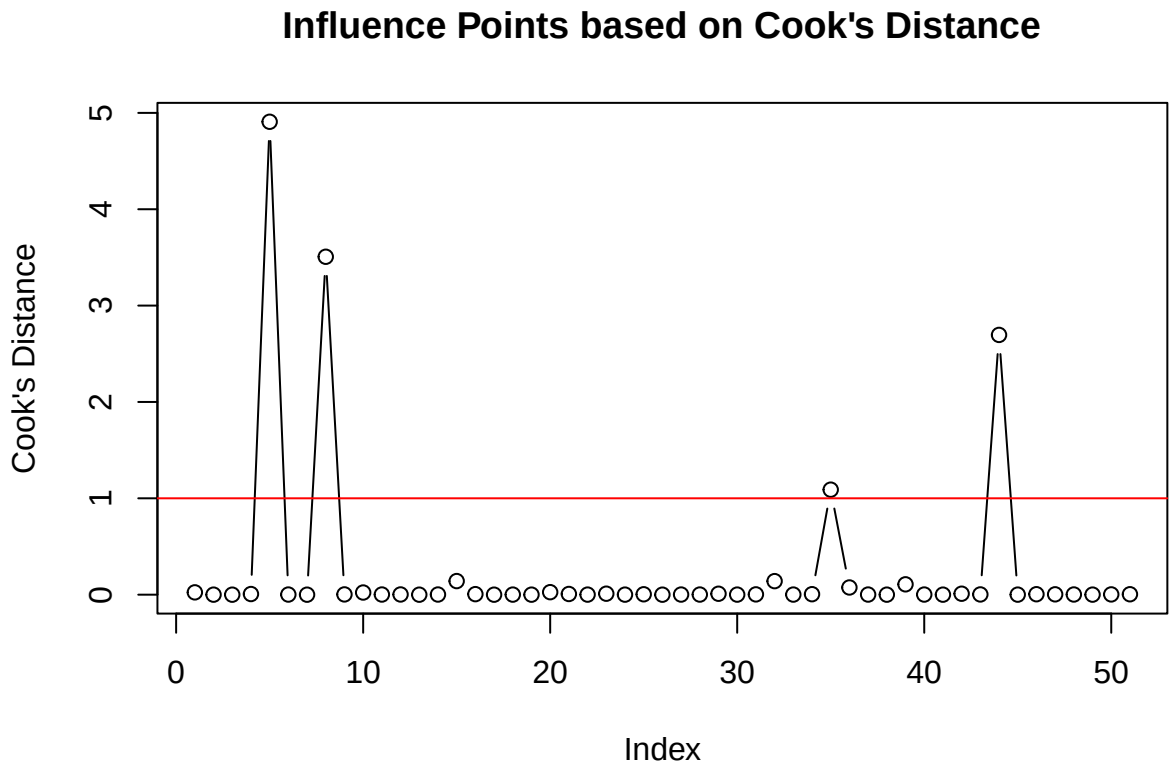
```
data <- read.table("expensescrime.txt", header = TRUE);
```

a)

```
# graphical summary?
pairs(data[,2:7])
```



```r
# Influence points
full_model <- lm(expend ~ bad + crime + lawyers + employ + pop, data = data)
cdist <- cooks.distance(full_model)
plot(1:length(cdist), cdist, xlab = "Index", ylab = "Cook's Distance",
     main = "Influence Points based on Cook's Distance", type="b")
abline(h=1, col="red")
```




```r
# Collinearity
print(vif(full_model))
```

```
##       bad     crime   lawyers    employ       pop
##  8.364321  1.487978 16.967470 33.591361 32.937517
```

Our graphical summary of the data shows that the crime rate, lawyers, employment and population are most notably related with expenditure. "Crime" shows random variation with all other variables.

Analyses of Influence points based on Cook's distance indicates that 4 observations with disproportionate influence on the model, with one marginally outside the cutoff of one. All four observations must undergo further inspection and potentially dropped from the model.

Concerning Colliniarity, our rule of thumb dicatates that a VIF > 5 is concerning, indicating $R^2 > .8$. In our dataset, all variables besides "crime" were found with VIF >5, further "Pop" and "Employ" show extreme VIF values with VIF >30. This indicates that at least one variable is a linear combination of the others. This corroborates the inpection of the scatter-matrix from before, indicating close relationships between the variables. We can try to drop a single term from the model, but may need to drop more.

Table 1: VIF Values by term dropped from full model and Variable

	vif.bad	vif.crime	vif.lawyers	vif.employ	vif.pop
Population	4.463587	1.305439	16.75263	21.09127	NA
Employment	8.243504	1.487593	10.08422	NA	20.68074
Lawyers	8.070920	1.308699	NA	19.96423	32.52047
Crime Rate	NA	1.233263	16.37229	33.10616	17.57698

Dropping the term with the highest vif did not reduce all VIFs to under five, as such we should use the step-down or step-up method to select the correct model.

b) Fit a linear regression model to the data using the step-up method.

This method can be summarized in 4 steps:

1. Build model with constant predictor (background model).
2. Find term that, if added to the model would maximise R^2.
3. If the variable is significant, add it to the model.
4. Go to 2 until 3 does not occur.

```
# find the next best term that significantly improves model fit
find_best_term <- function(formula, data) {
  best_fit <- summary(lm(formula, data))$adj.r.squared
  best_term <- 1
  # maximize r^2
  for (term in names(data[,-c(1:2)])) {
    if (!term %in% attr(formula, "term.labels")) {
      cur_formula <- paste(formula, term, sep="+")
      cur_model <- lm(cur_formula, data = data)
      cur_fit <- summary(cur_model)$r.squared
      if (cur_fit > best_fit) {
        best_fit <- cur_fit
        best_term <- term
        best_coef_pval <- summary(cur_model)$coefficients[
          nrow(summary(cur_model)$coefficients), "Pr(>|t|)"]

      }
    }
  }
  # test if increase is significant and return new formula
  if (best_coef_pval<0.05){
    return(best_term)
  } else {
    return(1)
  }
}
# recursively add terms
```

```
step_up <- function(formula, data){
  new_term <- find_best_term(formula,data)
  if (new_term == 1){
    return(formula)}
  step_up(paste(formula, new_term, sep="+"), data)
}

best_formula <- step_up("expend~1", data)
best_model <- lm(best_formula, data=data)
print(summary(best_model))
```

```
##
## Call:
## lm(formula = best_formula, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -599.47  -94.43   36.01   91.98  936.55
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.107e+02  4.257e+01  -2.600  0.01236 *
## employ       2.971e-02  5.114e-03   5.810 4.89e-07 ***
## lawyers      2.686e-02  7.757e-03   3.463  0.00113 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 232.6 on 48 degrees of freedom
## Multiple R-squared:  0.9632, Adjusted R-squared:  0.9616
## F-statistic: 627.7 on 2 and 48 DF,  p-value: < 2.2e-16
```

```
print(vif(best_model))
```

```
##   employ  lawyers
## 14.83915 14.83915
```

The model found includes only "employ" and "lawyers". It explains a high proportion of the variance ($R^2 = 0.96$). Analysis of collinearity found all VIF values larger than five. This model may not be the most suitable model as factors linearly depend on another.

c)

```
hypo_state= data.frame(bad = 50, crime = 5000, lawyers = 5000, employ = 5000, pop = 5000)

pred_interv_b <- predict(best_model,hypo_state,interval="prediction",level=0.95)
```

Given this model, we predict $p = 172.21$ for "expend". 95% and lower bounds are: $[-302.93, 647.35]$. We could improve this prediction by trying out different models (e.g., add more terms) and examine if the prediction interval becomes smaller. For example, we could try the full model from earlier:

```
pred_interv_f <- predict(full_model,hypo_state,interval="prediction",level=0.95)
```

The prediction interval of the full model is larger (1003.18) than using the model found with the step-up method (950.28).

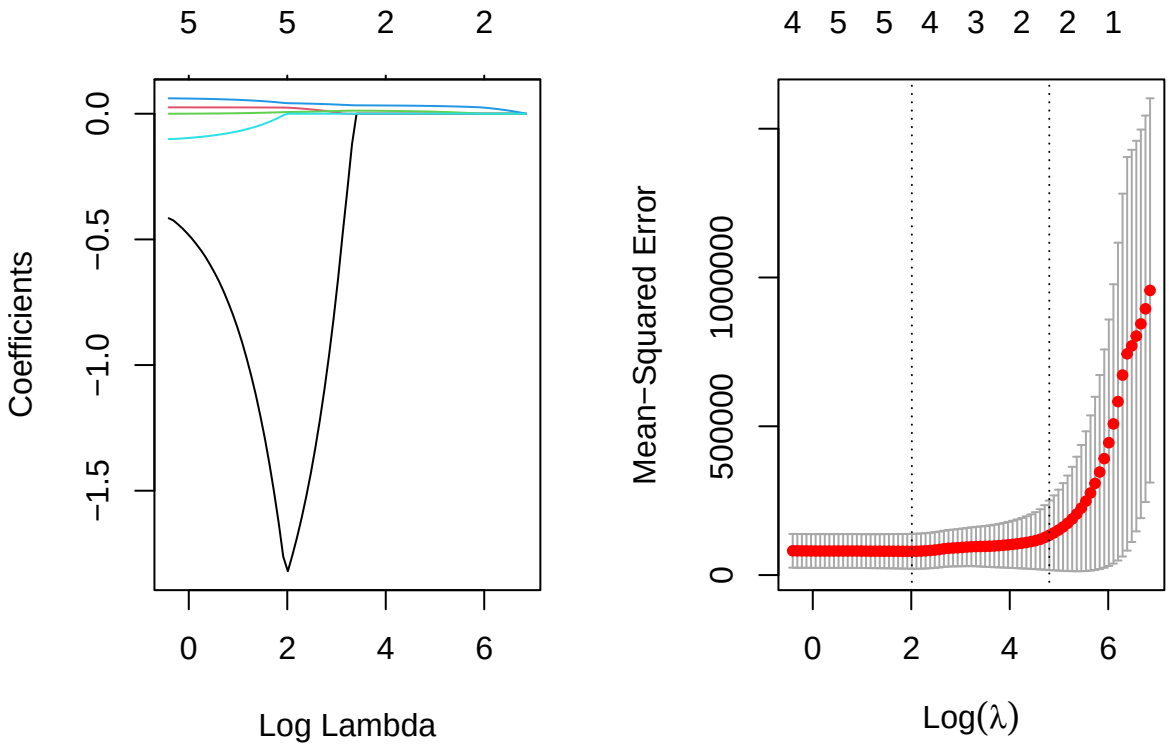
d) Lasso method

- compare with b

```
x <- as.matrix(data[,-2:-1]) # remove expend and state
y <- as.double(as.matrix(data[,2])) # expend is response

train=sample(1:nrow(x),0.67*nrow(x))
x.train=x[train,]; y.train=y[train]
x.test=x[-train,]; y.test=y[-train]

lasso.mod=glmnet(x.train,y.train,alpha=1)
lasso.cv=cv.glmnet(x.train,y.train,alpha=1,type.measure="mse")
par(mfrow=c(1,2))

lambda.min=lasso.cv$lambda.min; lambda.1se=lasso.cv$lambda.1se
coef(lasso.mod,lasso.cv$lambda.min) #beta's for the best lambda
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept) -1.523811e+02
## bad         -1.820418e+00
## crime        2.410908e-02
## lawyers      6.796655e-03
## employ       4.192648e-02
## pop          .
```

```
y.pred=predict(lasso.mod,s=lambda.min,newx=x.test) #predict for test
mse.lasso=mean((y.test-y.pred)^2) #mse for the predicted test rows

plot(lasso.mod,label=T,xvar="lambda")  #have a look at the lasso path         # idk if we need the plot
plot(lasso.cv) # the best lambda by cross-validation
```




```
# plot(lasso.cv$glmnet.fit,xvar="lambda",label=T) # same as before

print(mse.lasso)
```

```
## [1] 178435.6
```

```
lass_model = lm("expend~bad", data=data)
summary(lass_model)
```

```
##
## Call:
## lm(formula = "expend~bad", data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2744.80  -129.97   -69.01    91.78  2738.99
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  126.721    114.865   1.103    0.275
## bad           13.324      1.257  10.601  2.8e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 661 on 49 degrees of freedom
## Multiple R-squared:  0.6964, Adjusted R-squared:  0.6902
## F-statistic: 112.4 on 1 and 49 DF,  p-value: 2.796e-14
```

The model chosen by the LASSO method only includes "bad" (i.e., the crime rate) as a predictor. This model is much more parsimonious than the model chosen in b, as it only includes a single predictor (Occam's Razor). However, this model explains $R^2 = 0.70$ of the variance, which is lower than the variance explained by our previous model ($R^2 = 0.96$).