FSRM588 HW 1

Yaniv Bronshtein

2/16/2021

1.*Test MSE, Training MSE, EPE.*
Consider the linear regression model with p
parameters, fit by least squares to a set of training
$(x_1, y_1), ..., (x_N, y_N)$ drawn from the population distribution where
$x_i \sim N_p(0, \sum)$ and $y_i = x_i^T$ with $\epsilon_i \sim N(0, \sigma^2)$.
Let $\hat{\beta}$ be the least squares estimate. Suppose we have some test data
$(\tilde{x_1}, \tilde{y_1}), ..., (\tilde{x_M}, \tilde{y_M})$ drawn at random
from the same population as the training data. Define the training MSE as
$R_{tr}(\hat{\beta}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T \hat{\beta})^2$
and the test MSE as $R_{te}(\hat{\beta}) = \frac{1}{M} \sum_{i=1}^{M} (\tilde{y_i} - \tilde{x_i}^T \hat{\beta})^2$,
prove that

$$ER_{tr}(\hat{\beta}) <= ER_{te}(\hat{\beta}),$$

where the expectations are over all that is random
in each expression.

Solution Let us first provide some definitions:

$$\hat{\beta} = argmin_{\beta \in \mathbb{R}^{p+1}} \sum_{i=1}^{n} (y_i - \beta^T x_i)$$

Since we are given the the data consists of independent and identically distributed random variables, we can safely say that

$$\frac{1}{N} \sum_{i=1}^{N} \mathbb{E}[Y_i - \beta^T X_k)^2] = \mathbb{E}[Y_i - \beta^T X_i)^2],$$

This holds for each random vector β and for $i = 1, ..., N$

Let us divide both sides of the equation by N and take the expectations

2.*Confidence Set for regression Coefficients*
Consider the linear regression model with p parameters, fit by the least squares
to a set of training data $(x_1, y_1), ..., (x_N, y_N)$ drawn from the population distribution where
$x_i \sim N_p(0, \sum)$ and $y_i = x_i^T$ with $\epsilon_i \sim N(0, \sigma^2)$
Please use one method to construct confidence sets for β. If you have more than one practical
method of constructing confidence sets for β, you will obtain bonus points.

3.*Centering the data and penalized least squares. Consider the ridge regression problem

$$\beta^{ridge} = argmin_{\beta \in \mathbb{R}^{p+1}} \{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j^2)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \}$$

Show that this problem is equivalent to the problem

$$\beta^c = argmin_{\beta^c \in \mathbb{R}^{p+1}} \{ \sum_{i=1}^{N} (y_i - \beta_0^c - \sum_{j=1}^{p} (x_{ij} - \bar{x_j}) \beta_j^c)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \}$$

where $\bar{x_j} = N^{-1} \sum_{i=1}^{N} x_{ij}$. More specifically:
(a) Give the correspondence between $\hat{\beta}^c$ and the $\hat{\beta}^{ridge}$
(b). Show that the two predicted output vectors are the same.
(c). Give an expression of β_0^c using $y_1, ..., y_n$

Show that a similar result holds for the Lasso.

Applied Problems

4.*Simulations with ridge.* Let $\beta \in \mathbb{R}^p$ and let x, y be random variables such that the entries of x are i.i.d. Rademacher random variables (i.e., +1
with probability 0.5 and -1 with probability 0.5) and $y = \beta^T x + \epsilon$ where $\epsilon \sim N(0, 1)$ such that the entries of x (a). Show that for any function $f : \mathbb{R}^p \to \mathbb{R}$ of the form $f(u) = u^T \alpha$ for all $u \in \mathbb{R}^p$ we have

$$\mathbb{E}_{x,y}[(f(x) - y)^2 = 1 + |\alpha - \beta|_2^2.$$

We will denote this quantity by $R(f)$, the risk of f.

(b) We are looking for a decision rule $f : \mathbb{R}^p \to \mathbb{R}$ constructed with the data such that $R(f)$ is small. Simulate a dataset $(x_1, y_1), ...(x_n, y_n)$ as n i.i.d copies of the random variables x, y defined above, with $n = 1000, p = 10000$ and $\beta = (1, 1, ..., 1)$. On this dataset, perform 10-fold cross-validation with the ridge regression estimators $\hat{\beta}^{ridge} = \hat{\beta}_\lambda^{ridge}$ where $\lambda = 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10, 20, 50, 100$ (so that you compute 13 ridge estimators on this dataset). Use *glmnet* to fit the Ridge estimators, but you have to implement the cross-validation logic yourself. Report in a boxplot the training error of these estimators (the average, over the 10 splits, of the estimator on the training data). Report in a boxplot the test error of these estimators (the average, over the 10 splits, of the estimator on the test data). Report in a boxplot the risk of these estimators (the average, over the 10 splits, of the risk). Which parameter yields the smallest test error? Smallest training error? Smallest risk averaged over the 10 splits?

5.*Best Subset Selection.* Problem 8 at page 262-263 of ISL Problem 8: In this exercise,we will generate simulated data, and will then use this data to perform best subset selection.

a). Use the `rnorm()` function to generate a predictor `X` of length `n = 100`, as well as a noise vector ϵ of length `n = 100`.

```
set.seed(2021) #Make sure results are reproducible
X <- rnorm(n = 100)
epsilon <- rnorm(100)
```

b). Generate a response vector `Y' of lengthn` $= 100$‘ according to the model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$$

where $\beta_0, \beta_1, \beta_2, \beta_3$ are constants of your choice

```
beta <- c(1.3, -2.4, 5.6, 5.2)
Y <- beta[1] + beta[2] * X + beta[3] * X^2 + beta[4] * X^3 + epsilon
```

c). Use the `regsubsets()` function to perform best subset selection in order to choose the best model containing the predictors $X, X^2, ..., X^{10}$. What is the best model obtained according to C_p, `BIC` and adjusted R^2? Show some plots to provide evidence
for your answer, and report the coefficients of the best model obtained.
Note, you will need to use the `data.frame()` function to create a single data set containing
both X and Y

```
library(leaps)
library(tidyverse)
```

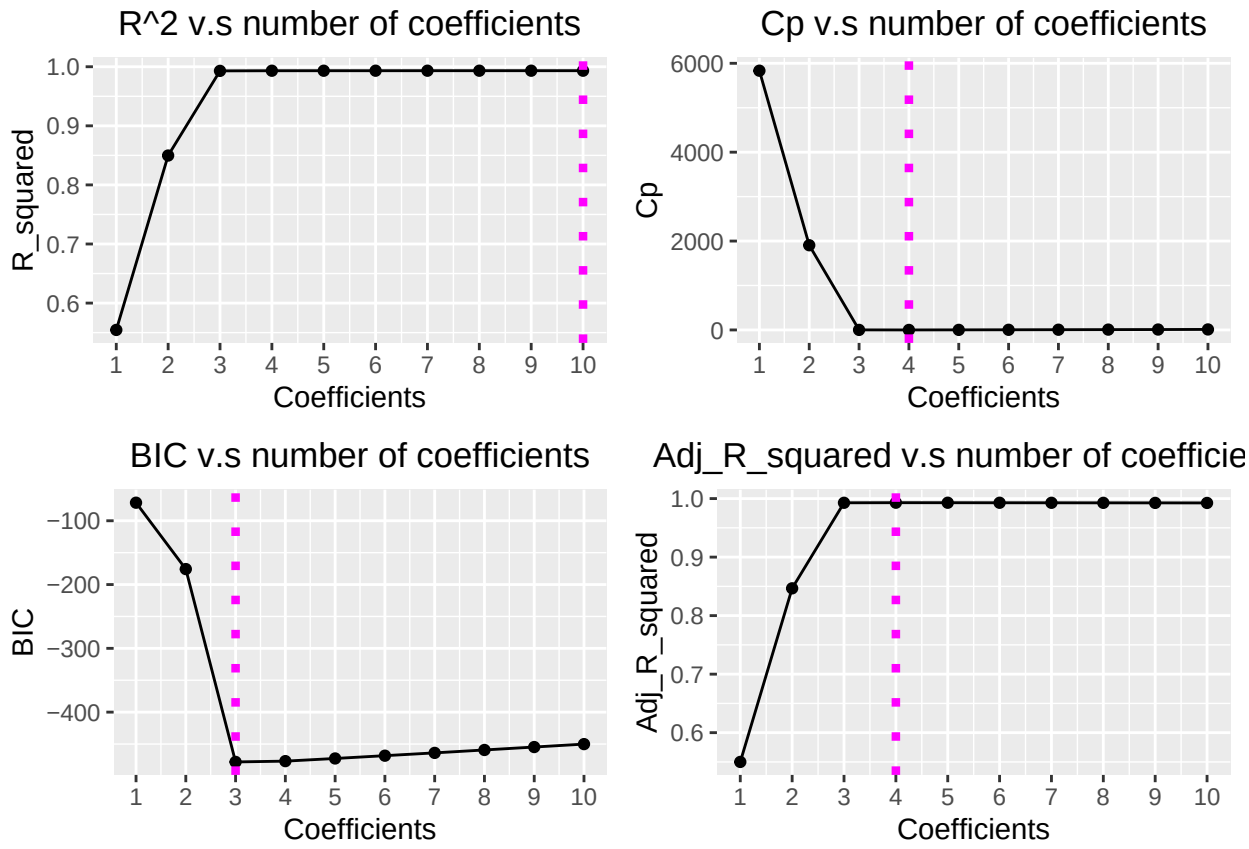
```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.5     v dplyr   1.0.3
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
reg_df =data.frame("X" = X, "Y" = Y)

#Use regsubsets() to perform best subset selection to choose best model containing
#predictors X_1,...,X_10.
regfit.full <- regsubsets(Y~poly(X,10), data=reg_df, nvmax = 10)
reg.summary <- summary(regfit.full)
reg_df =data.frame("X" = X, "Y" = Y)

#Use regsubsets() to perform best subset selection to choose best model containing
#predictors X_1,...,X_10.
regfit.full <- regsubsets(Y~poly(X,10), data=reg_df, nvmax = 10)
reg.summary <- summary(regfit.full)

plot_t <- tibble(Coefficients = 1:10,
                 R_squared = reg.summary$rsq,
                 Adj_R_squared = reg.summary$adjr2,
                 Cp = reg.summary$cp,
```

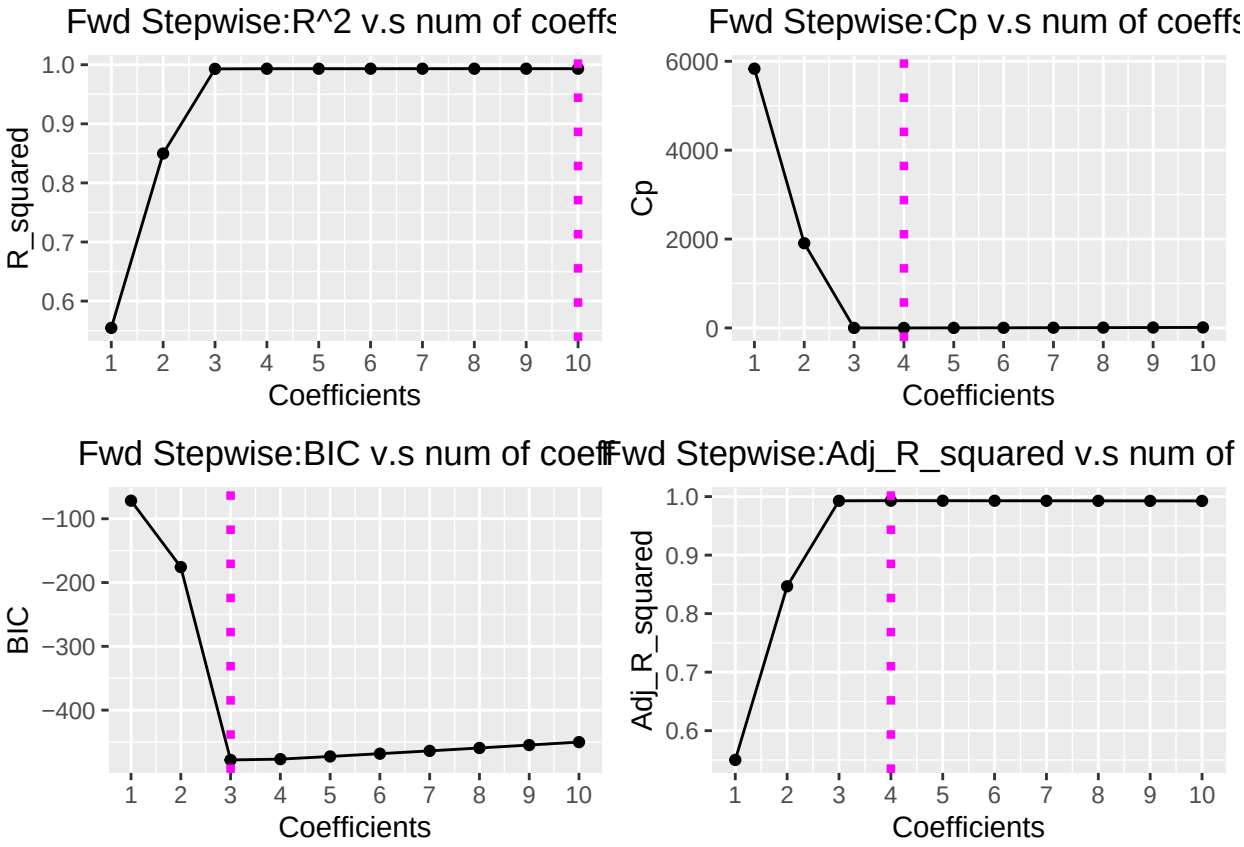
```
                BIC = reg.summary$bic
                )


p1 <- ggplot(data = plot_t, mapping = aes(x = Coefficients, y = R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_t$R_squared), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "R^2 v.s number of coefficients"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

p2 <- ggplot(data = plot_t, mapping = aes(x = Coefficients, y = Cp)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_t$Cp), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Cp v.s number of coefficients"
  ) +
  theme(plot.title = element_text(hjust = 0.5))


p3 <- ggplot(data = plot_t, mapping = aes(x = Coefficients, y = BIC)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_t$BIC), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "BIC v.s number of coefficients"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

p4 <- ggplot(data = plot_t, mapping = aes(x = Coefficients, y = Adj_R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_t$Adj_R_squared), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Adj_R_squared v.s number of coefficients"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

grid.arrange(p1, p2, p3, p4, nrow=2)
```



(d) Repeat (c), using forward stepwise selection and also using backwards stepwise selection. How does your answer compare to the results in (c)?

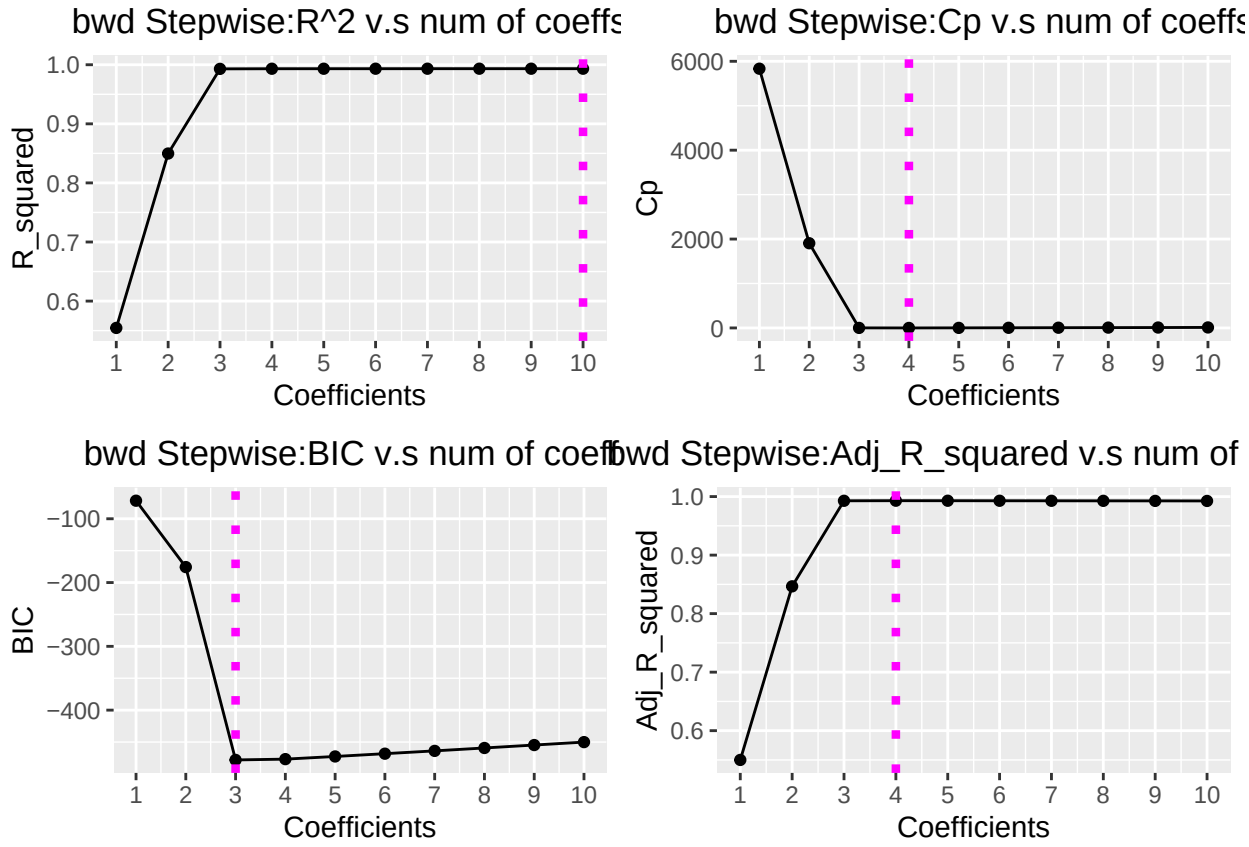
```
regfit.fwd <- regsubsets(Y~poly(X,10), data=reg_df, nvmax = 10, method = "forward")
regfit.fwd.summary <- summary(regfit.fwd)


plot_fwd_t <- tibble(Coefficients = 1:10,
                     R_squared = regfit.fwd.summary$rsq,
                     Adj_R_squared = regfit.fwd.summary$adjr2,
                     Cp = regfit.fwd.summary$cp,
                     BIC = regfit.fwd.summary$bic
)

p1 <- ggplot(data = plot_fwd_t, mapping = aes(x = Coefficients, y = R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_fwd_t$R_squared), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Fwd Stepwise:R^2 v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))
```

```r
p2 <- ggplot(data = plot_fwd_t, mapping = aes(x = Coefficients, y = Cp)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_fwd_t$Cp), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Fwd Stepwise:Cp v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))


p3 <- ggplot(data = plot_fwd_t, mapping = aes(x = Coefficients, y = BIC)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_fwd_t$BIC), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Fwd Stepwise:BIC v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

p4 <- ggplot(data = plot_fwd_t, mapping = aes(x = Coefficients, y = Adj_R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_fwd_t$Adj_R_squared), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "Fwd Stepwise:Adj_R_squared v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

grid.arrange(p1, p2, p3, p4, nrow=2)
```




```
regfit.bwd <- regsubsets(Y~poly(X,10), data=reg_df, nvmax = 10, method = "backward")
regfit.bwd.summary <- summary(regfit.bwd)

plot_bwd_t <- tibble(Coefficients = 1:10,
                     R_squared = regfit.bwd.summary$rsq,
                     Adj_R_squared = regfit.bwd.summary$adjr2,
                     Cp = regfit.bwd.summary$cp,
                     BIC = regfit.bwd.summary$bic
)

p1 <- ggplot(data = plot_bwd_t, mapping = aes(x = Coefficients, y = R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_bwd_t$R_squared), linetype="dotted",
             color = "magenta", size=1.5) +
  labs(
    title = "bwd Stepwise:R^2 v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

p2 <- ggplot(data = plot_bwd_t, mapping = aes(x = Coefficients, y = Cp)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_bwd_t$Cp), linetype="dotted",
```

```
            color = "magenta", size=1.5) +
  labs(
    title = "bwd Stepwise:Cp v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))


p3 <- ggplot(data = plot_bwd_t, mapping = aes(x = Coefficients, y = BIC)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.min(plot_bwd_t$BIC), linetype="dotted",
            color = "magenta", size=1.5) +
  labs(
    title = "bwd Stepwise:BIC v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

p4 <- ggplot(data = plot_bwd_t, mapping = aes(x = Coefficients, y = Adj_R_squared)) +
  scale_x_continuous(breaks = seq(1,10,1)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = which.max(plot_bwd_t$Adj_R_squared), linetype="dotted",
            color = "magenta", size=1.5) +
  labs(
    title = "bwd Stepwise:Adj_R_squared v.s num of coeffs"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

grid.arrange(p1, p2, p3, p4, nrow=2)
```



e). Now fit a lasso model to the simulated data, again using $X, X^2, ..., X^{10}$ as predictors. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates, and discuss the results obtained.

```
library(glmnet)
```

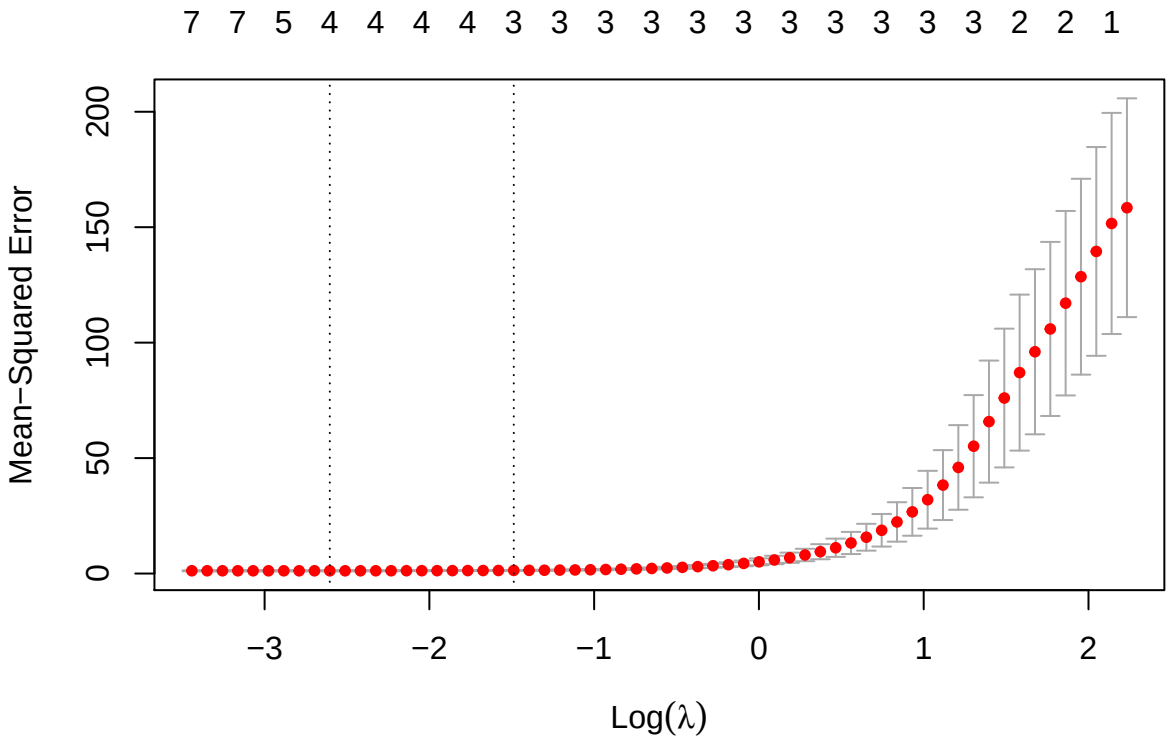
```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1
```

```
X_mat <- model.matrix(Y~poly(X, 10)-1, data = reg_df) #Create a model matrix
cv.out <- cv.glmnet(x = X_mat, y = Y, alpha = 1) #Perform cross validation for lambda
plot(cv.out) #Plot cross-validation as a function of lambda
```




```
best_lambda <- cv.out$lambda.min #Extract the optimal(minimal) lambda
lasso <- glmnet(X_mat, Y, alpha = 1, lambda = best_lambda) #Use glmnet with updated lambda
coef(lasso) #Print the coefficients
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)    5.327361
## poly(X, 10)1  92.584757
## poly(X, 10)2  46.697280
## poly(X, 10)3  67.352595
## poly(X, 10)4   .
## poly(X, 10)5   .
## poly(X, 10)6  -1.159033
## poly(X, 10)7   .
## poly(X, 10)8   .
## poly(X, 10)9   .
## poly(X, 10)10  .
```

f). Now regenerate a response vector Y according to the model

$$Y = \beta_0 + \beta_7 X^7 + \epsilon$$

and perform best subset selection and the lasso. Discuss the results obtained.

```r
beta <- c(beta, 4.2, -3.1,4.8, -2.1)
Y = beta[1] + beta[8]*X^7 + epsilon
reg_df_2 <- tibble(X=X,Y=Y)
reg.fit2 <- regsubsets(Y ~ poly(X, 10), data = reg_df_2, nvmax = 10)
regfit2.summary <- summary(reg.fit2)

plot2_t <- tibble(Coefficients = 1:10,
                  R_squared = regfit2.summary$rsq,
                  Adj_R_squared = regfit2.summary$adjr2,
                  Cp = regfit2.summary$cp,
                  BIC = regfit2.summary$bic
)


X_mat2 <- model.matrix(Y~poly(X, 10)-1, data = reg_df_2) #Create a model matrix
cv.out2 <- cv.glmnet(x = X_mat2, y = Y, alpha = 1) #Perform cross validation for lambda
#plot(cv.out2) #Plot cross-validation as a function of lambda
best_lambda2 <- cv.out$lambda.min #Extract the optimal(minimal) lambda
lasso2 <- glmnet(X_mat2, Y, alpha = 1, lambda = best_lambda2) #Use glmnet with updated lambda
coef(lasso2) #Print the coefficients
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)     16.31832
## poly(X, 10)1  -634.33478
## poly(X, 10)2   278.82394
## poly(X, 10)3  -682.04907
## poly(X, 10)4   114.49832
## poly(X, 10)5  -299.08532
## poly(X, 10)6    20.96581
## poly(X, 10)7   -42.57746
## poly(X, 10)8     .
## poly(X, 10)9     .
## poly(X, 10)10    .
```