borui sun ps3

borui sun

2/8/2020

Conceptual exercises

Training/test error for subset selection

1. (5 points) Generate a data set with $p = 20$ features, $n = 1000$ observations, and an associated quantitative response vector generated according to the model

$$Y = X\beta + \epsilon$$

where β has some elements that are exactly equal to zero.

```
n <- 1000
p <- 20

sample <- data.frame(id = 1:n)

set.seed(233)

for (i in 1:p){
  if (p%%2 == 0){
    x <- rnorm(n, mean = p - 1, sd = p^2) %>% data.frame()
  } else{
    x <- runif(n, min = 0, max = p) %>% data.frame()
  }

  colnames(x) <- paste0("x", i)
  sample <- bind_cols(sample, x)
}

beta <- sample(c(0:1, 20, replace = TRUE))*rnorm(20)

sample$id <- NULL
sample$epsilon <- rnorm(n)
sample$y <- (as.matrix(sample[,1:20]) %*% as.matrix(beta)) + sample$epsilon
```

2. (10 points) Split your data set into a training set containing 100 observations and a test set containing 900 observations.

```
set.seed(233)
split <- initial_split(sample[, -21], prop = 100/n)
train <- training(split)
test <- testing(split)
```

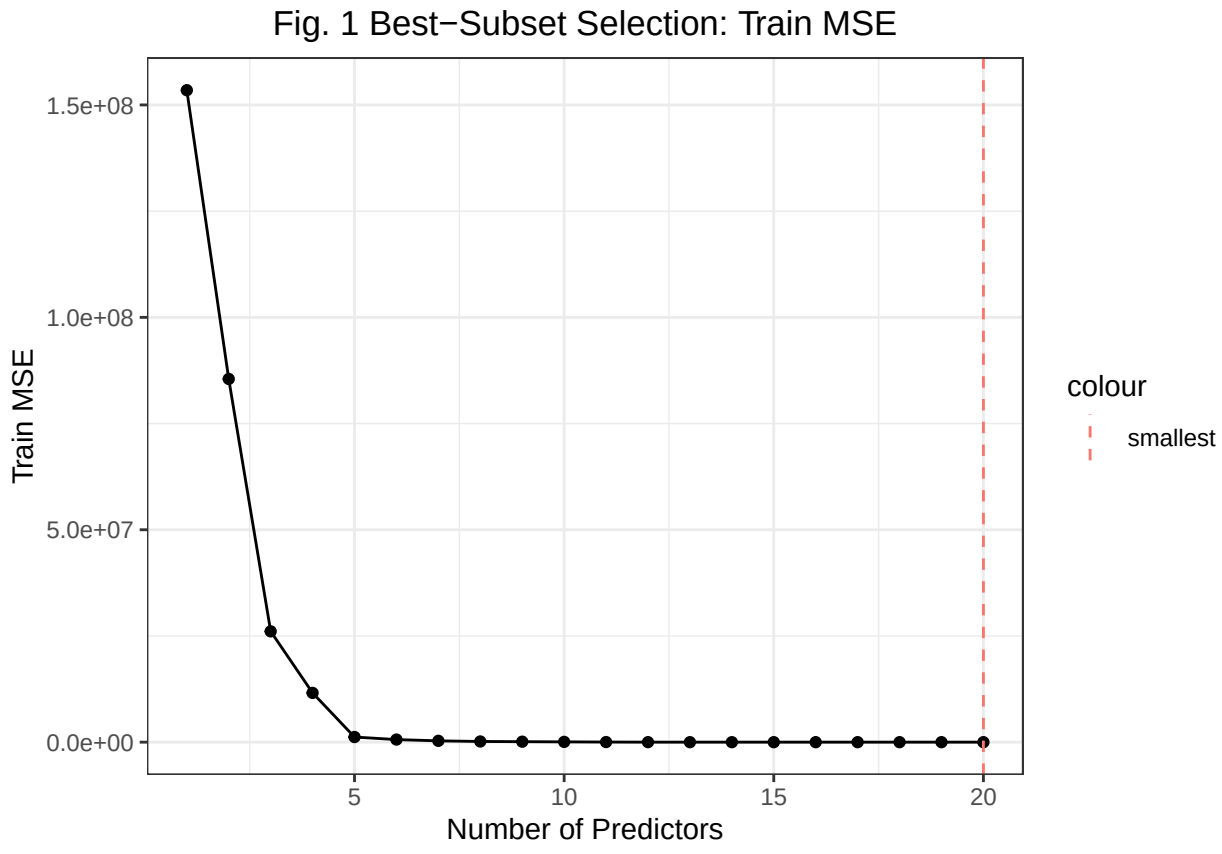
3. (10 points) Perform best subset selection on the training set, and plot the training set MSE associated with the best model of each size. For which model size does the training set MSE take on its minimum value?

As shown in Fig.1, the training set MSE decreases monotonically as the number of predictors increases, but it flattens after the number of predictors reaches 5. The training set MSE is the lowest when model uses all the available predictors, as adding an additional feature can never make MSE worse.

```
best_subset <- regsubsets(y~.,
                          data = train,
                          nvmax = p); results <- summary(best_subset)

train_mse <- tibble(P = 1:p,
                    MSE = results$rss/100)

ggplot(train_mse, aes(x = P, y = MSE)) +
  geom_line() +
  geom_point() +
  geom_vline(aes(
    xintercept = which(train_mse$MSE == min(train_mse$MSE)),
    color = "smallest"), linetype = 2) +
  labs(title = "Fig. 1 Best-Subset Selection: Train MSE",
       x = "Number of Predictors",
       y = "Train MSE")
```

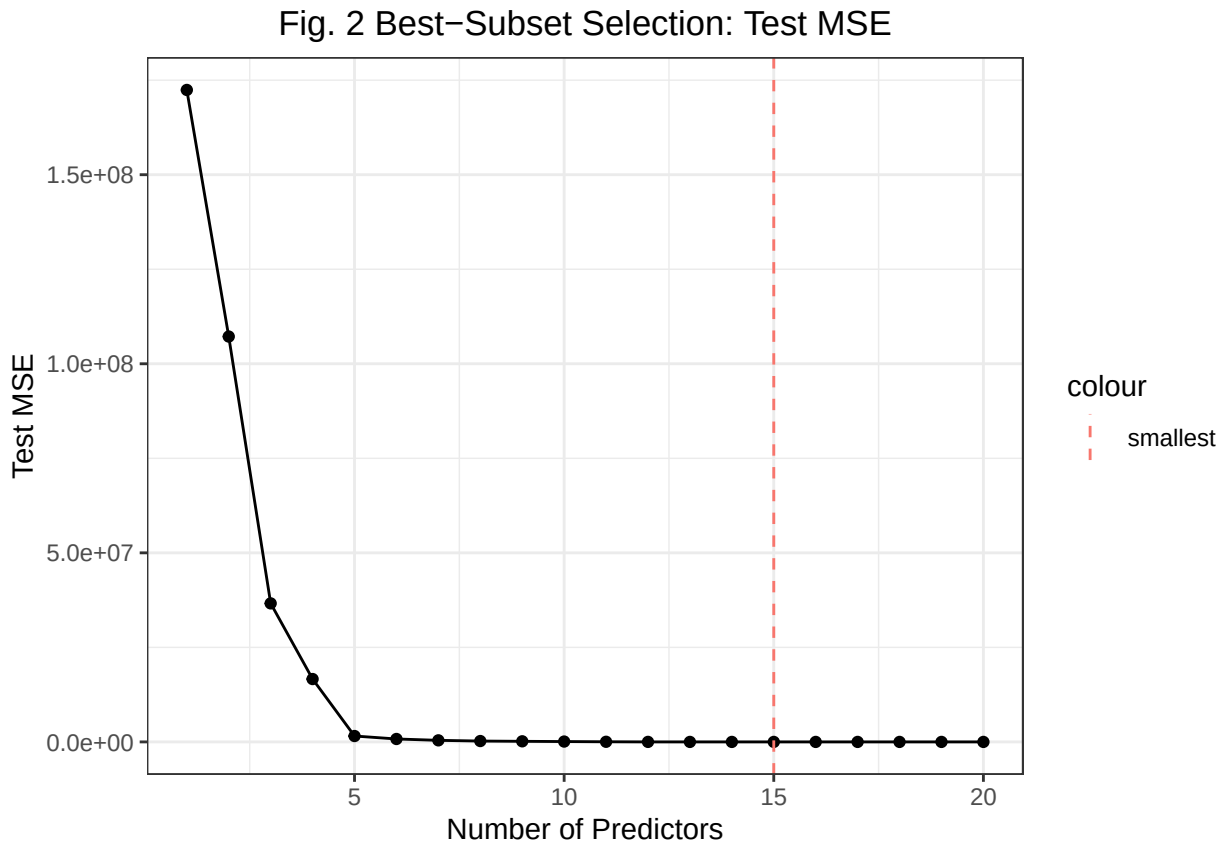


4. (5 points) Plot the test set MSE associated with the best model of each size.

```r
MSE <- function(selection, data, id){
  form <- as.formula(y ~ .)
  mat <- model.matrix(form, data)
  coefi <- coef(selection, id = id)
  xvars <- names(coefi)
  predicted <- as.vector(mat[, xvars] %*% coefi, mode = "double")
  mean((data[["y"]] - predicted)^2)
}

test_mse <- tibble(P = 1:p) %>%
  mutate(MSE = sapply(.$P, function(x) MSE(best_subset, test, id = x)))

ggplot(test_mse, aes(x = P, y = MSE)) +
  geom_line() +
  geom_point() +
  geom_vline(aes(xintercept = which(test_mse$MSE == min(test_mse$MSE)), color = "smallest"), linetype =
  labs(title = "Fig. 2 Best-Subset Selection: Test MSE",
       x = "Number of Predictors",
       y = "Test MSE")
```



5. (5 points) For which model size does the test set MSE take on its minimum value? Comment on your results.

   If it takes on its minimum value for a model containing only an intercept or a model containing all of the features, then play around with the way that you generate the data previously

until you create a data generating process in which the test set MSE is minimized for an intermediate model size.

As shown in Fig2, the test set MSE takes on its minimum value when model has 15 features. Since we know that there are 5 variables with a coefficient of 0 when we generate the data, it is reasonable to expect a model with size of 15 to have the minimum test MSE. Any model with size greater than 15 is overfitting and captures noises from irrelevant features, hence they have low train MSE but relatively high test MSE. On the other hand, any model with size less than 15 is underfitting and fails to include all effective features, hence they have both high train and high test MSE.

6. (10 points) How does the model at which the test set MSE is minimized compare to the true model used to generate the data? Comment on the coefficient sizes.

Looking at the table below, we can see that the model with lowest test MSE from best selection is virtually identical to our true model. The five irrelevant independent variables have a coefficient of exact 0 in both models, and the differences between the 15 parameters in two models are very minimal. None of the differences is greater than 0.0005.

```
coefs <- coef(best_subset, 1:20)%>%
  map_dfr(~as_data_frame(t(.))) %>%
  mutate(model = paste0("P", 1:20))

coefs <- beta %>%
  t() %>%
  as.data.frame() %>%
  `colnames<-`(c(paste0("x",1:20))) %>%
  mutate(model = "true") %>%
  full_join(coefs)
```

```
## Joining, by = c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8", "x9", "x10",
## "x11", "x12", "x13", "x14", "x15", "x16", "x17", "x18", "x19", "x20", "model")
```

```
coefs %>% filter(grepl("true|P15", model)) %>%
  mutate_all(~replace(., is.na(.), 0)) %>%
  t() %>% {.[1:20,]} %>%
  as.data.frame() %>%
  mutate_if(is.factor, ~as.numeric(as.character(.))) %>%
  mutate(diff = abs(V2-V1)) %>%
  `colnames<-`(c("True Model", "Best Selection with 15 Predictors", "Difference")) %>%
  kable()
```

True Model	Best Selection with 15 Predictors	Difference
-32.6957000	-32.6954900	0.0002100
-1.3501990	-1.3502240	0.0000250
0.0000000	0.0000000	0.0000000
0.5886638	0.5891519	0.0004881
-18.8818000	-18.8819200	0.0001200
0.6828131	0.6831658	0.0003527
0.0000000	0.0000000	0.0000000
-0.4277592	-0.4273991	0.0003601
-9.3683300	-9.3684300	0.0001000
-0.5535370	-0.5535566	0.0000196

True Model	Best Selection with 15 Predictors	Difference
0.0000000	0.0000000	0.0000000
0.0314575	0.0315514	0.0000938
10.5688800	10.5686700	0.0002100
0.0935592	0.0934012	0.0001580
0.0000000	0.0000000	0.0000000
0.1178763	0.1178489	0.0000274
20.5088900	20.5085900	0.0003000
2.0217000	2.0219270	0.0002270
0.0000000	0.0000000	0.0000000
1.1592660	1.1594280	0.0001620

7. (10 points) Create a plot displaying

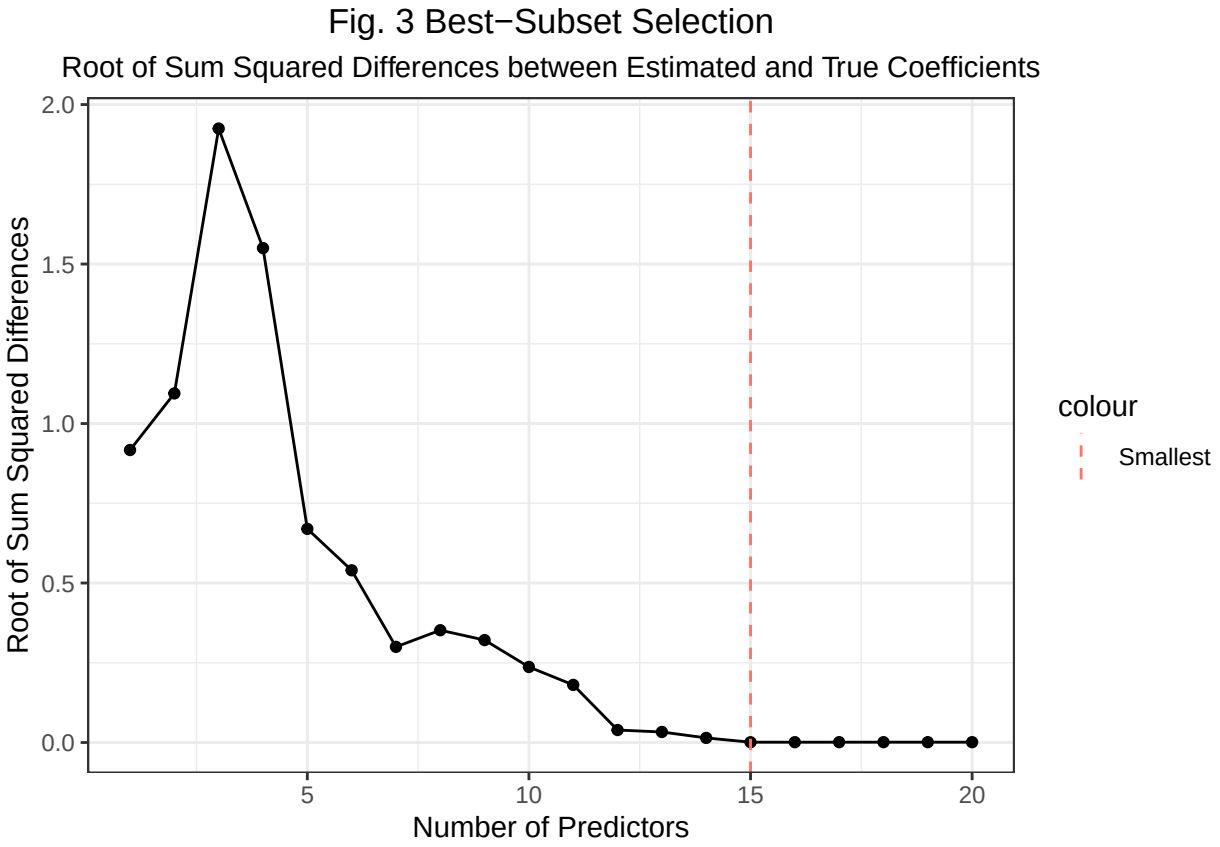
$$\sqrt{\sum_{j=1}^{p}(\beta_j - \hat{\beta}_j^r)^2}$$

for a range of values of r, where $\hat{\beta}_j^r$ is the jth coefficient estimate for the best model containing r coefficients. Comment on what you observe. How does this compare to the test MSE plot?

As shown in Fig.3, model with 15 predictors has the smallest root of sum squared difference bewteen estimated and true coefficients. Although the root of sum squared difference plot identifies the same best model as the test MSE plot did in Q.4, the two plots follow different trajectories. The test MSE seems to decrease monotonically at first, and then reaches a plateau after model size hits 5. However,as the model size increases, the root of sum squared difference plot rises at first, then decreases and reaches a plateau after model size hits 12. Model with size 3 have largest root of sum squared difference but not the largest test MSE.

```
coef_root_sum_diff <- apply(coefs[2:21, 1:20], 1, function(x) (coefs[1, 1:20]- x)^2) %>%
  unlist() %>%
  matrix(nrow = 20, byrow = TRUE) %>%
  as.data.frame() %>%
  apply(1, function(x) sqrt(sum(x, na.rm = TRUE))) %>%
  as.data.frame()%>%
  `colnames<-`("sum_diff") %>%
  mutate(P = 1:20)

ggplot(coef_root_sum_diff, aes(x = P, y = sum_diff)) +
  geom_line() +
  geom_point() +
  geom_vline(aes(
    xintercept = which(coef_root_sum_diff$sum_diff == min(coef_root_sum_diff$sum_diff)),
    color = "Smallest"), linetype = 2) +
  labs(title = "Fig. 3 Best-Subset Selection",
       subtitle = "Root of Sum Squared Differences between Estimated and True Coefficients",
       x = "Number of Predictors",
       y = "Root of Sum Squared Differences")
```



Application exercises

1. (10 points) Fit a **least squares linear** model on the training set, and report the test MSE.

```
ols <- lm(egalit_scale~., data = gss_train)
ols_mse <- mean((gss_test$egalit_scale - predict(ols, gss_test))^2)
cat("OLS Test MSE:", ols_mse)
```

```
## OLS Test MSE: 63.21363
```

2. (10 points) Fit a **ridge** regression model on the training set, with λ chosen by 10-fold cross-validation. Report the test MSE.

```
train.x <- model.matrix(egalit_scale ~ ., gss_train)[, -1]
train.y <- gss_train$egalit_scale

test.x <- model.matrix(egalit_scale ~ ., gss_test)[, -1]
test.y <- gss_test$egalit_scale

set.seed(233)


ridge <- cv.glmnet(
  x = train.x,
```

```
  y = train.y,
  alpha = 0,
  nfolds = 10,
)

ridge_mse <- mean((test.y - predict(ridge, s = ridge$lambda.min, newx = test.x))^2)

cat("Ridge Regression Test MSE:", ridge_mse)
```

```
## Ridge Regression Test MSE: 61.03781
```

3. (10 points) Fit a **lasso** regression on the training set, with λ chosen by 10-fold cross-validation. Report the test MSE, along with the number of non-zero coefficient estimates.

```
set.seed(233)

lasso <- cv.glmnet(
  x = train.x,
  y = train.y,
  alpha = 1,
  nfolds = 10,
)

lasso_mse <- mean((test.y - predict(lasso, s = lasso$lambda.min, newx = test.x))^2)

cat("LASSO Regression Test MSE:", lasso_mse, "\n",
    "Number OF Non-zero Estimates:",
    lasso$nzero[which(lasso$lambda == lasso$lambda.min)])
```

```
## LASSO Regression Test MSE: 61.19366
##  Number OF Non-zero Estimates: 32
```

4. (10 points) Fit an **elastic net** regression model on the training set, with α and λ chosen by 10-fold cross-validation. That is, estimate models with $\alpha = 0, 0.1, 0.2, \ldots, 1$ using the same values for λ across each model. Select the combination of α and λ with the lowest cross-validation MSE. *For that combination*, report the test MSE along with the number of non-zero coefficient estimates.

```
tuning_grid <- tibble(alpha = seq(0, 1, by = .1))

set.seed(233)

fold_id <- sample(1:10, size = length(train.y), replace = TRUE)
# maintain the same folds across all models

for(i in seq_along(tuning_grid$alpha)) {
  # fit CV model for each alpha value
  fit <- cv.glmnet(train.x,
                   train.y,
                   alpha = tuning_grid$alpha[i],
                   foldid = fold_id)

  # extract MSE and lambda values
```

```
  tuning_grid$mse_min[i]    <- fit$cvm[fit$lambda == fit$lambda.min]
  tuning_grid$lambda_min[i] <- fit$lambda.min
}

lowest <- which.min(tuning_grid$mse_min)
elnet <- cv.glmnet(train.x,
                   train.y,
                   alpha = tuning_grid$alpha[lowest],
                   foldid = fold_id)
elnet_mse <- mean((test.y - predict(elnet, s = tuning_grid$lambda_min[lowest], newx = test.x))^2)

cat("Elastic Net Regression Test MSE:", elnet_mse, "\n",
    "Number OF Non-zero Estimates:",
    elnet$nzero[which(elnet$lambda == elnet$lambda.min)])
```

```
## Elastic Net Regression Test MSE: 61.27006
##  Number OF Non-zero Estimates: 28
```

5. (5 points) Comment on the results obtained. How accurately can we predict an individual's egalitarianism? Is there much difference among the test errors resulting from these approaches?

According to the table below, our model's predictions of individuals' egaliarianism are on average approximately 8 units (sqrt of MSE) off the true values. Considering that our target ranges from between 1 and 35, the discrepancy is quite large. Our predictions are not very precise. Since we only use a simple linear regression, even though with up to 77 regressors, its explanatory power is relatively weak. The adjusted r-squared is 0.3714407, indicating that only less than 40% of variance in the target are explained by our model. It is possible that the underlying relationship is not linear.

There isn't much difference among the test erros resulting from the four appraoches. We also do not observe a strong improvement in our predictions by adopting shrinkage methods. Regularization on average improves the test MSE of OLS by 2. While Ridge model has the lowest test MSE, the differences between the three regularization methods are quite small (less than 0.3). The purpose of reguarization is to avoid overfitting, the little reduction in test MSE indicates that the OLS model may not suffer from a high risk of overfitting.

```
tibble(Model = c("OLS", "Ridge", "Lasso", "Elastic Net"),
       Test_MSE = c(ols_mse, ridge_mse, lasso_mse, elnet_mse),
       NonZero_Coefficients = c(77, 77, 32, 28)) %>% kable()
```

Model	Test_MSE	NonZero_Coefficients
OLS	63.21363	77
Ridge	61.03781	77
Lasso	61.19366	32
Elastic Net	61.27006	28