Statistics 452: Statistical Learning and Prediction

Chapter 6, Part 4: Considerations in High Dimensions

Brad McNeney

2017-10-07

High-Dimensional Data

- The "dimension" of the regression/prediction problem is the number of predictors, p, not the sample size, n.
- Low-dimensional: Traditional statistical inference was designed for $n \gg p$.
- When $n \approx p$, these methods are subject to over-fitting, or may fail.
- High-dimensional: Data collection technologies are allowing more predictors to be measured on observational units, but the number of units is still limited, leading to $p \gg n$.
- High-dimensional data example: Genome-wide association studies.
  - It is now possible to type about 1 million single nucleotide polymorphisms (SNPs) across the genome, but study sizes are still in the 1000's. $p \approx 1,000,000 \gg n \approx 2000$.

What Goes Wrong When p Large?

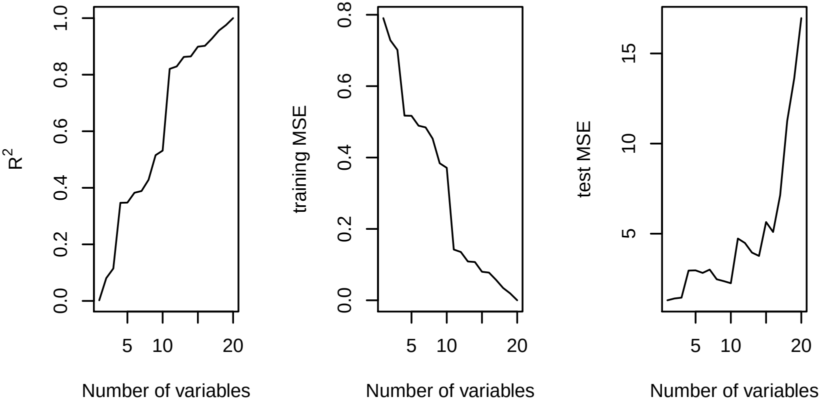
- In a word: overfitting.
- Replicate the experiment summarized in Figure 6.23 of the text.
  - A response Y on $n = 20$ subjects, completely unrelated to predictors $X_1, \ldots, X_{20}$.

```
set.seed(1)
R2 <- function(fit,TSS) { RSS <- sum(fit$residuals^2); 1-RSS/TSS }
n <- 20
y <- rnorm(n)
X <- matrix(NA,nrow=n,ncol=n)
for(i in 1:n) {
  X[,i] <- rnorm(n)
} # Equivalent to X <- matrix(rnorm(n*n),nrow=n,ncol=n)
fit <- lm.fit(X,y)
TSS <- sum((y-mean(y))^2)
R2(fit,TSS)
```

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

```r
# Simulate training data
Xtest <- matrix(rnorm(n*n),n,n)
ytest <- rnorm(n)
# Utility function to predict response based on lm.fit object
predict.lm.fit <- function(X,fit) { X%*%fit$coefficients }
R2vec <- trainMSE <- testMSE <- rep(NA,n) # vectors to hold our results
# Loop over different numbers of predictors
for(p in 1:n) {
  fit <- lm.fit(X[,1:p,drop=FALSE],y)
  R2vec[p] <- R2(fit,TSS)
  trainMSE[p] <- mean(fit$residuals^2)
  yhattest <- predict.lm.fit(Xtest[,1:p,drop=FALSE],fit)
  testMSE[p] <- mean((ytest-yhattest)^2)
}
```

```
opar <- par(mfrow=c(1,3))
plot(1:p,R2vec,xlab="Number of variables",ylab=expression(R^2),type="l")
plot(1:p,trainMSE,xlab="Number of variables",ylab="training MSE",type="l")
plot(1:p,testMSE,xlab="Number of variables",ylab="test MSE",type="l")
```

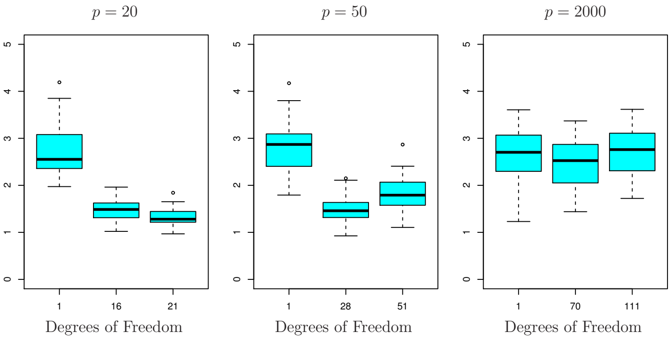


```
par(opar)
```

High-Dimensional Regression ($n \gg p$)

- When $p > n$ least squares has no unique solution.
- Other methods, such as forward selection, Lasso and PCR are useful.
- These approaches avoid overfitting by being less flexible than least squares.
- However, they also perform poorly if most of the predictors are noise.

Illustration



- Text, Fig. 6.24: Lasso with $n = 100$ and 20 predictors associated with response. Boxplots show test MSE for three values of tuning λ that lead to different numbers of non-zero coefficients (df).
    - When $p = 20$, no regularization (incl. all 21 terms) is best.
    - When $p = 50$, use a moderate amount of regularization.
    - When $p = 2000$ even the lasso is overwhelmed by the noise, and no amount of regularization works.

Curse of Dimensionality

- Notice that the test set error actually increases as the proportion of noise predictors increases.
- Conclusion: Adding noise predictors leads to a deterioration in the fitted model.

Collinearity in High Dimensions

- Collinearity becomes severe in high dimensions.
  - Just as combinations of noise predictor variables can predict y, so can they predict other predictors.
  - E.G., 19 noise predictors explain about 60% of the variation in the 20th in our simulated data example

```r
fit <- lm.fit(X[,1:19],X[,20])
TSS20 <- sum((X[,20]-mean(X[,20]))^2)
R2(fit,TSS20)
```

```
## [1] 0.592862
```

- When there are many predictors, a selected set is one of many that predicts the resonse.