SLP week 1 - Answers to exercises

ISLR Exercise 1

a) The sample size n is extremely large, and the number of predictors p is small.

Flexible method probably better. There is a low number of predictors, so no 'curse of dimensionality'; extremely large sample size, so probably enough information in the data to reliably estimate $f(x)$.

b) The number of predictors p is extremely large, and the number of observations n is small.

Inflexible method probably better. Extremely large number of predictors, so 'curse of dimensionality' applies; small sample size, so probably too little information in the data to reliably estimate $f(x)$ with many parameters / complex $f(x)$.

c) The relationship between the predictors and response is highly non-linear.

Flexible method probably better: need flexible method to deal with non-linear association.

d) The variance of the error terms, i.e. $\sigma^2 = Var(\epsilon)$, is extremely high.

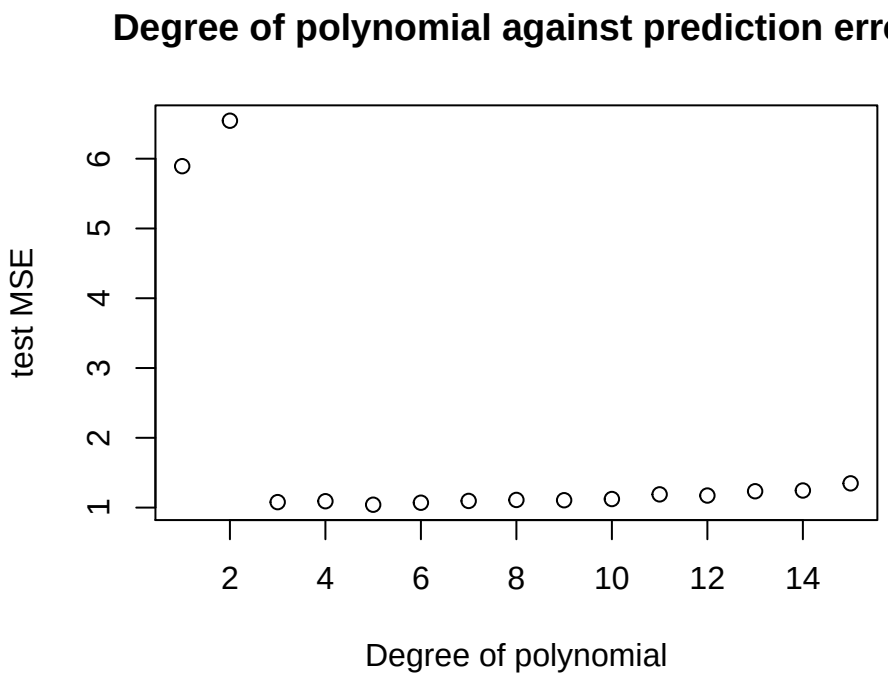
Inflexible method probably better. Flexible method may overfit on irreducible errors (i.e., deviation between predictions of true model and actual y). If sample size is large enough, though, an flexible method may still be able to recover accurate prediction function.

Exercise: Nonlinear Regression (from slides)

- Generate a training and test set (size 100) of data with a single predictor X (uniformly distributed from -5 till 5) following $f(x) = x + 8sin(x/2)$.
- Fit polynomial regression models to the training data of degree 1 to 15, make predictions on the test set and compute the prediction error.
- Plot the degree of the polynomial against prediction error on the test set.

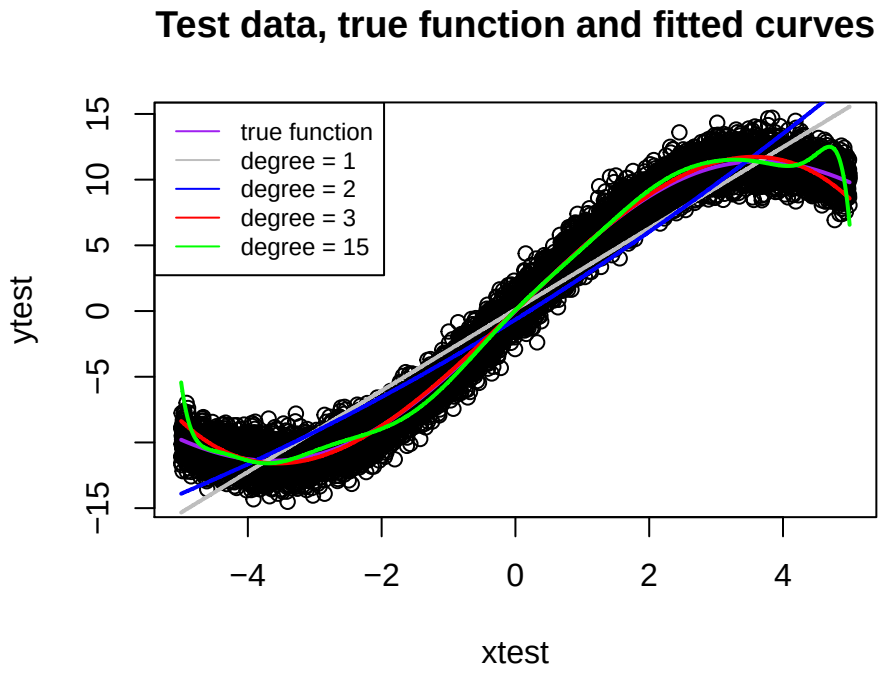
```
set.seed(1234)
n <- 100
sd_epsilon <- 1

# generate training data:
x <- runif(n, min = -5, max = 5)
y <- x + 8*sin(x/2) + rnorm(n, sd = sd_epsilon)
train <- data.frame(x, y)

# generate test data:
xtest <- runif(10000, min = -5, max = 5)
ytest <- xtest + 8*sin(xtest/2) + rnorm(10000, sd = sd_epsilon)
test <- data.frame(x = xtest, y = ytest)

fit <- pred <- err <- list()

for (order in 1:15) {
```

```r
  fit[[order]] <- lm(y ~ poly(x, degree = order), data = train)
  pred[[order]] <- predict(fit[[order]], newdata = test)
  err[[order+1]] <- mean((ytest - pred[[order]])^2)
}
plot(1:15, unlist(err), main = "Degree of polynomial against prediction error",
     xlab = "Degree of polynomial", ylab = "test MSE")
```



The MSE is depicted against the degree of the polynomial where we see that the MSE sharply decreases from degree 2 till 3 and increases again when the degree becomes larger than 3. Taking into account the irreducible error (which has a variance of 1), the cubic model does much better than the linear model.

```r
plot(xtest, ytest, main = "Test data, true function and fitted curves")
curve(x + 8*sin(x/2), add = TRUE, lwd = 2, col = "purple")
lines(sort(xtest), pred[[1]][order(xtest)], col = "grey", lwd = 2)
lines(sort(xtest), pred[[2]][order(xtest)], col = "blue", lwd = 2)
lines(sort(xtest), pred[[3]][order(xtest)], col = "red", lwd = 2)
lines(sort(xtest), pred[[15]][order(xtest)], col = "green", lwd = 2)
legend("topleft", legend = c("true function", paste("degree =", c(1, 2, 3, 15))),
       lty = 1, col = c("purple", "grey", "blue", "red", "green"), cex = .75)
```



The linear and quadratic clearly stand out, failing to capture non-linearities. The other curves follow closely the true conditional means, although the higher-order polynomials show aberrant behaviour at the boundaries and may adjust to the training data too much.