# 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.

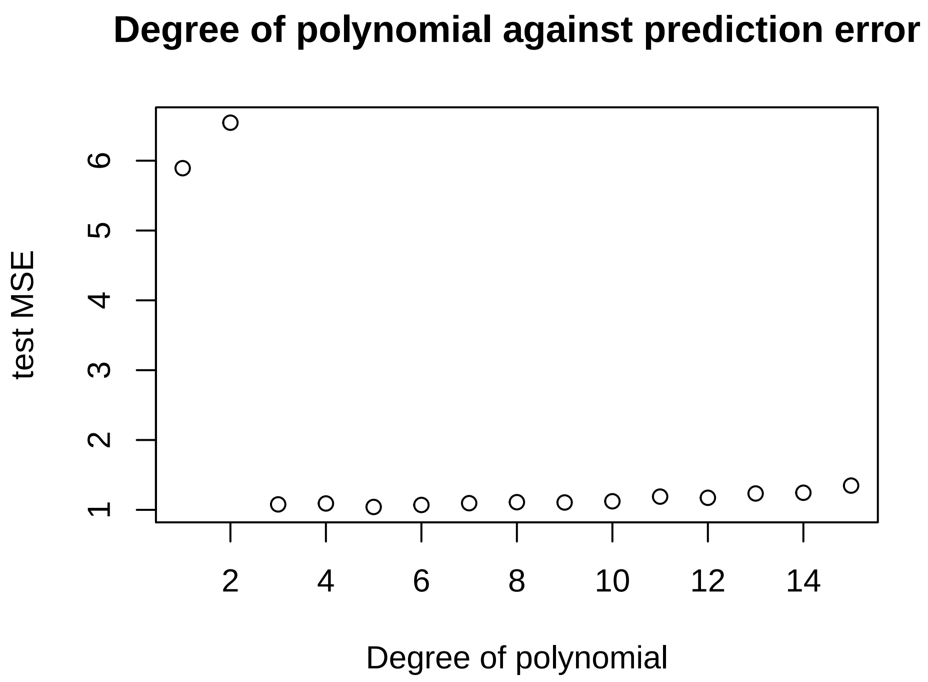
   Inflexible method probably better. Flexible method may overfit on irreducible errors (i.e., deviation between predictions of true model and actual $y$). If sample size is large enough, though, a flexible method may still be able to recover accurate prediction function.

## 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.

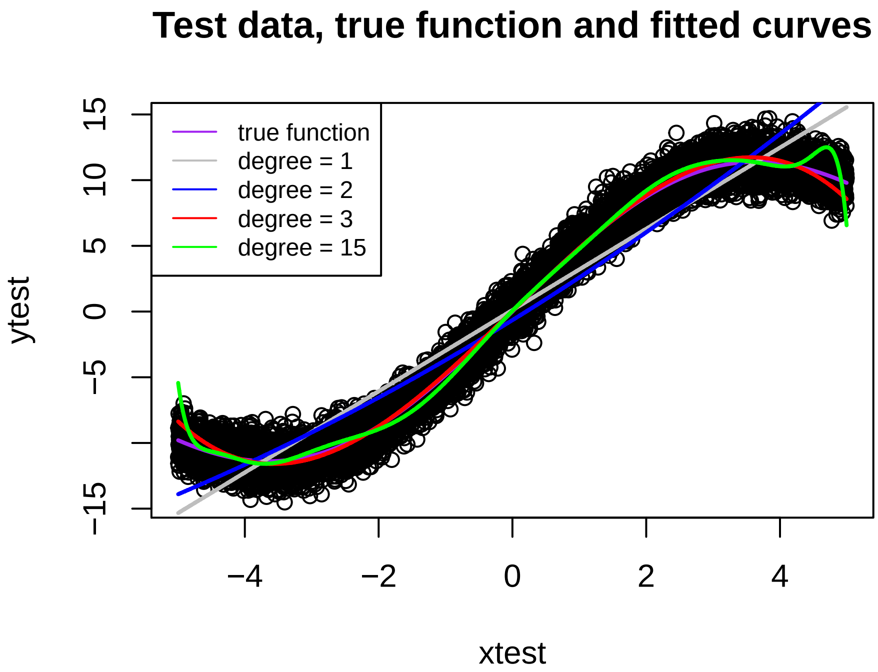
```r
set.seed(1234)
n <- 100
sd_epsilon <- 1

# 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) {
  fit[[order]] <- lm(y ~ poly(x, degree = order), data = train)
  pred[[order]] <- predict(fit[[order]], newdata = test)
  err[[order]] <- 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 and quadratic model.

```
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. This aberrant behavior will worsen when there are > 1 predictors.

## Exercise Nearest Neighbours (from slides)

With both nearest neighbour regression and classification, for making a prediction on a new observation, we would use information from the $K$ nearest training datapoints. For regression, the predicted value would be the mean of the response values for those datapoints. For classification, we would count the number of datapoints belonging to each class, yielding multiple probabilities; the new observation would be predicted to belong to the class for which the highest probability was obtained.

Note that, instead of taking the majority class, we could also take the class proportions of the nearest neighbours as the predictions. This would be more similar to nearest neighbour regression.

## ISLR Exercise 7

a)

```r
obs <- 1:6
x1 <- c(0, 2, 0, 0, -1, 1)
x2 <- c(3, 0, 1, 1, 0, 1)
x3 <- c(0, 0, 3, 2, 1, 1)
color <- c("red", "red", "red", "green" ,"green", "red")
## New observation has x1 = x2 = x3 = 0, compute distances:
dist <- (x1 - 0)^2 + (x2 - 0)^2 + (x3 - 0)^2
data.frame(obs, x1, x2, x3, color, dist)
```

```
##   obs x1 x2 x3 color dist
## 1   1  0  3  0   red    9
## 2   2  2  0  0   red    4
## 3   3  0  1  3   red   10
## 4   4  0  1  2 green    5
## 5   5 -1  0  1 green    2
## 6   6  1  1  1   red    3
```

b) With $K = 1$, our prediction will be "green", because the new observation is closest to a green observation (observation 5).

c) With $K = 3$, our prediction will be "red", because of the three closest observations, one is green (observation 5) and two are red (observations 6 and 2).

d) If the Bayes decision boundary is highly nonlinear, I would expect lower values of $K$ to yield better predictive accuracy. But lower values of $K$ yield higher variance, unless the irreducible error is (very) small or sample size is (very) large. If the decision boundary is highly nonlinear and irreducible error is substantial, it would be better to have a large sample size and use a higher value of $K$.