```{r, warning=FALSE}
library(ISLR2)
library(caret)
library(splines)
```

```{r, warning=FALSE}
data = Boston
dis = data$dis
nox = data$nox
set.seed(666)
```

#a
```{r, warning=FALSE}
model1 = lm(nox ~ poly(dis ,degree = 3))
model1
plot(dis,nox,col = "blue")
points(dis, fitted.values(model1), col = "red")
```

#b
```{r, warning=FALSE}
rss = c()
for (i in (1:10)){
  modelx = lm(nox ~ poly(dis, degree = i))
  rss[i] = sum(residuals(modelx)^2)
}
rss
```

#c
```{r, warning=FALSE}
folds = createFolds(nox, k = 5)
mse_results = c()
for (i in 1:15) {
  fold_mse = c()
  for (j in 1:5) {
    train_indices = unlist(folds[-j])
    test_indices = unlist(folds[j])
    modelx = lm(nox[train_indices] ~ poly(dis[train_indices], degree = i))
    predictions = predict(modelx, data.frame(dis[test_indices]))
    fold_mse[j] = mean((predictions - nox[test_indices])^2)
  }
  mse_results[i] = mean(fold_mse)
}
mse_results
plot((1:15), mse_results,xlab='Degree')
```

#d
```{r, warning=FALSE}
bs_s = bs(dis,df = 4)
model2 = lm(nox ~ bs_s)
model2
knots = attr(bs_s, "knots")
knots
predictions = predict(model2)
plot(dis, nox, main = "B-Spline Fitting", pch = 16, col = "lightgray")
points(dis, predictions, col = "blue", lwd = 2)
abline(v = knots, col = 'red',lty = 2)
```

#e
```{r, warning=FALSE}
rss = c()
par(mfrow = c(3, 4))
```

```
for (i in (1:12)){
  bs_s = bs(dis,df = i)
  modelx = lm(nox ~ bs_s)
  rss[i] = sum(residuals(modelx)^2)
  predictions = predict(modelx)
  knots = attr(bs_s, "knots")
  plot(dis, nox, main = "B-Spline Fitting", pch = 16, col = "lightgray")
  points(dis, predictions, col = "blue", lwd = 2)
  abline(v = knots, col = 'red',lty = 2)
}
rss
```

#f
```{r, warning=FALSE}
par(mfrow = c(1, 1))
folds = createFolds(nox, k = 5)
mse_results = c()
for (i in 1:15) {
  fold_mse = c()
  for (j in 1:5) {
    train_indices = unlist(folds[-j])
    test_indices = unlist(folds[j])
    bs_s = bs(dis[train_indices],df = i)
    modelx = lm(nox[train_indices] ~ bs_s)
    predictions = predict(modelx, data.frame(dis[test_indices]))
    fold_mse[j] = mean((predictions - nox[test_indices])^2)
  }
  mse_results[i] = mean(fold_mse)
}
mse_results
plot((1:15), mse_results,xlab='Degree')
```