

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

ЗВІТ
до індивідуального завдання №6
з дисципліни «Моделі статистичного навчання»

Виконав
студент групи ПМіМ-12:
Зелінський Олександр

Перевірив:
Проф. Заболоцький Т. М.

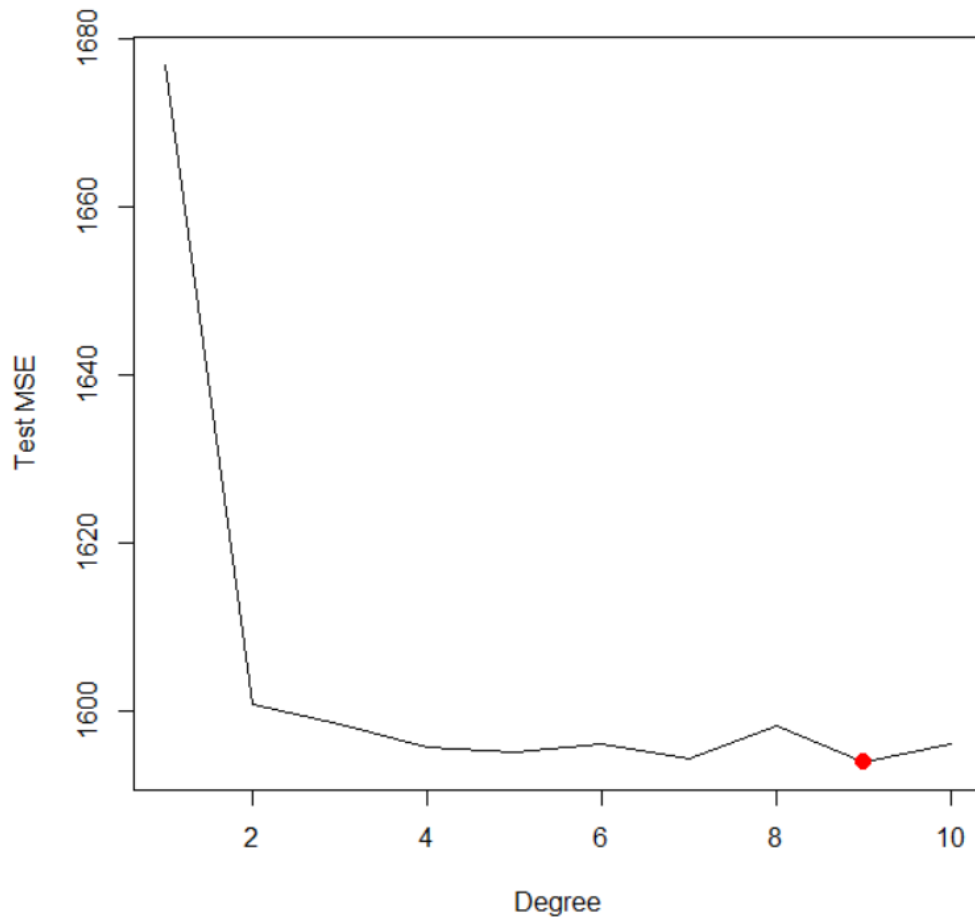
Львів – 2021

Хід виконання

1. Wage

1.1

```
> library(ISLR)
> library(boot)
> set.seed(1)
>
> deltas = rep(NA, 10)
> for (i in 1:10) {
+   fit = glm(wage ~ poly(age, i), data = Wage)
+   deltas[i] = cv.glm(Wage, fit, K = 10)$delta[1]
+ }
> plot(1:10, deltas, xlab = "Degree", ylab = "Test MSE", type = "l")
> d.min = which.min(deltas)
> points(which.min(deltas), deltas[which.min(deltas)], col = "red", cex = 2, pch = 20)
```



Зважаючи на результат можемо побачити, що $d=9$ є оптимальним степенем для полінома.

```

> fit1 = lm(wage ~ age, data = Wage)
> fit2 = lm(wage ~ poly(age, 2), data = Wage)
> fit3 = lm(wage ~ poly(age, 3), data = Wage)
> fit4 = lm(wage ~ poly(age, 4), data = Wage)
> fit5 = lm(wage ~ poly(age, 5), data = Wage)
> fit6 = lm(wage ~ poly(age, 6), data = Wage)
> fit7 = lm(wage ~ poly(age, 7), data = Wage)
> fit8 = lm(wage ~ poly(age, 8), data = Wage)
> fit9 = lm(wage ~ poly(age, 9), data = Wage)
> fit10 = lm(wage ~ poly(age, 10), data = Wage)
> anova(fit1, fit2, fit3, fit4, fit5, fit6, fit7, fit8, fit9, fit10)
Analysis of Variance Table

Model 1: wage ~ age
Model 2: wage ~ poly(age, 2)
Model 3: wage ~ poly(age, 3)
Model 4: wage ~ poly(age, 4)
Model 5: wage ~ poly(age, 5)
Model 6: wage ~ poly(age, 6)
Model 7: wage ~ poly(age, 7)
Model 8: wage ~ poly(age, 8)
Model 9: wage ~ poly(age, 9)
Model 10: wage ~ poly(age, 10)

```

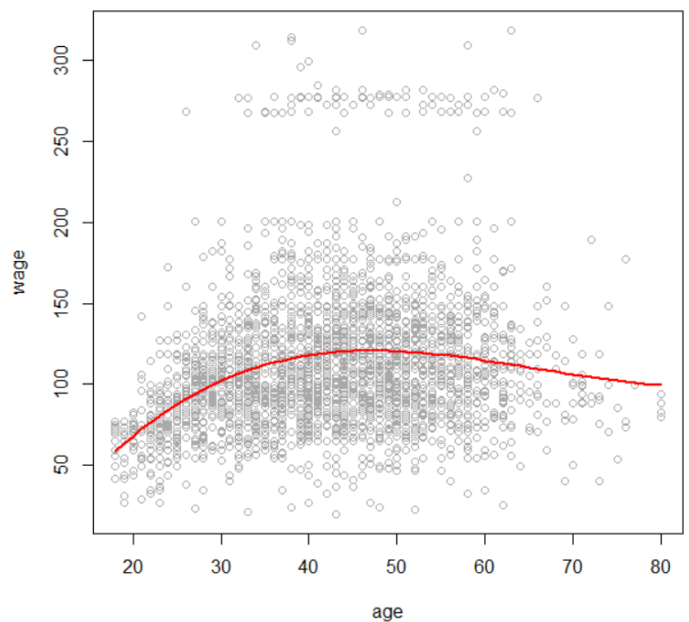
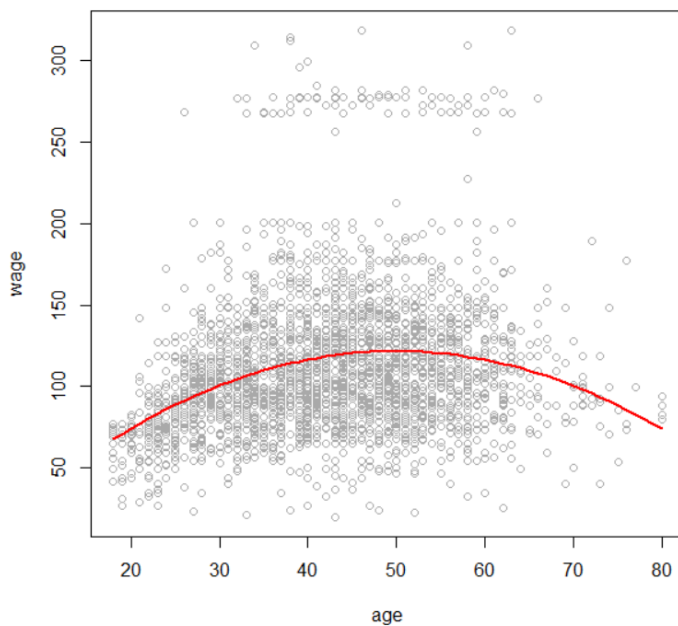
	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	2998	5022216				
2	2997	4793430	1	228786	143.7638	< 2.2e-16 ***
3	2996	4777674	1	15756	9.9005	0.001669 **
4	2995	4771604	1	6070	3.8143	0.050909 .
5	2994	4770322	1	1283	0.8059	0.369398
6	2993	4766389	1	3932	2.4709	0.116074
7	2992	4763834	1	2555	1.6057	0.205199
8	2991	4763707	1	127	0.0796	0.777865
9	2990	4756703	1	7004	4.4014	0.035994 *
10	2989	4756701	1	3	0.0017	0.967529

```

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

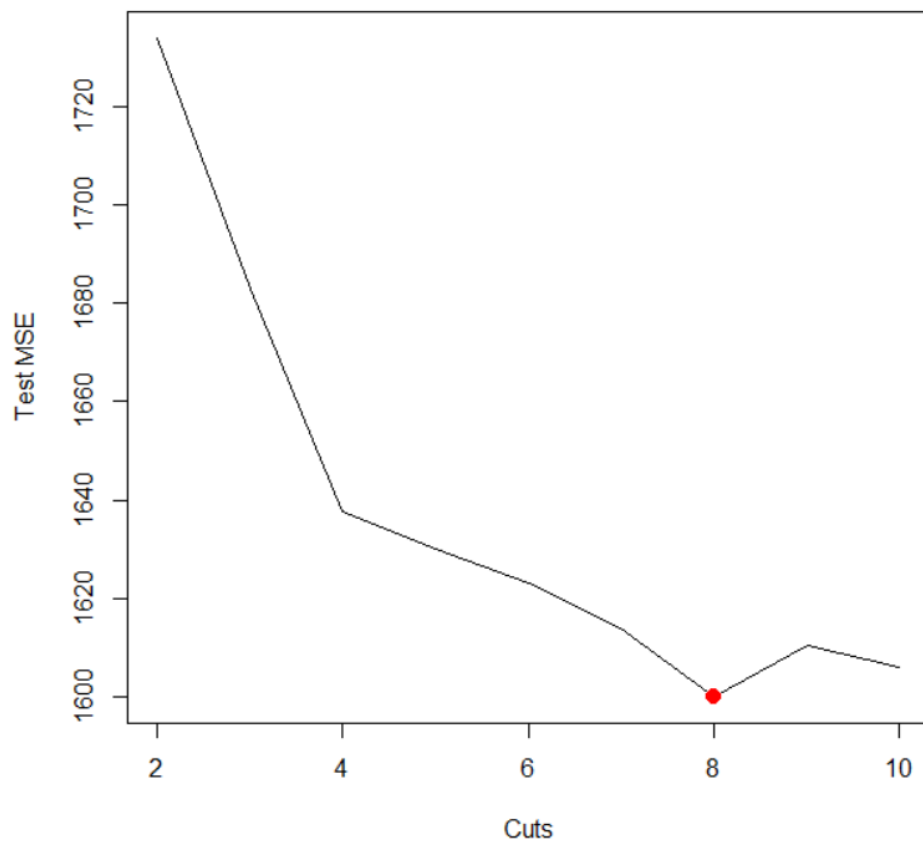
```

З результатів видно, що зважаючи на р-значення, ми можемо побачити, що поліном 2 і 3 степеня забезпечує найкращий результат. Побудуємо ці два поліноми.



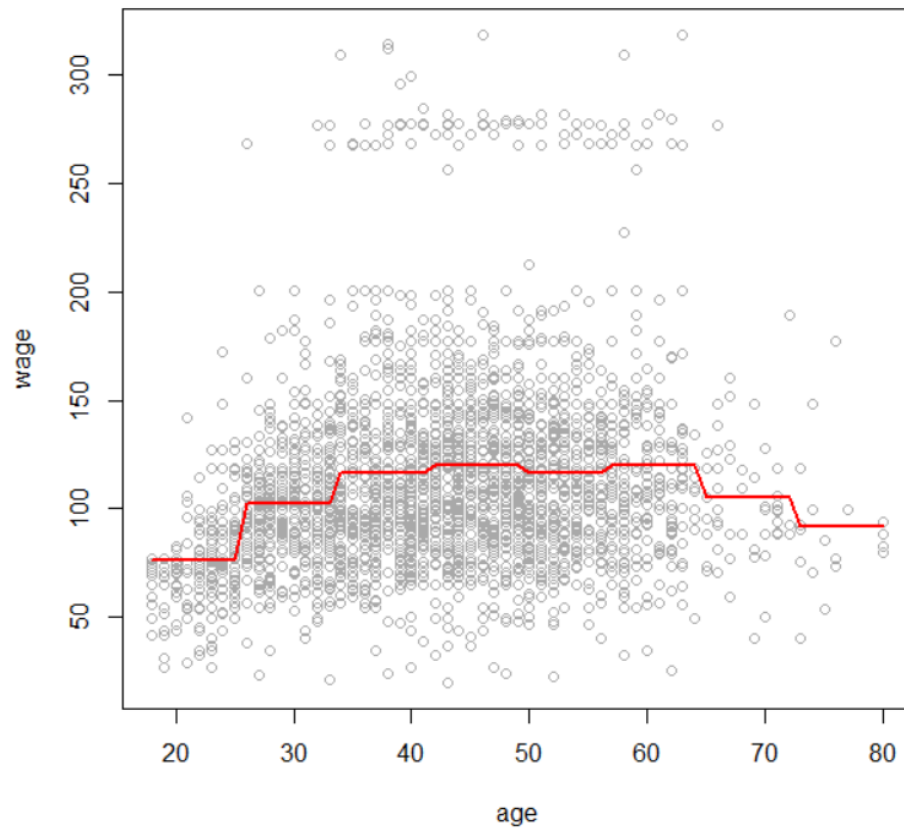
1.2

```
> cvs = rep(NA, 10)
> for (i in 2:10) {
+   Wage$age.cut = cut(Wage$age, i)
+   fit = glm(wage ~ age.cut, data = Wage)
+   cvs[i] = cv.glm(Wage, fit, K = 10)$delta[1]
+ }
> plot(2:10, cvs[-1], xlab = "Cuts", ylab = "Test MSE", type = "l")
> d.min = which.min(cvs)
> points(which.min(cvs), cvs[which.min(cvs)], col = "red", cex = 2, pch = 20)
```



Зважаючи на результат можемо побачити, що помилка буде мінімальною для 8 зрізів.

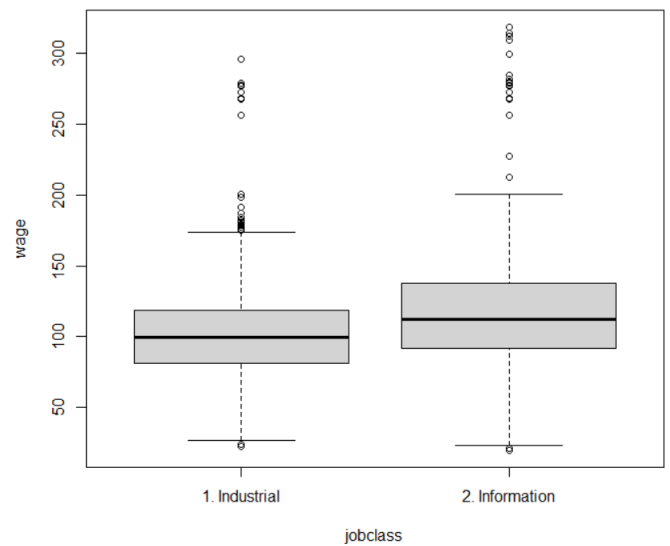
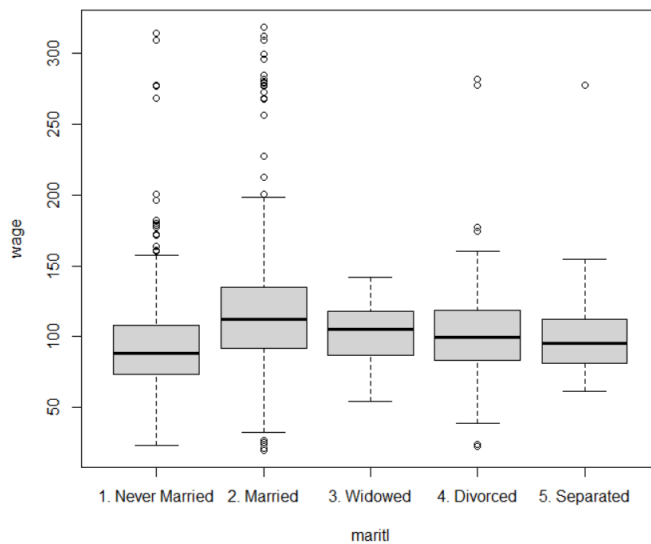
```
> plot(wage ~ age, data = Wage, col = "darkgrey")
> agelims = range(Wage$age)
> age.grid = seq(from = agelims[1], to = agelims[2])
> fit = glm(wage ~ cut(age, 8), data = Wage)
> preds = predict(fit, data.frame(age = age.grid))
> lines(age.grid, preds, col = "red", lwd = 2)
```



2. Wage

```
> summary(Wage$maritl)
1. Never Married      2. Married      3. Widowed      4. Divorced      5. Separated
      648              2074              19              204              55

> summary(Wage$jobclass)
1. Industrial 2. Information
      1544      1456
.
```

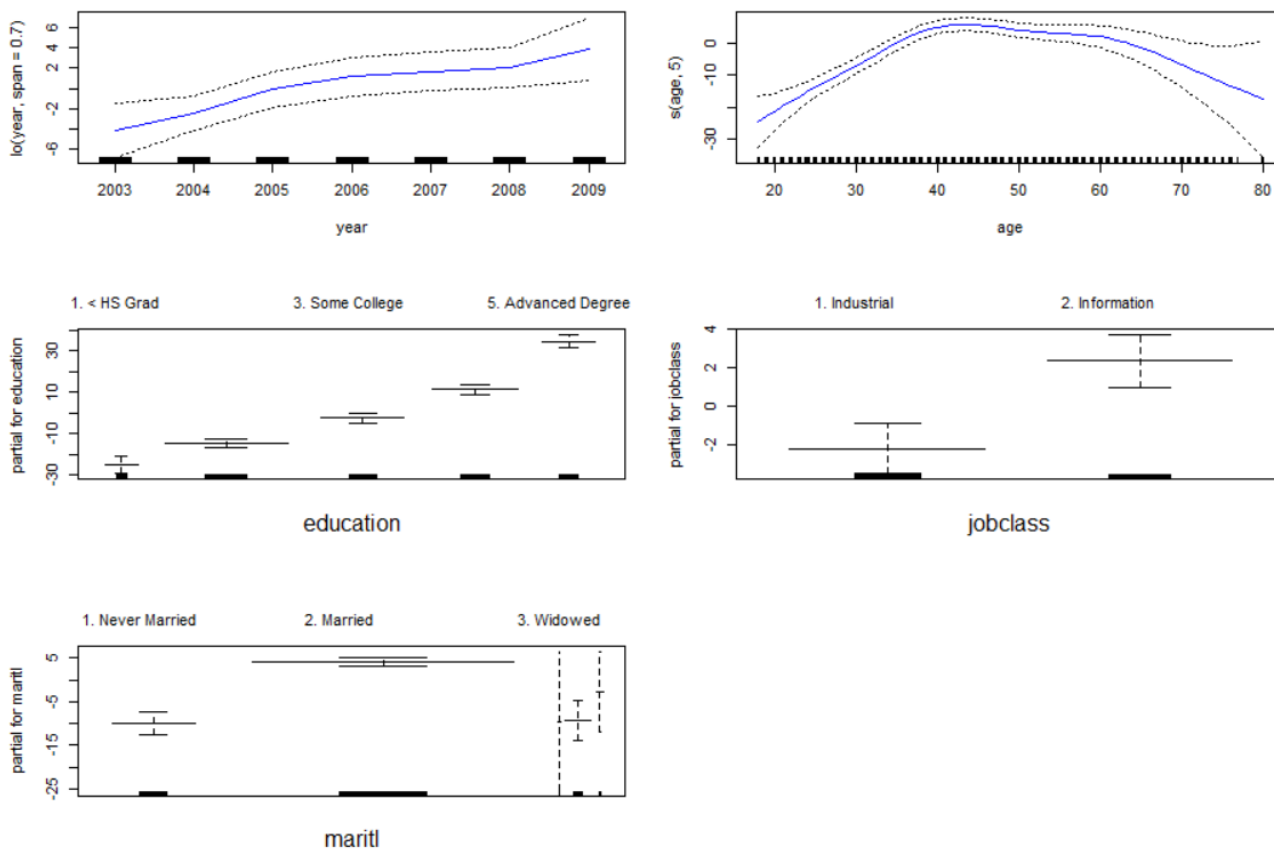


Отже, можна зробити висновок, що подружжя в середньому заробляє більше грошей ніж решта категорій, а також, що працівник в інформаційній сфері в середньому заробляє більше ніж в індустріальній.

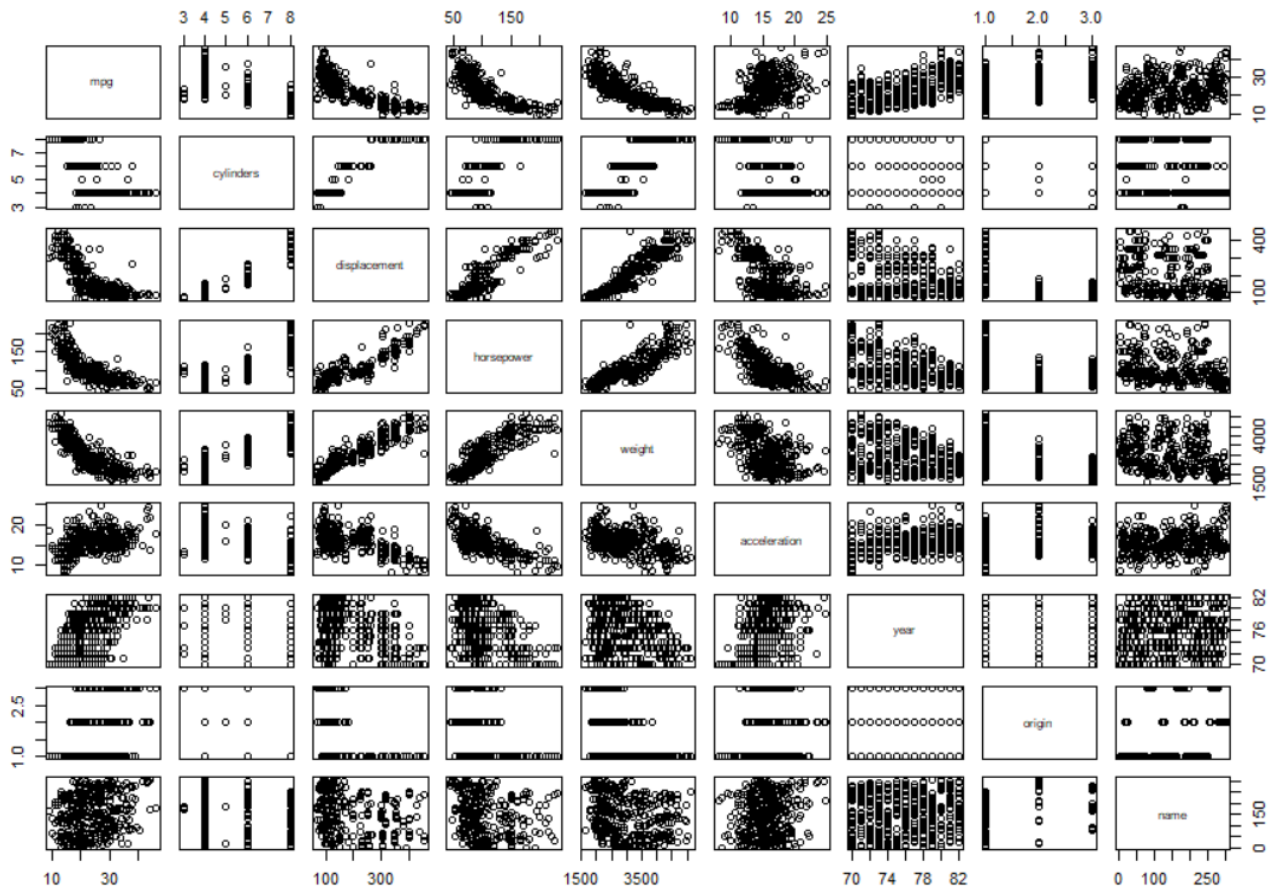
```
> fit0 = gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education, data = Wage)
> fit1 = gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass, data = Wage)
> fit2 = gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + maritl, data = Wage)
> fit3 = gam(wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass + maritl, data = Wage)
> anova(fit0, fit1, fit2, fit3)
Analysis of Deviance Table
```

```
Model 1: wage ~ lo(year, span = 0.7) + s(age, 5) + education
Model 2: wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass
Model 3: wage ~ lo(year, span = 0.7) + s(age, 5) + education + maritl
Model 4: wage ~ lo(year, span = 0.7) + s(age, 5) + education + jobclass +
maritl
      Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       2987.1    3691855
2       2986.1    3679689  1     12166 0.0014637 **
3       2983.1    3597526  3     82163 9.53e-15 ***
4       2982.1    3583675  1     13852 0.0006862 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

З результатів видно, що найкраще підходить третя модель.



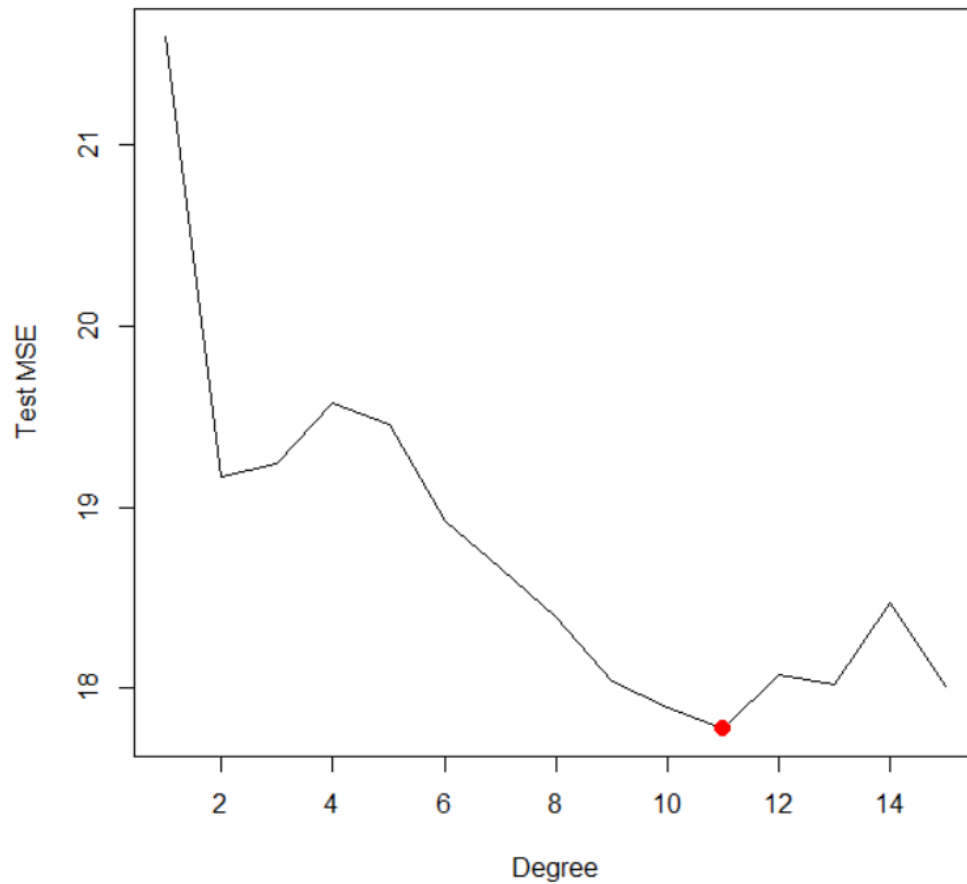
3. Auto



З графіків пар можна легко побачити, що mpg негативно корелює з cylinders, displacement, horsepower та weight.

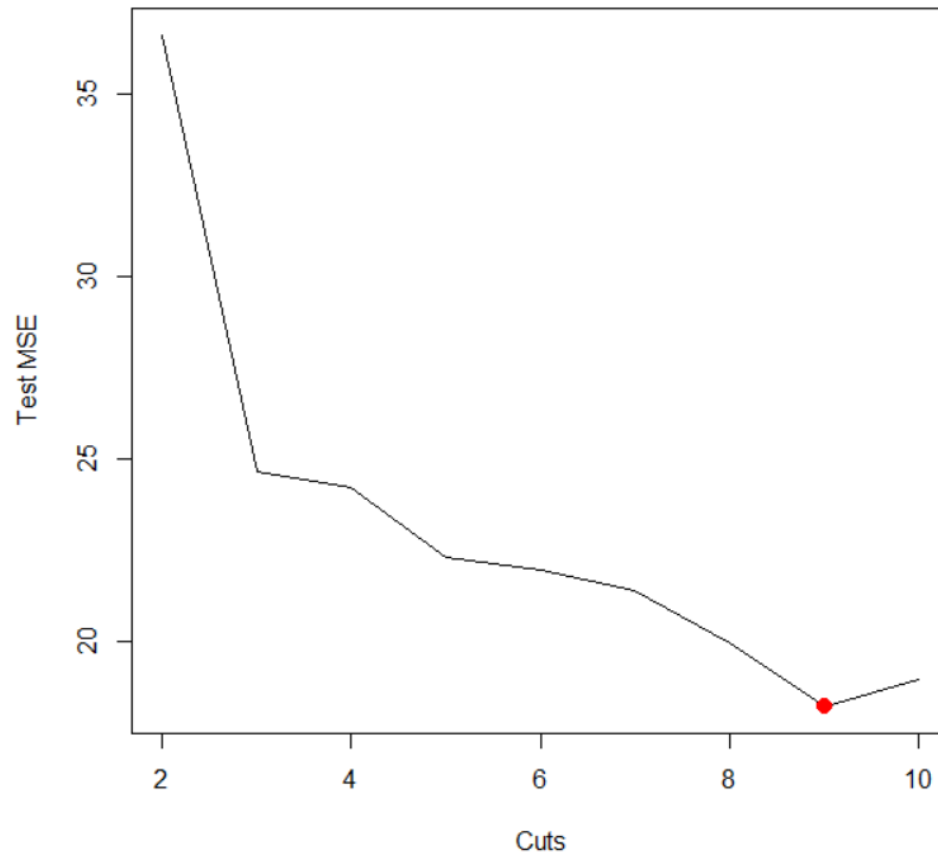
```
> deltas = rep(NA, 15)
> for (i in 1:15) {
+   fit = glm(mpg ~ poly(displacement, i), data = Auto)
+   deltas[i] = cv.glm(Auto, fit, K = 10)$delta[1]
+ }
> plot(1:15, deltas, xlab = "Degree", ylab = "Test MSE", type = "l")
> d.min = which.min(deltas)
> points(which.min(deltas), deltas[which.min(deltas)], col = "red", cex = 2, pch = 20)
```

Зважаючи на графік наведений нижче можемо побачити, що d=1 є оптимальним степенем для полінома.



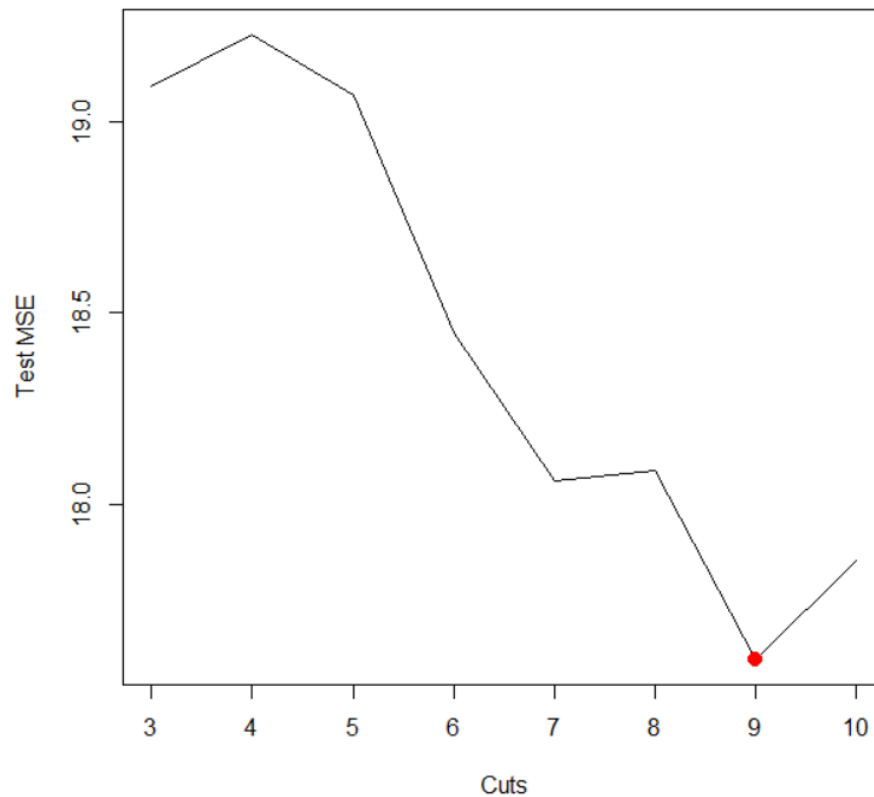
```
> cvs = rep(NA, 10)
> for (i in 2:10) {
+   Auto$dis.cut = cut(Auto$displacement, i)
+   fit = glm(mpg ~ dis.cut, data = Auto)
+   cvs[i] = cv.glm(Auto, fit, K = 10)$delta[1]
+ }
> plot(2:10, cvs[-1], xlab = "Cuts", ylab = "Test MSE", type = "l")
> d.min = which.min(cvs)
> points(which.min(cvs), cvs[which.min(cvs)], col = "red", cex = 2, pch = 20)
```

Зважаючи на графік наведений нижче можемо побачити, що помилка буде мінімальною для 9 зрізів.



```
> library(splines)
> cvs = rep(NA, 10)
> for (i in 3:10) {
+   fit = glm(mpg ~ ns(displacement, df = i), data = Auto)
+   cvs[i] = cv.glm(Auto, fit, K = 10)$delta[1]
+ }
> plot(3:10, cvs[-c(1, 2)], xlab = "Cuts", ylab = "Test MSE", type = "l")
> d.min = which.min(cvs)
> points(which.min(cvs), cvs[which.min(cvs)], col = "red", cex = 2, pch = 20)
```

Зважаючи на графік наведений нижче, можемо побачити, що похибка мінімальна для 9 ступенів свободи.



```
> fit = gam(mpg ~ s(displacement, 4) + s(horsepower, 4), data = Auto)
> summary(fit)
```

Call: gam(formula = mpg ~ s(displacement, 4) + s(horsepower, 4), data = Auto)

Deviance Residuals:

Min	1Q	Median	3Q	Max
-11.2982	-2.1592	-0.4394	2.1247	17.0946

(Dispersion Parameter for gaussian family taken to be 15.3543)

Null Deviance: 23818.99 on 391 degrees of freedom
 Residual Deviance: 5880.697 on 382.9999 degrees of freedom
 AIC: 2194.05

Number of Local Scoring Iterations: NA

Anova for Parametric Effects

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
s(displacement, 4)	1	15254.9	15254.9	993.524	< 2e-16 ***
s(horsepower, 4)	1	1038.4	1038.4	67.632	3.1e-15 ***
Residuals	383	5880.7	15.4		

 Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Anova for Nonparametric Effects

	Npar	Df	Npar F	Pr(F)
(Intercept)				
s(displacement, 4)	3	13.613	1.863e-08	***
s(horsepower, 4)	3	15.606	1.349e-09	***

 Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

4. Boston

4.1

```
> library(MASS)
> set.seed(1)
> fit = lm(nox ~ poly(dis, 3), data = Boston)
> summary(fit)
```

Call:
lm(formula = nox ~ poly(dis, 3), data = Boston)

Residuals:

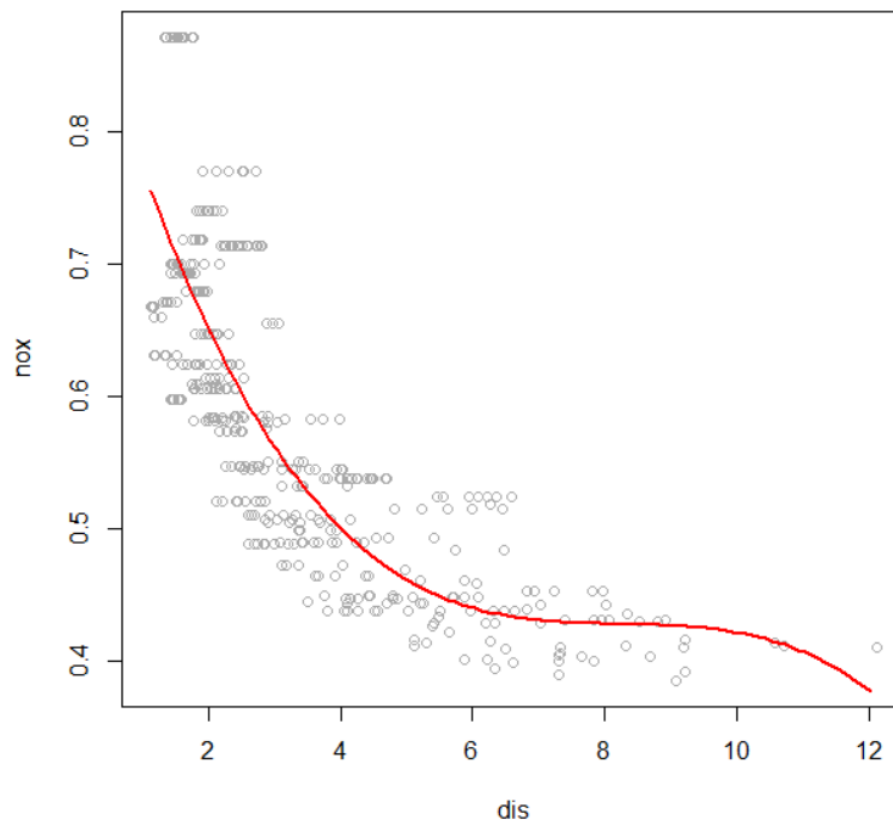
Min	1Q	Median	3Q	Max
-0.121130	-0.040619	-0.009738	0.023385	0.194904

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	0.554695	0.002759	201.021	< 2e-16 ***
poly(dis, 3)1	-2.003096	0.062071	-32.271	< 2e-16 ***
poly(dis, 3)2	0.856330	0.062071	13.796	< 2e-16 ***
poly(dis, 3)3	-0.318049	0.062071	-5.124	4.27e-07 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

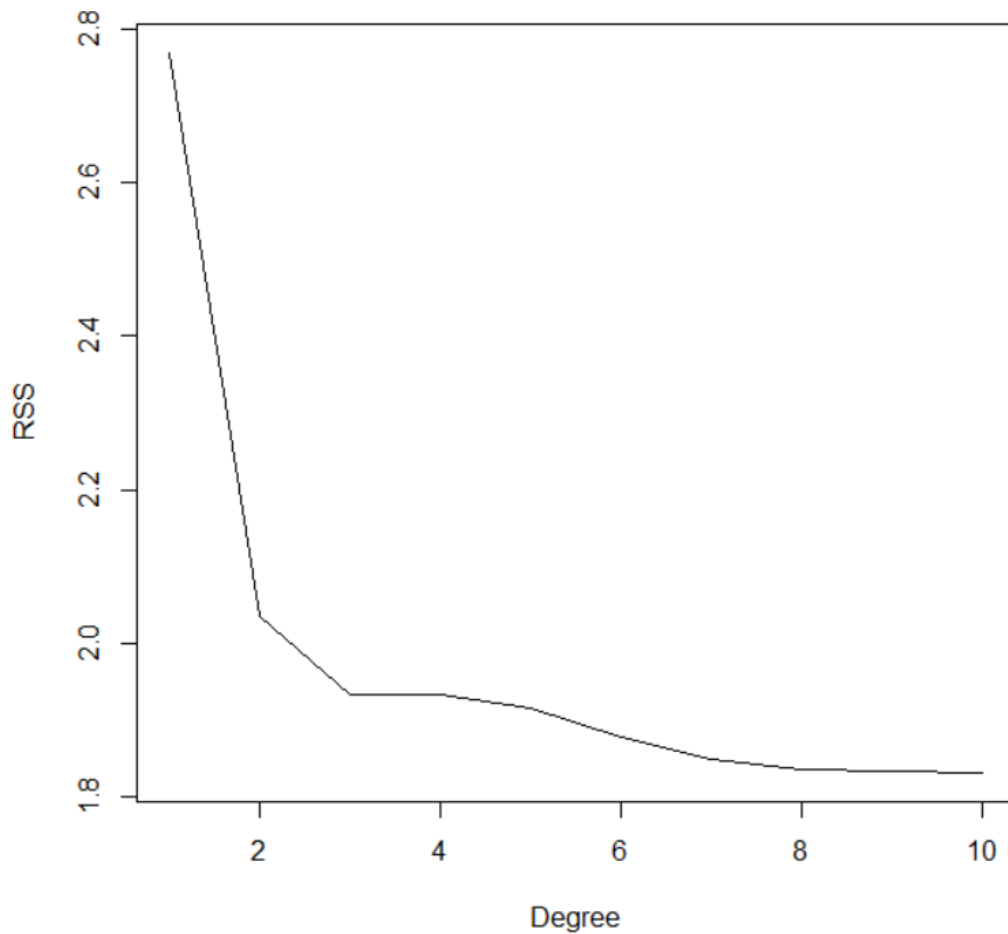
Residual standard error: 0.06207 on 502 degrees of freedom
Multiple R-squared: 0.7148, Adjusted R-squared: 0.7131
F-statistic: 419.3 on 3 and 502 DF, p-value: < 2.2e-16



З наведених вище результатів можна сказати, що всі доданки в поліномі є значущими.

4.2

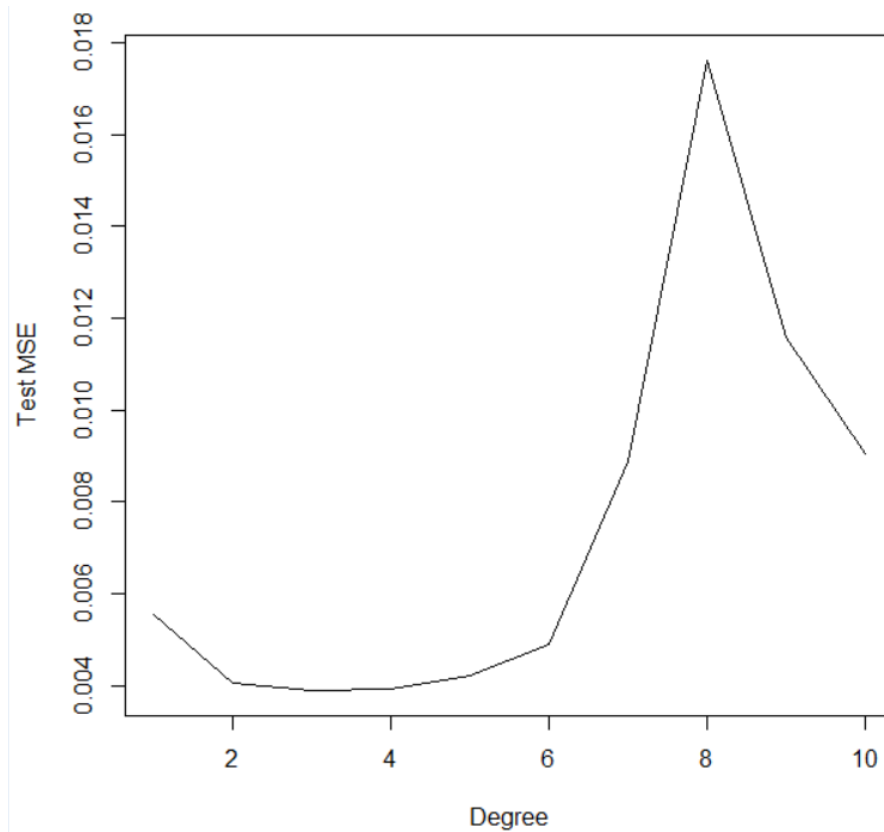
```
> rss = rep(NA, 10)
> for (i in 1:10) {
+   fit = lm(nox ~ poly(dis, i), data = Boston)
+   rss[i] = sum(fit$residuals^2)
+ }
> plot(1:10, rss, xlab = "Degree", ylab = "RSS", type = "l")
```



З графіка, можна сказати, що RSS зменшується зі збільшенням степеня полінома.

4.3

```
> deltas = rep(NA, 10)
> for (i in 1:10) {
+   fit = glm(nox ~ poly(dis, i), data = Boston)
+   deltas[i] = cv.glm(Boston, fit, K = 10)$delta[1]
+ }
> plot(1:10, deltas, xlab = "Degree", ylab = "Test MSE", type = "l")
```



З графіка видно, що найменше MSE при степені полінома чотири.

4.4

```
> fit = lm(nox ~ bs(dis, knots = c(4, 7, 11)), data = Boston)
> summary(fit)
```

Call:
lm(formula = nox ~ bs(dis, knots = c(4, 7, 11)), data = Boston)

Residuals:

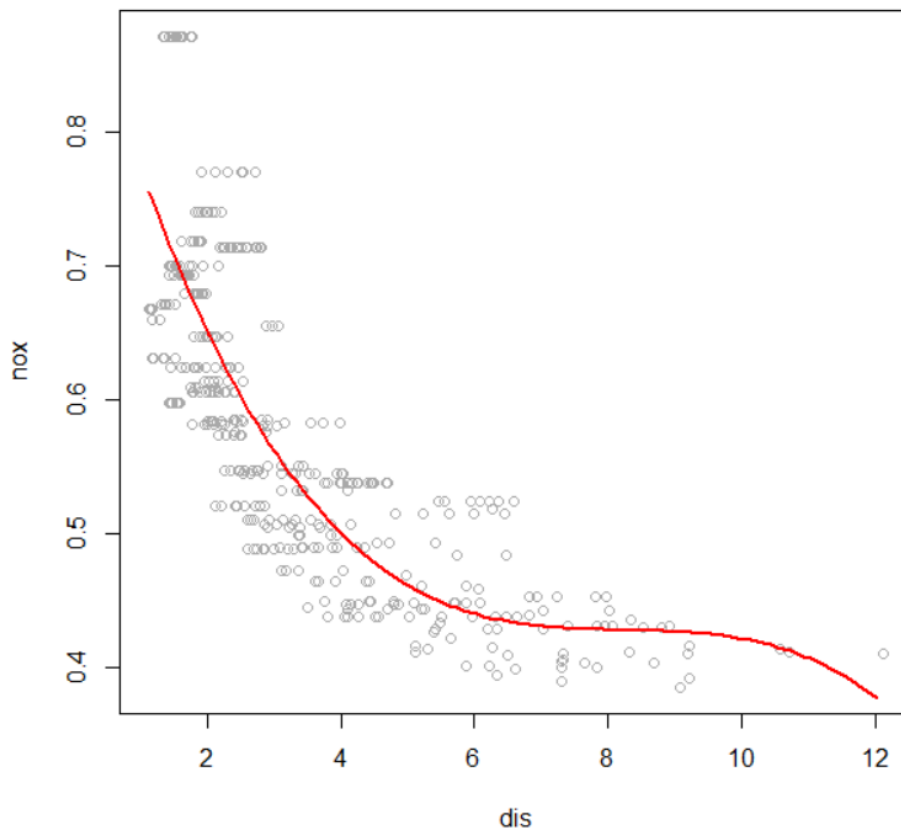
Min	1Q	Median	3Q	Max
-0.124567	-0.040355	-0.008702	0.024740	0.192920

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	0.73926	0.01331	55.537	< 2e-16 ***
bs(dis, knots = c(4, 7, 11))1	-0.08861	0.02504	-3.539	0.00044 ***
bs(dis, knots = c(4, 7, 11))2	-0.31341	0.01680	-18.658	< 2e-16 ***
bs(dis, knots = c(4, 7, 11))3	-0.26618	0.03147	-8.459	3.00e-16 ***
bs(dis, knots = c(4, 7, 11))4	-0.39802	0.04647	-8.565	< 2e-16 ***
bs(dis, knots = c(4, 7, 11))5	-0.25681	0.09001	-2.853	0.00451 **
bs(dis, knots = c(4, 7, 11))6	-0.32926	0.06327	-5.204	2.85e-07 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

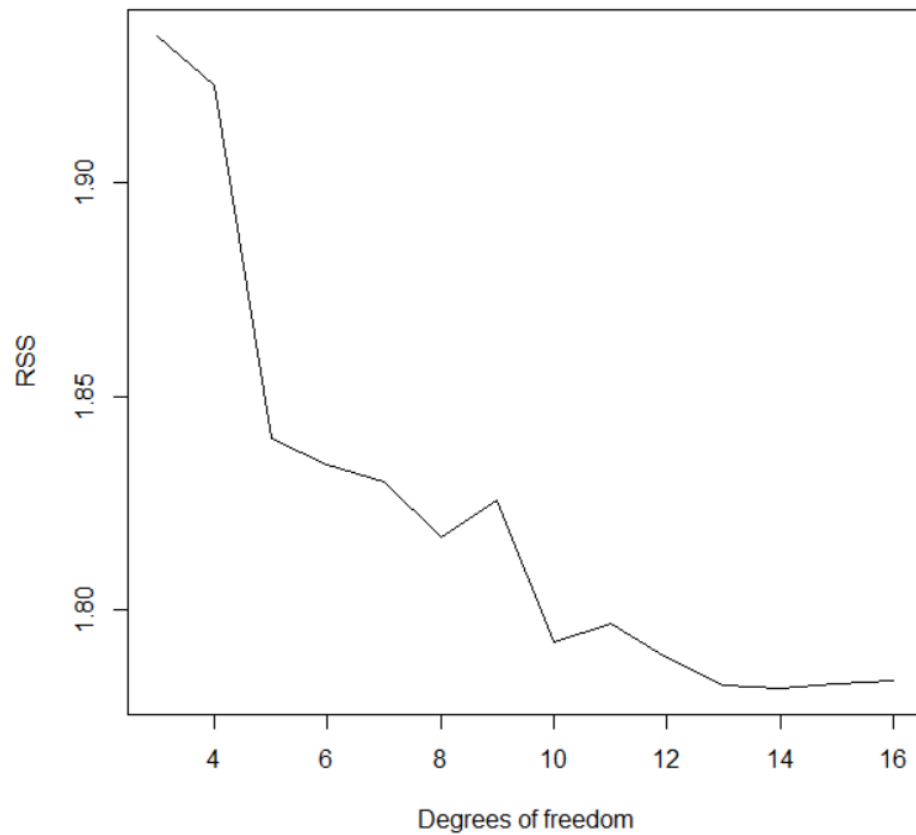
Residual standard error: 0.06185 on 499 degrees of freedom
Multiple R-squared: 0.7185, Adjusted R-squared: 0.7151
F-statistic: 212.3 on 6 and 499 DF, p-value: < 2.2e-16



З наведених вище результатів можна сказати, що всі терми у сплайні є значущими.

4.5

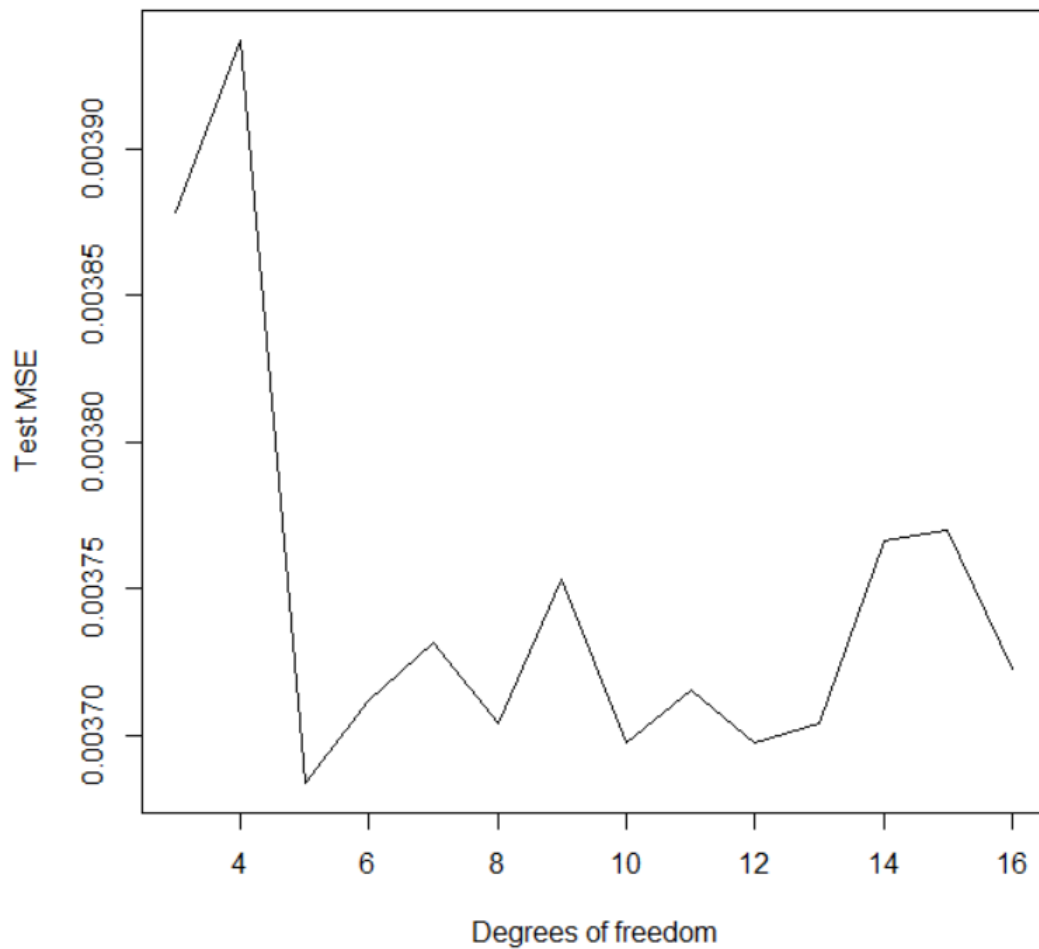
```
> rss = rep(NA, 16)
> for (i in 3:16) {
+   fit = lm(nox ~ bs(dis, df = i), data = Boston)
+   rss[i] = sum(fit$residuals^2)
+ }
> plot(3:16, rss[-c(1, 2)], xlab = "Degrees of freedom", ylab = "RSS", type = "l")
```



Можемо побачити, що RSS спадає до 14, а потім починає по трохи зростати.

4.6

```
> cv = rep(NA, 16)
> for (i in 3:16) {
+   fit = glm(nox ~ bs(dis, df = i), data = Boston)
+   cv[i] = cv.glm(Boston, fit, K = 10)$delta[1]
+ }
There were 50 or more warnings (use warnings() to see the first 50)
> plot(3:16, cv[-c(1, 2)], xlab = "Degrees of freedom", ylab = "Test MSE", type = "l")
```

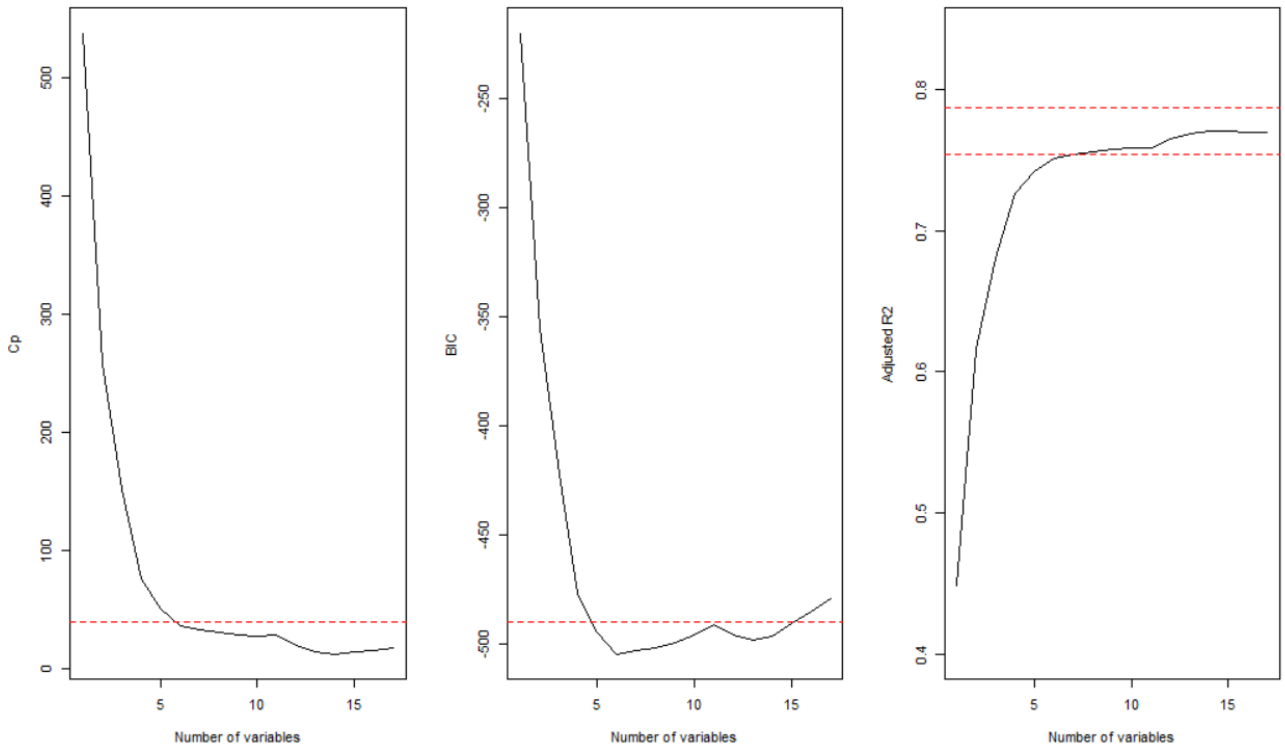


Тестова MSE мінімальна для п'яти ступенів свободи.

5. College

5.1

```
> train = sample(length(Outstate), length(Outstate) / 2)
> test = -train
> College.train = College[train, ]
> College.test = College[test, ]
> fit = regsubsets(Outstate ~ ., data = College.train, nvmax = 17, method = "forward")
> fit.summary = summary(fit)
> par(mfrow = c(1, 3))
> plot(fit.summary$cp, xlab = "Number of variables", ylab = "Cp", type = "l")
> min.cp = min(fit.summary$cp)
> std.cp = sd(fit.summary$cp)
> abline(h = min.cp + 0.2 * std.cp, col = "red", lty = 2)
> abline(h = min.cp - 0.2 * std.cp, col = "red", lty = 2)
> plot(fit.summary$bic, xlab = "Number of variables", ylab = "BIC", type='l')
> min.bic = min(fit.summary$bic)
> std.bic = sd(fit.summary$bic)
> abline(h = min.bic + 0.2 * std.bic, col = "red", lty = 2)
> abline(h = min.bic - 0.2 * std.bic, col = "red", lty = 2)
> plot(fit.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l", ylim = c(0.4, 0.84))
> max.adj2 = max(fit.summary$adjr2)
> std.adj2 = sd(fit.summary$adjr2)
> abline(h = max.adj2 + 0.2 * std.adj2, col = "red", lty = 2)
> abline(h = max.adj2 - 0.2 * std.adj2, col = "red", lty = 2)
```

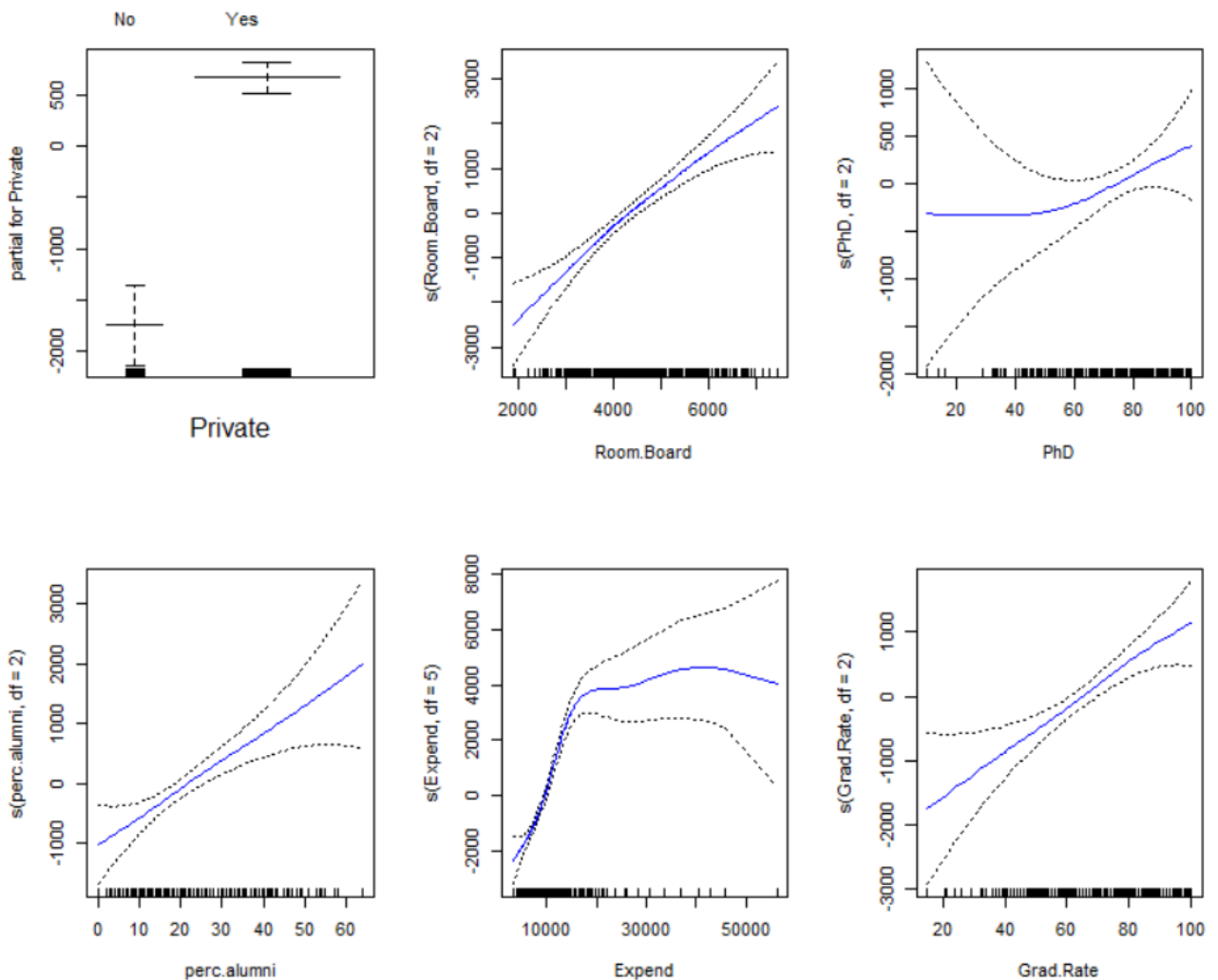


C_p , BIC і скорегований R^2 показують, що розмір 6 є мінімальним розміром для підмножини.

```
> fit = regsubsets(Outstate ~ ., data = College, method = "forward")
> coeffs = coef(fit, id = 6)
> names(coeffs)
[1] "(Intercept)" "PrivateYes" "Room.Board" "PhD" "perc.alumni" "Expend" "Grad.Rate"
```

5.2

```
> library(gam)
> fit = gam(Outstate ~ Private + s(Room.Board, df = 2) + s(PhD, df = 2) + s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate, df = 2),
+ data=College.train)
> par(mfrow = c(2, 3))
> plot(fit, se = T, col = "blue")
```



5.3

```
> preds = predict(fit, College.test)
> err = mean((College.test$Outstate - preds)^2)
> err
[1] 3349290
> tss = mean((College.test$Outstate - mean(College.test$Outstate))^2)
> rss = 1 - err / tss
> rss
[1] 0.7660016
```

Ми отримуємо тест R^2 0,766, використовуючи GAM з 6 предикторами.

5.4

```
> summary(fit)

Call: gam(formula = Outstate ~ Private + s(Room.Board, df = 2) + s(PhD,
      df = 2) + s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate,
      df = 2), data = College.train)
Deviance Residuals:
      Min       1Q   Median       3Q      Max
-7402.89 -1114.45  -12.67  1282.69  7470.60

(Dispersion Parameter for gaussian family taken to be 3711182)

Null Deviance: 6989966760 on 387 degrees of freedom
Residual Deviance: 1384271126 on 373 degrees of freedom
AIC: 6987.021

Number of Local Scoring Iterations: NA

Anova for Parametric Effects
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Private	1	1778718277	1778718277	479.286	< 2.2e-16 ***
s(Room.Board, df = 2)	1	1577115244	1577115244	424.963	< 2.2e-16 ***
s(PhD, df = 2)	1	322431195	322431195	86.881	< 2.2e-16 ***
s(perc.alumni, df = 2)	1	336869281	336869281	90.771	< 2.2e-16 ***
s(Expend, df = 5)	1	530538753	530538753	142.957	< 2.2e-16 ***
s(Grad.Rate, df = 2)	1	86504998	86504998	23.309	2.016e-06 ***
Residuals	373	1384271126	3711182		

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Anova for Nonparametric Effects
```

	Npar	Df	Npar F	Pr(F)
(Intercept)				
Private				
s(Room.Board, df = 2)	1	1.9157	0.1672	
s(PhD, df = 2)	1	0.9699	0.3253	
s(perc.alumni, df = 2)	1	0.1859	0.6666	
s(Expend, df = 5)	4	20.5075	2.665e-15	***
s(Grad.Rate, df = 2)	1	0.5702	0.4506	

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA показує чіткий не лінійний зв'язок між Outstate та Expend.

6. Метод підгонки

6.1

```
> set.seed(1)
> X1 = rnorm(100)
> X2 = rnorm(100)
> eps = rnorm(100, sd = 0.1)
> Y = -3.14 + 2.72 * X1 + 0.5 * X2 + eps
```

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$$

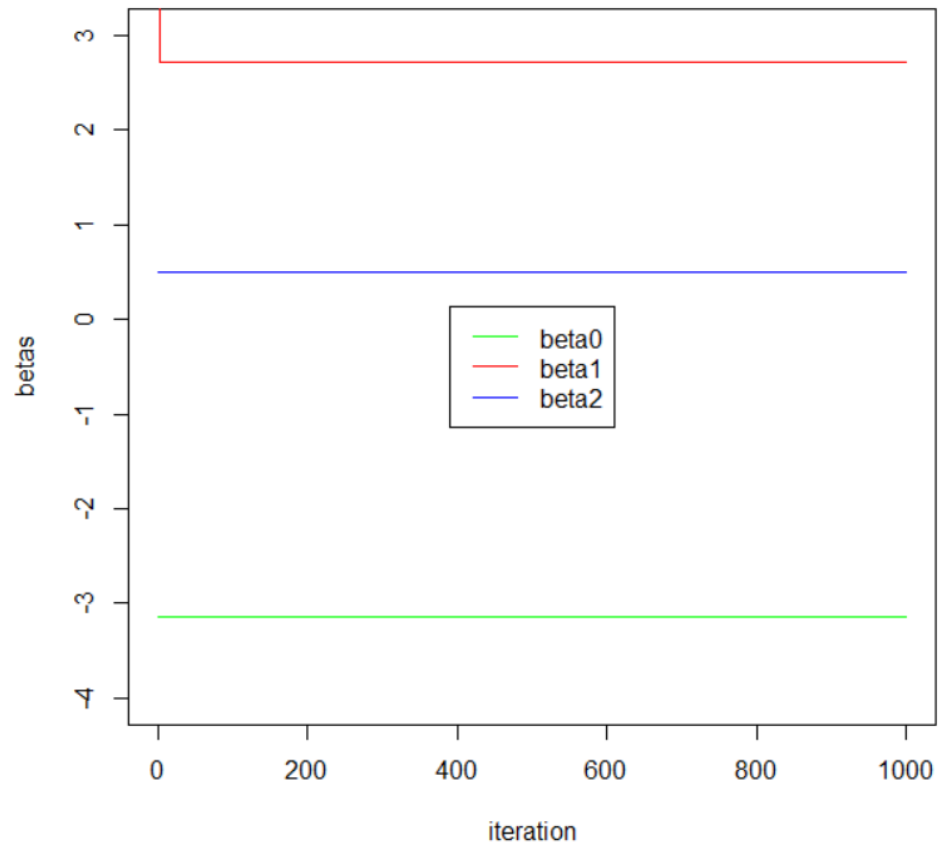
6.2

```
> beta0 = rep(0, 1000)
> beta1 = rep(0, 1000)
> beta2 = rep(0, 1000)
> beta1[1] = 7
```

Візьмемо $\beta_1 = 7$.

6.3–6.5

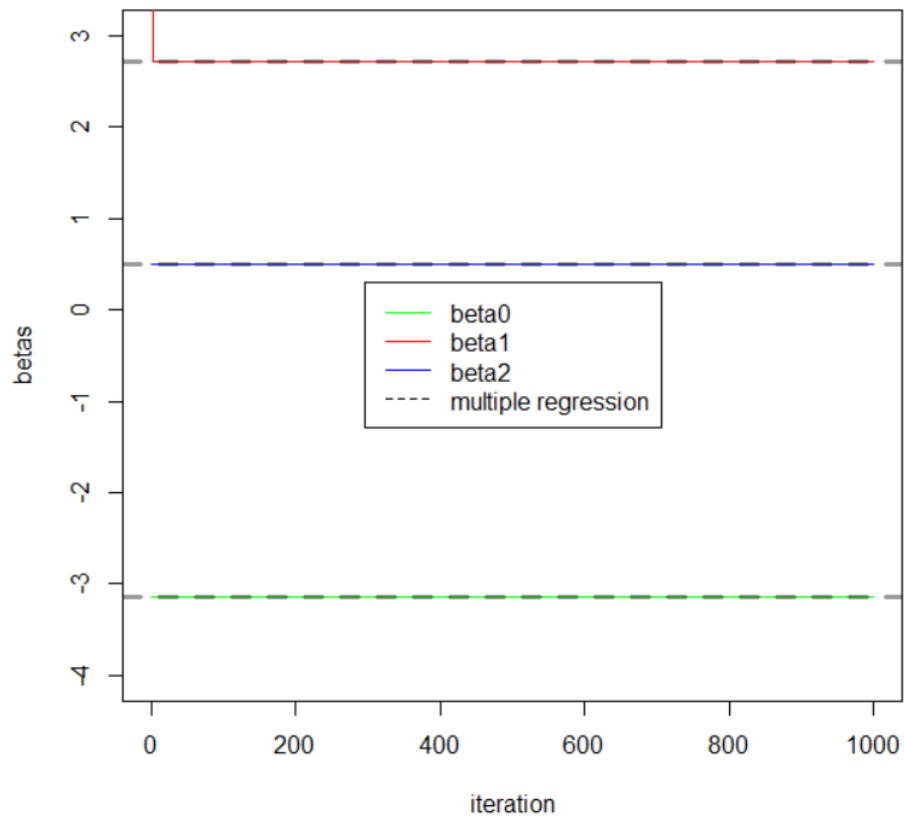
```
> for (i in 1:1000) {
+   a = Y - beta1[i] * X1
+   beta2[i] = lm(a ~ X2)$coef[2]
+   a = Y - beta2[i] * X2
+   lm.fit = lm(a ~ X1)
+   if (i < 1000) {
+     beta1[i + 1] = lm.fit$coef[2]
+   }
+   beta0[i] = lm.fit$coef[1]
+ }
> plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-4, 3), col = "green")
> lines(1:1000, beta1, col = "red")
> lines(1:1000, beta2, col = "blue")
> legend("center", c("beta0", "beta1", "beta2"), lty = 1, col = c("green", "red", "blue"))
```



Коефіцієнти швидко досягають значень найменших квадратів.

6.6

```
> lm.fit = lm(Y ~ X1 + X2)
> plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-4, 3), col = "green")
> lines(1:1000, beta1, col = "red")
> lines(1:1000, beta2, col = "blue")
> abline(h = lm.fit$coef[1], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
> abline(h = lm.fit$coef[2], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
> abline(h = lm.fit$coef[3], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))
> legend("center", c("beta0", "beta1", "beta2", "multiple regression"), lty = c(1,
+   1, 1, 2), col = c("green", "red", "blue", "black"))
```



Пунктирні лінії на графіку показують, що коефіцієнти множинної регресії точно збігаються з коефіцієнтами отриманими за допомогою backfitting-у.

6.7

Коли зв'язок між ігреком та іксами є лінійним, однієї ітерації достатньо для досягнення хорошого наближення до істинних коефіцієнтів регресії.

7.

```

> # 7
> set.seed(1)
> p = 100
> n = 1000
> x = matrix(ncol = p, nrow = n)
> coefi = rep(0, p)
> for (i in 1:p) {
+   x[, i] = rnorm(n)
+   coefi[i] = rnorm(1) * 100
+ }
> y = x %*% coefi + rnorm(n)
>
> beta = rep(0, p)
> max_iterations = 1000
> errors = rep(0, max_iterations + 1)
> iter = 2
> errors[1] = Inf
> errors[2] = sum((y - x %*% beta)^2)
> threshold = 1e-04
> while (iter < max_iterations && errors[iter - 1] - errors[iter] > threshold) {
+   for (i in 1:p) {
+     a = y - x %*% beta + beta[i] * x[, i]
+     beta[i] = lm(a ~ x[, i])$coef[2]
+   }
+   iter = iter + 1
+   errors[iter] = sum((y - x %*% beta)^2)
+   print(c(iter - 2, errors[iter - 1], errors[iter]))
+ }
[1]          1 1016122216   37472751
[1]          2 37472751  1669889
[1]          3.00 1669889.42   77923.75
[1]          4.000 77923.754   6157.425
[1]          5.000 6157.425 1277.046
[1]          6.0000 1277.0458   928.3072
[1]          7.0000 928.3072 904.7608
[1]          8.0000 904.7608 903.2173
[1]          9.0000 903.2173 903.1259
[1]         10.0000 903.1259 903.1232
[1]         11.0000 903.1232 903.1239
>
> plot(1:11, errors[3:13])

```

Десять ітерацій достатньо, щоб отримати хорошу апроксимацію визначену пороговим значенням суми квадратичних помилок між наступними ітераціями. Також видно, що похибка збільшується на 11-й ітерації.

