МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

ЗВІТ

до індивідуального завдання №4

з дисципліни «Моделі статистичного навчання»

Виконав

студент групи ПМіМ-12:

Зелінський Олександр

Перевірив:

Проф. Заболоцький Т. М.

Львів – 2021

Хід виконання

1. Метод валідаційного набору

Приклад даних з Default та результат функції summary для них.

	default	student	balance	income
1	No	No	729.5265	44361.63
2	No	Yes	817.1804	12106.13
3	No	No	1073.549	31767.14
4	No	No	529.2506	35704.49
5	No	No	785.6559	38463.5
6	No	Yes	919.5885	7491.559
7	No	No	825.5133	24905.23
8	No	Yes	808.6675	17600.45
9	No	No	1161.058	37468.53
10	No	No	0	29275.27
11	No	Yes	0	21871.07
12	No	Yes	1220.584	13268.56
13	No	No	237.0451	28251.7
14	No	No	606.7423	44994.56
15	No	No	1112.968	23810.17
16	No	No	286.2326	45042.41
17	No	No	0	50265.31
18	No	Yes	527.5402	17636.54
19	No	No	485.9369	61566.11

```
> library(ISLR)
> attach(Default)
> summary(Default)
 default     student       balance           income     
 No :9667   No :7056   Min.   :   0.0   Min.   :  772  
 Yes: 333   Yes:2944   1st Qu.: 481.7   1st Qu.:21340  
                       Median : 823.6   Median :34553  
                       Mean   : 835.4   Mean   :33517  
                       3rd Qu.:1166.3   3rd Qu.:43808  
                       Max.   :2654.3   Max.   :73554  
```

1.1 Логістична регресія

```
> set.seed(1)
> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial")
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4725  -0.1444  -0.0574  -0.0211   3.7245

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1579.0  on 9997  degrees of freedom
AIC: 1585

Number of Fisher Scoring iterations: 8
```

1.2

```
> # 1.2.1
> train = sample(dim(Default)[1], dim(Default)[1] / 2)
```

1.2.2

```
> # 1.2.2
> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
> summary(fit.glm)

Call:
glm(formula = default ~ income + balance, family = "binomial",
    data = Default, subset = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1634  -0.1446  -0.0553  -0.0203   3.3281

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.158e+01  6.008e-01 -19.281  < 2e-16 ***
income       1.975e-05  6.775e-06   2.916  0.00355 **
balance      5.723e-03  3.180e-04  17.996  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1543.58  on 4999  degrees of freedom
Residual deviance:  816.44  on 4997  degrees of freedom
AIC: 822.44

Number of Fisher Scoring iterations: 8
```

1.2.3

```
> # 1.2.3
> Default.test = Default[-train,]
> probs = predict(fit.glm, newdata = Default.test, type = "response")
> pred.glm = rep("No", length(probs))
> pred.glm[probs > 0.5] = "Yes"
```

1.2.4

```
> # 1.2.4
> paste("Коефіцієнт помилок: ", mean(pred.glm != Default.test$default))
[1] "Коефіцієнт помилок:  0.0244"
```

Можемо бачити що коефіцієнт тестової помилки буде 2,4% що є хорошим результатом.

1.3

```
> # 1.3
> for (i in (0:2)) {
+   train = sample(dim(Default)[1], dim(Default)[1] / 2)
+
+   fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = train)
+
+   Default.test = Default[-train,]
+   probs = predict(fit.glm, newdata = Default.test, type = "response")
+   pred.glm = rep("No", length(probs))
+   pred.glm[probs > 0.5] = "Yes"
+
+   print(paste("Коефіцієнт помилок: ", mean(pred.glm != Default.test$default)))
+ }
[1] "Коефіцієнт помилок:  0.027"
[1] "Коефіцієнт помилок:  0.0294"
[1] "Коефіцієнт помилок:  0.0254"
```

Видно, що коефіцієнт тестової помилки змінюється в залежності від того, які спостереження потрапляють в навчальний набір, проте в цілому ці значення не сильно відрізняються.

1.4

```
> train = sample(dim(Default)[1], dim(Default)[1] / 2)
>
> fit.glm = glm(default ~ income + balance + student, data = Default, family = "binomial", subset = train)
>
> Default.test = Default[-train, ]
>
> probs = predict(fit.glm, newdata = Default.test, type = "response")
> pred.glm = rep("No", length(probs))
> pred.glm[probs > 0.5] = "Yes"
>
> paste("Коефіцієнт помилок:", mean(pred.glm != Default.test$default))
[1] "Коефіцієнт помилок: 0.0284"
```

Додавання фіктивної змінної student не покращило коефіцієнт тестової помилки, можна стверджувати, що він залишився в тих самих межах.

2. Бутстрап

2.1

```
> library(ISLR)
> attach(Default)
The following objects are masked from Default (pos = 3):

    balance, default, income, student

>
> set.seed(1)
> fit.glm = glm(default ~ income + balance, data = Default, family = "binomial")
> summary(fit.glm)$coef
                 Estimate   Std. Error    z value      Pr(>|z|)
(Intercept) -1.154047e+01 4.347564e-01 -26.544680 2.958355e-155
income       2.080898e-05 4.985167e-06   4.174178  2.990638e-05
balance      5.647103e-03 2.273731e-04  24.836280 3.638120e-136
```

В результаті, стандартні відхилення для коефіцієнтів: $\beta_0 = 0{,}435, \beta_1 = 4{,}985 * 10^{-6}, \beta_2 = 2{,}274 * 10^{-4}$.

2.2

```
> boot.fn = function(data, index) {
+     fit = glm(default ~ income + balance, data = data, family = "binomial", subset = index)
+     return (coef(fit))
+ }
```

2.3

```
> library(boot)
> boot(Default, boot.fn, 100)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Default, statistic = boot.fn, R = 100)


Bootstrap Statistics :
         original        bias     std. error
t1* -1.154047e+01  8.556378e-03 4.122015e-01
t2*  2.080898e-05 -3.993598e-07 4.186088e-06
t3*  5.647103e-03 -4.116657e-06 2.226242e-04
```

Стандартні відхилення для коефіцієнтів: $\beta_0 = 0{,}412, \beta_1 = 4{,}186 * 10^{-6}, \beta_2 = 2{,}262 * 10^{-4}$.

2.4

Як можемо побачити стандартні похибки досить близькі проте в bootstrap дещо нижчі.

3. LOOCV

3.1

```
> fit.glm = glm(Direction ~ Lag1 + Lag2, data = Weekly, family = "binomial")
> summary(fit.glm)$coef
               Estimate Std. Error   z value     Pr(>|z|)
(Intercept)  0.22122405 0.06146572  3.599145 0.0003192652
Lag1        -0.03872222 0.02621658 -1.477013 0.1396722362
Lag2         0.06024830 0.02654589  2.269590 0.0232324586
```

3.2

```
> fit.glm2 = glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = "binomial")
> summary(fit.glm2)$coef
               Estimate Std. Error   z value     Pr(>|z|)
(Intercept)  0.22324305 0.06149894  3.630031 0.0002833875
Lag1        -0.03843317 0.02621860 -1.465874 0.1426825151
Lag2         0.06084763 0.02656088  2.290874 0.0219707105
```

3.3

```
> predict.glm(fit.glm2, Weekly[1, ], type = "response") > 0.5
   1
TRUE
```

З цього можемо зробити висновок, що перше спостережння “Up” було не правильно класифіковане, оскільки справжній напрямок “Down”

3.4

```
> err = rep(0, dim(Weekly)[1])
> for (i in 1:dim(Weekly)[1]) {
+     fit.glm = glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ],  family = "binomial")
+     if (predict.glm(fit.glm, Weekly[i, ], type = "response") > 0.5) {
+       if (Direction[i] == "Down") {
+         err[i] = 1
+       }
+     }
+ }
```

3.5

```
> paste("Оцінка LOOCV :", mean(err))
[1] "Оцінка LOOCV : 0.413223140495868"
```

Отже, бачимо, що оцінка LOOCV коефіцієнта тестової помилки рівна 41,32%, що є достатньо хорошим результатом.

4.

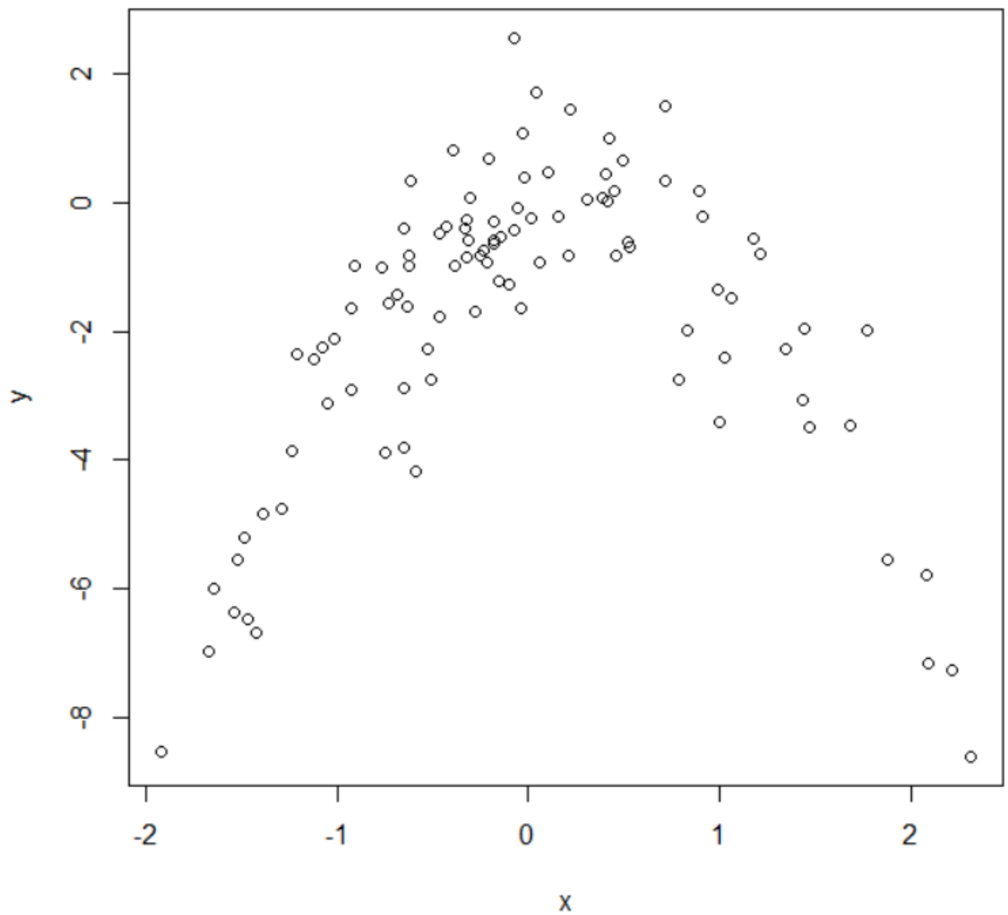
4.1

```
> set.seed(1)
> y = rnorm(100)
> x = rnorm(100)
> y = x - 2 * x^2 + rnorm(100)
```

У нашому випадку $n = 100, p = 2$, а модель у формі рівняння матиме наступний вигляд:

$$Y = X - 2 * X^2 + \varepsilon$$

4.2



З графіка розсіювання, очевидно, що в нас не лінійна, а скоріше квадратична залежність.

4.3

4.3.1

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

```
+       # 4.3.1
+     fit.glm = glm(y ~ x)
+     print(
+ paste("Оцінка LOOCV [beta0 - beta1]:", round(cv.glm(Coords, fit.glm)$delta[1], 2))
+     )
```

```
[1] "Оцінка LOOCV [beta0 - beta1]: 5.89"
```

4.3.2

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

```
+       # 4.3.2
+     fit.glm2 = glm(y ~ x + I(x^2))
+     print(
+ paste("Оцінка LOOCV [beta0 - beta2]:", round(cv.glm(Coords, fit.glm2)$delta[1], 2))
+     )
```

```
[1] "Оцінка LOOCV [beta0 - beta2]: 1.09"
```

4.3.3

$$Y = \beta_0 + \beta_1 X + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$$

```
+       # 4.3.3
+     fit.glm3 = glm(y ~ poly(x, 3))
+     print(
+ paste("Оцінка LOOCV [beta0 - beta3]:", round(cv.glm(Coords, fit.glm3)$delta[1], 2))
+     )
```

```
[1] "Оцінка LOOCV [beta0 - beta3]: 1.1"
```

4.3.4

$$Y = \beta_0 + \beta_1 X + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \varepsilon$$

```
+       # 4.3.4
+     fit.glm4 = glm(y ~ poly(x, 4))
+     print(
+ paste("Оцінка LOOCV [beta0 - beta4]:", round(cv.glm(Coords, fit.glm4)$delta[1], 2))
+     )
```

```
[1] "Оцінка LOOCV [beta0 - beta4]: 1.11"
```

4.4

```
> LOOCV(2)
[1] "Оцінка LOOCV [beta0 - beta1]: 5.89"
[1] "Оцінка LOOCV [beta0 - beta2]: 1.09"
[1] "Оцінка LOOCV [beta0 - beta3]: 1.1"
[1] "Оцінка LOOCV [beta0 - beta4]: 1.11"
> LOOCV(4)
[1] "Оцінка LOOCV [beta0 - beta1]: 5.89"
[1] "Оцінка LOOCV [beta0 - beta2]: 1.09"
[1] "Оцінка LOOCV [beta0 - beta3]: 1.1"
[1] "Оцінка LOOCV [beta0 - beta4]: 1.11"
> LOOCV(8)
[1] "Оцінка LOOCV [beta0 - beta1]: 5.89"
[1] "Оцінка LOOCV [beta0 - beta2]: 1.09"
[1] "Оцінка LOOCV [beta0 - beta3]: 1.1"
[1] "Оцінка LOOCV [beta0 - beta4]: 1.11"
```

Так результати такі самі, це пояснюється тим, що LOOCV при оцінці забезпечує те, що в не залежності від порядку даних кожен елемент буде використаний як тестовий, принаймні один раз.

4.5

Видно, що оцінка LOOCV для тестового мінімального квадратичного відхилення є найменшою для 4.3.2, що легко пояснюється тим, що наше відношення з 4.1 – квадратичне.

4.6

```
> fit.glm = glm(y ~ poly(x, 4))
> summary(fit.glm)

Call:
glm(formula = y ~ poly(x, 4))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8914  -0.5244   0.0749   0.5932   2.7796

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.8277     0.1041 -17.549   <2e-16 ***
poly(x, 4)1   2.3164     1.0415   2.224   0.0285 *
poly(x, 4)2 -21.0586     1.0415 -20.220   <2e-16 ***
poly(x, 4)3  -0.3048     1.0415  -0.293   0.7704
poly(x, 4)4  -0.4926     1.0415  -0.473   0.6373
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 1.084654)

    Null deviance: 552.21  on 99  degrees of freedom
Residual deviance: 103.04  on 95  degrees of freedom
AIC: 298.78

Number of Fisher Scoring iterations: 2
```

Значення p вказують на те, що лінійний і квадратичний члени є статистично значущими, а кубічний і 4-го ступеня не є. Це, ясна річ, узгоджується з нашими результатами LOOCV, які були мінімальними для квадратичної моделі.

5. Boston

5.1

```
> mean(medv)
[1] 22.53281
```

5.2

```
> medv_se = sd(medv) / sqrt(length(medv))
> paste("Стандартна похибка: ", round(medv_se, 2))
[1] "Стандартна похибка:  0.41"
```

З результатів видно, що стандартна похибка 41%.

5.3

```
> library(boot)
> set.seed(1)
>
> boot.fn = function(data, index) {
+     return (mean(data[index]))
+ }
>
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original      bias    std. error
t1* 22.53281 0.007650791   0.4106622
```

Зважаючи на отримані результати можна сказати, що стандартні похибки практично не відрізняються.

5.4

```
> t.test(medv)

        One Sample t-test

data:  medv
t = 55.111, df = 505, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 21.72953 23.33608
sample estimates:
mean of x
 22.53281

> ci = c(22.533 - 2 * 0.411, 22.533 + 2 * 0.411)
> ci
[1] 21.711 23.355
```

З результатів, можна сказати, що інтервал довіри для bootstrap є дуже близьким до того який видає функція t.test.

5.5

```
> median(medv)
[1] 21.2
```

5.6

```
> boot.fn2 = function(data, index) {
+      return (median(data[index]))
+ }
>
> boot(medv, boot.fn2, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn2, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*     21.2 -0.0386   0.3770241
```

Ми отримали таке саме значення як і медіана тобто 21,2 із відносно невеликою стандартною похибкою 0.377, що є невеликим в порівнянні із значенням медіани.

5.7

```
> quantile(medv, c(0.1))
  10%
12.75
```

5.8

```
> # 5.8
> boot.fn3 = function(data, index) {
+     return (quantile(data[index], c(0.1)))
+ }
>
> boot(medv, boot.fn3, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn3, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*    12.75  0.0186   0.4925766
```

Ми отримали таке саме значення як і десятий процентиль тобто 12,75 із відносно невеликою стандартною похибкою 0.492, що є відносно невеликим в порівнянні зі значенням процентиля.