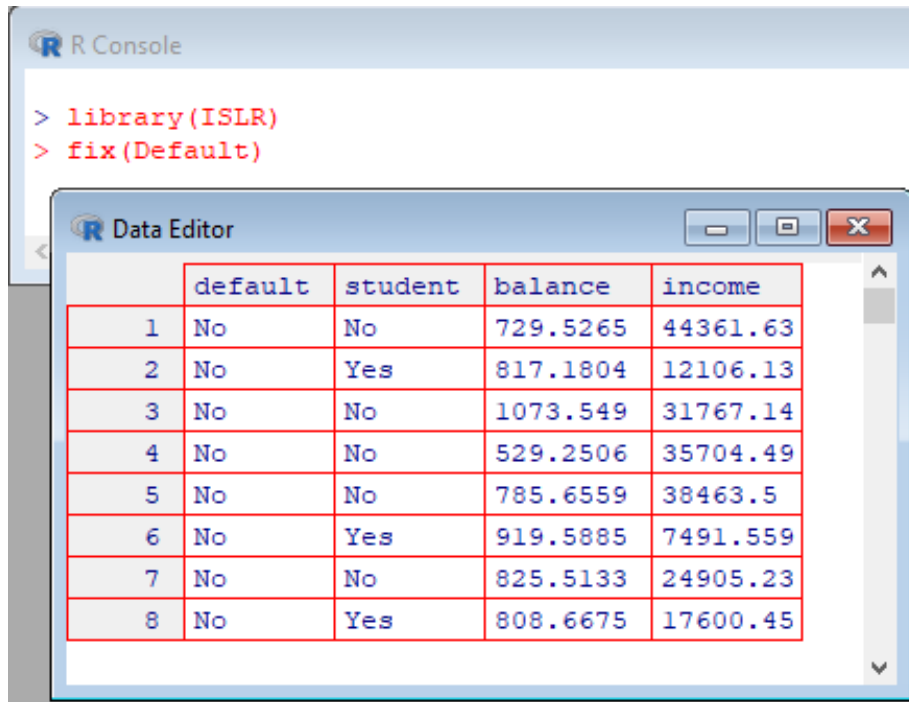


Звіт
до індивідуального завдання №4
з предмету Моделі статистичного навчання

Роботу виконала:
Мерцало Ірина Ігорівна,
студентка групи ПМІМ-11

Завдання 1. Використовуючи метод валідаційного набору, оцінити тестову помилку моделі логістичної регресії з даних Default, на основі income та balance.

1.1 Розглянула дані Default, які є частиною пакету ISLR.



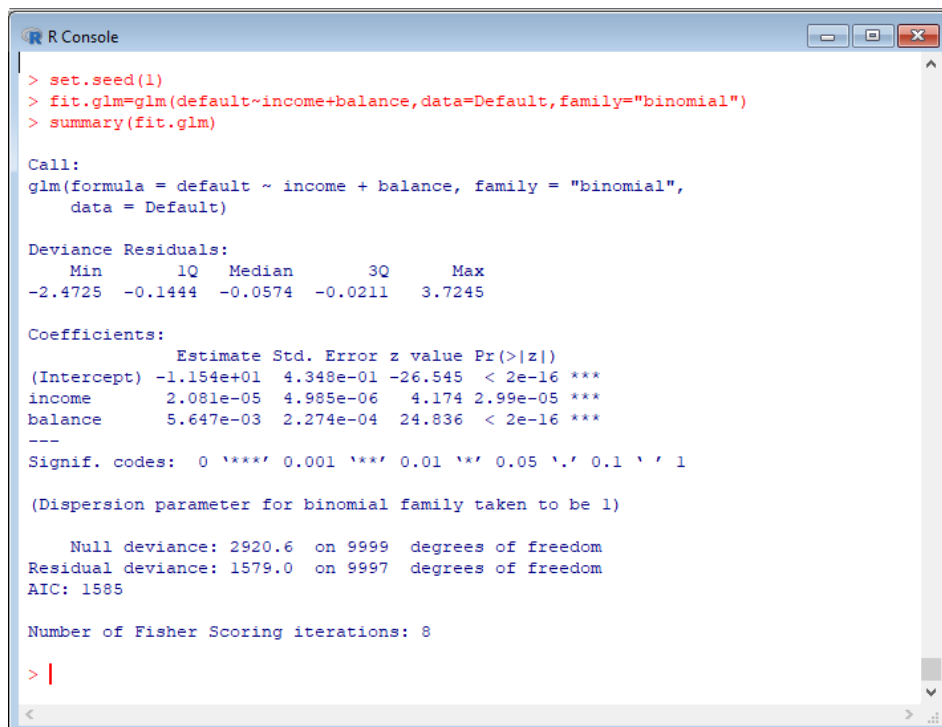
The image shows an R Console window with the following commands:

```
> library(ISLR)
> fix(Default)
```

Below the console is the R Data Editor window, which displays the first 8 rows of the 'Default' dataset. The columns are 'default', 'student', 'balance', and 'income'.

	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

Використовуючи функцію `glm()`, побудувала логістичну регресійну модель, яка використовує income та balance для передбачення default. Використала функцію `summary()` для виводу результатів.



The image shows the R Console window with the following commands and output:

```
> 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 Використовуючи метод валідаційного набору, оцінити тестову помилку цієї моделі.

1.2.1. Розділила вибірку на навчальний та тестовий набори:

```
R Console
> train=sample(dim(Default)[1],dim(Default)[1]/2)
> |
```

1.2.2 Оцінила логістичну регресійну модель, використовуючи навчальну вибірку. Використала функцію `summary()` для виводу результатів:

```
R Console
> fit.glm2=glm(default~income+balance,data=Default,family="binomial",subset=train)
> summary(fit.glm2)

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5830  -0.1428  -0.0573  -0.0213   3.3395

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.194e+01  6.178e-01 -19.333  < 2e-16 ***
income       3.262e-05  7.024e-06   4.644  3.41e-06 ***
balance      5.689e-03  3.158e-04  18.014  < 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: 1523.8  on 4999  degrees of freedom
Residual deviance:  803.3  on 4997  degrees of freedom
AIC: 809.3

Number of Fisher Scoring iterations: 8

> |
```

1.2.3. Спрогнозувала дефолт-статус для кожної людини в тестовій вибірці на основі передбачення апостеріорної ймовірності дефолту для цієї людини та встановлення статусу дефолт, якщо отримана ймовірність перевищує значення 0,5:

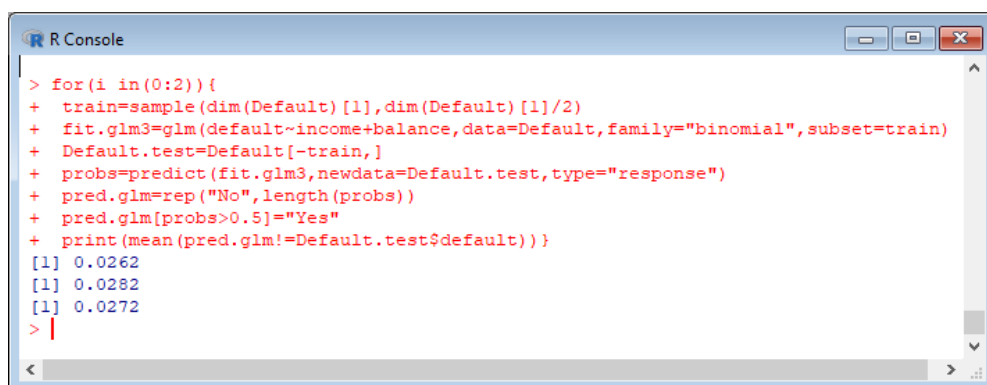
```
R Console
> Default.test=Default[-train,]
> probs=predict(fit.glm2,newdata = Default.test,type="response")
> pred.glm=rep("No",length(probs))
> pred.glm[probs>0.5]="Yes"
> |
```

1.2.4. Оцінила тестову помилку на валідаційній множині шляхом обчислення частки статусу осіб, які неправильно класифіковані:

```
R Console
> mean(pred.glm!=Default.test$default)
[1] 0.0254
> |
```

Її значення 2.5%, що є досить малим показником.

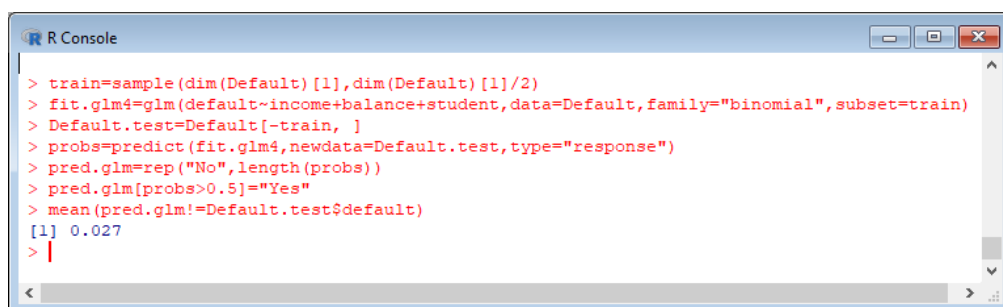
1.3. Повторила 1.2 три рази, використовуючи три різні розбиття вибірки на навчальний та тестовий набори:



```
> for(i in 0:2){
+   train=sample(dim(Default)[1],dim(Default)[1]/2)
+   fit.glm3=glm(default~income+balance,data=Default,family="binomial",subset=train)
+   Default.test=Default[~train,]
+   probs=predict(fit.glm3,newdata=Default.test,type="response")
+   pred.glm=rep("No",length(probs))
+   pred.glm[probs>0.5]="Yes"
+   print(mean(pred.glm!=Default.test$default))
+ }
[1] 0.0262
[1] 0.0282
[1] 0.0272
>
```

По результатах можна побачити, що значення тестової помилки на валідаційній множині змінюється не суттєво при різних розбиттях вибірки.

1.4. Розглянула модель логістичної регресії, яка передбачає ймовірність дефолту за допомогою змінних income, balance та фіктивної змінної для student:



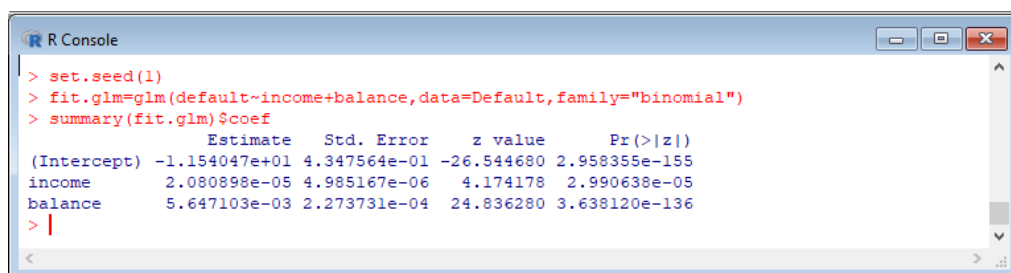
```
> train=sample(dim(Default)[1],dim(Default)[1]/2)
> fit.glm4=glm(default~income+balance+student,data=Default,family="binomial",subset=train)
> Default.test=Default[~train,]
> probs=predict(fit.glm4,newdata=Default.test,type="response")
> pred.glm=rep("No",length(probs))
> pred.glm[probs>0.5]="Yes"
> mean(pred.glm!=Default.test$default)
[1] 0.027
>
```

Тестова помилка для цієї моделі, використовуючи заданий підхід, 2.7%.

Включення фіктивної змінної для student не призводить до зменшення тестової помилки.

Завдання 2. Продовжила дослідження логістичної регресії для прогнозування ймовірності дефолту на основі income та balance з даних Default.

2.1 Використовуючи функції summary() та glm(), визначила оцінку середньоквадратного відхилення параметрів логістичної регресії, яка використовує income та balance для оцінки ймовірності дефолту:



```
> 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

```
>
```

Вона дорівнює 4.35e-01, 4.99e-06 та 2.27e-04 відповідно.

2.2 Написала функцію `boot.fn()`, яка приймає на вхід набір даних та індекси спостережень для використання і виводить оцінки коефіцієнтів логістичної регресії, яка використовує `income` та `balance` для оцінки ймовірності дефолту:

```
R Console
> boot.fn=function(data,index){
+   fit2=glm(default~income+balance,data=data,family="binomial",subset=index)
+   print(coef(fit2))
+ }
```

2.3 Використала функцію `boot()` разом із функцією `boot.fn()` для оцінки середньоквадратичного відхилення параметрів логістичної регресії, яка використовує `income` та `balance` для оцінки ймовірності дефолту:

```
R Console
> library(boot)
> boot(data=Default,statistic=boot.fn,R=100)
      (Intercept)      income      balance
-1.154047e+01  2.080898e-05  5.647103e-03
      (Intercept)      income      balance
-1.252108e+01  2.805769e-05  6.129378e-03
      (Intercept)      income      balance
-1.088944e+01  1.527579e-05  5.282210e-03
      (Intercept)      income      balance
-1.215956e+01  1.688150e-05  6.089004e-03
      (Intercept)      income      balance
-1.168986e+01  2.922130e-05  5.562715e-03
      (Intercept)      income      balance
-1.151987e+01  1.536641e-05  5.777309e-03
      (Intercept)      income      balance
-1.123242e+01  1.771597e-05  5.512791e-03
      (Intercept)      income      balance
```

```
R Console
-1.218948e+01  3.202779e-05  5.729780e-03
      (Intercept)      income      balance
-1.173123e+01  2.226363e-05  5.765358e-03

ORDINARY NONPARAMETRIC BOOTSTRAP

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

Bootstrap Statistics :
      original      bias      std. error
t1*  -1.154047e+01  1.335891e-02  3.967262e-01
t2*   2.080898e-05  4.875168e-08  4.997011e-06
t3*   5.647103e-03 -1.054066e-05  2.226478e-04
> |
```

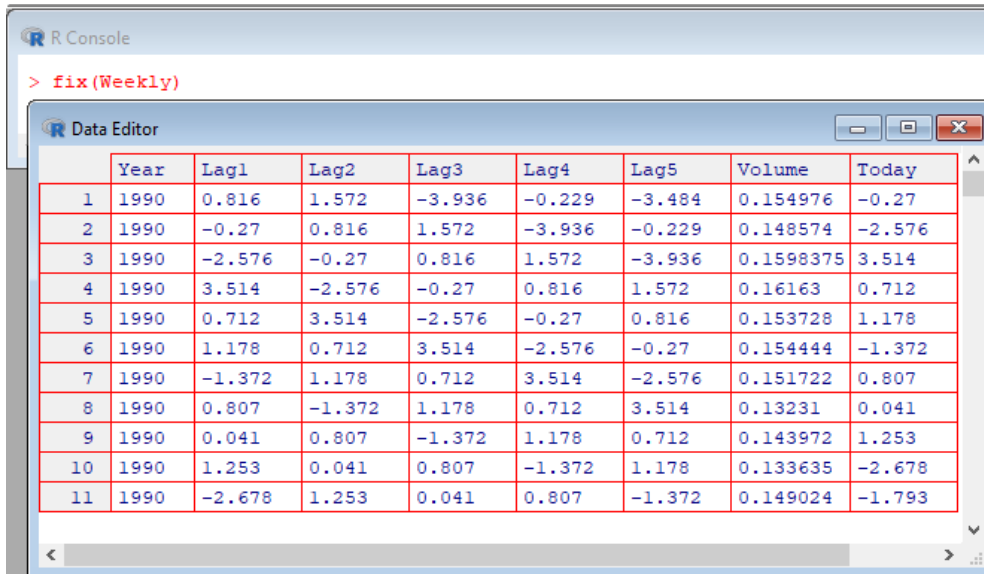
Вона дорівнює $3.97e-01$, $4.99e-06$ та $2.23e-04$ відповідно.

2.4. В результаті можна побачити, що середньоквадратичні відхилення параметрів логістичної регресії в другому випадку менші ніж в першому, але не сильно відрізняються.

Завдання 3. Обчислити оцінку тестової помилки методом LOOCV, використовуючи лише функції `glm()`, `predict.glm()` та цикл `for`. Застосувати такий

підхід для того, щоб обчислити оцінку тестової помилки методом LOOCV для логістичної регресійної моделі на наборі даних Weekly.

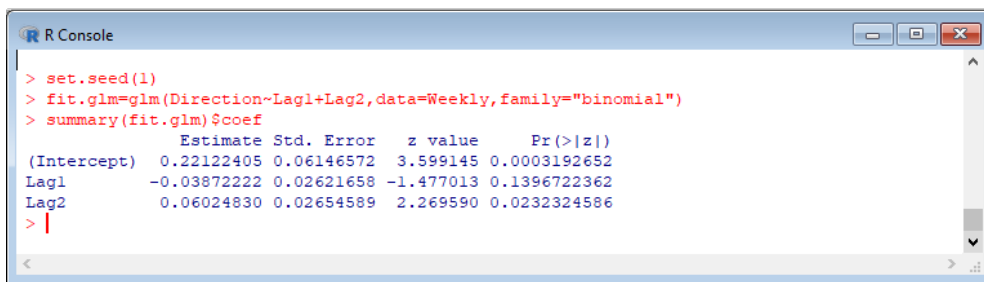
3.1 Розглянула дані Default, які є частиною пакету ISLR:



The image shows an R Console window with the command `> fix(Weekly)` and a Data Editor window displaying the 'Weekly' dataset. The Data Editor window shows a table with 11 rows and 9 columns: Year, Lag1, Lag2, Lag3, Lag4, Lag5, Volume, and Today. The data is as follows:

	Year	Lag1	Lag2	Lag3	Lag4	Lag5	Volume	Today
1	1990	0.816	1.572	-3.936	-0.229	-3.484	0.154976	-0.27
2	1990	-0.27	0.816	1.572	-3.936	-0.229	0.148574	-2.576
3	1990	-2.576	-0.27	0.816	1.572	-3.936	0.1598375	3.514
4	1990	3.514	-2.576	-0.27	0.816	1.572	0.16163	0.712
5	1990	0.712	3.514	-2.576	-0.27	0.816	0.153728	1.178
6	1990	1.178	0.712	3.514	-2.576	-0.27	0.154444	-1.372
7	1990	-1.372	1.178	0.712	3.514	-2.576	0.151722	0.807
8	1990	0.807	-1.372	1.178	0.712	3.514	0.13231	0.041
9	1990	0.041	0.807	-1.372	1.178	0.712	0.143972	1.253
10	1990	1.253	0.041	0.807	-1.372	1.178	0.133635	-2.678
11	1990	-2.678	1.253	0.041	0.807	-1.372	0.149024	-1.793

Побудувала модель логістичної регресії, яка передбачає Direction за допомогою змінних Lag1 та Lag2:

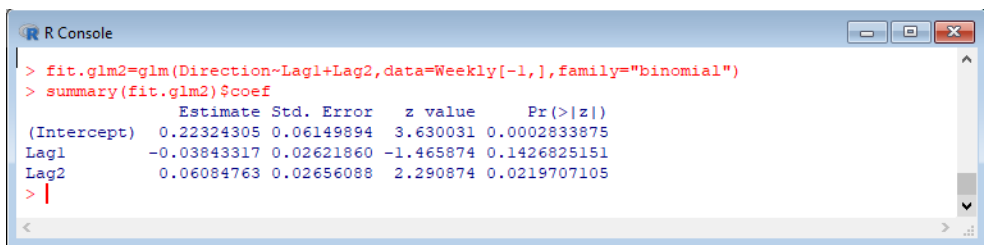


The image shows an R Console window with the following commands and output:

```
> set.seed(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 Побудувала модель логістичної регресії, яка передбачає Direction за допомогою змінних Lag1 та Lag2, використовуючи всі спостереження, крім першого:

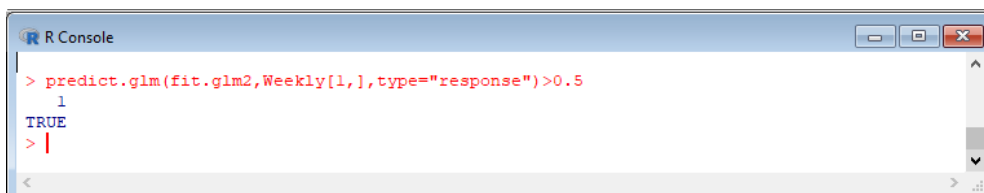


The image shows an R Console window with the following commands and output:

```
> 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 Використайте модель з 3.2, щоб передбачити Direction для першого спостереження:



The image shows an R Console window with the following command and output:

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

Спостереження “Up” було класифіковано неправильно, бо насправді це “Down”.

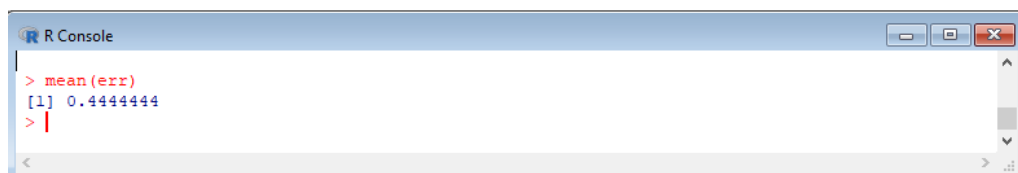
3.4 Написала цикл for від $i = 1$ до $i = n$, де n - число спостережень в наборі даних, який виконує наступні кроки:

- Будує модель логістичної регресії, яка передбачає Direction за допомогою змінних Lag1 та Lag2, використовуючи всі спостереження, крім i -ого.
- Обчислює апостеріорну ймовірність руху ринку вгору для i -го спостереження.
- Використовує апостеріорну ймовірність для i -го спостереження для прогнозування, чи рухатиметься ринок вгору чи ні.
- Визначає, чи допущена помилка при прогнозуванні Direction для i -го спостереження. Якщо була допущена помилка, то вказуємо це як 1, а в іншому випадку - як 0.



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

3.5 Обчислила середнє з n чисел, отриманих у 3.4.4, для того, щоб отримати оцінку LOOCV для тестової помилки:

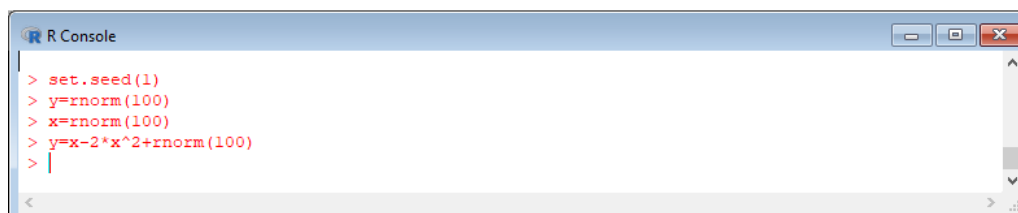


```
> mean(err)
[1] 0.4444444
> |
```

Вона дорівнює 44.4%, що є відносно непоганим результатом.

Завдання 4. Провести перехресну перевірку на змодельованому наборі даних.

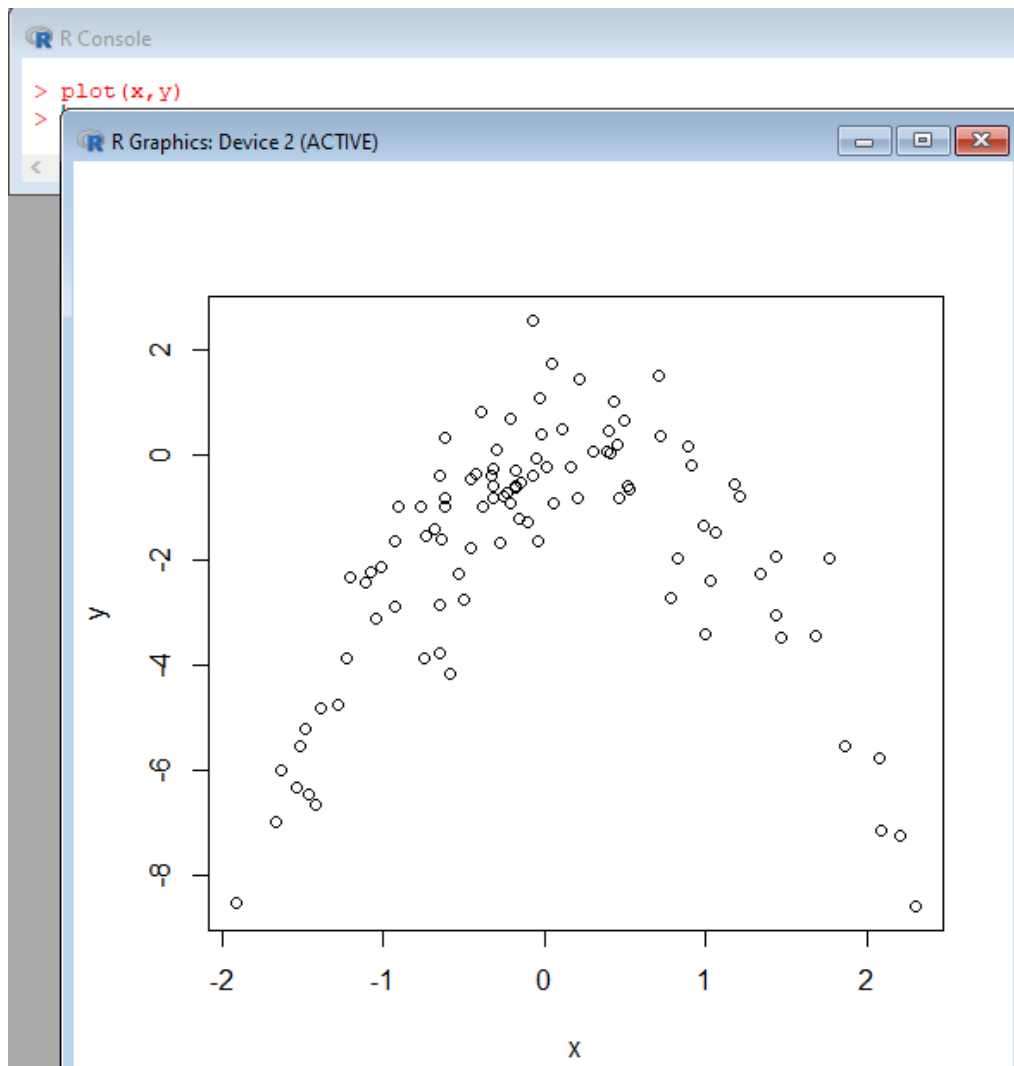
4.1 Створила змодельований набір даних наступним чином:



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

В цьому наборі $n=100$, $p=2$. Модель, що використовувалася для генерації даних у формі рівняння: $y=x-2*x^2+\epsilon$.

4.2 Побудувала діаграму розсіювання X vs Y:



В результаті можна побачити, що залежність квадратична.

4.3 Встановила `random.seed` та обчислила оцінки тестових помилок методом LOOCV, для моделей 4.3.1-4.3.4.

$$4.3.1 \ Y = \beta_0 + \beta_1 X + \varepsilon$$

```
R Console
> coordinates=data.frame(x,y)
> LOOCV=function(seed){
+   fit.glm=glm(y~x)
+   print(round(cv.glm(coordinates,fit.glm)$delta[1],2))
}
```

$$4.3.2 \ Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

```
R Console
+   fit.glm2=glm(y~x+I(x^2))
+   print(round(cv.glm(coordinates,fit.glm2)$delta[1],2))
```

$$4.3.3 \ Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$$

```
R Console
+   fit.glm3=glm(y~poly(x,3))
+   print(round(cv.glm(coordinates,fit.glm3)$delta[1],2))
```


$$4.3.4 \ Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \varepsilon$$

```
R Console
+ fit.glm4=glm(y~poly(x,4))
+ print(round(cv.glm(coordinates,fit.glm4)$delta[1],2))
```

4.3.1-4.3.4 Оцінки тестових помилок методом LOOCV відповідно:

```
R Console
> LOOCV(1)
[1] 5.89
[1] 1.09
[1] 1.1
[1] 1.11
>
```

4.4 Повторила 4.3, використовуючи інший random.seed:

```
R Console
> LOOCV(4)
[1] 5.89
[1] 1.09
[1] 1.1
[1] 1.11
> LOOCV(8)
[1] 5.89
[1] 1.09
[1] 1.1
[1] 1.11
> LOOCV(12)
[1] 5.89
[1] 1.09
[1] 1.1
[1] 1.11
>
```

Результати такі самі, як і в 4.3, бо як ми не ставили б ці елементи, вони принаймні по одному разу все одно будуть використані як тестові.

4.5 Друга модель з 4.3. мала найменшу тестову помилку LOOCV. Це відповідає очікуванням, адже відношення – квадратичне.

4.6 За допомогою функції summary(), по малому значенню p можна побачити, що статистично значимі є перші два коефіцієнти моделей розглянутих у 4.3, а третій і четвертий – ні.

```
R Console
> fit.glm_4_5=glm(y~poly(x,4))
> summary(fit.glm_4_5)$coef
      Estimate Std. Error  t value    Pr(>|t|)
(Intercept) -1.8277074   0.1041467 -17.5493533 1.444977e-31
poly(x, 4)1  2.3164010   1.0414671  2.2241711 2.850549e-02
poly(x, 4)2 -21.0585869   1.0414671 -20.2201171 3.457023e-36
poly(x, 4)3  -0.3048398   1.0414671  -0.2927023 7.703881e-01
poly(x, 4)4  -0.4926249   1.0414671  -0.4730105 6.372907e-01
>
```

Ці результати узгоджуються із зробленими висновками на основі результатів перехресної перевірки, бо статистично значимі є якраз ті, значення LOOCV яких були найменшими.

Завдання 5. Розглянула набір даних Boston з бібліотеки MASS:

R Console

```
> library(MASS)
> fix(Boston)
```

Data Editor

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax
1	0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296
2	0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242
3	0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242
4	0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222
5	0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222
6	0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222
7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311
8	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311
9	0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311
10	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311
11	0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311

5.1 На основі цього набору даних обчислила оцінку середнього змінної medv:

R Console

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

5.2 Визначила стандартну похибку цієї оцінки:

R Console

```
> medv_1=sd(Boston$medv)/sqrt(length(Boston$medv))
> round(medv_1,2)
[1] 0.41
>
```

Вона дорівнює 41%.

5.3 Оцінила стандартну похибку розглянутої вище оцінки середнього за допомогою бутстрапу:

R Console

```
> boot.fn=function(data,index){
+   print(mean(data[index]))
+ }
> boot(Boston$medv,boot.fn,100)
[1] 22.53281
[1] 22.81917
[1] 23.00198
[1] 22.59209
[1] 22.37727
[1] 22.37194
[1] 22.35237
[1] 22.18893
[1] 22.56957
[1] 22.65949
[1] 22.77885
[1] 22.94506
```

```
R Console

[1] 22.87668
[1] 22.00435
[1] 22.58597

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = Boston$medv, statistic = boot.fn, R = 100)

Bootstrap Statistics :
      original    bias      std. error
t1*  22.53281  0.009027668   0.3482331
> |
```

Вона майже така сама, як в 5.2

5.4 На основі бутстрап оцінки побудувала 95% довіри для середнього значення змінної medv:

```
R Console

> t.test(Boston$medv)

One Sample t-test

data:  Boston$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
> |
```

Можна побачити, що результати бутстрап оцінки в порівнянні з результатами t.test не сильно відрізняються.

5.5 На основі цього набору даних обчислила оцінку для медіани змінної medv:

```
R Console

> median(Boston$medv)
[1] 21.2
> |
```

5.6 Визначила стандартну помилку оцінки медіани змінної medv за допомогою бутстрапу:

```
R Console

> boot.fn1= function(data,index){
+   print(median(data[index]))}
> boot(Boston$medv,boot.fn1,100)
[1] 21.2
[1] 21.05
[1] 20.85
[1] 21.8
[1] 21.5
[1] 20.5
[1] 21
[1] 21.1
[1] 21.55
[1] 21.2
```

```
R Console

[1] 21.7
[1] 22
[1] 21.2

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = Boston$medv, statistic = boot.fn1, R = 100)

Bootstrap Statistics :
      original    bias      std. error
t1*      21.2   0.0215   0.3585362
> |
```

Можна побачити, що значення медіани 21.2 і це дорівнює обчисленому у попередньому пункті, а значення похибки досить невелике.

5.7 Використовуючи функцію `quantile()`, на основі цього набору даних обчислила оцінку десятого процентиля змінної `medv`:

```
R Console

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

5.8 Використала бутстрап, щоб оцінити стандартну похибку десятого процентиля змінної `medv`:

```
R Console

> boot.fn2=function(data,index){
+   print(quantile(data[index],c(0.1)))}
> boot(Boston$medv,boot.fn2,100)
10%
12.75
10%
13.1
10%
12.55
10%
11.9
10%
12.05
10%
13.35
10%
13.3
> |
```

```
R Console

12.7
10%
12.65
10%
12.75

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = Boston$medv, statistic = boot.fn2, R = 100)

Bootstrap Statistics :
      original    bias      std. error
t1*      12.75  -0.0045   0.4830509
> |
```

Можна побачити, що значення оцінки десятого процентиля змінної $medv$ 12.75 і це дорівнює обчисленому у попередньому пункті, а значення похибки досить невелике.