Лабораторна робота №6

Виконали: Кузьменко Юрій, Болотов Єгор

Побудова та статистичний аналіз нелінійної множинної регресії.

Опис dataset

Назва dataset:

Spotify Top 10000 Streamed Songs

Link на dataset:

https://www.kaggle.com/datasets/rakkesharv/spotify-top-10000-streamed-songs

Опис dataset та постановку задачі:

Це набір даних, зібраний з веб-сайту Spotify, котрий містить потоки виконавця та кількість прослiховувань (було взято саме топ-10000) Основна мета: вплив факторів на популярність пісні й дізнатись найпопулярніших виконавців та треки.

Змінні та їх опис:

Position - Spotify Ranking

Artist Name - Artist Name

Song Name - Song Name

Days - No of days since the release of the song

Top 10 (xTimes) - No of times inside top 10

Peak Position - Peak position attained

Peak Position (xTimes) - No of times Peak position attained

Peak Streams - Total no of streams during Peak position

Total Streams - Total song streams

```
df <- read_csv("../Spotify_final_dataset.csv")

y <- df$Top_ten_times
x1 <- df$Peak_position_times
x2 <- df$Peak_streams
x3 <- df$Days
x4 <- df$Total_streams
x5 <- df$Peak_position
```

Завдання 1: Нелінійні моделі (A) Побудувати нелінійні моделі; Модель R + (B) Представити графічно;

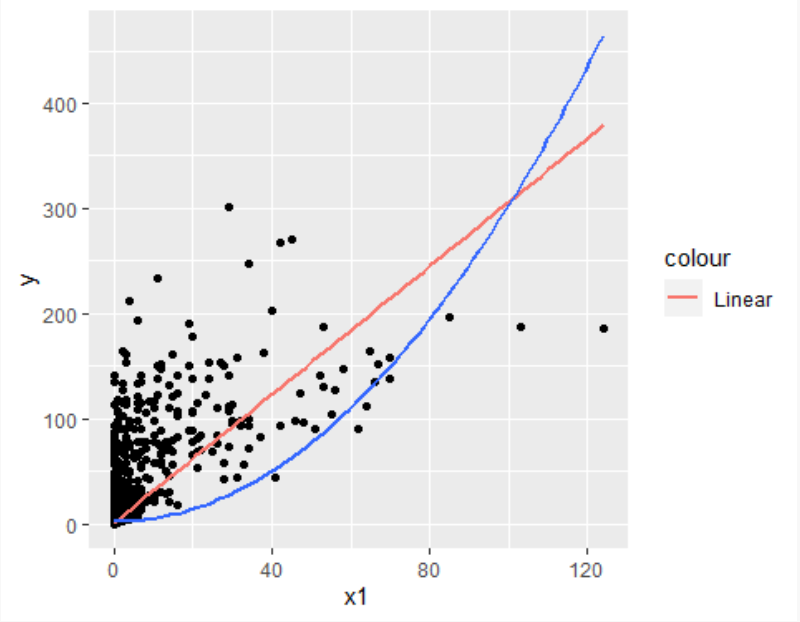
У даному завданні в ролі X будемо використовувати Peak_positon_times У будемо використовувати Top_ten_times

(A) Побудувати нелінійні моделі;

a) Розіювання x та y;
b) Накласти на розсіювання a) лінійну модель 1;
c) Накласти на розсіювання a) модель 2;

1. $y = b0 + b1x$

```
mod_1 <- lm(y~x1)
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(aes(color = "Linear"), method = "lm", se = FALSE)+
  geom_smooth(method = "lm", se = FALSE, formula = y~I(x^2))
```



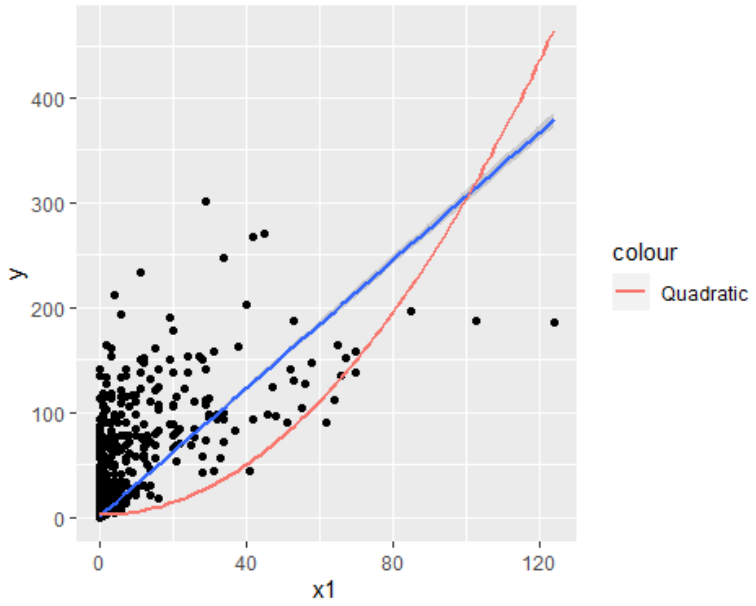

```
summary(mod_1)
```

```
##
## Call:
## lm(formula = y ~ x1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -193.265   -1.535   -1.535   -1.535  212.125
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.53490    0.10593   14.49   <2e-16 ***
```

```
## x1           3.04621    0.02923  104.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.09 on 11082 degrees of freedom
## Multiple R-squared:  0.495,  Adjusted R-squared:  0.495
## F-statistic: 1.086e+04 on 1 and 11082 DF,  p-value: < 2.2e-16
```

2. $y = b0 + b1x^2$

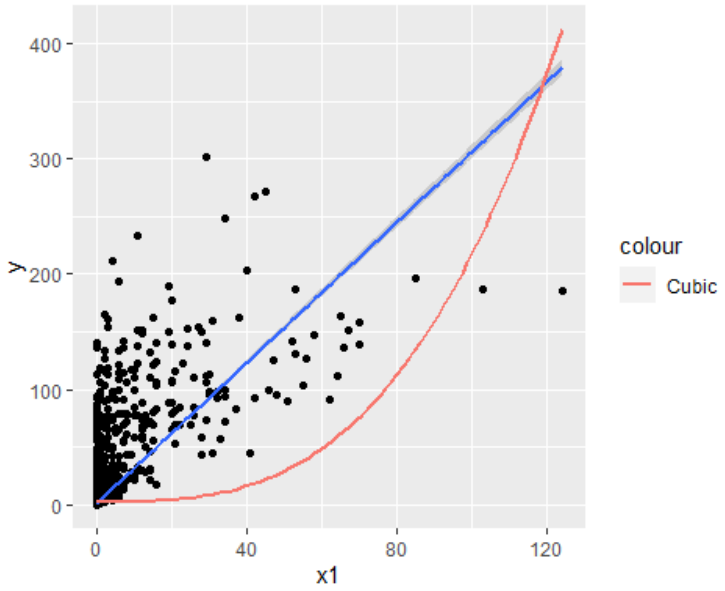
```
mod_2 <- lm(y~I(x1^2))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Quadratic"), method = "lm", se = FALSE, formula =
y~I(x^2))
```




```
summary(mod_2)
## Call:
## lm(formula = y ~ I(x1^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -278.770   -2.318   -2.318   -2.318  274.387
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.318497   0.131099   17.68   <2e-16 ***
## I(x1^2)     0.030076   0.000538   55.90   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Residual standard error: 13.78 on 11082 degrees of freedom
## Multiple R-squared:   0.22,  Adjusted R-squared:  0.2199
## F-statistic:  3125 on 1 and 11082 DF,  p-value: < 2.2e-16
```

3.$y = b0 + b1x$^3

```
mod_3 <- lm(y~I(x1^3))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Cubic"), method = "lm", se = FALSE, formula =
y~I(x^3))
```

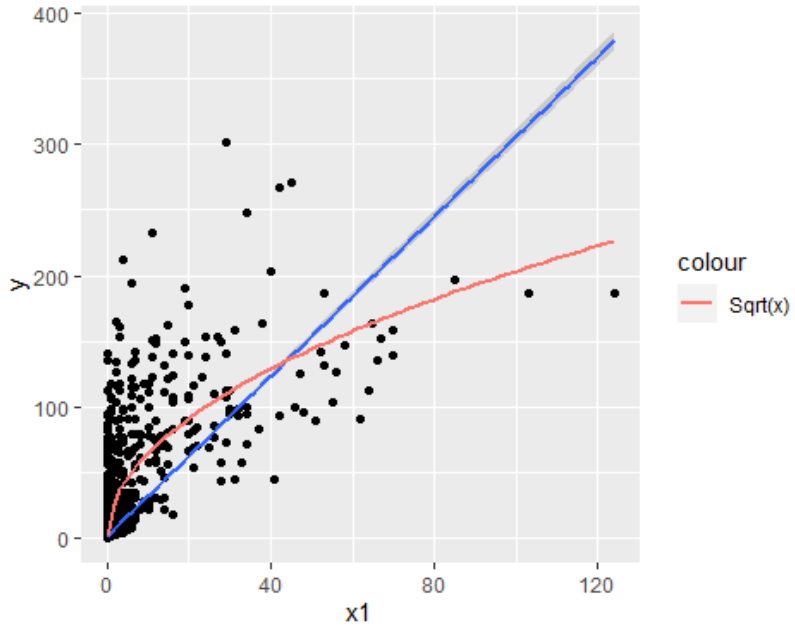



```
summary(mod_3)

##
## Call:
## lm(formula = y ~ I(x1^3))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -226.362   -2.548   -2.548   -2.548  294.210
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.548e+00  1.404e-01   18.15   <2e-16 ***
## I(x1^3)     2.149e-04  5.993e-06   35.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.77 on 11082 degrees of freedom
## Multiple R-squared:  0.104,  Adjusted R-squared:  0.1039
## F-statistic:  1286 on 1 and 11082 DF,  p-value: < 2.2e-16
```

4. $y = b0 + b1*\text{sqrt}(x)$

```
mod_4 <- lm(y~I(x1^(0.5)))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Sqrt(x)"), method = "lm", se = FALSE, formula =
y~I(x^0.5))
```



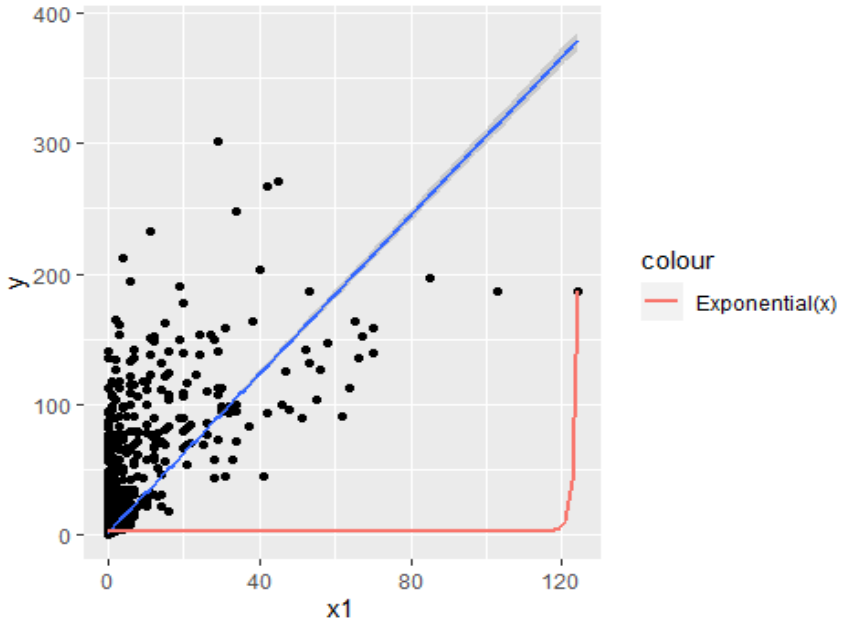

```
summary(mod_4)

##
## Call:
## lm(formula = y ~ I(x1^(0.5)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.449  -0.616  -0.616  -0.616 192.192
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.61595    0.09077   6.786 1.21e-11 ***
## I(x1^(0.5)) 20.27647    0.14592 138.960  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.423 on 11082 degrees of freedom
## Multiple R-squared:  0.6354, Adjusted R-squared:  0.6353
## F-statistic: 1.931e+04 on 1 and 11082 DF,  p-value: < 2.2e-16
```

5. $y = b0 + b1\ \exp(x)$

```
mod_5 <- lm(y~exp(x1))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Exponential(x)"), method = "lm", se = FALSE, formula
= y~exp(x))

## `geom_smooth()` using formula = 'y ~ x'
```



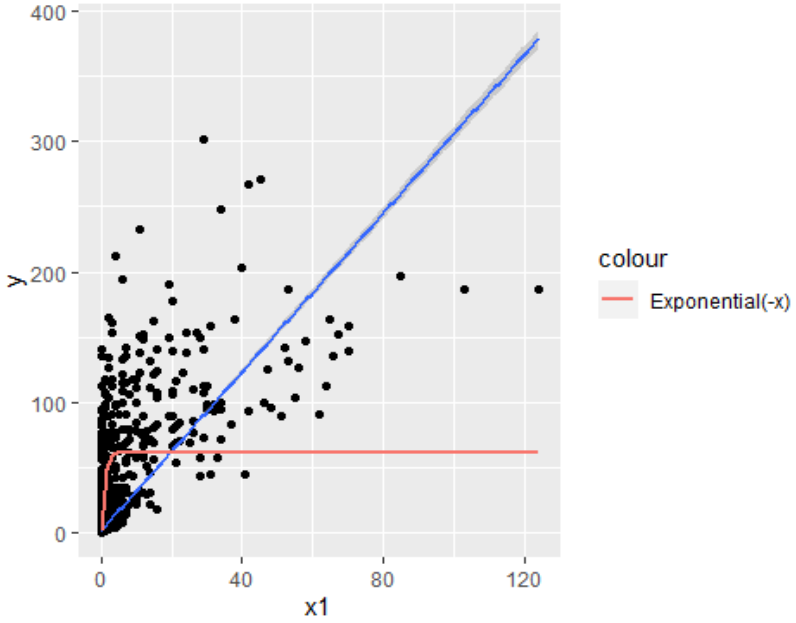

```
summary(mod_5)

##
## Call:
## lm(formula = y ~ exp(x1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -2.697  -2.697  -2.697  -2.697 299.303
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.697e+00  1.473e-01   18.31   <2e-16 ***
## exp(x1)     2.574e-52  2.178e-53   11.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.51 on 11082 degrees of freedom
## Multiple R-squared:  0.01245,    Adjusted R-squared:  0.01236
## F-statistic: 139.7 on 1 and 11082 DF,  p-value: < 2.2e-16
```

6. $y = b0 + b1\ \exp(-x)$

```
mod_6 <- lm(y~exp(-x1))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Exponential(-x)"), method = "lm", se = FALSE, formula
= y~exp(-x))

## `geom_smooth()` using formula = 'y ~ x'
```



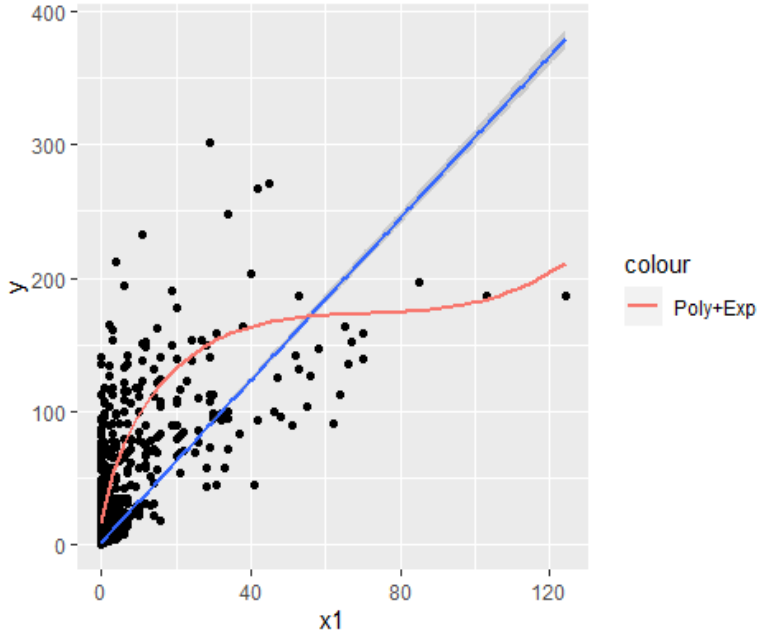

```
summary(mod_6)

##
## Call:
## lm(formula = y ~ exp(-x1))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -56.130  -0.473  -0.473  -0.473 240.757
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  61.2433     0.6017  101.78   <2e-16 ***
## exp(-x1)    -60.7706     0.6146  -98.88   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.37 on 11082 degrees of freedom
## Multiple R-squared:  0.4687, Adjusted R-squared:  0.4687
## F-statistic:  9777 on 1 and 11082 DF,  p-value: < 2.2e-16
```

7. $y = b0 + b1((X + 1)3 - \log(X + 5))$

```
mod_7 <- lm(y~I((x1+1)^3-log(x1+5)))
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(method=lm)+
  geom_smooth(aes(color = "Poly+Exp"), method = "lm", se = FALSE, formula =
y~I(poly(x+1,3)-log(x+5)))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
summary(mod_7)
```

```
##
## Call:
## lm(formula = y ~ I((x1 + 1)^3 - log(x1 + 5)))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -230.643   -2.542   -2.542   -2.542  293.734
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  2.542e+00  1.401e-01   18.14   <2e-16 ***
## I((x1 + 1)^3 - log(x1 + 5)) 2.120e-04  5.808e-06   36.51   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.74 on 11082 degrees of freedom
## Multiple R-squared:  0.1073, Adjusted R-squared:  0.1073
## F-statistic:  1333 on 1 and 11082 DF,  p-value: < 2.2e-16
```

(C) Порівняти моделі 1 – 7 та зробити висновки про найкращу модель.

№	Модель	R^2	F	RSE
1	$y = b0 + b1x$	0.495	1.086e+04	11.09
2	$y = b0 + b1x$^2	0.22	3125	13.78
3	$y = b0 + b1x$^3	0.104	1286	14.77
4	$y = b0 + b1\sqrt{x}$	0.6354	1.931e+04	9.423
5	$y = b0 + b1\ \exp(x)$	0.01245	139.7	15.51
6	$y = b0 + b1\ \exp(-x)$	0.4687	9777	11.37
7	$y = b0+b1((X + 1)^3-\log(X + 5))$	0.1073	1333	14.74

Порівнявши моделі можна побачити, що моделі 2,3,5,7 є поганими моделями, 1,6 в цілком можна використовуватити, але найкращою моделлю серед представлених є 4 модель

Завдання 2: Нелінійні моделі за допомогою класичного поліному

(A) Побудуйте поліноми $Y = \beta 0 + \beta 1X + \cdots + \beta kXk + \varepsilon$ до 5-го ступеня та 10-й;

```
mod_5p <- lm(y ~ poly(x1, 5, raw = TRUE))
mod_10p <- lm(y ~ poly(x1, 10, raw = TRUE))
```

(B) Визначте оптимальний поліном за допомогою BIC(*);

```
BIC(mod_1,mod_2,mod_3,mod_5p,mod_10p)
```

```
##          df      BIC
## mod_1     3 84816.05
## mod_2     3 89635.80
## mod_3     3 91172.05
## mod_5p    7 81192.24
## mod_10p  12 81019.56
```

mod_10 краща (менше - краще)

(C) Побудуйте:

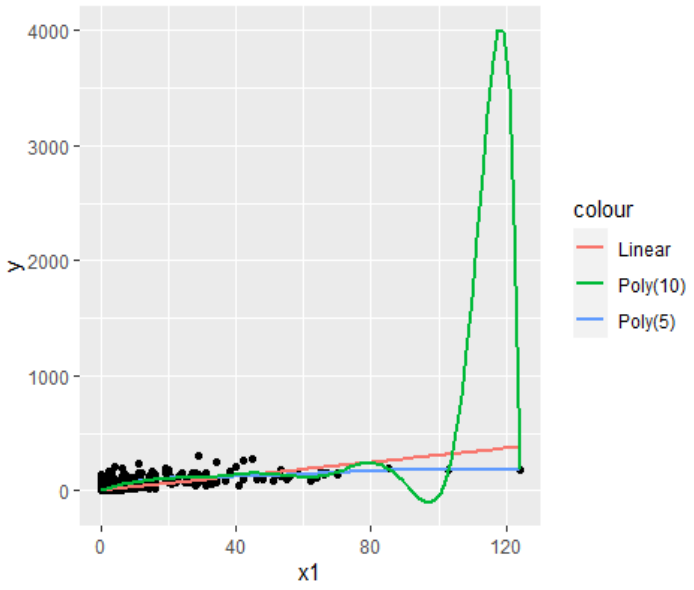
a) розсіювання x та y;

b) накладіть пряму лінію;

c) поліноміальні моделі.

```
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(aes(color = "Linear"), method = "lm", se = FALSE)+
  geom_smooth(aes(color = "Poly(5)"), method = "lm", se = FALSE, formula = y ~
poly(x, 5,raw = TRUE))+
  geom_smooth(aes(color = "Poly(10)"), method = "lm", se = FALSE, formula = y ~
poly(x, 10, raw = TRUE))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Завдання 3: Нелінійні моделі за допомогою поліному Лежанра.

(A) Побудуйте ортогональні поліноми Лежанра до 5-го ступеня та 10-й;

```
mod_5pl <- lm(y ~ poly(x1, 5, raw = FALSE))
mod_10pl <- lm(y ~ poly(x1, 10, raw = FALSE))
```

(B) Визначте оптимальний поліном за допомогою BIC(*);

```
BIC(mod_1,mod_2,mod_3,mod_5pl,mod_10pl)
```

```
##           df      BIC
## mod_1      3 84816.05
## mod_2      3 89635.80
## mod_3      3 91172.05
## mod_5pl    7 81192.24
## mod_10pl 12 81019.56
```

Найкраща модель з 10 ступенями

(C) Побудуйте:

a) розсіювання *x* та *y*;

b) накладіть пряму лінію;

c) поліноміальні моделі Лежанра:

```
ggplot(data = df, aes(x = x1, y = y)) +
  geom_point()+
  stat_smooth(aes(color = "Linear"), method = "lm", se = FALSE)+
  geom_smooth(aes(color = "PolyL(5)"), method = "lm", se = FALSE, formula = y ~
poly(x, 5, raw = FALSE))+
  geom_smooth(aes(color = "PolyL(10)"), method = "lm", se = FALSE, formula = y ~
poly(x, 10, raw = FALSE))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```