Regresszió - Predikció

Lukovszki Csaba

2021. március

Tartalomjegyzék

1. Regresszió **1**
1.1. Gépi tanulás és a regresszió 3

2. Lineáris regresszió **3**

3. A regresszió kiértékelése **6**
3.1. Változók közötti összefüggés 6
3.2. Hiba eloszlása 6
3.3. Model kiértékelés a `plot()` függvénnyel 8
3.4. Mérőszámok 10
3.5. Predikció 11
3.6. Feladatok 12
3.6.1. Egyváltozós lineáris regresszió 12
3.6.2. Többváltozós lineáris regresszió 16
3.6.3. Predikció lineáris regresszió esetében 19

4. Nemlineáris regresszió **19**
4.1. Visszavezetés lineáris regresszióra 20
4.2. Nemlináris függvény alapján 23
4.3. Feladatok 25
4.3.1. Egyváltozós nemlineáris regresszió 25

5. Logisztikus regresszió **26**
5.1. Binomiális prediktorok 27
5.2. Diszkrét, kategorikus prediktorok 28

```
## Loading required package: carData
```

```
require(graphics)
```

1. Regresszió

Két változó között számított kovariancia, illetve korreláció megmutatja, hogy két változó között van-e kapcsolat, a regressziószámítás viszont azt mutatja meg, hogy milyen kapcsolat áll fenn közöttük. A regresszió egy statisztikai technika két, vagy több változó közözzi összefüggés megállapítására.

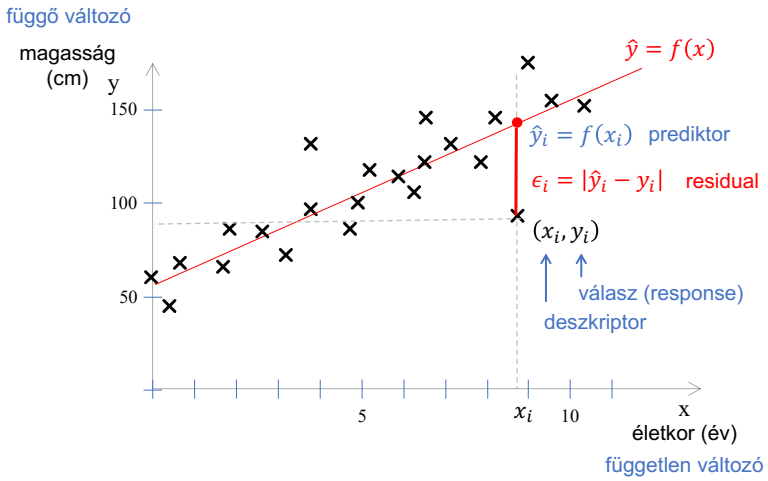
A regressziószámítás vagy regresszióanalítis során két vagy több véletlen változóként modellezhető minta között fennálló kapcsolatot modellezzük. A regressziószámítás során függő változó (y) viselkedését viszgáljuk független változó (x) vagy változókhoz képest ($x_1, x_2, ...x_N$).

A gyakorlatban számos helyen találkozhatunk egyes paraméterek közötti összefüggésekkel. Jellemzően a regressziószámítást széles körben a banki hitelbírálatok során alkalmazták, ahol vizsgáljuk a hitel rizikó függését az ügyfél életkorától, bevételeitől, költéseitől, az eltartottak számától, a teljes banki betététől stb... Ugyanakkor az élet számos területén találkozhatunk egyéb példákkal.

A 1. ábra egy példát illusztrál. Gyerekek életkorát és magasság értékeit ábrázoljuk egy koordináta rendszerben, ahol x az életkor, mint független változó, és y a magasság, mint függő változó. Minden egyes életkor-magasság (x_i, y_i) pár egy-egy gyerek értékeit mutatja, azaz egy objektumot alkot. Jól látható, hogy az életkor és a magasság között összefüggés van, hiszen minél nagyobb az életkor, a várható magasság is egyre nagyobb. Ekkor, matematikailag létezik két vektorunk $x = (x_1, x_2, ...x_N)$, és $y = (y_1, y_2, ...y_N)$, ahol az i-dik elemek egy adott objektumhoz tarozó független és függő változók.

A regressziószámítás során keressük azt a $\hat{y} = f(x)$ függvényt, mely a legkisebb hibával közelíti az $(x_i, y_i), i = 1..N$ mintákat, ahol a hiba (*residual*) az adott x_i értékhez (*deszkriptor*) tartozó mintaérték y_i (*response*), valamint a regressziós függvény által becsült érték $\hat{y}_i$ (*predictor*) abszolút különbsége.

$$\epsilon_i = |\hat{y}_i - y_i|$$



1. ábra. Regresszió szemléltetése

Gyakorlatban a regressziószámítás során a négzetes hibát $(\hat{y}_i - y_i)^2$ minimalizáljuk.

A független változók számossága szerint a következő regressziókról beszélhetünk:

- *Egyszerű regresszió* (Simple Regression): Egy független változó hatását vizsgáljuk egy függő változó értékén
- *Többváltorzós regresszió* (Multivariate Regression): A független változók száma kettő, vagy akár több is lehet.

Amennyiben több függő változóról beszélünk, minden egyes függő változóra állapítjuk meg a regressziót, azaz változónként külön feladatról, az adott összegüggésre speciális modellről beszélünk.

A változók közötti összefüggések szerint a következő regressziókról beszélhetünk:

- *Lineáris regresszió* (Linear regression)
- *Nemlineáris regresszió* (Non-linear regression)
  - *Explonenciális* (Exponential)

- *Hatványfüggvénnyel leírható*
- *Hatványpolinommal leírható* (Polinomial)
- *Hiperbolikus* (Hyperbolic)
- *Logisztikus* (Logistic)

A regresszió alkalmazása körültekintést igényel! Nem minden esetben alkalmas egy adathalmaz az elemei közötti regressziós összefüggés megállapítására. Alkalmazzuk a regressziót abban az esetben, ha valamilyen kérdésre szeretnénk választ kapni, vagy feltáró elemzés során amennyiben a két változó közötti nagy korrelációt tapasztalunk.

1.1. Gépi tanulás és a regresszió

A regresszió a irányított gépi tanulás egyik eszköze. Ezért gyakorta használjuk a rendelkezésünkre álló adathalmazt egyszerre tanításra és tesztelésre, úgy, hogy az adathalmazunkat a következő halmazokra bontjuk:

$$Adathalmaz = Tanítóhalmaz(70\%) + Validálóhalmaz(15\%) + Teszthalmaz(15\%)$$

Az egyes részhalmazokat a következőképpen használhatjuk:

- *Tanító halmaz*: A tanító halmazt használjuk a modell építésére
- *Validáló halmaz*: A validáló halmazt használhatjuk paraméteroptimalizálásra, vagy modellek közüli választásra. Amennyiben nincs szükség validációra a vonatkozó 15%-ot használhatjuk a teszt adathalmazunk kibővítésére.
- *Teszt halmaz*: A modellünk kiértékelésére használhatjuk. Modellünk teszthalmazon való alkalmazása egy valóságos alkalmazási esetet reprezentál.

2. Lineáris regresszió

Az egymással összefüggő attribútumok közötti kapcsolat leírását a regressziószámítás segítségével tehetjük meg. **Egyszerű lineáris regresszió** esetében az y függő változó és az x független változó között lineáris kapcsolatot feltételezünk, a függő változó a független változó egyes mintáinak (y_i, x_i) lineáris függvényként írható fel a következőképpen.

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

A fenti egyenlet egy független és egy függő változó közötti lineáris kapcsolatot ír le, mely az `x-y` síkban egy egyenest definiál. Ennek alapján a prediktor értékek $(\hat{y}_i)$ a következőképpen írhatók le.

$$\hat{y}_i = \beta_0 + \beta_1 x_i$$

Többváltozós lineáris regresszió esetében az y függő változó értéke több, esetünkben p darab független változó értéktől is függ $(x_1, x_2, ...x_p)$. Ebben ay esetben a mintákra vonatkozó lineáris összefüggés a következőképpen írható fel.

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + ... + \beta_p x_{p,i} + \epsilon_i$$

Az egyenlet egy hipersíkot definiál, ahol ϵ_i a regressziós hipersík hibatagja (reziduálja) az adott mintára vonatkozóan.

Ebben az esetben a prediktor értéke (a hipersík) a következőképpen formalizálható.

$$\hat{y}_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + ... + \beta_p x_{p,i}$$

Mind a két esetben a $\beta_0, \beta_1 \ldots \beta_p$ együtthatók (koefficiensek) meghatározása a hibataok négyzetösszegének ($\epsilon = \sum_i \epsilon_i^2$) minimalizálásával történik.

$$\sum_{i=1}^{N} \epsilon_i^2 = \sum (y_i - \hat{y}_i)^2$$

, ahol N az adatpontok száma.

Az együtthatók becslésére a következő eljárások alkalmazhatók:

- Legkisebb négyzetek módszere (Ordinary Least Squares - OLS)
- Általánosított legkisebb négyzetek összege (Generalized Least Squares - GLS)
- Általánosított momentumok módszere (Generalized method of Momments - GMM)
- A legnagyobb valószínűség módszere (Maximum Likelihood estimation - ML)
- Ridge regression
- Lassoo Regression

A következőkben nézzünk meg egy egyszerű lineáris regressziót.

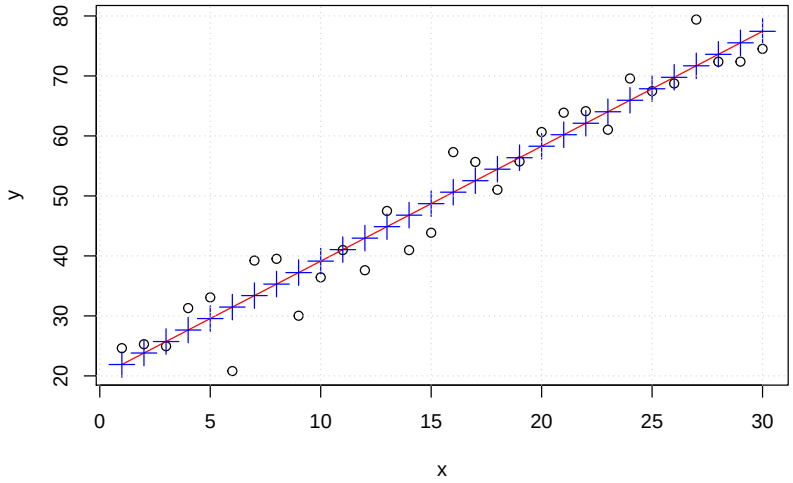
```
# Mintahalmaz előállítása normális eloszlás szerinti hibával
x <- 1:30
y <- 2 * x + 20 + rnorm(30, 0, 5)
# Lineáris regresszió (formula: y ~ x)
model.lin <- lm(y ~ x)
summary(model.lin)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.6669  -2.9771  -0.2369   3.4258   7.7085
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.98126    1.65344   12.09 1.26e-12 ***
## x            1.91517    0.09314   20.56  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.415 on 28 degrees of freedom
## Multiple R-squared:  0.9379, Adjusted R-squared:  0.9357
## F-statistic: 422.8 on 1 and 28 DF,  p-value: < 2.2e-16
```

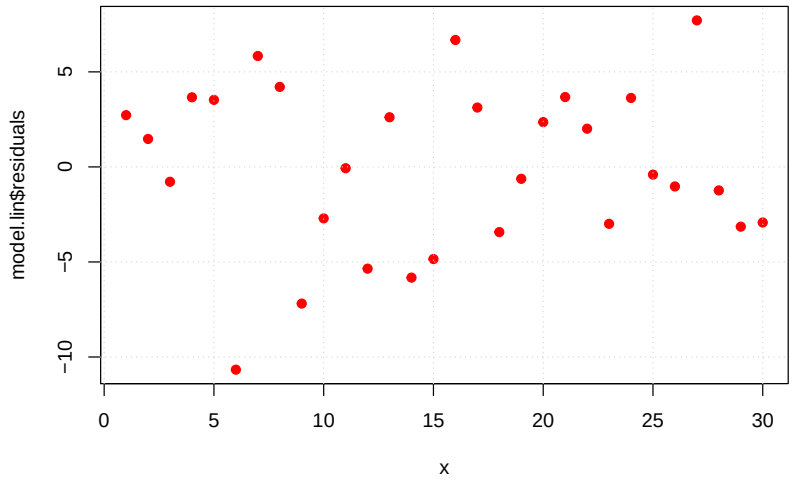
```
attributes(model.lin)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

```r
# --- Ábrázolás
plot(x, y)
# Az lm modell számolt értékei
lines(x, model.lin$fitted.values, col = "red", pch = 4, cex = 3) # pch:4=x
# Az együtthatók alapján
points(x, coef(model.lin)["x"] * x + coef(model.lin)["(Intercept)"],
       col = "blue", pch = 3, cex = 2)
grid()
```




```r
# A hibák ábrázolása
plot(x, model.lin$residuals, pch = 19, col = "red")
grid()
```



A `summary()` függvény segítségével megtudhatjuk a hibatagok eloszlásának kvantiliseit, valamint az egyes független változókhoz tartozó koefficiens és az `y` tengely metszéspontját. Az `attributes()` függvény mutatja a model attribútumait.

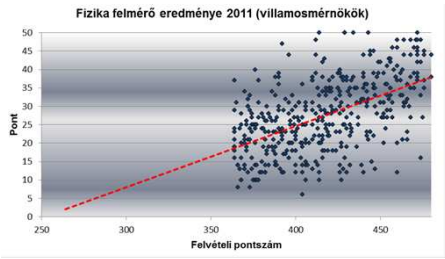
Az `lm()` függvény fontosabb attribútumai:

- `$coefficients`: A lineáris kötelítés együtthatói (intercept: az y tengely metszése, x: meredekség). Az együtthatók értékeit a `coef()` függvénnyel is lekérdezhetjük.
- `$residuals`: A regressziós egyenes, valamint a valós értékek közötti maradvány, hiba. Az értéke a `residuals()` függvénnyel is lekérdezhető.
- `$fitted.values`: A megadott független változóhoz tartozó függvényértékek a regressziós egyenes értékeinek megfelelően.

- `$model`: Az eredeti függvényértékek.

3. A regresszió kiértékelése

A regresszió alkalmazásával nagyon körültekintően kell eljárni, mert gyakran téves következtetéseket is levonhatunk a felelőtlen alkalmazása során. A 2. ábrán egy elhamarkodott regressziós ábrázolást láthatunk.



2. ábra. Regresszió hibás alkalmazása

A regressziós modellünk megfelelőségeinek a feltételeként a következőket fogalmazhatjuk meg.

1. *Összefüggés*: A független és függő változók között létezzen összefüggés. Lineáris regresszió esetében fontos a lineáris kapcsolat. Ezt a használt változók (vagy azok megfelelő függvény szerinti transzformáljai) közötti korreláció segítségével vizsgálhatjuk meg.
2. *Függetlenség*: Az egyes mintákra számolt hibatagok (residual-ok) értékei legyenek egymástól függetlenek.
3. *Homogenitás*: A hibatagok szórása a teljes értelmezési tartományon homogén legyen.
4. *Normalitás*: A minták hibája kövesse a normális eloszlást.

3.1. Változók közötti összefüggés

A változók közötti lineáris összefüggést legyeszerűbben a változók közötti korreláció értékével vizsgálhatjuk.

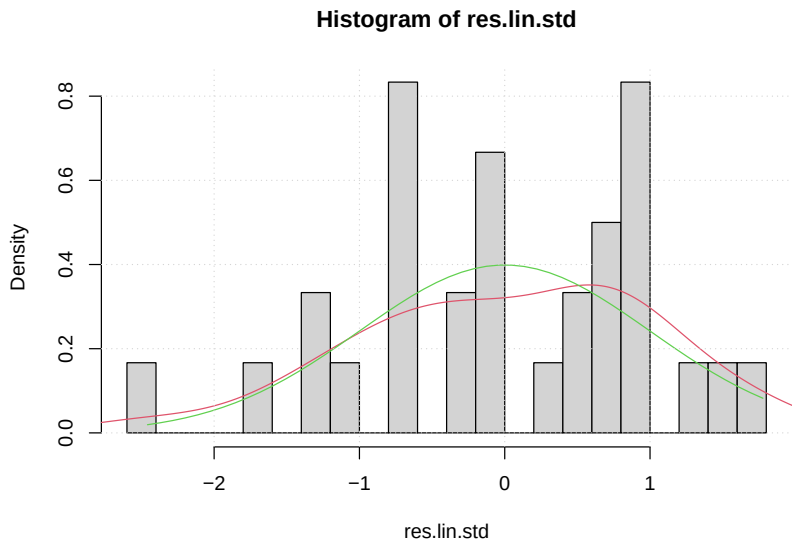
3.2. Hiba eloszlása

A regresszió diagnosztikájának érdekében nézzük meg a korábbi regressziónk hibátagját. A hiba eloszlás vizsgálatánál elsősorban arra vagyunk kíváncsiak, hogy a hibatagok eloszlása normális eloszlás szerinti-e.

- *Várható érték*: Várható értékként 0 értéket várunk.
- *Szórás*: Alacsony szórást várunk egy megfelelő regresszió esetében.
- *Eloszlás vizsgálat*: Arra vagyunk kíváncsiak, hogy a residual értéke normális eloszlás szerinti-e. Ezt vizsgálhatjuk a normalitás vizsgálatánál tanult módon.

```
# A közelítési hibák ábrázolása
res.lin <- residuals(model.lin)
# A közelítés hibájának várható értéke, (m = 0)
res.lin.m <- mean(res.lin)
# A közelítés hibájánka szórása, (v = 10)
res.lin.s <- sd(res.lin)
# --- Az eloszlás normalitás vizsgálata
# A hiba eloszlás standardizálása
res.lin.std = (res.lin - res.lin.m) / res.lin.s
# A hiba hisztogramja
```

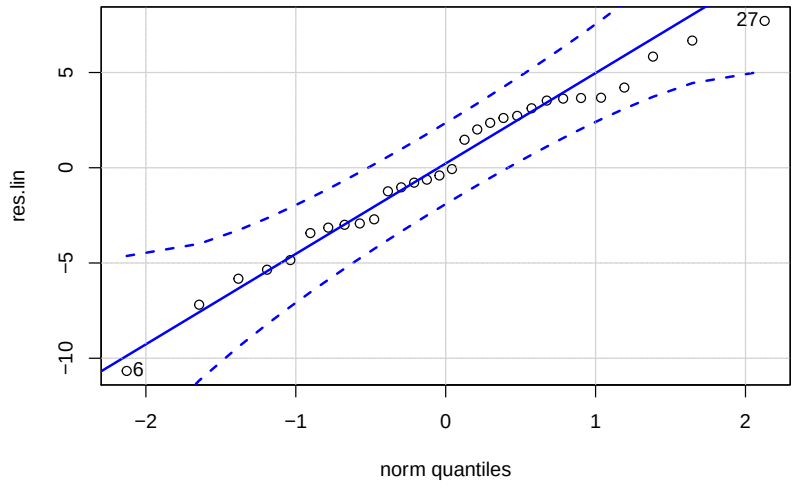
```
hist(res.lin.std, freq = F, breaks = 20)
# Sűrűségfüggvénye
lines(density(res.lin.std), col = 2)
# Standard (mean=0, std=1) normális eloszlás készítése
xfit <- seq(min(res.lin.std), max(res.lin.std), length = 50)
yfit <- dnorm(xfit, 0, 1)
lines(xfit, yfit, col = 3)
grid()
```




```
# QQ plot
qqPlot(res.lin)
```

```
## [1]  6 27
```

```
grid()
```




```
# Shapiro-Wilk teszt alkalmazása
# az eredmény szerint a hibánk normál eloszlásúnak tekinthető
library("stats")
shapiro.test(res.lin)
```

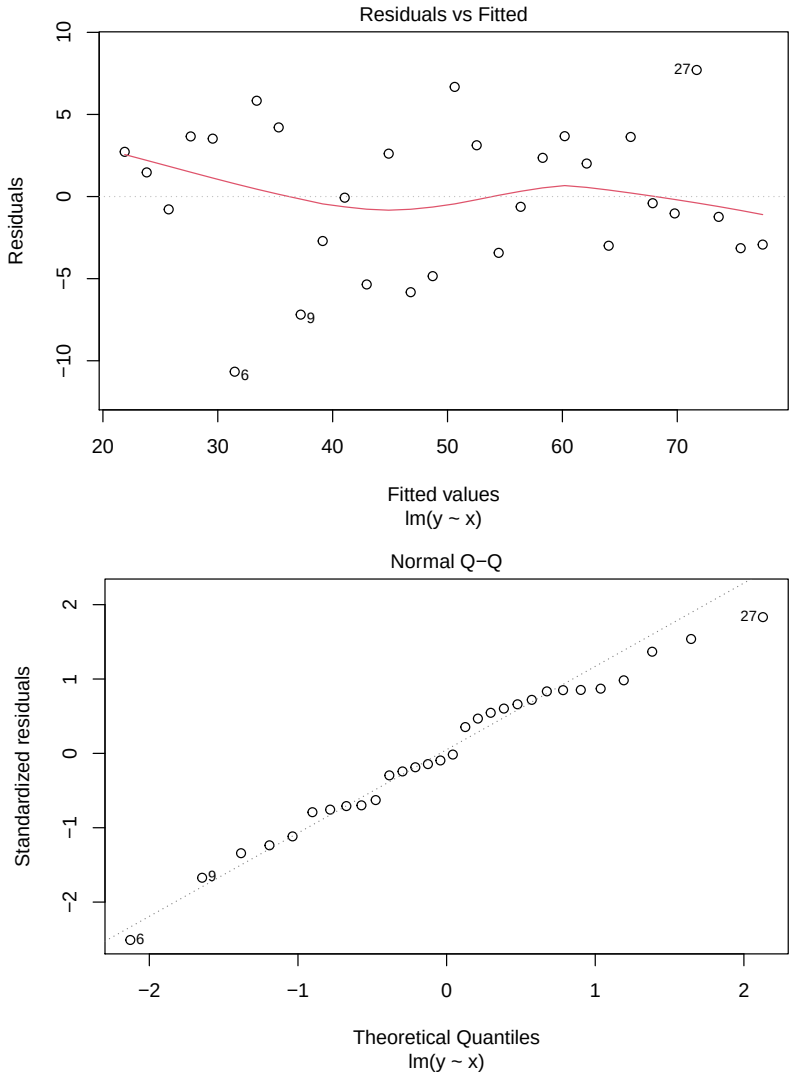
```
##
##  Shapiro-Wilk normality test
```

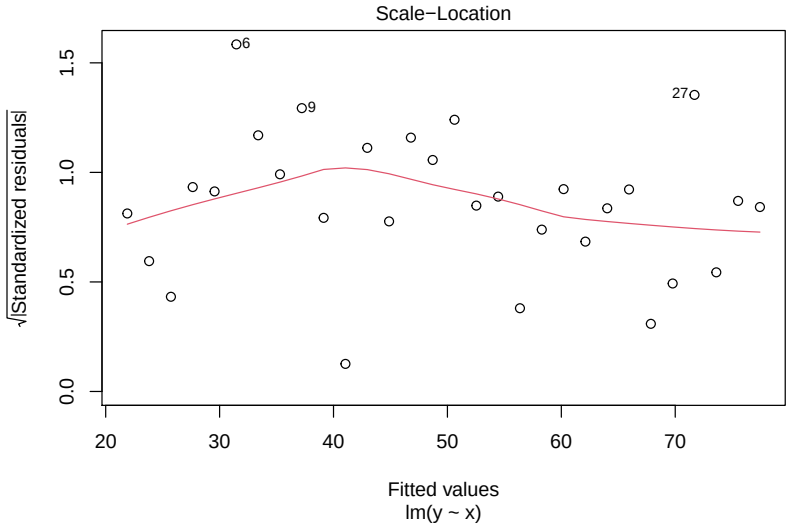
```
##
## data:  res.lin
## W = 0.97516, p-value = 0.6873
```

3.3. Model kiértékelés a `plot()` függvénnyel

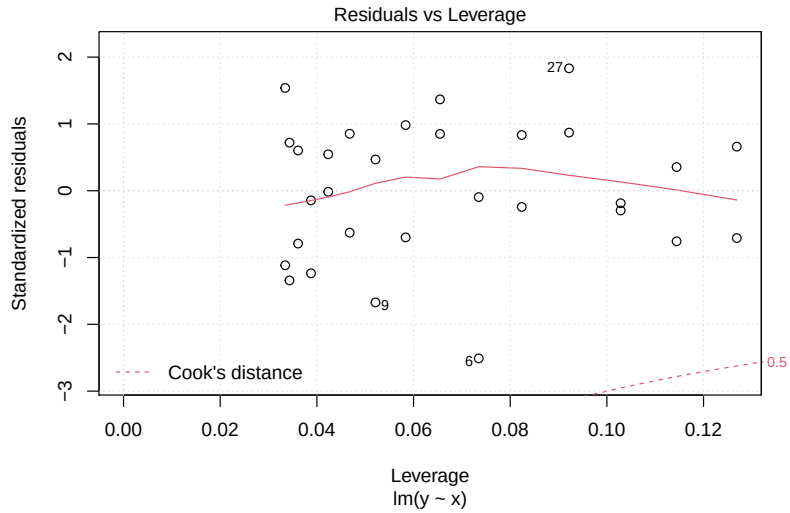
A lineáris regresszió eredményeinek ábrázolására használható a `plot()` függvény is. A `which` paraméter megaásával beállíthatjuk, hogy mely adatok ábráját szeretnénk látni.

```
plot(model.lin)
```






```
#plot(model.lin, which=c(1,2))
grid()
```



Az `lm()` függvény által nyújtott vizuális adatok:

- *Residuals vs. Fitted (1)*: Az ábra a hibatagokat ϵ_i-t ábrázolja a $\hat{y}_i$ függvényében.
- *Normal Q-Q (2)*: A QQ plot a hibatagok (ϵ_i) standardizált és elméleti normál eloszlások $\mathcal{N}(0,1)$ összehasonlítására
- *Standard residuals vs. Fitted (3)*: A hibatagok standardizált residual értékeit ábrázolja a prediktor értékek függvényében.
- *Standard residuals vs. Leverage (4)*: Az egyes hibatagok standardizált értékeit ábrázolja a befolyás függvényében.

A *befolyás* (leverage) megmutatja, hogy az egyes mintáknak mekkora befolyása lehet a regressziós modell kialakítására. Ezért, amennyiben egy mintának nagy a residual értéke (4.`y` tengely) és nagy a befolyása is (4.`x` tengely), azoka pontok nagyon meghatározzák a regressziós modell értékét. Az (4) ábra segítségével ezeket határozhatjuk meg.

Az (1) és (3) ábrák segítségével állapíthatjuk meg, hogy a residual értékek mennyire viselkednek homogén módon. Erre a (3) ábra alkalmasabb, mivel standardizált értékeket tartalmaz.

A (2) ábrával lehetőségünk nyílik, hogy a residual eloszlását egy standard normális eloszlással hasonlíthassuk össze.

A (4) ábra segítségével azonosíthatjuk azokat a mintákat, melyeknek túl nagy a befolyása a regresszió jellegének meghatározása során.

3.4. Mérőszámok

Az egyszerű értkelés érdekében a regresszió általi közelítés hibáját egyszerű jellemzők formájában értékeljük ki, ezek jellemzően a következők lehetnek.

3.4.0.1. MAE (Mean Absolute Error) A MAE mérőszám esetében nem számít az eltérés irány, és valójában a hiba abszolút értékének várható értékét adja meg.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

A MAE mérőszám a mintákra számolt *abszolút közepes eltérés*.

3.4.0.2. MAPE (Mean Absolute Percentage Error) A MAPE egy statisztikai mérőszám, ami megmutatja, hogy milyen pontos a regressziós modell. A jellemző egy arányszám, mely a tényleges értékhez képset százalékosan mutatja meg a hiba várható értékét.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

3.4.0.3. RMSE (Route Mean Square Error) A mérőszám megmutatja a négyzetes eltérések átlagának négyzetgyökét. A statisztikai jellemzők közül ezt a jellemzőt használjuk a leggyakrabban és a legszélesebb körben.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

Az RMSE a mintákra számolt *szórás* értéke.

3.4.0.4. R négyzet Gyakran alkalmazott mérőszám a regresszió kiértékelésére az R négyzet metóduson alapul. Ennek során a hibatagok értékét a minták értékeinek átlagtól való eltérésének függvényében vizsgáljuk. Az R^2 értéke a következőképpen származtatható.

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

, ahol $\bar{y}$ a minták átlagát jelöli.

Az R^2 értéke minél közelebb van az 1-hez, annál inkább igaz a feltevés, hogy az aktuális regressziós modell jól fedi a mintákat.

Valójában az R^2 parméter a korreláció értékének a négyzete. Az eredeti minták és a prediktor értékekből egyszerűen számolható a `cor()` függvénnyel.

```
1 - sum((y - model.lin$fitted.values)^2) / sum((y - mean(y))^2)
```

```
## [1] 0.9378937
```

```
cor(y, model.lin$fitted.values)^2
```

```
## [1] 0.9378937
```

3.5. Predikció

A predikció függő változó értékének becslése a független változó értéke alapján. A predikció során a létrehozott regressziós modell alapján egy tetszőleges független változó x_j értékéhez határozzuk meg, predikáljuk, becsüljük a függő változó $\hat{y}_j$ értékét. Egyszerűbb predikció esetében a `predict()` függvényt használhatjuk.

Az egyszerű függvényhívással meghatározhatjuk az eredeti független értékekhez tartozó predikált értékeket.

```
predict(model.lin)
```

```
##        1        2        3        4        5        6        7        8
## 21.89642 23.81159 25.72676 27.64192 29.55709 31.47226 33.38742 35.30259
##        9       10       11       12       13       14       15       16
## 37.21775 39.13292 41.04809 42.96325 44.87842 46.79359 48.70875 50.62392
##       17       18       19       20       21       22       23       24
## 52.53908 54.45425 56.36942 58.28458 60.19975 62.11492 64.03008 65.94525
##       25       26       27       28       29       30
## 67.86042 69.77558 71.69075 73.60591 75.52108 77.43625
```

A regressziós modell alapján a függő változó értékét prediktálhatjuk a vizsgált adathalmazon kívüli független változó értékeire is.

Természetesen a predikció egy becslés, tehát hozzájuk megbízhatósági intervallumkat is megadhatunk. A `predict()` függvény a megbízhatósági intervallum meghatározására két lehetőséget kínál, melyet az `interval` paraméterrel adhatunk meg.

1. Becslési intervallum (`prediction`): A becslési intervallum egy minta körüli bizonytalanságot írja le. Megadja azt az intervallumot, melybe várhatóan adott x értékhez tartozó y mintaértékek 95%-os valószínűséggel fognak esni. Mivel a realizációkra szeretnénk valamint mondani, általában ezt használjuk.
2. Konfidencia intervallum (`confidence`): A konfidencia intervallum a minták átlagának bizonytalanságát mutatja. Egy x értékhez tartozó y mintaértékek átlagának 95%-os szignifikanciájához tartozó intervallumát adja meg. Természetesen az átlagok szórása kisebb, mint az egyedi értékeké, ezért a konfidencia intervallum jellemzően szűkebb, mint az értékekre vonatkozó becslési intervallum.

A következőkben az 1..30 tartományon létrehozott lineáris regressziós modell alapján predikáljuk a függő változók értékeit az 1..100 intervallumra, valamint állapítsuk meg a predikcióhoz tartozó konfidencia intervallum határait és ábrázoljuk az összefüggéseket.

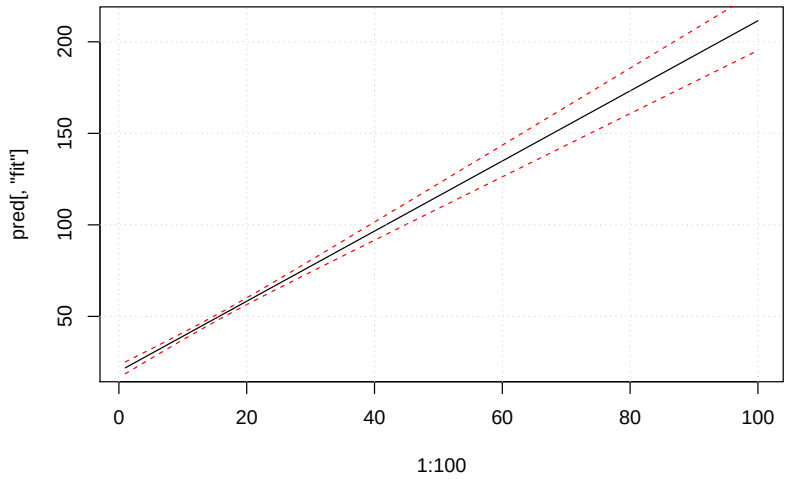
```
pred <- predict(model.lin, newdata = data.frame(x = 1:100), interval = "confidence")
class(pred)
```

```
## [1] "matrix" "array"
```

```
head(pred)
```

```
##        fit      lwr      upr
## 1 21.89642 18.67473 25.11812
## 2 23.81159 20.75215 26.87103
## 3 25.72676 22.82609 28.62742
## 4 27.64192 24.89596 30.38788
## 5 29.55709 26.96102 32.15316
## 6 31.47226 29.02039 33.92412
```

```
plot(1:100, pred[,"fit"], type = "l")
lines(1:100, pred[, "lwr"], lty = "dashed", col = "red")
lines(1:100, pred[, "upr"], lty = "dashed", col = "red")
grid()
```



A predikció eredményeképpen kapott tömb soronként tartalmazza az adott x értékhez tartozó predikált értékeket (`fit`), az intervallum alsó ('lwr') és felső ('upr') határait.

3.6. Feladatok

3.6.1. Egyváltozós lineáris regresszió

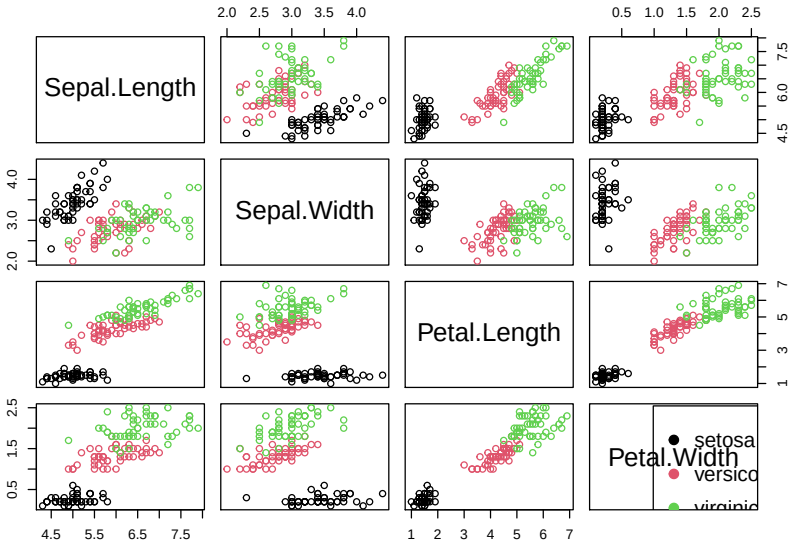
Alkossunk lineáris regressziós modellt az `iris` adathalmazunkon

```
data(iris)
data <- iris
```

Ábrázoljuk az egyes adatok közötti összefüggéseket, mely során színezzük a különféle fajtákhoz tartozó mintákat más és más színűekre.

Nézzük meg a `virginica` fajtára vonatkozóan az egyes értékek közötti korrelációt, és válasszuk ki azokat az attribútumokat, melyek között legnagyobb a korreláció.

```
plot(data[,1:4], col = data$Species)
legend("bottomright", legend = sort(unique(data$Species)),
       pch = 19, col = sort(unique(data$Species)))
```




```
cor(data[data$Species == "virginica", 1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000   0.4572278    0.8642247   0.2811077
## Sepal.Width     0.4572278   1.0000000    0.4010446   0.5377280
## Petal.Length    0.8642247   0.4010446    1.0000000   0.3221082
## Petal.Width     0.2811077   0.5377280    0.3221082   1.0000000
```

A `Sepal.Length` és `Petal.Length` értékek között hozzunk létre lineáris regressziós modellt.

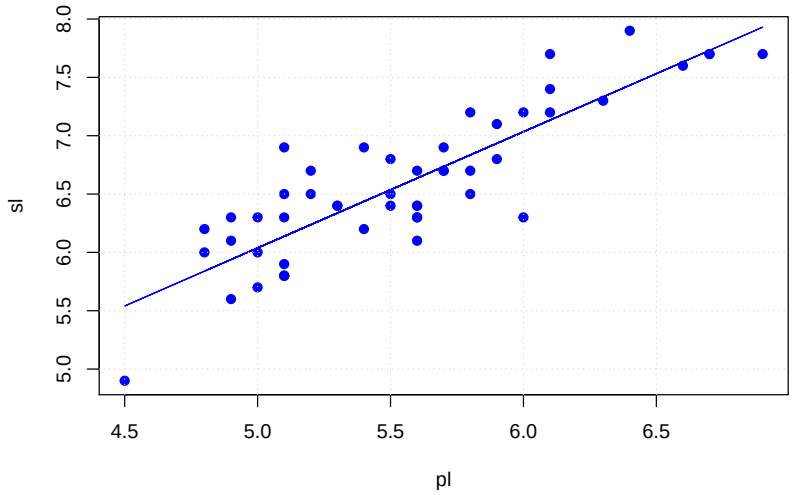
```
sl <- data$Sepal.Length[data$Species == "virginica"]
pl <- data$Petal.Length[data$Species == "virginica"]
model <- lm(sl ~ pl)
summary(model)
```

```
##
## Call:
## lm(formula = sl ~ pl)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73409 -0.23643 -0.03132  0.23771  0.76207
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.05966    0.46677    2.27   0.0277 *
## pl           0.99574    0.08367   11.90  6.3e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3232 on 48 degrees of freedom
## Multiple R-squared:  0.7469, Adjusted R-squared:  0.7416
## F-statistic: 141.6 on 1 and 48 DF,  p-value: 6.298e-16
```

Látható, hogy az értékeket meglehtősen kis szórással (`Std. Error`) sikerült meghatározni.

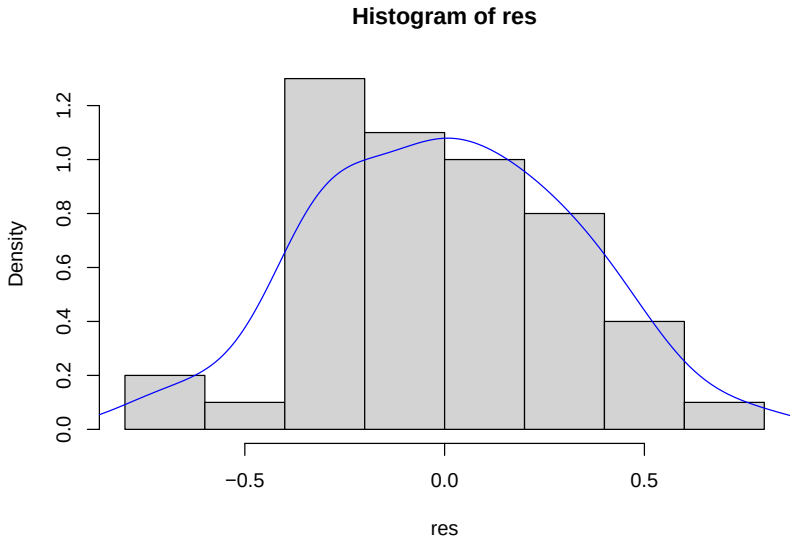
Ábrázoljuk az eredeti és predikált értékeket.

```
plot(sl ~ pl, pch = 19, col = "blue")
lines(pl, predict(model), col = "blue")
grid()
```



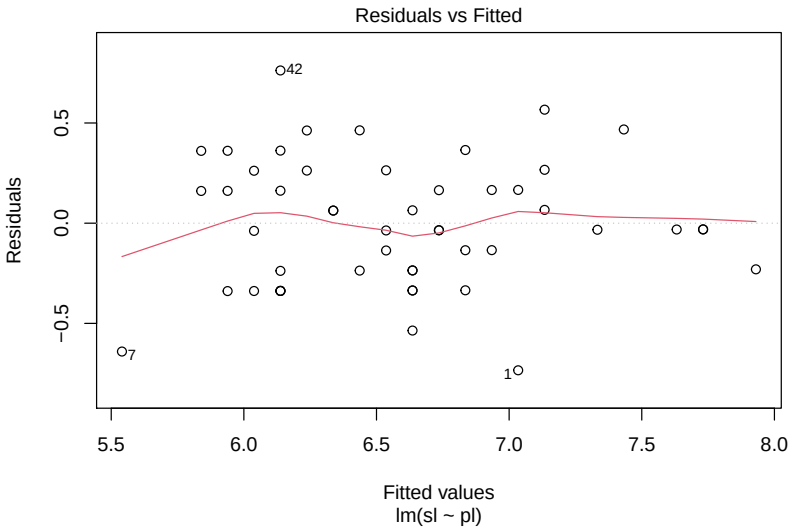
Ábrázoljuk a hibatagok hisztogramját és becsült sűrűségfüggvényét.

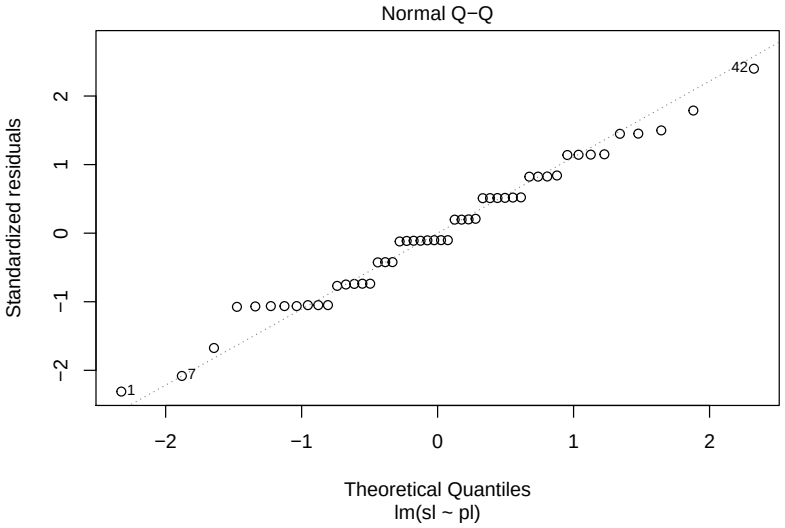
```
res <- residuals(model)
hist(res, probability = T)
lines(density(res), col = "blue")
```

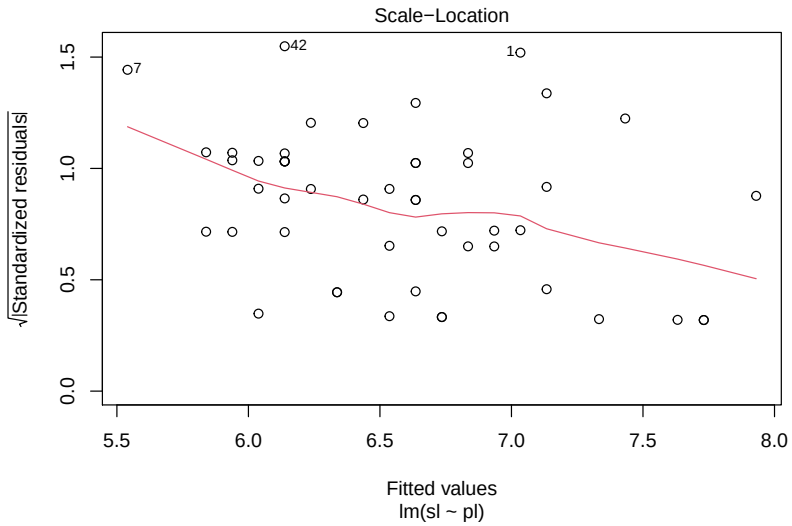


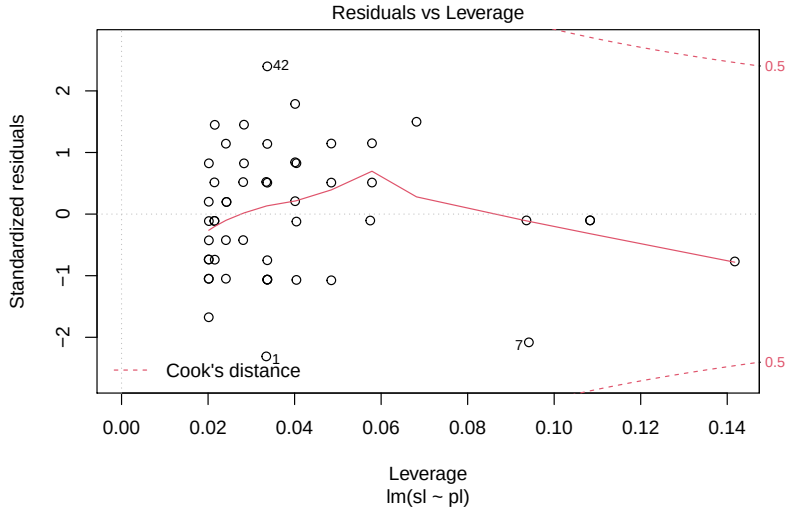
A modellen alkalmazott analítis alapján vonjunk le következtetéseket a lineáris regresszióról.

```
plot(model)
```


Residuals vs Fitted
Residuals
Fitted values
lm(sl ~ pl)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(sl ~ pl)


Scale−Location
√|Standardized residuals|
Fitted values
lm(sl ~ pl)



- (1) A hibatagok eloszlása egyenletes az értelmezési tartományon.
- (2) Az hibatagok eloszlása normálisnak tekinthető.
- (4) Nincs olyan értékünk, mely különösen bagy hatással lenne a modell kialakítására.

Tehát a lineáris regressziós modellünk használható és az adott hibatag eloszlással jól közelíti a tesztminta értékeket. Ezt igazolja az eredeti és a prediktált értékek közötti korreláció is.

```
cor(sl, model$fitted.values)
```

```
## [1] 0.8642247
```

3.6.2. Többváltozós lineáris regresszió

A `marketing` adathalmaz egyes termékek (mint objektumok) `youtube`, `facebook` és `newspaper` hirdetésekre költött összegei és a termékek értékesítési (`sales`) volumenét tartalmazza. Töltsük be az adathalmazt a következő módon.

```
marketing <- read.csv("./data/marketing.csv") # ../lukovszki/marketing.csv
```

Alkossunk lineáris regressziós modellt annak érdekében, hogy meghatározzuk, hogy az egyes médiumok miként függenek össze az értékesítésekkel. A `confint()` segítségével határozzuk meg az egyes koefficiensek konfidencia intervallumát. Ez alapján értékeljük az összefüggést.

```
model <- lm(sales ~ youtube + facebook + newspaper, data = marketing)
summary(model)
```

```
##
## Call:
## lm(formula = sales ~ youtube + facebook + newspaper, data = marketing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.5932  -1.0690   0.2902   1.4272   3.3951
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.526667   0.374290   9.422   <2e-16 ***
## youtube      0.045765   0.001395  32.809   <2e-16 ***
## facebook     0.188530   0.008611  21.893   <2e-16 ***
```

```
## newspaper   -0.001037   0.005871  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.023 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

```
confint(model)
```

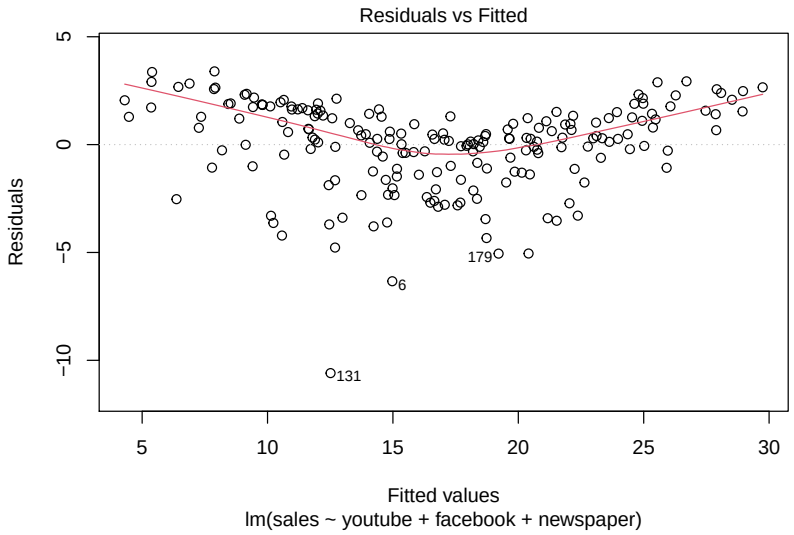
```
##                   2.5 %     97.5 %
## (Intercept)  2.78851474 4.26481975
## youtube      0.04301371 0.04851558
## facebook     0.17154745 0.20551259
## newspaper   -0.01261595 0.01054097
```

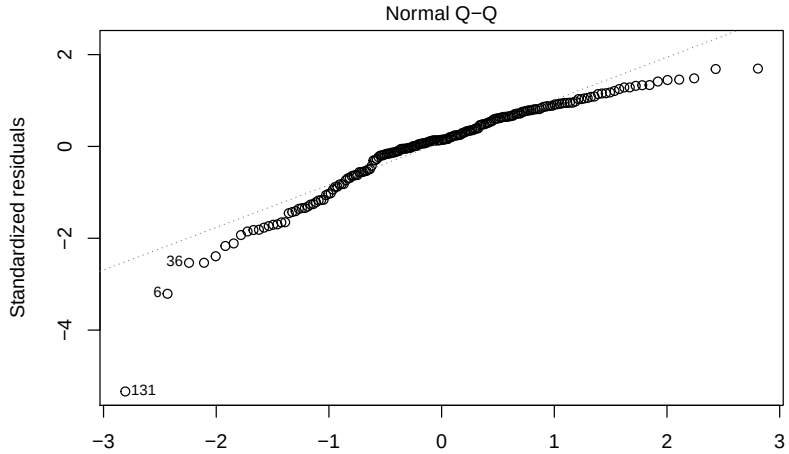
Látható, hogy míg `youtube`-on elköltött hirdetések csak 0.045 együtthatóval, addig a `facebook`-ra költött hirdetések majd egy nagyságrenddel, 0.188 együtthatóval határozzák meg az eladások értékeit. Az újsághirdetések a számok alapján negatív hatással vannak, de olyan kicsi az érték, hogy gyakorlatilag nem gyakorolnak befolyást.

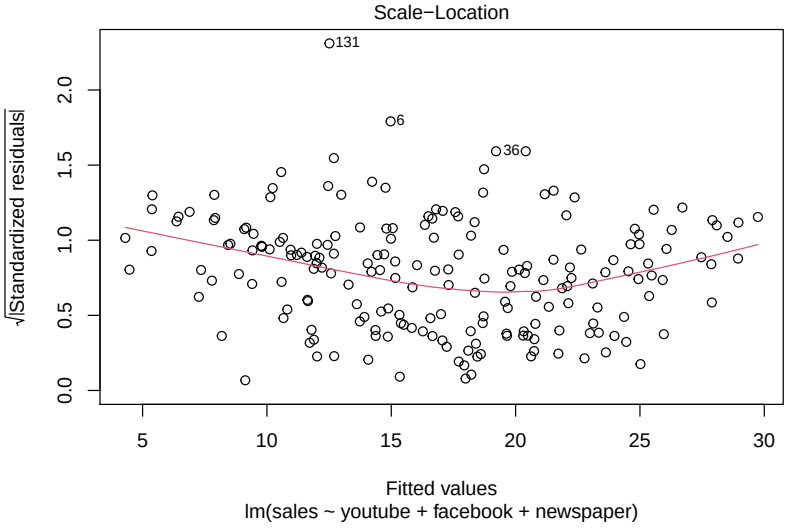
Ez a következtetésünket az együtthatók konfidencia intervallumai is alátámasztják, hiszen a 95% szignifikanciához tartozó értékek még csak nem is lapolódnak egymásba.

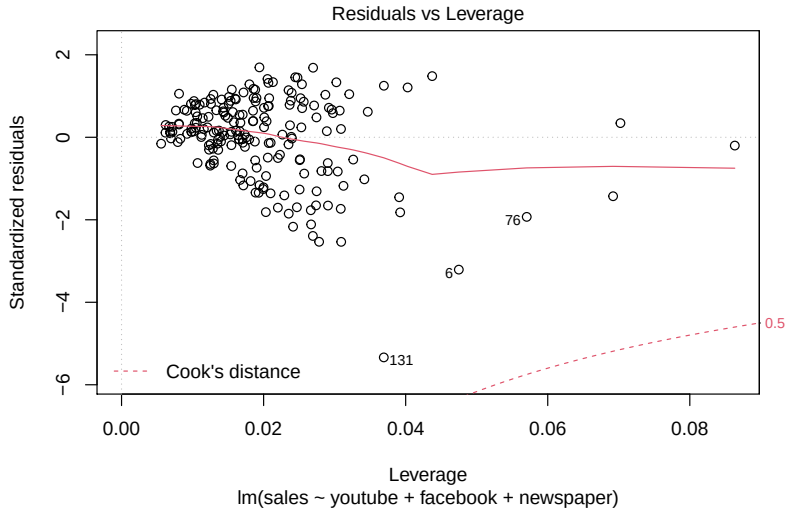
A következőkben értékeljük a modellünket a `plot()` függvény segítségével.

```
plot(model)
```









Vizsgáljuk meg a korreláció értékét a teszthalmaz mintái és a prediktált értékek között.

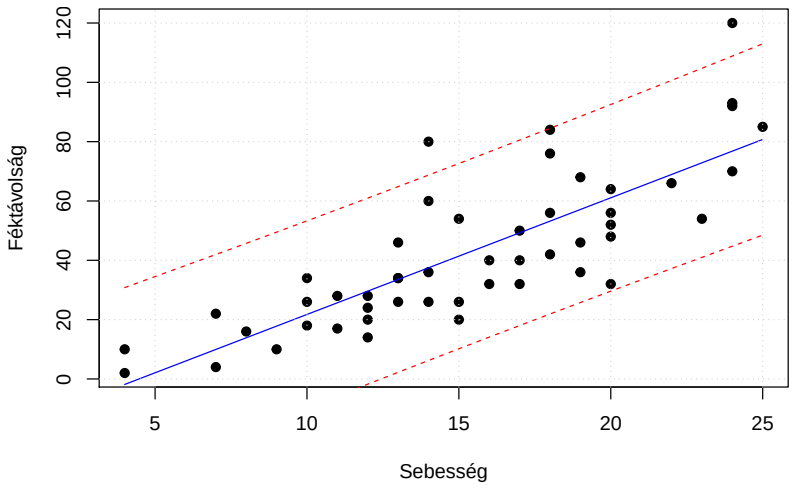
```
cor(marketing$sales, model$fitted.values)
```

```
## [1] 0.947212
```

3.6.3. Predikció lineáris regresszió esetében

A `datasets` csomagban található `car` adathalmaz egyes autók sebességét (`speed`) és féktávolságát (`dist`) tartalmazza. Végezzünk lináris regressziót a két érték között. Végezzük el az értékek predikcióját az adathalmazon és ábrázoljuk a becslési intervallum határokat is. Ábrázoljuk az eredeti értékeket, a lineáris modellt, valamint az becslési intervallum határokat.

```
data("cars", package = "datasets")
model <- lm(dist ~ speed, data = cars)
pred <- predict(model, interval = "prediction")
plot(cars$speed, cars$dist, pch = 19, xlab = "Sebesség", ylab = "Féktávolság")
lines(cars$speed, predict(model), col = "blue")
lines(cars$speed, pred[,"lwr"], col = "red", lty = "dashed")
lines(cars$speed, pred[,"upr"], col = "red", lty = "dashed")
grid()
```



Mi történne, ha az autó a regressziós modell alapján 40mph-val haladna. 95%-os valószínűséggel milyen távolság határok között állna meg?

```
pred2 <- predict(model, newdata = data.frame(speed = 40), interval = "prediction")
pred2
```

```
##        fit      lwr      upr
## 1 139.7173 102.3311 177.1034
```

Az autónak váthazóan 102ft lenne a féktávolsága. Az esetek 95%-ban 139.7 és 177ft lenne a féktávolsága.

---

1. Reprodukálja a három minta feladatot önállóan.
2. Végezze el harmadik feladat (autók sebessége és fékutjai közözzi lineáris regesszió) hibatag vizsgálatát!

---

4. Nemlineáris regresszió

Változók között létezhet nemlineáris kapcsolat is. Nemlineáris esetben az `x` független változó és `y` függő változó között tetszőleges nemlineáris kapcsolat is fennálhat, de jellemzően a következő nemlineáris kapcsolatot alkalmazhatjuk.

1. *Exponenciális regresszió*: Változók közötti exponenciális kapcsolat esetében ay y függővál-tozó és az x tényezővváltozó között exponenciális kapcsolat áll fenn, azaz a regressziót a következő formában keressük:

$$y_i = ab^{x_i} + \epsilon_i$$

2. *Polinomiális regresszió*: Polinomiális regressziós modellt abban az esetben alkalmazzuk, ha az összefüggés nem monoton, azaz várhatóan minimuma vagy maximuma van a változók közötti összefüggésnek. Ebben az esetben az összefüggés a következőképpen írható le.

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + ... + \beta_p x_i^p + \epsilon_i$$

3. *Logisztikus regresszió*: Logisztikus regressziós modellt olyan a gyakorlatban is előforduló folyamatok esetében alkalmazzuk, melyek felbonthatók a következő szakaszokra:

    1. Lassú növekedés
    2. Gyors növekedés
    3. Lassuló növekedés
    4. Telítés

$$y_i = A\frac{e^{\beta_0+\beta_1 x_i}}{1 + e^{\beta_0+\beta_1 x_i}}$$

Nemlineáris regresszió meghatározása esetében két megoldást alkalmazhatunk:

1. A regressziót visszavezetjük lineáris regresszióre
2. Nemlineáris függvény szerinti regressziót alkalmazunk

4.1. Visszavezetés lineáris regresszióra

A következőkben nézzük meg, hogy miként vezethető vissza az exponenciális regresszió lineáris regresszióra.

Az exponenciális regresszió formája a következő:

$$\hat{y} = ab^x$$

Ebben az esetben az y változó logaritmusa és az x változó között lineáris kapcsolat áll fenn:

$$\log y = \log a + x \log b$$

A fenti egyeblet $y' = a' + b'x'$ formában is írhatjuk, melyben bevezetésre kerülő új jelölések a változókra

$$y' = \log y, \quad x' = x$$

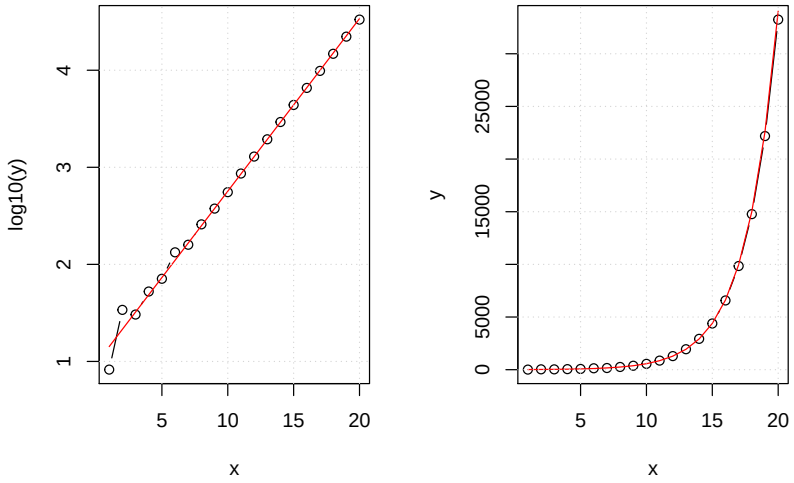
, valamint a paraméterekre a következő.

$$a' = \log a, \quad b' = \log b$$

A fentiek alapján az exponenciális regresszió megvalósítható lineáris regresszió formájában az `lm()` függvény használatával a következő lépések használatával.

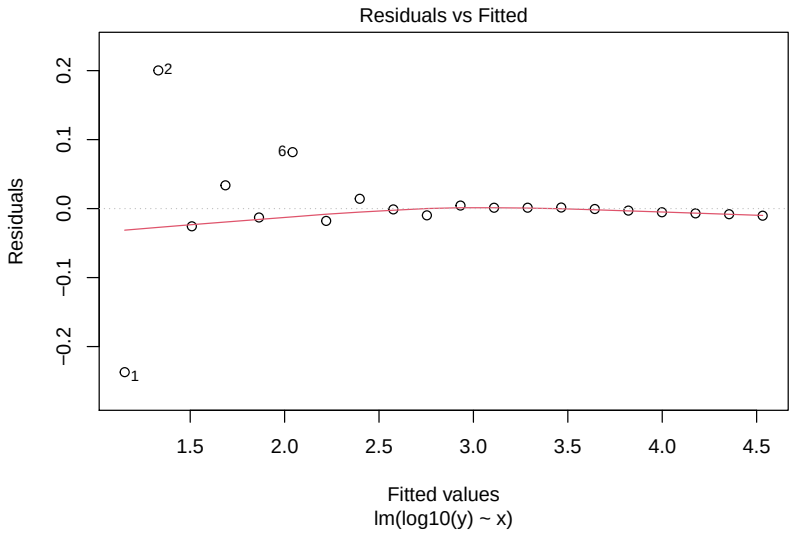
1. Az x és y értékeket traszformáljuk a fentiek szerint x' és y'-re.
2. Futtatjuk a lináris regressziót az `lm()` függvény segítségével, melynek eredménye az a' és b' értékek.
3. Az a' és b' értékekből meghatározzuk a és b értékeit.

```
n <- 20
x <- 1:n
y <- 1.5^x*10.0 + rnorm(n,0,10)
# model meghatározása
exp.model <- lm(log10(y) ~ x)
par(mfrow = c(1,2))
# lineáris regresszió
plot(x, log10(y), type = "b")
lines(x, exp.model$fitted.values, col = "red")
grid()
# visszatranszformált
plot(x, y, type = "b")
lines(x, (10^coef(exp.model)[1])*(10^coef(exp.model)[2])^x, col="red")
grid()
```



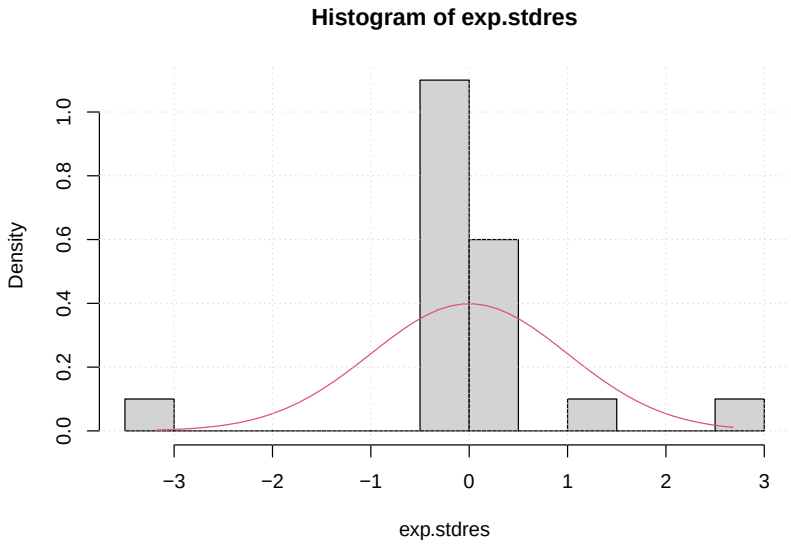
A regresszió hibájának vizsgálata:

```
#plot(exp.fit$residuals, col="red")
plot(exp.model, which=1)
```

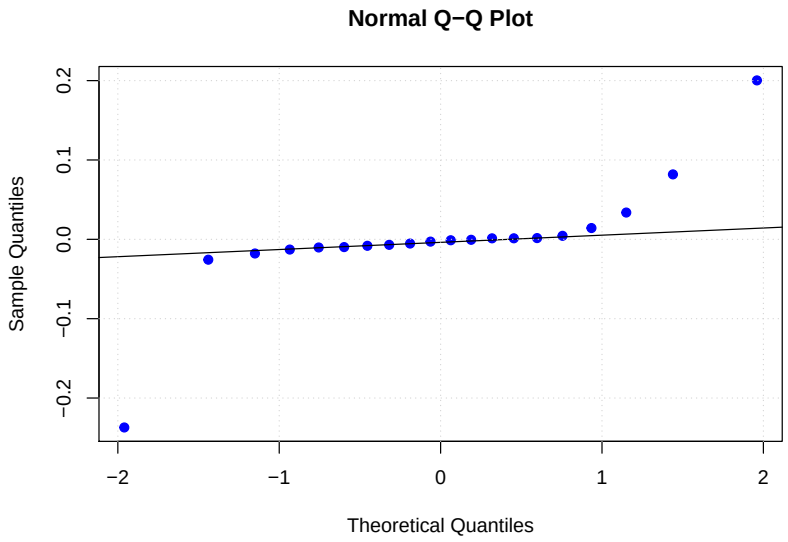



```
exp.res.m <- mean(exp.model$residuals)
exp.res.s <- sd(exp.model$residuals)
# --- Az eloszlás normalitás vizsgálata
```

```
exp.stdres = (exp.model$residuals - exp.res.m) / exp.res.s
# A hiba hisztogramja
hist(exp.stdres, freq=F, breaks=10) # nem a számosságát, hanem az előfordulási arányt ábráz
# Standard (mean=0, std=1) normális eloszlás készítése
xfit<-seq(min(exp.stdres), max(exp.stdres), length=50)
yfit<-dnorm(xfit, 0, 1)
lines(xfit, yfit, col=2)
grid()
```




```
# QQ plot
qqnorm(exp.model$residuals, pch = 19, col = "blue")
qqline(exp.model$residuals)
grid()
```



A hiba vizsgálatnál láthatjuk, hogy a logaritmus torzítja az eredeti értékekre rakódó normális eloszlású hibát. A logaritmus függvény tulajdonságai miatt a kis értékekre rakódó hibát nagyobb arányban transzformálja, és ez el is ronthatja jelentősen a regressziót, hiszen a kis értékek nagyobb súllyal vesznek részt a regresszió meghatározásában. Vegyük észre, hogy a regresszió hibája a lináris térben már nem normális eloszlású (lásd Q-Q plot). A fenti számolás a lineáris regressziót jól, viszont az exponenciális (nem lineáris) regressziót már rosszul jellemzi.

1. Adott az alább generált mintasorozat, mely esetében a regressziót az $y = ax^b$ alakban keressük. Határozza a regressziót lineáris regresszióra való visszavezetéssel az `lm()` függvény segítségével. Ábrázolja az eredeti és a prediktált értékeket is egy ábrán.

```
n <- 20
x <- 1:n
a <- 20
b <- 2
y <- a*x^b + rnorm(n, 0, 10)
```

2. Vegye vizsgálat alá az előző feladat hibatagjait.

4.2. Nemlináris függvény alapján

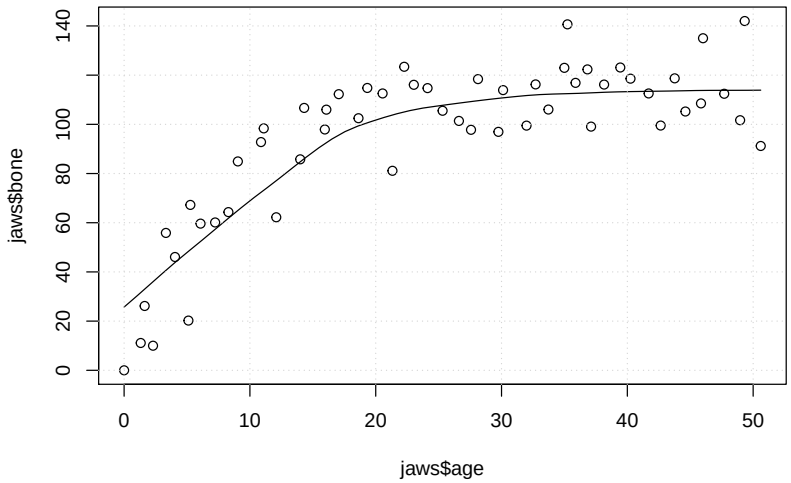
Általánosságbana nemlineáris regressziót az `nls()` függvény segítségével határozhatjuk meg.

A következőkben olvassuk be a `jaws.csv` adatkeretet, mely szarvasok életkorát (`age`) és álkapcsának hosszát (`bone`) tartalmazza. Ezen az adathalmazon nézzük meg, hogy az életkor milyen módon határozza meg az álkapcsok hosszát.

```
rm(list = ls())
jaws <- read.csv("./data/jaws.csv")
```

A `scatter.smooth()` függvény segítségével ábrázoljuk az adatokat, valamint a közelítő folytonos vonalat, melyet elmosással határoz meg a függvény.

```
scatter.smooth(x = jaws$age, y = jaws$bone)
grid()
```



Ezek alapján keressük a regressziót $y = a - be^{-cx}$ alakban.

```
model <- nls(bone ~ a - b * exp(-c * age),
             start = list(a = 110, b = 110, c = 0.1), data = jaws)
summary(model)
```

```
##
## Formula: bone ~ a - b * exp(-c * age)
##
```

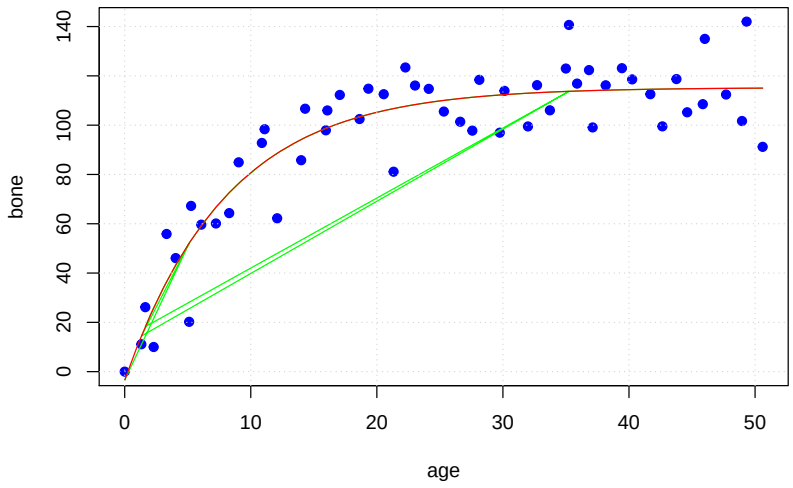
```
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a 115.2527     2.9139   39.55  < 2e-16 ***
## b 118.6877     7.8925   15.04  < 2e-16 ***
## c   0.1235     0.0171    7.22 2.44e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.21 on 51 degrees of freedom
##
## Number of iterations to convergence: 3
## Achieved convergence tolerance: 4.118e-06
```

Az `nls()` függvény használatánál a kezdeti értékeket körültekintően kell meghatározni. Az `a` érték megadásánál gyakorlatilag a maximumot adjuk meg, `b` érték esetében ugyancsak, mivel egyre növekvő értékeknél, ahol az exponenciális tag nullához tart a függvény a maximumhoz. A `c` érték meghatározásánál a kezdeti meredekséget határozzuk meg, melyet a `b` értékével osztunk. Ennek értelmében $c = \Delta y/\Delta x/b = 100/10/100$.

Amennyiben a kezdeti értékeink kívül esnek a praktikus tartományon, az eredményünk könnyen divergál.

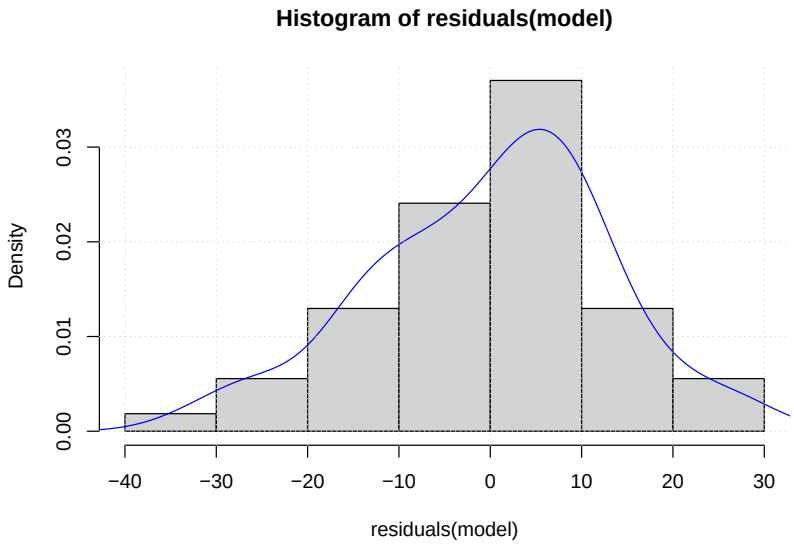
Ábrázoljuk az eredeti és a predikált értékeket.

```
plot(jaws, col = "blue", pch = 19)
# az adatsorban a kor nincs növekvő sorrendben, ezért
# segítségével nem lehet egyenest rajzolni
lines(jaws$age, predict(model), col = "green")
# ehelyett új független értékre határozzuk meg a predikált értékeket
age <- seq(min(jaws$age), max(jaws$age), length.out = 50)
lines(age, predict(model, newdata = data.frame(age = age)), col = "red")
grid()
```



A következőkben vizsgáljuk meg a residual-ok eloszlását.

```
hist(residuals(model), probability = T)
lines(density(residuals(model)), col = "blue")
grid()
```




```
# biztosabbak lehetünk egy teszt használatával
shapiro.test(residuals(model))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(model)
## W = 0.98068, p-value = 0.5301
```

A vizsgálatok alapján ugyancsak elmondható, hogy a hibatag értékei normális eloszlást követnek.

4.3. Feladatok

4.3.1. Egyváltozós nemlineáris regresszió

A következőkben végezzünk el a `population` adathalmazon nemlineáris regressziót az országok emberekre vonatkozó GDP értéke (`gdpPercap`) és a várható élettartam (`lifeExp`) között az egyes országokban.

```
cdata <- read.csv("data/population.csv")
summary(cdata)
```

```
##    country               year           pop              continent
##  Length:1704        Min.   :1952   Min.   :6.001e+04   Length:1704
##  Class :character   1st Qu.:1966   1st Qu.:2.794e+06   Class :character
##  Mode  :character   Median :1980   Median :7.024e+06   Mode  :character
##                     Mean   :1980   Mean   :2.960e+07
##                     3rd Qu.:1993   3rd Qu.:1.959e+07
##                     Max.   :2007   Max.   :1.319e+09
##     lifeExp        gdpPercap
##  Min.   :23.60   Min.   :   241.2
##  1st Qu.:48.20   1st Qu.:  1202.1
##  Median :60.71   Median :  3531.8
##  Mean   :59.47   Mean   :  7215.3
##  3rd Qu.:70.85   3rd Qu.:  9325.5
##  Max.   :82.60   Max.   :113523.1
```

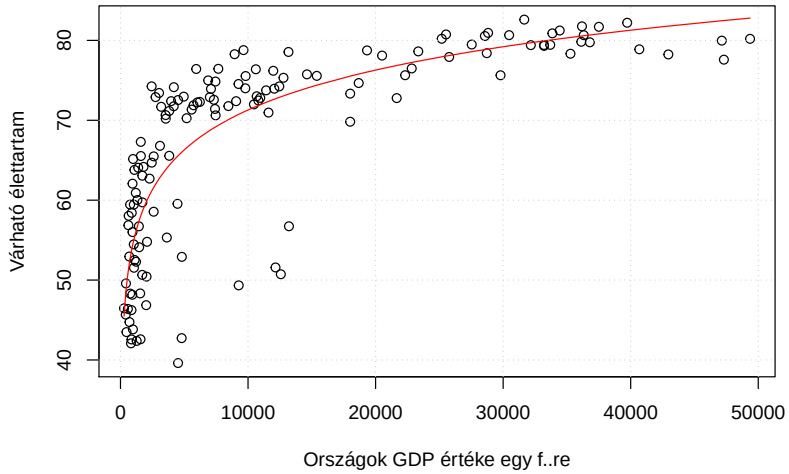
```
cdata7 <- cdata[cdata$year == 2007, c("gdpPercap","lifeExp")]
cdata7 <- cdata7[order(cdata7$gdpPercap),]
```

```
plot(cdata7$gdpPercap, cdata7$lifeExp,
     xlab = "Országok GDP értéke egy főre",
     ylab = "Várható élettartam")
```

```
## Warning in title(...): conversion failure on 'Országok GDP értéke egy főre' in
## 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in title(...): conversion failure on 'Országok GDP értéke egy főre' in
## 'mbcsToSbcs': dot substituted for <91>
```

```
# nemlineáris regresszió a + b * log_10(x) formában
cdata.model <- nls(lifeExp ~ a + b * log10(gdpPercap),
                   start = list(a = 0, b = 1), data = cdata7)
lines(cdata7$gdpPercap, predict(cdata.model), col = "red")
grid()
```



---

1. Végezze el a hibatagok analízisét!
2. Vizsgálja meg, hogy mely értékek kerültek kívül a 95%-os becslési határon!
    a. Mely országokhoz tartoznak ezek?
    b. Írja ki az országok nevét az ábrára!

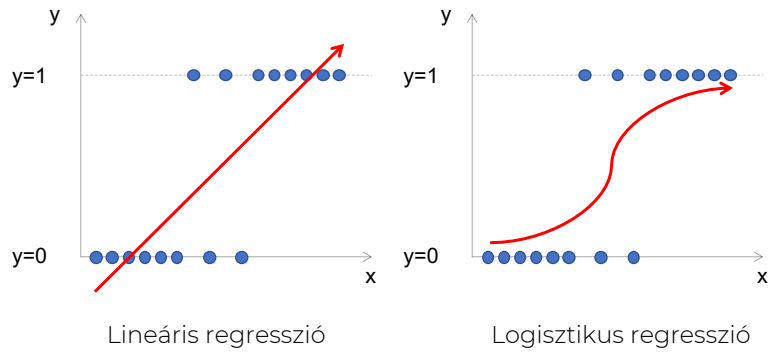
---

5. Logisztikus regresszió

A logisztikus regressziót használjuk binomiális (két értékű) függő változók becslésénél, legyen szó akár egy, akár több független változóról! Általánosságban elmondható, hogy a logisztikus regresszió segítségével határozhatjuk meg kategorikus változók és független változók közötti összefüggéseket.

Míg a például a lineáris regressziót használhatjuk kvantitatív értékek regressziójára, a kvalitatív értékek meghatározására haszontalan. Kvalitatív értékek meghatározása esetében valószínűséget rendelhetünk az egyes értékek bekövetkezéséhez, ami lehet egy kategória, vagy osztályba való tartozás.

A valószínűség meghatározása a logisztikus függvény alapján történhet, melyet a következőképpen



3. ábra. Lineáris és logisztikus regresszió

fejezhetjük ki.

$$p(x) = P(y = 1|x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

Ebből kifejezhető az esélyességi ráta (odds ratio), mely értékkészlete már $[0, \infty)$ értéktartományt vehet vel.

$$\frac{p(x)}{1 - p(x)} = e^{\beta_0+\beta_1 x}$$

A fenti összefüggés logaritmusát képezve kaphatjuk a logisztikus regressziónál használt `logit` függvényt.

$$\log \frac{p(x)}{1 - p(x)} = \beta_0 + \beta_1 x$$

5.1. Binomiális prediktorok

Az `ucla` adatbázis hallgatók különféle értékelését tartalmazza (`gre`, `gpa`, `rank`), valamint, hogy felvételt nyert-e az egyetemre (`admit`). A következőkben meg szeretnénk határozni, hogy az egyes értékelések miként határozzák meg, hogy a hallgató felvételt nyer-e.

A logisztikus regressziót a `glm()` függvény segítségével határozhatjuk meg.

```
ucla <- read.csv("./data/ucla.csv")
ucla$rank <- as.factor(ucla$rank)
str(ucla)
```

```
## 'data.frame':    400 obs. of  4 variables:
##  $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
##  $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
##  $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
##  $ rank : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
```

```
logit <- glm(admit ~ gre + gpa + rank, data = ucla, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = "binomial",
##     data = ucla)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

Az eredményekből a következőket határozhatjuk meg:

1. A `p-value` értékei alapján mindegyik összefüggés szignifikáns.
2. A `gre`, `gpa` értékeinek egy egységnyi növekedése a rendre 0.002 és 0.804 értékkel növeli a `logit` értékét.
3. A `rank` kategorikus változó (`factor`) értéke esetében a 1-2, a 2-3, a 3-4 rang visszaesés rendre -0.67, -1.34 és -1.55 értékekkel csökkenti a `logit` értékét, azaz a bekerülési esélyt (nem valószínűséget!).

Nézzük meg, hogy 750 gre, 3.8 gpa és 1 rank értékei esetében egy hallgató felvételt nyerne-e az UCLA-ra az adatok alapján.

```
predict(logit, data.frame(gre=790,gpa=3.8,rank=as.factor(1)))
```

```
##       1
## 0.85426
```

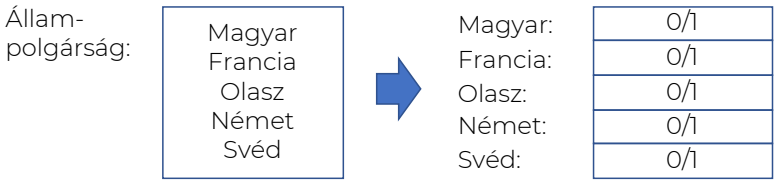
Tehát a hallgatónak 85% esélye van, hogy bekerül az UCLA-ra.

5.2. Diszkrét, kategorikus prediktorok

A logisztikus regresszió felhasználható kategorikus változók predikciójára is. Ebben az esetben a kategorikus változót visszavezetjük bináris változó értékekre, binomiális változókra. Erre két lehetőség van a gyakorlatban.

1. *One-hot coding*: Ennek a kódolásnak az esetében a kategorikus változó minden egyes állapotát egy-egy külön bináris változóba. Az egyes bináris változó jelzi az adott kategóriában való tartózkodást.

2. *Dummy coding*: Dummy kódolás esetében a kategorikus változó n értékéhez $n-1$ bináris változót rendelünk, az n-dik referencia kategóriához való rendelést az $n-1$ bináris változó 0 értéke jelzi.



4. ábra. One-hot kódolás