

# Linearis regresszio

Kekecs Zoltan

March 31 2020

## A linearis regresszio alapjai

Ennek az orának a célja hogy megismerkedjunk a linearis regresszióval, annak logikájával, es az értelmezesehez szükséges alapfogalmakkal.

A dokumentum legfrissebb változata itt található: <https://osf.io/5a4kj/>

## Package-ek betöltése

Betöltjük a következő package-eket:

```
library(psych) # for describe
library(gsheet) # to read data from google sheets
```

```
## Warning: package 'gsheet' was built under R version 3.6.3
```

```
library(tidyverse) # for tidy code
```

## Saját funckiok betöltése

Az alábbi funkció csak az orai vizualizációhoz kell, nem feltétlenül kell megérteni a tartalmát. Ezt a funkciót arra használjuk majd hogy a linearis regresszióban fennmaradó (rezidualis) hibát vizualizáljuk.

```
error_plotter <- function(mod, col = "black", x_var = NULL){
  mod_vars = as.character(mod$call[2])
  data = as.data.frame(eval(parse(text = as.character(mod$call[3]))))
  y = substr(mod_vars, 1, as.numeric(gregexpr(pattern = '~', mod_vars))-2)
  x = substr(mod_vars, as.numeric(gregexpr(pattern = '~', mod_vars))+2, nchar(mod_vars))

  data$pred = predict(mod)

  if(x == "1" & is.null(x_var)){x = "response_ID"}
  data$response_ID = 1:nrow(data)} else if(x == "1"){x = x_var}

  plot(data[,y] ~ data[,x], ylab = y, xlab = x)
  abline(mod)

  for(i in 1:nrow(data)){
    clip(min(data[,x]), max(data[,x]), min(data[i,c(y,"pred")]), max(data[i,c(y,"pred")]))
    abline(v = data[i,x], lty = 2, col = col)
  }
}
```

## Adatmenedzsment és adat bemutatása

### Adatok betöltése

Mondjuk, hogy egy turisták koreben gyakran látogatott cipoboltban dolgozunk, és mivel a világon sokfajta cipomeretet használnak és az emberek gyakran nem tudják a saját európai cipomeretüket, szeretnénk a magasságuk alapján megbecsülni, mekkora az európai cipomeretük.

Az alábbi kóddal betölthetjük az adattáblát, amiben a korábbi orákon felvett kérdőívekből szerepelnek a magasság és cipomeret adatok.

```
mydata = as_tibble(gsheets2tbl("https://docs.google.com/spreadsheets/d/1GXx2YoktyIdXLqKdm4f_MMWHUzXYgxWRl"))
```

### Adatok ellenőrzése

Szokás szerint az adatok ellenőrzésével kezdünk, pl. View(), describe(), és summary() funkciókkal.

```
# descriptive statistics
describe(mydata)
```

```
## Warning in describe(mydata): NAs introduced by coercion

##           vars  n   mean    sd median trimmed  mad   min max  range
## jelige*      1 25  32.00    NA    32   32.00 0.00 32.00  32    0.00
## magassag     2 25 161.07 34.32   168  166.62 8.90   1.82 190 188.18
## cipomeret    3 25  38.94  3.34    38   38.60 2.97 35.00  47   12.00
## alvas_tegnap_1 4 25   7.92  1.26     8    7.90 1.48   6.00  10    4.00
## alvas_tegnap_3 5 23   7.61  1.03     8    7.68 1.48   5.00   9    4.00
## alvas_altalaban_1 6 25   7.04  1.02     7    7.10 1.48   5.00   9    4.00
## alvas_altalaban_3 7 23   7.13  0.63     7    7.16 0.00   6.00   8    2.00
## energiaszint_1  8 25   5.48  2.16     5    5.62 2.97   1.00   8    7.00
## energiaszint_3  9 23   6.17  1.92     7    6.26 1.48   3.00   9    6.00
##           skew kurtosis    se
## jelige*      NA      NA    NA
## magassag    -3.95    15.59 6.86
## cipomeret     0.99    -0.05 0.67
## alvas_tegnap_1 -0.22    -1.12 0.25
## alvas_tegnap_3 -0.63    -0.11 0.22
## alvas_altalaban_1 -0.30    -0.71 0.20
## alvas_altalaban_3 -0.07    -0.63 0.13
## energiaszint_1 -0.27    -1.21 0.43
## energiaszint_3 -0.38    -1.35 0.40
```

```
mydata %>%
  summary()
```

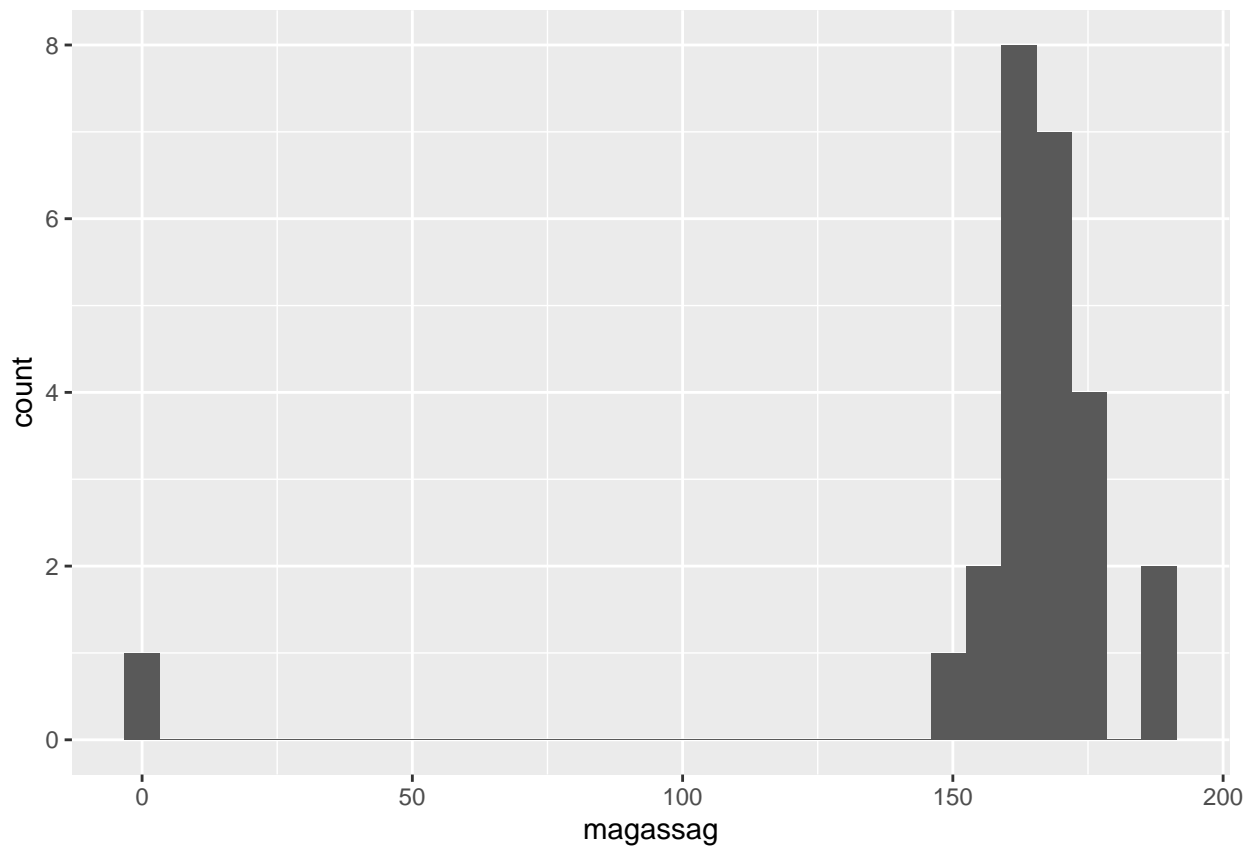
```
##      jelige      magassag      cipomeret      alvas_tegnap_1
## Length:25      Min.   : 1.82      Min.   :35.00      Min.   : 6.00
## Class :character 1st Qu.:162.00    1st Qu.:37.00    1st Qu.: 7.00
## Mode  :character Median :168.00    Median :38.00    Median : 8.00
##           Mean   :161.07    Mean   :38.94    Mean   : 7.92
##           3rd Qu.:170.00    3rd Qu.:40.00    3rd Qu.: 9.00
##           Max.   :190.00    Max.   :47.00    Max.   :10.00
##
## alvas_tegnap_3 alvas_altalaban_1 alvas_altalaban_3 energiaszint_1
## Min.   :5.000      Min.   :5.00      Min.   :6.00      Min.   :1.00
## 1st Qu.:7.000      1st Qu.:6.00      1st Qu.:7.00      1st Qu.:4.00
## Median :8.000      Median :7.00      Median :7.00      Median :5.00
```

```
## Mean :7.609 Mean :7.04 Mean :7.13 Mean :5.48
## 3rd Qu.:8.000 3rd Qu.:8.00 3rd Qu.:7.50 3rd Qu.:8.00
## Max. :9.000 Max. :9.00 Max. :8.00 Max. :8.00
## NA's :2 NA's :2
## energiaszint_3
## Min. :3.000
## 1st Qu.:4.500
## Median :7.000
## Mean :6.174
## 3rd Qu.:8.000
## Max. :9.000
## NA's :2
```

```
# histograms
```

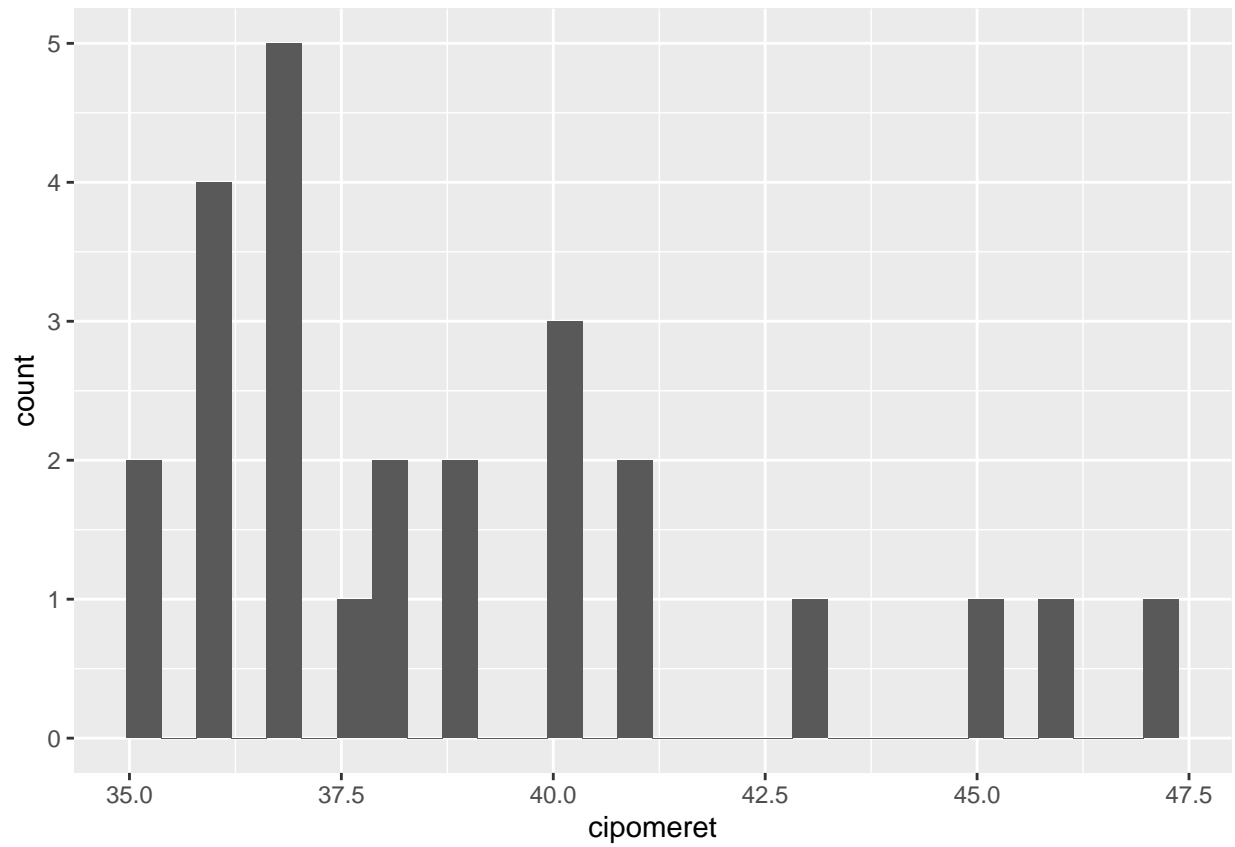
```
mydata %>%
  ggplot() +
  aes(x = magassag) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

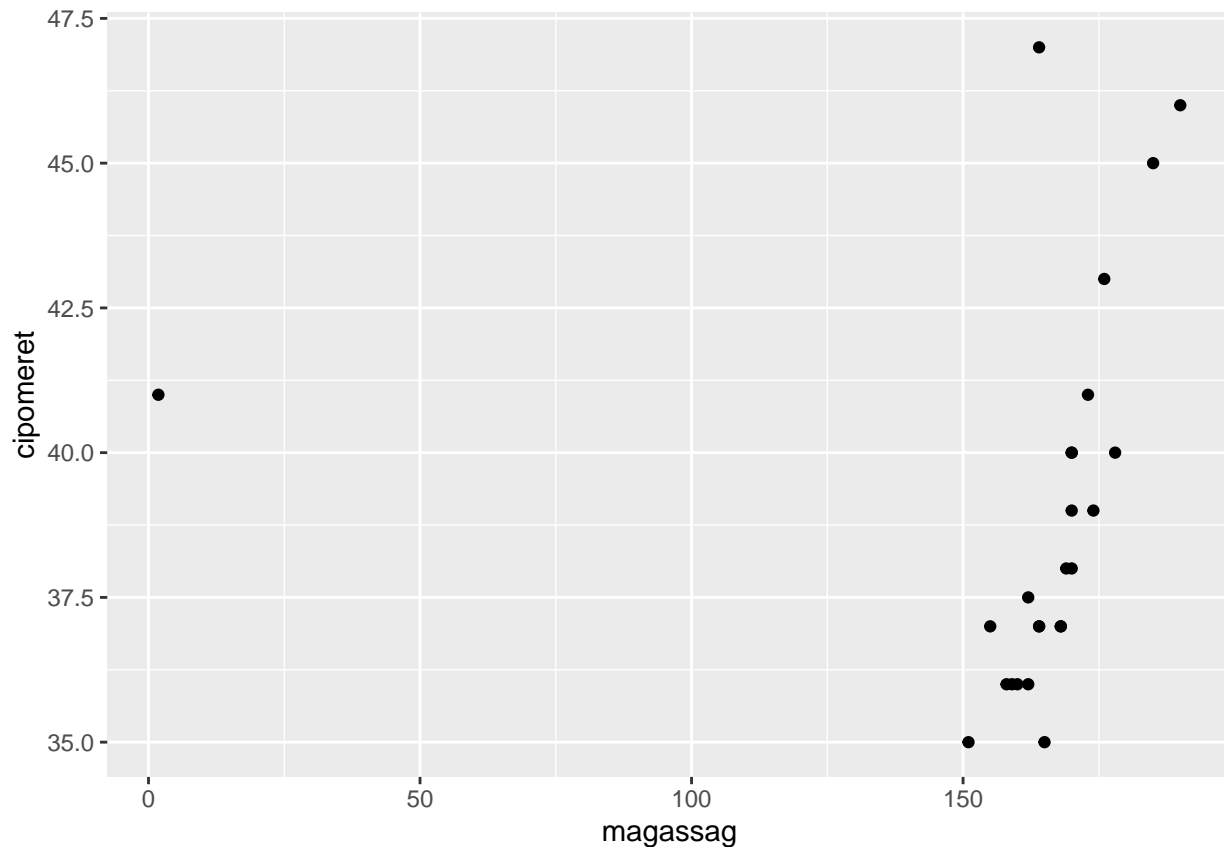


```
mydata %>%
  ggplot() +
  aes(x = cipomeret) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# scatterplot
mydata %>%
  ggplot() +
  aes(x = magassag, y = cipomeret) +
  geom_point()
```



### Adattisztítás és ellenőrzés

Latható hogy van egy ember aki centimeter helyett meterben adta meg a magassagát. Ezt javítjuk és ellenőrizzük hogy minden jól néz-e ki ezek után.

```
mydata_corrected = mydata %>%
  mutate(magassag = replace(magassag, magassag == 1.82, 182))

# descriptive statistics
describe(mydata_corrected)
```

```
## Warning in describe(mydata_corrected): NAs introduced by coercion
```

```
##           vars  n   mean    sd median trimmed  mad min max range skew
## jelige*      1 25  32.00   NA     32   32.00  0.00  32  32     0   NA
## magassag     2 25 168.28  9.22    168  167.90  8.90 151 190    39  0.43
## cipomeret    3 25  38.94  3.34     38   38.60  2.97  35  47    12  0.99
## alvas_tegnap_1 4 25   7.92  1.26      8    7.90  1.48   6  10     4 -0.22
## alvas_tegnap_3 5 23   7.61  1.03      8    7.68  1.48   5   9     4 -0.63
## alvas_altalaban_1 6 25   7.04  1.02      7    7.10  1.48   5   9     4 -0.30
## alvas_altalaban_3 7 23   7.13  0.63      7    7.16  0.00   6   8     2 -0.07
## energiaszint_1  8 25   5.48  2.16      5    5.62  2.97   1   8     7 -0.27
## energiaszint_3  9 23   6.17  1.92      7    6.26  1.48   3   9     6 -0.38
##           kurtosis    se
## jelige*           NA    NA
## magassag        -0.25 1.84
## cipomeret        -0.05 0.67
```

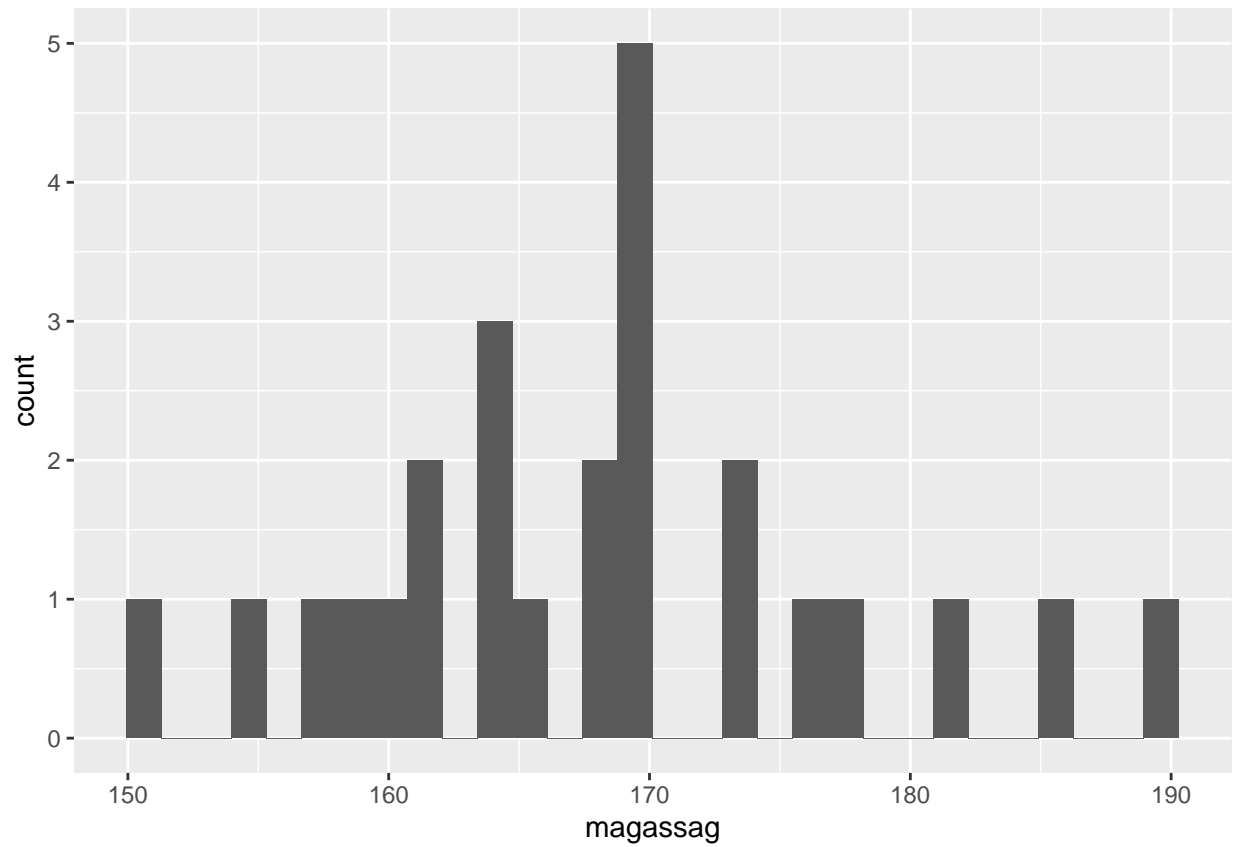
```
## alvas_tegnap_1      -1.12 0.25
## alvas_tegnap_3      -0.11 0.22
## alvas_altalaban_1   -0.71 0.20
## alvas_altalaban_3   -0.63 0.13
## energiaszint_1      -1.21 0.43
## energiaszint_3      -1.35 0.40
```

```
mydata_corrected %>%
  summary()
```

```
##      jelige      magassag      cipomeret      alvas_tegnap_1
## Length:25      Min.   :151.0    Min.   :35.00    Min.   : 6.00
## Class :character 1st Qu.:162.0    1st Qu.:37.00    1st Qu.: 7.00
## Mode  :character Median :168.0    Median :38.00    Median : 8.00
##                      Mean   :168.3    Mean   :38.94    Mean   : 7.92
##                      3rd Qu.:173.0    3rd Qu.:40.00    3rd Qu.: 9.00
##                      Max.    :190.0    Max.    :47.00    Max.    :10.00
##
## alvas_tegnap_3 alvas_altalaban_1 alvas_altalaban_3 energiaszint_1
## Min.   :5.000    Min.   :5.00    Min.   :6.00    Min.   :1.00
## 1st Qu.:7.000    1st Qu.:6.00    1st Qu.:7.00    1st Qu.:4.00
## Median :8.000    Median :7.00    Median :7.00    Median :5.00
## Mean   :7.609    Mean   :7.04    Mean   :7.13    Mean   :5.48
## 3rd Qu.:8.000    3rd Qu.:8.00    3rd Qu.:7.50    3rd Qu.:8.00
## Max.   :9.000    Max.   :9.00    Max.   :8.00    Max.   :8.00
## NA's    :2              NA's    :2
## energiaszint_3
## Min.   :3.000
## 1st Qu.:4.500
## Median :7.000
## Mean   :6.174
## 3rd Qu.:8.000
## Max.   :9.000
## NA's    :2
```

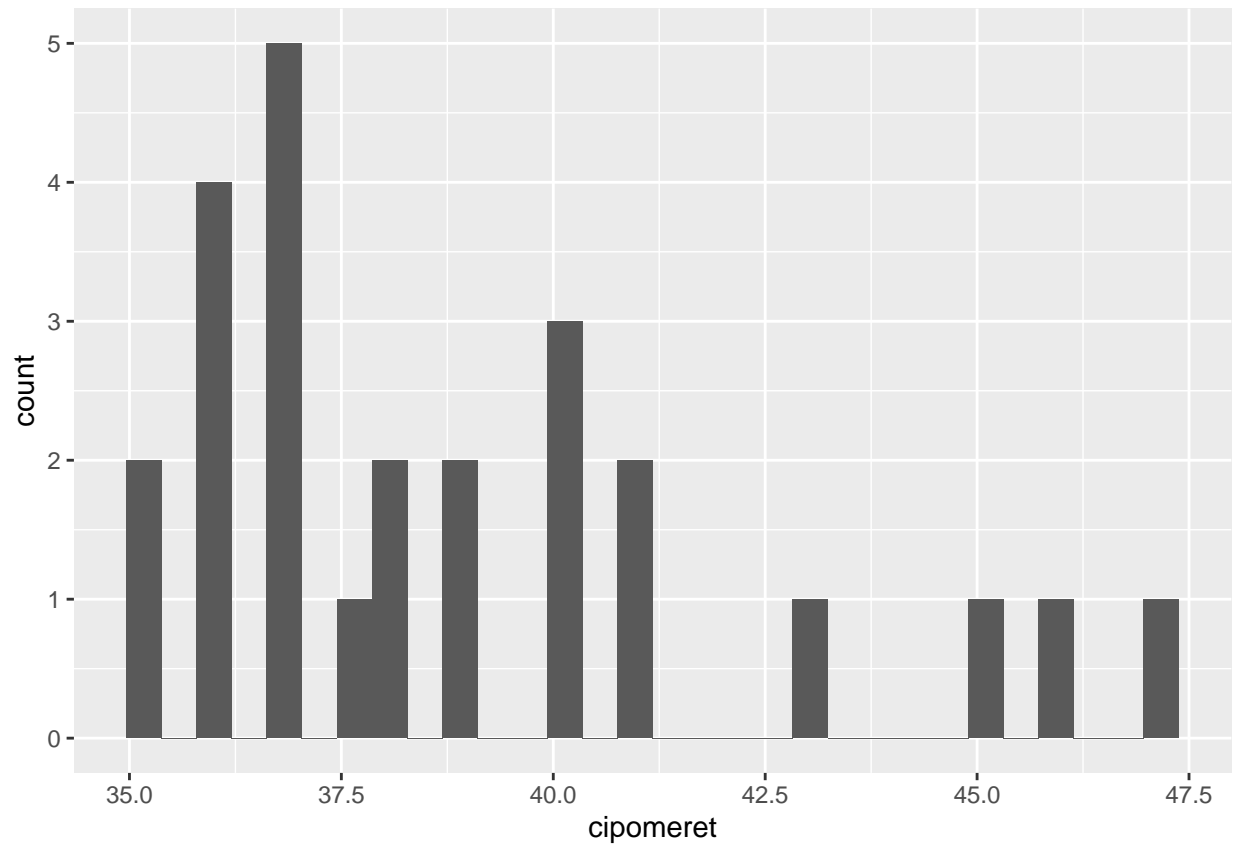
```
# histograms
mydata_corrected %>%
  ggplot() +
  aes(x = magassag) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



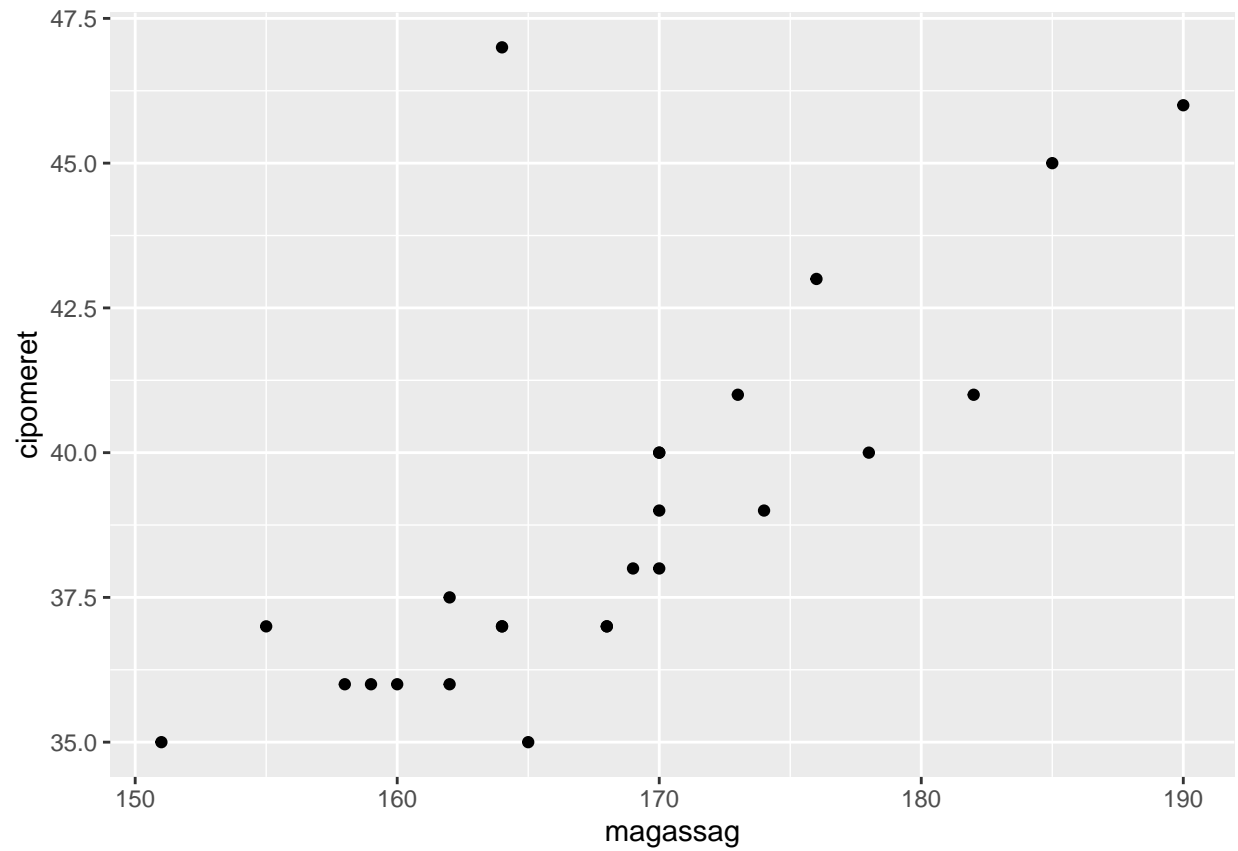
```
mydata_corrected %>%  
  ggplot() +  
  aes(x = cipomeret) +  
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

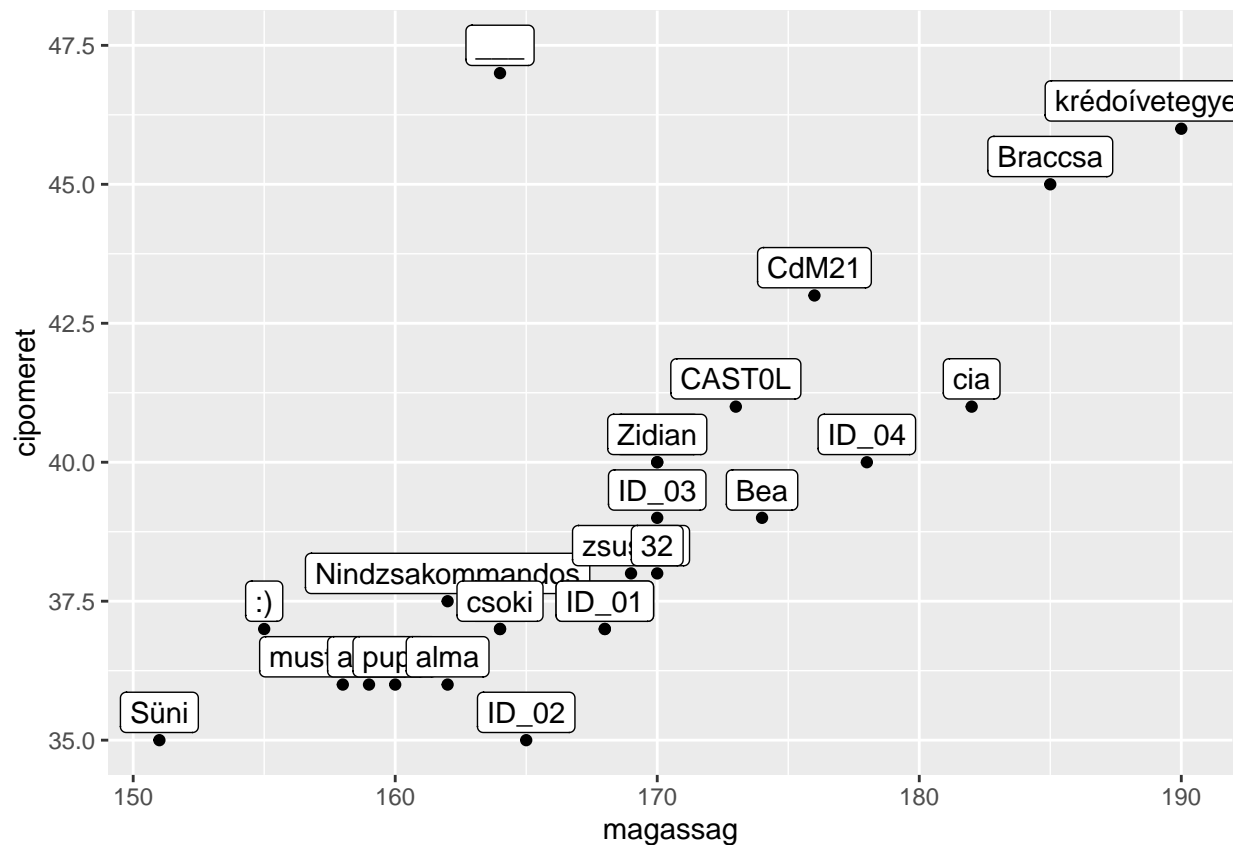


```
# scatterplot
mydata_corrected %>%
  ggplot() +
  aes(x = magassag, y = cipomeret) +
  geom_point()
```





```
# scatterplot with labels
mydata_corrected %>%
  ggplot() +
  aes(x = magassag, y = cipomeret, label = jelige) +
  geom_point() +
  geom_label(nudge_y = 0.5)
```

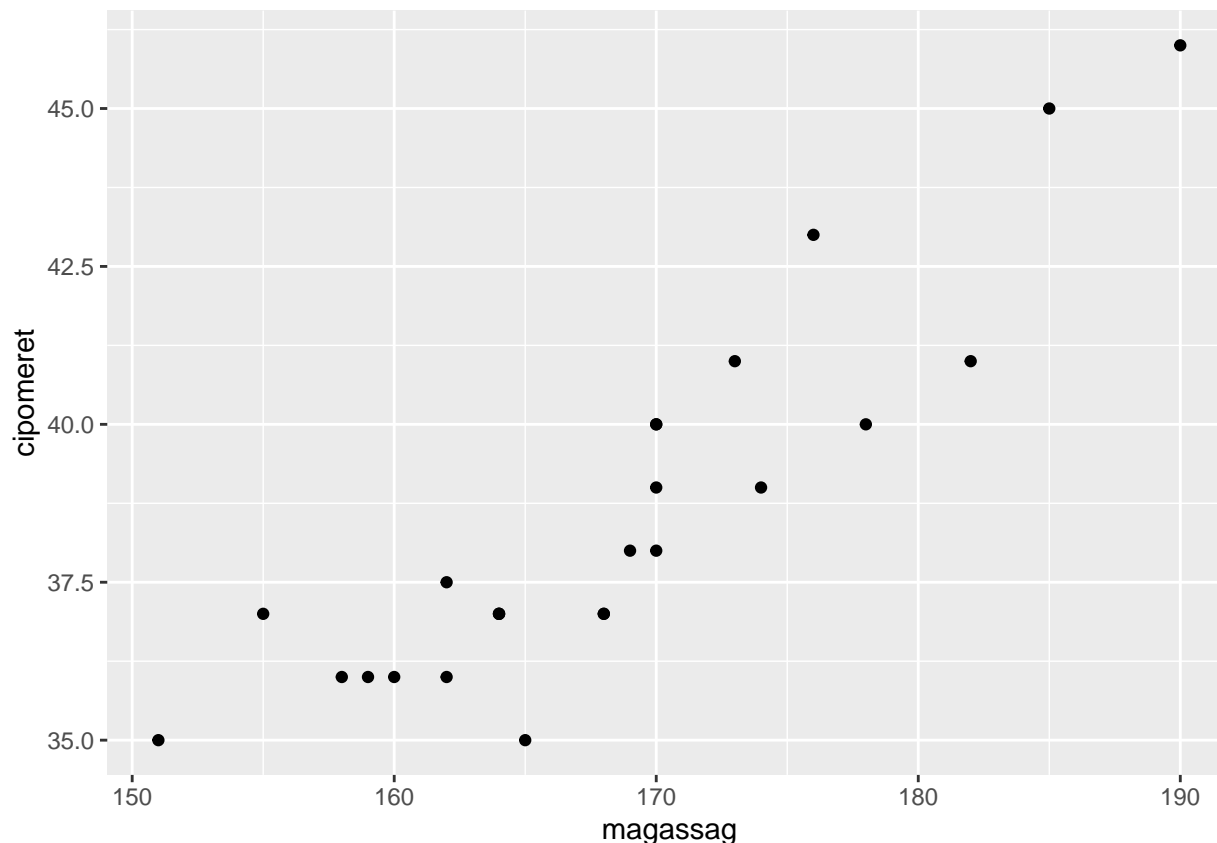


Van egy ember akinek a cipomeret értéke kiugro a többi adatban látható trendből a magassággal való összehasonlításban. A vizsgálati személy jelezte hogy ez egy hibás adatbevitel volt, a helyes cipomeret 37, nem 47. Ezt javítjuk.

(Általában ilyen beszó információ nem áll rendelkezésre, ilyen esetben a hasonló kiugró adatot vagy kizárnánk teljesen az elemzésből, vagy megtartanánk hogy realis az érték, de megjegyeznénk a kutatási jelentésben/cikkben, hogy volt egy kiugró eset.)

```
mydata_corrected = mydata_corrected %>%
  mutate(cipomeret = replace(cipomeret, jelige == "___", 37))

# scatterplot
mydata_corrected %>%
  ggplot() +
  aes(x = magassag, y = cipomeret) +
  geom_point()
```



## Bejoslas linearis modellel

### Egyszeru linearis modell felepítése

A regresszio **bejoslasra** vagy becslesre valo. Vagyis szeretnenk megtudni egy valtozo erteket (ezt altalaban bejosolt valtozonak vagy kimeneti valtozonak nevezzuk) mas bejoslo (prediktor) valtozok erteke alapjan.

Az alabbi peldaban szeretnenk megbecsulni (bejosolni/prediktalni) az egyes személyek EU cipomeretét (ez a bejosolt/kimeneti valtozó) a személy magassaganak ismeretében (ez a bejoslo/prediktor valotozo). Ehhez eloszor az elozetes adataink hasznalataval felepitunk egy regressziós modellt.

A linearis regressziós modellt az **lm()** funkcióval építjük. Mindig úgy kell felepíteni, hogy a bejosolni kívánt valtozoval kezdünk (cipomeret), majd a  $\sim$  jel után írjuk a bejoslo valtozot (magassag). A kód végén pedig azt specifikáljuk, melyik adattáblában találhatóak ezek a valtozok a “data = ...” parameter megadasaval. A modellt elmentjük egy objektumba (ezt most mod1-nek neveztük el, de bárminek elnevezhetnénk).

Az egyszeru linearis regresszionál (simple linear regression) csak egy bejoslo valtozonk van.

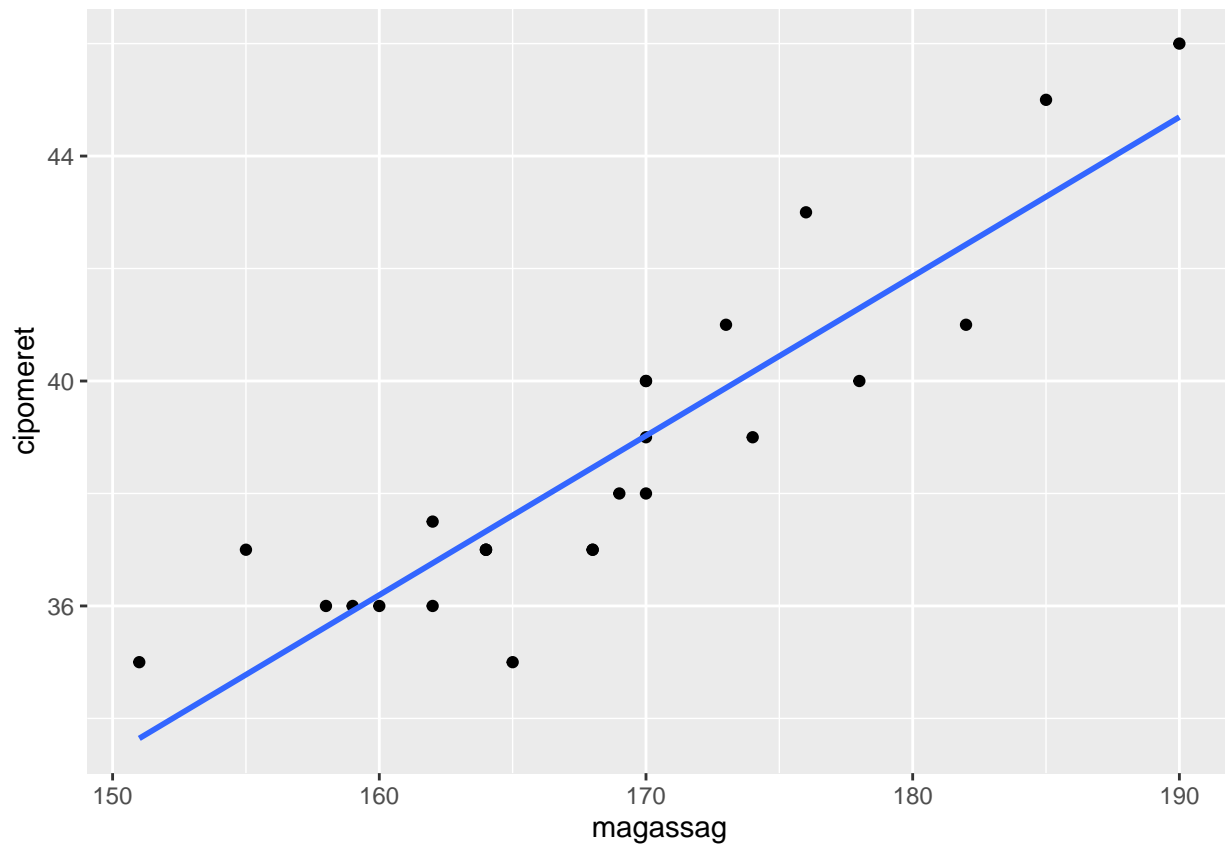
A linearis regresszióban több bejoslo valtozot is használhatunk, ilyenkor többszörös linearis regresszionak nevezzük az eljárást (multiple linear regression). Errol majd később lesz szó.

```
mod1 <- lm(cipomeret ~ magassag, data = mydata_corrected)
```

Az linearis regresszióban a kimeneti valtozo és a prediktor közötti kapcsolatot egy egyenessel modellezzük. A modell az az egyenes lesz ami a legközelebb van a pont diagram pontjaihoz.

```
mydata_corrected %>%
  ggplot() +
  aes(x = magassag, y = cipomeret) +
```

```
geom_point() +  
geom_smooth(method = "lm", se = F)
```



A regressziós modell megad egy matematikai egyenletet, amibe a prediktor változó értéket behelyettesítve megkaphatjuk a legjobb becslést a kimeneti változó értékeire. Ezt az egyenletet regressziós egyenletnek (regression equation) nevezzük.

A **regressziós egyenletet** így formalizáljuk:  $Y = b_0 + b_1 \cdot X_1$ , amelyben  $Y$  a kimeneti (bejósolt) változó becslés értéke, a  $b_0$  egy állandó/konstans érték, amit legtöbbször intercept-nek neveznek, a  $b_1$  a regressziós együttható, az  $x_1$  pedig a bejósoló (prediktor) értéke az adott személynél.

Vagyis úgy kaphatunk egy becslést az  $Y$  bejósolt változó értékeire (magasság), ha a konstanshoz hozzáadjuk a regressziós együttható és a prediktor értékek szorzatát.

Ha klistázzuk a modell objektumot (mod1), akkor megkaphatjuk a **regressziós egyenletet** erre a modellre amit most építettünk.

```
mod1  
  
##  
## Call:  
## lm(formula = cipomeret ~ magassag, data = mydata_corrected)  
##  
## Coefficients:  
## (Intercept)      magassag  
##      -9.1332       0.2833
```

Tegyük fel hogy a regressziós egyenlet elemei a következők:

- intercept (b0) = -9.13
- a magassaghoz tartozó regressziós együttható (b1) = 0.28

Ezeket az adatokat a modell objektum kilistázásával olvashatjuk le a `Coefficients:` részből.

Ez azt jelenti, hogy a cipómeretet bejósoló regressziós egyenlet a következő:

$$\text{cipómeret} = -9.13 + 0.28 * \text{magassag}$$

vagyis egy 170 cm magas ember esetén a modell által becsült cipómeret:

$$-9.13 + 0.28 * 170 = 38.47$$

Ezt a számítást nem kell kézzel vagy fejben megcsinálni, ehelyett használhatod az R `predict()` függőjét a bejósolt érték kiszámítására bármilyen, vagy akár több prediktor-értékre is.

A `predict()` függő használatahoz meg kell adnunk egy adattáblát (`data.frame` vagy `tibble`-t) ami a prediktor értékeit tartalmazza, amit a kimeneti változó megbecsülésére, bejósolására szeretnénk használni.

```
magassag = c(150, 160, 170, 180, 190)
magassag_df = as.data.frame(magassag)

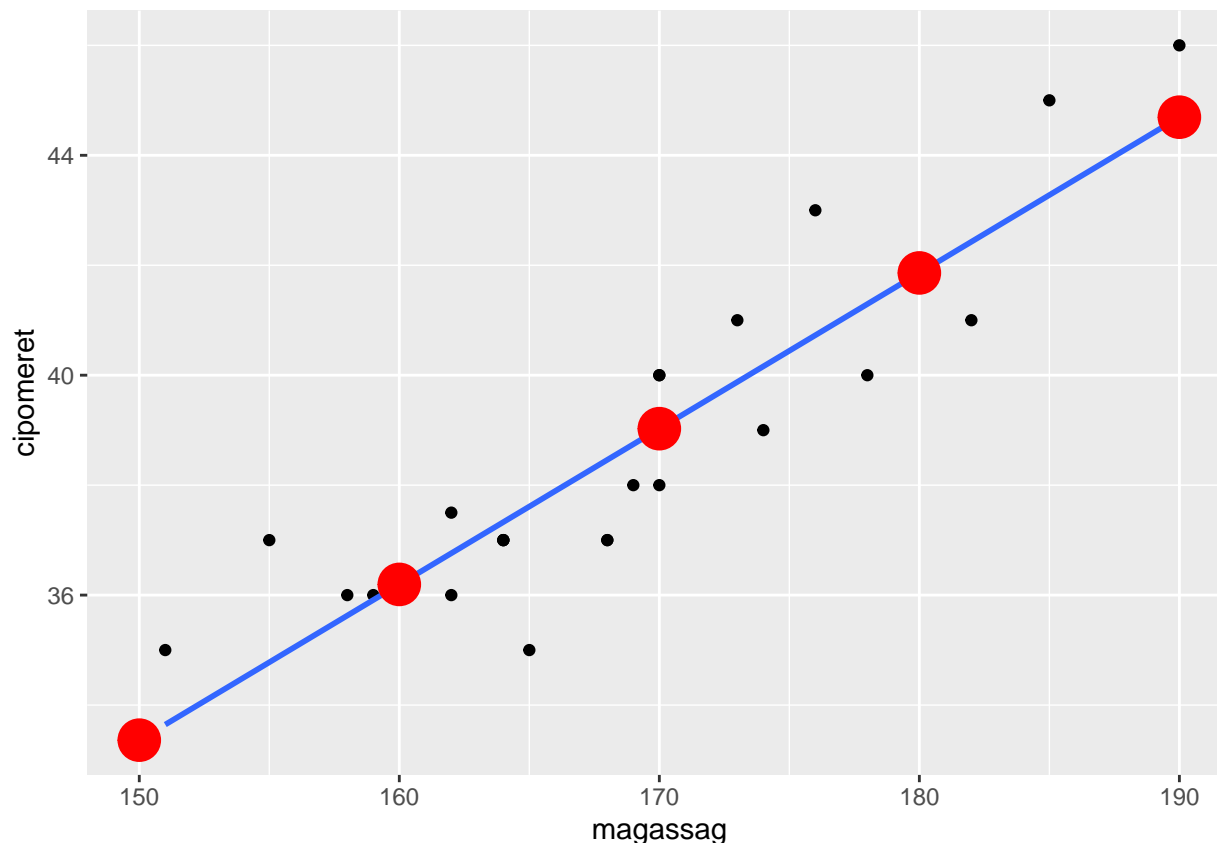
predictions = predict(mod1, newdata = magassag_df)

magassag_df_with_predicted = cbind(magassag_df, predictions)
magassag_df_with_predicted
```

```
##   magassag predictions
## 1      150      33.36134
## 2      160      36.19430
## 3      170      39.02727
## 4      180      41.86024
## 5      190      44.69321
```

Vagyuk észre hogy a bejósolt értékek mind pontosan a regressziós egyenesre esnek. Mászóval a regressziós egyenes minden lehetséges prediktorértékre megadja a kimeneti változó bejósolt értékét.

```
mydata_corrected %>%
  ggplot() +
  aes(x = magassag, y = cipómeret) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  geom_point(data = magassag_df_with_predicted, aes(x = magassag, y = predictions), col = "red", size =
```



### Gyakorlas

1. Epits egy egyszeru linearis regresszio modellt az `lm()` fugvennyel amiben az **energiaszint\_1** a kimeneti valtozo es az **alvas\_atalaban\_1** a prediktor. A modell eredmenyet mentsd el egy objektumba.
2. Ird le a regresszios fuggvenyt amivel bejosolhato az energiaszint.
3. Ertelmezd a regresszios fuggvenyt. Aki tobbet alszik annak magasabb vagy alacsonyabb az energiaszintje? (Egy abra segithet)
4. Ertelmezd a regresszios fuggvenyt. Aki egy oraval tobbet alszik mint masok, annak mennyivel varhato hogy magasabb lesz az energiaszintje?
5. Ennek a modellnek a segitsegevel becsuld meg az energiaszintjet olyan embereknek akik altalaban 5, 7, vagy 9 orat alszanak.

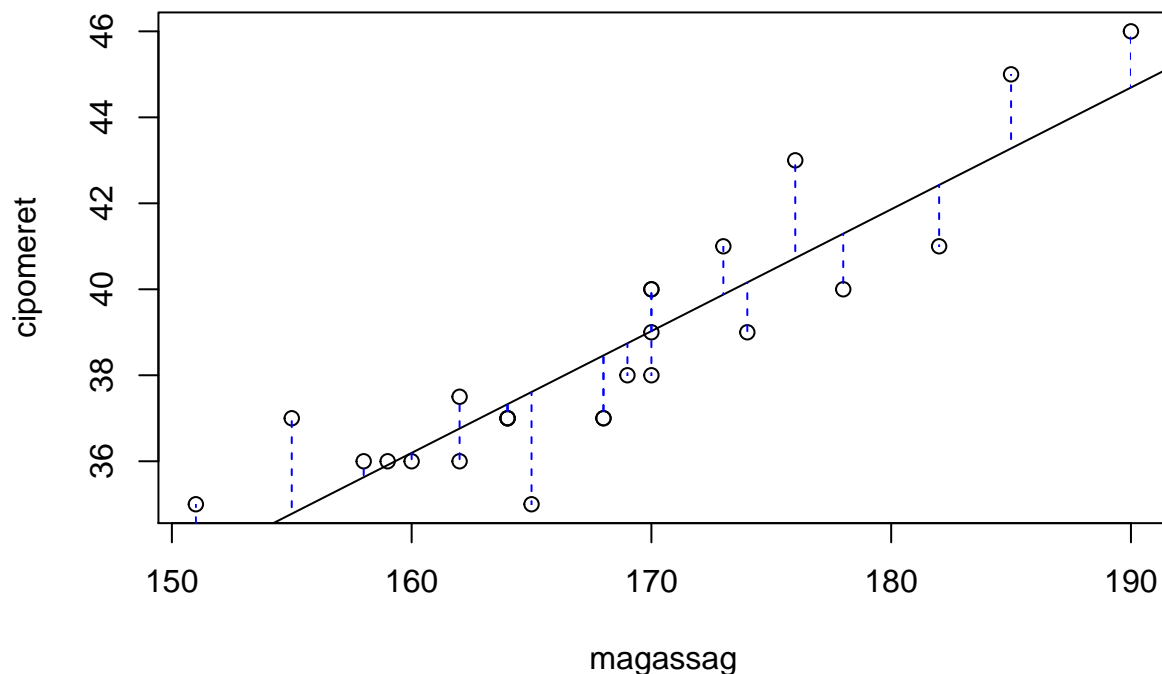
### Milyen jo a modellem? (modellilleszkedes)

#### Hogyan merheto a becslesi/bejoslasi hatekonysag?

A modell becslesi hatekonysagat tobb fele keppen lehet merni. A legkezenfekvobb modszert, hogy meghatarozzuk, a modell becslese mennyire esett tavol a valos bejosolni kivant ertekektol. Vagyis megmerjuk a modell figyelembevetele utan fennmarado "hibat".

Ezt konnyen megtehetjuk egy olyan adatbazisban, ahol rendelkezesunkre all a bejosolni kivant valtozo valos erteke, ugy hogy kivonjuk egymasbol a valos ertekeket es a modell által becsult ertekeket. Ez a rezidualis (fennmarado) hiba, masneven **residual error**.

```
error_plotter(mod1, col = "blue")
```



Ha vesszük az összes ilyen hiba érték abszolútértéket, és összeadjuk őket, megkapjuk a modell reziduális abszolút hiba (residual absolute difference - RAD) értéket.

Ennel azonban jóval gyakoribb hogy a reziduális hiba négyzetösszeget használják (**residual sum of squares** - RSS) a statisztikában. Vagyis az egyes reziduális hiba értékeket négyzetre emelik, majd összeadják őket.

Az alábbi példában a `mod1` eredeti adattáblájának magasságértékeit használjuk a cipomeret becslt értékek kiszámítására (`predict(mod1)`), és ezt vonjuk ki az ugyan ezen adattáblában szereplő valós cipomeret értékekből, így kapjuk meg a reziduális hibaértékeket. Majd egyenként vagy az abszolútértékeket (RAD), vagy a négyzetüket vesszük (RSS), és összeadjuk őket a `sum()` függvénnyel.

```
RAD = sum(abs(mydata_corrected$cipomeret - predict(mod1)))
RAD
```

```
## [1] 26.29851
```

```
RSS = sum((mydata_corrected$cipomeret - predict(mod1))^2)
RSS
```

```
## [1] 39.15215
```

### Hasznos a modellünk?

Azt, hogy mennyire hasznos a modellünk (mennyit nyerünk azzal, hogy ezt a modellt használjuk), meghatározhatjuk úgy, hogy összehasonlítjuk a reziduális hibát abban az esetben amikor a modellünket használjuk (vagyis amikor figyelembe vesszük a prediktoraink értéket) egy olyan esettel, amikor a prediktorokat egyáltalán nem vesszük figyelembe, csak a bejósolni kívánt változó átlagát használjuk a becslésre.

Az alábbi kodban építünk egy olyan új modellt, ahol nem vesszük figyelembe semmilyen másik változót, csak

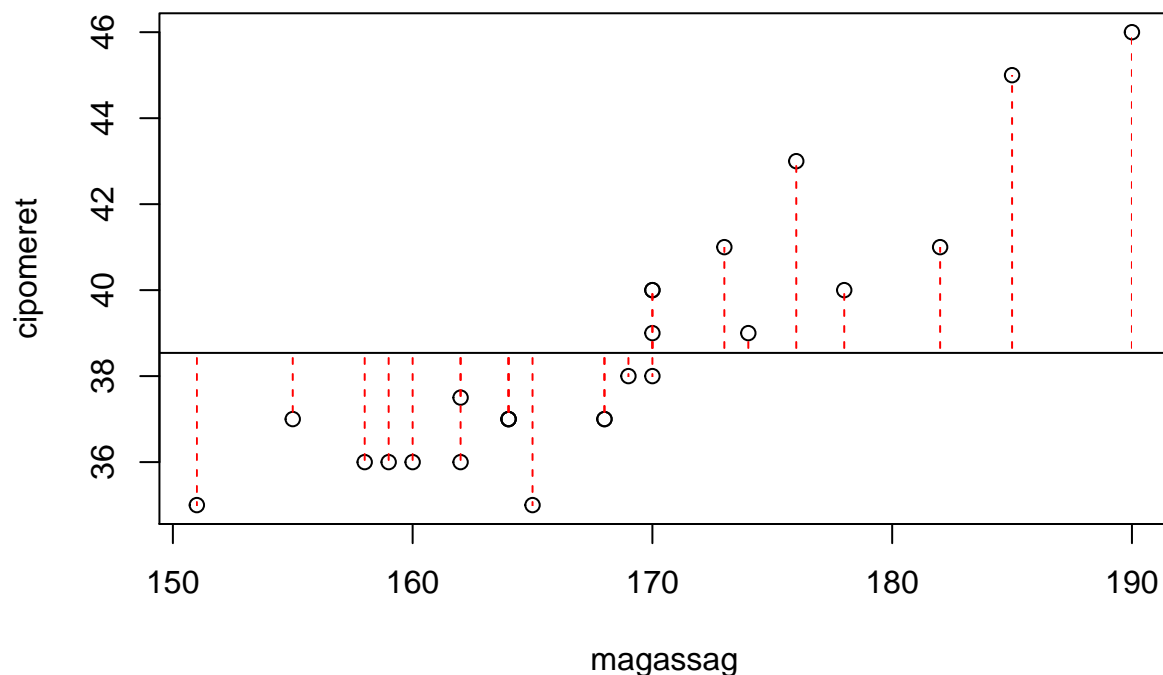
a cipomeret átlagát, és azt használjuk fel a cipomeret becsléseként. (pl. ha tudjuk, hogy a populációban az átlagos cipomeret 38, akkor mindenkinek ezt a cipomeretet becsüljük majd, függetlenül attól, hogy milyen magas az illető). Ezt a modellt **null modellnek** nevezzük. Azt, hogy a bejósolt változó átlagát akarjuk becslesre használni, úgy adhatjuk meg, hogy a  $\sim$  után csak egy 1-et rakunk, nem írunk más változónevet.

Ez persze nagy reziduális hibához vezet (hiszen bár ez a populációban az átlagos, mégis a legtöbb embernek nem pont 38-as a lába). A null modell által produkált reziduális hibát ugyan úgy számoljuk ki, mint a többi modellnél a residual sum of squares-et, viszont ennek van egy speciális neve is az irodalomban, ezt úgy hívják, hogy **total sum of squares** (TSS), mert ez a lehetséges legegyszerűbb meg értelmes modell, ami általában azért nagy hibával jár, így ezt vesszük a “teljes” hiba mennyiségnek, és ehhez viszonyítjuk a többi modell által elért hibát.

Alább kiszámoljuk a TSS-t. Latható hogy a formula ugyan az mint az RSS eseten.

```
mod_mean <- lm(cipomeret ~ 1, data = mydata_corrected)

error_plotter(mod_mean, col = "red", x_var = "magassag") # visualize error
```



```
TSS = sum((mydata_corrected$cipomeret - predict(mod_mean))^2)
TSS
```

```
## [1] 202.96
```

### Mennyivel jobb a modellünk a null modellnél?

Azt, hogy mennyi információt nyertünk a kimeneti változó változékonyságáról (variance) a prediktorok figyelembevételével ahhoz képest ha a null modellt vettük volna figyelembe, az  $R^2$  statisztika mutatja meg. Ennek a formulája:  $1 - (RSS/TSS)$



```
R2 = 1-(RSS/TSS)
R2
```

```
## [1] 0.8070942
```

Tegyuk fel hogy az  $R^2$  ebben az esetben 0.81. Ez azt jelenti, hogy a prediktorok figyelembevetelevel (a mi esetunkben ez a magassag), a cipomeret valtozekonysaganak 81%-at tudjuk megmagyarazni.

$R^2 = 1$  azt jelenti, hogy a kimeneti valtozo variabilitasat teljesen meg tudjuk magyarazni a prediktorok ismereteben.  $R^2 = 0$  azt jelenti, hogy a kimeneti valtozo variabilitasat egyaltalan nem magyarazzak meg a prediktorok

### Hasznos a modellunk a populaciora nezve is?

Azt, hogy a modellunk hasznos-e a kimeneti valtozo bejoslasara populacio-szinten is, ugy tudjuk meghatarozni, hogy meghatarozzuk, a prediktorokat tartalmazo modell **szignifikansan jobb-e** mint a null modell a kimeneti valtozo becslesere?

Egy F tesztet használhatunk a szignifikancia szint meghatarozosahoz. Ezt ugy kaphatjuk meg az R-ben hogy a ket modell által produkalt rezidualis hibát az `anova()` funkcióval hasonlíthatjuk össze, melybe a null modell és a prediktorokat tartalmazó modell objektumot kell beletenni (akár többet mint két modellt is lehet egyszerre). Itt az F-teszthez tartozó teszt statisztikát és p-értéket nezzük, ha a szignifikanciára vagyunk kíváncsiak.

```
anova(mod_mean, mod1)
```

```
## Analysis of Variance Table
##
## Model 1: cipomeret ~ 1
## Model 2: cipomeret ~ magassag
##   Res.Df    RSS Df Sum of Sq    F    Pr(>F)
## 1      24 202.960
## 2      23  39.152   1    163.81 96.229 1.097e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### Az egyszeru megoldas

A fentiekben lepesrol lepesre elvegeztuk a regresszio legfontosabb szamitasait saját magunk által írt kóddal, hogy megertsetek, mi folyik egy linearis regresszio során a “motorhazteto alatt”. De ahogy sejtthetitek, mindegy az R-ben van egy gyorsabb és jóval egyszerubb megoldas:

A `model summary()` kikeresevel mindez a fenti információ megkapható, és még több is.

Itt megtalálod az  $R^2$  értéket, a modell null modellel való összehasonlításának F teszt statisztikáját és szignifikanciáját, és még a regressziós egyenletet elemeit is.

```
summary(mod1)
```

```
##
## Call:
## lm(formula = cipomeret ~ magassag, data = mydata_corrected)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.6108  -1.0273  -0.1943   0.9727   2.2729
##
## Coefficients:
```

```
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.13318    4.86683  -1.877  0.0733 .
## magassag    0.28330    0.02888   9.810 1.1e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.305 on 23 degrees of freedom
## Multiple R-squared:  0.8071, Adjusted R-squared:  0.7987
## F-statistic: 96.23 on 1 and 23 DF,  p-value: 1.097e-09
```

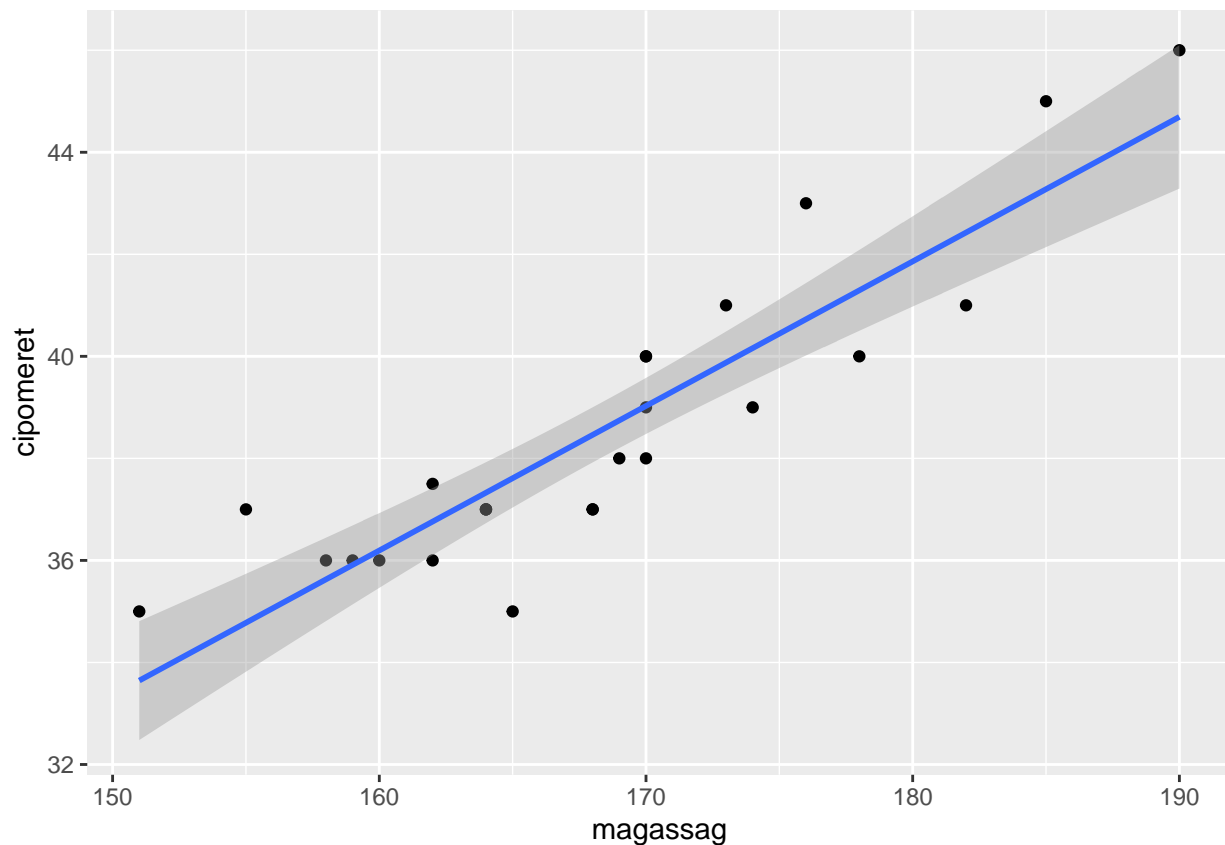
A regressziós együtthatók (regression coefficients) konfidencia intervallumát a `confint()` paranccsal lehet kilistázni.

```
confint(mod1)
```

```
##           2.5 %    97.5 %
## (Intercept) -19.2009731  0.9346187
## magassag      0.2235552  0.3430383
```

A regressziós becslés konfidencia intervallumát pedig a `geom_smooth()`-al lehet vizualizálni.

```
ggplot(mydata_corrected, aes(x = magassag, y = cipomeret))+
  geom_point()+
  geom_smooth(method='lm', formula=y~x)
```



## Gyakorlás

1. (Ezt nem kell megtenned ha ezt már megtetted az előző gyakorlatban, csak használd ugyan azt a model objektumot) Építs egy egyszerű lineáris regresszió modellt az `lm()` függvénnyel amiben az

**energiaszint\_1** a kimeneti változó és az **alvas\_atalaban\_1** a prediktor. A modell eredményét mentsd el egy objektumba mentsd.

2. Listázd ki a model summary-t a `summary()` függvénnyel
  3. Olvasd le a model RSS-jét
  4. Olvasd le hogy a model ami tartalmazza az `alvas_atalaban_1` prediktort szignifikánsan jobb bejelölje-e az energiaszintnek mint a null model.
  5. Határozd meg a regressziós együtthatók konfidencia intervallumát a `confint()` függvénnyel
-