Lineáris Regresszió A-tól Z-ig

Böjte Berta

Data Klub

2025.02.12.

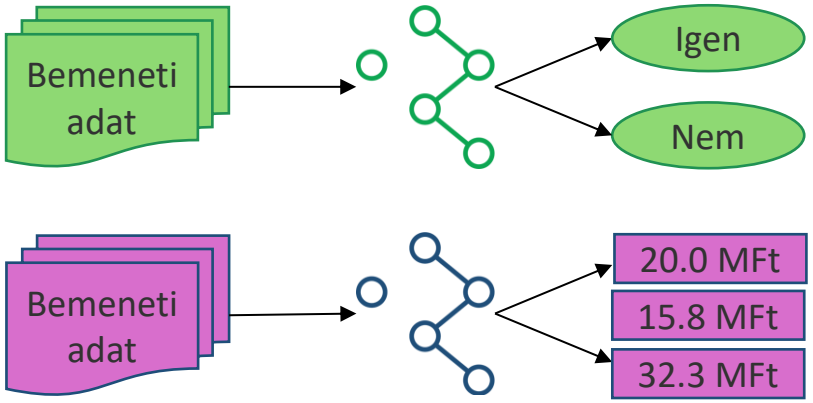
Gépi tanulási módszerek – áttekintő

Felügyelt

- Egy célváltozó jövőbeli előrejelzése
- Tanulás címkézett adatpárok alapján
  - Célváltozó: amit előre jelezni
  - Magyarázó változók: ami alapján előre jelzünk

Típusok és algoritmusok

- **Klasszifikáció** – Címke előrejelzése
  - Logisztikus regresszió
  - Döntési fa alapú algoritmusok
- **Regresszió** – Folytonos szám előrejelzése
  - Lineáris regresszió
  - Döntési fa alapú algoritmusok

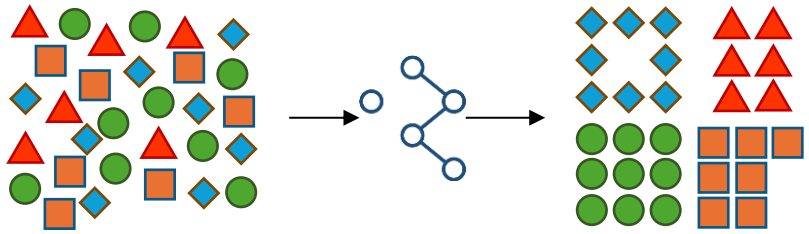


Nem felügyelt

- Mintázatok keresése az adatokban
- Címkézetlen adatok alapján

Típusok és algoritmusok

- **Anomália detekció**
  - Isolation forest
- **Klaszterezés**
  - K-means (K-közép)
  - Hierarchikus klaszterezés

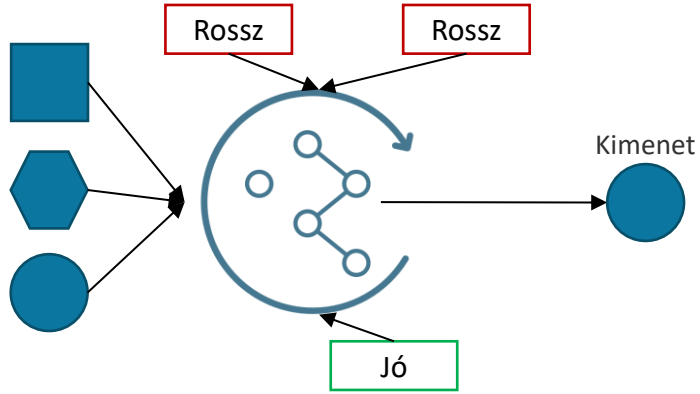


Megerősítő

- Gépek tanítása „emberi” módon
- Tanulás jutalmazás és büntetés alapján.

Típusok és algoritmusok

- **Cselekvés – jutalom alapú tanulás**
  - Monte Carlo
  - Q-learning
  - SARSA



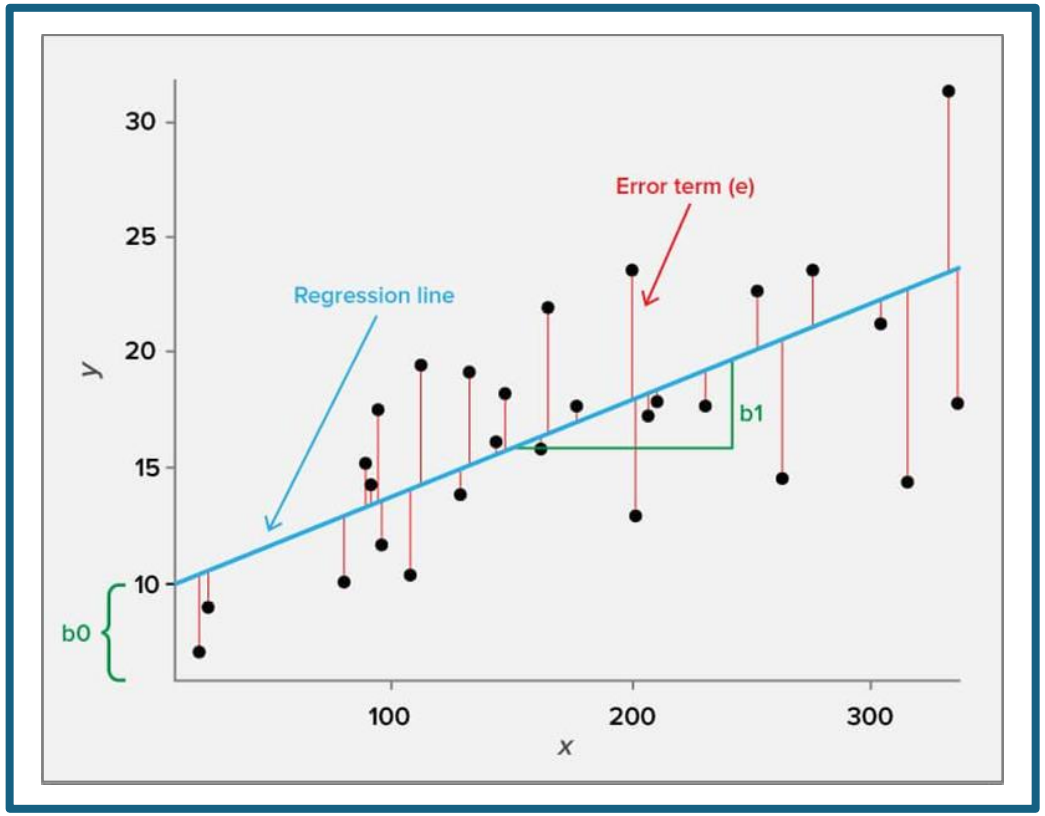
Lineáris regresszió – az alapok

Lineáris Regresszió - alapok

$$\hat{y} = \beta_0 + \beta_1 x_1 + \varepsilon$$

Célváltozó | Tengelymetszet | Koefficiensek | Magyarázó változók | Hiba

- **Cél:** egyenes/sík/hipersík illesztése, amely minimalizálja a négyzetes hibát (SSE)
- $\hat{y}$ - célváltozó (target)
  - függő változó, ezt szeretnénk minél pontosabban előre jelezni/vizsgálni
- β_0 - tengelymetszet (intercept)
  - konstans tag – ha minden magyarázó változó 0, akkor ez az érték lesz a predikció
- β_1 - Koefficiens / béta / meredekség /súly (coefficient/beta)
  - a célváltozó változását jelzi a hozzá tartozó magyarázó változó egységnyi változása esetén, miközben a többi változót állandónak tartja.
- x_1 - magyarázó / független változó (feature / independent variable):
  - Ezekkel szeretnénk előre jelezni a célváltozót, egy adatpont tulajdonságai.
- ε – hiba (residual):
  - A célváltozó valós, és előre jelzett értéke közötti különbség, a predikció tévedése.



Többváltozós lineáris regresszió

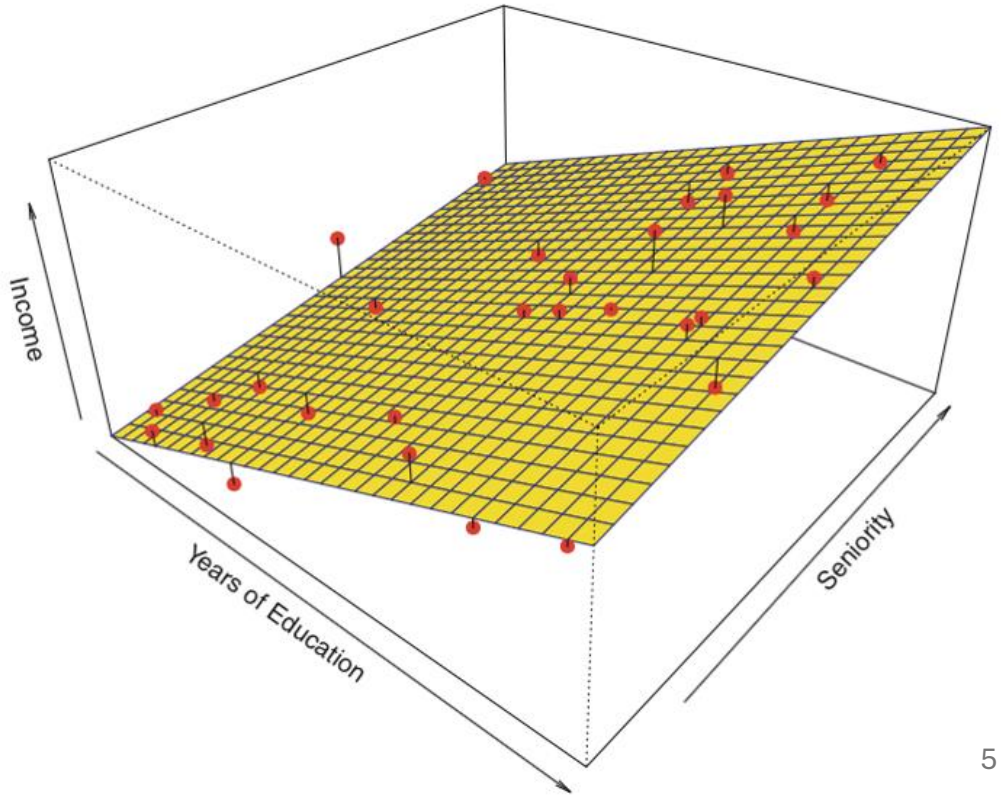
- **Egyváltozós lineáris regresszió:**
  - Egy változó segítségével szeretnénk a célváltozót előre jelezni
  - Angolul simple linear regression
- **Többváltozós lineáris regresszió**
  - Több változó segítségével szeretnénk a célváltozót előre jelezni
  - Angolul multiple linear regression / multivariable linear regression
  - FONTOS, hogy létezik a multivariate linear regression kifejezés is, de ez mást takar – itt több, egymással összefüggő célváltozónk van, amiket ugyanazzal a magyarázó változó halmazzal szeretnénk előre jelezni

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon$$

$\hat{y}$ → Income (fizetés)

x_1 → Years of education (tanulással töltött évek)

x_2 → Seniority (tapasztalati szint)



Tanítási módszertanok

Egyenes illesztése – tanítási módszerek

- **Cél:** A koefficiensek és a tengelymetszet meghatározása úgy, hogy a hibákat a lehető legkisebbre csökkentsük.
- **Négyzetes hibaösszeg (SSE – Sum of Squared Errors)** vagy az **átlagos négyzetes hiba (MSE)** minimalizálása

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

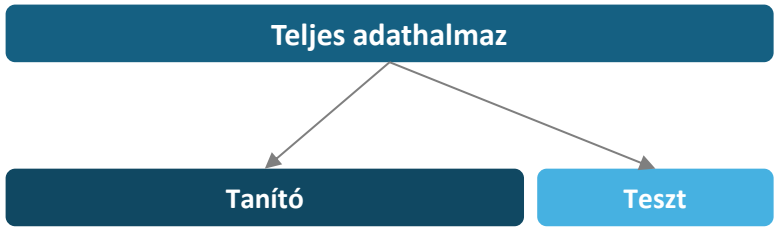
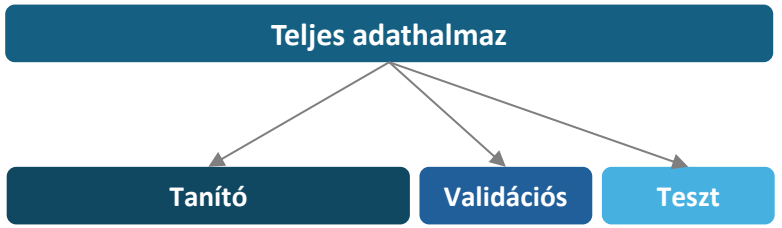
	Ordinary Least Squares (OLS)	Stochatic Gradient Descent (SGD)
MÓDSZER	Zárt képlet alkalmazása az összes adatpontra – mátrix műveletekkel	• Random koefficiens értékekkel kezd • Kiszámítja a költségfüggvény gradiensét • Gradienssel ellentétes irányban változtatja a koefficiens értékeket • 2-3 lépések ismétlése, amíg el nem éri a költségfüggvény minimumát vagy az előre meghatározott lépésszámot
ELŐNY	• Gyors és hatékony • Legjobb becslést adja az együtthatókra	• Nagy adathalmazokon is hatékony • Alkalmas online tanulásra
HÁTRÁNY	• Kis / közepes adathalmazokon hatékony csak • Nem alkalmas online tanulásra	• Nekünk kell a hiperparamétereit (pl.: iterációk száma, ütemezés) beállítani • Nem biztos, hogy megtalálja a legjobb becslést az együtthatókra

Erről bővebben Kozma-Renge Kriszta *Statisztika Data Science-hez - 4) Lineáris Regresszió* előadásában hallhattok

Reguralizáció

Tanító-teszt szétosztás

- Annak érdekében, hogy mérni tudjuk a modellünk teljesítményét, mennyire tanulta meg a szabályokat, és mennyire tud jól általánosítani, szétosztjuk az adatokat több egységre.
- Az adatok egy részén tanítjuk a modellt, a másik(okon) pedig visszamérjük
- Erre 3 módszertanból választhatunk

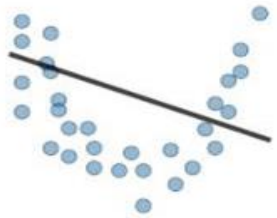
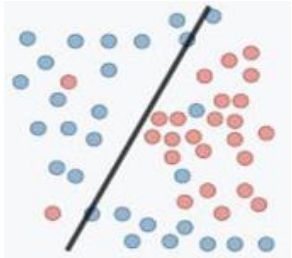
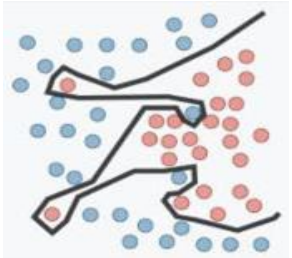
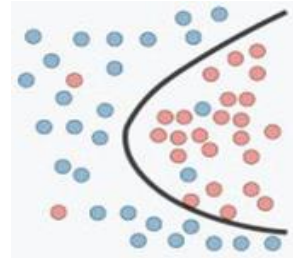
	Train – Test Split	Train – Validation – Test Split	Cross-validation*
LEÍRÁS	Egyszerűen csak ketté osztjuk az adatokat: • 70-80% → tanító adat • 30-20% → teszt adat	Az adatot 3 részre osztjuk: • 60% → tanító adat • 20% → validációs adat • 20% → teszt adat	Az adatokat több részre osztjuk – egy rész mindig a teszt, a többi pedig a tanító rész. Több modellt építünk a különböző részeken, és a végén átlagoljuk a modellek teljesítményét a kiértékeléshez.
PRO - CON	+ Jó opció kis elemszámú adatok esetén - Teszt adatot „látja” a modell a hiperparaméter optimalizáláskor	+ Nincs adatszivárgás, mivel a validációs halmazt használjuk a hiperparaméterek állítgatására, a teszt halmazt pedig a végleges kiértékelésre + „Kellően nagy” adathalmaz kell hozzá	+ Kis elemszámú adatnál is jól használható + Pontosabb eredmények a teszthalmazon, csökkentheti a túltanulást - Erőforrás és időigényes lehet
ILLUSZTRÁCIÓ			

Erről bővebben Windhager Eszter *Machine Learning Algoritmusok Kiértékelése: Mutatók, Mérőszámok, Fontos Értékek* előadásában hallhattok

*Több keresztvalidációs módszer létezik, a K-fold cross validation technikát reprezentálja az ábra

Alul- és túltanulás

- A prediktív modellezés célja egy adathalmaz mintázatainak és kontextusainak megismerése oly módon, hogy a későbbiekben ezek a tanult tulajdonságok segíthetnek előre látni egy korábban nem látott adatpontot.
- Ennek eléréséhez meg kell találnunk az egyensúlyt az alul- és túlillesztés között

	Alultanulás	Túltanulás	Pont jó
DEF	Alultanulás esetén a modell nem tudja sem a tanító adatokat modellezni, sem új adatokra általánosítani	Túltanuláskor a modell oly mértékben tanulja meg a tanító adatok részleteit, hogy az negatívan hat az új adatokon a teljesítményére	Adathalmaz mintáinak, kontextusának megtanulása pont annyira, hogy jó előrejelzést tudjunk adni a nem látott adatokról
TÜNETEK	• Magas hiba a tanító adatokon • Tanító adati hiba közel van a teszt adati hibához • Magas torzítás (bias)	• Nagyon alacsony tanító adati hiba • A tanító adati hiba sokkal kisebb, mint a teszt adati hiba • Magas variancia	• A tanító adati hiba kicsit kisebb, mint a teszt adati hiba
ILLUSZTRÁCIÓ	Regresszió  Klasszifikáció 	Regresszió  Klasszifikáció 	Regresszió  Klasszifikáció 
KEZELÉS	• Összetettebb modell alkalmazása • Több változó bevonása	• Reguralizáció alkalmazása • Több adat bevonása a modellezésbe	

Erről bővebben Windhager Eszter *Machine Learning Algoritmusok Kiértékelése: Mutatók, Mérőszámok, Fontos Értékek* előadásában hallhattok

Reguralizáció

- Túltanulás elkerülése érdekében használjuk, arra kényszerítjük a modellt, hogy általánosabb érvényű szabályokat tanuljon meg, ne pedig a specifikumokat
- OLS veszteségfüggvény (loss function): $\mathcal{L}(\beta; \mathrm{A}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_{1^*} x_i))^2 + \mathrm{A}\sum_{j=1}^{p}|\beta_j|$

Lasso (L1)

DEF.

A veszteségfüggvényhez hozzáadjuk a súlyok abszolút értékének az összegét, szorozva a „büntető” paraméterrel

KÉPLET

A – alfa, büntető paraméter
β_j – koefficiensek
j – paraméterek száma

$$\mathrm{A}\sum_{j=1}^{p}|\beta_j|$$

HASZ.

Változó szelekcióra használhatjuk magas dimenziójú, sok változós adat esetén

PROS

- Nem fontos változók bétáját 0-ra csökkenti
- Jobban értelmezhető modellt hoz létre
- Darabszámra, sok, fontosságra kevés változó esetén működik jól

CONS

Korreláló változók esetén megbízhatatlan lehet, mert „önkényesen” választhat közülük

Reguralizáció

- Túltanulás elkerülése érdekében használjuk, arra kényszerítjük a modellt, hogy általánosabb érvényű szabályokat tanuljon meg, ne pedig a specifikumokat
- OLS veszteségfüggvény (loss function): $\mathcal{L}(\beta; \mathrm{A}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_{1^*} x_i))^2 + \mathrm{A}\sum_{j=1}^{p}{\beta_j}^2$

	Lasso (L1)	Ridge (L2)
DEF.	A veszteségfüggvényhez hozzáadjuk a súlyok abszolút értékének az összegét, szorozva a „büntető” paraméterrel	A veszteségfüggvényhez hozzáadjuk a súlyok négyzetének az összegét, szorozva a „büntető” paraméterrel
KÉPLET	A – alfa, büntető paraméter β_j – koefficiensek j – paraméterek száma $\mathrm{A}\sum_{j=1}^{p}\lvert\beta_j\rvert$	A – alfa, büntető paraméter β_j – koefficiensek j – paraméterek száma $\mathrm{A}\sum_{j=1}^{p}{\beta_j}^2$
HASZ.	Változó szelekcióra használhatjuk magas dimenziójú, sok változós adat esetén	Multicollinearitás csökkentésére, használjuk. Nem nullázza ki a súlyokat, de „összenyomja” őket
PROS	• Nem fontos változók bétáját 0-ra csökkenti • Jobban értelmezhető modellt hoz létre • Darabszámra, sok, fontosságra kevés változó esetén működik jól	• Minden változót megtart • Csökkenti a multicollinearitás negatívumait a korrelációs hatás „szétosztásával”
CONS	Korreláló változók esetén megbízhatatlan lehet, mert „önkényesen” választhat közülük	Magas dimenziószám mellett nehéz lehet a modell értelmezése

Reguralizáció

- Túltanulás elkerülése érdekében használjuk, arra kényszerítjük a modellt, hogy általánosabb érvényű szabályokat tanuljon meg, ne pedig a specifikumokat
- OLS veszteségfüggvény (loss function): $\mathcal{L}(\beta; \mathrm{A}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_{1^*} x_i))^2$ $+ \mathrm{A}\left(a\sum_{i=1}^{n}|\beta_j| + (1-a)\sum_{i=1}^{n}\beta_j^2\right)$

	Lasso (L1)	Ridge (L2)	Elastic Net
DEF.	A veszteségfüggvényhez hozzáadjuk a súlyok abszolút értékének az összegét, szorozva a „büntető” paraméterrel	A veszteségfüggvényhez hozzáadjuk a súlyok négyzetének az összegét, szorozva a „büntető” paraméterrel	Kombinálja az L1 és L2 reguralizációt
KÉPLET	A – alfa, büntető paraméter β_j – koefficiensek j – paraméterek száma $\mathrm{A}\sum_{j=1}^{p}\lvert\beta_j\rvert$	A – alfa, büntető paraméter β_j – koefficiensek j – paraméterek száma $\mathrm{A}\sum_{j=1}^{p}\beta_j{}^2$	α – L1 és L2 "büntetés" arányát határozza meg $\mathrm{A}\left(a\sum_{i=1}^{n}\lvert\beta_j\rvert + (1-a)\sum_{i=1}^{n}\beta_j^2\right)$
HASZ.	Változó szelekcióra használhatjuk magas dimenziójú, sok változós adat esetén	Multicollinearitás csökkentésére, használjuk. Nem nullázza ki a súlyokat, de „összenyomja” őket	Amikor nem szeretnénk választani a két technika között.
PROS	• Nem fontos változók bétáját 0-ra csökkenti • Jobban értelmezhető modellt hoz létre • Darabszámra, sok, fontosságra kevés változó esetén működik jól	• Minden változót megtart • Csökkenti a multicollinearitás negatívumait a korrelációs hatás „szétosztásával”	• Amikor több a változó, mint a megfigyelés • Szelektál is a változók között, és csökkenti a multicollinearitást is
CONS	Korreláló változók esetén megbízhatatlan lehet, mert „önkényesen” választhat közülük	Magas dimenziószám mellett nehéz lehet a modell értelmezése	• Két hiperparaméter beállítására van szükség • Nehezebb interpretálni, mint mikor csak L1/L2 van

Kiértékelési metrikák

Kiértékelési metrikák

- Folytonos célváltozó →nyilvánvaló mértéke az előre jelzett és a valós érték különbsége → de hogyan tudjuk ezt 1 értékbe tömöríteni?
- **Mean Absolute Error (MAE) – Átlagos abszolút hiba**
  - Az abszolút hiba átlaga (különbség a becsült és a valós értékek között)
  - Kevésbé érzékeny a nagy hibákra
  - Minél alacsonyabb, annál jobb (a pontos érték az adott feladattól függ)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

- **Mean Squared Error (MSE) – Átlagos négyzetes hiba**
  - A négyzetes hiba átlaga (különbség a becsült és a valós értékek között)
  - A hibákat négyzetesen veszi figyelembe így a nagy hibákra érzékenyebbek
  - Minél alacsonyabb, annál jobb (a pontos érték az adott feladattól függ)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- **Root Mean Squared Error (RMSE) – Átlagos négyzetes hiba gyöke**
  - Előnye, hogy értéke a célváltozó tartományával összemérhető
  - Minél alacsonyabb, annál jobb (a pontos érték az adott feladattól függ)

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

- **R^2**
  - Az R^2 azt mutatja meg, hogy az eredmény variabilitásának mekkora részét tudja előre jelezni a modell, megmutatva a modell magyarázó erejét.
  - Egy szám általában 0-1 között → minél magasabb, annál jobb → 1-nél tökéletes modellt jelent
  - Ha negatív, akkor a modell hibája nagyobb mintha az átlagot használnák előrejelzésnek

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

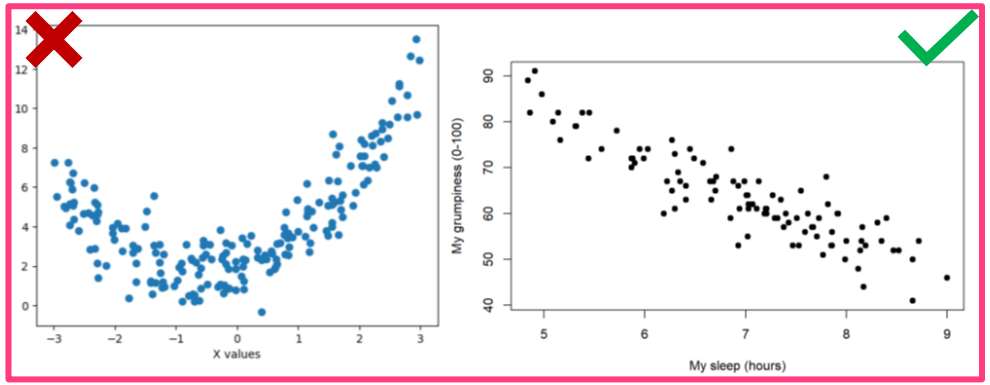
Erről bővebben Windhager Eszter *Machine Learning Algoritmusok Kiértékelése: Mutatók, Mérőszámok, Fontos Értékek* előadásában hallhattok

Python notebook

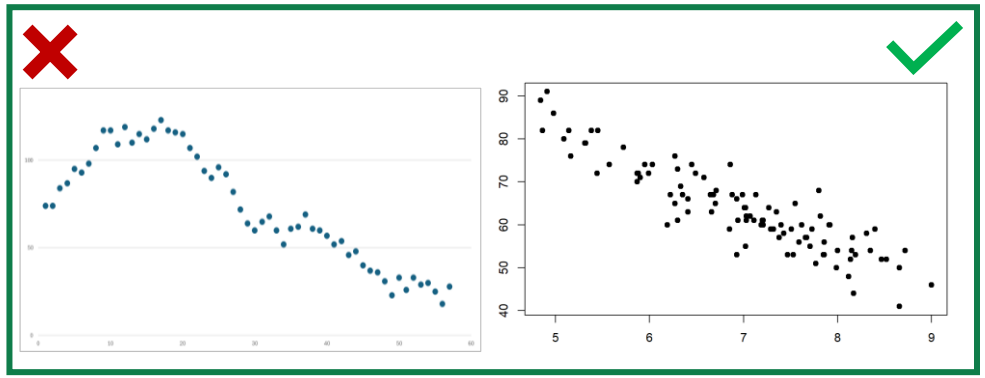
Lineáris Regresszió feltételek és kezelésük

Lineáris Regresszió – feltételek – 1

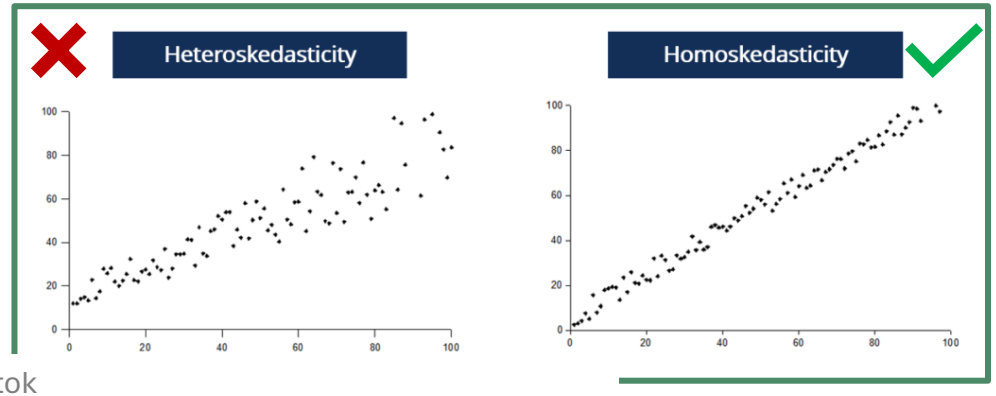
- **Lineáris kapcsolat**
  - A magyarázó változó(k) és a célváltozó között lineáris a kapcsolat



- **Függetlenség**
  - A megfigyelések függetlenek egymástól (pl idősoros adat gond lehet)



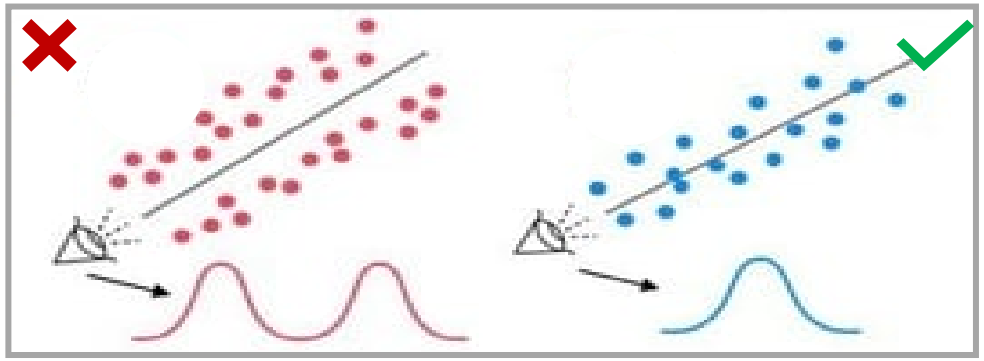
- **Homoszkedaszticitás**
  - A hibák szórása minden megfigyelésnél nagyjából azonos.



Erről bővebben Kozma-Renge Kriszta *Statisztika Data Science-hez - 4) Lineáris Regresszió* előadásában hallhattok

Lineáris Regresszió – feltételek – 2

- **Normális eloszlású hibák**
  - Utólag ellenőrzendő, szükséges a megalapozott statisztikai következtetésekhez.

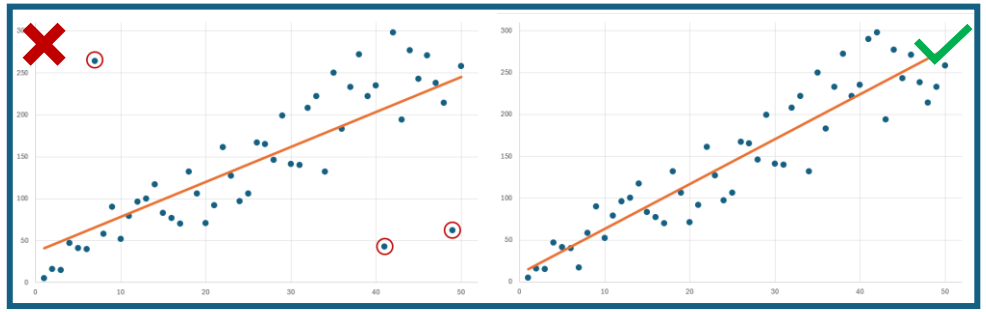


- **Nincs multicollinearitás**
  - A magyarázó változók között nincs (jelentős) korreláció

	F - 1	F - 2	F - 3	F - 4	F - 5	F - 6
F - 1	1	0,39	0,44	0,43	0,43	0,66
F - 2	0,39	1	0,85	0,32	0,32	0,8
F - 3	0,44	0,85	1	0,58	0,61	0,5
F - 4	0,43	0,32	0,58	1	0,57	0,79
F - 5	0,43	0,32	0,61	0,57	1	0,67
F - 6	0,66	0,8	0,5	0,79	0,67	1

	F - 1	F - 2	F - 3	F - 4	F - 5	F - 6
F - 1	1	0,07	0,39	0,25	0,3	0,17
F - 2	0,07	1	0,01	0,39	0,05	0,37
F - 3	0,39	0,01	1	0,08	0,1	0,21
F - 4	0,25	0,39	0,08	1	0,11	0,16
F - 5	0,3	0,05	0,1	0,11	1	0,13
F - 6	0,17	0,37	0,21	0,16	0,13	1

- **Nincsenek kilógó értékek**
  - Az adatban nincsenek jelentős kilógó értékek



Erről bővebben Kozma-Renge Kriszta *Statisztika Data Science-hez - 4) Lineáris Regresszió* előadásában hallhattok

Lineáris kapcsolat

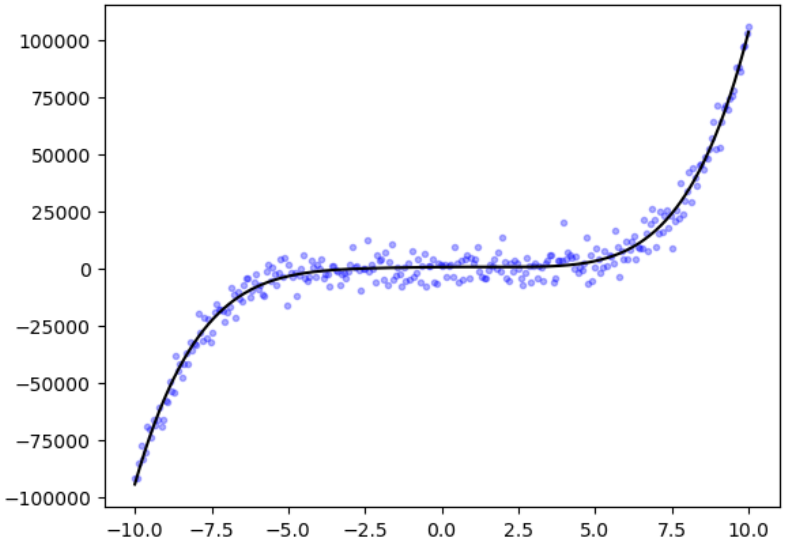
Görbe illesztése – polinomiális regresszió

- **Lineáris kapcsolat:** A magyarázó változó(k) és a célváltozó között lineáris a kapcsolat
- **A lényeg** hogy a koefficiensek legyenek lineárisak, a magyarázó változók azonban lehetnek több-fokúak is

- **Polinomiális regresszió** esetén egy speciális többváltozós regressziót alkalmazunk, ahol új, többfokú (négyzetes/harmadfokú/sokadfokú) változókat képzünk. Ezzel elérhetjük, hogy ne csak egyenest, hanem görbét tudjunk illeszteni, így komplexebb kapcsolatokat is modellezhetünk.

Azonban ez is lineáris regresszió

Hogy van ez?!?!



- **Egyenlet:** Egy magyarázó változó több fokra emelésével egy többváltozós lineáris regressziós egyenletet kapunk.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_1^3 + \beta_m x_1^m + \varepsilon$$

Görbe illesztése – polinomiális regresszió

Használat

- Amikor a célváltozó és a magyarázó változó között nemlineáris kapcsolat van
- Hatékony lokális minták modellezésére, de a magas fokú polinomok instabillá válhatnak a hosszú távú előrejelzések esetében
- Kombinálható a „sima” többváltozós regresszióval, a változó lista összeállhat több független, és azok polinomiális változataiból is
- Tanítása és kiértékelése megegyezik az eddig bemutatott technikákkal, nem igényel speciális módszertant

Pros

- Nemlineáris kapcsolatok modellezése alkalmas
- Rugalmas, sokféle görbét le lehet vele írni
- Lineáris regresszió módszertanát alkalmazza

Cons

- Hajlamos a túltanulásra, főleg magas fokú polinómok használatakor → megfelelő fok kiválasztása fontos
- Érzékeny a kiugró értékekre
- A modell interpretációja nehezebbé válik (mit is jelentenek a változók)

Alkalmazáskor megfontolandó

- Polinóm fokának megválasztása:
  - Magasfokú polinomiális model hajlamos a túltanulásra.
  - A megfelelő komplexitás megválasztásához használjunk keresztvalidációt vagy információs kritérium mérőszámot
- Túltanulás visszaszorítása
  - A túltanulás elkerülés érdekében érdemes lehet reguralizációs technikák alkalmazása
- Magyarázó változók skálázása
  - A polinómok miatt nagyon nagy/kicsi számokat kaphatunk, ezért érdemes normalizációt alkalmazni *(min-max scaling, z-score standardization)*

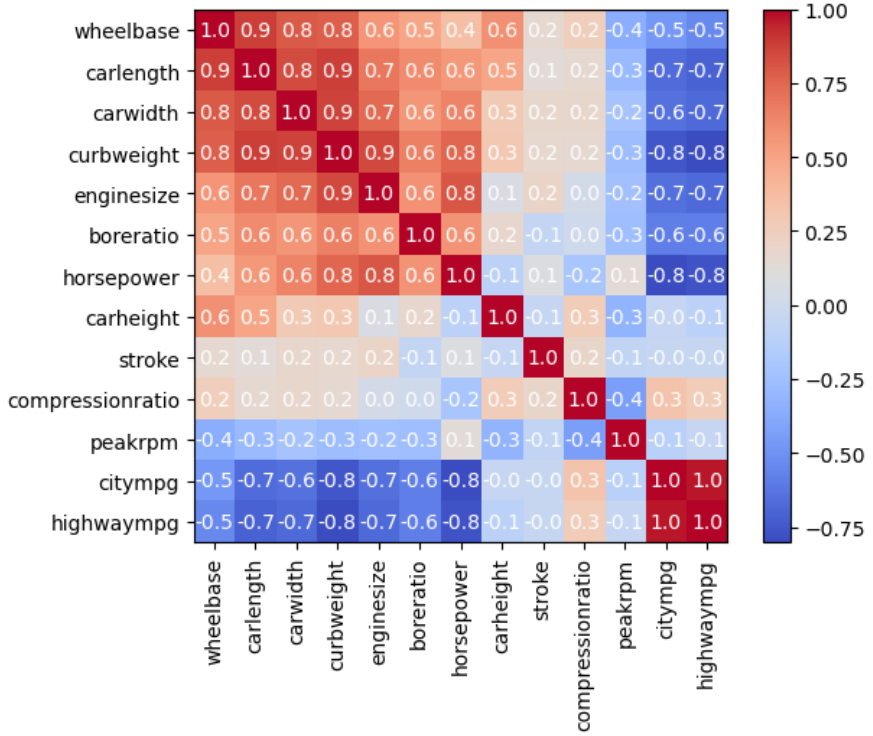
Python notebook

Multicollinearitás

Multicollinearitás

- **Nincs multicollinearitás**
  - A magyarázó változók között nincs (jelentős) korreláció
- **Multicollinearitás azonosítása**
  - Korrelációs mátrix a magyarázó változókra
  - Variance Inflation factor (VIF) kiszámítása
- **Multicollinearitás kezelése**
  - A korreláló változók közül csak az egyiket engedjük be a modellbe
    - Domain tudás alapján
    - Megtartva azt, amelyiknek a célváltozóval magasabb a korrelációja
  - A változók lineáris kombinációját használjuk (pl összeadva őket)
  - Reguralizációs technikákat alkalmazunk
  - PCA-t illesztünk a változókra, és az így kapott főkomponensekre építjük a modellt (torzítja az modell interpretálhatóságát)

A korreláló magyarázó változók torzíthatják a modellt



- **VIF kiszámítása**
  - VIF esetén úgy építünk regressziós modelleket, hogy az épp vizsgált magyarázó változót szeretnénk előre jelezni a többi magyarázó változó segítségével
  - Az így kapott modellekre kiszámoljuk az R^2–t, és a VIF-et
  - VIF érkékek értelmezése:
    - VIF = 1 → nincs multicollinearitás
    - 2 <= VIF <= 5 → alacsony multicollinearitás, ez még belefér
    - 5 < VIF <10 → közepes multicollinearitás, érdemes utánanézni
    - VIF > 10 → súlyos multicollinearitás, kezelni kell!

$$VIF_j = \frac{1}{1 - R_j^2}$$

Kiugró értékek kezelése

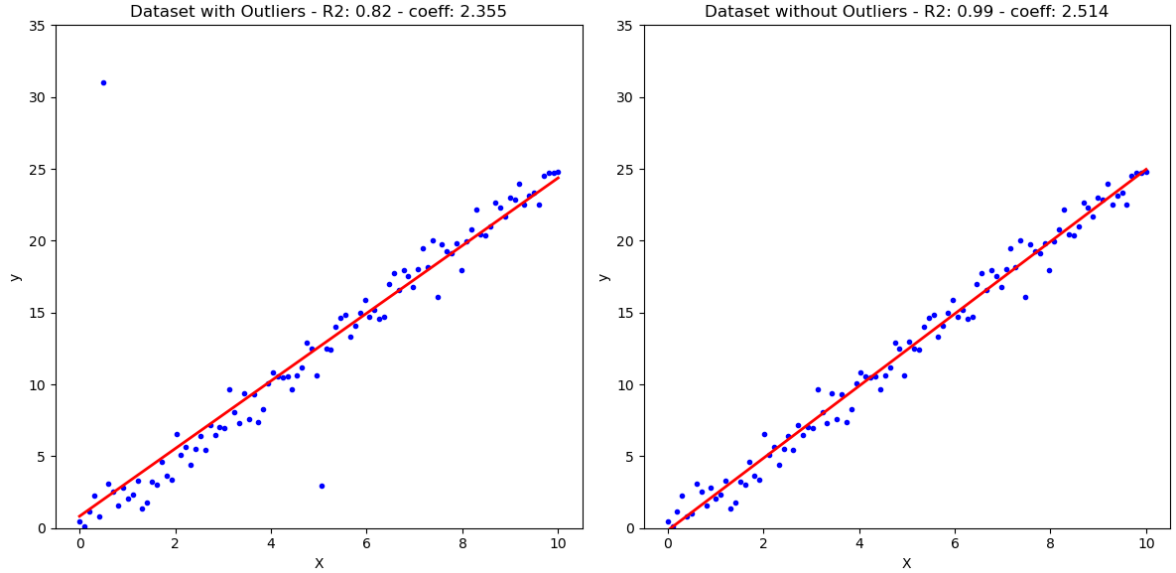
Kiugró értékek kezelése

- **Nincsenek kilógó értékek**
  - Az adatban nincsenek jelentős kilógó értékek
  - Kilógó értékek előfordulhatnak mind a célváltozóban, mind a magyarázó változókban

- **Kiugró értékek azonosítása**
  - Domain tudás
  - Vizualizáció (scatterplot, boxplot)
  - Reziduálisok vizsgálata

Mert ez történhet:

A kiugró értékek torzíthatják az illesztést



- **Kiugró értékek kezelése**
  - Adatpont eldobása/javítása → csak akkor, ha tudjuk, hogy hibás (domain tudás!)
  - Kiugró értékek levágása → a kiugró értéket levágjuk egy meghatározott számra (pl. percentilis alapján), így csökkentjük a hatását, de mégis megőrizzük az információt
  - Adattranszformáció – logaritmus vagy négyzetgyök transzformáció alkalmazása