Regression i maskininlärning

Vad är Regression?

- Regression = förutsäga ett tal
- Target y är kontinuerlig
- Exempel:
  - Pris
  - Tid
  - Temperatur
  - Försäljning nästa vecka

Data

- X = features (input)
- y = target (output)

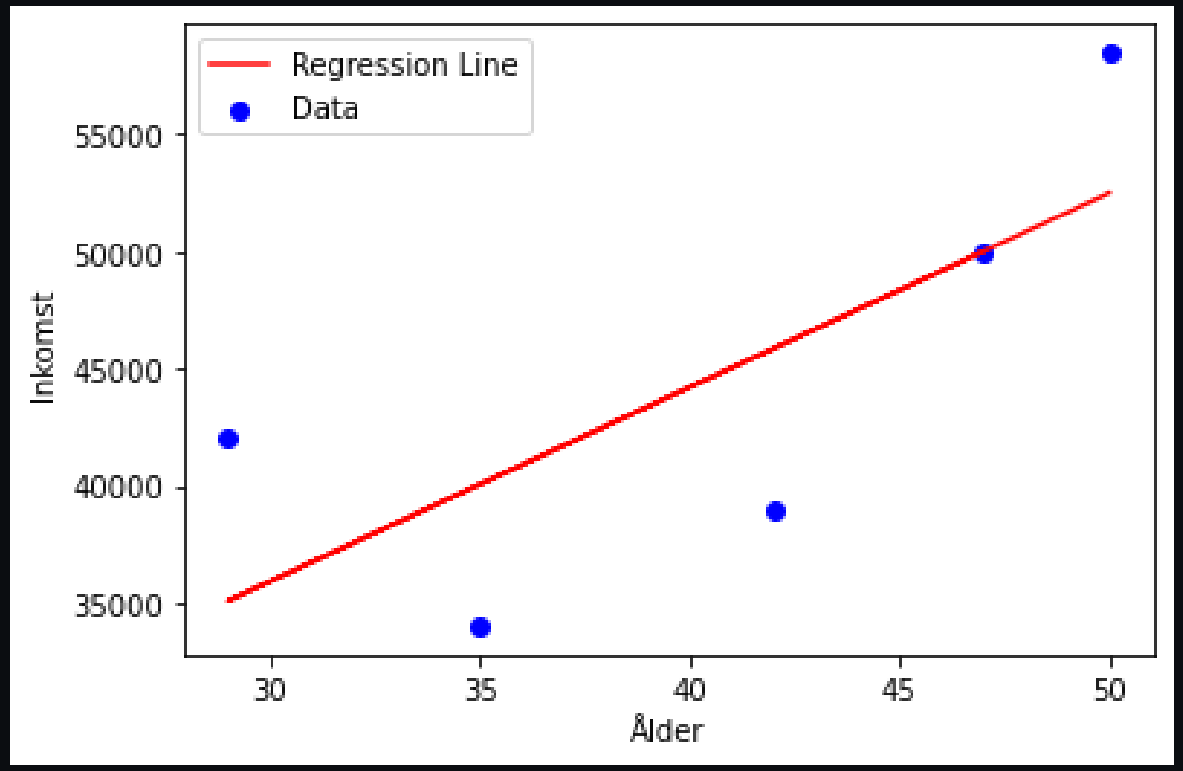
X			y
Area	Rooms	Age	Price

Regressionsproblem

- Regression = problem
- Linjär regression = modell

Linjär regression

Enkel och tydlig referensmodell

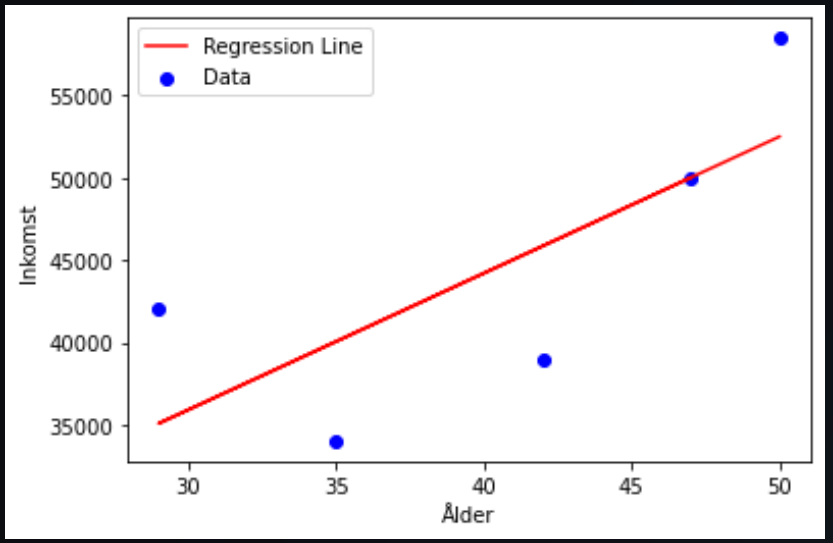


Enkel linjär regression

- Modellen försöker hitta en linje/ett plan som passar data

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

Inkomst (y)	Ålder (x)
58500	58
42000	29
34000	35
39000	42



Multipel linjär regression

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon$$

- Exempel: area, rooms, age

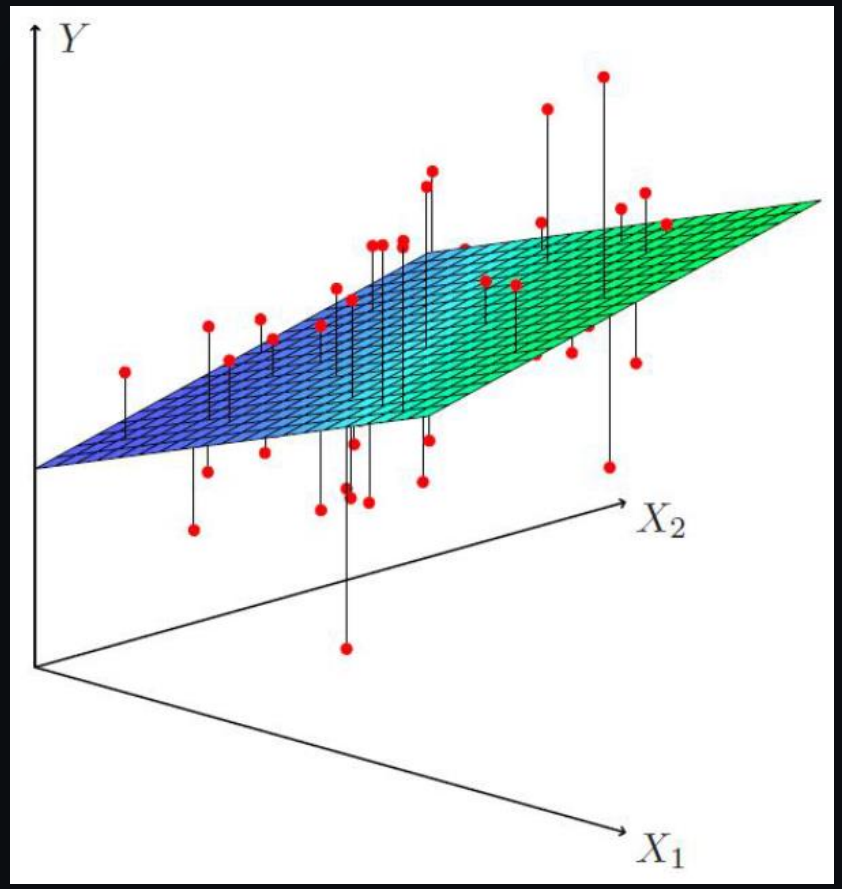
Ekvationssystem

$$\begin{bmatrix} \mathbf{y_1} \\ \mathbf{y_2} \\ \mathbf{y_3} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{x_{11}} & \mathbf{x_{12}} \\ \mathbf{1} & \mathbf{x_{21}} & \mathbf{x_{22}} \\ \mathbf{1} & \mathbf{x_{31}} & \mathbf{x_{32}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta_0} \\ \boldsymbol{\beta_1} \\ \boldsymbol{\beta_2} \end{bmatrix} + \begin{bmatrix} \boldsymbol{e_1} \\ \boldsymbol{e_2} \\ \boldsymbol{e_3} \end{bmatrix}$$

$$\vec{\boldsymbol{y}} = X\vec{\boldsymbol{\beta}} + \mathbf{e}$$

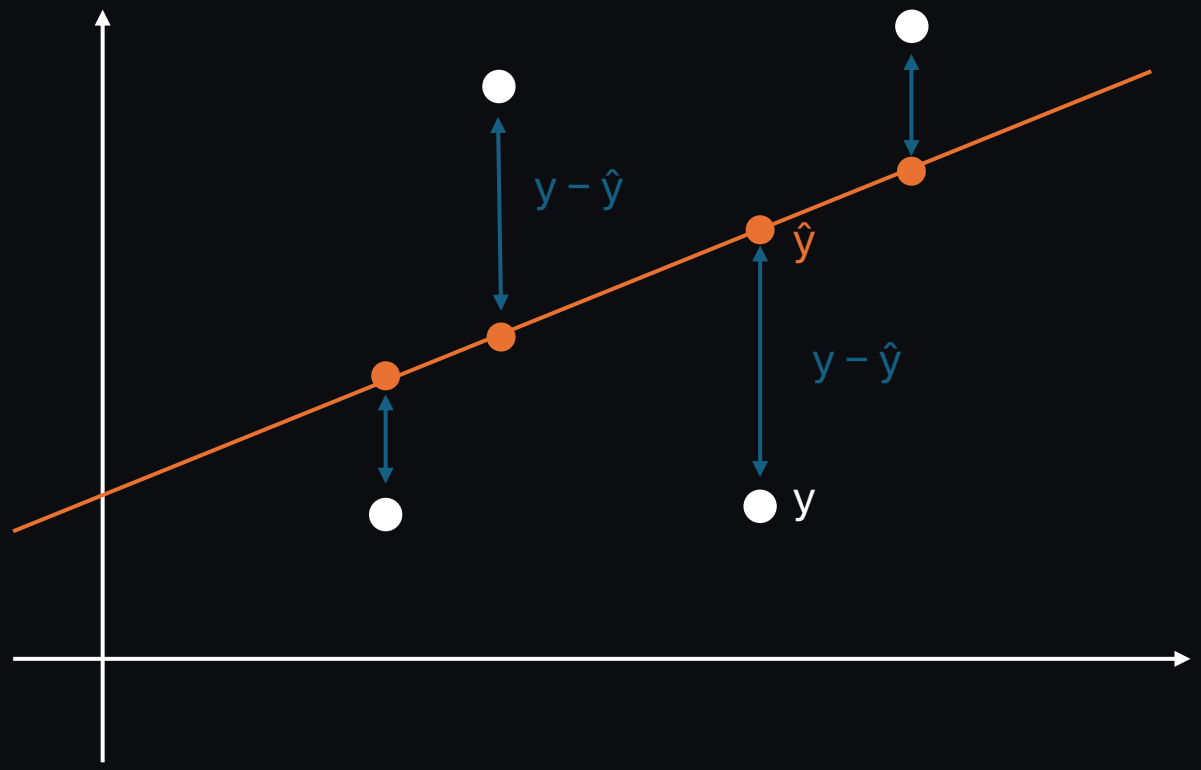
- y = värden vi vill förklara
- X = datamatris (features)
- $\boldsymbol{\beta}$ = koefficienter
- e = fel / residualer

Multipel linjär regression



- Med två variabler så har man ett plan istället för en linje.
- Har man fler variabler så kan vi inte visualisera det.

Residualer



- Sant värde: y
- Prediktion: ŷ
- Fel / residual: y – ŷ

Hur ”lär” sig modellen?

- Vi väljer beta så att felet blir så litet som möjligt
- Det blir ett optimeringsproblem

Parameter vs hyperparameter

- Parameter = lärs från data (ex: β-koefficienter)
- Hyperparameter = väljs av oss innan/under träning (ex: regulariseringsstyrka, träd-djup, learning rate)
- Påverkar hur modellen generaliserar

Bias-Variance Trade-off

- Underfitting: för enkel modell → hög bias
- Overfitting: för flexibel modell → hög variance
- Målet: bra på ny data
- Metrics + validation hjälper oss upptäcka detta

Regularisering

Ridge / Lasso / Elastic Net

- Idé: lägg till ett “straff” → mindre koefficienter → mindre överanpassning
- Ridge: straffar stora koefficienter (tenderar att göra dem små)
- Lasso: kan trycka vissa koefficienter till exakt 0 (feature selection)
- Elastic Net: blandning
- Hyperparameter: styr hur starkt straffet är

Ridge Regression (L2)

- Ridge Regression lägger till ett extra straff i kostnadsfunktionen som vi försöker minimera

$$J(\beta) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}_{\text{Data-fit (MSE)}} + \underbrace{\alpha \sum_{j=1}^{p}\beta_j^2}_{\text{Straffterm (L2)}}$$

- Våra beta kommer dras mot 0 såvida MSE inte minskar mer än vad extra straffet ökar.

- α är en hyperparameter och kan väljas med hjälp av GridSearchCV i Scikit-learn.

Lasso Regression (L1)

$$J(\beta) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}_{\text{Data-fit (MSE)}} + \underbrace{\alpha\sum_{j=1}^{p}|\beta_j|}_{\text{Straffterm (L1)}}$$

Lasso kan sätta vissa ”oviktiga” koefficienter exakt till 0 vilket gör den populär då man automatiskt får fram vilka variabler som ska användas i en regressions modell

Elastic Net (L1 + L2)

Elastic Net är en blandning av Ridge och Lasso,
bra när features är korrelerade och du vill ha lite ur båda världar

$$J(\beta) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}_{\text{MSE}} + \alpha\left(\underbrace{(1-p)\sum_{j=1}^{p}\beta_j^2}_{\text{Ridge}} + \underbrace{p\sum_{j=1}^{p}|\beta_j|}_{\text{Lasso}}\right)$$

- α = hur mycket straff totalt
- p (ibland kallad l1_ratio) = mix mellan L1 och L2

Effekten av regularisering (α) i praktiken

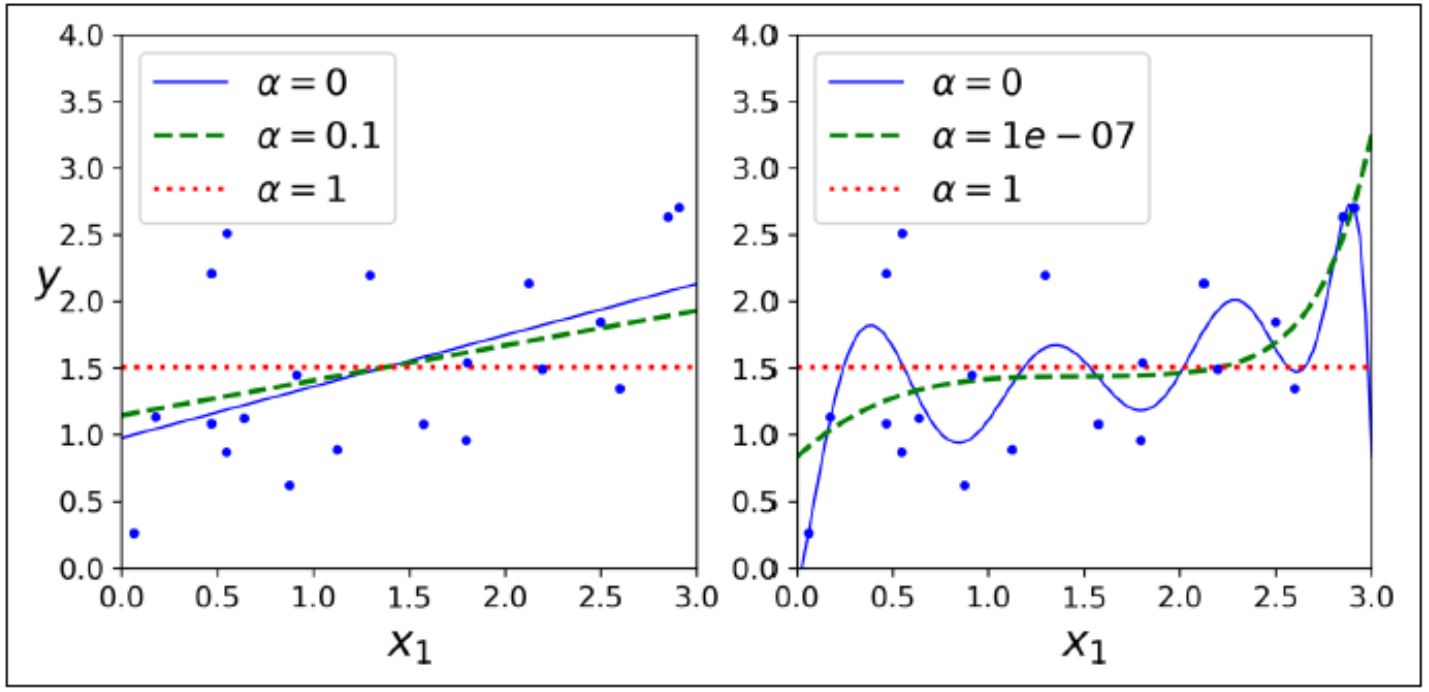


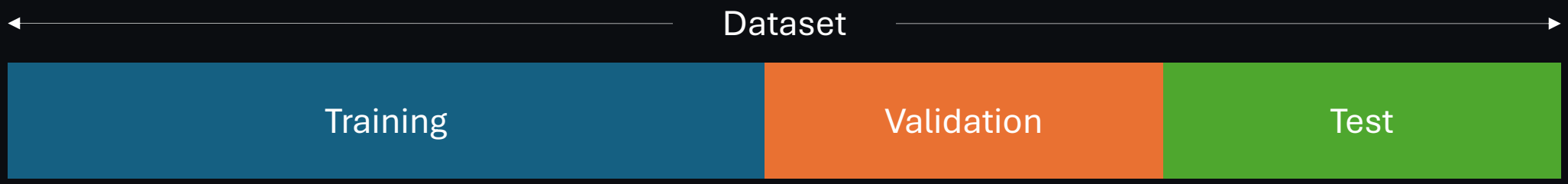
Figure 4-18. A linear model (left) and a polynomial model (right), both using various levels of Lasso regularization

Bild tagen från (s.138) i andra upplagan av Hands on machine Learning with Scikit-Learn Keras & Tensorflow, Aurélien Géron.

Varför metrics?

- Vi kan inte bara ”känna” att modellen är bra
- Jämföra modeller objektivt
- Upptäcka överanpassning
- Mål att optimera mot

Training / Validation / Test



Test används **en** gång på slutet

Baseline

- För regression: gissa medelvärdet eller median
- Om ML-modellen inte slår baseline → något är fel

- Sant y: [100, 120, 80, 110]
- Baseline gissar: 102.5 på allt

MAE: Mean Absolute Error

medelvärde av $|y - \hat{y}|$

- Genomsnittligt absolut fel
- Samma enhet som y

- y: [10, 8, 15]
- ŷ: [12, 7, 14]
- |fel|: [2, 1, 1] → MAE = 1.33

MSE & RMSE

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

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

- Kvadraten straffar stora fel mer
- RMSE tillbaka i samma enhet som y

MAE vs RMSE

MAE	RMSE
Lätt att tolka	Straffar stora fel mer
Mindre känslig för outliers	Bra när stora missar är extra känsliga

I kursen använder vi ofta RMSE som standard

R^2 – Förklaringsgrad

- R^2 ≈ hur mycket bättre än baseline (medelvärde)
- 1.0 = perfekt
- 0.0 = samma som baseline
- Kan vara negativ

Vanliga fallgropar

- Utvärdera på träningsdata → falsk trygghet
- Ingen baseline → du vet inte om modellen är “bra”
- Jämföra modeller med olika datasplit → orättvist
- RMSE vs MAE: välj metric som matchar problemet
- R^2 kan lura — komplettera alltid med RMSE/MAE

Regression i maskininlärning

Joakim Lindh