# 732G12 Data Mining

Föreläsning 6

Josef Wilzén
IDA, Linköping University, Sweden

# Dagens föreläsning

- Bayesianska klassificerare
- Ensamblemetoder
  - Bagging
  - Boosting
  - Random forest

# Bayesiansk klassificerare

Att direkt modellera en icke-deterministisk funktion kan vara mycket svårt.

Exempel:

- $(\text{diet}, \text{träning}) \to (\text{hjärtinfarkt})$ är svårt
- $(\text{diet}, \text{träning}) \to \mathbb{P}(\text{hjärtinfarkt})$ lättare

Använd Bayes sats för att hjälpa till i modelleringen

$$\mathbb{P}(Y \mid \mathbf{X}) = \frac{\mathbb{P}(\mathbf{X} \mid Y)}{\mathbb{P}(\mathbf{X})} \cdot \mathbb{P}(Y) \propto \mathbb{P}(\mathbf{X} \mid Y) \cdot \mathbb{P}(Y)$$

$$\text{posterior} = \frac{\text{likelihood}}{\text{evidence}} \cdot \text{prior} \propto \text{likelihood} \cdot \text{prior}$$

# Kategoriska attribut

$\mathbb{P}(Y = y)$ är andelen datapunkter med klass $y$.
$\mathbb{P}(X_i = x_i \mid Y = y)$ andelen datapunkter med attribut $x_i$ av datapunkterna med klass $y$.

| | binary | categorical | continuous | class |
|---|---|---|---|---|
| Tid | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |

# Kontinuerliga attribut

För kontinuerliga attribut finns olika tillvägagångssätt.

- Diskretisera data i olika kategorier.
  - För få intervall gör att man missar information.
  - För många intervall kan ge intervall utan observationer.
- Anta en sannolikhetsfördelning för variabeln och skatta parametrarna från träniningsdatan.
  - Normalfördelningen är vanlig.
  - Conjugate prior.

# Grundläggande princip

Träningfasen:

Skatta sannolikheten $\mathbb{P}(Y|\mathbf{X})$ för alla möjliga $\mathbf{X}$ och $y$.

Klassificeringsfas:

Givet $\mathbf{X}'$ skatta klass $Y' = \max_Y \mathbf{P}(Y|\mathbf{X}')$.

# Naiv Bayes klassificerare

Modellantagande:

$$\mathbb{P}(\mathbf{X} \mid Y) = \prod_i \mathbb{P}(X_i \mid Y),$$

alla $X_i$ är oberoende av varandra. Vi kan då faktorisera likelihooden över $\mathbf{X}$.

Använder vi detta får vi en sannolikhet

$$\mathbb{P}(Y \mid \mathbf{X}) = \prod_i \mathbb{P}(X_i \mid Y)\mathbb{P}(Y),$$

det räcker med att skatta sannolikheten för varje $X_i$. Detta ger oss en enklare modell som går att skatta.

# Egenskaper hos naiv Bayes

- Metoden är robust mot isolerade bruspunkter.
- Metoden är robust mot irrelevanta attribut.
- Lätt att skatta.
- Korrelerade attribut kan väsentligt försämra prestandan.
  - Behöver en mer komplex modell för att hantera.
  - Simultan sannolikhetsfördelning för likelihooden.

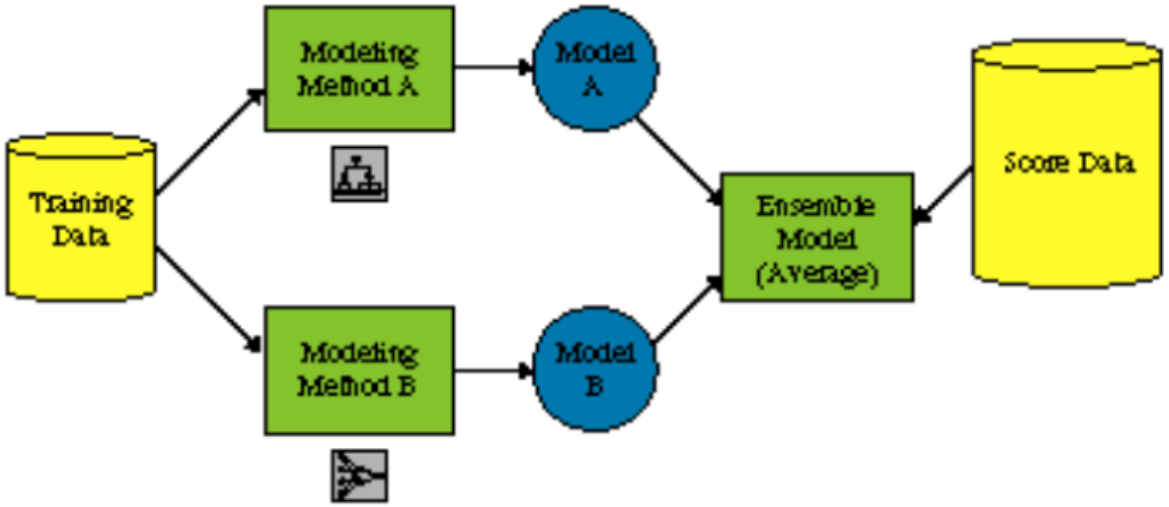
# Ensamblemetoder

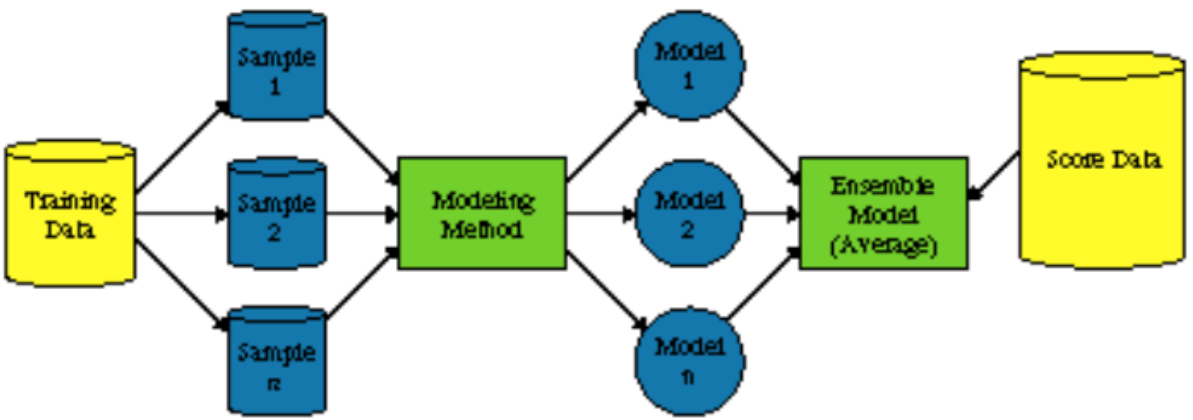
Grundidén är att skatta många modeller på träningsdata och sen kombinera dessa för att göra prediktioner.

Finns många olika varainter:

- Göra slumpmässiga urval från träningsdata och skatta modeller på dessa urval (Bootstraping/Bagging)
- Skapa en mängd dataset där observationerna har olika vikter i modellanpassningen i olika dataset (Boosting)
- Skatta ett antal olika modeller (kan vara av helt olika sort) och sen kombinera dessa vid prediktioner
- Vissa modeller har slumpmässiga element vid optimieringen (tänk neurala nätverk): optimera samma modell flera gånger men med olika seed, vilket ger olika modeller/parameterar. Kombinera dessa sedan vid prediktionen.

# Ensamblemetoder



# Bootstraping

Idé: Skapa $B$ stickprov av datan genom att **med återläggning** välja nya datapunkter. Använd dessa stickprov för att skatta modell eller funktioner.

Exempel:

Vi vill skatta $\mathbb{V}(e^{\bar{X}})$.

Skapa $B$ stickprov.

Skatta $T_k = e^{\bar{Z}_k}$ för $k = 1, \ldots, B$.

Beräkna $\mathbb{V}(\mathbf{T})$.

# Bagging

Idé: Om man tar medelvärdet av oberoende observationer (modeller) så minskar variansen.

**Bagging (Bootstrap aggregating)**: Använd Bootstrap för att skapa $B$ träningsdataset och skatta en modell $\hat{f}_b$ för varje av dessa set.
Den slutgiltliga modellen får vi genom att ta medelvärdet av alla dessa modeller:

$$\hat{f}_{\text{bag}}(\mathbf{X}) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}_b(\mathbf{X}).$$

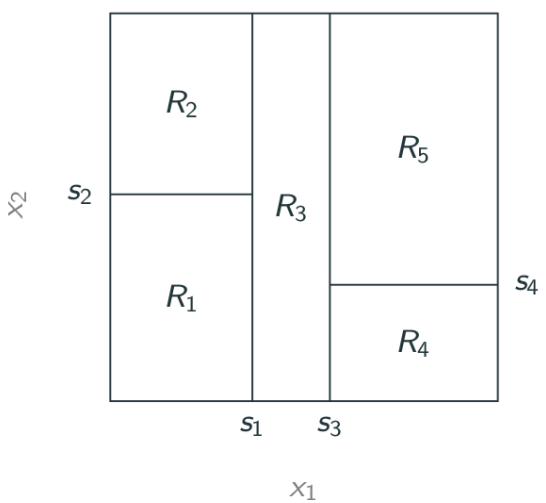
- Sänker variansen av den anpassade funktionen.
- Påverkas *mycket* av kvalitén av modellen. En bra modell blir bättre, en dålig blir sämre.
- För klassificering, använd majoritetsröstning.

# Classification and Regression Trees (CART)

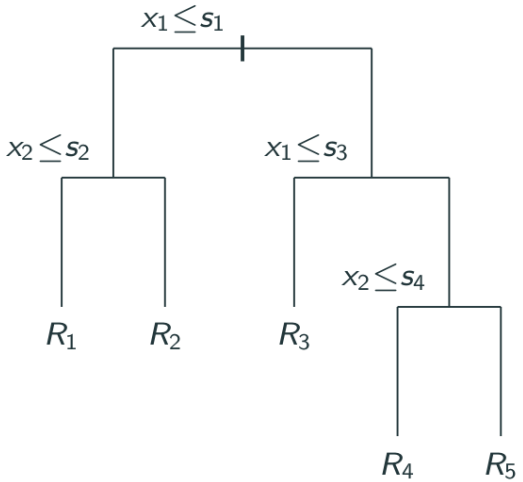
Vi delar upp variabelrummet genom att rekrusivt göra binära splittar.

För klassificering används vanligaste klassen, för regression medelvärdet inom regionen.

Partitioning of input space




Tree representation



# Förbättre CART

Flexibilitet/komplexitet för trädmodeller beror på djupet.

Ett djupt träd ger litet bias, men mycket varians.

Förbättringar:

- Efterbeskärning (post-pruning):
  - Skapa ett djupt träd och beskär det till ett mindre (minska variansen).
- Ensamblemetoder:
  - Ta ett genomsnitt över många trädmodeller.
  - Bagging
  - Random forest
  - Boosted trees

# Random forest

Bagging kan ge stora förbättringar för trädmodeller. Men det finns vissa problem:

- De $B$ bootstrap-urvalen är korrelerade.
- Reduktionen i varians blir liten när vi tar medelvärde över korrelerade variabler.

Idé: Avkorrelera de $B$ trädmodellerna genom att göra slumpmässiga ändringar av modellerna.

# Random forest

- Använd bagging för att skatta $B$ träd,
  - Vid varje uppdelning/regel använd endast en slumpmässig delmängd $q \leq p$ av de förklarande variablerna.
- Tumregel: (Förslag från Leo Breiman)
  - Klassificering: $q = \sqrt{p}$
  - Regression: $q = p/3$

# Random forest

Slumpmässigt val av variabler leder till:

- \- Minskar bias, men ofta mycket långsamt.
- \- Lägger till varians till varje träd.
- \+ Avkorrelerar träden.

Ofta dominerar den avkorrelerade effekten vilket leder till att MSE minskar på testdata.

# Random forest

Beräkningsmässiga fördelar:

- Lätt att parallellisera.
- $q < p$ minskar kostanden för vaje regel/uppdelning.
- Inte så många hyperparametrar.

# Boosting

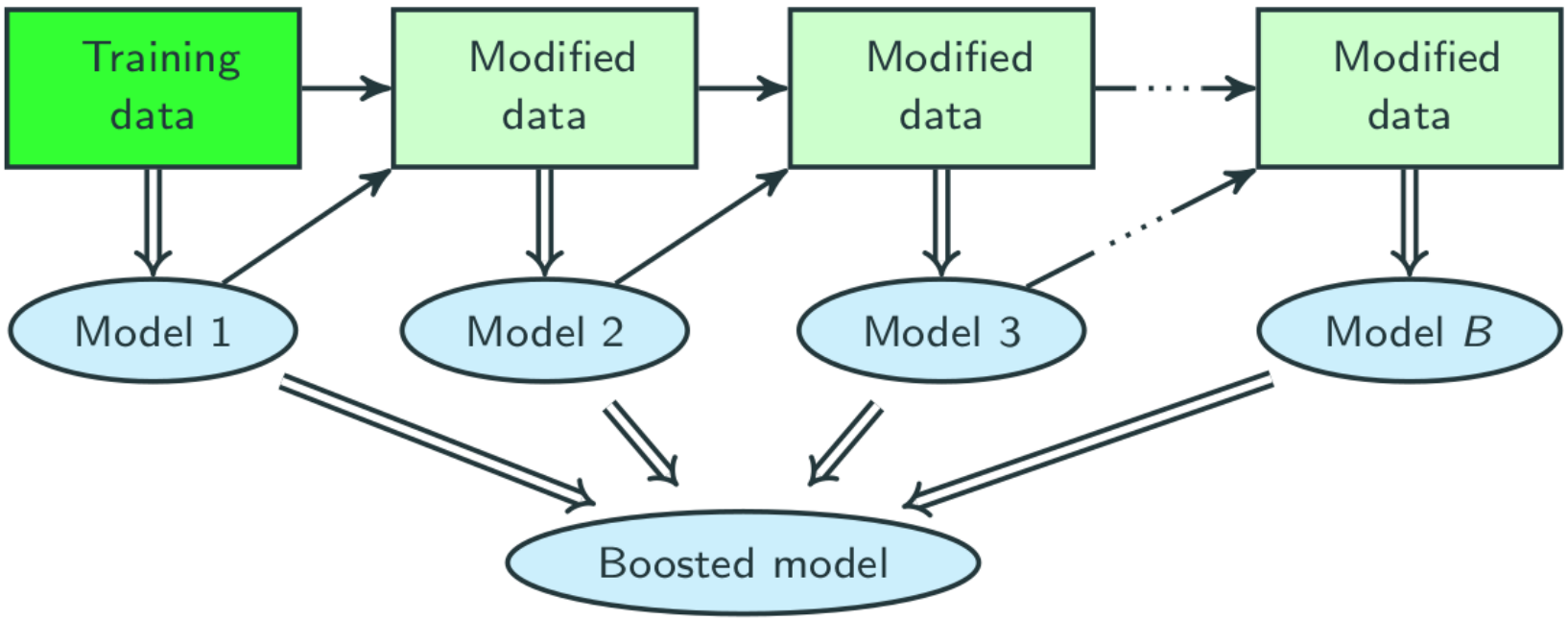
En enkel modell kan vanligtvis fånga vissa aspekter av datan.

Kan vi lära oss en stor mängd enkla modeller som var och en lär sig en liten del av datarelationen och sen kombinera dessa ”dåliga” modeller till en stark modell.

Hur skulle vi göra detta?

# Boosting

- Lär sig sekventiellt en ensamble av "svaga" modeller.
- Kombinerar dessa till en "stark" modell.
- Generell metod som kan användas till all form av övervakad inlärning.
- Mycket framgångsrikt inom maskininlärning.



# Binär klassificering

Vi kommer begränsa oss nu till binär klassificering.

Vi låter klasserna vara $-1$ och $1$ (möjliga $y$ värden).

Använder vi detta kan vi skriva majoritetsröstninig av $B$ klassificerare $\hat{y}^b(\mathbf{x})$ som

$$\text{sign}\left(\sum_{b=1}^{B} \hat{y}^b(\mathbf{x})\right).$$

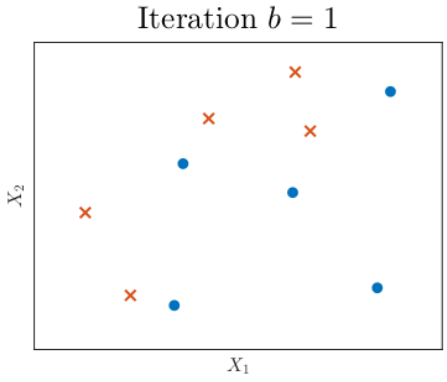
# Boosting för klassificering

1. Ge varje datapunkt en vikt $w_i^1 = 1/n$.
2. För $b = 1, \ldots, B$
   a Träna en "svag" klassificerare $\hat{y}^b(\mathbf{x})$ på den **viktade träniningsdatan** $\{(\mathbf{x}_i, y_i, w_i^b)\}_{i=1}^n$.
   b Uppdatera vikterna $\{w_i^{b+1}\}_{i=1}^n$ från $\{w_i^b\}_{i=1}^n$
      i Öka vikterna för missklassificerade datapunkter.
      ii Minska vikterna för korrekt klassificeradet datapunkter.

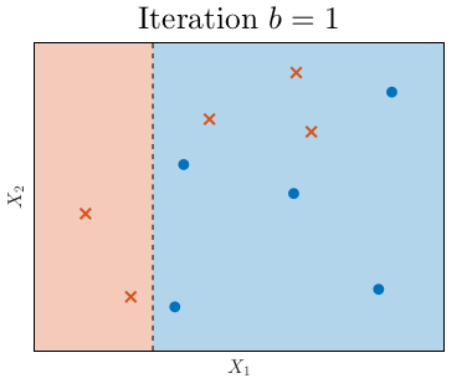
Predikationen från de $B$ klassificerarna kombineras genom att använda en viktad majoritetsomröstning,

$$\hat{y}_{\text{boost}}^B(\mathbf{x}) = \text{sign}\left(\sum_{b=1}^{B} \alpha^b \hat{y}^b(\mathbf{x})\right).$$

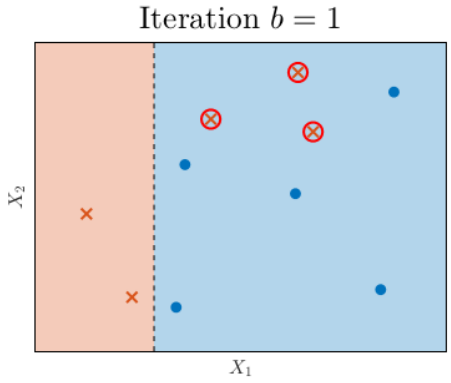
# Boosting exempel



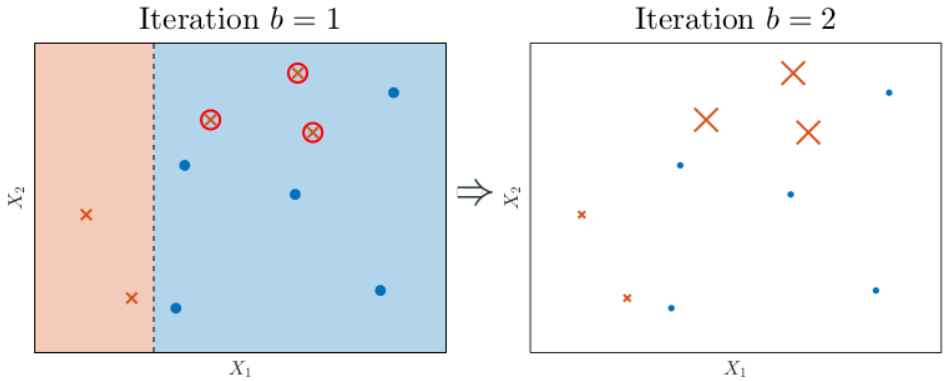
# Boosting exempel



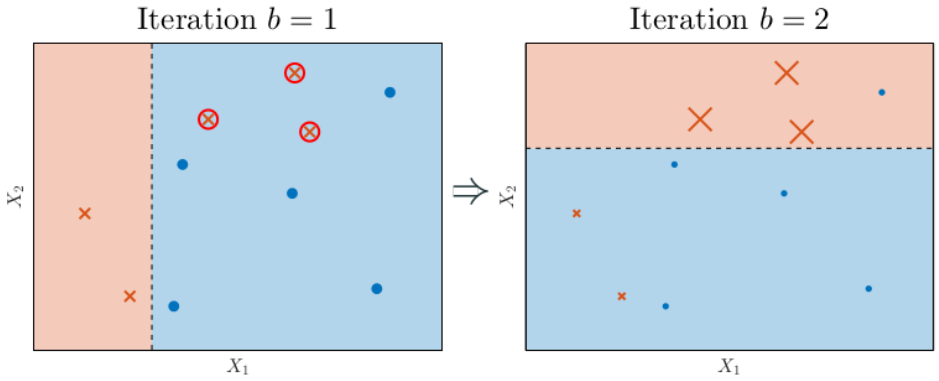
# Boosting exempel



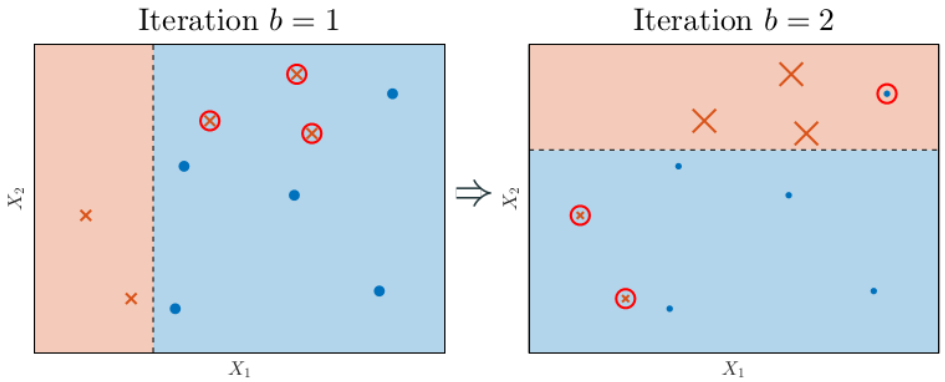
# Boosting exempel



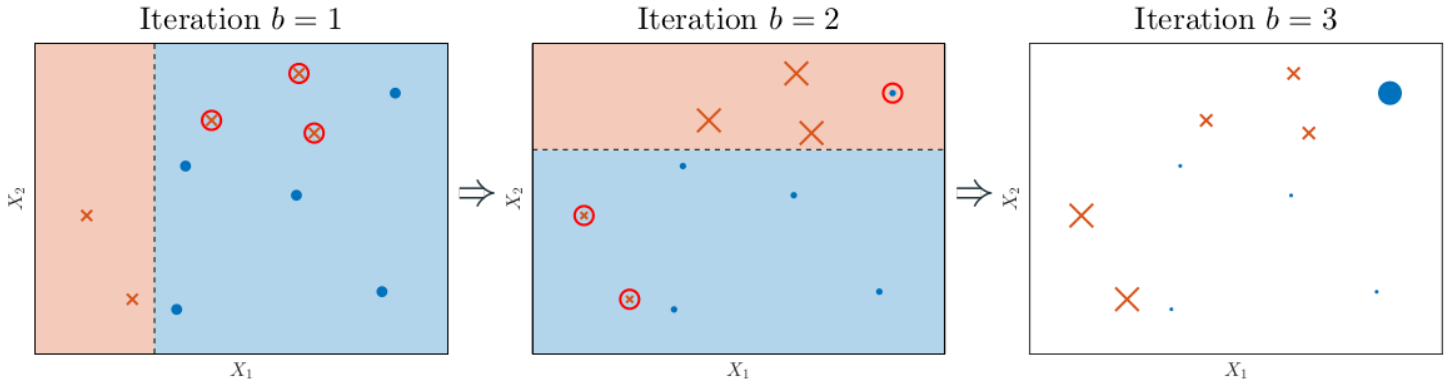
# Boosting exempel



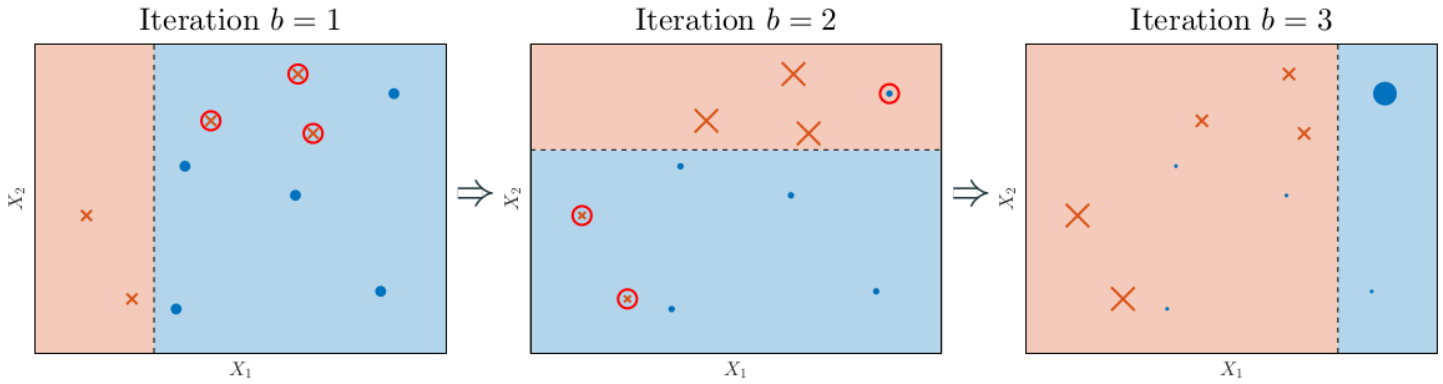
# Boosting exempel



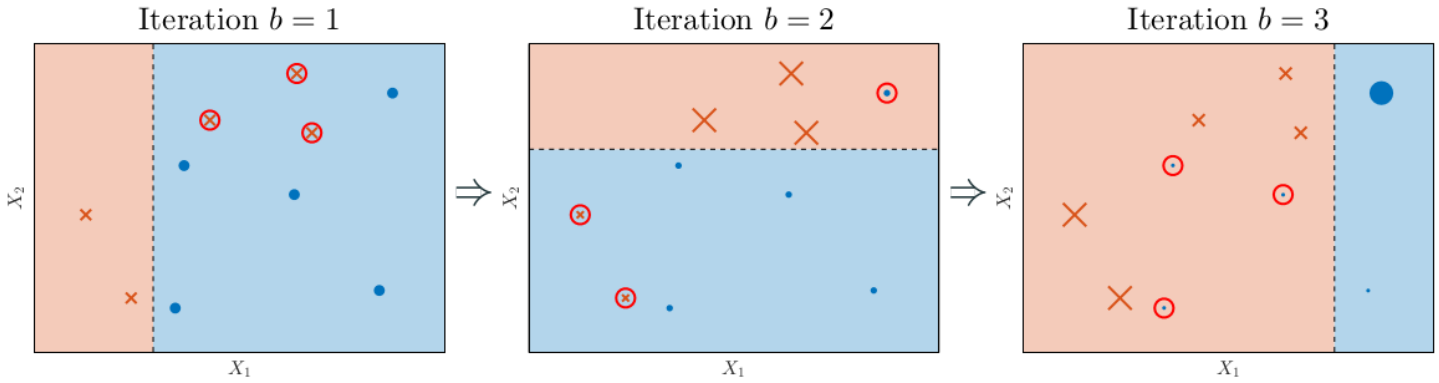
# Boosting exempel



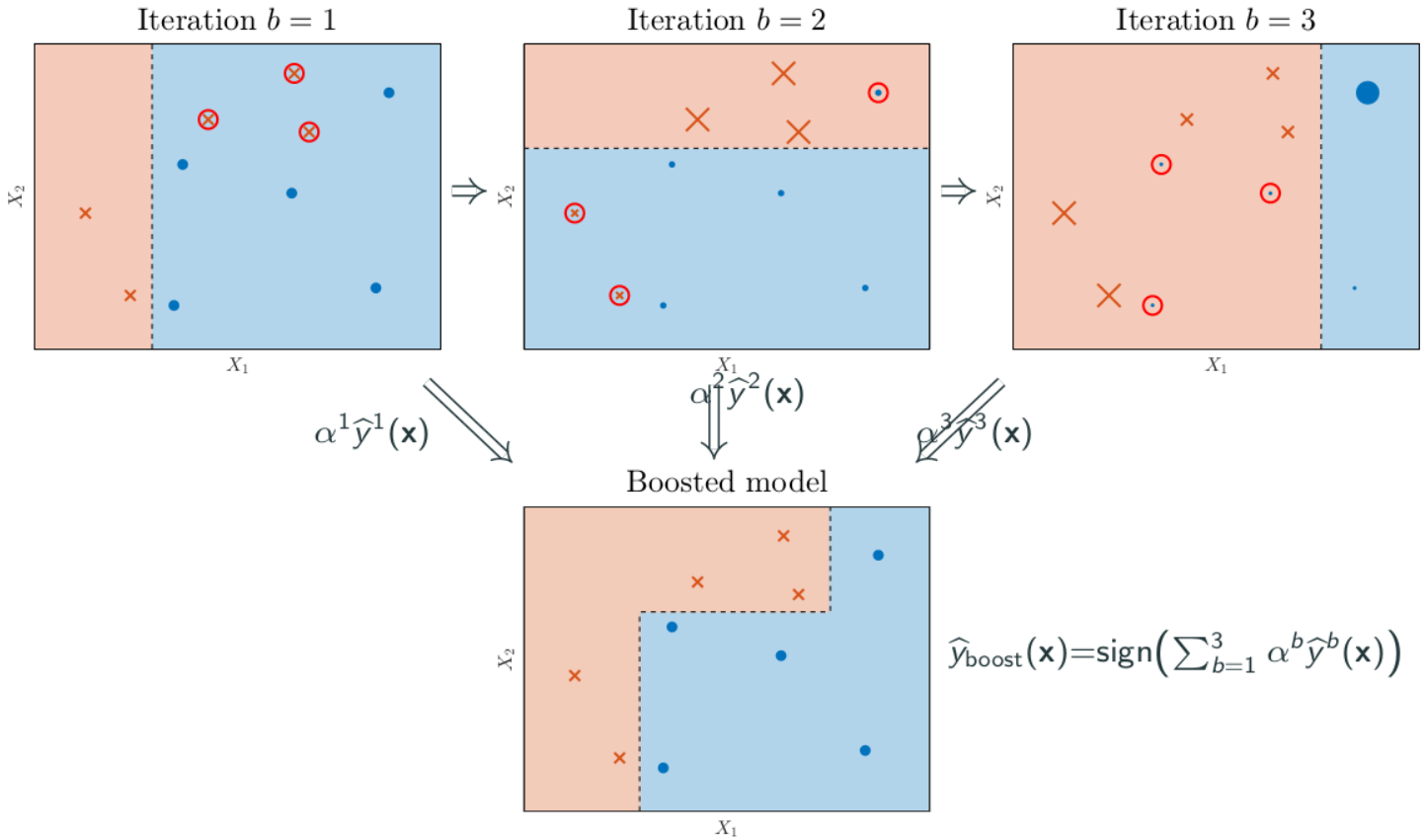
# Boosting exempel



# Boosting exempel



# Boosting exempel



# Boosting - Detaljer

Boosting fungerar bra, men vi har lite detaljer vi måste reda ut först.

1. Hur ska vi vikta om data?
2. Hur ska vi vikta koefficienterna $\alpha^b$?

Olika boostingalgoritmer svarar olika på dessa frågor.

Den första praktiska algoritmen AdaBoost, svarade på dessa frågor genom att minimera exponentialförlust.

# AdaBoost

1. Ge varje datapunkt en vikt $w_i^1 = 1/n$.
2. För $b = 1, \ldots, B$
   a Träna en "svag" klassificerare $\hat{y}^b(\mathbf{x})$ på den **viktade träninings**data**n** $\{(\mathbf{x}_i, y_i, w_i^b)\}_{i=1}^n$.
   b Uppdatera vikterna $\{w_i^{b+1}\}_{i=1}^n$ från $\{w_i^b\}_{i=1}^n$
      i Beräkna $E_{\text{train}}^b = \sum_{i=1}^n w_i^b \mathbb{I}\{y_i \neq \hat{y}^b(\mathbf{x}_i)\}$.
      ii Beräkna $\alpha^b = 0.5 \log((1 - E_{\text{train}}^b)/E_{\text{train}}^b)$.
      iii Beräkna $w_i^{b+1} = w_i^b \exp(-\alpha^b y_i \hat{y}^b(\mathbf{x}_i)), i = 1, \ldots, n$.
      iv Normalisera $w_i^{b+1}$.
3. Output $\hat{y}_{\text{boost}}^B(\mathbf{X}) = \text{sign}\left(\sum_{b=1}^B \alpha^b \hat{y}^b(\mathbf{x})\right)$.

# Boosting för regressionsträd

**Algorithm 8.2** *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all $i$ in the training set.

2. For $b = 1, 2, \ldots, B$, repeat:

   (a) Fit a tree $\hat{f}^b$ with $d$ splits ($d+1$ terminal nodes) to the training data $(X, r)$.

   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$

   (c) Update the residuals,

   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

# Boosting - Sammanfattninig

Finns många andra varianter:

- Gradient boosting:
  - XGboost
  - LightGBM
  - CatBoost
- Presterar bra och vinner ofta tävlingar.

Om vi jämför med baggning kan vi se:

| Bagging | Boosting |
|---|---|
| Kan träna modeller parallellt | Tränar modeller sekventiellt |
| Använder bootstrappade dataset | Använder viktade dataset |
| Överanpassar inte med ökande $B$ | Kan överanpassa när $B$ ökar |
| Minskar variansen men inte bias | Minska varians och bias. |