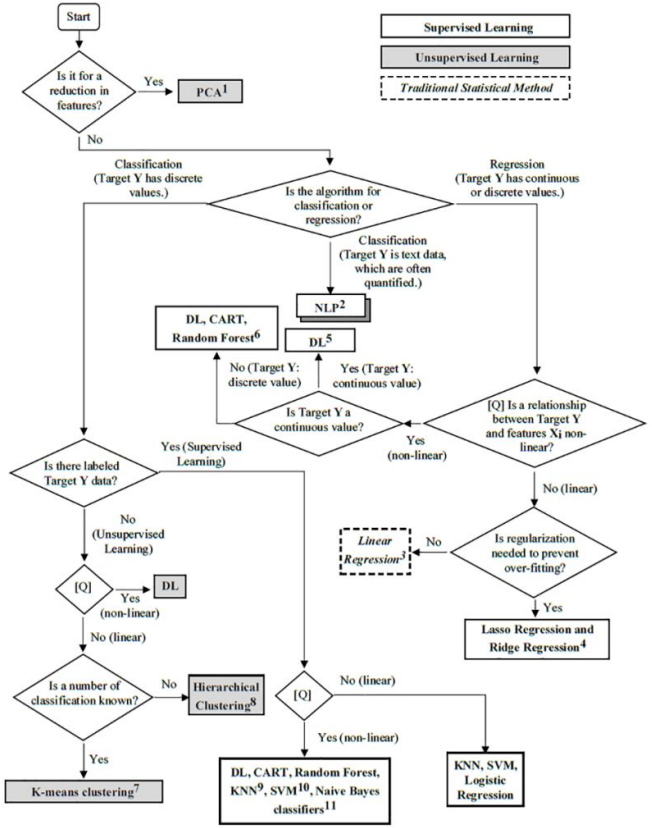
# Stabla odlučivanja

Decision trees, bagging i boosting

# Šta su stabla odlučivanja?

- OSI - struktura slična dijagramu toka
- Gradi se rekurzivno
- Leaf nodes - konačne odluke
- Ostali čvorovi - uslovi grananja
- Može i klasifikacija i regresija
- Osnovna ideja i dalje ista - minimizacija greške



# Zar ne možemo mi napraviti naše stablo?

- Možemo!
- Često za male probleme za koje poznajemo domen i bolji pristup
- Zapravo ništa više nego if-else
- Ipak to nije mašinsko učenje
- Realni slučaj - veliki skupovi podataka gdje veze nisu lako uočljive
- Dodatni problem - promjenljivost domena (npr navike kupaca)
- U praksi često datasetovi sa stotinama kolona i milionima redova - kako uočiti uslove grananja?
- Mašinsko učenje - iterativno gradimo stablo minimizacijom greške

# Algoritam za kreiranje stabla odlučivanja

- Osnovna ideja - biramo kolonu i vrijednost na osnovu kog ćemo dijeliti naš dataset u dva podskupa
- Bitno definisati grešku kako bismo znali da li je izabrana kolona i vrijednost dobra
- Posmatramo prvo problem binarne klasifikacije
- Prilikom svake podjele imamo određeni broj redova koji pripada jednoj i koji pripada drugoj klasi
- Šta želimo da minimizujemo?

| A | B | Class |
|---|---|---|
| 1 | 5 | 0 |
| 2 | 6 | 1 |
| 3 | 7 | 0 |
| 4 | 8 | 1 |

A < 1.5

A > 1.5

| A | B | Class |
|---|---|---|
| 1 | 5 | 0 |

| A | B | Class |
|---|---|---|
| 2 | 6 | 1 |
| 3 | 7 | 0 |
| 4 | 8 | 1 |

# Gini index

- Najčešće korištena metrika u radu sa stabilma odlučivanja
- Mjera statističke disperzije - koliko je distribucija raširena ili sužena - koliko je heterogen ili homogen naš split
- Nastala kao mjera za mjerenje nejednakosti u primanjima

$$Gini = 1 - \sum_j p_j^2$$

- Kad izračunamo za svaki split - tražimo weighted average gini indexa za svaki split (weight - procenat dataseta koji se nalazi u splitu)

# Pronalaženje uslova grananja

- Idealno - prolazimo kroz sve mogućnosti (sve kombinacije kolona i njihovih vrijednosti) i pronalazimo split koji daje najmanju grešku
- Problem - veliki datasetovi - previše mogućnosti koji značajno usporavaju rad (za dovoljno velike datasetove algoritam gotovo neupotrebljiv)
- Heuristike i aproksimacije - ubrzavanje rada, dovoljno dobri rezultati
- Iteracija sve dok nije zadovoljen broj kriterijuma - određena dubina, broj listova, broj iteracija...

# Heuristike za određivanje podjele

1. Redukovani kandidatski splitovi
    - pronađemo sve moguće vrijednosti kolone i sortiramo ih
    - odaberemo određeni podskup vrijednosti koje se nalaze na istoj distanci
2. Radnom subspace sampling
    - biramo radnom podskup kolona pri svakoj iteraciji čije ćemo vrijednost gledati za split

# Spriječavanje overfittinga

- Podložno overfittingu - svaka iteracija nova podjela, novi listovi
- Tehnike za spriječavanje:
  - Early stopping (rano prekidanje) nakon nekog broja iteracija, maksimalne dubine ili broja listova
  - Pruning - pronalazimo listove i putanje koji ne doprinose previše predikciji:
    - pre-pruning - prije izgradnje stabla, slično early stopping, moguće i dodatne metrike (npr. min broj redova u splitu)
    - post-pruning - nakon izgradnje, Cost complexity pruning

# Cost complexity pruning (Weakest link pruning)

- Izbacivanje putanja koja nam ne doprinose previše
- Cost complexity - koliko data putanja doprinosi prediktivnoj moći našeg stabla - mijenjamo cijelu putanju listom koji reprezentuje klasu koja se najčešće pojavljuje
- Ideja za implementaciju:
  - Izračunamo Gini index (impurity) čvora
  - Izračamo sumu gini indexa podstabla (svih listova u podstablu)
  - Cost Complexity = (Total Impurity - Node Impurity) / (Number of Leaves in Subtree - 1)

# Zadatak za vježbu

- Kako modifikovati algoritam da radi sa regresijom?
- Kako računamo grešku pri radu sa regresijom?
- Kako biramo vrijednost za predikciju lista kod regresije?
- Da li se javljaju neki novi problemi kod regresije?

# Ensemble learning

- Učenje u ansamblu
- Nije vezano samo za stabla odlučivanja
- Podrazumijeva kombinaciju više metoda, modela algoritama prilikom predikcije za isti problem
- Osnovna ideja - više modela, manja šasna za grešku
- Kod stabala odlučivanja 2 osnovne tehnike
  - Bagging (Boostrap Aggregating)
  - Boosting

# Bagging - Random Forest

- Treniramo više nezavisnih baznih modela za podskupove podataka
- Bitni hiperparametar - koji broj baznih modela i koja veličina podskupa
- Kreiranje podskupa - Boostramp sampling - iz osnonog skupa biramo nasumično redove, s tim da je dozvoljeno ponavljanje
- Treniramo posebno stablo za svaki podskup
- Predikcija - agregiramo predikcije svih zasebnih stabala
- Paralelizacija jako bitna za brzinu

# Boosting

- Sekvencijalno treniramo više modela - cilj svakog novog modela je da ispravi greške prethodnog (pogrešne klasifikacije)
- Veliki broj različitih algoritama
- AdaBoost, XGBT, LightGBM, CatBoost...
- AdaBoost - osnovna ideja dodjeljivanje vecih tezina trening instancama na kojima model grijesi kako bi se povecala vjerovatnoca da model nauči da ih klasifikuje kako treba

# AdaBoost

- Inicijalizacija dataseta - dodijelimo svakoj trening instaci istu težinu - 1/N
- Iterativni trening slabih modela (weak learners)
  - Pri svakoj iteraciji treniramo model na težinskom datasetu
  - Računamo težinsku grešku = suma težina pogrešnko klasifikovanih instanci / suma težina svih instanci
  - Računamo težinu weak learner-a - alpha = ln((1 - error_rate) / error_rate)

# AdaBoost

- Update težina trening instanci
  - svaka tačno klasifikovana instanca - množimo njenu težinu sa exp(-alpha)
  - svaka pogrešno klasifikovana instanca - množimo njenu težinu sa exp(alpha)
  - normalizacija težina tako da im je suma 1
- Greška direktno zavisi od netačnih predikcija - svaka sljedeća iteracija teži da umanji grešku - fokus na netačne
- Predikcija - težinska suma slabih modela (težina = alpha)

# Bagging vs Boosting

- Nema tačnog odgovora - isprobati i vidjeti šta je bolje
- Heuristike:
  - bagging - dobre performanse kod visoke varijanse u modelima, dobre performanse kod treninga (laka paralelizacija)
  - boosting - dobre performanse kod visokog biasa u modelima i kod podataka sa puno nebalansiranosti i kompleksnijim vezama, podložnije overfitting-u

# Zaključak

- Stabla odlučivanja - uslovi na osnovu kojih radimo grananje prilikom donošenja odluka
- Iterativni, trening i minimizacija greške
- Gini index
- Heuristike za pretragu uslova grananja
- Izbjegavanje overfittinga - pruning, early stopping
- Bagging - vise baznih modela nad podskupom podataka
- Boosting - iterativno unaprijeđivanje weak learner-a na osnovu greške

# Hvala na pažnji!