Masinõpe

Sotsiaalse analüüsi meetodid:
kvantitatiivne lähenemine

Indrek Soidla

Masinõpe

Machine learning

- Traditsiooniline lähenemine andmeanalüüsis vs masinõpe
- Analüüsimeetodid võivad olla paljuski samad, erinev on eesmärk
- Järeldav statistika: seletada maailma, leida seoseid, järeldada populatsiooni tasandil
- Masinõpe: ennustada/prognoosida

Masinõpe vs järeldav statistika

- Mõlema lähenemise puhul koostatakse mudeleid
- JS: teooriast lähtudes
- MÕ: andmetest lähtudes, andmetest õppides
  - Ei alusta teooriast, vaid laseme algoritmidel andmete põhjal leida, milline parameetrite ja hüperparameetrite kombinatsioon võimaldab leida täpseima prognoosi
  - Masinõpe = teooria asemel õpitakse andmeid mudeldama andmete endi põhjal
- JS: kuidas *x* on *y*-ga seotud, kuidas *x* prognoosib *y*-t
  - nt regressioonikordaja väärtus
- MÕ: konkreetsed seosed (nende tugevus ja iseloom) pole olulised
  - Oluline on leida mudel, mis võimaldaks *y* väärtusi prognoosida (võimalikult täpselt)
  - Kui indiviidi kirjeldavad väärtused mingis tunnuste kogumis x_1, x_2,..., x_k, siis kuidas prognoosida võimalikult täpselt indiviidi väärtust tunnuses *y*?

Masinõpe vs järeldav statistika

- JS: teoreetilised eeldused => vähem tunnuseid => tõlgendamine lihtsam
- MÕ: andmetest lähtumine => palju andmeid, palju tunnuseid => must kast => tõlgendus pole keskne
  - Pole tähtis, mis värvi on kass, peaasi, et ta hiiri püüab
  - Kas siis teooria polegi oluline?
  - On küll, nt Google Flu
  - Samas, isesõitvad autod töötavad
  - Ei tähenda, et me ei peaks teadma, kuidas konkreetne masinõppe meetod töötab
  - Kui andmete esinduslikkus pole oluline, siis kuidas saab mudeli töötamises kindel olla?
  - Suurandmete rohkuse olulisus

Juhendatud ja juhendamata õpe

- Liigitamine *(classification)*
  - rühmakuuluvuse kriteeriumid on teada, indiviidid liigitatakse nende alusel
  - Juhendatud õpe *(supervised learning)*
  - Nt spämmifiltrid (eeldusel, et teada, millised meilid on spämm)
- Klasterdamine, klasteranalüüs *(clustering)*
  - teatud tunnuste alusel sarnastest indiviididest moodustatakse rühmad, kriteeriumid (tunnuste väärtuste kombinatsioonid) pole eelnevalt teada
  - Juhendamata õpe *(unsupervised learning)*
  - Nt liikluskindlustuses kõrge riskikoefitsiendiga klientide hulgas eristuvate gruppide leidmiseks

Mudel

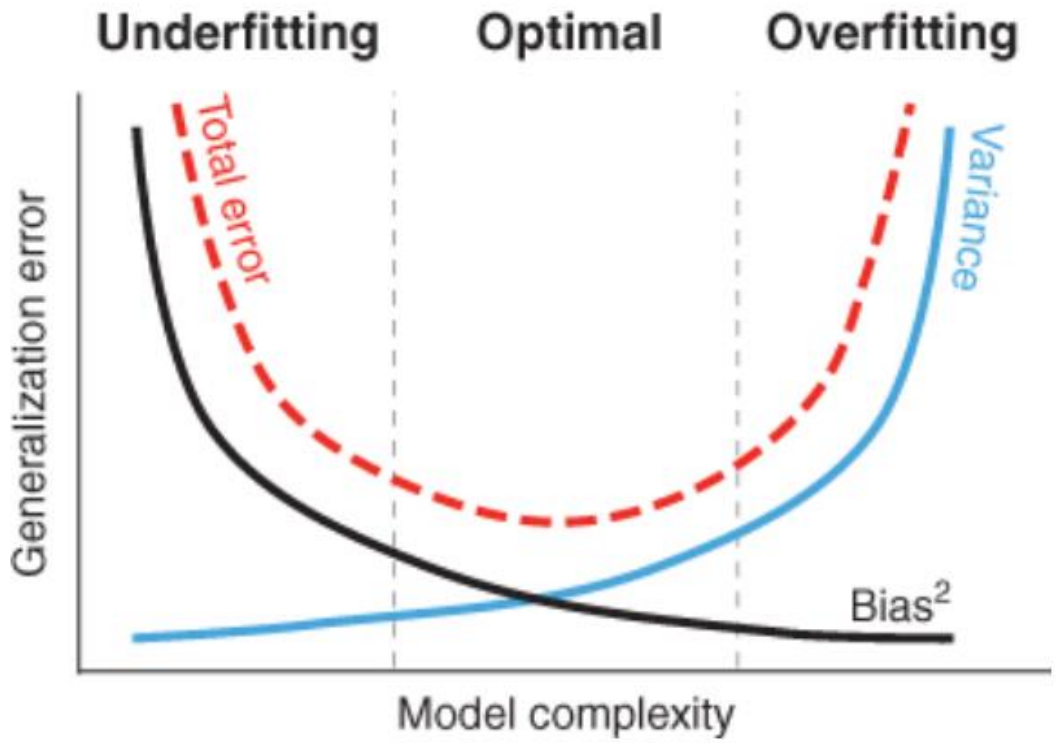
- Klassifitseerimise aluseks tavaliselt mingi mudel
- Regressioonis ka mudel => erinevus masinõppest?
- Ei pruugigi olla, logistiline regressioon üks enimkasutatavaid lihtsamaid meetodeid masinõppes
  - Vt eespool erinevus järeldavast statistikast
- Keskne ülesanne luua mudel, mis prognoosiks võimalikult täpselt
- Millest mudeli täpsus sõltub?
- Kui palju olulisi tegureid mudel arvestab
- Mida rohkem tegureid (tunnuseid), seda täpsem mudel?

Mudeli täpsus

- Näide klassifitseerimisest
- Alasobitus *(underfitting)* vs ülesobitus *(overfitting)*
- *Bias-variance tradeoff:*
- Mida lihtsam mudel (võtab arvesse vähem tunnuseid),
  - seda suurem nihe prognooside ja y tegelike väärtuste vahel
  - seda väiksem variatiivsus prognoosides erinevate andmestike lõikes
- Mida keerulisem mudel,
  - seda väiksem nihe prognooside ja y tegelike väärtuste vahel
  - seda suurem variatiivsus prognoosides erinevate andmestike lõikes
- Alasobituse korral jätame olulise osa andmete variatiivsusest kirjeldamata
- Ülesobituse korral mudeldame andmetes müra
  - Püüame mõtestada: kust tuleb müra andmetesse küsitlusandmete puhul? Mida tähendab müra mudeldamine sel juhul?

Mudeli täpsus

- Nii ala- kui ülesobituse korral tekib viga
- Masinõppes mudeli optimeerimisel keskne optimaalse tasakaalu leidmine



(Rhys 2020)

Mudeli optimeerimine

- Parameeter
  - Näitaja, mille väärtust püüame mudeliga hinnata
  - Nt regressioonikordaja
- Hüperparameeter
  - Näitaja, mille väärtuse varieerimisega püüame mudelit optimeerida
  - Nt ruutliikme olemasolu (astme väärtus) mudelis
  - Ei ole parameeter, vaid midagi, millega püüame parameetri väärtust täpsemaks muuta
- Suur osa masinõppe protsessist
  - sobivate hüperparameetrite väärtuste leidmine nii, et parameetrid oleksid võimalikult täpsed
  - mudeli optimaalse sobitusastme juures

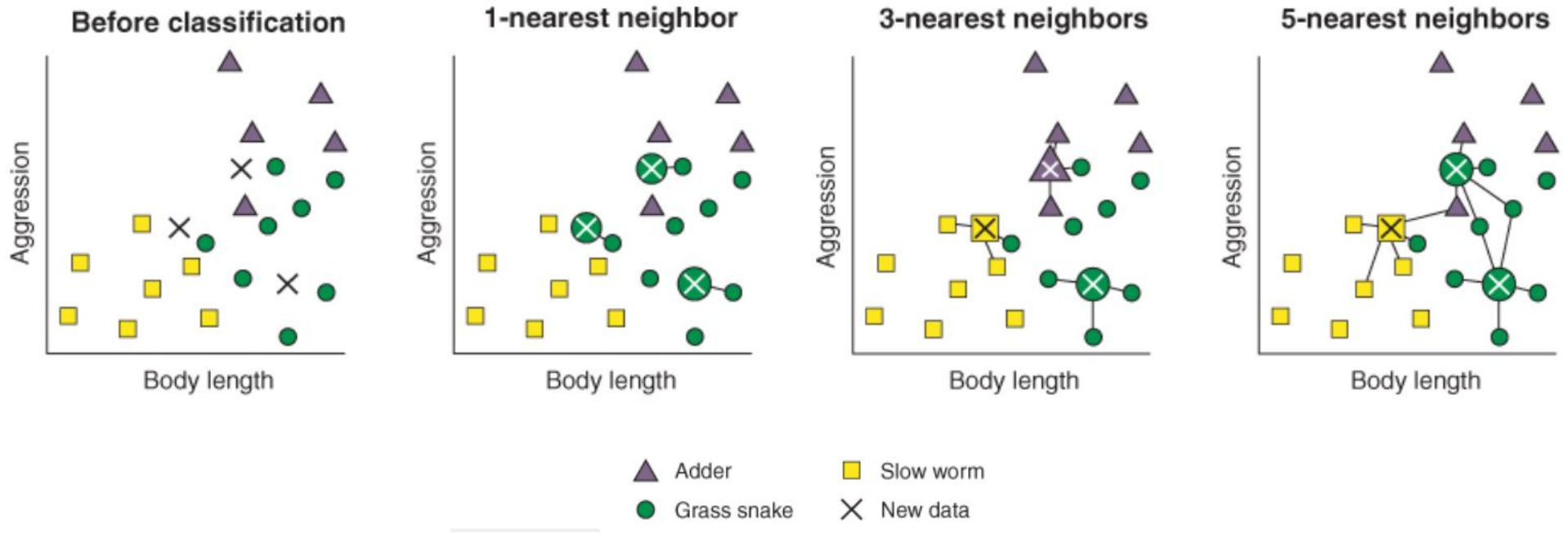
Mudeli optimeerimine

- Optimaalne sobitusaste – eeldab hindamist, kuidas mudel prognoosib uute andmetega
- Testime mudelit erinevatel andmetel?
  - Ajakulukas
  - Täpsuse hindamiseks eeldab y tegelike väärtuste olemasolu
- Testime mudelit samadel andmetel, millega mudel treeniti?
- Täpselt samade andmete kasutamine ülehindab mudeli täpsust
- Jaotame andmestiku treeningandmeteks ja valideerimisandmeteks
- Erinevad hinnangud, kui suur kumbki grupp, reeglina 80%/20%

Klassifitseerimise näide: kNN

- k lähima naabri algoritm *(k Nearest Neighbour)*
- Põhimõte sarnane k-keskmiste klasterdusele
- Leitakse indiviidile lähimad *k* indiviidi (naabrit) ja prognoositakse klassikuuluvust nende naabrite klassikuuluvuse põhjal
- *k* = arvessevõetavate naabrite arv
- Prognoositakse = omistatakse sama klass, mis enamikul naabritel
- Indiviidide lähedus/kaugus arvutatakse klassifitseerimise aluseks olevate tunnuste väärtuste põhjal
  - Suuremad erinevused => kaugemad/erinevamad indiviidid
  - Kauguse arvutamiseks erinevad valemid
  - Enimkasutatav eukleidiline kaugus
  - Eeldab arvulisi tunnuseid (klassikuuluvuse tunnus kategoriaalne)

Klassifitseerimise näide: kNN



(Rhys 2020)

- Millise *k* väärtuse valime, sellest sõltub klassifitseerimise tulemus
- Väike *k* => suur täpsus treeningandmetes, suur varieeruvus hiljem
- Suur *k* => ebatäpsem treeningandmetes, robustsem tulemus hiljem
- *k* on antud juhul hüperparameeter

KNN R-s

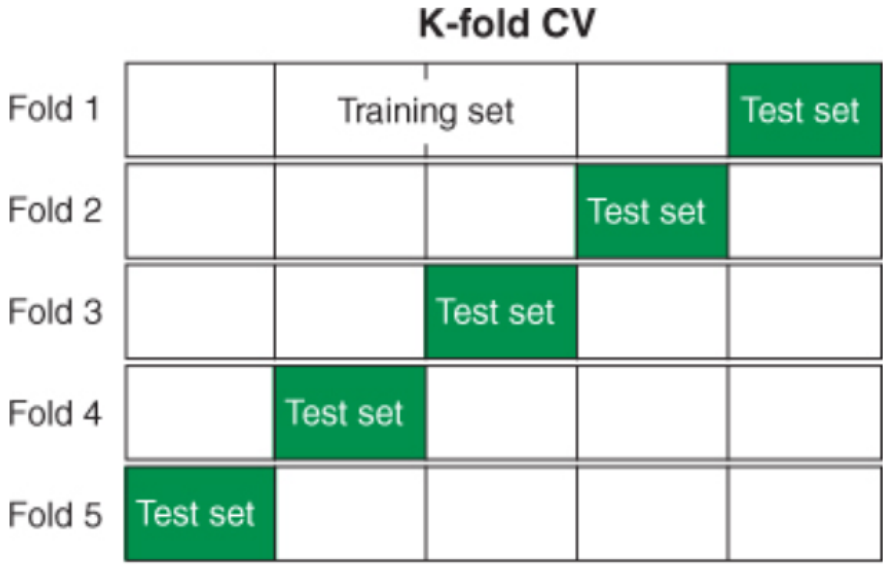
- Skriptifail Moodle-s

Ühekordne valideerimine *(holdout cross-validation)*

- Annab mõnevõrra varieeruvad tulemused, sõltuvalt sellest, kes täpselt valideerimiskogumisse satuvad
- Mõttekas kasutada ainult juhul, kui valideerimine väga ressursimahukas

K-fold CV

- Jaotame andmestiku *k* osaks (võrdsete suurustega)
- Viime valideerimise läbi *k* korda, iga kord on valideerimiskogumiks üks (erinev) osa andmetest
- Treeningandmeteks ülejäänud andmestik (samuti iga kord mõnevõrra erinev osa andmetest)
- Igal valideerimisel saadakse mudeli täpsuse näitajad, *k* näitajat keskmistatakse
- Oluline: *k* kogumid omavahel ei kattu, iga indiviid valideerimiskogumis üks kord

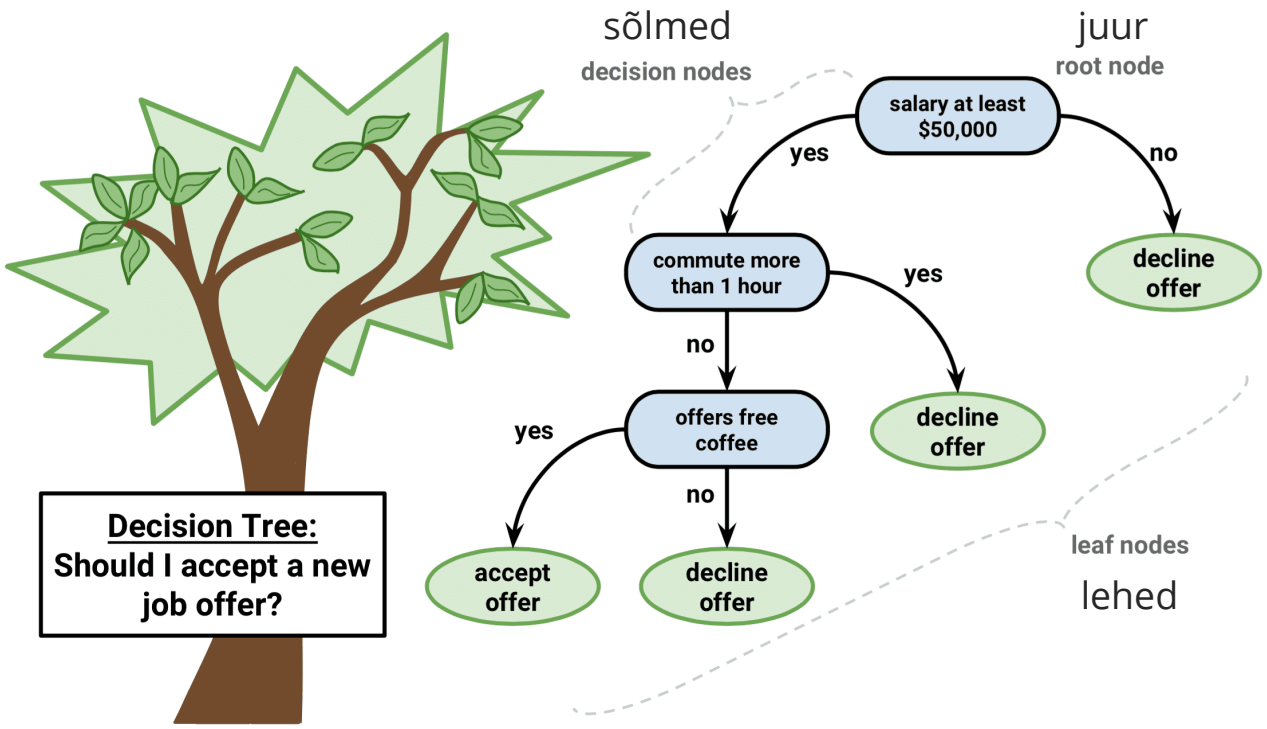


Leave One Out Cross-Validation ehk LOOCV

- Testkogumis üks indiviid, mudel treenitakse ülejäänud andmete peal
- Valideeritakse iga indiviidi peal eraldi
- Andestikus *n* indiviidi => *n* valideerimist => ressursimahukas
- Sisuliselt *k-fold CV*, kus *k* = *n*
- Igal valideerimisel saadakse mudeli täpsuse näitajad, mis keskmistatakse
- Igal valideerimisel testitakse ühe indiviidi peal
  - Testitulemused varieeruvad palju
  - Variatiivsus siiski väiksem kui *k-fold CV* puhul, kui andmestik väike
  - Ka ressursimahukuse tõttu mõttekas eelkõige väiksema andmestiku puhul

Otsustuspuu

Decision tree, CART (classification and regression tree)



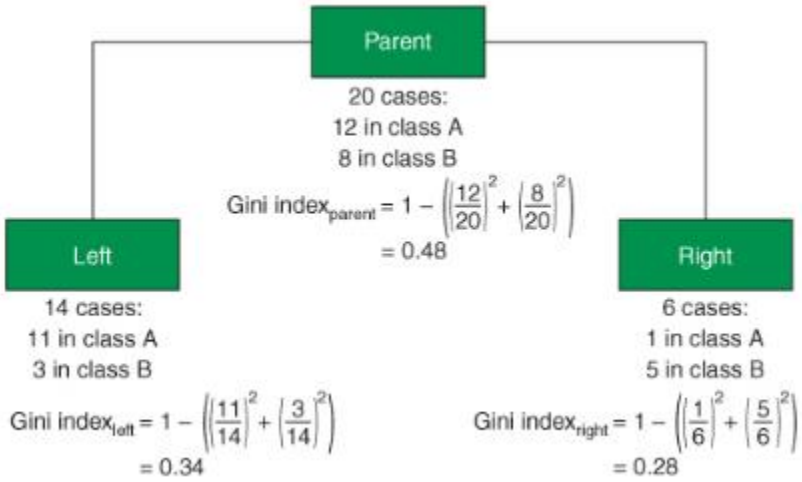
(https://www.datacamp.com/community/tutorials/decision-trees-R)

- Jaotame andmed väiksematesse osadesse
- Kui..., siis...
- Igas sõlmes jaotus kaheks
  - arvuline tunnus => lävend
  - kategoriaalne tunnus => üks kategooria vs ülejäänud
- Üks jaotus (lahutus, poolitus) korraga
  - st ühe tunnuse alusel korraga

Jaotamine

Splitting

- Jaotamine tehakse igas etapis selle tunnuse ja lävendi põhjal, mis maksimeerib vea vähenemise
- Mille põhjal vea vähenemist hinnatakse?
  - Puhtus/ebapuhtus *(purity/impurity)* – kui homogeenne on leht/sõlm (kas indiviidid langevad samasse klassi)
- Mõnevõrra erinevad algoritmid
  - Entroopia
  - Gini indeks – hinnatakse klasside osakaalu varieeruvust paarikaupa
  - Erinevad kaugusmõõdud ja statistikud (nt χ^2)
  - Algoritmi valik mängib tunduvalt väiksemat rolli kui erinevate hüperparameetrite valik



$$\text{Gini index}_{\text{split}} = p(\text{left}) \times \text{Gini index}_{\text{left}} + p(\text{right}) \times \text{Gini index}_{\text{right}}$$

$$\text{Gini index}_{\text{split}} = \frac{14}{20} \times 0.34 + \frac{6}{20} \times 0.28 = 0.32$$

Vea vähenemine: 0,48 – 0,32 = 0,16

(Rhys 2020)

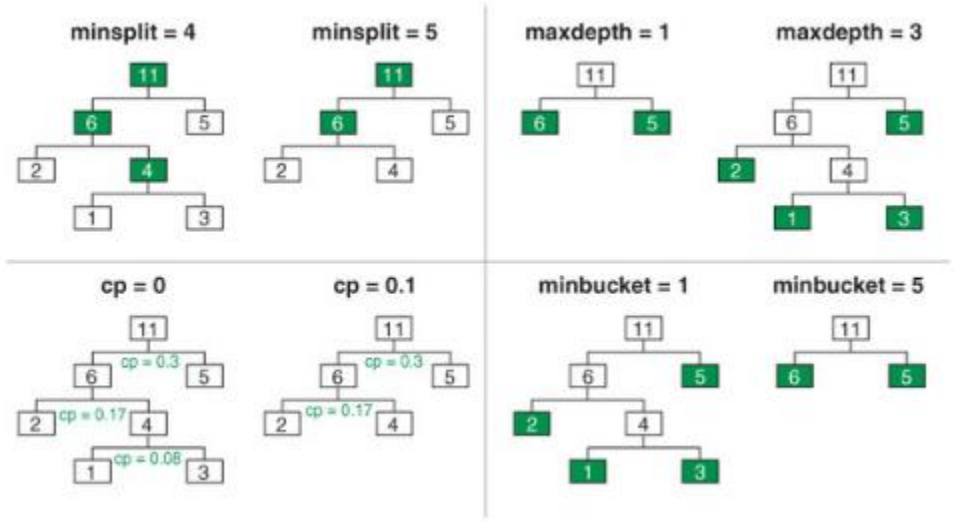
Jaotamine

Splitting

- Kui leht ei ole homogeenne, klassifitseeritakse ta enamuse kuuluvuse alusel
- Jaotamine toimub nii kaua kuni alles on vaid lehed
  - St midagi enam jaotada pole või jaotamine ei muuda puud täpsemaks
- Mida rohkem lehti, seda komplekssem puu
  - Klassifitseerimise tulemus täpsem, aga tõenäoliselt ülesobitatud
  - Andmestike lõikes suurem variatiivsus
  - Mida teha?
  - Saab seada erinevaid kriteeriume, et puu suurust kontrollida
  - = hüperparameetrid

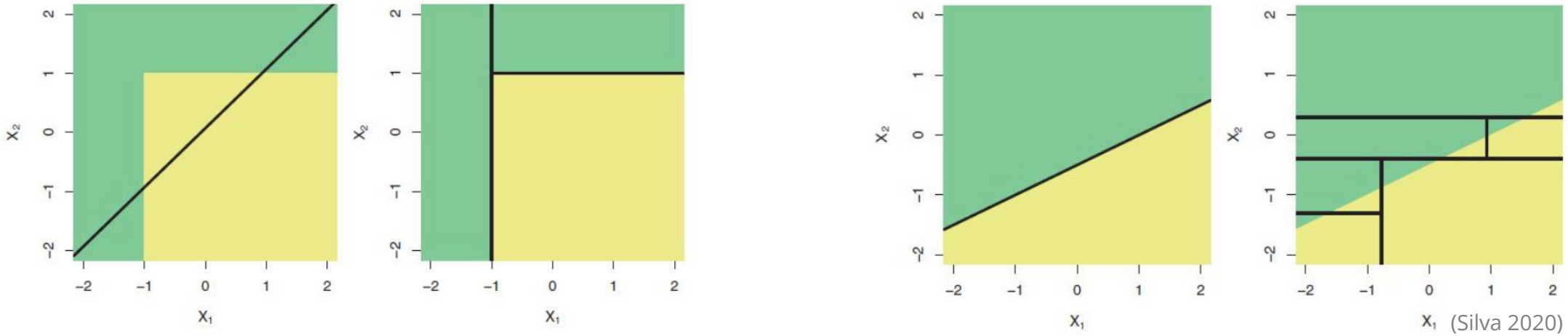
Otsustuspuu hüperparameetrid

- *minsplit* – väikseim lubatud indiviidide arv sõlmes enne jaotamist
  - Kui indiviide on sellest kriteeriumist vähem, sõlme enam ei jaotata, sõlm = leht
- *maxdepth* – puu suurim lubatud sügavus (kõrgus)
  - Järjestikuste sõlmede arv
- *minbucket* – väikseim lubatud indiviidide arv lehes
- *cp (complexity parameter)* – väikseim lubatud mudeli paranemise määr



Otsustuspuu + ja -

- Sobib igat tüüpi andmetele
- Pole eeldusi tunnuste jaotuse osas
- Tunnuste skaala ei mängi rolli => pole vaja tunnuseid standardiseerida
- Saab hakkama andmelünkadega
- Lihtsasti tõlgendatav
  - Ei pruugi olla täpseim meetod, aga tihti optimaalne tasakaal täpsuse ja mudeli tõlgendatavuse vahel
- Sobib hästi, kui prognoosivad tunnused on seotud klassikuuluvusega mittelineaarselt
  - Ei sobi hästi, kui seotud lineaarselt



(Silva 2020)

Otsustuspuu + ja -

- Ahne algoritm – tehakse parim valik igas etapis üksikult, mitte terve puu kohta üldiselt
  - Ei pruugi jõuda tervikuna parima mudelini
  - Puu kasvab kuni kõik lehed on puhtad => ülesobitus
  - Suurte andmestike puhul ressursimahukas
- Ei pruugi olla tõhus
  - paljude tunnuste korral
  - ühe mõjuka tunnuse korral
- Ülesobituse oht üksiku puu puhul => edasiarendused

Otsustuspuu edasiarendused

- Kuidas saada samast andmestikust rohkem kui üks puu?
- Jaotame andmestiku osadeks?
  - Igas osas vähem indiviide kui algandmestikus => mudelid ei ole niivõrd täpsed
- Appi tuleb *bootstrapping*
  - andmestikust võetakse tagasipanekuga valimid
  - iga valim sama suur kui algne andmestik
  - igas valimis mingid indiviidid mitme eest
  - valimis esindatud keskmiselt 63% algsetest indiviididest
- *Bagging (Bootstrap AGGregating)*
  - mudel treenitakse iga valimi põhjal (üks valim = üks puu), tulemused agregeeritakse
  - lõpliku mudeli variatiivsus *(variance)* väheneb
  - kui puud omavahel tugevalt seotud, kasu vähem
- Juhumets *(random forest)* – iga jaotamise puhul tehakse *P* tunnusest juhuslik valik *F*
  - kui *P* = *F*, on tegu *bagging*'uga
  - muudab puud üksteisest erinevamaks => väheneb nii *bias* kui *variance*
  - ühe mõjuka tunnuse probleem väheneb