Strojno učenje in rudarjenje podatkov

Seminarska naloga pri predmetu Izbrana poglavja iz optimizacije

Nejc Trdin

Fakulteta za računalništvo in informatiko,
Fakulteta za matematiko in fiziko

5. 5. 2011

Pregled

Kako se učimo?

- Učenje s podajanjem znanja: "Učitelj" nam podaja znanje. Pojavi se težava, da ne razumemo učitelja. Dolgotrajno.
- Učenje z odkrivanjem: "Učenci" konstruiramo poizkuse in skušamo najti dobre posplošitve naših opažanj.
- Učenje iz primerov: "Učitelj" nam poda primere, katere poizkušamo učenci posplošiti.

Kako se učimo?

- Učenje s podajanjem znanja: "Učitelj" nam podaja znanje. Pojavi se težava, da ne razumemo učitelja. Dolgotrajno.
- Učenje z odkrivanjem: "Učenci" konstruiramo poizkuse in skušamo najti dobre posplošitve naših opažanj.
- Učenje iz primerov: "Učitelj" nam poda primere, katere poizkušamo učenci posplošiti.

Kako se učimo?

- Učenje s podajanjem znanja: "Učitelj" nam podaja znanje. Pojavi se težava, da ne razumemo učitelja. Dolgotrajno.
- Učenje z odkrivanjem: "Učenci" konstruiramo poizkuse in skušamo najti dobre posplošitve naših opažanj.
- Učenje iz primerov: "Učitelj" nam poda primere, katere poizkušamo učenci posplošiti.

Pregled

Učenje iz primerov

- Induktivno učenje.
- Domenski eksperti v večini primerov lažje podajo dobre primere, kot pa eksplicitne splošne teorije.
- Primeri:
  - Diagnoza pacienta.
  - Napovedovanje vremena.
  - Napoved biološke aktivnosti neke kemijske spojine

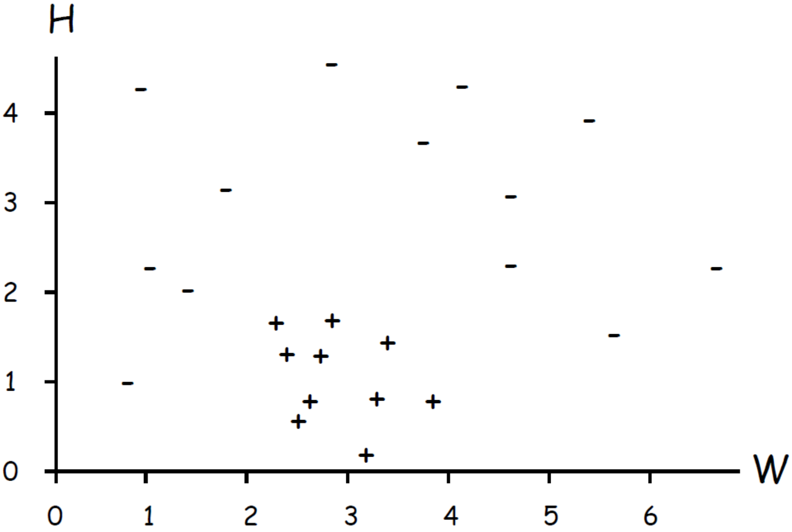
Učenje iz primerov

- Induktivno učenje.
- Domenski eksperti v večini primerov lažje podajo dobre primere, kot pa eksplicitne splošne teorije.
- Primeri:
  - Diagnoza pacienta.
  - Napovedovanje vremena.
  - Napoved biološke aktivnosti neke kemijske spojine

Pregled

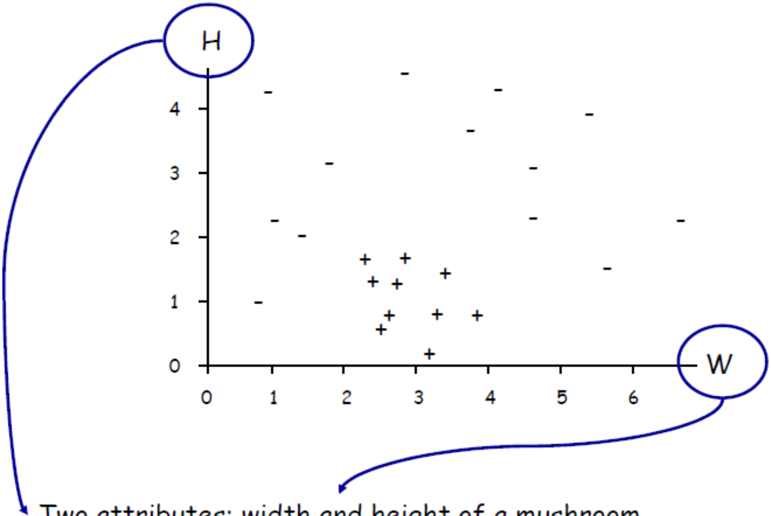
Primer

Gobe



Primer

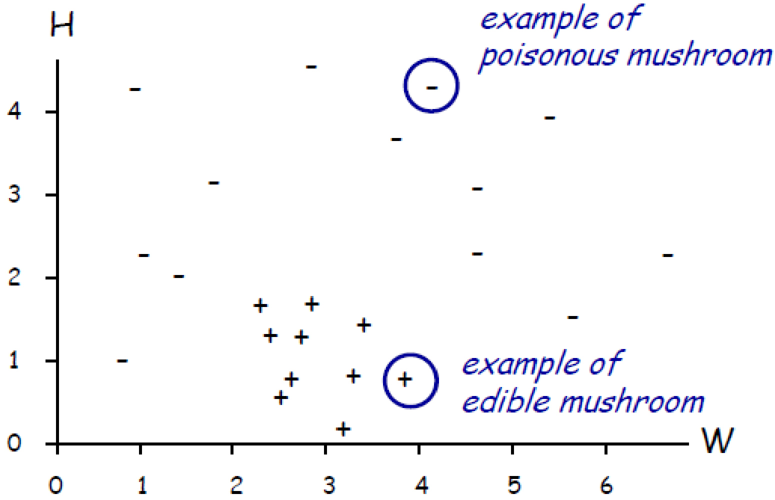
Atributi, razred



Two attributes: width and height of a mushroom

Primer

Atributi, razred



Primer

Učenje koncepta iz primerov

- Koncept si lahko predstavljamo kot podmnožico v prostoru vseh primerov.
  - U, množica vseh objektov (prostor primerov),
  - koncept C, $C \subset U$.
- Naučiti se koncepta C pomeni, da znamo določiti:
  - $\forall x \in U : x \in C \oplus x \notin C$.

Primer

Učenje koncepta iz primerov

- Koncept si lahko predstavljamo kot podmnožico v prostoru vseh primerov.
  - U, množica vseh objektov (prostor primerov),
  - koncept C, $C \subset U$.
- Naučiti se koncepta C pomeni, da znamo določiti:
  - $\forall x \in U : x \in C \oplus x \notin C$.

Primer

Učenje koncepta iz primerov

- Ponavadi je rezultat učenja **opis koncepta** ali **klasifikator**.
- Klasifikator je lahko podan na različne načine, z uporabo različnih formalizmov. (Concept description language, hypothesis languages)
- Klasifikatorji opisujejo hipoteze učenca (osnovane na učnih primerih!!!), glede ciljnega razreda. V splošnem nismo nikoli prepričani ali klasifikator izdelan iz primerov opisuje isto stvar, kot ciljni koncept.

Primer

Učenje koncepta iz primerov

- Ponavadi je rezultat učenja **opis koncepta** ali **klasifikator**.
- Klasifikator je lahko podan na različne načine, z uporabo različnih formalizmov. (Concept description language, hypothesis languages)
- Klasifikatorji opisujejo hipoteze učenca (osnovane na učnih primerih!!!), glede ciljnega razreda. V splošnem nismo nikoli prepričani ali klasifikator izdelan iz primerov opisuje isto stvar, kot ciljni koncept.

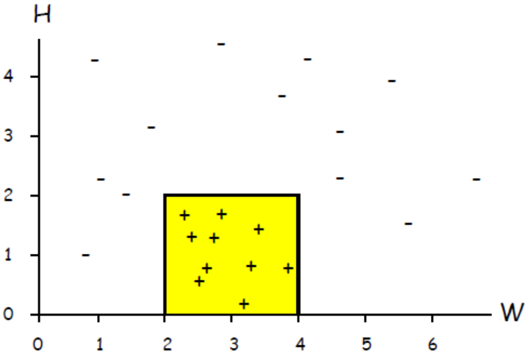
Primer

Učenje koncepta iz primerov

- Ponavadi je rezultat učenja **opis koncepta** ali **klasifikator.**
- Klasifikator je lahko podan na različne načine, z uporabo različnih formalizmov. (Concept description language, hypothesis languages)
- Klasifikatorji opisujejo hipoteze učenca (osnovane na učnih primerih!!!), glede ciljnega razreda. V splošnem nismo nikoli prepričani ali klasifikator izdelan iz primerov opisuje isto stvar, kot ciljni koncept.

Primer

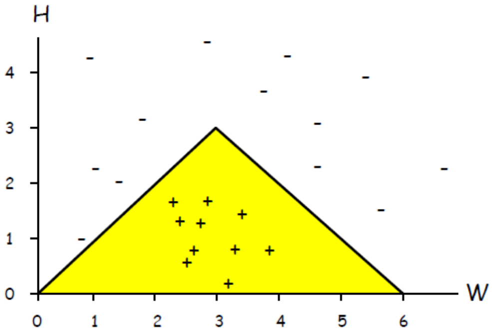
Hipoteza 1



IF 2 < W and W < 4 and H < 2 THEN ’užitna’
ELSE ’strupena’

Primer

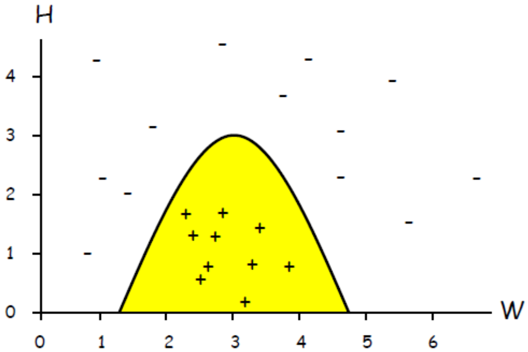
Hipoteza 2



IF H > W THEN 'strupena'
ELSE IF H > 6-W THEN 'strupena'
ELSE 'užitna'

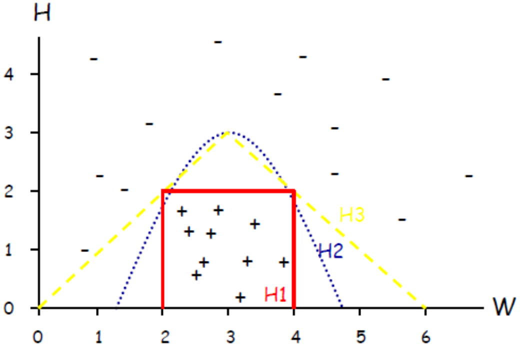
Primer

Hipoteza 3



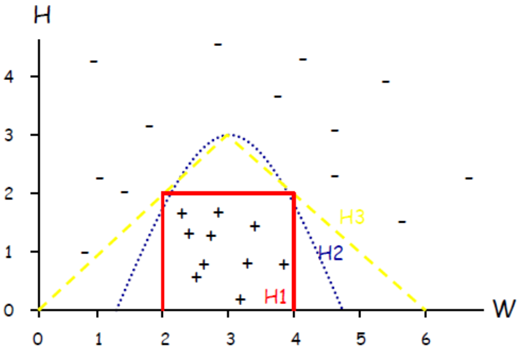
IF H < 3 - $(W-3)^2$ THEN 'užitna'
ELSE 'strupena'

Primer
Hipoteze



- Vse 3 hipoteze so konsistentne s podatki.
- Obstajajo razlike pri klasifikaciji novih primerov.

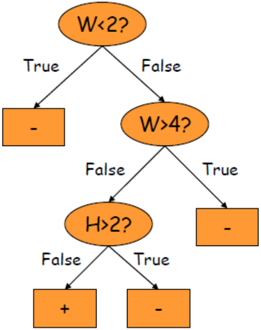
Primer
Hipoteze



- Vse 3 hipoteze so konsistentne s podatki.
- Obstajajo razlike pri klasifikaciji novih primerov.

Primer

Klasifikacijska drevesa



Opomba: Hipoteza 1 je predstavljena s prikazanim odločitvenim drevesom.

Točnost hipotez - klasifikatorjev

- Naj bo C ciljni koncept in C' inducirani koncept (naučeni klasifikator).
- Točnost klasifikatorja je delež pravilnih klasifikacij:

$$Acc(C') = \frac{|U - (C - C') - (C' - C)|}{|U|}$$

Točnost hipotez - klasifikatorjev

- Naj bo C ciljni koncept in C' inducirani koncept (naučeni klasifikator).
- Točnost klasifikatorja je delež pravilnih klasifikacij:

$$Acc(C') = \frac{|U - (C - C') - (C' - C)|}{|U|}$$

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Hipoteze

- Najbolj splošna hipoteza,
- najbolj specifična hipoteza,
- najkrajša hipoteza (v nekem formalizmu),
- najbolj točna hipoteza,
- najbolj razumljiva hipoteza (včasih je bolj zaželjena, kot točnejša hipoteza),
- Occamovo rezilo!

Pregled

Odločitvena drevesa

Primer 1: Podatki

#	*Attribute*			*Class*
	Outlook	Company	Sailboat	Sail?
1	sunny	big	small	yes
2	sunny	med	small	yes
3	sunny	med	big	yes
4	sunny	no	small	yes
5	sunny	big	big	yes
6	rainy	no	small	no
7	rainy	med	small	yes
8	rainy	big	big	yes
9	rainy	no	big	no
10	rainy	med	big	no

Odločitvena drevesa

Primer 1: Vprašanja

#	*Attribute*			*Class*
	Outlook	Company	Sailboat	Sail?
1	sunny	no	big	?
2	rainy	big	small	?

Pregled

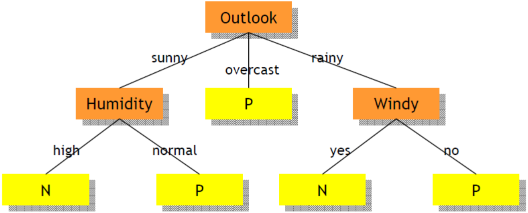
Odločitvena drevesa

Primer 2: Podatki

Igranje golfa

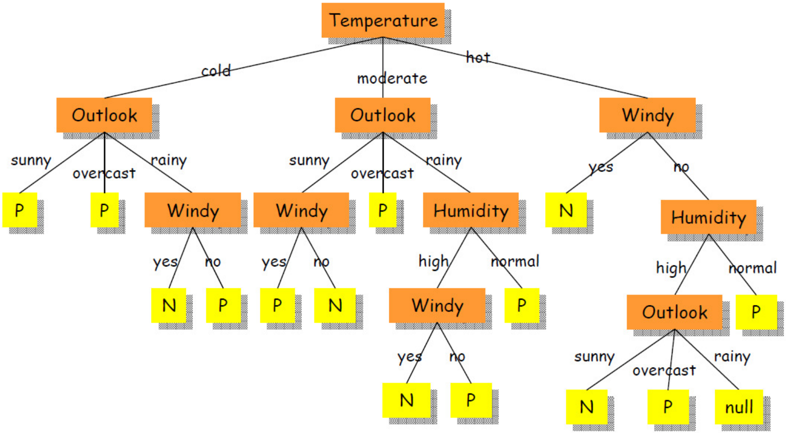
#	Attribute				Class
	Outlook	Temperature	Humidity	Windy	Play
1	sunny	hot	high	no	N
2	sunny	hot	high	yes	N
3	overcast	hot	high	no	P
4	rainy	moderate	high	no	P
5	rainy	cold	normal	no	P
6	rainy	cold	normal	yes	N
7	overcast	cold	normal	yes	P
8	sunny	moderate	high	no	N
9	sunny	cold	normal	no	P
10	rainy	moderate	normal	no	P
11	sunny	moderate	normal	yes	P
12	overcast	moderate	high	yes	P
13	overcast	hot	normal	no	P
14	rainy	moderate	high	yes	N

Odločitvena drevesa
Primer 2: Majhno drevo



Odločitvena drevesa

Primer 2: Veliko drevo



Pregled

1. Uvod v strojno učenje
   - Učenje iz primerov
   - Primer
2. Odločitvena drevesa
   - Primer 1
   - Primer 2
   - TDIDT
   - Rezanje odločitvenih dreves
3. Ocenjevanje rezultatov
   - Splošno
   - Prečno preverjanje
4. Naivni Bayesov klasifikator
   - Uvod
   - Nomogrami
5. SVM
   - Splošno
   - Izpeljava

TDIDT - Top-Down Induction of Decision Trees

Splošni algoritem

- Znan tudi kot ID3 (Quinlan).
- ```
  INPUT: Learning set S
  OUTPUT: Decision tree T
  ConstructTree(S):
  IF all e in S are of same class C THEN
    return leaf with label C
  ELSE
    select "most informative" attribute A
    partition S according to A's values
    recursively construct subtrees T1, T2,
      ..., Tk for subsets of S
    return central node with subtrees T1, T2,
      ..., Tk
  ```
  ```

TDIDT - Top-Down Induction of Decision Trees

Splošni algoritem

- Znan tudi kot ID3 (Quinlan).
-
```
INPUT: Learning set S
OUTPUT: Decision tree T
ConstructTree(S):
IF all e in S are of same class C THEN
  return leaf with label C
ELSE
  select "most informative" attribute A
  partition S according to A's values
  recursively construct subtrees T1, T2,
    ..., Tk for subsets of S
  return central node with subtrees T1, T2,
    ..., Tk
```

TDIDT

Izbira najbolj informativnega atributa?

- Glavni princip - Izberemo atribut, ki particionira množico S v čim bolj čiste podmnožice.
-
    - Entropija,
    - Gini index,
    - ReliefF,
    - χ^2,
    - $\ldots$,

TDIDT

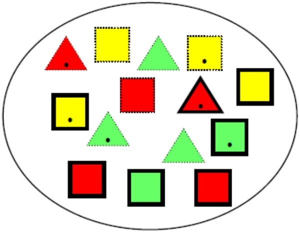
Izbira najbolj informativnega atributa?

- Glavni princip - Izberemo atribut, ki particionira množico S v čim bolj čiste podmnožice.
-
    - Entropija,
    - Gini index,
    - ReliefF,
    - χ^2,
    - $\ldots$,

TDIDT

Izračun mer

#	Attribute			Shape
	Color	Outline	Dot	
1	green	dashed	no	triange
2	green	dashed	yes	triange
3	yellow	dashed	no	square
4	red	dashed	no	square
5	red	solid	no	square
6	red	solid	yes	triange
7	green	solid	no	square
8	green	dashed	no	triange
9	yellow	solid	yes	square
10	red	solid	no	square
11	green	solid	yes	square
12	yellow	dashed	yes	square
13	yellow	solid	no	square
14	red	dashed	yes	triange



TDIDT
Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT
Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT

Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT

Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT

Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT

Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT
Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

TDIDT

Izračun mer

- Entropija, zmanjševanje entropije, entropija vrednosti atributa in residualna entropija.
- Infromation Gain
  - Hevristika - atribut z večjim IG je izbran, kot najbolj informativen.
  - Hevristika je lokalna! Lokalno minimizira nečistost.
  - Hevristika bolje oceni atribute z več vrednostmi, saj se S razbije v več podmnožic, katere (če so majhne) so lahko zelo čiste.
  - Rešitev - Information Gain Ratio
- Information Gain Ratio
- Gini index in Gini gain.

Pregled

Rezanje odločitvenih dreves

Motivacija

- Šum v podatkih - napake v meritvah, napake med pretvarjanjem podatkov, napake v primerih, manjkajoče vrednost, ...
- Protislovni primeri - v nekaterih domenah lahko imata dva primera ista atributna vektorja in različna razreda.
- Problemi:
  - Kompleksni modeli,
  - slabo razumljivi modeli,
  - predoločen klasifikator (overfitting),
  - nizka klasifikacijska točnost na novih primerih.

Rezanje odločitvenih dreves

Motivacija

- Šum v podatkih - napake v meritvah, napake med pretvarjanjem podatkov, napake v primerih, manjkajoče vrednost, ...
- Protislovni primeri - v nekaterih domenah lahko imata dva primera ista atributna vektorja in različna razreda.
- Problemi:
  - Kompleksni modeli,
  - slabo razumljivi modeli,
  - predoločen klasifikator (overfitting),
  - nizka klasifikacijska točnost na novih primerih.

Rezanje odločitvenih dreves

Motivacija

- Šum v podatkih - napake v meritvah, napake med pretvarjanjem podatkov, napake v primerih, manjkajoče vrednost, ...
- Protislovni primeri - v nekaterih domenah lahko imata dva primera ista atributna vektorja in različna razreda.
- Problemi:
  - Kompleksni modeli,
  - slabo razumljivi modeli,
  - predoločen klasifikator (overfitting),
  - nizka klasifikacijska točnost na novih primerih.

Rezanje odločitvenih dreves
Primer iz prakse

- Problem: Iskanje primarnega tumorja
- Podatki: 20 razredov. Večinski razred ima delež 24.7%.
- Predoločeno odločitveno drevo: 150 vozlišč. Klasifikacijska točnost 41%.
- Rezano odločitveno drevo: 15 vozlišč. Klasifikacijska točnost 45%.
- Domenski ekspert: Klasifikacijska točnost 42%.

Rezanje odločitvenih dreves

Primer iz prakse

- Problem: Iskanje primarnega tumorja
- Podatki: 20 razredov. Večinski razred ima delež 24.7%.
- Predoločeno odločitveno drevo: 150 vozlišč. Klasifikacijska točnost 41%.
- Rezano odločitveno drevo: 15 vozlišč. Klasifikacijska točnost 45%.
- Domenski ekspert: Klasifikacijska točnost 42%.

Rezanje odločitvenih dreves

Rezanje splošno

- Glavna vprašanja:
  - Koliko rezanja?
  - Kje rezati?
  - Katero je najboljše odrezano drevo?
- Pre-pruning (forward pruning):
  - Majhen Information Gain,
  - majhno število primerov v poddrevesu,
  - število primerov statistično nesignifikatno.
- Post-pruning.

Rezanje odločitvenih dreves

Rezanje splošno

- Glavna vprašanja:
  - Koliko rezanja?
  - Kje rezati?
  - Katero je najboljše odrezano drevo?
- Pre-pruning (forward pruning):
  - Majhen Information Gain,
  - majhno število primerov v poddrevesu,
  - število primerov statistično nesignifikatno.
- Post-pruning.

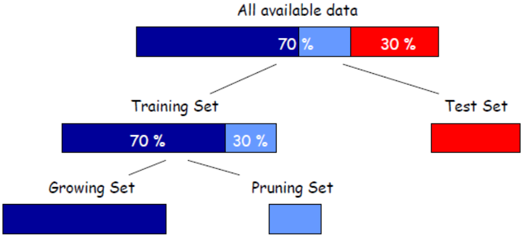
Rezanje odločitvenih dreves

Rezanje splošno

- Glavna vprašanja:
  - Koliko rezanja?
  - Kje rezati?
  - Katero je najboljše odrezano drevo?
- Pre-pruning (forward pruning):
  - Majhen Information Gain,
  - majhno število primerov v poddrevesu,
  - število primerov statistično nesignifikatno.
- Post-pruning.

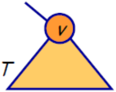
Rezanje odločitvenih dreves

Post-pruning



Rezanje odločitvenih dreves

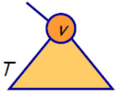
REP - Reduced Error Pruning



- Uporabljamo pruning set za ocenjevanje točnosti v poddrevesih in v posameznih vozliščih odločitvenega drevesa.
- Naj bo T poddrevo s korenom v v.
- Definiramo: $Gain_{pruning}(v) = \#(misclassifications\ in\ T) - \#(missclassifications\ at\ v)$
- Ponavljaj: odreži pri vozlišču v z največjim $Gain_{pruning}(v)$, dokler nimajo vsa vozlišča negativen $Gain_{pruning}$.

Rezanje odločitvenih dreves

REP - Reduced Error Pruning



- Uporabljamo pruning set za ocenjevanje točnosti v poddrevesih in v posameznih vozliščih odločitvenega drevesa.
- Naj bo T poddrevo s korenom v v.
- Definiramo: $Gain_{pruning}(v) = \#(misclassifications\ in\ T) - \#(missclassifications\ at\ v)$
- Ponavljaj: odreži pri vozlišču v z največjim $Gain_{pruning}(v)$, dokler nimajo vsa vozlišča negativen $Gain_{pruning}$.

Rezanje odločitvenih dreves

REP

- "Bottom-up restriction": T lahko odrežemo samo takrat, ko ne vsebuje poddrevesa T' z manjšo napako, kot jo ima T.
- **Trditev(Esposito)**: REP z bottom-up omejitvijo najde najmanjše in najbolj točno poddrevo v odvisnosti od pruning set.
- Z drugimi besedami, REP z bottom-up omejitvijo izmed množice vseh odrezanih poddreves najde tisto, ki ima najmanjšo napako na pruning set-u.

Rezanje odločitvenih dreves
REP

- "Bottom-up restriction": T lahko odrežemo samo takrat, ko ne vsebuje poddrevesa T' z manjšo napako, kot jo ima T.
- **Trditev(Esposito)**: REP z bottom-up omejitvijo najde najmanjše in najbolj točno poddrevo v odvisnosti od pruning set.
- Z drugimi besedami, REP z bottom-up omejitvijo izmed množice vseh odrezanih poddreves najde tisto, ki ima najmanjšo napako na pruning set-u.

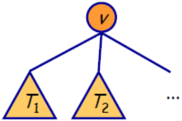
Rezanje odločitvenih dreves

MEP - Minimal Error Pruning

- Ne potrebuje pruning set-a za delovanje.
- Za ocenjevanje verjetnosti uporablja Bayes-ove metode (Laplace ali m-ocena).
- Glavni ideji:
    - Režemo od spodaj navzgor.
    - Režemo tako, da je ocena klasifikacijske napake najmanjša.

Rezanje odločitvenih dreves

MEP - Minimal Error Pruning



- Odločamo se o rezanju poddrevesa T s korenom v v in poddrevesi $T_1, T_2, \ldots, T_k$.
- Definirajmo: statična napaka v v, kot $e(v) = Pr[class \neq C|v]$, kjer je C večinski razred v vozlišču v.
- Če je T odrezano poddrevo (vozlišče), potem $E(T) = e(v)$
- sicer je $E(T) = p_1 E(T_1) + p_2 E(T_2) + \ldots + p_k E(T_k)$.

Rezanje odločitvenih dreves

MEP - Minimal Error Pruning

- Režemo bottom-up, če $e(v) \leq E(T)$.
- Napaka v poddrevesu T je tako $error(T) = min(e(v), \sum_i p_i E(T_i))$
- Kako ocenimo $e(v)$?
- Z m-oceno ali Laplace-ovo oceno.

Rezanje odločitvenih dreves

MEP - Minimal Error Pruning

- Režemo bottom-up, če $e(v) \leq E(T)$.
- Napaka v poddrevesu T je tako $error(T) = min(e(v), \sum_i p_i E(T_i))$
- Kako ocenimo $e(v)$?
- Z m-oceno ali Laplace-ovo oceno.

Rezanje odločitvenih dreves

MEP - Minimal Error Pruning

- Režemo bottom-up, če $e(v) \leq E(T)$.
- Napaka v poddrevesu T je tako $error(T) = min(e(v), \sum_i p_i E(T_i))$
- Kako ocenimo $e(v)$?
- Z m-oceno ali Laplace-ovo oceno.

Rezanje odločitvenih dreves

MEP - Minimal Error Pruning

- Režemo bottom-up, če $e(v) \leq E(T)$.
- Napaka v poddrevesu T je tako $error(T) = min(e(v), \sum_i p_i E(T_i))$
- Kako ocenimo $e(v)$?
- Z m-oceno ali Laplace-ovo oceno.

Rezanje odločitvenih dreves

Laplace-ova ocena

- V vozlišu v je N primerov, n_C je število primerov z večinskim razredom in k je število razredov v celotni množici primerov.
- Tedaj je Laplace-ova ocena $p_C(v) = \frac{n_C+1}{N+k}$
- Problemi z oceno:
  1. Predvideva, da so verjetnosti razredov apriorno enako verjetne.
  2. Stopnja rezanja je odvisna od števila razredov.

Rezanje odločitvenih dreves

Laplace-ova ocena

- V vozlišu v je N primerov, n_C je število primerov z večinskim razredom in k je število razredov v celotni množici primerov.
- Tedaj je Laplace-ova ocena $p_C(v) = \frac{n_C+1}{N+k}$
- Problemi z oceno:
  1. Predvideva, da so verjetnosti razredov apriorno enako verjetne.
  2. Stopnja rezanja je odvisna od števila razredov.

Rezanje odločitvenih dreves

m-ocena

- Veljajo enake veličine, kot pri Laplace-ovi oceni, z dodanim p_{C_a}, ki je apriorna verjetnost večinskega razreda na celotni množici primerov. Parameter m pa je nenegativno število, ki ga določi domenski ekspert.
- Tedaj je m-ocena $p_C(v) = \frac{p_{C_a} m + n_C}{N + m}$
- Opombe:
  - V račun vzame apriorne verjetnosti razredov.
  - Rezanje ni občutljivo na število razredov.
  - S spreminjanjem parametra m dobimo več različnih odrezanih dreves.
  - Izbira parametra m je odvisna tudi od 'sigurnosti' v podatkih. Malo šuma, majhen m in malo rezanja. Če je v podatkih veliko šuma, izberemo večji m in pride do veliko rezanja.

Rezanje odločitvenih dreves

m-ocena

- Veljajo enake veličine, kot pri Laplace-ovi oceni, z dodanim p_{C_a}, ki je apriorna verjetnost večinskega razreda na celotni množici primerov. Parameter m pa je nenegativno število, ki ga določi domenski ekspert.
- Tedaj je m-ocena $p_C(v) = \frac{p_{C_a} m + n_C}{N + m}$
- Opombe:
  - V račun vzame apriorne verjetnosti razredov.
  - Rezanje ni občutljivo na število razredov.
  - S spreminjanjem parametra m dobimo več različnih odrezanih dreves.
  - Izbira parametra m je odvisna tudi od 'sigurnosti' v podatkih. Malo šuma, majhen m in malo rezanja. Če je v podatkih veliko šuma, izberemo večji m in pride do veliko rezanja.

Rezanje odločitvenih dreves

m-ocena

- Veljajo enake veličine, kot pri Laplace-ovi oceni, z dodanim p_{C_a}, ki je apriorna verjetnost večinskega razreda na celotni množici primerov. Parameter m pa je nenegativno število, ki ga določi domenski ekspert.
- Tedaj je m-ocena $p_C(v) = \frac{p_{C_a} m + n_C}{N + m}$
- Opombe:
  - V račun vzame apriorne verjetnosti razredov.
  - Rezanje ni občutljivo na število razredov.
  - S spreminjanjem parametra m dobimo več različnih odrezanih dreves.
  - Izbira parametra m je odvisna tudi od 'sigurnosti' v podatkih. Malo šuma, majhen m in malo rezanja. Če je v podatkih veliko šuma, izberemo večji m in pride do veliko rezanja.

Pregled

Ocenjevanje rezultatov

Splošno

- Kriteriji:
  - Natančnost induciranih konceptov. (Natančnost = verjetnost pravilne klasifikacije novega primera)
  - Razumljivost modela.
- Obe točki sta pomembni, vendar je težko izmeriti razumljivost modela.
- Različne natančnosti:
  - Natančnost na učnih podatkih.
  - Natančnost na novih podatkih (veliko bolj pomembno). Velik dela se vloži tudi v iskanje različnih cenlik za ocenjevanje natančnosti modelov na novih podatkih.
- Pogosta napaka: Ocena natančnosti na novih podatkih naj bo kar natančnost na učnih podatkih.

Ocenjevanje rezultatov

Splošno

- Kriteriji:
  - Natančnost induciranih konceptov. (Natančnost = verjetnost pravilne klasifikacije novega primera)
  - Razumljivost modela.
- Obe točki sta pomembni, vendar je težko izmeriti razumljivost modela.
- Različne natančnosti:
  - Natančnost na učnih podatkih.
  - Natančnost na novih podatkih (veliko bolj pomembno). Velik dela se vloži tudi v iskanje različnih cenlik za ocenjevanje natančnosti modelov na novih podatkih.
- Pogosta napaka: Ocena natančnosti na novih podatkih naj bo kar natančnost na učnih podatkih.

Ocenjevanje rezultatov

Splošno

- Kriteriji:
  - Natančnost induciranih konceptov. (Natančnost = verjetnost pravilne klasifikacije novega primera)
  - Razumljivost modela.
- Obe točki sta pomembni, vendar je težko izmeriti razumljivost modela.
- Različne natančnosti:
  - Natančnost na učnih podatkih.
  - Natančnost na novih podatkih (veliko bolj pomembno). Velik dela se vloži tudi v iskanje različnih cenlik za ocenjevanje natančnosti modelov na novih podatkih.
- Pogosta napaka: Ocena natančnosti na novih podatkih naj bo kar natančnost na učnih podatkih.

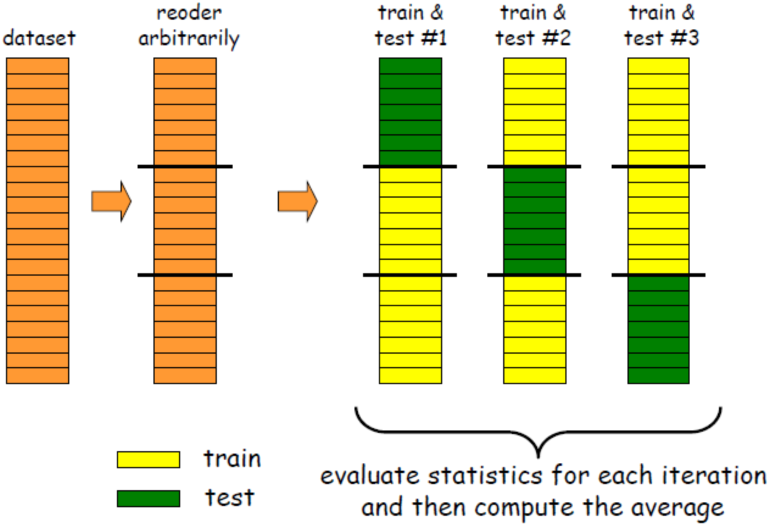
Ocenjevanje rezultatov

Splošno

- Kriteriji:
  - Natančnost induciranih konceptov. (Natančnost = verjetnost pravilne klasifikacije novega primera)
  - Razumljivost modela.
- Obe točki sta pomembni, vendar je težko izmeriti razumljivost modela.
- Različne natančnosti:
  - Natančnost na učnih podatkih.
  - Natančnost na novih podatkih (veliko bolj pomembno). Velik dela se vloži tudi v iskanje različnih cenlik za ocenjevanje natančnosti modelov na novih podatkih.
- Pogosta napaka: Ocena natančnosti na novih podatkih naj bo kar natančnost na učnih podatkih.

Pregled

Ocenjevanje rezultatov
Prečno preverjanje



Ocenjevanje rezultatov

k-kratno prečno preverjanje

- Razbij podatke v k podmnožic približno enake velikosti.
- Za $i = 1$ do k: i-to podmnožico uporabi za testiranje modela, ki si ga zgradil na preostalih $(k-1)$ množicah.
- Izračunaj povprečno točnost.

Ocenjevanje rezultatov

k-kratno prečno preverjanje

- Razbij podatke v k podmnožic približno enake velikosti.
- Za $i = 1$ do k: i-to podmnožico uporabi za testiranje modela, ki si ga zgradil na preostalih $(k-1)$ množicah.
- Izračunaj povprečno točnost.

Ocenjevanje rezultatov

k-kratno prečno preverjanje

- Razbij podatke v k podmnožic približno enake velikosti.
- Za $i = 1$ do k: i-to podmnožico uporabi za testiranje modela, ki si ga zgradil na preostalih $(k - 1)$ množicah.
- Izračunaj povprečno točnost.

Pregled

Naivni Bayesov klasifikator

Uvod

$$Pr[e|X] = Pr[e]\frac{Pr[X|e]}{Pr[X]} = Pr[e]\frac{Pr[\cap_{i=1}^{n} x_i|e]}{Pr[X]}$$

Če so x_i neodvisne potem velja:

$$Pr[\cap_{i=1}^{n} x_i|e] = \prod_{i=1}^{n} Pr[x_i|e]$$

od koder lahko zapišemo:

$$Pr[e|X] = Pr[e]\frac{\prod_{i=1}^{n} Pr[x_i|e]}{Pr[X]}$$

Naivni Bayesov klasifikator
Uvod

$$Pr[e|X] = Pr[e]\frac{Pr[X|e]}{Pr[X]} = Pr[e]\frac{Pr[\cap_{i=1}^{n} x_i|e]}{Pr[X]}$$

Če so x_i neodvisne potem velja:

$$Pr[\cap_{i=1}^{n} x_i|e] = \prod_{i=1}^{n} Pr[x_i|e]$$

od koder lahko zapišemo:

$$Pr[e|X] = Pr[e]\frac{\prod_{i=1}^{n} Pr[x_i|e]}{Pr[X]}$$

Naivni Bayesov klasifikator
Uvod

$$Pr[e|X] = Pr[e]\frac{Pr[X|e]}{Pr[X]} = Pr[e]\frac{Pr[\cap_{i=1}^{n} x_i|e]}{Pr[X]}$$

Če so x_i neodvisne potem velja:

$$Pr[\cap_{i=1}^{n} x_i|e] = \prod_{i=1}^{n} Pr[x_i|e]$$

od koder lahko zapišemo:

$$Pr[e|X] = Pr[e]\frac{\prod_{i=1}^{n} Pr[x_i|e]}{Pr[X]}$$

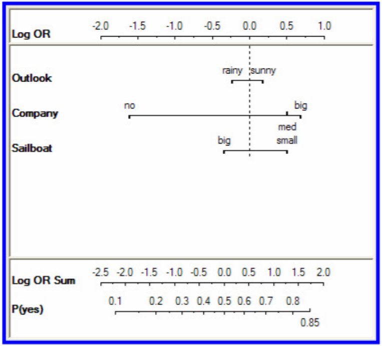
Naivni Bayesov klasifikator
Uvod

$$\frac{Pr[e|X]}{Pr[\neg e|X]} = \frac{Pr[e]\prod_{i=1}^{n} Pr[x_i|e]}{Pr[\neg e]\prod_{i=1}^{n} Pr[x_i|\neg e]}$$

Pregled

Naivni Bayesov klasifikator
Nomogrami

#	Outlook	Company	Sailboat	Sail
1	rainy	big	big	yes
2	rainy	big	small	yes
3	rainy	med	big	yes
4	rainy	med	small	yes
5	sunny	big	big	yes
6	sunny	big	small	yes
7	sunny	med	big	yes
8	sunny	med	big	yes
9	sunny	med	small	yes
10	sunny	no	small	yes
11	sunny	no	big	no
12	rainy	med	big	no
13	rainy	no	big	no
14	rainy	no	big	no
15	rainy	no	small	no
16	rainy	no	small	no
17	sunny	big	big	no
18	sunny	big	small	no
19	sunny	med	big	no
20	sunny	med	big	no



Pregled

Support Vector Machines
Splošno

- p-dimenzionalen vektor z dodanim razredom.
- razred je binaren.
- iščemo takšno $(p-1)$-dimenzionalno hiperravnino, da bodo negativni primeri ležali na eni strani in pozitivni na drugi strani hiperravnine.
- Ker jih obstaja več, želimo najti takšno, ki predstavlja največjo razmaknjenost dveh razredov.
- Izberemo takšno, ki ima največjo razdaljo do najbližjih primerov na vsaki strani.

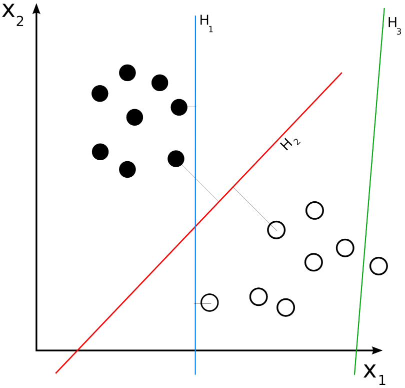
Support Vector Machines
Splošno

- p-dimenzionalen vektor z dodanim razredom.
- razred je binaren.
- iščemo takšno $(p-1)$-dimenzionalno hiperravnino, da bodo negativni primeri ležali na eni strani in pozitivni na drugi strani hiperravnine.
- Ker jih obstaja več, želimo najti takšno, ki predstavlja največjo razmaknjenost dveh razredov.
- Izberemo takšno, ki ima največjo razdaljo do najbližjih primerov na vsaki strani.

Support Vector Machines
Splošno

- p-dimenzionalen vektor z dodanim razredom.
- razred je binaren.
- iščemo takšno $(p-1)$-dimenzionalno hiperravnino, da bodo negativni primeri ležali na eni strani in pozitivni na drugi strani hiperravnine.
- Ker jih obstaja več, želimo najti takšno, ki predstavlja največjo razmaknjenost dveh razredov.
- Izberemo takšno, ki ima največjo razdaljo do najbližjih primerov na vsaki strani.

Support Vector Machines
Splošno



Support Vector Machines
Formalno

$$\mathfrak{D} = \{(x_i, y_i) | x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\}_{i=1}^n$$

- Iščemo maximal-margin hiperravnino, ki razdeli negativne in pozitivne primere.

Support Vector Machines

Formalno

$$\mathfrak{D} = \{(x_i, y_i) | x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\}_{i=1}^{n}$$

- Iščemo maximal-margin hiperravnino, ki razdeli negativne in pozitivne primere.

Pregled

Support Vector Machines

Izpeljava

- Vsako hiperravnino lahko zapišemo kot množico točk $\vec{x}$, ki veljajo za enačbo $\vec{w} \cdot \vec{x} - b = 0$
- $\cdot$ je skalarni produkt, vektor $\vec{w}$ je normalni vektor na ravnino, parameter $\frac{b}{||\vec{w}||}$ pa označuje oddaljenost hiperravnine od izhodišča v smeri vektorja $\vec{w}$.
- Želimo izbrati $\vec{w}$ in b, ki bosta maksimizirala razdaljo med vzporednima hiperravninama, ki sta kar se da narazen, pri čemer bosta še vedno delili primere. Ti dve ravnini lahko opišemo z:
  - $\vec{w} \cdot \vec{x} - b = 1$
  - $\vec{w} \cdot \vec{x} - b = -1$

Support Vector Machines

Izpeljava

- Vsako hiperravnino lahko zapišemo kot množico točk $\vec{x}$, ki veljajo za enačbo $\vec{w} \cdot \vec{x} - b = 0$
- $\cdot$ je skalarni produkt, vektor $\vec{w}$ je normalni vektor na ravnino, parameter $\frac{b}{||\vec{w}||}$ pa označuje oddaljenost hiperravnine od izhodišča v smeri vektorja $\vec{w}$.
- Želimo izbrati $\vec{w}$ in b, ki bosta maksimizirala razdaljo med vzporednima hiperravninama, ki sta kar se da narazen, pri čemer bosta še vedno delili primere. Ti dve ravnini lahko opišemo z:
  - $\vec{w} \cdot \vec{x} - b = 1$
  - $\vec{w} \cdot \vec{x} - b = -1$

Support Vector Machines

Izpeljava

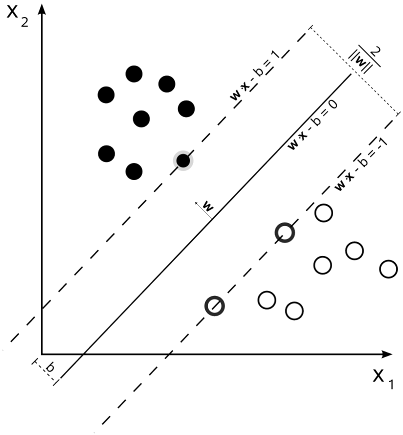
- Vsako hiperravnino lahko zapišemo kot množico točk $\vec{x}$, ki veljajo za enačbo $\vec{w} \cdot \vec{x} - b = 0$
- $\cdot$ je skalarni produkt, vektor $\vec{w}$ je normalni vektor na ravnino, parameter $\frac{b}{||\vec{w}||}$ pa označuje oddaljenost hiperravnine od izhodišča v smeri vektorja $\vec{w}$.
- Želimo izbrati $\vec{w}$ in b, ki bosta maksimizirala razdaljo med vzporednima hiperravninama, ki sta kar se da narazen, pri čemer bosta še vedno delili primere. Ti dve ravnini lahko opišemo z:
  - $\vec{w} \cdot \vec{x} - b = 1$
  - $\vec{w} \cdot \vec{x} - b = -1$

Support Vector Machines

Izpeljava

- Vsako hiperravnino lahko zapišemo kot množico točk $\vec{x}$, ki veljajo za enačbo $\vec{w} \cdot \vec{x} - b = 0$
- $\cdot$ je skalarni produkt, vektor $\vec{w}$ je normalni vektor na ravnino, parameter $\frac{b}{||\vec{w}||}$ pa označuje oddaljenost hiperravnine od izhodišča v smeri vektorja $\vec{w}$.
- Želimo izbrati $\vec{w}$ in b, ki bosta maksimizirala razdaljo med vzporednima hiperravninama, ki sta kar se da narazen, pri čemer bosta še vedno delili primere. Ti dve ravnini lahko opišemo z:
  - $\vec{w} \cdot \vec{x} - b = 1$
  - $\vec{w} \cdot \vec{x} - b = -1$

Support Vector Machines
Izpeljava



Support Vector Machines

Izpeljava

- Če so podatki linearno ločljivi, potem lahko izberemo hiperravnini tako, da med njima ni nobenih primerov. Nato pa maksimiziramo njuno razdaljo.
- Z uporabo geometrije vidimo, da je razdalja med hiperravninama $\frac{2}{||\vec{w}||}$, torej želimo minimizirati $||\vec{w}||$.
- Poleg tega želimo preprečiti, da bi kakšen primer padel v prostor med hiperravninama, zato nastavimo še pogoja:
  - $\vec{w} \cdot \vec{x_i} - b \geq 1$, za $\forall \vec{x_i}$ iz pozitivnega razreda.
  - $\vec{w} \cdot \vec{x_i} - b \leq -1$, za $\forall \vec{x_i}$ iz negativnega razreda.
- Zgornji pogoj lahko zapišemo kar enotno:
  $\forall i = 1 \ldots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$

Support Vector Machines
Izpeljava

- Če so podatki linearno ločljivi, potem lahko izberemo hiperravnini tako, da med njima ni nobenih primerov. Nato pa maksimiziramo njuno razdaljo.
- Z uporabo geometrije vidimo, da je razdalja med hiperravninama $\frac{2}{||\vec{w}||}$, torej želimo minimizirati $||\vec{w}||$.
- Poleg tega želimo preprečiti, da bi kakšen primer padel v prostor med hiperravninama, zato nastavimo še pogoja:
  - $\vec{w} \cdot \vec{x_i} - b \geq 1$, za $\forall \vec{x_i}$ iz pozitivnega razreda.
  - $\vec{w} \cdot \vec{x_i} - b \leq -1$, za $\forall \vec{x_i}$ iz negativnega razreda.
- Zgornji pogoj lahko zapišemo kar enotno: $\forall i = 1 \ldots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$

Support Vector Machines

Izpeljava

- Če so podatki linearno ločljivi, potem lahko izberemo hiperravnini tako, da med njima ni nobenih primerov. Nato pa maksimiziramo njuno razdaljo.
- Z uporabo geometrije vidimo, da je razdalja med hiperravninama $\frac{2}{||\vec{w}||}$, torej želimo minimizirati $||\vec{w}||$.
- Poleg tega želimo preprečiti, da bi kakšen primer padel v prostor med hiperravninama, zato nastavimo še pogoja:
  - $\vec{w} \cdot \vec{x_i} - b \geq 1$, za $\forall \vec{x_i}$ iz pozitivnega razreda.
  - $\vec{w} \cdot \vec{x_i} - b \leq -1$, za $\forall \vec{x_i}$ iz negativnega razreda.
- Zgornji pogoj lahko zapišemo kar enotno: $\forall i = 1 \dots n: \ y_i(\vec{w} \cdot \vec{x} - b) \geq 1$

Support Vector Machines

Izpeljava

- Če so podatki linearno ločljivi, potem lahko izberemo hiperravnini tako, da med njima ni nobenih primerov. Nato pa maksimiziramo njuno razdaljo.
- Z uporabo geometrije vidimo, da je razdalja med hiperravninama $\frac{2}{||\vec{w}||}$, torej želimo minimizirati $||\vec{w}||$.
- Poleg tega želimo preprečiti, da bi kakšen primer padel v prostor med hiperravninama, zato nastavimo še pogoja:
  - $\vec{w} \cdot \vec{x}_i - b \geq 1$, za $\forall \vec{x}_i$ iz pozitivnega razreda.
  - $\vec{w} \cdot \vec{x}_i - b \leq -1$, za $\forall \vec{x}_i$ iz negativnega razreda.
- Zgornji pogoj lahko zapišemo kar enotno:
  $\forall i = 1 \ldots n : \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$

Support Vector Machines

Izpeljava

Nastavimo lahko torej optimizacijski problem:

- Minimiziraj $||\vec{w}||$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \ldots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- Pojavi se problem, saj za izračun $||\vec{w}||$ potrebujemo kvadratni koren, zato optimizacijski problem malo spremenimo.
- Minimiziraj $\frac{1}{2}||\vec{w}||^2$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \ldots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- To je optimizacijski problem, ki ga rešujemo s pomočjo orodji za reševanje problemov iz kvadratičnega programiranja.

Support Vector Machines

Izpeljava

Nastavimo lahko torej optimizacijski problem:

- Minimiziraj $||\vec{w}||$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \ldots n : \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- Pojavi se problem, saj za izračun $||\vec{w}||$ potrebujemo kvadratni koren, zato optimizacijski problem malo spremenimo.
- Minimiziraj $\frac{1}{2}||\vec{w}||^2$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \ldots n : \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- To je optimizacijski problem, ki ga rešujemo s pomočjo orodji za reševanje problemov iz kvadratičnega programiranja.

Support Vector Machines
Izpeljava

Nastavimo lahko torej optimizacijski problem:

- Minimiziraj $||\vec{w}||$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \dots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- Pojavi se problem, saj za izračun $||\vec{w}||$ potrebujemo kvadratni koren, zato optimizacijski problem malo spremenimo.
- Minimiziraj $\frac{1}{2}||\vec{w}||^2$, s spremenljivkama $\vec{w}$ in b pri pogojih $\forall i = 1 \dots n: \; y_i(\vec{w} \cdot \vec{x} - b) \geq 1$
- To je optimizacijski problem, ki ga rešujemo s pomočjo orodji za reševanje problemov iz kvadratičnega programiranja.

Support Vector Machines

Izpeljava

Mogoče bi lahko nastavili optimizacijski problem z Lagrange-vimi multiplikatorji α_i:

$$min_{\vec{w},b,\vec{\alpha}}\{\frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i[y_i(\vec{w}\cdot\vec{x} - b) - 1]\}$$

TEŽAVA!!

Support Vector Machines
Izpeljava

Mogoče bi lahko nastavili optimizacijski problem z Lagrange-vimi multiplikatorji α_i:

$$min_{\vec{w},b,\vec{\alpha}}\{\frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n}\alpha_i[y_i(\vec{w}\cdot\vec{x} - b) - 1]\}$$

TEŽAVA!!

Support Vector Machines

Izpeljava

Mogoče bi lahko nastavili optimizacijski problem z Lagrange-vimi multiplikatorji α_i:

$$min_{\vec{w},b,\vec{\alpha}}\{\frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n}\alpha_i[y_i(\vec{w}\cdot\vec{x} - b) - 1]\}$$

TEŽAVA!!

Support Vector Machines

Izpeljava

Vseeno lahko prej opisan problem rešujemo na podoben način:

$$min_{\vec{w},b} \max_{\vec{\alpha}}\{\frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i[y_i(\vec{w} \cdot \vec{x} - b) - 1]\}$$

- Iščemo sedlo v tem optimizacijskem problemu. S tem vse točke, ki jih lahko ločimo ($y_i(\vec{w} \cdot \vec{x} - b) - 1 > 0$), niso pomembne, saj moramo pripadajoče α_i nastaviti na nič.
- Problem lahko spet rešujemo s standardnimi postopki za kvadratično programiranje.
- Rešitev lahko tako zapišemo kot linearno kombinacijo učnih vektorjev: $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$.

Support Vector Machines
Izpeljava

Vseeno lahko prej opisan problem rešujemo na podoben način:

$$min_{\vec{w},b} \max_{\vec{\alpha}} \{ \frac{1}{2} ||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i [y_i(\vec{w} \cdot \vec{x} - b) - 1] \}$$

- Iščemo sedlo v tem optimizacijskem problemu. S tem vse točke, ki jih lahko ločimo ($y_i(\vec{w} \cdot \vec{x} - b) - 1 > 0$), niso pomembne, saj moramo pripadajoče α_i nastaviti na nič.
- Problem lahko spet rešujemo s standardnimi postopki za kvadratično programiranje.
- Rešitev lahko tako zapišemo kot linearno kombinacijo učnih vektorjev: $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$.

Support Vector Machines

Izpeljava

Vseeno lahko prej opisan problem rešujemo na podoben način:

$$min_{\vec{w},b} \max_{\vec{\alpha}} \{ \frac{1}{2} ||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i [y_i(\vec{w} \cdot \vec{x} - b) - 1] \}$$

- Iščemo sedlo v tem optimizacijskem problemu. S tem vse točke, ki jih lahko ločimo ($y_i(\vec{w} \cdot \vec{x} - b) - 1 > 0$), niso pomembne, saj moramo pripadajoče α_i nastaviti na nič.
- Problem lahko spet rešujemo s standardnimi postopki za kvadratično programiranje.
- Rešitev lahko tako zapišemo kot linearno kombinacijo učnih vektorjev: $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$.

Support Vector Machines

Izpeljava

Vseeno lahko prej opisan problem rešujemo na podoben način:

$$min_{\vec{w},b} \max_{\vec{\alpha}} \{ \frac{1}{2} ||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i [y_i(\vec{w} \cdot \vec{x} - b) - 1] \}$$

- Iščemo sedlo v tem optimizacijskem problemu. S tem vse točke, ki jih lahko ločimo ($y_i(\vec{w} \cdot \vec{x} - b) - 1 > 0$), niso pomembne, saj moramo pripadajoče α_i nastaviti na nič.
- Problem lahko spet rešujemo s standardnimi postopki za kvadratično programiranje.
- Rešitev lahko tako zapišemo kot linearno kombinacijo učnih vektorjev: $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$.

Support Vector Machines

Izpeljava

Le nekaj členov α_i bo večjih od 0. Ravno pripadajočim $\vec{x}_i$ pravimo, da so podporni vektorji, saj oni ravno ležijo na meji in za njih velja $y_i(\vec{w} \cdot \vec{x} - b) = 1$.

Od tod lahko izpeljemo tudi formulo za $b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (\vec{w} \cdot \vec{x}_i - y_i)$

Support Vector Machines

Izpeljava

Le nekaj členov α_i bo večjih od 0. Ravno pripadajočim $\vec{x}_i$ pravimo, da so podporni vektorji, saj oni ravno ležijo na meji in za njih velja $y_i(\vec{w} \cdot \vec{x} - b) = 1$.

Od tod lahko izpeljemo tudi formulo za $b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (\vec{w} \cdot \vec{x}_i - y_i)$

Support Vector Machines
Dualni problem

Dualni problem nam pokaže, da je iskana hiperravnina in tudi problem klasifikacije funkcija le podpornih vektorjev.

- Iz dejstva $||\vec{w}||^2 = \vec{w} \cdot \vec{w}$ in z zamenjavo $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$, lahko izpeljemo spodnji optimizacijski problem:
- $L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\vec{x}_i, \vec{x}_j)$
- pri pogojih $\alpha_i \geq 0$ za $i = 1, \ldots n$ in $\sum_{i=1}^{n} \alpha_i y_i = 0$

Support Vector Machines
Dualni problem

Dualni problem nam pokaže, da je iskana hiperravnina in tudi problem klasifikacije funkcija le podpornih vektorjev.

- Iz dejstva $||\vec{w}||^2 = \vec{w} \cdot \vec{w}$ in z zamenjavo $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$, lahko izpeljemo spodnji optimizacijski problem:
- $L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \vec{x}_i \cdot \vec{x}_j = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\vec{x}_i, \vec{x}_j)$
- pri pogojih $\alpha_i \geq 0$ za $i = 1, \ldots n$ in $\sum_{i=1}^{n} \alpha_i y_i = 0$

Pregled

Support Vector Machines
Nelinearna klasifikacija

- Vsak skalarni produkt nadomestimo z funkcijo jedra (kernel), ki ni linearna.
- To dovoljuje algoritmu, da išče hiperravnine v transformiranem prostoru primerov.
- Primeri nelinearnih jeder:
  - Polinomsko (homogeno): $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j)^d$
  - Polinomsko (nehomogeno): $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j + 1)^d$
  - Hiperbolični tangens: $K(\vec{x}_i, \vec{x}_j) = tanh(k\vec{x}_i \cdot \vec{x}_j + c)$, za nekatere $k > 0$ in $c < 0$.

Support Vector Machines

Nelinearna klasifikacija

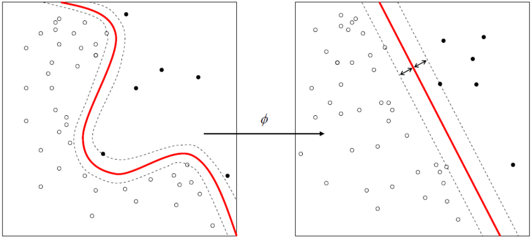
- Vsak skalarni produkt nadomestimo z funkcijo jedra (kernel), ki ni linearna.
- To dovoljuje algoritmu, da išče hiperravnine v transformiranem prostoru primerov.
- Primeri nelinearnih jeder:
  - Polinomsko (homogeno): $K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j})^d$
  - Polinomsko (nehomogeno): $K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j} + 1)^d$
  - Hiperbolični tangens: $K(\vec{x_i}, \vec{x_j}) = tanh(k\vec{x_i} \cdot \vec{x_j} + c)$, za nekatere $k > 0$ in $c < 0$.

Support Vector Machines

Nelinearna klasifikacija

- Vsak skalarni produkt nadomestimo z funkcijo jedra (kernel), ki ni linearna.
- To dovoljuje algoritmu, da išče hiperravnine v transformiranem prostoru primerov.
- Primeri nelinearnih jeder:
    - Polinomsko (homogeno): $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j)^d$
    - Polinomsko (nehomogeno): $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j + 1)^d$
    - Hiperbolični tangens: $K(\vec{x}_i, \vec{x}_j) = tanh(k\vec{x}_i \cdot \vec{x}_j + c)$, za nekatere $k > 0$ in $c < 0$.

Support Vector Machines
Nelinearna klasifikacija



http://www.youtube.com/watch?v=3liCbRZPrZA