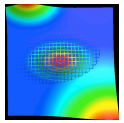
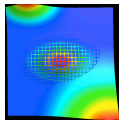


Nagy adathalmazok labor

2018-2019 őszi félév

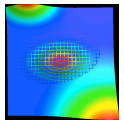
2018.09.12-13

1. Kiértékelés
2. Döntési fák
3. Ipython notebook



Féléves terv

- Kiértékelés: cross-validation, bias-variance trade-off
- Supervised learning (discr. és generative modellek): nearest neighbour, decision tree (döntési fák), logisztikus regresszió, nem lineáris osztályozás, neurális hálózatok, support vector machine, idős osztályozás és dynamic time warping
- Lineáris, polinomiális és sokdimenziós regresszió és optimalizálás
- Tree ensembles: AdaBoost, GBM, Random Forest
- Clustering (klaszterezés): k-means, sűrűség és hierarchikus modellek
- Principal component analysis (PCA, SVD), collaborative filtering, ajánlórendszerek
- Anomália keresés
- Gyakori termékhalmazok
- Szublineáris algoritmusok: streaming, Johnson-Lindenstrauss, count-min sketch



Következő hetek

Szeptember 12: bevezetés, osztályozás-klaszterezés, kiértékelés, kNN, DT

Szeptember 13: gyakorlat, ipython notebook (akinek van kérem hozzon laptopot), NN

Szeptember 19: Naive Bayes, logisztikus regresszió

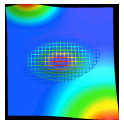
Szeptember 26: logisztikus regresszió, generalizáció

Szeptember 27: gyakorlat, ipython, LR

Október 3: polinomiális szeparálás, SVM

Október 10: ANN, MLP, CNN

Október 11: gyakorlat

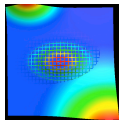


Gépi tanulás (machine learning)

Legyen adott egy véges, valós halmaz $X=\{x_1,..,x_T\}$ ahol minden x_i R^d-beli és minden elemhez legyen adott egy célváltozó $y=\{y_1,..,y_T\}$ általában {-1,1}. A bináris osztályozás feladata egy olyan f(x) függvény keresése amely jól közeliti az y-t X-en.

Hogyan mérhető a közelités jósága?
Mi alapján válasszunk függvényosztályt?
Hogyan találjuk meg a megfelelő elemet a függvényosztályon belül?
Mennyire generalizált a választott modellünk?

Osztályozás vagy regresszió? p(x) vs. p(y | x)?

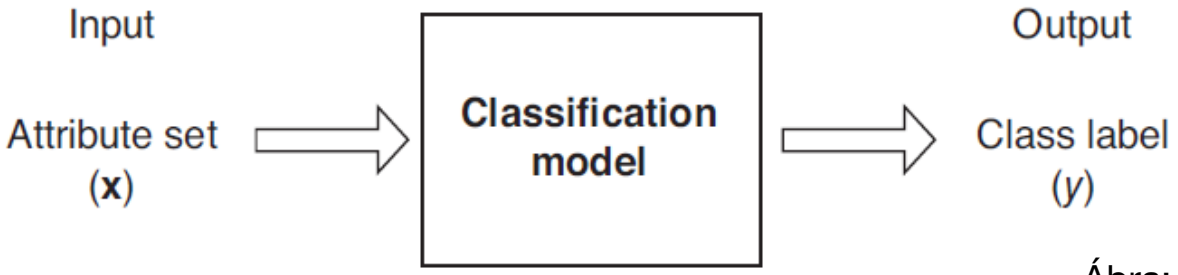


Emlékeztető: Osztályozás

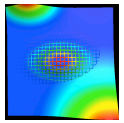
Pl. lineráris szeparálás. Keressünk egy olyan d dimenziós hipersikot amely

$$w \cdot x_i > b \text{ minden } x_i \text{ -re melynek cimkéje } +1$$
$$w \cdot x_i < b \text{ minden } x_i \text{ -re melynek cimkéje } -1$$

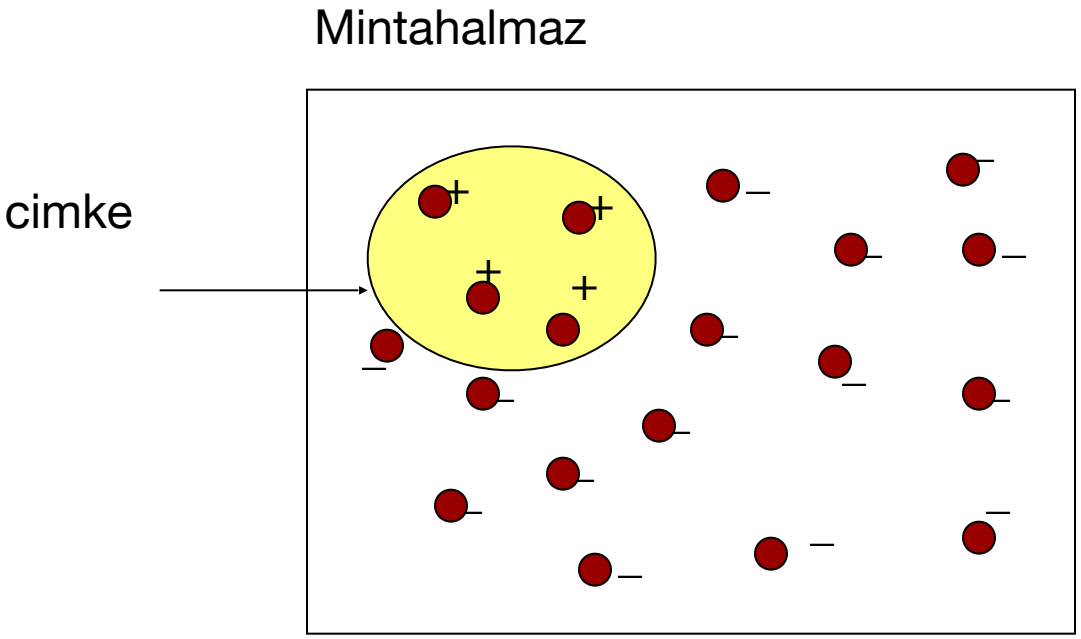
Ha egy pár (w, b) amely kielégiti a fenti egyenlőtlenségeket, akkor lineáris szeparátornak nevezzük. Létezik?



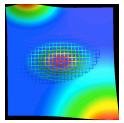
Ábra: TSK



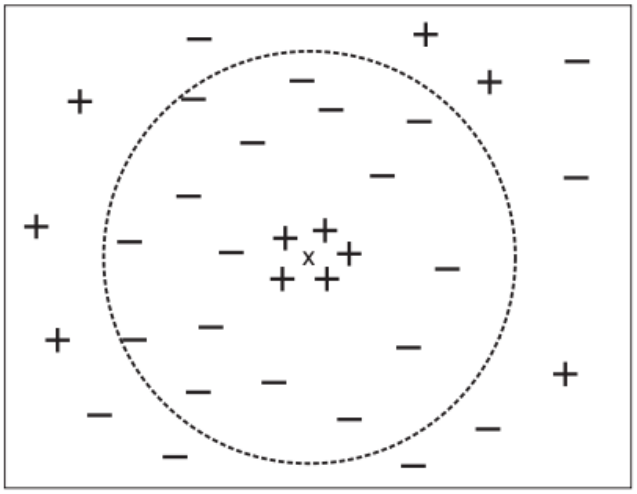
Emlékeztető: osztályozás



Ábra: TSK



K- nearest neighbor (K-NN)



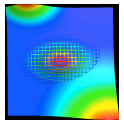
Mi a gond?

- mohó (eager) algoritmus: kész modell készítése amint a tanuló adat létezik → többé nincs szükség a tanulóhalmazra
- lusta (lazy) algoritmus: kész jóslatot a tanulóhalmaz segítségével készít → szükséges a tanulóhalmaz a jósláshoz

A K-NN lusta algoritmus :

Méret?
Bonyolultság?
Skálázhatóság?

Ábra: TSK



K- nearest neighbor (K-NN)

Távolság mértékek (pl.)

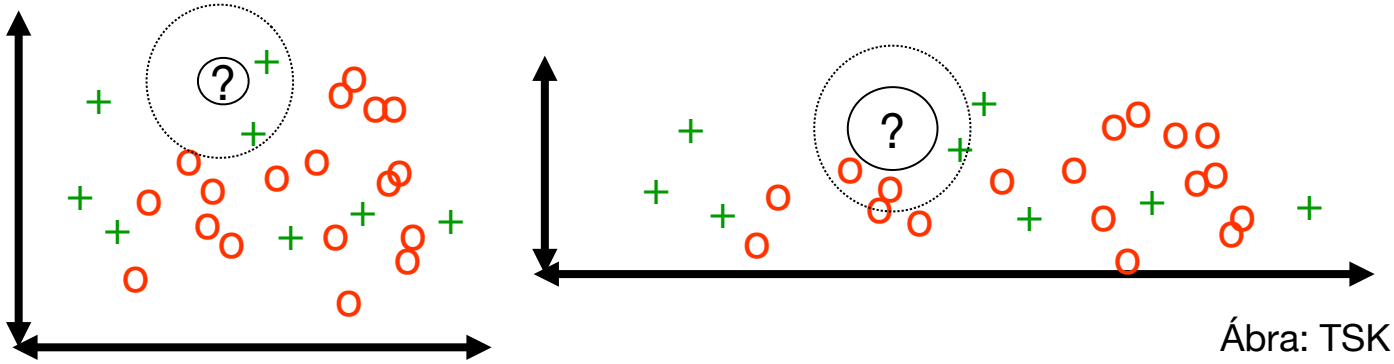
- Minkowski távolság

- Mahalanobis $D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}.$

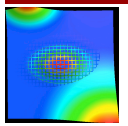
- Cosine, Jaccard, Kullback-Leibler, Jensen-Shannon, DTW stb.

Problémák:

- skála
- autómatikus távolság választás
- normalizáció



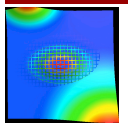
Ábra: TSK



Kiértékelés

Confusion matrix:

Alapigazság/ predikció	p	n	Total
p	True Positive (TP)	False Negative (FN)	TP+FN
n	False Positive (FP)	True Negative (TN)	FP+TN
Total	TP+FN	FP+TN	



Kiértékelés

Accuracy: a helyesen klasszifikálás valószínűsége

TP+TN/(TP+FP+TN+FN)

Precision (p): egy releváns dokumentum helyes klasszifikálásának valószínűsége

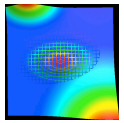
TP/(TP+FP)

Recall (r): annak a valószínüsége, hogy egy releváns dokumentumot helyesen klasszifikálunk

TP/(TP+FN)

F-measure: a precision és a recall harmónikus közepe

(2*p*r/(p+r))



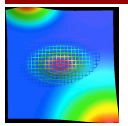
Órai feladat 1: kiértékelés

Melyik mérték fejezi ki jobban a céljainkat a következő esetekben, s miért?

Egészségügyi szűrést végzünk. Szeretnénk minél több fertőzött pácienst megtalálni, hogy megkezdhessük a kezelésüket.

Ajánlórendszert készítünk. Minden felhasználó számára szeretnénk öt könyvet javasolni.

Képek annotálása



Kiértékelés

False-Positive Rate (FPR) =
FP/(FP+TN)

True-Positive Rate (TPR) =
TP/(TP+FN)

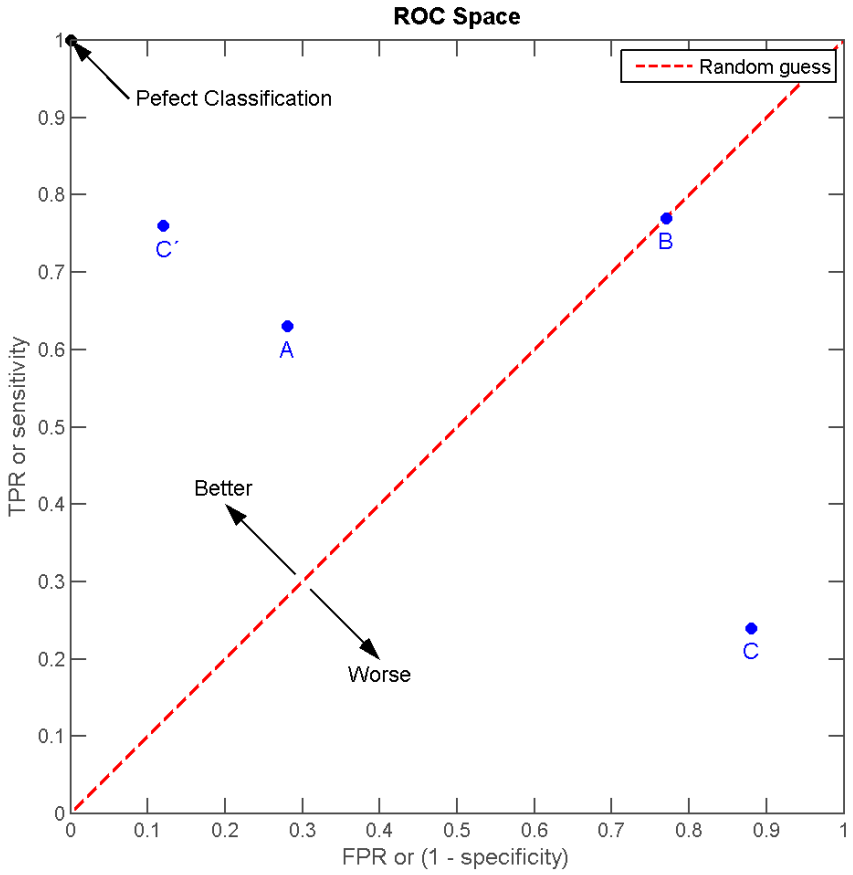
ROC: Receiver Operating Characteristic

MAP: Mean Average Precision

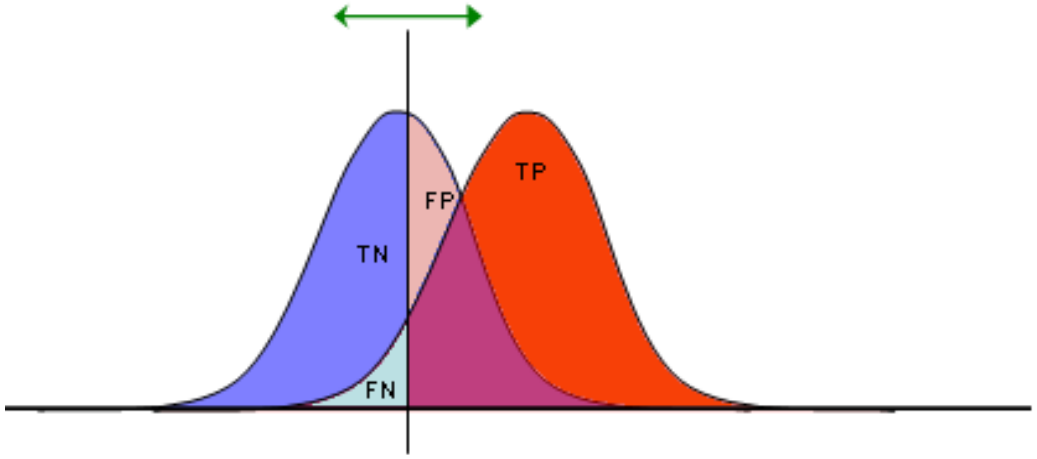
$$\text{AveP} = \frac{\sum_{r=1}^{N}(P(r) \times \text{rel}(r))}{\text{number of relevant documents}}$$

nDCG: normalized Discriminative Cummulative Gain

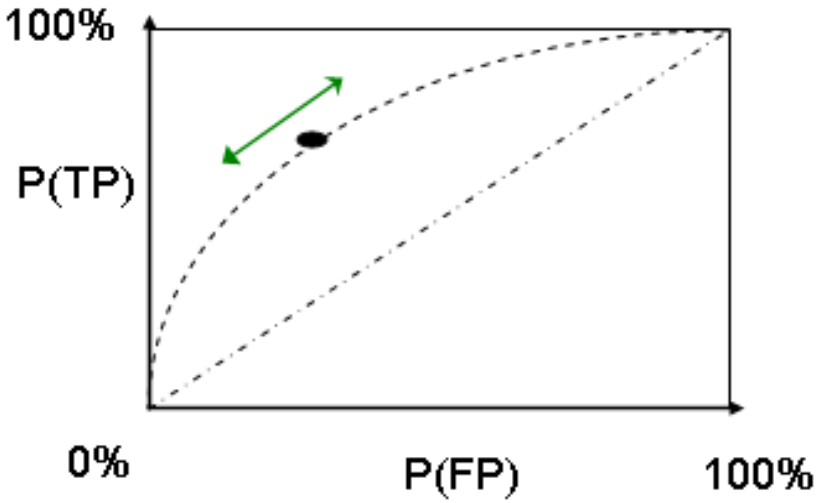
$$\text{DCG}_\text{p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$$

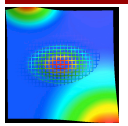


Kiértékelés



TP	FP
FN	TN
1	1





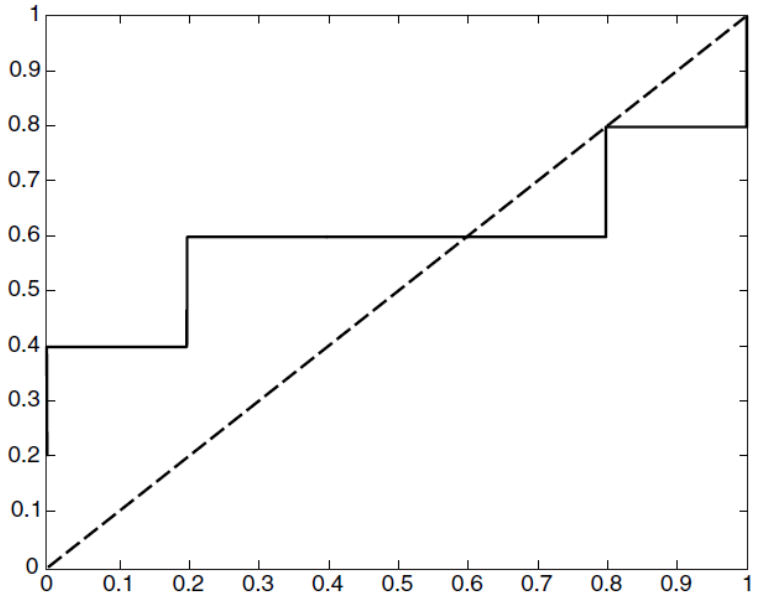
ROC: Receiver Operating Characteristic

- csak bináris osztályozásnál használható (i.e. egy osztályra)
- Area Under Curve: AUC annak a valószínűsége, hogy véletlen pozitív elemeket előrébb sorol mint véletlen negatívakat
- mivel az jóslatok sorrendjéből számítjuk, a klasszifikálónak nem csak bináris jóslatokat kell visszaadnia
- előnye, hogy nem függ a vágási ponttól
- Milyen szempontból lehetne továbbfejleszteni?

Class	+	−	+	−	−	−	+	−	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC: Receiver Operating Characteristic

Class	+	−	+	−	−	−	+	−	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0



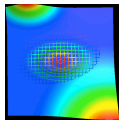
AUC=?

+	+	-	+	-	-	+	+	-	+	
0.16	0.32	0.42	0.44	0.45	0.51	0.78	0.87	0.91	0.93	Score
										TP
										FN
										TN
										FP
										TPR
										FPR

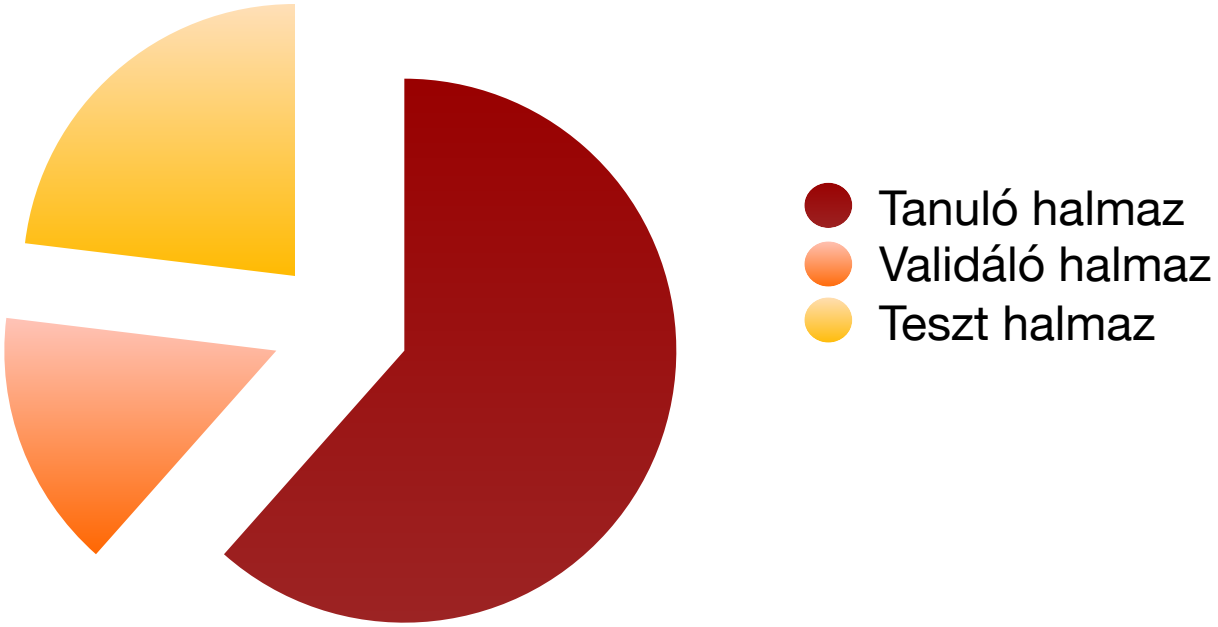
Órai feladat 2:

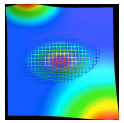
Mennyi az AUC?
(a nagyobb score-al rendelkező elemek előrébb kerülnek a találati listán)

+	+	-	-	-	+	+	-	+	+	
0.43	0.56	0.62	0.78	0.79	0.86	0.89	0.89	0.91	0.96	Score
										TP
										FN
										TN
										FP
										TPR
										FPR

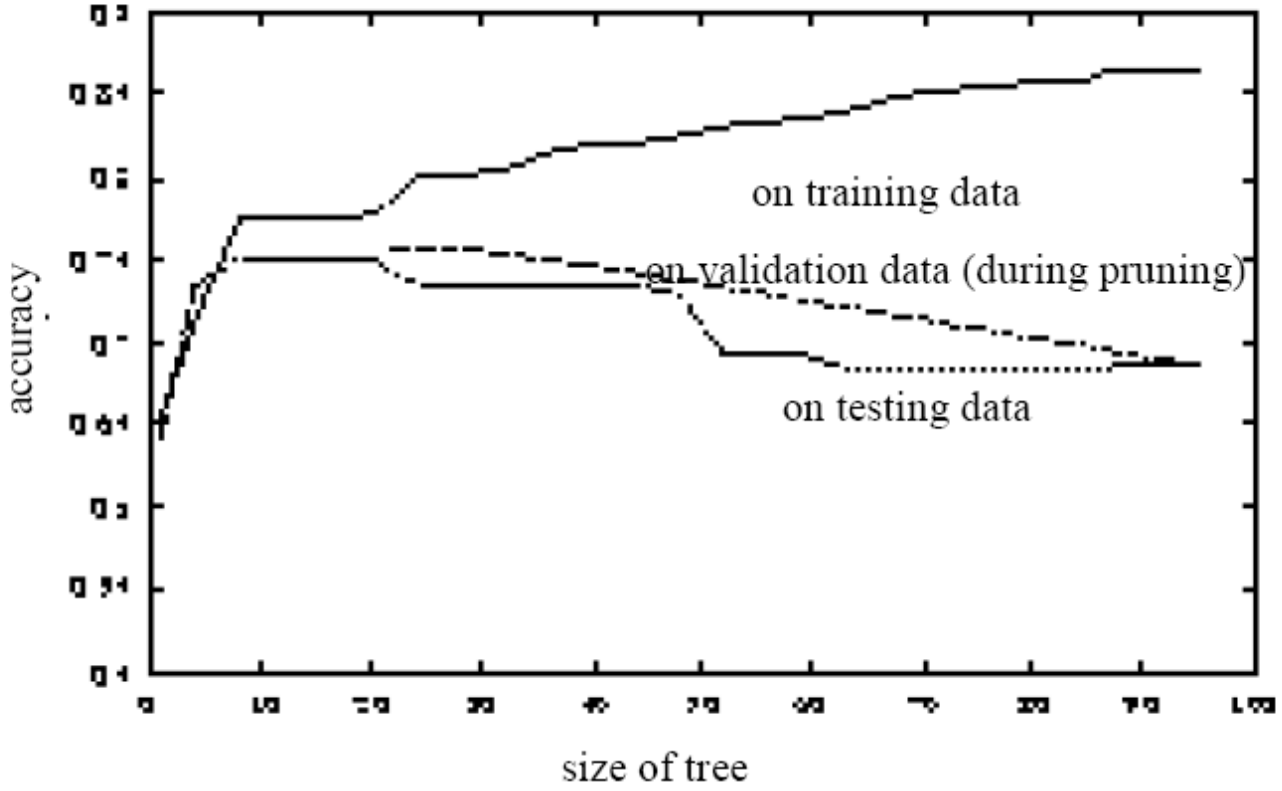


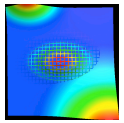
Validáció





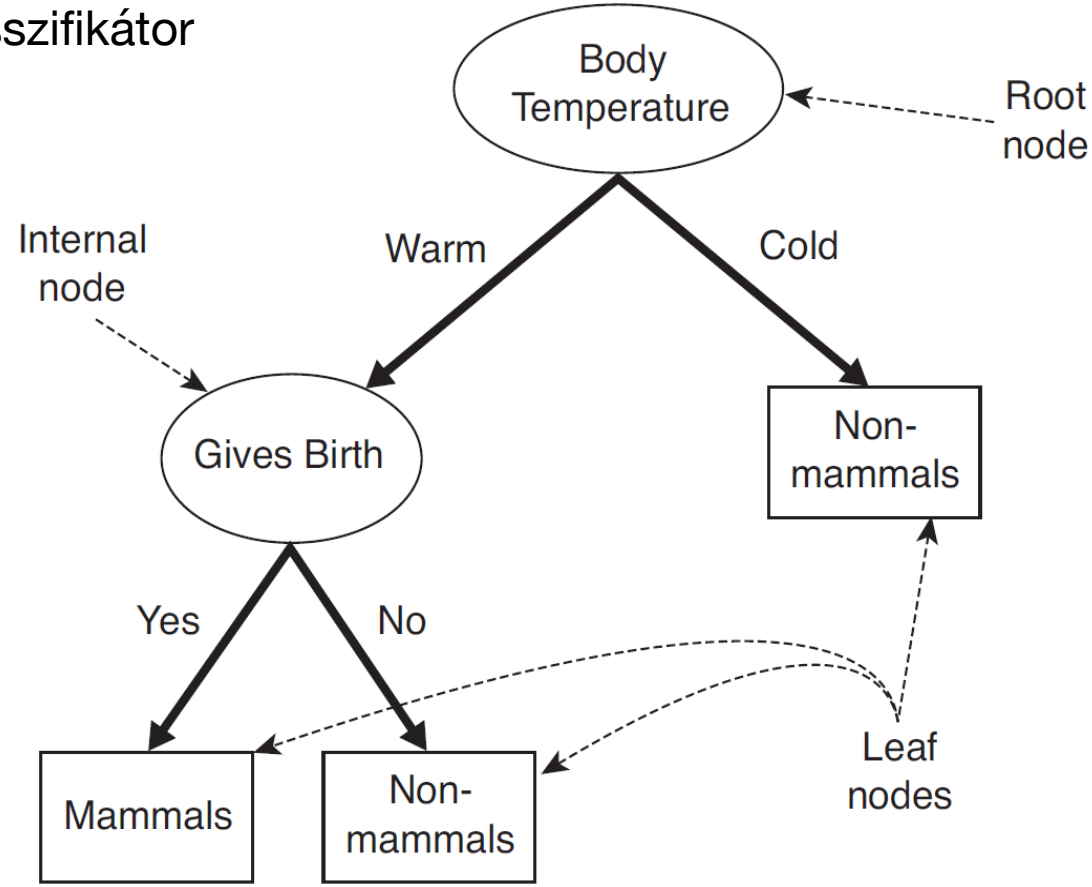
Validáció


accuracy
size of tree
on training data
on validation data (during pruning)
on testing data

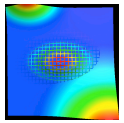


Döntési fák

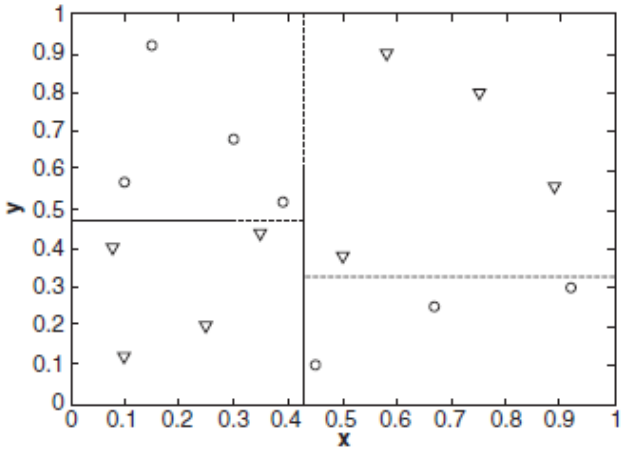
Pl. emlős klasszifikátor

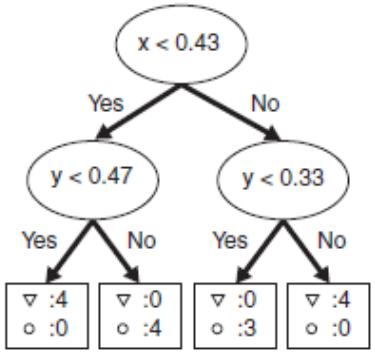


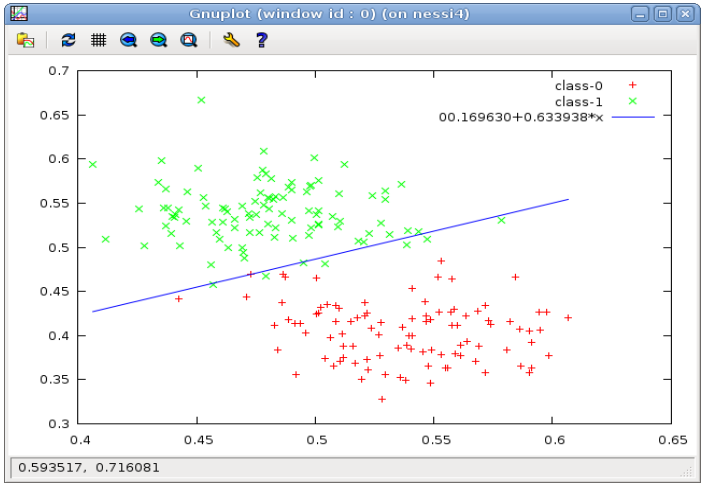
Ábra: TSK



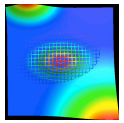
Döntési fák





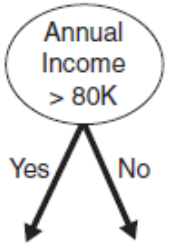


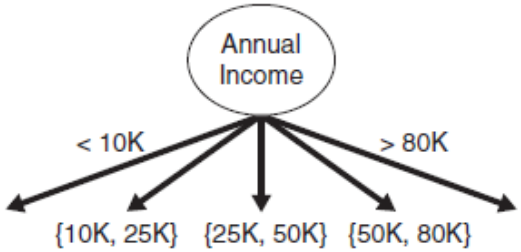
Megfelelő döntési fa?



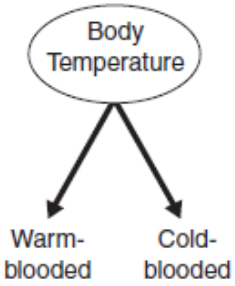
Döntési fák

Vágás attribútumok típusa szerint

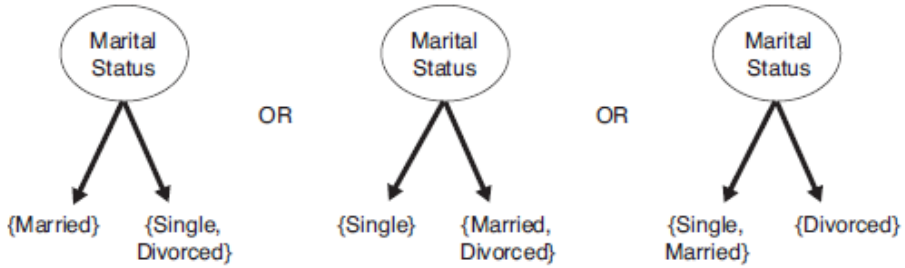




skála szerint

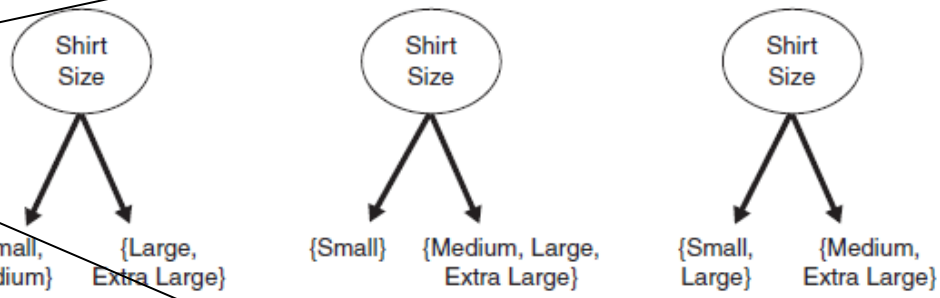


bináris

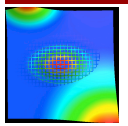


nominális

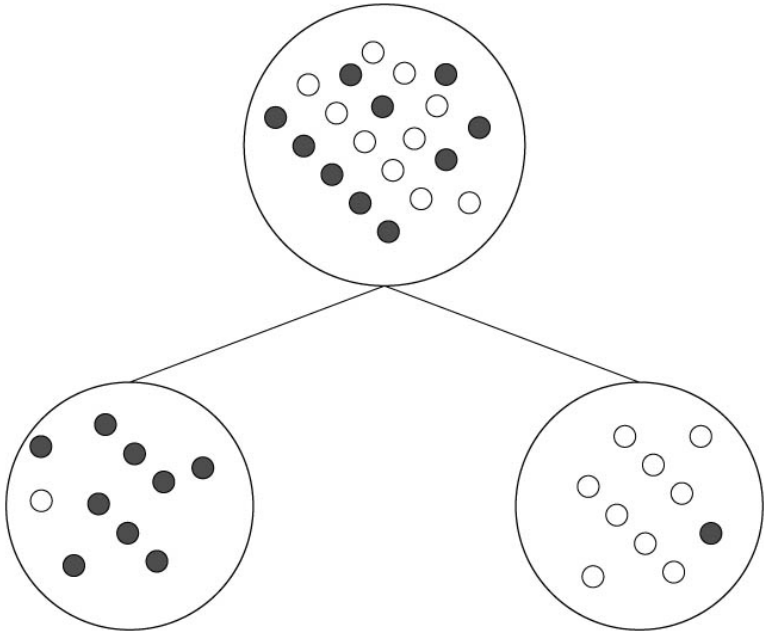
Különbség?



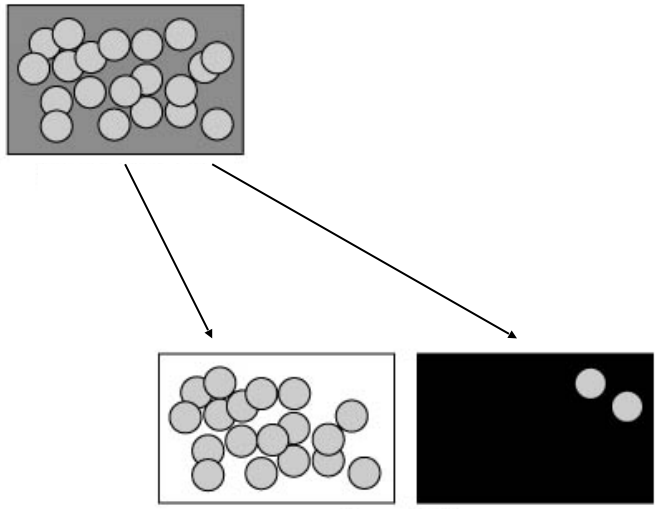
ordinális



Döntési fák

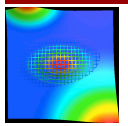


Jó vágás



Rossz vágás

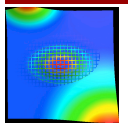
Ábra: TSK



Döntési fa építés (C4.5)

Feltétel: minden attribútum nominális

Procedure TreeBuilding (D)
    If (the data is not classified correctly)
        Find best splitting attribute A
            For each a element in A
            Create child node N_a
                D_a all instances in D where A=a
                TreeBuilding (D_a)
            Endfor
    Endif
EndProcedure



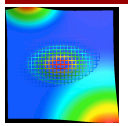
Döntési fák

Jó vágási attribútum:

- jól szeparálja az egyes osztályokhoz tartozó elemeket külön ágakra
  → növeli a tisztaságot (purity)
- kiegyensúlyozott

Purity mértékek lehetnek:

- klasszifikációs hiba
- entrópia
- gini
- vagy ami épp szükséges az adathoz



Döntési fák

Klasszifikációs hiba:

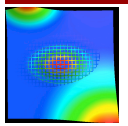
p(i,t) : egy adott t csomópontban az i osztályba tartozó elemek aránya

Classification error: 1-max(p(i,t))

Növekmény(gain):

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

Itt I(parent) az adott csomópont tisztasága, k a vágás után keletkező ágak száma, $N(v_j)$ az adott ágon található elemek száma, N a csomópontban található elemek száma, $I(v_j)$ pedig a j-dik ág tisztasága

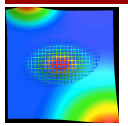


Döntési fák

Példa:

A vagy B attribútum szerint érdemes vágni, ha a tisztasági mérték a klasszifikációs hiba?

A	B	Class Label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−



Döntési fák

Példa:

A vagy B attribútum szerint érdemes vágni, ha a tisztasági mérték a klasszifikációs hiba?

A	B	Class Label
T	F	+
T	T	+
T	T	+
T	F	−
T	T	+
F	F	−
F	F	−
F	F	−
T	T	−
T	F	−

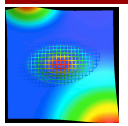
Vágás A alapján:

MCE = 0+3/7

Vágás B alapján:

MCE = 1/4+1/6

Vágjunk B szerint!



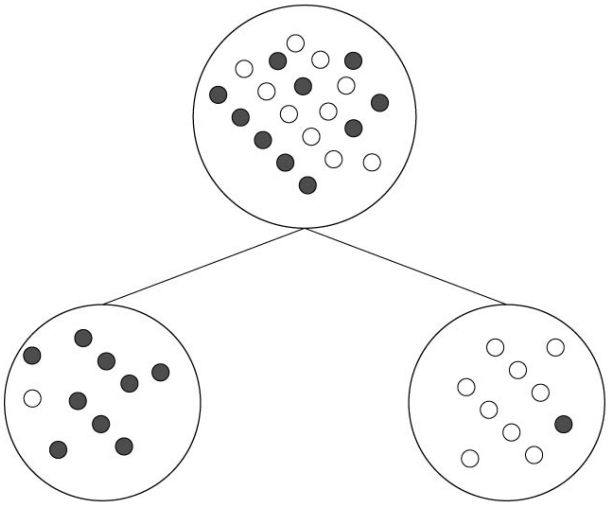
Döntési fák

Gini (population diversity)

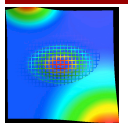
p(i|t) : egy adott t csomópontban az i osztályba tartozó elemek aránya

Gini:

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$



Gini a gyökérben?
Gini a leveleknél?



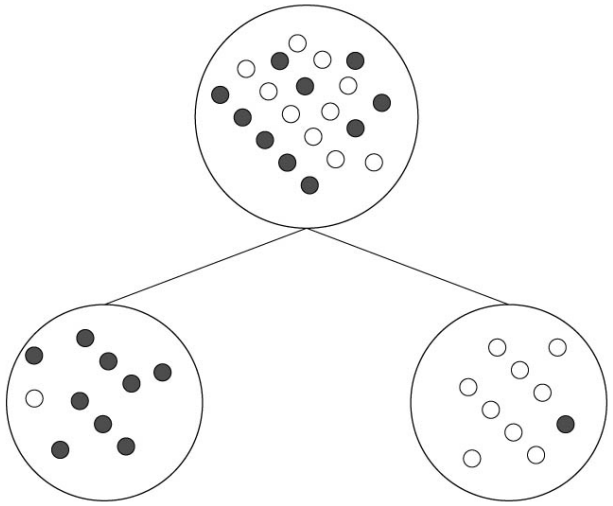
Döntési fák

Gini (population diversity)

p(i|t) : egy adott t csomópontban az i osztályba tartozó elemek aránya

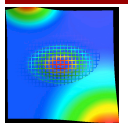
Gini:

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$



Gini(gyökér)= 0.5 = 0.5^2 + 0.5^2

Gini(level)= 0.82 = 0.1^2 + 0.9^2



Döntési fák

Entrópia (információ)

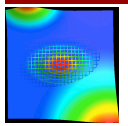
p(i|t) : egy adott t csomópontban az i osztályba tartozó elemek aránya

Entrópia:

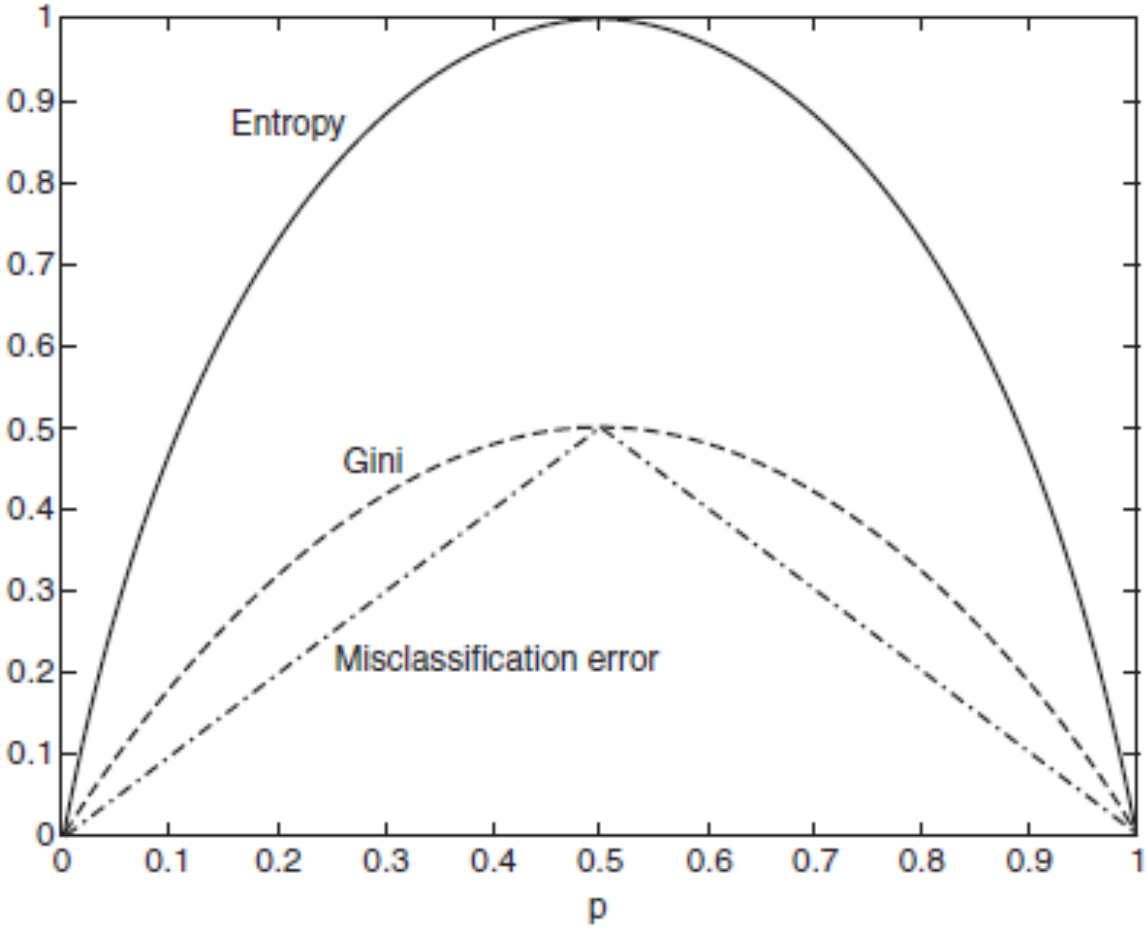
$$I(v_j) - \sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t)$$

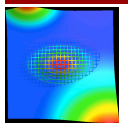
Mindegyik mérték közös tulajdonsága, hogy maximumukat 0.5-nél érik el. Mind a több részre szabdalást preferálja. (nem bináris attribútumoknál)

$$\text{Gain ratio} = \frac{\Delta_{\text{info}}}{-\sum_{i=1}^{k} P(v_i) \log_2 P(v_i)}$$



Döntési fák


Entropy
Gini
Misclassification error
p
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



Döntési fák

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

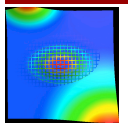
$$\text{Classification error}(t) = 1 - \max_i [p(i|t)],$$

Node N_1	Count
Class=0	0
Class=1	6

Node N_2	Count
Class=0	1
Class=1	5

Node N_3	Count
Class=0	3
Class=1	3

Mennyi az entrópia, klasszifikációs hiba és a gini?



Döntési fák

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1}[p(i|t)]^2,$$

$$\text{Classification error}(t) = 1 - \max_i[p(i|t)],$$

Node N_1	Count
Class=0	0
Class=1	6

$\text{Gini} = 1 - (0/6)^2 - (6/6)^2 = 0$
$\text{Entropy} = -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$
$\text{Error} = 1 - \max[0/6, 6/6] = 0$

Node N_2	Count
Class=0	1
Class=1	5

$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$
$\text{Entropy} = -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$
$\text{Error} = 1 - \max[1/6, 5/6] = 0.167$

Node N_3	Count
Class=0	3
Class=1	3

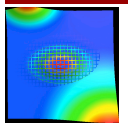
$\text{Gini} = 1 - (3/6)^2 - (3/6)^2 = 0.5$
$\text{Entropy} = -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$
$\text{Error} = 1 - \max[3/6, 3/6] = 0.5$

Pár direktíva:

- ha két fa hasonlóan teljesít mindig válasszuk a kevésbé komplexet
- csak adott threshold feletti javulás esetén vágjunk (early pruning)
- utólag összevonjuk azon leveleket, melyek a legkevesebb hibát okozzák, s a leggyakoribb osztályra döntünk (post-pruning)
- MDL(Minimum Description Length): azt a fát válasszuk, melynek a leírása kisebb

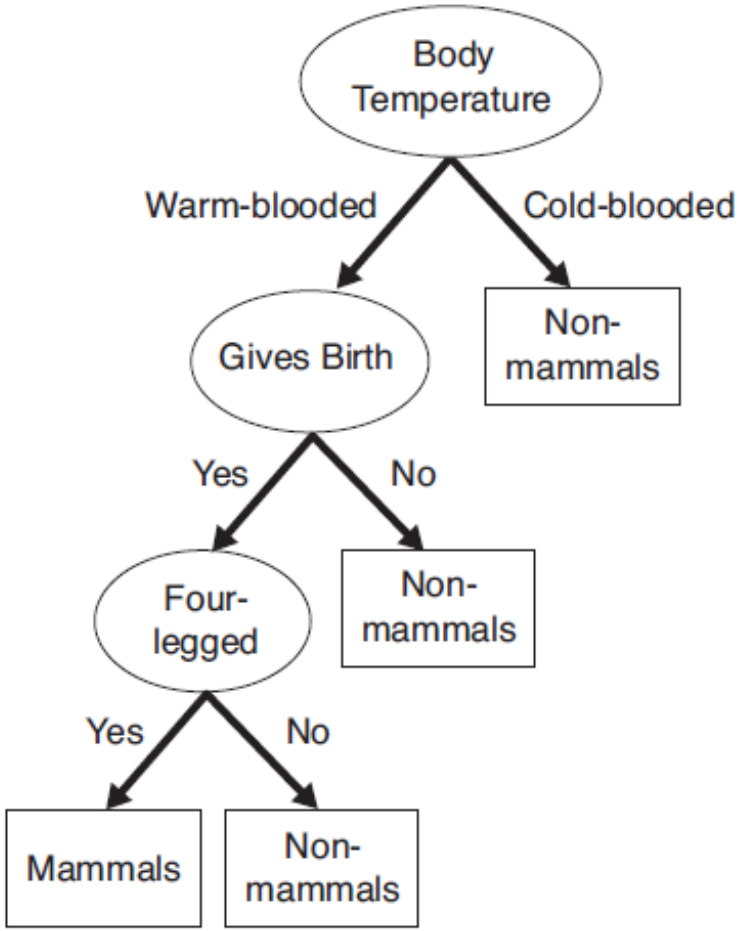
Példa hibára: mi lesz egy delfinnel? (tesztadat)

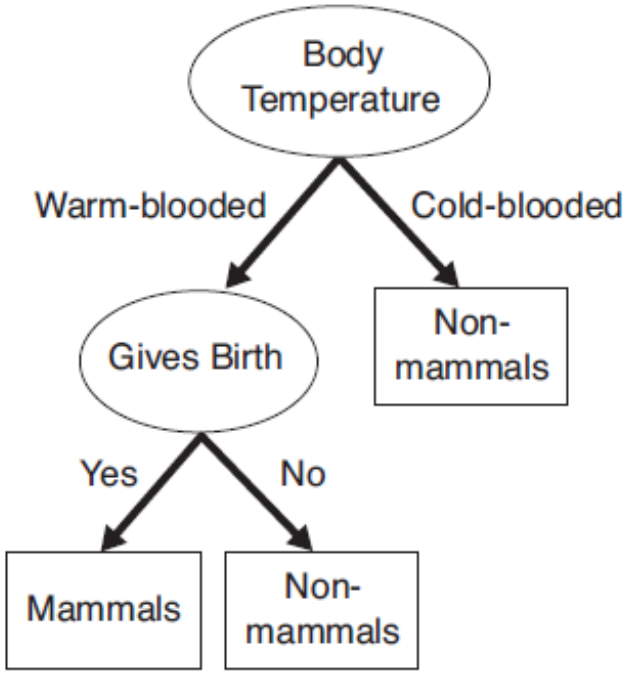
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

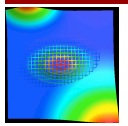


Döntési fák

Zaj?







Döntési fák

Megjegyzések:

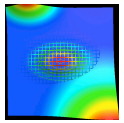
DT-k képesek kezelni mind nominális, mind pedig numerikus adatokat
(dátumok és string-ek?)
könnyen értelmezhetőek
Zajra robusztusak (?)
De a C4.5 algoritmus előállíthatja ugyanazt az alfát ("subtree") többször
Túltanulás ("overfitting")

Miért?

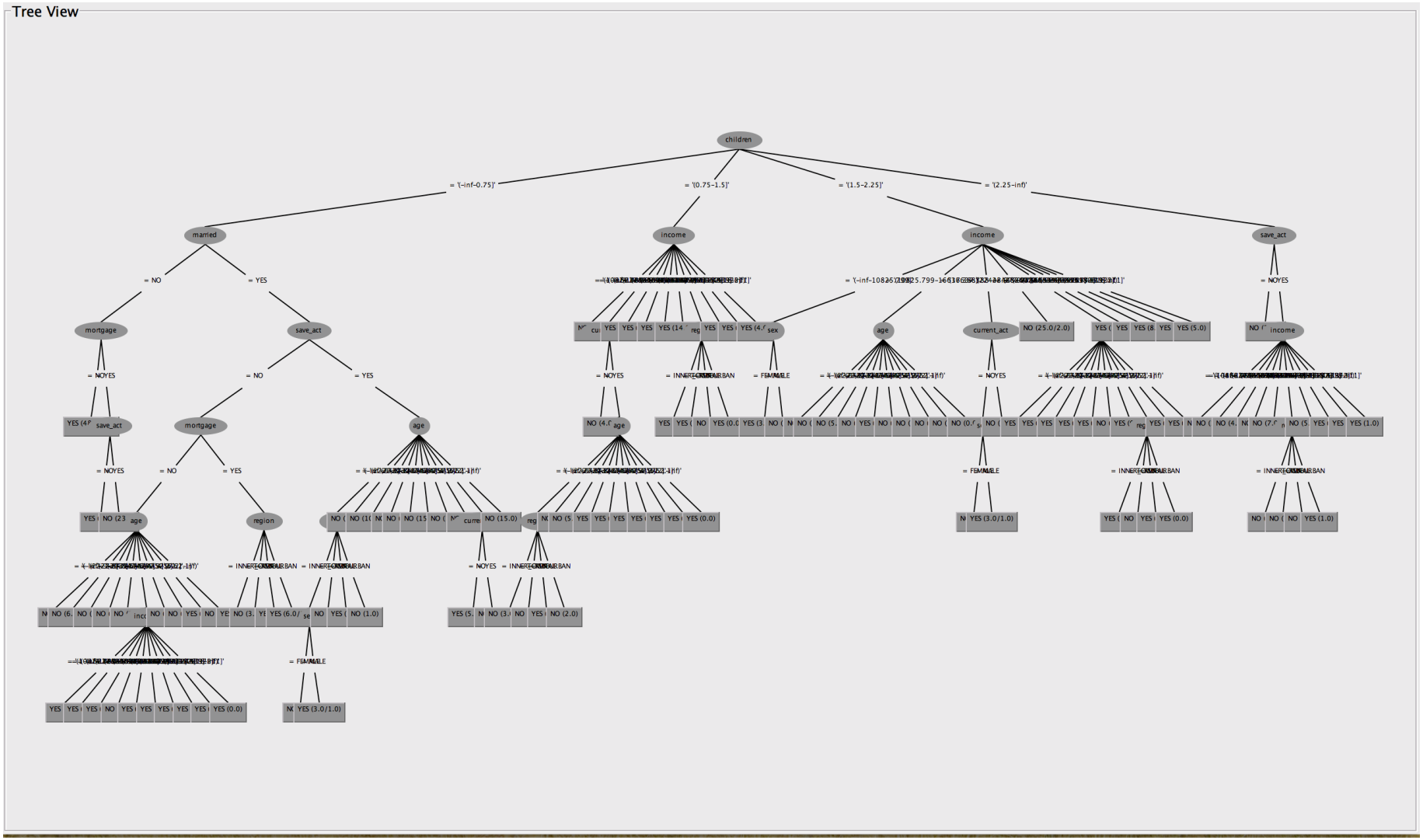
Jellemző okok:

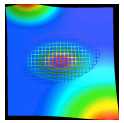
- túl mély és széles a DT kevés tanuló elemmel az egyes levelekben
- kiegyensúlyozatlan ("unbalanced") tanuló halmaz

Megoldás: "pruning"!



Unpruned tree





Subtree raising vs. replacement (rep)

Decision Tree:

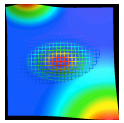
```
depth = 1:
|  breadth> 7 :  class 1
|  breadth<= 7:
|  |  breadth <= 3:
|  |  |  ImagePages> 0.375:  class 0
|  |  |  ImagePages<= 0.375:
|  |  |  |  totalPages<= 6:  class 1
|  |  |  |  totalPages> 6:
|  |  |  |  |  breadth <= 1:  class 1
|  |  |  |  |  breadth > 1:  class 0
|  |  width > 3:
|  |  |  MultiIP = 0:
|  |  |  |  ImagePages<= 0.1333:  class 1
|  |  |  |  ImagePages> 0.1333:
|  |  |  |  breadth <= 6:  class 0
|  |  |  |  breadth > 6:  class 1
|  |  |  MultiIP = 1:
|  |  |  |  TotalTime <= 361:  class 0
|  |  |  |  TotalTime > 361:  class 1
depth> 1:
|  MultiAgent = 0:
|  |  depth > 2:  class 0
|  |  depth <= 2:
|  |  |  MultiIP = 1:  class 0
|  |  |  MultiIP = 0:
|  |  |  |  breadth <= 6:  class 0
|  |  |  |  breadth > 6:
|  |  |  |  |  RepeatedAccess <= 0.322:  class 0
|  |  |  |  |  RepeatedAccess > 0.322:  class 1
|  MultiAgent = 1:
|  |  totalPages <= 81:  class 0
|  |  totalPages > 81:  class 1
```

Subtree Raising

Subtree Replacement

Simplified Decision Tree:

```
depth = 1:
|  ImagePages <= 0.1333:  class 1
|  ImagePages > 0.1333:
|  |  breadth <= 6:  class 0
|  |  breadth > 6:  class 1
depth > 1:
|  MultiAgent = 0:  class 0
|  MultiAgent = 1:
|  |  totalPages <= 81:  class 0
|  |  totalPages > 81:  class 1
```



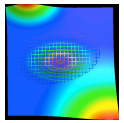
Subtree raising

Levelek: 20
Fa: 39

```
income <= 51284.3
|  children <= 1
|  |  children <= 0
|  |  |  married = NO
|  |  |  |  mortgage = NO: YES (43.0/3.0)
|  |  |  |  mortgage = YES
|  |  |  |  |  save_act = NO: YES (12.0)
|  |  |  |  |  save_act = YES: NO (22.0)
|  |  |  married = YES
|  |  |  |  save_act = NO
|  |  |  |  |  mortgage = NO
|  |  |  |  |  |  income <= 21506.2
|  |  |  |  |  |  |  age <= 41: NO (11.0/1.0)
|  |  |  |  |  |  |  age > 41: YES (5.0/1.0)
|  |  |  |  |  |  income > 21506.2: NO (20.0)
|  |  |  |  |  mortgage = YES: YES (25.0/3.0)
|  |  |  |  save_act = YES: NO (115.0/12.0)
|  |  children > 0
|  |  |  income <= 15538.8
|  |  |  |  age <= 41
|  |  |  |  |  income <= 12683.6: NO (14.0)
|  |  |  |  |  income > 12683.6
|  |  |  |  |  |  income <= 13381: YES (2.0)
|  |  |  |  |  |  income > 13381: NO (6.0)
|  |  |  |  age > 41: YES (2.0)
|  |  |  income > 15538.8: YES (101.0/5.0)
|  children > 1
|  |  income <= 30404.3: NO (124.0/12.0)
|  |  income > 30404.3
|  |  |  children <= 2: YES (36.0/5.0)
|  |  |  children > 2
|  |  |  |  income <= 44288.3: NO (19.0/2.0)
|  |  |  |  income > 44288.3: YES (6.0)
income > 51284.3
|  children <= 0
|  |  age <= 62: YES (4.0)
|  |  age > 62: NO (6.0/1.0)
|  children > 0: YES (27.0)
```

Levelek: 17
Fa: 33

```
children <= 1
|  children <= 0
|  |  married = NO
|  |  |  mortgage = NO: YES (48.0/3.0)
|  |  |  mortgage = YES
|  |  |  |  save_act = NO: YES (12.0)
|  |  |  |  save_act = YES: NO (23.0)
|  |  married = YES
|  |  |  save_act = NO
|  |  |  |  mortgage = NO
|  |  |  |  |  income <= 21506.2
|  |  |  |  |  |  age <= 41: NO (11.0/1.0)
|  |  |  |  |  |  age > 41: YES (5.0/1.0)
|  |  |  |  |  income > 21506.2: NO (20.0)
|  |  |  |  mortgage = YES: YES (25.0/3.0)
|  |  |  save_act = YES: NO (119.0/12.0)
|  children > 0
|  |  income <= 15538.8
|  |  |  age <= 41
|  |  |  |  income <= 12683.6: NO (14.0)
|  |  |  |  income > 12683.6
|  |  |  |  |  income <= 13381: YES (2.0)
|  |  |  |  |  income > 13381: NO (6.0)
|  |  |  age > 41: YES (2.0)
|  |  income > 15538.8: YES (111.0/5.0)
children > 1
|  income <= 30404.3: NO (124.0/12.0)
|  income > 30404.3
|  |  children <= 2: YES (51.0/5.0)
|  |  children > 2
|  |  |  income <= 44288.3: NO (19.0/2.0)
|  |  |  income > 44288.3: YES (8.0)
```



Pruning hatása

Subtree replacement vs. raising

Tanuló halmaz

ID	jármű	szín	gyorsulás
T1	motor	piros	nagy
T2	motor	kék	nagy
T3	autó	kék	nagy
T4	motor	kék	nagy
T5	autó	zöld	kicsi
T6	autó	kék	kicsi
T7	autó	kék	nagy
T8	autó	piros	kicsi

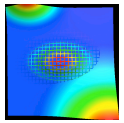
Validációs halmaz

ID	jármű	szín	gyorsulás
Valid 1	motor	piros	kicsi
Valid 2	motor	kék	nagy
Valid 3	autó	kék	nnagy
Valid 4	autó	kék	nagy

Test set

ID	jármű	szín	gyorsulás
Teszt 1	motor	piros	kicsi
Teszt 2	motor	zöld	kicsi
Teszt 3	autó	piros	kicsi
Teszt 4	autó	zöld	kicsi

Építsünk egy két szintű döntési fát
Csökkentsük a fa méretét (pruning) a validációs halmaz alapján
Hogyan változott a modell teljesítménye a teszt halmazon?

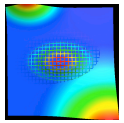


Súlyozott hiba és C4.5

A	B	C	“+”	“-”
I	I	I	5	0
H	I	I	0	20
I	H	I	20	0
H	H	I	0	5
I	I	H	0	0
H	I	H	25	0
I	H	H	0	0
H	H	H	0	25

Építsünk egy két szintű döntési fát
Létezik ideális dönntési fa?
Az alábbi "cost" mátrix alapján változtassuk meg a predikciót:

Jósolt/"ground truth"	“+”	“-”
“+”	0	1
“-”	2	0



Unix shell

Adatok formázásához:

file másolása unix alatt (Windows alatt cygwin)

```
cp vmit valahova
```

file megnézése:

```
less valami
```

első oszlop kivágása:

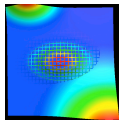
```
cut -d' ' -f1,2 adatfajl
```

rendezés numerikusan:

```
cut -d' ' -f1 adatfajl | sort -n
```

azonos sorok eldobása:

```
cut -d' ' -f1 adatfajl | sort -n |uniq
```



Unix shell

sorok száma:

```
cut -d' ' -f1 adatfajl | sort -n |uniq |wc -l
```

első oszlopbeli értékek száma:

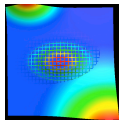
```
cut -d' ' -f1 adatfajl | sort -n |uniq -c
```

ugyanez awk-val:

```
awk '{v[$1]++}END{for(i in v) print i,v[i]}' adatfajl
```

első oszlop átlaga:

```
awk '{s+=$1}END{print s/NR;}' adatfajl
```



Unix shell

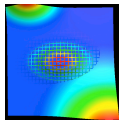
második oszlopbeli értékek átlaga első oszlop mint kulcs szerint
awk '{v[$1]+=$2;c[$1]++}END{for (i in v) print i, v[i]/c[i]}' adatfajl

file-ba irás echo-val
echo "AKARMI" > output
echo "MEG VALAMI" >> output

egy adott karakter kicserélése:
tr ',' '.' < input > output

két file konkatenálása:
cat file1 file2 > output

Egyéb hasznos eszközök:
du,df ,ls,paste,for,chmod,mc,sed,for,seq stb.



Unix shell

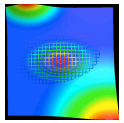
getline < „file”: „file” következő sora
BEGIN, END: az aktuális stream olvasásán kívül

Mit csinál a következő parancs?

```
awk –v ID=„lista” ‘BEGIN{while(getline < ID) id[$1]=1}
    {if(id[$1]==1) print $0;}’ logfile
```

Lista: <id>

Logfile: <id> <nev> <varos>



Távolság ism.

Távolság mértékek (pl.)

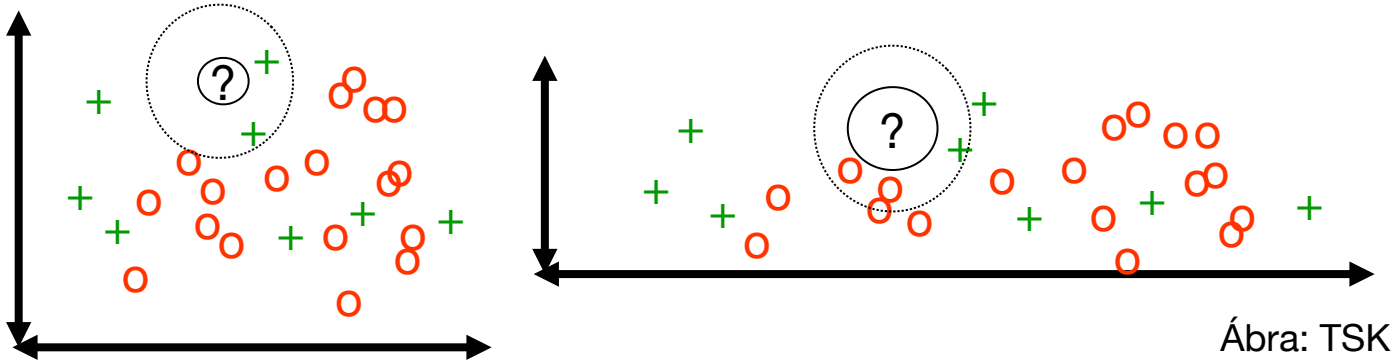
- Minkowski távolság

- Mahalanobis $D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}.$

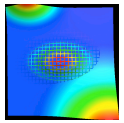
- Cosine, Jaccard, Kullback-Leibler, Jensen-Shannon stb.

Problémák:

- skála
- autómatikus távolság választás
- normalizáció



Ábra: TSK



Órai feladat 2

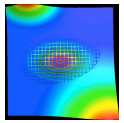
NN_data.zip

image_histograms.txt és sample_histogram.txt:

Adottak az images könyvtár képein számolt 3x8 dimenziós RGB hisztogramok

Rendezzük sorrendbe távolság szerint a sample_histogram.txt fájl-ban található kép hisztogramjától számított L2 távolság alapján az image_histograms.txt képeit!

Mi lehet a hiba? Mi történik ha L2 normalizáljuk a hisztogrammokat?



iPython notebook

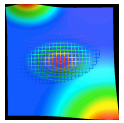
Anaconda:

wget "http://repo.continuum.io/archive/Anaconda3-4.0.0-Linux-x86_64.sh"
chmod +x Anaconda3-4.0.0-Linux-x86_64.sh
./Anaconda3-4.0.0-Linux-x86_64.sh
source .bashrc
conda update conda
conda update anaconda

conda create -n jupyter-env python=3.5 anaconda
source activate jupyter-env

pip install <module_name>

Telepítsük:
pip install pandas
pip install chainer



iPython notebook

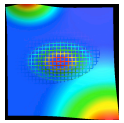
jupyter notebook --generate-config
mcedit .jupyter/jupyter_notebook_config.py
c.NotebookApp.port = 9992

Érdemes nem a default 8888-as portot használni.

Végül:

jupyter notebook --no-browser

Browser-ben nyissuk meg a localhost:9992-ot.



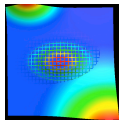
iPython notebook

Egyszerű példa:

```
import numpy as np
import pandas as pd

v = np.random.random((3))
m = np.random.random((2,3))

v.dot(m)
```

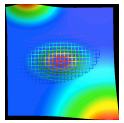
Próbáljuk megoldani az előző feladatot:

- pd.read_csv()
- dataframe index és values
- for i in range(10):
  <work>
- np.linalg.norm(v1-v2) -> L2 távolság
- np.argmax()



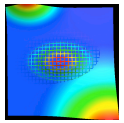
iPython notebook

```
# Olvassuk be az adatot

img_hists = pd.read_csv('/home/daroczyb/datasets/NN/image_histograms.txt',sep=' ')
act_hist = pd.read_csv('/home/daroczyb/datasets/NN/sample_histogram.txt',sep=' ')
```



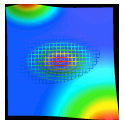
iPython notebook

```
# Olvassuk be az adatot

img_hists = pd.read_csv('/home/daroczyb/datasets/NN/image_histograms.txt',sep=' ')
act_hist = pd.read_csv('/home/daroczyb/datasets/NN/sample_histogram.txt',sep=' ')

# távolság értékek -> numpy array

dist = np.zeros((len(img_hists.index)))
dist_norm = np.zeros((len(img_hists.index)))
```



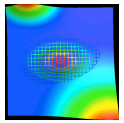
iPython notebook

```
# Olvassuk be az adatot

img_hists = pd.read_csv('/home/daroczyb/datasets/NN/image_histograms.txt',sep=' ')
act_hist = pd.read_csv('/home/daroczyb/datasets/NN/sample_histogram.txt',sep=' ')

# távolság értékek -> numpy array

dist = np.zeros((len(img_hists.index)))
dist_norm = np.zeros((len(img_hists.index)))

# pandas dataframe -> numpy array

hist_vecs = np.array(img_hists.values).astype(np.float32)
hist_vecs_norm = np.copy(hist_vecs).astype(np.float32)
```



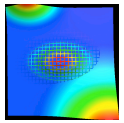
iPython notebook

```
# Olvassuk be az adatot

img_hists = pd.read_csv('/home/daroczyb/datasets/NN/image_histograms.txt',sep=' ')
act_hist = pd.read_csv('/home/daroczyb/datasets/NN/sample_histogram.txt',sep=' ')

# távolság értékek -> numpy array

dist = np.zeros((len(img_hists.index)))
dist_norm = np.zeros((len(img_hists.index)))

# pandas dataframe -> numpy array

hist_vecs = np.array(img_hists.values).astype(np.float32)
hist_vecs_norm = np.copy(hist_vecs).astype(np.float32)

# normalizáció (L2)

act_vec = p.array(act_hist.values).astype(np.float32)
act_vec_norm = act_vec/np.linalg.norm(act_vec).astype(np.float32)
for i in range(hist_vecs[:,0].size):
        norm= np.linalg.norm(hist_vecs[i])
        hist_vecs_norm[i] = hist_vecs[i]/norm

# Norma és távolság?
```



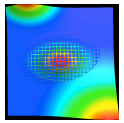
iPython notebook

```
# távolság számítás

for i in range(hist_vecs[:,0].size):
        dist[i] = np.linalg.norm(hist_vecs[i]-act_vec)
        dist_norm[i] = np.linalg.norm(hist_vecs_norm[i]-act_vec_norm)
```



iPython notebook

```
# távolság számítás

for i in range(hist_vecs[:,0].size):
        dist[i] = np.linalg.norm(hist_vecs[i]-act_vec)
        dist_norm[i] = np.linalg.norm(hist_vecs_norm[i]-act_vec_norm)

# min, max

top = np.argmin(dist)
top_val = np.min(dist)
top_norm = np.argmin(dist_norm)
top_norm_val = np.min(dist_norm)
```



iPython notebook

```
# távolság számítás

for i in range(hist_vecs[:,0].size):
        dist[i] = np.linalg.norm(hist_vecs[i]-act_vec)
        dist_norm[i] = np.linalg.norm(hist_vecs_norm[i]-act_vec_norm)

# min, max

top = np.argmin(dist)
top_val = np.min(dist)
top_norm = np.argmin(dist_norm)
top_norm_val = np.min(dist_norm)

# ellenőrzés

print('normalizacio elott: %s,%s %f' % (act_hist.index[0], img_hists.index[top], top_val))
print('normalizacio utan: %s,%s %f' % (act_hist.index[0], img_hists.index[top_norm], top_norm_val))
```