# PAC 2

## Presentació

Segona activitat d'avaluació continuada del curs. En aquesta PAC es practicaran els algorismes bàsics de classificació.

## Objectius

L'objectiu d'aquesta prova d'avaluació és classificar les dades dels arxius adjunts relacionats amb els guanys de les persones a partir de la informació del cens. Volem predir si superen els 50k dòlars l'any o no.

Els arxius de dades "csv" tenen un format tipus taula, on cada fila correspon a un exemple. L'última columna és la classe i la resta corresponen als atributs de l'exemple. L'arxiu adjunt "txt" conté la descripció d'aquests atributs.

Aquests arxius pertanyen al problema "Census Income Data Set" del repositori d'aprenentatge de l'UCI:

http://archive.ics.uci.edu/ml/

## Solució de la PAC

### Exercici 1

a) Construïu els models de classificació basats en el veí més proper per valors de k u i tres a partir de l'arxiu train.csv. És a dir, heu de construir els models: 1NN i 3NN. Un cop tingueu el model, heu de classificar els exemples de l'arxiu test.csv.

Ens proporcionen un arxiu amb dades relacionat amb guany anuals de persones a partir de dades del cens. No hi ha valors absents, no els hem de tractar.

Els atributs són molt diferents. Tenim un atribut numèric (age), un cardinal (education) i tres nominals o binaris (marital-status, occupation i native-country). El numèric el tractarem aplicant ranging ((valor – mínim) / (màxim – mínim)). Al cardinal li assignarem 0, 0.5 o 1 en funció del seu valor i als nominals 0 o 1. El resultat global serà:

```
0.17 0.5 1 0 0 0.71 0 pobre
0.48 1 1 0 0.71 0 1 pobre
1.0 0 1 0 0.71 0 1 ric
0.28 1 0 0 0 0.71 1 ric
0.66 0.5 1 0 0.71 0 1 ric
0.0 0.5 0 0.71 0 0 1 pobre
```

Apliquem el mateix al conjunt de test (s'han d'aplicar els mínims i màxims del conjunt de training). El resultat és:

```
0.52 1 1 0 0.71 0 1 ric
0.03 0 0 0 0.71 0 1 pobre
```

Per aplicar el kNN, calculem les distàncies euclidees de tots els exemples de test a tots els exemples de train:

```
           [,1]     [,2]
[1,] 1.53912510 1.808045
[2,] 0.03448276 1.483560
[3,] 1.11043050 1.390044
[4,] 1.43466511 1.434665
[5,] 0.51867617 1.278771
[6,] 1.58667534 1.118566
```

Per a 1NN, assignarem la classe "pobre" als dos exemples, ja que els més petits (marcats en vermell) són d'aquesta classe, que correspon a una precisió del 50% ($=\frac{0+1}{2}$).

Per al 3NN els vots serien (pobre, ric, ric) i (ric, ric, pobre) que corresponen a la predicció "ric" per als dos exemples; que equival a una precisió del 50% ($=\frac{1+0}{2}$).

b) Construïu els models de classificació basats en el k–means per a un valor de k de dos a partir de l'arxiu train.csv. Un cop tingueu el model, heu de classificar els exemples de l'arxiu test.csv.

Apliquem en primer lloc el 2-means dues vegades: una per als 3 exemples de cadascuna de les classes. Agafant com a centroides inicials els dos primers exemples en cada cas ens queden els quatre centroides:

```
0.3275862 0.75  1 0.0000000 0.3535534 0.3535534 0.5 pobre
0.0000000 0.50  0 0.7071068 0.0000000 0.0000000 1.0 pobre
0.8275862 0.25  1 0.0000000 0.7071068 0.0000000 1.0 ric
0.2758621 1.00  0 0.0000000 0.0000000 0.7071068 1.0 ric
```

Apliquem 1NN agafant com a training els quatre centroides obtinguts en el pas anterior i com a test el conjunt de test. El tractarem com en l'exercici anterior. Obtenim les distàncies següents:

```
           [,1]     [,2]
[1,] 0.7736078 1.465745
[2,] 1.5866753 1.118566
[3,] 0.8116735 1.300582
[4,] 1.4346651 1.434665
```

Que corresponen a les prediccions "pobre" per als dos exemples. Això equival a un 50% de precisió.

c) Construïu un arbre de decisió a partir de l'arxiu train.csv i classifiqueu amb ell els exemples de l'arxiu test.csv.

Per a construir arbres de decisió no cal normalitzar, donat que no treballarem amb distancies, ni codificar els atributs nominals, donat que és un algorisme que ja els tracta. Treballarem amb els conjunts originals.

Comencem per trobar les bondats de tots els atributs nominals. Calculem les bondats dels diferents atributs assignant la classe majoritària a cada valor. Obtenim:

```
education      67%  =(2+1+1)/6
marital-status 50%  =(2+1)/6
occupation     67%  =(1+2+1)/6
native-country 67%  =(1+3)/6
```

Per a l'atribut numèric *age* utilitzarem els punts de tall per tractar-los. Els punts de tall es calculen ordenant tots els valors d'un atribut i calculant la mitjana aritmètica de cada dos valors consecutius. A continuació es mostra els valors ordenats sense repeticions, els punts de tall i la seva bondat:

```
23   25,5 67%  =(1+3)/6
28   29,5 83%  =(2+3)/6
31   34   67%  =(2+2)/6
37   39,5 83%  =(3+2)/6
42   47   67%  =(3+1)/6
52
```

El millor que obtenim és el punt de tall 39,5 de l'atribut 'age', amb el que aconsegueixen un 83% de bondat. La part per damunt del punt de tall queda complerta i li assignem la classe 'ric'. Hem de tornar a iterar amb els quatre exemples que queden per sota del punt de tall. Les bondats per als atributs nominals seran ara:

```
education      75%  =(2+1)/4
marital-status 75%  =(2+1)/4
occupation     75%  =(1+1+1)/4
native-country 75%  =(1+2)/4
```

I per a l'atribut numèric:

```
23   25,5 75%  =(1+2)/4
28   29,5 75%  =(2+1)/4
31   34   75%  =(2+1)/4
37
```

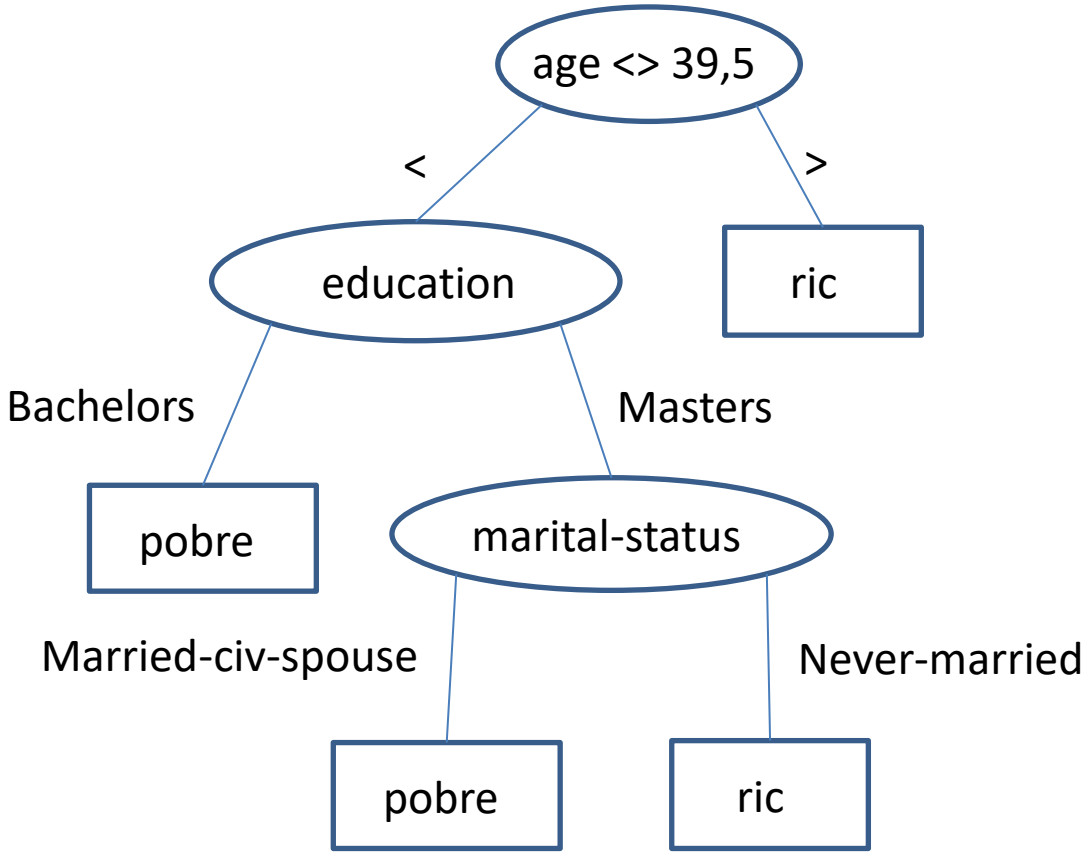
En aquest cas obtenim un empat entre tots el casos. Seleccionem un qualsevol; en el nostre cas *education*. Els dos exemples amb valor 'Bachelors' queden ben classificats i els assignem la classe 'pobre'. Tornem a iterar per als dos que tenen el valor 'Masters'. Hem d'eliminar la columna *education*. Les bondats per als atributs nominals seran ara:

```
marital-status 100% =(1+1)/2
occupation     100% =(1+1)/2
native-country 50%  =1/2
```

I per a l'atribut numèric:

```
31    34    100% =(1+1)/2
37
```

En aquest cas obtenim un empat entre tots el casos excepte *native-country*. Seleccionem un qualsevol; en el nostre cas *marital-status*. Tenim tot ja ben classificat. El arbre resultant serà:




Les prediccions són 'pobre' per al primer exemple de test i per al segon no donarà predicció ja que no existeix el valor 'HS-grad' de *education* en l'arbre. Per tant, obtindrem una precisió del 0%.

d) Apliqueu l'algorisme del PCA per reduir la dimensionalitat dels conjunts anteriors conservant el 95% de la variància. Utilitzeu el 1-NN i 3-NN sobre els conjunts reduïts

de la mateixa forma que en el primer apartat. S'obtenen resultats comparables? Expliciteu les diferències en aplicar el PCA sobre el conjunt d'entrenament i sobre el de test.

Apliquem el PCA sobre el conjunt de l'arxiu. Obtenim els percentatges de variàncies:

```
0.50 0.28 0.14 0.07 0.00 0.00
```

I acumulant:

```
0.50 0.79 0.93 1.00 1.00 1.00 1.00 1.00
```

El resultat és:

```
 0.26  1.06  0.18 -0.08
-0.41 -0.13 -0.46 -0.34
-0.88 -0.14  0.32  0.32
 0.84 -0.10 -0.43  0.35
-0.60 -0.13 -0.07 -0.05
 0.78 -0.57  0.46 -0.21
```

I projectant el test:

```
-0.42 -0.13 -0.47 -0.32
 0.15 -0.50  0.41  0.18
```

Un cop obtingudes les dades projectades, fem el mateix que al primer apartat, calcular la matriu de distàncies:

```
             [,1]       [,2]
[1,] 1.53844318 1.6055842
[2,] 0.02405457 1.2177786
[3,] 1.11007890 1.1038245
[4,] 1.43386419 1.1703794
[5,] 0.51114069 0.9976114
[6,] 1.58603260 0.7478283
```

Per a 1NN, assignarem la classe 'pobre' als dos exemples, ja que els més petits (marcats en vermell) són d'aquestes classes, que correspon a una precisió del 50% ($\frac{0+1}{2}$).

Per al 3NN els vots serien ('pobre', 'ric', 'ric') i ('ric', 'ric', 'pobre') que corresponen a les prediccions de 'ric' per als dos exemples; que equival a una precisió del 50% ($\frac{1+0}{2}$).

## Exercici 2

L'objectiu d'aquest segon exercici és la construcció d'un model amb un conjunt de dades proper al que utilitzaríem en una aplicació pràctica per a un cas real. Per realitzar aquest exercici, teniu a la vostra disposició una eina anomenada Weka a la Web; la seva direcció és:

http://www.cs.waikato.ac.nz/ml/weka

L'arxiu adjunt "adult.data.csv" conté les dades del cens de 32.561 persones. Aquest arxiu està en un dels formats d'entrada del Weka.

Se us demana l'estudi de cara a construir un classificador. Per a realitzar-lo heu de:

- Provar els algorismes per validació creuada (cross-validation). És a dir, no utilitzar un sol arxiu de train i un de test. Aquesta opció la permet realitzar el Weka de forma automàtica.
- Provar almenys els algorismes: naïve bayes, algun tipus d'arbre de decisió (mostrant a l'informe l'arbre generat), AdaBoost, xarxes neuronals (perceptró multicapa) i algun altre algorisme.
- Provar les Support Vector Machines (SMO i/o libSVM al Weka) amb diferents tipus de kernel.

Comenteu els resultats obtinguts i justifiqueu tot el que feu.

Hem escollit realitzar les proves que demana l'enunciat amb un 10-fold crossvalidation sobre els algorismes: Naïve Bayes, 1NN (IB1), 3NN (IB3), Decision Stumps, l'arbre de decisió J48, l'AdaBoost.M1, el perceptró multicapa i les màquines de vectors de suport (amb kernel lineal i rbf).

Hem aplicat els filtres ReplaceMissingValues a tots els atributs, el Normalize als numèrics i el NominalToBinary als nominals.

La taula següent mostra els resultats per al Naïve Bayes, Decision Stumps, 1NN, 3NN i AdaBoost per al conjunt; concretament mostra el temps de construcció dels models, la precisió global, una mesura per cadascuna de les classes i les dades de l'estadístic Kappa (mesura la proporció de la precisió entre les diferents classes):

| | NB | DecisionStump | 1NN | 3NN | AdaBoost.M1 |
|---|---|---|---|---|---|
| **Temps (s)** | 1,75 | 2,64 | 0,06 | 0,03 | 25,88 |
| **Oks** | 27324 | 24720 | 25815 | 26667 | 27340 |
| **Precision** | 84% | 76% | 79% | 82% | 84% |
| **F-mesure <=50K** | 0,90 | 0,86 | 0,86 | 0,88 | 0,90 |
| **F-mesure >50K** | 0,66 | 0,00 | 0,56 | 0,61 | 0,59 |
| **Kappa stat.** | 0,55 | 0 | 0,43 | 0,49 | 0,50 |

i a continuació les matrius de confusió:

| | | |
|---|---|---|
| **Naïve Bayes** | 22359 2361 \| a = <=50K<br>2876 4965 \| b = >50K |
| **Decision Stump** | 24720 0 \| a = <=50K<br>7841 0 \| b = >50K |
| **1NN** | 21462 3258 \| a = <=50K<br>3488 4353 \| b = >50K |
| **3NN** | 22137 2583 \| a = <=50K<br>3311 4530 \| b = >50K |

| | |
|---|---|
| **AdaBoost.M1** | 23593 1127 \| a = <=50K<br>4094 3747 \| b = >50K |

En intentar aplicar el perceptró multicapa i les màquines de vectors de suport hem comprovat que no convergeixen en un temps "raonable" (menys de mitja hora). Les possibles aproximacions que podem tantejar es reduir el conjunt d'entrenament o aplicar tècniques d'aprenentatge incremental. Nosaltres aplicarem la reducció del conjunt (a 1000 exemples) donat que l'aprenentatge incremental se'n surt del objectius d'aquest curs. A continuació teniu els resultats per aquest conjunt reduït:

| | NB | DecStu | 1NN | 3NN | AdaB | PercMult | SMO(lin) | SMO(rbf) |
|---|---|---|---|---|---|---|---|---|
| **Temps (s)** | 0,03 | 0,02 | 0 | 0 | 0,27 | 116,68 | 0,41 | 0,69 |
| **Oks** | 853 | 775 | 774 | 791 | 838 | 797 | 836 | 775 |
| **Precision** | 85% | 78% | 77% | 79% | 84% | 80% | 84% | 78% |
| **F-mesure <=50K** | 0,91 | 0,87 | 0,86 | 0,87 | 0,89 | 0,87 | 0,90 | 0,87 |
| **F-mesure >50K** | 0,67 | 0,00 | 0,49 | 0,51 | 0,59 | 0,54 | 0,60 | 0 |
| **Kappa stat.** | 0,58 | 0 | 0,34 | 0,38 | 0,49 | 0,41 | 0,50 | 0 |

i a continuació les matrius de confusió:

| | |
|---|---|
| **Naïve Bayes** | 704 71 \| a = <=50K<br>76 149 \| b = >50K |
| **Decision Stump** | 775 0 \| a = <=50K<br>225 0 \| b = >50K |
| **1NN** | 666 109 \| a = <=50K<br>117 108 \| b = >50K |
| **3NN** | 681 94 \| a = <=50K<br>115 110 \| b = >50K |
| **AdaBoost.M1** | 724 51 \| a = <=50K<br>111 114 \| b = >50K |
| **MultilayerPerceptron** | 677 98 \| a = <=50K<br>105 120 \| b = >50K |
| **SMO(linear)** | 713 62 \| a = <=50K<br>102 123 \| b = >50K |
| **SMO(rbf)** | 775 0 \| a = <=50K<br>225 0 \| b = >50K |

## Exercici 3

Realitzeu una valoració global comparant els mètodes i els diferents exercicis i redacteu unes conclusions globals sobre l'aplicació dels mètodes a aquests conjunts de dades. Els criteris de correcció de la PAC invaliden una A si tots els processos no estan ben justificats i comentats.

En aquest apartat s'espera que extrèieu conclusions generals sobre l'exercici. Aquestes conclusions dependran dels resultats obtinguts. A mode d'exemple enumerarem algunes de les qüestions sobre les que podeu argumentar:

- La possibilitat de que sigui pràctica l'aplicació d'algun dels mètodes sobre el problema donat. En cas que no, que faltaria afegir.
- Comparativa dels diferents mètodes emprats. Enumeració dels avantatges i inconvenients en funció de: precisió, eficiència, categories, models...
- La representació del problema. Com es comporten els atributs? És una bona representació? Com afecta el preprocés de les dades al funcionament dels algorismes.
- Avantatges dels models que generen els diferents mètodes. Comparativa dels models generats durant tot l'exercici.
- En general, intent de justificació i/o explicació dels resultats que es van obtenint: fixant-se no només en la precisió.
- Com es comporten els algorismes en funció del nombre d'exemples d'entrenament que es disposen?
- Quin cost computacional té cadascun dels mètodes? Tant en el procés de training com en el de test.

En comparar els resultats, és important notar que els conjunts de dades tenen diferent número de classes.

### Exercici 4

Cerca informació i dona raonadament la diferència entre els mètodes de "clustering": k-means, k-medians i k-medoids. Dóna també per a quin tipus de dades es recomanable escollir un mètode o un altre d'entre aquests.

La diferència bàsica entre els tres mètodes que menciona l'enunciat és la mesura de distància que s'utilitza per calcular els centroides. En el k-means s'utilitza el promig, en el k-medians la mediana i en el k-medoids el medoide.

El promig ens donarà la mitjana aritmètica dels valors. La mediana els valors que hi ha al mig si els ordenem. El medoide calcula el valor menys llunyà a la resta de valors; sempre ha de ser un element del conjunt, a diferència dels dos anteriors.

Els dos primers mètodes són adequats per atributs numèrics. El k-medoid és adequat per atributs nominals; en aquest cas no té sentit que el resultat sigui un valor que no pertanyi als valors possibles.