**Data Mining**

**Lab 3: Clasificarea datelor**

---

Sumar:

- Problema clasificarii
- Clasificatori bazati pe arbori de decizie (rpart)
- Clasificatori bazati pe reguli de decizie (OneR, JRip)
- Clasificatori bazati pe instanțe (k- NearestNeighbor)
- Clasificatori probabilişti (Naïve Bayes)
- Clasificatori bazaţi pe reţele neuronale (perceptron multinivel)
- Clasificatori bazaţi pe vectori suport

## 1. Problema clasificării

*Scopul unui model de clasificare:* stabilirea clasei căreia îi aparțin datele de intrare

*Construirea unui clasificator* = extragerea unui model de clasificare pornind de la date de antrenare; procesul de construire este denumit învățare sau antrenare.

*Validarea unui clasificator*: tehnica validării încrucișate (setul de date este fragmentat în k felii și se construiesc k clasificatori – fiecare clasificator fiind construit folosind (k-1) felii din setul de date și fiind validat folosind a k-a felie)

*Evaluarea performanței:* matrice de confuzie, acurateţe, senzitivitate, specificitate, regăsire (recall), precizie (precision), F-measure, curba receiver-operating characteristic (ROC) şi aria de sub curbă (AUC=area under curve).

Accuracy = (TP+TN)/(TP+TN+FP+FN) = nr date clasificate corect/ nr total de date
Sensitivity = TP/(TP+FN) (TP rate sau recall = rata de regăsire)
Specificity= TN/ (TN+FP) (TN rate), 1-specificitate=FP/(TN+FP) = FP rate
Precision = TP/(TP+FP) (nr cazuri real pozitive/ nr cazuri clasificate ca fiind pozitive)
F-measure=2*precision*recall/(precision+recall)

Kappa = (Accuracy – ExpectedAcc)/(1-ExpectedAcc) - ajustare a măsurii de acurateţe care ţine cont de proporţia cazurilor în care clasificatorul generează răspuns corect datorită întâmplării – este util în particular în cazul seturilor nebalansate de date.

$$\text{ExpectedAcc} = (TP+FP)(TP+FN)/N^2 + (TN+FN)(TN+FP)/N^2 \qquad (N = \text{nr total de date})$$

Curba ROC: TP rate (sensitivitate) vs. FP rate (1-specificitate). Se calculează perechi de valori (FP rate, TP rate) pentru fiecare felie din setul de antrenare (în contextual validării încrucişate cu k felii) şi se unesc punctele (0,0), $P_1$, …,Pk, (1,1) (unde Pi este punctul având a i-a valoare în ordine crescătoare a valorii FP rate). Ideal ar fi ca toate punctele să aibă coordonatele (0,1), iar aria de sub curbă să coincidă cu aria pătratului definit de (0,0) şi (1,1), adică 1.

Pentru seturile de date nebalansate (semnificativ mai multe exemple intr-o clasă în raport cu celelalte clase) se foloseşte coeficientul de corelaţie Matthew (MCC):

MCC=(TP*TN-FP*FN)/sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))

**Exerciţiul 1.** Familiarizare cu pachetul "caret" – Classification And Regression Training (interfaţă unitară pentru peste 200 de variante de algoritmi de clasificare şi regresie implementaţi în diverse pachete R). Facilităţi principale:

- Partiţionare seturi de date în subset de antrenare şi subset de testare (function createDataPartition)
- Antrenarea unui model (funcţia train) şi definirea unei proceduri de control a procesului de antrenare şi determinare a valorilor parametrilor de control (hiperparametri) (funcţia trainControl); permite specificarea unei strategii de validare: selecție aleatoare cu repetare - *bootstrapping* (implicit – cu 25 de repetări), *cross-validation* - validare încrucişată etc.
- Includerea unui pas de pre-procesare în faza de antrenare (e.g. parametrul preProc în funcţia train); exemple de preprocesare: centrare ("center"), scalare cu valoarea abaterii standard ("scale"), transformare PCA ("pca"), imputarea valorilor absente ("knnImpute")

Punct de pornire: templateUsingCaret.r
Detalii:
https://cran.r-project.org/web/packages/caret/caret.pdf , http://topepo.github.io/caret/index.html

## 2. Clasificatori bazați pe arbori de decizie

Un arbore de decizie conține:

- *Noduri interne* (fiecare nod intern are asociat un atribut); fiecare nod intern are asociate condiții de ramificare, fiecare condiție corespunzând unei ramuri în arbore)
- *Noduri frunză* (fiecare nod frunză are asociată o clasă)

Mod de construire:

- pentru nodul rădăcină se identifică atributul și condițiile de ramificare caracterizate prin cel mai mare câștig informational (information gain – IG) in ceea ce privește clasificarea datelor din setul de date
- pentru fiecare dintre ramuri și setul de date corespunzător ramurii se aplică în mod recursiv aceeași strategie până este îndeplinită o condiție de oprire (s-a ajuns la un set care conține date din aceeași clasă – nod pur - sau la un set mic de date)

*Observatie*: un aspect important este cel referitor la complexitatea arborelui (sunt de preferat arbori cât mai simpli); simplificarea unui arbore (*tree pruning*) se poate realiza:

- în etapa de construire (stopând procesul de ramificare dacă numărul de elemente din setul curent este mai mic decât un prag)
- după construirea arborelui, prin înlocuirea unor subarbori cu noduri frunză (în cazul în care performanța clasificatorului nu este alterată în mod semnificativ)

*Mod de utilizare:* pentru o anumită dată se identifică ramura din arborele de decizie care se potrivește cu valorile atributelor iar eticheta nodului frunză la care se ajunge reprezintă clasa asociată datei.

*Pachete R:* rpart, party

**Exerciţiul 2/ R:** Construire, vizualizare şi evaluarea performanţelor unor arbori de clasificare pentru: (a) iris dataset; (b) Titanic dataset; (c) Breast Cancer dataset; (d) [bioinfo] DNA (din pachetul mlbench). Utilizare pachete: party, rpart şi caret

*Punct de pornire:* DecisionTrees_PartyRpart.R, DecisionTrees_caret.R

*Implementare în Weka*: Id3, J48

## 3. Clasificatori bazați pe reguli de clasificare

Regulile de clasificare sunt de forma IF <antecedent> THEN <eticheta clasa>, unde partea de antecedent conține condiții referitoare la valorile atributelor combinate prin operator de conjuncție sau disjuncție.

*Mod de construire:*

- Pornind de la un arbore de decizie, pentru fiecare ramură se poate construi o regulă de clasificare (condițiile din antecedent sunt combinate prin conjuncții)
- Direct pornind de la date prin algoritmi de acoperire ("covering algorithms")

*Pachete R:* caret include clasificatori bazaţi pe reguli din RWeka (JRip şi OneR)

*Implementare in Weka:*

- Pornind de la arbori de decizie: M5rules, PART
- Covering algorithms: OneR, JRip

**Exerciţiul 3/ R:** Construire, vizualizare şi evaluarea unor clasificatori bazaţi pe reguli pentru: (a) iris dataset; (b) Breast Cancer dataset; (c) Titanic dataset; (d) [bioinfo] DNA (din pachetul mlbench). Utilizare pachete: caret si RWeka

*Punct de pornire:* ClassificationRules.R

## 4. Clasificatori bazati pe instanțe (Instance Based Learning Classifiers)

Cei mai simpli clasificatori bazat pe instanțe sunt cei din categoria "cel mai apropiat vecin" (nearest neighbor)

*Mod de construire:*

- modelul de clasificare constă chiar din setul de date (nu există etapă propriu-zisă de antrenare)

*Mod de utilizare:*

- in procesul de clasificare, pentru o nouă instanță se identifică cele mai similare instanțe din setul de antrenare și clasa dominantă va fi considerată ca fiind cea corespunzătoare noii date.

*Pachete R*: pachet caret/ metoda knn, pachet class / metode knn, knn1

*Implementare Weka*: IB1 (se utilizează un singur vecin), IBk (se utilizează k vecini)

**Exerciţiul 4/ R:** Construirea şi evaluarea performanţelor unui clasificator de tip kNN pentru: (a) Breast Cancer dataset; (b) Titanic dataset. (c) [bioinfo] DNA (din pachetul mlbench).

(i) Analizaţi influenţa tipului atributelor şi a preprocesărilor (e.g. standardization) asupra performanţei în clasificare şi asupra valorii determinate a parametrului de control (k = numărul de vecini utilizaţi în faza de predicţie)

(ii) Visualizaţi curba ROC (in cazul problemelor de clasificare binară) şi analizaţi valoarea ariei AUC.

*Punct de pornire*: kNN.r

## 5. Modelele probabiliste

Permit estimarea probabilității ca o dată să aparțină unei clase ($P(C_k|(a_1,a_2,\ldots,a_n)$). Clasa corespunzătoare datei este cea pentru care probabilitatea este maximă. Calculele se bazează pe formula lui Bayes: $P(C_k| a_1,a_2,\ldots,a_n)=P(a_1,a_2,\ldots,a_n | C_k)P(C_k)/ P(a_1,a_2,\ldots,a_n)$.

*Variante:*

- Model Bayesian simplu (Naïve Bayes): presupune că atributele nu sunt corelate (probabilitatea observării unei date este produsul probabilităților de a observa fiecare dintre valorile atributelor).
- Rețea cauzală (Bayesian Net sau Belief network): permite descrierea unui model sub forma unui graf orientat în care sunt specificate relații de cauzalitate.

*Pachete R:* pt Naïve Bayes: caret / metoda bn; pt rețele Bayesiene: pachet bnlearn

**Exerciţiu 5/ R:** Construiţi şi evaluaţi performanţa unui clasificator de tip Naïve Bayes pt: (a) Breast Cancer dataset; (b) Titanic dataset; (d) [bioinfo] DNA (din pachetul mlbench).

*Punct de pornire*: NaiveBayes.R

## 6. Rețele neuronale

Etape în proiectarea unei rețele neuronale pentru clasificare:

- *Stabilirea arhitecturii*. In cazul rețelelor multi-nivel (multi-layer perceptrons) se fixează:
  - Număr de unitati de intrare = numărul atributelor
  - Număr de nivele/unități ascunse – depinde de complexitatea problemei
  - Număr de unități de ieșire:
    - Clasificare binară: o unitate (rezultatul se interpretează folosind o valoare prag) sau două unități (unitatea care produce cea mai mare valoare va indica clasa)
    - Clasificare multiplă: numărul de unități coincide cu numărul de clase iar clasa dominantă este cea considerată clasa răspuns
- *Stabilirea funcțiilor de activare*. Pentru rețelele antrenate folosind algoritmul Backpropagation:

- Unitățile ascunse au funcții de activare de tip sigmoidal (functia logistică sau tanh)
- Unitățile de ieșire au funcții de activare de tip sigmoidal sau funcții liniare

- *Alegerea algoritmului de antrenare* și a parametrilor acestuia. In cazul algoritmului Backpropagation parametrii uzuali de control sunt:
  - Numărul de epoci de antrenare
  - Rata de învățare
  - Coeficientului termenului de tip „moment"

*Implementare R:* pachet nnet (inclus in caret) – rețea neuronală cu un nivel ascuns(cu diferite funcţii de activare la nivelul de ieşire – logistică (implicit), liniară, softmax) antrenată folosind un algoritm de tip Backpropagation ce minimizează MSE (mean squared error – implicit) sau entropia încrucişată (când e utilizată softmax ca funcţie de activare la nivelul de ieşire). Inclusă în Rattle.

*Implementare Weka*: MultilayerPerceptron = rețea feedforward antrenată cu Backpropagation (varianta cu moment); arhitectura rețelei este implicit cu un nivel de unități ascunse; numărul de unităti ascunse se poate specifica prin opţiunea hiddenLayers

**Exerciţiul 6/ R:** Construiţi şi evaluaţi performanţa unei reţele neuronale cu un nivel ascuns pt: (a) Breast Cancer dataset; (b) Titanic dataset; (d) [bioinfo] DNA (din pachetul mlbench).

Punct de pornire: NeuralNetworks.R

## 7. Clasificare folosind vectori suport (Support Vector Machines)

**Exerciţiul 7/ R:** Construiţi şi evaluaţi performanţa unui clasificator de tip SVM pt: (a) Breast Cancer dataset; (b) Titanic dataset; (d) [bioinfo] DNA (din pachetul mlbench). Analizaţi influenţa nucleului (metode svmRadial, svmLinear, svmPoly).

Punct de pornire: SVM.r

Varianta de clasificatori bazați pe vectori suport implementată în Weka folosește un algoritm rapid pentru rezolvarea problemei de optimizare (Sequential Minimal Optimization). In cazul clasificării in M clase problema se transformă în mai multe probleme de clasificare binară (corespunzătoare perechilor de clase)

Exemple adiționale (scikit-learn)

- https://scikit-learn.org/stable/auto_examples/index.html#classification

**Temă:**

1. Utilizaţi Rattle sau R (la alegere) pentru a construi şi compara performanţele unor clasificatori bazaţi pe (i) arbori de decizie; (ii) vectori suport; (iii) reţele neuronale.

   Seturi de date (la alegere):
   - Glass (from mlbench);

- LetterRecognition (from mlbench);
- German credit data (GermanCredit in caret) [http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29]
- [bioinfo] Molecular Biology (Promoter Gene Sequences) Data Set [http://archive.ics.uci.edu/ml/datasets/Molecular+Biology+%28Promoter+Gene+Sequences%29]