

# Temă Învățare Automată

- The one with all the ML Challenges -

## 1. Descriere generala

În practica de zi cu zi a unui inginer sau cercetător în domeniul învățării automate intra frecvent următoarele trei aspecte:

- Vizualizarea și “explorarea” datelor unei probleme (Exploratory Data Analysis)
- Încercarea de a extrage attribute ale datelor problemei pentru a fi utilizate în obiectivul de analiza (e.g. clasificare, regresie, detectie de anomalii) ales
- Investigarea de modele de rețele neurale potrivite pentru extragerea “automată” de attribute care ajută în obiectivul de analiză ales

În perioada recentă, date de la senzori de orice fel ajung să fie folosiți cu scopul de a colecta date despre procese dinamice, care evoluează în timp - de la gesturi umane până la date despre trafic sau tiparul de consum electric al unei gospodării tipice.

Seturile de date alese spre explorare și analiză vor surprinde astfel de procese. În particular, problemele propuse spre analiză fac parte din categoria celor de **clasificare a seriilor temporale**.

Sarcinile de lucru voastre vor solicita utilizarea de biblioteci de **vizualizare a datelor (crearea de diagrame)**, **extragerea de attribute (feature extraction)** pentru folosirea algoritmilor de clasificare discutați la curs, precum și utilizarea modelelor de rețele neurale convoluționale și recurente.

## 2. Descrierea Seturilor de Date

Veți folosi două seturi de date ce surprind serii temporale.

### PEMS-SF

PEMS-SF este un set de date oferit de California Department of Transportation si inclus în [arhiva seturilor de date UCI](#). Setul conține date zilnice înregistrate pe parcursul a 15 luni de către departamentul de transport al statului California pe site-ul [www.pems.dot.ca.gov](http://www.pems.dot.ca.gov).

Datele descriu gradul de ocupare, între 0 și 1, a diferitelor benzi auto de pe autostrăzile din zona golfului San Francisco. Măsurătorile acoperă perioada de la 1 ianuarie 2008 până la 30 martie 2009 și sunt prelevate la fiecare 10 minute. Fiecare zi din această bază de date este o singură serie temporală de dimensiunea 963 (numărul de senzori care au funcționat constant pe parcursul perioadei studiate) și lungimea  $6 \times 24 = 144$ .

Sărbătorile legale au fost eliminate din setul de date, precum și două zile cu anomalii (8 martie 2009 și 9 martie 2008) în care toți senzorii au fost dezactivați între 2:00 și 3:00 AM.

Rezultă o bază de date de 440 de serii temporale (267 pentru antrenare și 173 pentru test).

**Sarcina** este de a **clasifica fiecare zi observată ca zi corectă a săptămânii**, de luni până duminică, de ex. etichetându-o cu un număr întreg între 1,2,3,4,5,6,7 (i.e. **problema de clasificare multi-class cu 7 clase**).

**Notă:** Descărcarea setului de date PEMS-SF se face de la [aceasta adresa](#).

### UWaveGestureLibrary

Setul de date cuprinde exemple a 8 gesturi înregistrate folosind accelerometre (a se vedea figura de mai jos). Datele cuprind valori pe dimensiunile x, y, z corespunzând fiecărui gest în parte.

Pe fiecare dimensiune, seriile de timp au 315 de observații. Datele au fost descrise prima oară în [Liu et al., 2009]. Sunt **2238 de serii pentru antrenare** și **2241 pentru test**.









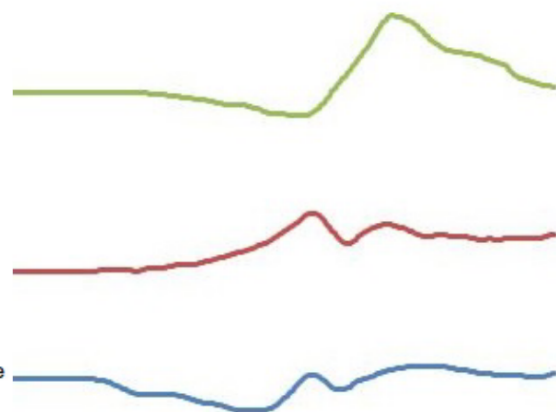
1	2	3	4
			
5	6	7	8
			

Figure 3: Gesture vocabulary adopted from [KKM+06]. The dot denotes the start and the arrow the end



Astfel, sarcina este de a clasifica serii de timp de dimensiune 3 (cele trei axe x, y și z) și lungime 315 în 8 clase de gesturi (etichete numerice de la 1 la 8).

**Notă:** Descărcarea setului de date UWaveGestureLibrary se face de la [aceasta adresa](#).

### Citirea seturilor de date

Seturile de date sunt furnizate în trei formaturi diferite: **text**, **.ts** (time series file for sktime), și **.arff** (format de date specific framework-ului [WEKA](#) pentru Machine Learning în Java).

Încărcarea lor (de exemplu, ca [DataFrame-uri în Pandas](#)) se poate face cu biblioteca [sktime](#). Sktime poate încărca atât formatul .ts, cât și .arff (un tutorial pentru acest lucru se [găsește aici](#))

**Notă:** Seturile de date sunt deja împărțite în train / test. Folosiți împărțirea furnizată.

## 3. Cerințe

### 3.1. Explorarea Datelor (Exploratory Data Analysis) [2p]

Primul pas cerut în rezolvarea unei probleme de clasificare este câștigarea unor cunoștințe asupra caracteristicilor principale ale problemei.

De regulă, foarte folositoare în această etapă este aplicarea unor metode de **vizualizare a datelor** și de **raportare a distribuțiilor de valori** pe fiecare variabila folosită în predicție.

#### Analize sugerate

##### 1. Analiza echilibrului de clase

Realizați un grafic al frecvenței de apariție a fiecărei etichete (clase) în setul de date de antrenare / test, folosind **barplot-uri** / **countplot** (a se vedea exemplul de mai jos).

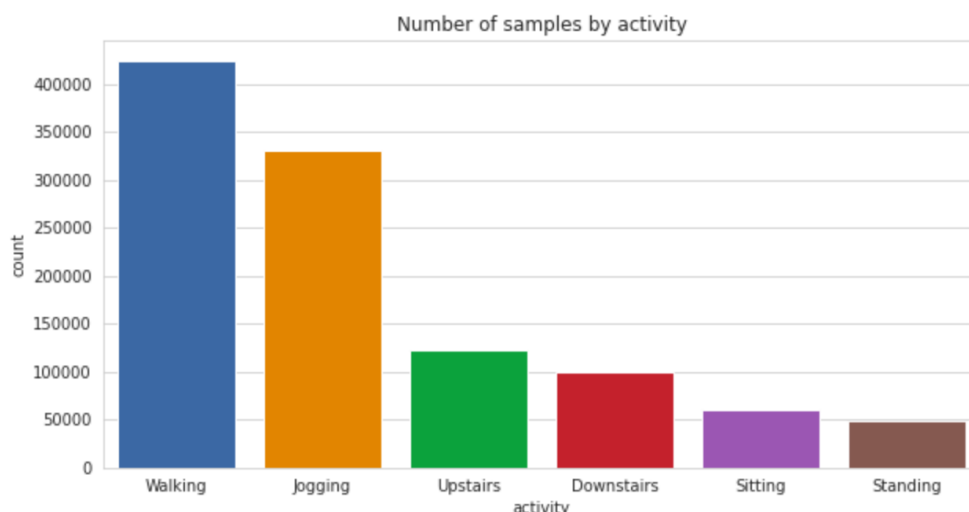


Fig. 1 Exemplu de utilizare a unui barplot pentru a vizualiza frecvența de apariție a claselor în setul de date.

Pentru realizarea unor astfel de barplot-uri puteți folosi mai multe biblioteci:

- Folosind biblioteca seaborn pentru [barplot](#) sau [countplot](#)
- Direct dintr-un DataFrame Pandas folosind [pandas.DataFrame.plot.bar](#)

Este evident de așteptat că pentru setul de date PEMS-SF această distribuție să fie relativ uniformă, având în vedere că etichetele reprezintă zile ale săptămânii contorizate pe o perioada contiguă.

Pe de altă parte, este de analizat la setul de date UWaveGestures dacă etichetele sunt echilibrate.

## 2. Diagrame sugerate pentru setul de date UWaveGestures

Setul de date UWaveGestures cuprinde serii de timp ale căror valori denotă poziții într-un pseudo-sistem de coordonate pe fiecare axa. Fiecare serie are 315 observații.

**Afișați câte un exemplu de serie** pentru fiecare tip de gest (a se vedea exemplul de graf din Figura 2). **Afișați valorile pe dimensiunile x, y și z pe același grafic.**

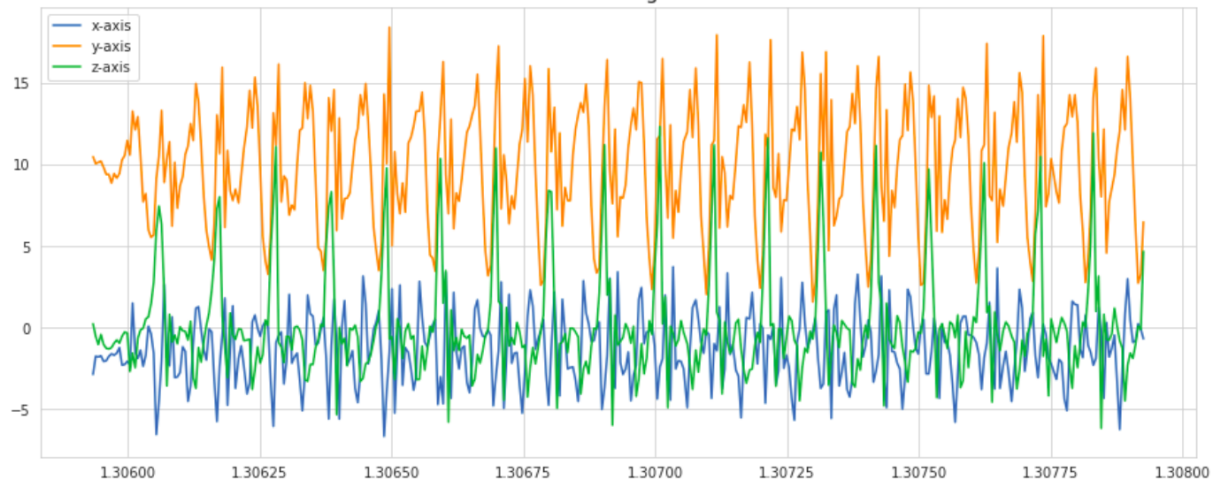


Figura 2. Exemplu de afișare a valorilor unei serii temporale.

**Afișați distribuția valorilor per fiecare axă în parte / per gest** (a se vedea Figura 3). O astfel de analiză este utilă pentru a determina dacă există niște șabloane imediate care se pot observa în termen de valori specifice pe x, y și z pentru fiecare gest în parte.

Aceste șabloane (e.g. intervale de valori specifice) pot fi folosite în etapa de definire a atributelor și pot totodată informa dacă problema este una ușoară (șabloane de valori clar diferențiabile per gest) sau grea (distribuții similare per fiecare gest).

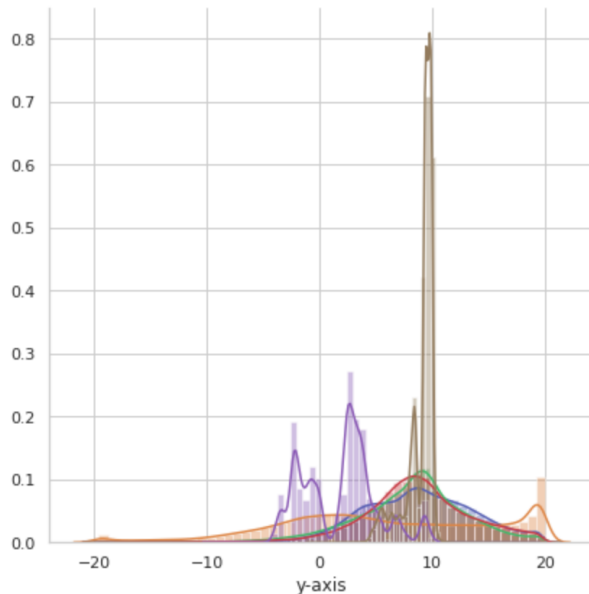


Figura 3. Exemplu de diagramă a distribuție de valori per axă, per gest.

Diagrame ca cea din Figura 3 pot fi obținute folosind plot-ul de tip [FacetGrid](#) din biblioteca seaborn. În particular, pentru diagrama în cauză, apelul este de forma:

```
sns.FacetGrid(df, hue = "<nume variabila target>", size = <numar
clase posibile>).map(sns.distplot, "<nume dimensiune x, y sau z>"
).add_legend()
```

Unde *<nume variabila target>* reprezintă numele atributului din setul de date ce identifică eticheta gestului, *<numar clase posibile>* este 8 pentru setul de date UWaveGestures, iar *<nume dimensiune x, y, sau z>* este numele atributului pe care îl dați la încărcare setului de 315 valori care corespund axei x, y sau z după caz.

### 3. Diagrame sugerate pentru setul de date PEMS-SF

Seriile de timp din setul de date PEMS-SF conțin valori în intervalul (0, 1), reprezentând rata de ocupare a unei benzi de circulație.

Prin urmare, aici ar fi relevantă vizualizarea unor metrici care ne pot da informații precum:

- Selectați 8 zile uniform distribuite de-a lungul anului de înregistrări. Generați un grafic de tip [boxplot](#) pentru a vedea gradul de variere a *ratei de ocupare*. Din cei 963 de senzori, faceți boxplot-uri pentru top-10 senzori ordonați descrescător după **deviația standard** a ratei de ocupare din cursul zilei.
- Pentru fiecare senzor calculați media zilnică a ratei de ocupare. Trasați **pe aceeași diagramă** un grafic al seriilor de timp ce corespund mediilor zilnice ale ratei de ocupare de-a lungul anului. Includeți în grafic top 10 serii de timp ale mediilor zilnice de ocupare, ordonate după **deviația standard a mediilor** calculată de-a lungul anului.

- Puteți repeta analiza această și împărțind calculând mediile per intervale orare (e.g. 0AM - 6AM, 6AM - 9AM, 9AM-12PM, 12PM - 3PM, 3PM-6PM, 6PM-9PM, 9PM-12AM)

Un grafic de tip **boxplot** este util pentru a înțelege distribuția valorilor și dacă aceasta poate fi folosită ca **atribut în clasificare**.

De exemplu, valorile de min, max, medie, *mediana* (50th [percentile](#)), 25th percentile, 75th percentile calculate pe baza numărului de activări (e.g. [pandas.DataFrame.describe\(\)](#)).

**Notă! Diagramele din această secțiune sunt cele *minimal cerute*:  
NU sunt singurele pe care le puteți face :-)**

## 3.2. Extragere manuala a atributelor și utilizarea algoritmilor clasici de Învățare Automată [4p]

### 3.2.1. Algoritmi propuși și evaluarea acestora [2p]

Pentru analiza celor două seturi de date veți folosi următorii algoritmi:

- RandomForest - folosiți [implementarea din scikit-learn](#)
- GradientBoosted Trees - folosiți [implementarea din biblioteca xgboost](#)
- SVM - folosiți [implementarea din scikit-learn](#)

Folosiți înțelegerea datelor câștigată la pasul 3.1 pentru a determina dacă este necesară [standardizarea datelor](#). Acest pas este unul des întâlnit etapă de pre-procesare a datelor înainte de antrenarea unui clasificator, în vederea uniformizării valorilor numerice aferente fiecărui tip de atribut (e.g. nu este dorit ca unele atribute să aibă valori de ordinul miilor, iar altele de ordinul unităților).

Partea de extragere a atributelor propusă în secțiunile 3.2.2 și 3.2.3 poate duce la un număr mare de atribute extrase (e.g. pentru PEMS-SF pot fi extrase atribute la nivelul fiecărui senzor). Frecvent se întâmplă ca nu toate atributele să aibă o contribuție importantă în cadrul predicției.

Ca atare, investigați aplicarea tehnicilor de [selectare a atributelor \(eng. Feature selection\) oferite în scikit-learn](#). Folosiți cel puțin una din metodele **Variance Threshold** sau **Select Percentile**.

Fiecare algoritm din cei propuși are o serie de **hiper-parametrii** care influențează funcționarea acestuia. Pentru a găsi valorile potrivite pentru aceștia veți folosi o procedură de **căutare a hiper-parametrilor** pe bază [de Grid Search cu Cross Validation](#).

Setul minim de hiper-parametrii de căutat este:

- SVM: tipul de kernel, parametru C de regularizare
- RandomForest: numărul de arbori, adâncimea maximă a unui arbore, procentul din input folosit la antrenarea fiecărui arbore
- GradientBoostedTrees: numărul de arbori, adâncimea maximă a unui arbore, learning rate

### Evaluarea algoritmilor

**În raportul vostru trebuie să prezentați următoarele:**

- Rezultatul procedurii de feature selection: numărul total de feature-uri considerate și numărul total de feature-uri utilizate la antrenare (ca urmare a procedurii de feature selection)
- Pentru fiecare algoritm, realizați un tabel în care să prezentați **media și varianța** pentru **acuratețea generală de clasificare, precizie / recall / F1 la nivelul fiecărei clase în parte**
  - Pe linii va fi indexată configurația de hiper-parametrii rezultată din procedura de GridSearch.
  - Pe coloane vor fi prezentate metricile cerute

- **Relevați prin bolduire** valorile maxime pentru fiecare metrică
- Pentru **cea mai bună variantă a hiper-parametrilor**, pentru fiecare algoritm, realizați o [matrice de confuzie](#) peste cele 8 clase

### 3.2.2. Extragerea atributelor pentru setul de date PEMS-SF [1p]

Datele din PEMS-SF indică rata de ocupare a unor benzi de circulație (exprimat ca valoare reală între 0 și 1) detectată de cei 963 de senzori, odată la fiecare 10 minute.

Scopul algoritmilor dezvoltati este să clasifice corect seria unei singure zile (conținând 144 de observații).

Ca atare, **sugestiile de attribute** de tip agregativ extrase la nivel de serie temporală zilnică (a fiecărui senzor) includ:

- Medie, deviație standard, min, max, mediană (50th [percentile](#)), 25th percentile, 75th percentile
- Diferența de valori maxime si minime
- Număr de vârfuri (număr de maxime locale - într-o vecinătate selectată)
- Asimetrie ([skewness](#))
- Curtoză ([kurtosis](#))

Agregările sugerate pot fi făcute la nivel de zi sau la nivelul intervalelor orare.

**NOTĂ:** indicațiile de mai sus vor genera attribute *per senzor* (rezultând potențial un număr foarte mare de attribute). Este esențial în acest caz să aplicați metode de Feature Selection (vedeți Secțiunea 3.2.1)

**NOTĂ:** Nu trebuie să vă limitați la aceste sugestii, sunteți încurajați să găsiți alte attribute care pot fi extrase din date.

### 3.2.3. Extragerea atributelor pentru setul de date UWaveGestures [1p]

Datele din UWaveGestures reprezintă valori continue pe fiecare axă ale unor gesturi cu 315 de observații per serie.

Ca atare, setul de attribute agregate pe care le putem extrage se referă la analiza statistică a *semnalelor* pe fiecare axă.

**Sugestii de attribute statistice** de extras per serie (sau sub-fereastră a seriei) / per axă:

- medie
- abaterea standard
- abaterea medie absolută
- valoare minimă
- valoare maximă
- diferența de valori maxime si minime
- mediană



- abaterea mediană absolută
- intervalul intercuartil
- Număr de valori negative
- Număr de valori pozitive
- număr de valori peste medie
- număr de vârfuri
- Energia semnalului
  - **Energia unui semnal** pe fiecare axă este calculată luând media sumei pătratelor valorilor dintr-o fereastră pe axa respectivă.
- Asimetrie (skewness)
- Curtoză (kurtosis)
- Accelerația medie rezultantă
  - **Accelerația medie rezultantă** peste fereastră este calculată luând media rădăcinilor pătrate ale valorilor din fiecare dintre cele trei axe pătrate și adunate împreună.
- Aria mărimii semnalului
  - **Aria mărimii semnalului** este definită ca suma valorilor absolute ale celor trei axe mediate pe o fereastră.

**În afară de attributele statistice**, pentru secvențe numerice interpretate ca semnale se pot calcula attribute extrase pe baza **interpretării în regim de frecvență** a semnalului (i.e. aplicând o **transformată Fourier**).

**Pentru exemple și cod** de extragere a atributelor, atât statistice cât și Fourier, [urmăriți acest tutorial](#).

**NOTĂ:** extragerile de attribute prezentate mai sus pot fi aplicate **pe toată lungimea seriei** sau pot fi aplicate pe **ferestre de lungime H**, unde  $H < \text{lungimea secvenței}$ . Aceasta înseamnă că puteți împărți secvența voastră în **subsecvențe (cu sau fără suprapunere)** și să calculați attributele pe fiecare subsecvență în parte. Pentru setul de date UWaveGestures (unde fiecare serie are lungime 315) puteți încerca:

- Împărțirea în 3 ferestre a câte 105 observații, fără suprapunere
- Împărțirea în 8 ferestre a câte 70 de observații, cu suprapunere de 35

### 3.3. Utilizarea modelelor de Rețele Neurale [4p]

**Veți aplica modele de rețele neurale pe setul de date UWaveGestures.**

Sunt propuse spre evaluare următoarele arhitecturi de rețele neurale:

- **Arhitectură de tip MLP** (Multi-Layered Perceptron) care primește ca intrare întreaga secvență a gestului [1p]
  - Experimentați cu **numărul de straturi și dimensiunea acestora**
- **Arhitectură de tip convoluțională**, folosind straturi de **convoluții 1D** și **global average pooling** [1.5p]
  - Puteți genera propria voastră arhitectură explorând:
    - Numărul de straturi de convoluție și dimensiunea canalelor (channels) a fiecăruia
    - Combinarea cu straturi liniare pentru partea finală a rețelei
    - Exemple de tutoriale găsiți [aici](#) și [aici](#).
  - Puteți folosi o arhitectură dată, **InceptionTime** [Fawaz et al., 2020], care utilizează straturi convoluționale 1D adaptând arhitectura Inception (definită pentru imagini) pe cazul seriilor de timp.
    - [Implementare în Pytorch pentru InceptionTime](#)
    - [Impelementare în Keras pentru InceptionTme](#)
    - Sugestii:
      - Folosiți 1 modul de Inception (în loc de default-ul de 2 din paper)
      - Variați dimensiunea kernelelor de convoluție
- **O arhitectură de tip recurent (LSTM)** [1.5p]
  - [Exemplu de tutorial](#) (folosiți doar partea de LSTM)
  - Sugestii:
    - Variați numărul de celule LSTM folosite (e.g. 1 sau 2)
    - Variați dimensiunea hidden size-ului din celula LSTM
    - Variați între un strat LSTM și BiLSTM (LSTM bidirecțional)

**NOTĂ:** datele pe care le aveți la antrenare sunt relativ puține din punct de vedere numeric. **Luați în considerare** (a se observa și în tutorialele referențiate) utilizarea metodelor de regularizare, e.g. prin utilizarea straturilor de **Dropout** sau prin utilizarea unui mecanism de weight decay în optimizator (a se vedea [detalii aici](#)).

**NOTĂ:** în afară de arhitectura în sine, performanța unui model neural este influențată și de **optimizatorul ales și de parametrizarea acestuia. Sugestii:**

- Folosiți un optimizator adaptiv (e.g. ADAM) și unul cu rată de învățare (learning rate) configurabilă (e.g. SGD cu momentum și un [Learning Rate Scheduler](#))
- Explorați influența **dimensiunii batch-urilor**
- Explorați influența numărului de epoci de antrenare

## **Evaluarea Modelelor**

**În raportul vostru trebuie să includeți următoarele:**

- Pentru fiecare model trebuie inclusă o detaliere a setup-ului de antrenare:
  - Descrierea arhitecturii folosite
  - Descrierea **parametrilor de optimizare** (optimizator folosit, batch size, learning rate, learning rate scheduler, weight decay)
- Afișați **pe același grafic** curbele de loss pentru **antrenare și test**
  - Dacă ați evaluat mai multe *variațiuni* ale aceleiași arhitecturi generale (e.g. un model LSTM cu hidden size diferit, un model conv1D cu dimensiune diferită a kernelului de convoluție), trasați curbele de loss la antrenare și test pentru fiecare variație **pe același grafic**, astfel încât să se poată observa diferențele între ele
- Creați un **tabel** (similar specificării de la punctul 3.2.1) în care să indexați **pe linii** configurațiile arhitecturale și de optimizare încercate, iar pe coloane metricile de performanță (acuratețea generală de clasificare, precizie / recall / F1 la nivelul fiecărei clase în parte)
- Creați matricea de confuzie pentru clasificarea celor 8 gesturi

## **3.4. Criterii Bonus**

**Se va acorda un bonus de 2p pentru oricare din următoarele:**

- Extragerea unor atribute diferite de cele propuse și **explicarea (obiectivă) a relevanței lor** (e.g. prin a arată ca metodele de feature selection includ atributul construit în selecție)
- Utilizarea la punctul 3.2 a unor algoritmi de [clasificare a seriilor temporale din suita pyts](#) și compararea lor cu implementările de la 3.2 și 3.3
- Utilizarea unei model de rețea neurală pentru setul de date PEMS-FS și compararea acestuia cu modelele propuse la punctul 3.2

## 4. Etapele predării temei

Tema este gândită ca un proiect și va avea următoarele etape de predare:

### 1. Etapa 1: 01/04 - 22/04 (deadline hard)

- Crearea unui raport sub formă de **fișier PDF**, care include:
  - i. **Cerința 3.1 (EDA) completă**, incluzând toate vizualizările și statisticile cerute. **Este obligatorie** prezența în text a **unei interpretări / analize** a diagramei rezultate.
  - ii. **Cerința 3.2 parțială**: va include raportarea extragerii de atribute și a evaluării algoritmilor de clasificare **pentru cel puțin unul** din cele două seturi de date propuse. **Este obligatorie** prezența în text a **unei interpretări / analize** a rezultatelor obținute (e.g. care atribute sunt cele mai predictive, cât de puternic este impactul hiper-parametrilor asupra performanței fiecărui algoritm considerat, care sunt clasele cu cele mai bune predicții).

### 2. Etapa 2: 27/04 - 22/05 (deadline soft, deadline hard +3 zile)

- Completarea raportului inițial cu
  - i. **Cerința 3.2 completă**: va include o raportare similară celei din prima etapă pentru setul de date rămas de analizat. Se aplică aceleași cerințe ca în etapa 1.
  - ii. **Cerința 3.3 completă + Bonus**: va include raportarea asupra performanței modelelor de rețele neurale, precum și a modelelor încercate a bonusului, după caz. **Este obligatorie** prezența în raport a unei interpretări analize a graficelor și statisticilor de antrenare și testare a modelelor neurale (e.g. explicație asupra configurației arhitecturale celei mai bune, analiză a gradului de influență al parametrizării optimizatorului asupra performanței).

**Rapoartele cerute pentru fiecare etapă se vor încărca în secțiunea aferentă pe Moodle, până la finalul deadline-ului corespunzător etapei.**

Rezultatele etapei 1 vor fi prezentate în cadrul laboratoarelor de Învățare Automată, ținute între deadline-ul etapei 1 și deadline-ul etapei 2. **Discuția la laborator se va face exclusiv pe baza rapoartelor încărcate.**

Rezultatele etapei 2 vor fi prezentate în săptămânile dedicate recuperărilor de laborator și a prezentărilor de teme (ultimele două săptămâni din semestru).

## Referințe

[Liu et al., 2009] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uwave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 657–675, 2009

[Fawaz et al., 2020] Ismail Fawaz H, Lucas B, Forestier G, Pelletier C, Schmidt DF, Weber J, Webb GI, Idoumghar L, Muller PA, Petitjean F. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*. 2020 Nov;34(6):1936-62.