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 atribute ale datelor problemei pentru a fi utilizate în obiectivul de analiza ales (e.g. clasificare, regresie, detectie de anomalii)
- Investigarea de modele de rețele neurale potrivite pentru extragerea "automată" de atribute 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 voastre de lucru vor solicita utilizarea de biblioteci de **vizualizare a datelor (crearea de diagrame)**, **extragerea de atribute (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.

RacketSports
RacketSports este un set de date ce conține înregistrări de smart watch făcute în timpul unui joc de badminton sau squash. Datele descriu semnale de accelerație (pe axele x, y și z) și giroscop (rotație pe axele x, y și z), surprinzând înregistrări de 3 secunde etichetate ca reprezentând tipul de mișcare din joc (*forehand / rever* pentru squash și *clear / smash* pentru badminton).
Senzorii sunt eșantionați cu o frevență de 10 Hz, astfel că fiecare secvență din setul de date are o lungime de 30. Există 151 de secvențe în setul de date de antrenare și 152 în setul de test.
Scopul este cel de a clasifica fiecare secvența de valori inerțiale ca una din cele 4 acțiuni (forehand, rever, clear, smash).
Notă: Descărcarea setului de date RacketSports se face de la aceasta adresa.

ECG Heartbeat Categorization Dataset

Acest set de date este compus din două colecții de semnale ale bătăilor inimii derivate din două seturi de date celebre în clasificarea bătăilor inimii: **setul de date MIT-BIH pentru aritmii** și **baza de date PTB Diagnostic** pentru **ECG**. Numărul de secvențe din ambele colecții este suficient de mare pentru antrenarea clasificatorilor pe bază de **rețele neurale**.
Semnalele corespund formelor electrocardiogramei (ECG) ale bătăilor inimii pentru cazul normal și cazurile afectate de diferite aritmii și infarct miocardic. Aceste semnale sunt preprocesate și segmentate, fiecare segment corespunzând unei bătăi de inimă.

Setul de date MIT-BIH Arrhythmia propune un task de clasificare in 5 clase (tipuri de aritmii), pe cand PTB Diagnostic ECG propune o clasificare binara (bătaie normala sau anormala a inimii).

Pentru ambele seturi de date, un segment are 188 de valori, dintre care unele valori pot fi *pad-uri* finale cu valori de 0.
Pentru ambele seturi de date, ultima coloana din cele 188 reprezinta eticheta tipului de aritmie (intre 0-4 pentru MITBIH, sau 0/1 - normal/anormal - pentru PTB).

Scopul este acela de a implementa algoritmi de clasificare a segmentelor de bătăi de inimă.

Notă: Descărcarea setului de date ECG Heartbeat Categorization Dataset se face de la aceasta adresa.

Citirea seturilor de date

Seturile de date sunt furnizate in două formaturi diferite: **.csv**, si **.arff** (format de date specific framework-ului WEKA pentru Machine Learning in Java). CSV-urile se pot citi ușor în Pandas. Pentru formatul .arff citirea se poate face cu biblioteca sktime (un tutorial pentru acest lucru se găsește aici)

Notă: Seturile de date RacketSports și MIT-BIH sunt deja împărțite în train / test. **Folosiți împărțirea furnizată.**
Pentru setul de date PTB sunteți liber să faceți o împărțire proprie a datelor în train și test.

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**.
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

2. Vizualizarea seriilor de timp

1) **Afișați câte un exemplu de serie** pentru **fiecare tip de acțiune** (în setul de date RacketSports)
    a) **Afișați valorile de accelerometru pe dimensiunile x, y și z pe același grafic.**
    b) Afișați valorile de giroscop pe dimensiunile x, y și z pe același grafic.

2) **Afișați câte un exemplu de serie** pentru fiecare **categorie de aritmie** din seturile de date MIT-BIH / PTB.

3) Pentru seturile de date cu aritmii afișați un grafic al **mediei și deviației standard per unitate de timp**, pentru fiecare clasă de aritmie. Media și deviația standard se calculează peste toate exemplele (atât din train, cât și din train set).

4) **Pentru setul de date RacketSports afișați distribuția valorilor per fiecare axă de accelerometru și giroscop în parte / per acțiune** (a se vedea un exemplu fictiv de output în Figura 1)**.** 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).

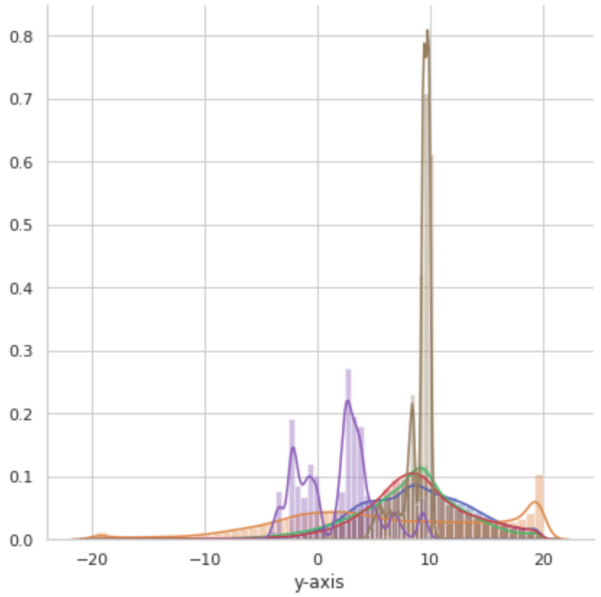

Figura 1. Exemplu de diagramă a distribuție de valori per axă, per gest.

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>* reprezinta numele atributului din setul de date ce identifică eticheta gestului, *<numar clase posibile>* este 4 pentru setul de date RacketSport, iar *<nume dimensiune x, y, sau z>* este numele atributului pe care îl dați la încărcare setului de 300 valori care corespund axei x, y sau z după caz.

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 **[0.5p]**
- GradientBoosted Trees - folosiți implementarea din biblioteca xgboost **[0.75p]**
- SVM - folosiți implementarea din scikit-learn **[0.75p]**

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 sa aibă valori de ordinul miilor, iar altele de ordinul unităților).

Partea de extragere a atributelor propusă în secțiunea 3.2.2 poate duce la un număr mare de atribute extrase. 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 sa 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 si 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 clase.

3.2.2. Extragerea atributelor [2p]

A. Extragere de atribute pentru RacketSports [1p]

Datele din RacketSports reprezinta valori continue pe fiecare axă ale unor acțiuni cu 300 de observații per serie. Ca atare, setul de atribute agregate pe care le putem extrage se referă la analiza statistică a *semnalelor* pe fiecare axă.

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

- medie
- abaterea standard
- abaterea medie absolută
- valoare minimă
- valoare maximă
- diferenta 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 atributele statistice, pentru secvențe numerice interpretate ca semnale se pot calcula atribute 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.

B. Extragere de atribute pentru Seturile de date cu Aritmii [1p]

Pentru seturile MIT-BIH si PBT veți încerca să antrenați algoritmii de la punctul 3.2.1 folosind:

- Direct datele de intrare (fiecare din cele 187 de puncte ale unei serii reprezentând bătaia inimii este un atribut)
- Atributele statistice descrise la litera **A.**

NOTĂ: extragerile de atribute prezentate mai sus pot fi aplicate **pe toată lungimea seriei** sau pot fi aplicate pe **ferestre de lungime H**, unde H < lungimea secvenței. Aceasta înseamnă că puteți împărți secvența voastră în **subsecvențe (cu sau fără suprapunere)** și să calculați atributele pe fiecare subsecvență în parte.

3.3. Utilizarea modelelor de Rețele Neurale [4p]

NOTA! Veți aplica modele de rețele neurale pe seturile de date cu aritmii ale bătăilor de inimă (MIT-BIH și PBT).

Sunt propuse spre evaluare următoarele arhitecturi de rețele neurale:

- **Arhitectură de tip MLP** (Multi-Layered Perceptron) care primește ca intrare întreaga secvență a unei bătăi de inimă **[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**, care utilizează straturi convoluționale 1D adaptând arhitectura Inception (definită pentru imagini) pe cazul seriilor de timp.
    - Implementare în Pytorch pentru InceptionTime
    - Impelemtare 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 sau BiLSTM) [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 influență **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 kernelelor 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 specificației 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 5 tipuri de aritmii (MIT-BIH) și pentru clasificarea între normal și anormal a datelor din PTB.

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 la punctul 3.2 a unor metode de gestionare a dezechilibrului de clase (class imbalance) prezent în setul de date MIT-BIH și în PTB (e.g. tutorial).
- Utilizarea la punctul 3.3 a unei metode de antrenare care exploatează **ambele seturi de date (MI-BIH și PBT)** și obține rezultate mai bune la test decât metodele care folosesc fiecare training set separat.

4. Etapele predării temei

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

1. **Etapa 1: 04/04 - 30/04 (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 diagramelor rezultate.
        ii. **Cerința 3.2 completă:** va include raportarea extragerii de atribute și a evaluării algoritmilor de clasificare pentru cele două tipuri de 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: 01/05 - 24/05 (deadline HARD)**
    - 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).