Proiect PCLP3

Mihai Nan[*], Andrei-Daniel Voicu, George Alexandru Tudor
Departamentul de Calculatoare

30 aprilie 2025

Notă!

Proiectul poate fi realizat individual sau în echipă de 2 studenți. În cazul în care alegeți să realizați proiectul individual, atunci veți realiza doar prima parte. Dacă alegeți să realizați proiectul în echipă, atunci fiecare student își va alege o parte.

Cuprins

[*]mihai.nan@upb.ro

1 Introducere

Python a devenit indispensabil în domeniul Data Science și Inteligența Artificială, mai ales în explorarea datelor și dezvoltarea de modele, datorită ecosistemului său bogat de biblioteci specializate. Cu biblioteci precum `numpy`, `pandas`, `matplotlib`, `seaborn`, Python oferă instrumente puternice pentru manipularea, vizualizarea și analiza datelor statistice. Aceste biblioteci permit cercetătorilor și analiștilor să exploreze seturile de date, să identifice modele și tendințe, și să obțină înțelegeri profunde din datele brute.

2 Partea I – Construirea și explorarea unui dataset tabelar

În prima parte a proiectului, veți construi un set de date tabelar propriu, folosind una dintre următoarele metode:

1. Extragerea de informații de pe internet prin tehnici de web scraping sau utilizarea de API-uri publice (ex: OpenWeatherMap, SpaceX Launch Data etc.);
2. Generarea sintetică a unui dataset care să aibă sens contextual (ex: simularea datelor medicale despre pacienți, date financiare despre companii, date despre specii de plante/animale);
3. Pornirea de la un dataset simplu de pe platforme precum Kaggle și extinderea acestuia prin adăugarea de noi coloane calculate, simularea de date lipsă, generarea de etichete pentru clasificare sau introducerea de zgomot.

2.1 Cerințe detaliate

1. **Tipul problemei:** setul de date propus trebuie să fie destinat fie unei probleme de regresie, fie unei probleme de clasificare. Tipul problemei trebuie specificat clar în documentație.
2. **Structura setului de date:** Setul de date final trebuie să fie împărțit în două subseturi:
   - Subset de antrenare: cel puțin 500 de instanțe (rânduri);
   - Subset de testare: cel puțin 200 de instanțe (rânduri).

   Împărțirea poate fi realizată prin extragere randomizată (ex: folosind funcții din `scikit-learn` sau `pandas`).
3. **Numărul minim de caracteristici:** fiecare instanță trebuie să aibă minimum 8 coloane relevante, inclusiv coloana țintă. Pentru aceste coloane veți selecta cel puțin 3 tipuri diferite de date (spre exemplu: numere întregi, numere reale, valori categoriale, șiruri de caractere etc.).
4. **Salvare dataseturilor:** Exportul subsetului de antrenare și al subsetului de testare în fișiere CSV separate.
5. **Documentare:** Explicarea clară a modului de construcție a setului de date: surse, metode de generare, eventuale ipoteze sau procesări suplimentare.

6. **Analiza exploratorie a datelor (EDA complex):** Se va realiza o explorare detaliată pentru fiecare dintre cele două subseturi propuse (cel de antrenare și cel de testare) care să includă obligatoriu:
   a) **Analiza valorilor lipsă:** număr și procent de valori lipsă pe coloană; strategii de tratare a acestora (ex: imputare, ștergere).
   b) **Statistici descriptive:** utilizarea `describe()` și interpretarea principalelor statistici pentru variabile numerice și categorice.
   c) **Analiza distribuției variabilelor:**
      - Histogramă pentru fiecare caracteristică numerică;
      - Grafice de tip countplot/barplot pentru variabilele categorice.
   d) **Detectarea outlierilor:** Utilizarea boxplot-urilor sau altor tehnici (ex: IQR rule) pentru identificarea valorilor *aberante*.
   e) **Analiza corelațiilor:** Matrice de corelații (heatmap) pentru variabilele numerice.
   f) **Analiza relațiilor cu variabila țintă:** Scatter plots sau violin plots pentru relația dintre caracteristici și variabila țintă (în funcție de tipul problemei).
   g) **Comentarii și interpretări personale:** Fiecare grafic trebuie însoțit de o scurtă interpretare textuală care să răspundă la următoarele întrebări:
      - Ce observăm?
      - Ce suspiciuni/idei putem formula?
      - Ce preprocesări ar trebui să aplicăm?
7. **Antrenarea și evaluarea unui model de bază:** Se va antrena un model simplu din biblioteca `scikit-learn`, potrivit pentru problema aleasă:
   - Exemplu: regresie liniară pentru probleme de regresie, logistic regression sau random forest pentru probleme de clasificare.
   - Modelul va fi antrenat pe subsetul de antrenare și evaluat pe subsetul de testare.
   - Se vor raporta și interpreta rezultatele utilizând metrici adecvate:
     a) Pentru regresie: RMSE, MAE sau R^2.
     b) Pentru clasificare: acuratețe, precizie, recall, F1-score.
   - Se vor include grafice relevante pentru performanța modelului (ex: matrice de confuzie, grafice de erori etc.).

3 Partea a II-a – Prelucrarea avansată a datelor și compararea modelelor

În această parte a proiectului, se va continua lucrul pornind de la datasetul realizat de colegul de echipă la Partea I (2).

3.1 Cerințe detaliate

1. **Prelucrarea datelor:** Se vor aplica tehnici de prelucrare a datelor, după caz:
   - Normalizare sau standardizare pentru variabilele numerice;
   - Encodare pentru variabilele categorice (ex: `OneHotEncoder`, `LabelEncoder`);
   - Înlocuirea valorilor lipsă prin metode adecvate (ex: medie, mediană, modă, imputare avansată).

   Se va explica pentru fiecare tehnică aleasă de ce a fost necesară și ce impact are asupra modelelor de machine learning (pe baza rezultatelor obținute pentru rularea modelului cu sau fără aplicarea tehnicii).

2. **Analiza exploratorie a datelor (EDA complex) după aplicarea prelucrărilor:** Se va realiza o explorare detaliată pentru fiecare dintre cele două subseturi propuse (cel de antrenare și cel de testare) care să includă obligatoriu:

   a) **Analiza valorilor lipsă:** număr și procent de valori lipsă pe coloană; strategii de tratare a acestora (ex: imputare, ștergere).

   b) **Statistici descriptive:** utilizarea `describe()` și interpretarea principalelor statistici pentru variabile numerice și categorice.

   c) **Analiza distribuției variabilelor:**
      - Histogramă pentru fiecare caracteristică numerică;
      - Grafice de tip countplot/barplot pentru variabilele categorice.

   d) **Detectarea outlierilor:** Utilizarea boxplot-urilor sau altor tehnici (ex: IQR rule) pentru identificarea valorilor *aberante*.

   e) **Analiza corelațiilor:** Matrice de corelații (heatmap) pentru variabilele numerice.

   f) **Analiza relațiilor cu variabila țintă:** Scatter plots sau violin plots pentru relația dintre caracteristici și variabila țintă (în funcție de tipul problemei).

   g) **Comentarii și interpretări personale:** Fiecare grafic trebuie însoțit de o scurtă interpretare textuală care să răspundă la următoarele întrebări:
      - Ce observăm?
      - Ce suspiciuni/idei putem formula?
      - Ce preprocesări ar trebui să aplicăm?

   Puteți colabora cu colegul de echipă și să preluați codul realizat de el pe care să-l adaptați pentru a putea fi aplicat pe seturile de date procesate. Identificați și documentați eventualele limitări / puncte tari ale dataset-ului produs de colegul de echipă.

3. **Antrenarea și compararea a cel puțin 3 algoritmi diferiți:** Se vor alege minimum 3 modele diferite din biblioteca `scikit-learn`, potrivite tipului de problemă (regresie sau clasificare). Exemple:
   - Pentru regresie: Linear Regression, Ridge Regression, Decision Tree Regressor, Random Forest Regressor, SVR etc.

- Pentru clasificare: Logistic Regression, Decision Tree Classifier, Random Forest Classifier, SVM, KNN etc.

Fiecare model va fi antrenat pe datele prelucrate (subsetul de antrenare) și evaluat pe subsetul de testare.

4. **Evaluarea performanței:** Performanțele modelelor se vor compara utilizând aceeași metrică (aleasă de voi) pentru toate modelele:
   - Regresie: RMSE, MAE, R^2 etc.
   - Clasificare: acurateță, F1-score, ROC AUC etc.

   Se va construi un tabel comparativ care să includă: numele algoritmului, valorile obținute pentru fiecare metrică relevantă.

   În plus, pentru o analiză mai detaliată:
   - În cazul problemelor de clasificare, veți reprezenta grafic matricea de confuzie pentru fiecare model și, dacă este relevant, curbele ROC.
   - În cazul problemelor de regresie, veți include diagrame de tip *scatter plot* (valoare reală vs. valoare prezisă) și/sau distribuția reziduurilor.

   Aceste vizualizări trebuie să ajute la interpretarea performanței și la identificarea eventualelor puncte slabe ale fiecărui model.

4 Bonus – Interfață grafică

Indiferent de ce parte alegeți să rezolvați, se poate dezvolta o interfață grafică pentru acest proiect, care să permită utilizatorului să introducă valori pentru variabilele de intrare și să vizualizeze predicțiile și performanțele modelelor de învățare automată. Iată câteva sugestii pentru implementarea acesteia:

- **Input pentru date:**
  - Crează câmpuri de introducere a valorilor pentru variabilele numerice și categorice (de exemplu, câmpuri de text pentru valori numerice și dropdown pentru variabilele categorice).
  - Include un buton de "Predicție" care să preia datele introduse și să le paseze unui model antrenat.
- **Predicție și vizualizare:**
  - După ce utilizatorul apasă butonul de predicție, aplicația poate rula modelele selectate (de exemplu, regresie sau clasificare) și să afișeze predicțiile pentru fiecare algoritm.
  - În cazul clasificării, ar putea fi afișate probabilitățile fiecărei clase.
  - În cazul regresiei, ar putea fi afișată valoarea prezisă și o comparație cu valoarea reală, dacă există.

  Vizualizări grafice:

- Clasificare: Afișează matricea de confuzie pentru fiecare model, utilizând o funcție de plotare. Dacă este cazul, adaugă curbele ROC pentru fiecare model.
- Regresie: Creează un scatter plot cu valorile reale față de valorile prezise și/sau distribuția reziduurilor pentru fiecare model.

5 Punctaj

Proiectul va fi încărcat pe Moodle, de fiecare membru al echipei (fiecare student își încarcă partea lui), sub forma unei arhive **.zip** cu următorul conținut:

- un director cu numele **ParteaI** (**pentru cei care au ales Partea I** – 2) ce conține următoarele subdirectoare:
  - **Surse** - toate fișierele cu cod folosite în realizarea temei (`.py` / `.ipynb`)
  - **README.pdf** - fișierul care conține toate histogramele rezultate, toate graficele create și documentația.
  - **train.csv** și **test.csv** - datele care compun cele două subseturi ale setului de date realizat

  În fișierul fișierul README veți descrie modul de rezolvare pentru fiecare cerință din Partea I, răspunsurile la întrebările din cerință și alte observații; prima linie a fișierului va conține numele complet, seria și grupa studentului care a rezolvat partea I.
- un director cu numele **ParteaII** (**pentru cei care au ales Partea a II-a** – 3) ce conține următoarele subdirectoare:
  - **Surse** - toate fișierele cu cod folosite în realizarea temei (`.py` / `.ipynb`);
  - **Date** - toate fișierele rezultate din modificări / prelucrări ale setului de date;
  - **README.pdf** - fișierul care conține toate graficele create și documentația.

  În fișierul README veți descrie modul de rezolvare pentru fiecare cerință din Partea a II-a, răspunsurile la întrebările din cerință și alte observații; prima linie a fișierului va conține numele complet, seria și grupa studentului care a rezolvat partea a II-a.

Punctajul pentru fiecare parte este împărțit după cum urmează:

Partea I	Punctaj
Cerințe	50 puncte (40% cod + 40% rezultat + 20% documentație)
BONUS folosire Git	20 puncte
Realizare interfață grafică	20 puncte
TOTAL MAXIM ACORDAT	**70 puncte**

Partea a II-a	Punctaj
Cerințe	50 puncte (40% cod + 40% rezultat + 20% documentație)
BONUS folosire Git	20 puncte
Realizare interfață grafică	20 puncte
TOTAL MAXIM ACORDAT	**70 puncte**

Atenție!

- Pentru a primi punctaj, trebuie să **prezentați** proiectul în ultima săptămână a semestrului.
- **Toate soluţiile trimise vor fi verificate, folosind o unealtă pentru detectarea plagiatului. În cazul depistării codului copiat (de pe Internet, colegi, din surse generate cu tool-uri tip ChatGPT), întregul punctaj pentru proiect este anulat.**
- Pentru orice întrebare puteți folosi forumul.
- Punctajul bonus pentru folosirea utilitarului `git` este acordat raportat la numărul de cerințe realizate și la complexitatea funcționalităților utilizate.
- Pentru bonus alegeți dacă îl primiți pentru utilizarea `git` sau pentru realizarea unei interfețe grafice.