Version 1.0
10 Noiembrie, 2024

Documentul de Proiectare a Soluţiei Aplicaţiei Software

[SDD]

Proiect software ce vizează analiza experimentală a unui set de date medicale

Realizat de ↓

- Mihălucă Mădălina-Maria
- Popa Andrei
- Balan Iulia
- Maieczki Petronela-Sînziana
- Cîrja Ioan
- Butnaru Raimond Eduard
- Pintilie Justinian
- Florea Alexandra

Opis

1. Scopul documentului

1.1. Scurtă descriere

Acest document descrie soluția propusă pentru analiza și clasificarea unui set de date clinic privind hepatita. Scopul principal este de a oferi o metodologie structurată pentru pre-procesarea, explorarea, analizarea și clasificarea datelor. Documentul servește drept ghid unic de construire a soluţiei pentru echipa de dezvoltare a proiectului, explicitând toți pașii de la manipularea inițială a datelor și procesarea statistică până la implementarea algoritmilor de clasificare și evaluarea performanței modelului. Astfel, documentul va susține echipa în aplicarea celor mai bune practici în analiza datelor medicale și în atingerea obiectivelor de performanță stabilite pentru acest proiect.

1.2. Lista de obiective

- **Structurarea setului de date**

Asigurarea că setul de date este adecvat pentru analiza ulterioară prin prelucrarea și curățarea acestuia, inclusiv manipularea valorilor lipsă, normalizarea variabilelor și convertirea atributelor pentru utilizarea lor în model.

- **Analiza statistică a datelor**

Calcularea măsurilor descriptive, cum ar fi media, dispersia, valorile minime și maxime, pentru identificarea și corectarea eventualelor anomalii și evaluarea omogenității setului de date.

- **Divizarea setului de date pentru antrenare și testare**

Crearea de seturi de date echilibrate prin divizarea aleatoare a eșantioanelor în diverse proporții (80%-20%, 70%-30%, etc.), asigurând menținerea distribuției caracteristicilor în ambele seturi.

- **Construirea unui model de clasificare**

Implementarea unui algoritm de clasificare bazat pe Random Forest pentru a diferenția între clasele de pacienți, testând performanța acestuia în contextul setului de date.

- **Selecția trăsăturilor relevante**

Identificarea trăsăturilor semnificative din setul de date pentru îmbunătățirea preciziei modelului, analizând beneficiile și limitările reducerii dimensiunilor în contextul proiectului.

- **Evaluarea performanței modelului**

Validarea și optimizarea rezultatelor obținute pe baza indicatorilor de performanță, precum acuratețea, precizia și analiza vizuală a rezultatelor.

1.3. Definiții, acronime și abrevieri

MATLAB este un mediu de programare și o platformă numerică dezvoltată de MathWorks, folosită extensiv pentru calcul științific, inginerie, simulări și analiză de date.

CSV (Comma-Separated Values) regasit ca extensie de fisier. Aceste fișiere "nume.csv**"** sunt folosite pentru a stoca date într-un format simplu, organizat în rânduri și coloane. Datele din fiecare rând sunt separate prin virgule sau alte delimitatoare.Tipul acesta de fișier e frecvent utilizat în ML pentru pregătirea și manipularea seturilor de date.

ML(Machine Learning) face referire la algoritmi de învățare[1] automată

SVM (Support Vector Machine) este un algoritm de învățare automată pentru clasificare și regresie. În clasificare, SVM-ul găsește o hiperplană optimă care separă datele din două clase diferite astfel încât să maximizeze distanța dintre cele mai apropiate puncte de pe marginea fiecărei clase (numite support vectors). În alte cuvinte, SVM-ul ajută la găsirea limitei de decizie care separă cel mai bine două clase. Poate fi folosit și pentru seturi de date cu mai multe clase.

Mean(Media) este valoarea medie a unui set de date, calculată prin suma tuturor valorilor împărțită la numărul total de valori.

$$\sum_{i=1}^{n} \frac{x_i}{n} \quad *$$

[1] * unde i reprezintă iterația, x[i] reprezintă elementul la iterația i iar n reprezintă numărul de elemente

Dispersia /Variance este o măsură a variației datelor față de media lor. Cu cât dispersia este mai mare, cu atât valorile sunt mai îndepărtate de valoarea medie. Dispersia ajută la înțelegerea "răspândirii" valorilor dintr-un set de date.

$$\frac{\sum_{i=1}^{n}(x_i - Media)^2}{n}$$ **

Devierea standard /Standard Deviation este rădăcina pătrată a dispersiei și oferă o măsură a răspândirii datelor într-un mod mai intuitiv decât dispersia. O deviere standard mică sugerează că datele sunt mai apropiate de medie, în timp ce una mare indică o difuzie mai mare a datelor.[2]

Modul /Mode reprezintă valoarea care apare cel mai frecvent într-un set de date. Este util în analiza datelor când vrei să identifici valoarea cea mai comună.

Mediana /Median este valoarea de mijloc a unui set de date ordonat. Dacă numărul de date este impar, mediana este valoarea de la mijloc. Dacă este par, este media dintre cele două valori de la mijloc. Mediana este utilă pentru a elimina efectele valorilor extreme asupra mediei.

Normalizarea datelor /Normalization presupune ajustarea valorilor dintr-un set de date astfel încât acestea să se încadreze într-un anumit interval, de obicei între 0 și 1. Aceasta este o tehnică comună în preprocesarea datelor.

Train-Test Split/Împărțirea datelor în seturi de antrenament și test, reprezintă o metodă de validare a modelului. Setul de antrenament este folosit pentru a învăța modelul, iar setul de test pentru a evalua performanța modelului pe date noi, pentru a se evita overfitting-ul.

Overfitting /Supra-învățare, apare atunci când un model învață foarte bine setul de antrenament, dar are performanță scăzută pe setul de test, indicând că modelul nu generalizează bine.

Underfitting /Sub-învățare, apare atunci când modelul este prea simplu pentru a capta tiparele din date, având performanță slabă atât pe setul de antrenament, cât și pe setul de test.

[2] ** unde i reprezintă iterația, x[i] reprezintă elementul la iterația i iar n reprezintă numărul de elemente, iar Media=media elementelor

Confusion Matrix/Matricea de confuzie este un tabel folosit pentru a evalua performanța unui model de clasificare, indicând câte predicții au fost corecte și câte greșite. Este împărțită în patru categorii: true positives, true negatives, false positives, și false negatives.

2. Conţinutul documentului

Documentul este format din câteva secţiuni esenţiale, precum:

- **Modelul datelor** – prezintă principalele structuri de date folosite, precum şi schema bazei de date
- **Modelul arhitectural /Modelul componentelor** – prezintă arhitectura sistemului şi descrie componentele arhitecturii
- **Indicatori de performanta** – prezinta standardele de evaluare a eficienței și fiabilității sistemului.
- **Elemente de testare** – prezintă componentele critice şi alternative de proiectare a acestora

3. Modelul datelor

3.1. Structuri de date globale

Structura de date globală folosită este DataSet, care reprezintă setul complet de date medicale importate din fișierelul HepatitisC.csv. Aceasta este o variabilă globală de tip table sau array în MATLAB, care conține toate informațiile necesare pentru analiza ulterioară (de exemplu, datele demografice ale pacienților și rezultatele analizelor medicale).

Datorită rolului său central în aplicație, DataSet este accesibil tuturor componentelor sistemului, asigurându-se astfel că fiecare funcție implicată în preprocesare, analiză statistică, și clasificare utilizează aceleași date.

3.2. Structuri de date de legătură

Structura de date de legătură principală este SampleSubset, un subset din DataSet (structura de date globală). SampleSubset este creat pentru a transmite eșantioanele selectate între modulele de preprocesare și funcțiile de antrenare și evaluare a modelului. Astfel, funcțiile de pregătire a datelor și clasificare pot accesa și utiliza doar datele specifice necesare în fiecare etapă, fără a modifica structura de date globală.

3.3. Structuri de date temporare

Un exemplu ar putea fi dat de structura TempFilteredData, care este folosită temporar pentru a stoca datele care au fost curățate (ex. eliminarea valorilor lipsă sau corectarea erorilor din date). Această structură este utilizată doar în cadrul funcțiilor de preprocesare și nu este păstrată după ce datele sunt procesate.
Alte exemple pot include structura TempNormalizedData, care stochează datele normalizate pentru o scurtă perioadă de timp, necesară doar pentru antrenarea și evaluarea modelului. Aceste structuri de date sunt esențiale pentru a păstra informațiile intermediare necesare în procesul de analiză, fără a afecta structurile globale sau de legătură ale aplicației.

3.4. Formatul fişierelor utilizate

Fișierul utilizat pentru importul datelor este de forma HepatitisC.csv. Acest tip fișiere conțin informații despre pacienți, rezultate de analize medicale și alti biomarkeri. Fișierele .csv sunt structurate sub forma unor tabele, în care fiecare linie reprezintă un pacient, iar fiecare coloană reprezintă o variabilă (de exemplu, vârstă, sex, rezultate ale analizelor etc.).

Structura fișierului:

- Fiecare fișier .csv începe cu un header (prima linie) care conține numele coloanelor, fiecare coloană corespunzând unui atribut al pacientului (ex. ID pacient, vârstă, sex, diagnostic).
- Datele sunt separate prin virgule (sau alte caractere de delimitare, în funcție de specificațiile regionale), iar fiecare linie conține informațiile corespunzătoare unui pacient.
- Valorile lipsă sunt reprezentate prin celule goale sau cu un caracter special, precum isnan, pentru a semnala lipsa unor informații.

3.5. Descrierea datelor folosite

Fișierul este organizat din mai multe coloane, acestea fiind descrise în cele ce urmează:

- **Number**: Un identificator unic pentru fiecare pacient, care asigură confidențialitatea și facilitează urmărirea datelor individuale în analiză.

- **Category**: O etichetă care indică starea pacientului pe baza istoricului medical și a analizelor:
    - 0 = blood donor (donator de sânge sănătos, fără hepatită C),
    - 1 = hepatitis (pacient diagnosticat cu hepatită C activă),
    - 2 = fibrosis (pacient cu fibroză hepatică, o etapă a cicatrizării ficatului ca urmare a hepatitei C),
    - 3 = cirrhosis (pacient cu ciroză hepatică, o afecțiune gravă și avansată a ficatului),
    - 0s = suspectBloodDonor (donator de sânge suspectat de a fi purtător al virusului hepatitic).
- **Age**: Vârsta pacientului în ani. Vârsta poate influența evoluția bolii și răspunsul la tratament.

- **ALB (Albumină)**: O proteină produsă de ficat care ajută la menținerea presiunii osmotice și la transportul de diverse substanțe în sânge. Nivele scăzute de albumină pot indica o funcție hepatică deficitară, frecvent întâlnită în hepatită și ciroză.

- **ALP (Fosfataza alcalină)**: O enzimă asociată cu ficatul, oasele și alte organe. Nivele crescute de ALP pot semnala o afectare a funcției hepatice, blocaj biliar sau alte afecțiuni hepatice.

- **ALT (Alanina aminotransferază)**: O enzimă hepatică. Nivele ridicate de ALT indică leziuni sau inflamatii ale ficatului, fiind un marker comun în hepatita C și alte boli hepatice.

- **AST (Aspartat aminotransferază)**: O altă enzimă hepatică. Nivele crescute de AST pot sugera afectarea ficatului. Raportul AST/ALT poate ajuta la diferențierea între diferite tipuri de afecțiuni hepatice.

- **BIL (Bilirubină)**: Un pigment rezultat din degradarea hemoglobinei. Nivelele ridicate de bilirubină pot provoca icter și pot indica o funcție hepatică deficitară, frecvent întâlnită în hepatita C.

- **CHE (Colinesterază)**: O enzimă produsă de ficat. Nivelele scăzute de CHE pot reflecta deteriorarea funcției hepatice, fiind un indicator important în evaluarea afecțiunilor hepatice.

- **CHOL (Colesterol)**: Nivelul de colesterol din sânge. Ficatul este esențial în metabolismul colesterolului, iar bolile hepatice pot afecta nivelurile normale de colesterol.

- **CREA (Creatinină)**: Un produs de degradare a creatinei, eliminat de rinichi. În mod indirect, creatinina oferă informații despre funcția renală, importantă de monitorizat la pacienții cu hepatită C, deoarece afectarea ficatului poate influența și funcția renală.

- **GGT (Gamma-glutamil transferază)**: O enzimă implicată în metabolismul glutationului și asociată cu ficatul. Nivelele ridicate de GGT pot indica o boală hepatică sau un blocaj biliar și pot fi folosite pentru a evalua afectarea ficatului.

- **PROT (Proteine totale)**: Nivelul total al proteinelor din sânge, incluzând albumina și globulinele. Nivele anormale ale proteinelor pot sugera o funcție hepatică compromisă, aspect comun în bolile hepatice avansate, cum ar fi ciroza.

4. Modelul arhitectural /Modelul componentelor

4.1. Arhitectura sistemului

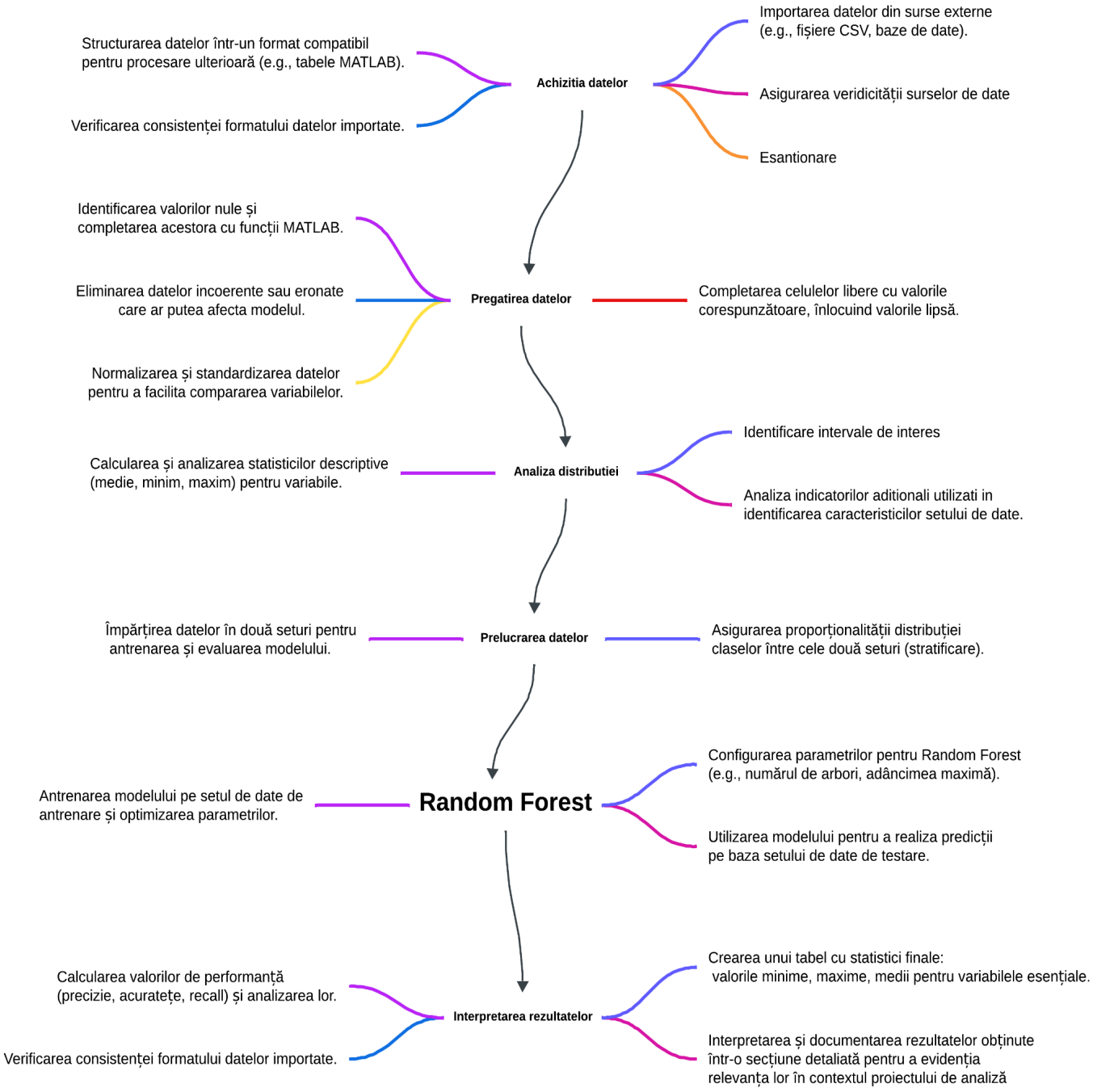

Fig.1. Ilustrarea diagramei secvențiale a componentelor

4.2. Descrierea componentelor

Diagrama secvențială din Fig.1. este alcătuită din următoarele componente:

4.2.1. Componenta de achiziție de date

Aceasta reprezintă în special o etapă de research, în care se stabilește setul sau seturile de date ce vor fi prelucrate. Mai presupune, de asemenea, asigurarea veridicității informațiilor, convertirea datelor într-un format compatibil procesării (csv, xls) și eșantionarea, ce are ca scop extragerea unui subset relevant de date. Se va utiliza setul de date **“Hepatitis C Prediction Dataset”,** disponibil la: https://www.kaggle.com/datasets/fedesoriano/hepatitis-c-dataset?resource=download. Acesta include date centralizate de laborator de la mai mulți donatori de sânge, în funcție de vârstă și sex.

O posibilă implementare pentru funcția utilizată pentru citirea setului de date:

```
function data = loadData(filepath)
  %continut relevant
end
```

unde:

- filepath -> calea către setul de date
- data -> variabilă în care sunt stocate datele de intrare

4.2.2. Componenta de pregătire a datelor

Pregătirea datelor este necesară pentru că prin intermediul ei se va face trierea datelor relevante, se elimină valorile nule sau irelevante, se înlocuiesc spațiile libere cu date corecte în cazurile posibile și se normalizează valorile pentru a fi mai ușoară compararea lor. Setul de date ales nu prezintă la prima vedere cazuri de date neconforme, nule sau irelevante, însă este necesară o căutare cu un algoritm specializat, care va utiliza metoda ismissing și rmmissing pentru a elimina liniile cu valori nule.

O implementare primitivă pentru funcția atribuită acestei componente este:

```
function data = cleanData(data)
        %continut satisfacator cerintelor
end
```
unde:

- data -> variabilă în care sunt stocate datele de intrare

4.2.3. Componenta de analiză a distribuției

Prin intermediul acesteia, se vor identifica intervalele de interes și se vor calcula statisticile de interes(medie, minim, maxim). În cazul acestui proiect se selectează valorile de interes ca fiind toate datele numerice și se vor calcula media(mean în Matlab), dispersia(std în Matlab), min, max pentru fiecare coloană.

O posibilă implementare pentru funcția utilizată în analiza distribuției poate fi:

```
function [meanVals, stdVals, minVals, maxVals] = computeStatistics(X)
        %continut relevant
end
```
unde:

- meanVals -> media fiecărei coloane din X
- stdVals -> dispersia fiecărei coloane din X
- minVals -> valoarea minimă pentru fiecare coloană din X
- maxVals -> valoarea maximă pentru fiecare coloană din X

Pentru afișarea statisticilor se poate pleca de la:

```
function displayStatistics(meanVals, stdVals, minVals, maxVals)
        %continut
 end
```

4.2.4. Componenta de prelucrare a datelor

Se va împărți setul de date în două seturi mai mici, unul pentru antrenarea și unul pentru testarea modelului.

Modelul din cadrul proiectului va fi antrenat cu secvențe mai mici din setul de date de 80%, 70%, 60% sau 50%, alese **random**, setul de testare reprezentând în fiecare caz, restul de 20%, 30%, 40%, respectiv 50%. Împărțirea setului de date în subseturi se face cu cvpartition.

Pentru împărțirea datelor în subseturi, se poate folosi ca exemplu:

```
function [XTrain, YTrain, XTest, YTest] = splitData
                %continut satisfacator cerintelor
end
```

unde:

- XTrain -> setul de date de antrenament.
- YTrain -> etichetele (sau variabila de ieșire) pentru setul de antrenament.
- XTest -> setul de date de testare
- YTest -> etichetele pentru setul de testare

4.2.5. Componenta de antrenare cu Random Forest

Random Forest este un algoritm de învățare automată, foarte utilizat în cadrul modelelor, bazat pe tehnica de bagging. Algoritmul construiește mai mulți arbori de decizie și folosește votul majoritar pentru a face o predicție finală. Tehnica de bagging presupune antrenarea și testarea modelului cu subseturi de date alese aleatoriu, ceea ce face ca fiecare arbore să aibă variații și să fie expus la diferite porțiuni ale setului de date. Fiind un algoritm ce antrenează arborii cu subseturi diferite, se ajunge ca modelul final să fie unul mai robust și precis(spre deosebire de algoritmii care au un singur arbore de decizie și pot suferi de overfitting.

În cadrul proiectului se poate pleca de la funcția:

```
function model = trainRandomForest(XTrain, YTrain, numTrees)
                    %continut
end
```

unde:

- *numTrees* -> numărul de arbori
- *XTrain* -> matricea de caracteristici
- *YTrain* -> matricea de etichete a setului de date

Pentru predicții pe setul de testare se poate utiliza ca punct de pornire:

```
function YPred = predictModel(model, XTest)
        %continut
end
```

unde:

- model -> este modelul antrenat
- XTest -> setul de date de testare pentru care se vor face predicții

4.2.6. Componenta de interpretare a rezultatelor

În cadrul acestei etape, se va face verificarea consistenței formatului datelor importante, se va calcula indicatorii de performanță, se vor crea tabele cu statistici finale și se va face o documentare a rezultatelor obținute.

Pentru a obține indicatorii de performanță, se va crea o funcție specială de calcul, tabele se vor face cu plot, iar documentarea și analiza rezultatelor se va face împreună cu întreaga echipă.

Un exemplu de funcție de calcul al indicatorilor de performanță este:

```
function [accuracy, recall, specificity, precision, f1score] =
computePerformance(YTrue, YPred)
        %continut relevant
end
```

unde:

- YTrue -> reprezintă etichetele adevărate (corecte)
- YPred -> reprezintă etichetele prezise de model

- accuracy -> acuratețea
- recall -> sensibilitatea
- specificity -> specificitatea
- precision -> precizia
- f1score -> medie armonică între precizie și sensibilitate

4.3. Restricţiile de implementare

- Aplicația trebuie să fie compatibilă cu versiunea de MATLAB specificată (de exemplu, MATLAB R2020a). Funcțiile folosite și sintaxa trebuie să fie conforme cu această versiune pentru a asigura portabilitatea și buna funcționare.

- Fișierele .csv importate nu trebuie să depășească o anumită dimensiune (de exemplu, 100 KB), pentru a evita problemele de performanță în MATLAB și a asigura o procesare eficientă. Fișierele prea mari ar putea încetini sistemul și ar necesita optimizări suplimentare.

- Datele din fișierele .csv trebuie să respecte formatul predefinit, adică ordinea coloanelor și tipurile de date, altfel procesarea datelor poate eșua sau poate conduce la rezultate incorecte. Este necesar ca fișierele să aibă o linie de antet și să folosească un delimitator standard, in cazul nostru: virgula.

- Pentru a asigura confidențialitatea datelor medicale, aplicația trebuie să fie accesibilă doar utilizatorilor autorizați, iar datele de intrare și rezultate trebuie stocate și accesate în condiții de securitate, conform standardelor specifice domeniului medical.

4.4. Interacțiunea dintre componente

Procesul începe cu achiziția datelor, care implică importarea datelor din fișiere externe (de exemplu, fișiere .csv). Această componentă asigură validitatea și calitatea surselor de date, pregătind astfel un set brut care este transmis mai departe către componentele de preprocesare.

Datele importate sunt preluate de componenta de pregătire a datelor. În această etapă, datele sunt curățate prin identificarea și completarea valorilor lipsă, precum și eliminarea datelor incoerente sau eronate. De asemenea, datele sunt normalizate și standardizate pentru a facilita analiza ulterioară. Rezultatul acestei etape este un set de date curat și omogen, gata pentru analiza distribuției.

După pregătirea datelor, componenta de analiză a distribuției preia setul curățat pentru a calcula statistici descriptive (de exemplu, medie, valoare minimă, valoare maximă). În plus, aici sunt identificate intervalele de interes și sunt adăugați indicatori relevanți pentru a evalua caracteristicile setului de date. Aceste informații sunt utilizate în etapa următoare pentru a asigura proporționalitatea claselor.

În etapa de prelucrare a datelor, setul de date este împărțit în două subseturi — unul pentru antrenarea modelului și unul pentru testare. Componentele de prelucrare asigură stratificarea datelor pentru menținerea proporționalității între clase, aspect esențial pentru performanța modelului. Seturile rezultate sunt trimise apoi către componenta de modelare.

Componenta de modelare utilizează un algoritm de clasificare Random Forest pentru a antrena modelul pe baza datelor de antrenare. După antrenare, modelul este aplicat pe setul de date de testare pentru a genera predicții. Performanța modelului este evaluată folosind indicatorii relevanți (precizie, acuratețe, recall), iar rezultatele sunt trimise către etapa de interpretare.

În ultima etapă se interpretează și documentează rezultatele obținute de model. Sunt create tabele de statistici finale, cu valorile esențiale (de exemplu, minime, maxime) pentru fiecare variabilă. De asemenea, rezultatele sunt analizate și sintetizate într-o secțiune detaliată, evidențiind relevanța acestora în contextul proiectului de analiză a datelor medicale.

5. Indicatori de performanță

- **Timpul de execuție** necesar pentru antrenarea și testarea modelului este un indicator important pentru eficiența modelului, mai ales dacă modelul urmează să fie aplicat în timp real, iar în MATLAB, timpul de execuție poate fi măsurat folosind tic și toc
- **Matricea de confuzie/Confusion Matrix** oferă o vedere detaliată asupra clasificărilor corecte și incorecte pentru fiecare clasă, fiind esențială pentru înțelegerea performanței modelului pe fiecare clasă.Pentru asta există funcția confusionmat în MATLAB.
- **AUC/Zona de sub curba ROC** măsoară capacitatea modelului de a diferenția între clase. Valori mai mari indică o capacitate mai bună de clasificare, iar in matlab AUC se poate calcula cu funcția perfcurve.
- **Recall-ul/Rata de Sensibilitate / Sensitivity** măsoară proporția de cazuri pozitive corect clasificate față de totalul cazurilor pozitive.
- **Precizia** măsoară proporția de instanțe prezise pozitiv care sunt corect clasificate, fiind utilă mai ales când ai clase dezechilibrate.

- **Acuratețea/accuracy** măsoară procentul de predicții corecte din totalul predicțiilor.

6. Elemente de testare

Performanța aplicației pentru analiza datelor medicale este influențată în principal de două componente critice:

- **Modulul de prelucrare a datelor**
  Performanța acestui modul este esențială, deoarece prelucrarea inițială (curățarea, normalizarea și divizarea setului de date) poate avea un impact major asupra eficienței și acurateței modelului de clasificare. Orice întârziere sau eroare în acest modul poate afecta calitatea rezultatelor obținute, ducând la clasificări inexacte sau la un proces de antrenare/testare lent.
- **Modulul de clasificare (Random Forest)**
  Performanța algoritmului Random Forest este de asemenea o componentă critică, deoarece acuratețea și timpul de execuție al aplicației depind de parametrii și configurația acestuia (cum ar fi numărul de arbori sau adâncimea maximă). Întregul proces de clasificare depinde de modul în care acest algoritm gestionează datele și de cât de rapid poate genera predicții precise. Ineficiența acestui modul poate duce la o performanță scăzută a aplicației și la rezultate nesatisfăcătoare.

Pentru îmbunătățirea performanței globale a aplicației, se pot lua în considerare următoarele alternative de proiectare a componentelor critice:

- **Alternative pentru modulul de prelucrare a datelor**
  - **Utilizarea funcțiilor MATLAB optimizate**: Se pot folosi funcții MATLAB specializate pentru manipularea eficientă a datelor (de exemplu, fillmissing pentru completarea valorilor lipsă sau normalize pentru normalizare), reducând astfel timpul de preprocesare.
  - **Paralelizarea procesării datelor**: Dacă setul de date este mare, MATLAB oferă posibilitatea de a paraleliza procesarea folosind *Parallel Computing Toolbox*, reducând astfel timpul necesar pentru pregătirea datelor.

- **Alternative pentru modulul de clasificare (Random Forest)**
    - **Utilizarea unui alt algoritm de clasificare**: În cazul în care Random Forest nu oferă performanța dorită, alternative precum *Support Vector Machine (SVM)* sau *Gradient Boosting* pot fi explorate, acestea oferind adesea o precizie și un timp de execuție comparabile.
    - **Optimizarea parametrilor Random Forest**: O altă opțiune este optimizarea parametrilor Random Forest prin tehnici de căutare a hiperparametrilor, cum ar fi *Grid Search* sau *Bayesian Optimization*, pentru a obține un model mai eficient și mai precis.

7. Anexe

https://www.geeksforgeeks.org/support-vector-machine-algorithm/

https://www.geeksforgeeks.org/confusion-matrix-machine-learning/#what-is-a-confusion-matrix

https://www.medicover.ro/despre-sanatate/hepatita-c-manifestari-diagnostic-si-tratament,509,n,295

https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

https://www.geeksforgeeks.org/how-to-do-train-test-split-using-sklearn-in-python/

https://www.kaggle.com/datasets/fedesoriano/hepatitis-c-dataset?resource=download