Documentație finală

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

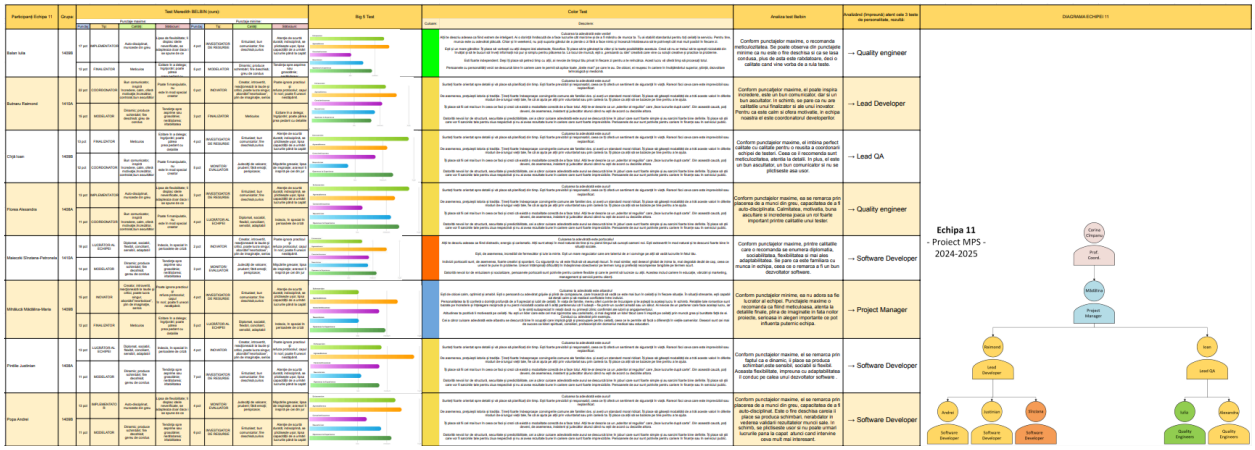
Realizat de ↓

Nume student:	Grupa:
Mihălucă Mădălina-Maria	1409B
Popa Andrei	1409B
Balan Iulia	1409B
Maieczki Petronela-Sînziana	1410A
Cîrja Ioan	1409B
Butnaru Raimond Eduard	1410A
Pintilie Justinian	1408A
Florea Alexandra	1408A

Cuprins:

1.Etapa_1:Organizarea echipei

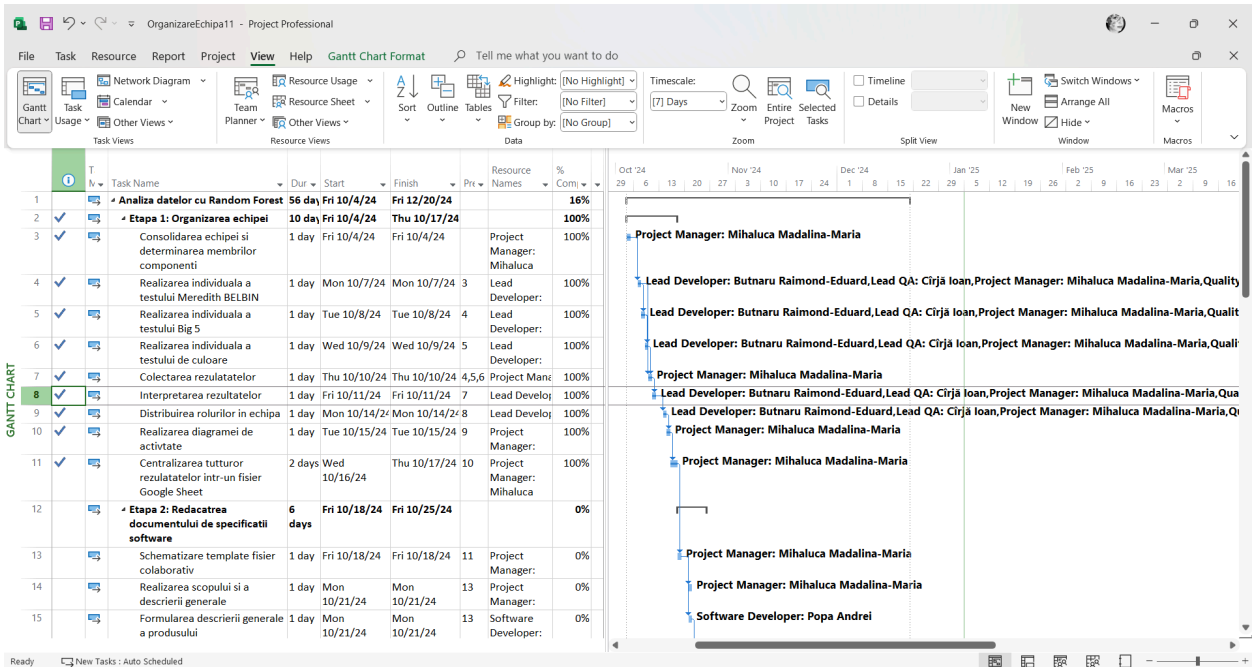
1.1. Interpretarea testelor. Stabilirea funcțiilor si atribuțiilor membrilor



[Click aici pentru a vedea în detaliu foaia de calcul]

1.2. Planificarea activităților și progres

Raport din Microsoft Project



	Task Name	Dur	Start	Finish	Pre	Resource Names	% Com
1	**Analiza datelor cu Random Forest**	**56 days**	**Fri 10/4/24**	**Fri 12/20/24**			**16%**
2	**Etapa 1: Organizarea echipei**	**10 days**	**Fri 10/4/24**	**Thu 10/17/24**			**100%**
3	Consolidarea echipei si determinarea membrilor componenti	1 day	Fri 10/4/24	Fri 10/4/24		Project Manager: Mihaluca	100%
4	Realizarea individuala a testului Meredith BELBIN	1 day	Mon 10/7/24	Mon 10/7/24	3	Lead Developer:	100%
5	Realizarea individuala a testului Big 5	1 day	Tue 10/8/24	Tue 10/8/24	4	Lead Developer:	100%
6	Realizarea individuala a testului de culoare	1 day	Wed 10/9/24	Wed 10/9/24	5	Lead Developer:	100%
7	Colectarea rezulatatelor	1 day	Thu 10/10/24	Thu 10/10/24	4,5,6	Project Mana	100%
8	Interpretarea rezultatelor	1 day	Fri 10/11/24	Fri 10/11/24	7	Lead Develo	100%
9	Distribuirea rolurilor in echipa	1 day	Mon 10/14/24	Mon 10/14/24	8	Lead Develo	100%
10	Realizarea diagramei de activtate	1 day	Tue 10/15/24	Tue 10/15/24	9	Project Manager:	100%
11	Centralizarea tuturor rezulatatelor intr-un fisier Google Sheet	2 days	Wed 10/16/24	Thu 10/17/24	10	Project Manager: Mihaluca	100%
12	**Etapa 2: Redacatrea documentului de specificatii software**	**6 days**	**Fri 10/18/24**	**Fri 10/25/24**			**0%**
13	Schematizare template fisier colaborativ	1 day	Fri 10/18/24	Fri 10/18/24	11	Project Manager:	0%
14	Realizarea scopului si a descrierii generale	1 day	Mon 10/21/24	Mon 10/21/24	13	Project Manager:	0%
15	Formularea descrierii generale a produsului	1 day	Mon 10/21/24	Mon 10/21/24	13	Software Developer:	0%

2.Etapa_2:Documentul de specificații software [SRS]

2.1. Scopul documentului

Acest document este destinat să descrie cu exactitate capabilitățile proiectului software pentru analiza și clasificarea datelor referitoare la Hepatita C, utilizând algoritmi de învățare automată.

Clarifică specificațiile proiectului, obiectivele și constrângerile, ajutând la înțelegerea modului în care produsul va îndeplini cerințele funcționale și nefuncționale, facilitând astfel o implementare precisă și o evaluare corectă a performanței în procesarea datelor (a biomarker-ilor) și clasificarea rezultatelor.

2.2. Descriere generală

Această aplicație software, realizată în Matlab, este destinată clasificării pacienților cu hepatită C utilizând un set de date disponibil pe Kaggle, care conține biomarkeri și categorii de diagnostic.

Setul de date conține valori de laborator ale donatorilor de sânge și ale pacienților cu Hepatita C, precum și valori demografice, cum ar fi vârsta, etc. Proiectul implică încărcarea și curățarea setului de date, analizarea și preprocesarea variabilelor, inclusiv verificarea și corectarea valorilor lipsă, și calcularea unor statistici descriptive pentru detectarea eventualelor anomalii sau discrepanțe în ceea ce privește datele. Se vor construi diverse partitii ale seturilor de date de antrenament și testare pentru evaluarea performanței modelelor. Algoritmul Random Forest va fi utilizat pentru clasificare, urmând să se realizeze selecția trăsăturilor relevante și evaluarea avantajelor și dezavantajelor acestei abordări, luând în considerare variații în proporțiile de împărțire a datelor pentru antrenare și testare (80%-20%, 70%-30%, 60%-40%, 50%-50%). Analiza noastră experimentelă vizează diferențierea între donatorii de sânge și pacienții cu Hepatita, evidențiind/nu evoluția bolii la Hepatita C, Fibroză și Ciroză.

2.3. Descrierea generală a produsului

2.3.1. Situația curentă

Setul de date conține valori de laborator ale donatorilor de sânge și ale pacienților cu Hepatita, precum și valori demografice, cum ar fi vârsta și sexul. Datele au fost obținute din *UCI Machine Learning Repository.* În prezent, datele brute conțin valori lipsă și potențiale anomalii

care pot afecta rezultatele analizei. Fără o preprocesare adecvată și un model de clasificare eficient, aceste date nu pot fi utilizate corespunzător pentru a sprijini clasificarea, diagnosticarea sau prevenirea bolii hepatice.

2.3.2. Scopul produsului

Algoritmul software are ca scop analiza experimentală și clasificarea datelor medicale (provenite din mediul unui laborator de analize), utilizând tehnici de învățare automată pentru a sprijini diagnosticarea și prevenirea bolii. Acesta va permite preprocesarea datelor brute, identificarea trăsăturilor relevante și aplicarea unui model de clasificare, cum ar fi Random Forest, pentru a oferi predicții precise despre starea pacienților. Prin intermediul acestei soluții, datele vor fi curățate, structurate și analizate într-un mod eficient, oferind informații utile atât cercetătorilor, cât și practicienilor medicali pentru o mai bună înțelegere și gestionare a bolilor hepatice.

2.3.3. Contextul produsului și motivarea implementării

Bolile hepatice, inclusiv hepatita, reprezintă o problemă de sănătate publică majoră la nivel mondial. Diagnosticarea corectă în timp util a acestor afecțiuni este crucială pentru tratarea eficientă a pacienților și pentru prevenirea complicațiilor grave care pot apărea.

În contextul curent de avansare tehnologică și medicală, potențialul de îmbunătățire a metodelor clasice de diagnosticare a bolilor hepatice este în creștere. Aplicațiile ce sunt bazate pe modele IA(Invatare Automata)/ML(Machine Learning) pot analiza mult mai rapid și chiar mai precis volumele mari de date, identificând tipare care pot scăpa ochiului uman, oferind astfel un sprijin în luarea deciziilor medicale, bazat pe date concrete. Setul de date selectat pentru această aplicație include diferiți biomarkeri care sunt utilizați pentru a determina dacă un pacient poate fi donator sau dacă pacientul respectiv prezintă semne de afecțiuni hepatice.

2.3.4. Beneficii

Printre beneficii se enumeră:

- **Diagnostice îmbunătățite** - Utilizarea tehnicilor de ML cum ar fi, spre exemplu, Random Forest, aplicația poate clasifica datele medicale pentru a identifica pacienții predispuși la boli hepatice, aducând astfel o diagnosticare mai rapidă, mai precisă și mult mai eficientă.

- **Costuri și timp de analiză reduse** - Analizele tradiționale sunt costisitoare și consumatoare de timp. Utilizând algoritmi automați pentru prelucrarea datelor, aplicația poate reduce semnificativ timpul necesar pentru a obține un diagnostic, reducând astfel costurile și facilitând accesul la diagnosticare pentru mai mulți pacienți.

- **Prevenirea și intervenția timpurie** - Prin identificarea pacienților cu risc încă din fazele incipiente ale bolii, se poate interveni mai devreme, prevenind agravarea afecțiunii și reducând necesitatea unor tratamente mai agresive și costisitoare în stadiile avansate.

- **Validarea și analiza datelor** - Aplicația va efectua o serie de pași pentru pre-procesarea datelor, cum ar fi eliminarea valorilor lipsă și corectarea datelor incorecte. În plus, analiza statistică (medii, dispersii, valori minime și maxime) și indicatorii specifici vor oferi o imagine de ansamblu asupra caracteristicilor setului de date, permițând identificarea tiparelor importante și a corelațiilor relevante pentru diagnostic

2.4. Specificații funcționale

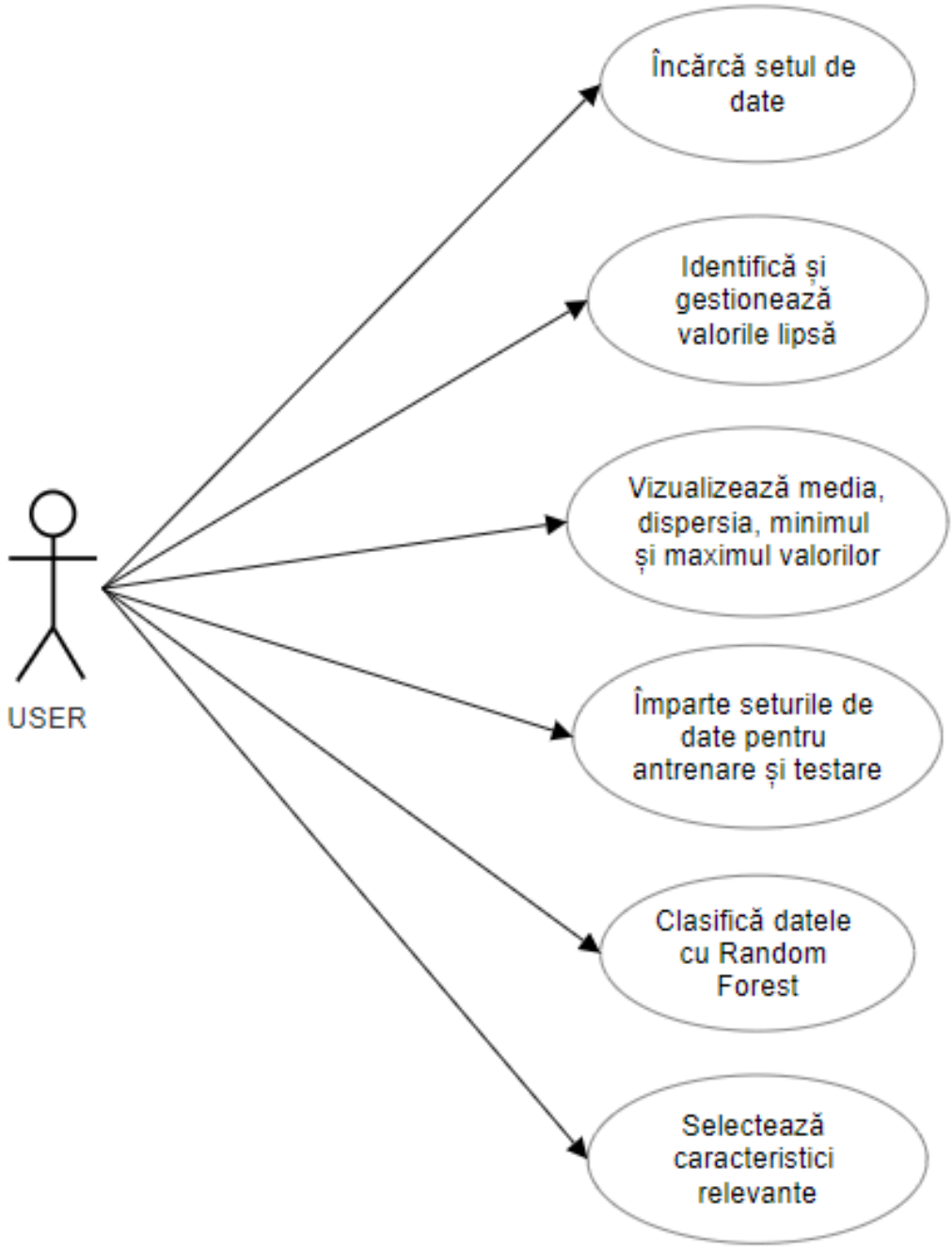
2.4.1. Actori

În cazul acestui proiect singurul actor este utilizatorul(User-ul). Mai jos sunt descrise toate acțiunile pe care le poate executa.

Acțiuni User:

- Încarcă setul de date
- Identifică și gestionează valorile lipsă
- Vizualizează media, dispersia, minimul și maximul valorilor
- Împarte seturile de date pentru antrenare și testare
- Clasifică datele cu Random Forest
- Selectează caracteristici relevante

2.4.2. Diagrama cazurilor de utilizare



2.4.3. Descrierea cazurilor de utilizare

2.4.3.1. Încărcarea setului de date;

USE CASE:	**Încărcare set de date**
Descriere:	Setul de date este încărcat pentru analiza și procesarea ulterioară.
Prioritate:	Esențial
Declanșator:	Cercetătorul sau utilizatorul selectează fișierul setului de date.
Precondiție:	Utilizatorul are acces la setul de date.
Calea de bază:	Utilizatorul încarcă fișierul CSV cu datele despre medicale.
Calea alternativă:	Dacă fișierul este corupt sau formatul nu este recunoscut, se returnează un mesaj de eroare.
Postcondiții:	Datele sunt încărcate și pregătite pentru preprocesare.
Calea pentru excepții:	Dacă fișierul nu este disponibil sau nu poate fi încărcat, sistemul returnează un mesaj de eroare și revine la starea inițială.

2.4.3.2. Identificarea și gestionarea valorilor lipsă;

USE CASE:	**Identificare și gestionarea a valorilor lipsă**
Descriere:	Se identifică și se gestionează valorile lipsă sau nedefinite din setul de date despre hepatită.
Prioritate:	Esențial
Declanșator:	Utilizatorul începe analiza setului de date și verifică dacă există date lipsă.
Precondiție:	Datele sunt încărcate și accesibile pentru procesare.
Calea de bază:	Valorile lipsă sunt gestionate fie prin eliminarea eșantioanelor incomplete, fie prin completarea lor folosind metode statistice.
Calea alternativă:	Dacă nu există valori lipsă, se trece direct la analiza statistică.
Postcondiții:	Setul de date este complet și pregătit pentru analiza statistică și preprocesare.
Calea pentru excepții:	Dacă prea multe date lipsesc, se sugerează utilizatorului eliminarea unor coloane sau reîncărcarea datelor.

2.4.3.3. Vizualizarea mediei, dispersiei, a minimului și a maximului;

USE CASE:	**Analiza statistică a variabilelor de intrare**
Descriere:	Calcularea unor statistici de bază (media, dispersia, valorile minime și maxime) pentru variabilele de intrare.
Prioritate:	Esențial
Declanșator:	Utilizatorul dorește să verifice distribuția valorilor biomarkerilor și să identifice eventuale eșantioane izolate sau discrepanțe.
Precondiție:	Datele sunt complete și pregătite pentru analiză.
Calea de bază:	1. Se identifică și se raportează eventualele valori anormale sau discrepanțe în setul de date. 2. Se realizează o vizualizare simplă, cum ar fi histogramele, pentru a observa distribuția valorilor.
Calea alternativă:	Dacă datele sunt deja normalizate sau corectate, se poate trece direct la pasul următor de preprocesare.
Postcondiții:	Caracteristicile principale ale setului de date sunt identificate, și eventualele eșantioane izolate sunt raportate.
Calea pentru excepții:	Dacă setul de date conține erori majore sau valori neașteptate, utilizatorul este informat să revizuiască preprocesarea.

2.4.3.4. Împărțirea seturilor de date pentru antrenare și testare;

USE CASE:	**Împărțire seturi de date pentru antrenare și testare**
Descriere:	Setul de date despre hepatită este împărțit în seturi de antrenare și testare pentru modelarea ulterioară.
Prioritate:	Esențial
Declanșator:	Utilizatorul dorește să pregătească datele pentru antrenarea și testarea unui model de clasificare.
Precondiție:	Datele sunt curățate și analizate.
Calea de bază:	1. Utilizatorul împarte setul de date folosind funcția în proporții de 80%-20%, 70%-30%, 60%-40% și 50%-50%. 2. Se verifică distribuția valorilor țintă în ambele seturi pentru a se asigura că nu există dezechilibre majore.
Calea alternativă:	În cazul dezechilibrelor de distribuție, se poate utiliza stratificarea pentru a menține distribuția uniformă a clasei.
Postcondiții:	Seturile de date pentru antrenare și testare sunt pregătite și echilibrate pentru următoarea etapă de clasificare.
Calea pentru excepții:	Dacă distribuția este dezechilibrată, utilizatorul este avertizat și se recomandă reîmpărțirea setului de date.

2.4.3.5. Clasifică datele cu Random Forest;

USE CASE:	**Clasificare de date cu Random Forest**
Descriere:	Datele despre hepatită sunt clasificate folosind algoritmul Random Forest.
Prioritate:	Esențial
Declanșator:	Utilizatorul dorește să clasifice pacienții în funcție de biomarkerii lor și să determine dacă sunt donatori sau au boli hepatice.
Precondiție:	Seturile de date de antrenare și testare sunt pregătite.
Calea de bază:	1. Modelul este antrenat pe setul de date de antrenare. 2. Performanța modelului este evaluată pe setul de testare folosind metrici precum acuratețea, precizia și matricea de confuzie.
Calea alternativă:	Dacă performanța este slabă, utilizatorul poate ajusta parametrii modelului.
Postcondiții:	Modelul Random Forest este antrenat și evaluat, iar rezultatele sunt raportate.
Calea pentru excepții:	Dacă modelul nu converge sau are o performanță slabă, se recomandă reanalizarea setului de date.

2.4.3.6. Selectează caracteristici relevante;

USE CASE:	**Selecția caracteristicilor relevante**
Descriere:	Se identifică trăsăturile biomarkerilor relevanți pentru clasificarea eficientă a pacienților.
Prioritate:	Esențial
Declanșator:	Utilizatorul dorește să optimizeze modelul prin selecția caracteristicilor relevante.
Precondiție:	Modelul de clasificare a fost construit și evaluat.
Calea de bază:	Caracteristicile irelevante sau redundante sunt eliminate din setul de date.
Calea alternativă:	Dacă selecția de trăsături nu îmbunătățește modelul, se recomandă utilizarea unor tehnici de reducere a dimensionalității.
Postcondiții:	Setul de date conține doar trăsăturile relevante pentru o clasificare eficientă.
Calea pentru excepții:	Dacă nu există diferențe semnificative în performanță, modelul poate fi recalibrat cu toate caracteristicile.

2.5. Specificații non-funcționale

2.5.1. Specificațiile interfeței cu utilizatorul

- Utilizatorul trebuie să fie familiarizat cu Matlab, fiind capabil să pornească un script și să înțeleagă rezultatele generate.
- Interacțiunea cu programul trebuie să fie minimă, cu instrucțiuni clare și ușor de urmărit.

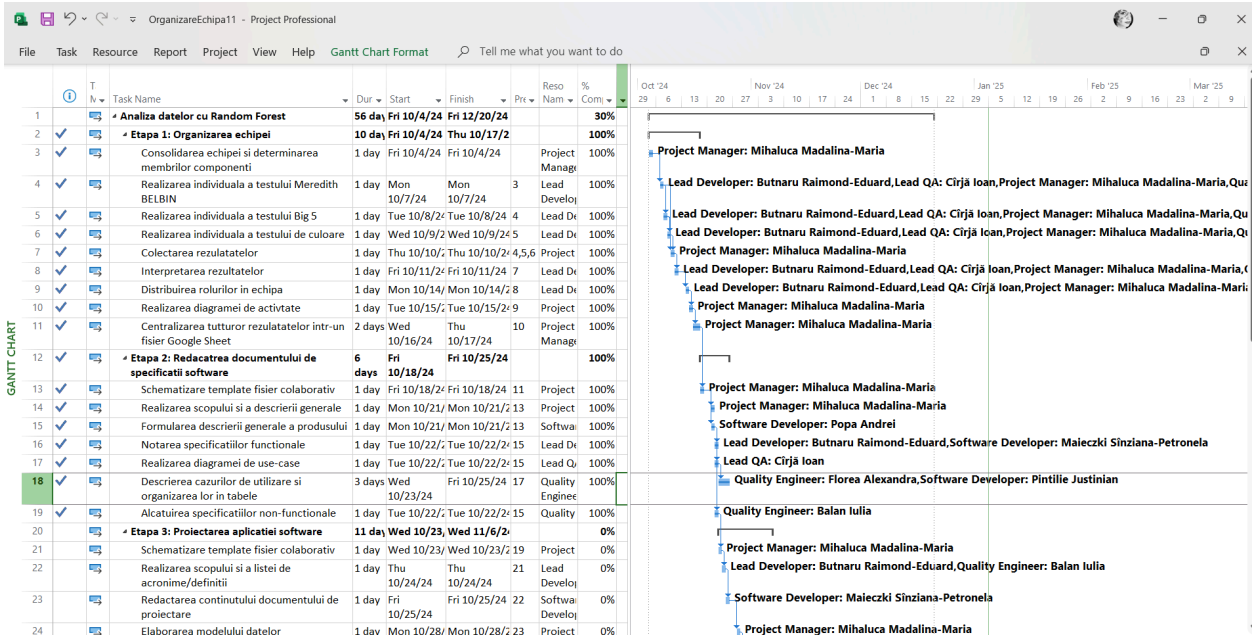
2.5.2. Specificațiile de performanță

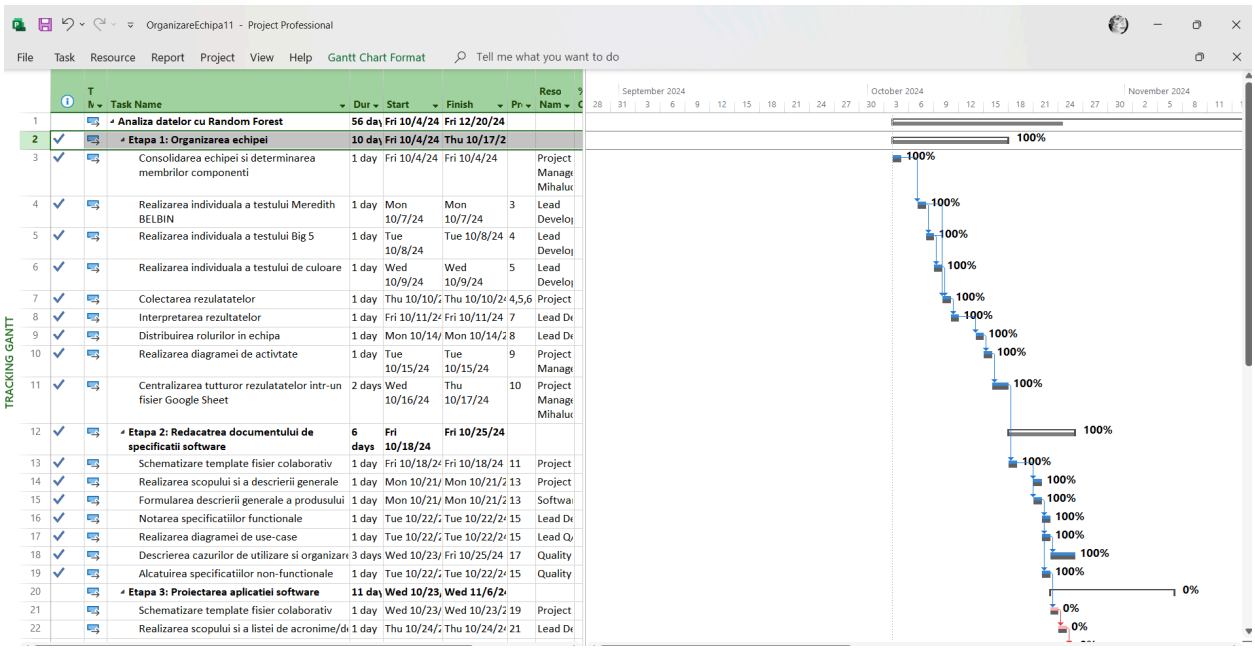
- Programul trebuie să ruleze eficient pe orice sistem care suportă Matlab, fără a necesita resurse hardware suplimentare.
- Timpul de execuție trebuie să fie optim pentru a gestiona seturi de date mari, iar calculele complexe să fie finalizate într-un interval rezonabil.

2.5.3.Disponibilitatea și fiabilitatea

- Programul trebuie să producă întotdeauna rezultatul corect și să fie capabil să semnaleze erorile umane, cum ar fi date lipsă sau formate incorecte.
- În cazul apariției unor probleme, programul trebuie să returneze mesaje de eroare clare pentru a ajuta utilizatorul să corecteze situația.

2.6. Planificarea activităților și progres





3.Etapa_3:Documentul de proiectare a soluţiei aplicaţiei software[SDD]

3.1.Scopul documentului

3.1.1. 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.

3.1.2. 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.

3.1.3. 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.

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

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.

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

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

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.

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

3.2. 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

3.3. Modelul datelor

3.3.1. 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.

3.3.2. 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.

3.3.3. 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.

3.3.4. 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.

3.3.5. 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.

3.4. Modelul arhitectural /Modelul componentelor

3.4.1. Arhitectura sistemului

Fig.1. Ilustrarea diagramei secvențiale a componentelor

3.4.2. Descrierea componentelor

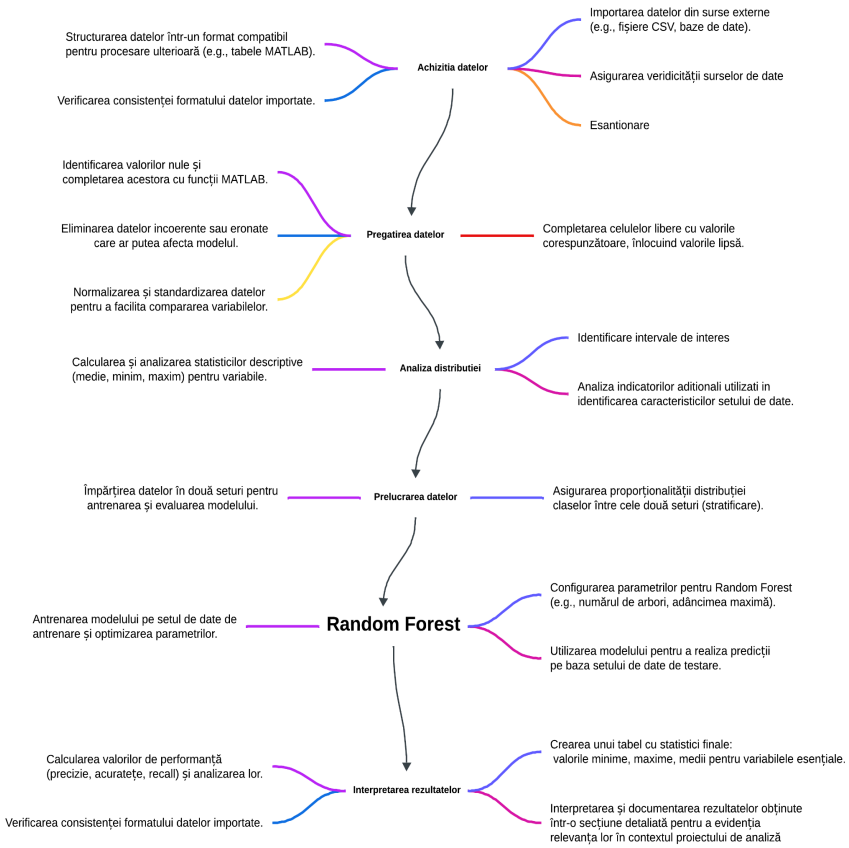


Diagrama secvențială din Fig.1. este alcătuită din următoarele componente:

3.4.2.1. 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.

3.4.2.2. 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.

3.4.2.3. 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.

3.4.2.4. 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.

3.4.2.5. 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 că 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.

3.4.2.6. 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.

3.4.3. 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.

3.4.4. 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.

3.5. 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.

3.6. 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:

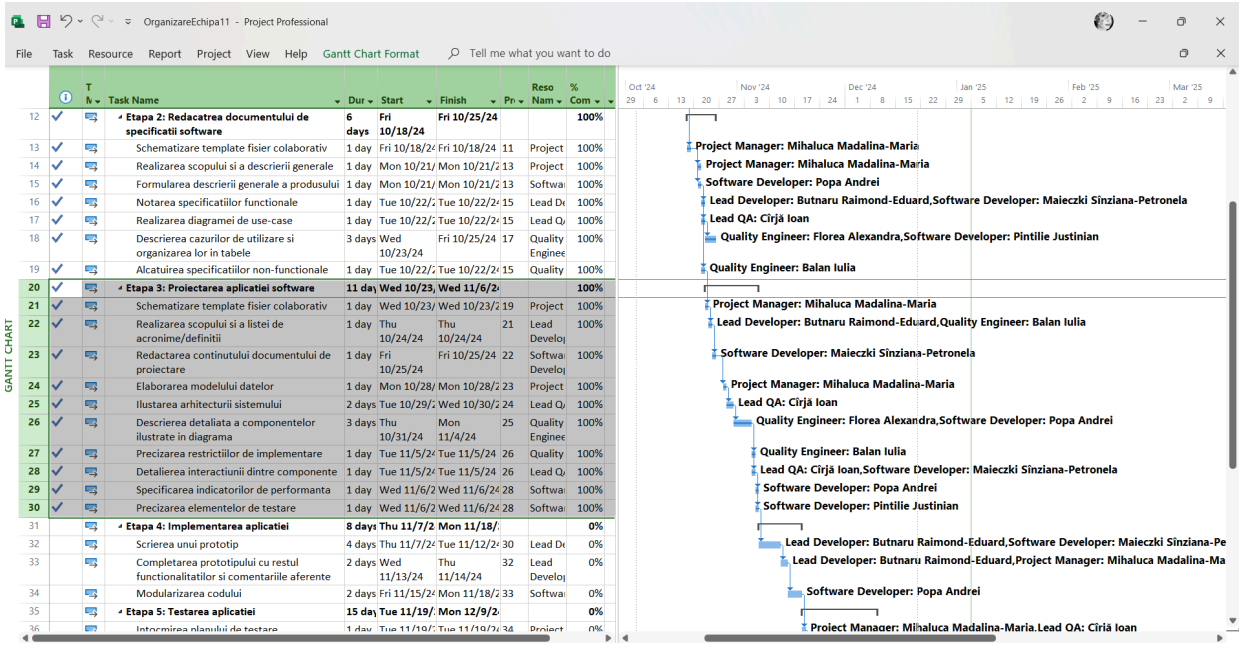
- **Alternative pentru modulul de prelucrare a datelor**
  - **Utilizarea funcțiilor MATLAB optimizate**: Se pot folosi funcții MATLAB specializate pentru manipularea eficientă a datelor (de exemplu, fillmissing pentru

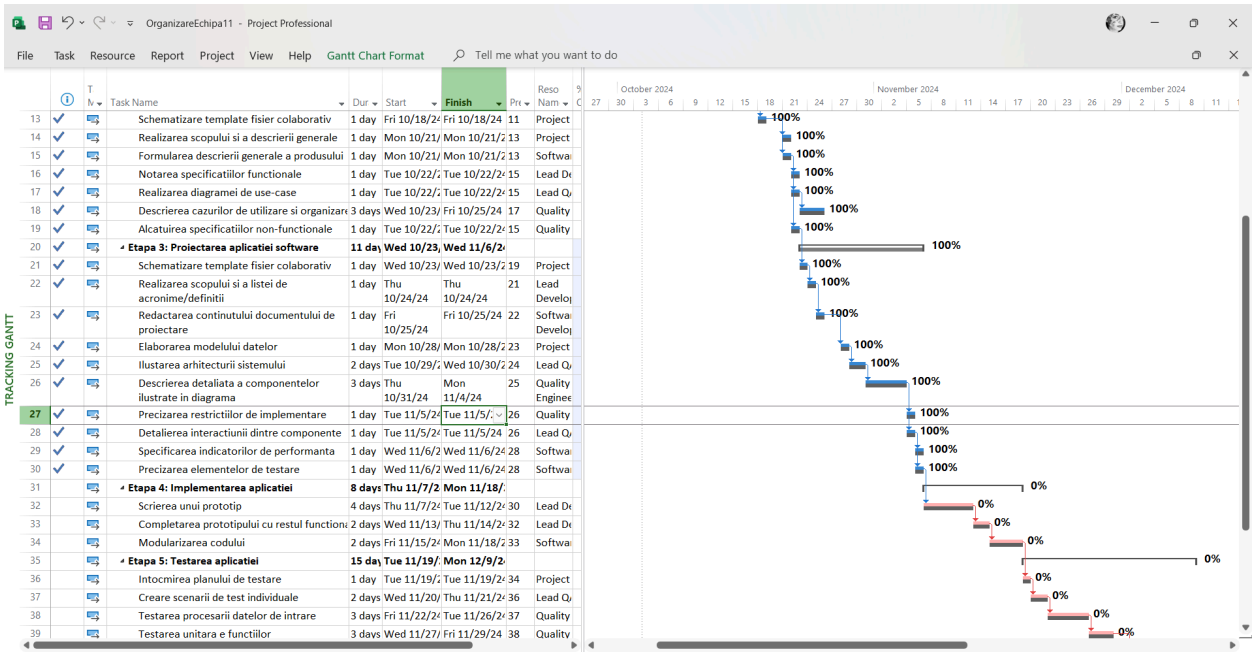
completarea valorilor lipsă sau normalize pentru normalizare), reducând astfel timpul de preprocesare.

- **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.

3.7. Planificarea activităților și progres





4.Etapa_4:Implementarea aplicației

4.1.Componentele aplicației

4.1.1.Componenta de achiziție de date

Implementarea pentru funcția atribuită acestei componente este:

```
% se citesc date din csv
csvFileNames = {
        'HepatitisC.csv',
         'HepatitisCdata-50sanatosi.csv',
        'HepatitisCdata-doarBolnavi.csv',
        'HepatitisCdata-putiniSanatosiMultiBolnavi.csv'};
data = readtable(csvFileNames{4});
results_folder = {
        'results-HepatitisC',
        'results-HepatitisCdata-50sanatosi',
        'results-HepatitisCdata-doarBolnavi',
        'results-HepatitisCdata-putiniSanatosiMultiBolnavi'};
% se valideaza datele din tabel
```

```
checked_data = validare_date(data);

% se genereaza graficul validarii datelor
bar_chart_validare(checked_data);

% s-a decis eliminarea datelor nevalide
cleared_data = clear_data(readtable('HepatitisC-Checked.csv'));

% se verifica buna functionalitate a curatarii datelor
bar_chart_validare(cleared_data);
```

4.1.2.Componenta de pregătire a datelor

Implementarea pentru funcția atribuită acestei componente este:

```
function dateValidate = validare_date(data)
    % Verifică dacă inputul este un tabel
    if ~istable(data)
        error('Inputul trebuie să fie un tabel.');
    end

    % Conversie la celule pentru prelucrare
    data_out = table2cell(data);

    % Iterare prin fiecare element și înlocuire a valorilor goale cu NaN
    for i = 1:size(data_out, 1)
        for j = 1:size(data_out, 2)
            if isempty(data_out{i, j}) || (ischar(data_out{i, j}) &&
isempty(strtrim(data_out{i, j})))
                data_out{i, j} = NaN;
            end
        end
    end

    % Conversie înapoi la tabel cu aceleași nume de coloane
    dateValidate = cell2table(data_out, 'VariableNames',
data.Properties.VariableNames);

    % Salvare în fișier CSV
    outputFileName = 'HepatitisC-Checked.csv';
```

```
    writetable(dateValidate, outputFileName);
    disp(['Fișierul validat a fost salvat ca: ', outputFileName]);
end
```

4.1.3.Componenta de analiză a distribuției

Implementarea câtorva funcții atribuite acestei componente sunt:

```
% Functie care primeste ca parametru un set de date si care returneaza
% un vector ce contine maximul pentru fiecare coloana in parte.
% In cazul in care functia nu are deloc valori numerice, va afisa NaN.
function valoriMaxim = maxim(data)
    maximColoane = NaN(1, width(data));

    for col = 1:width(data)
        if isnumeric(data{:, col}) || islogical(data{:, col})
            maximColoane(col) = max(data{:, col}, [], 'omitnan');
        else
            maximColoane(col) = NaN;
        end
    end
    valoriMaxim = maximColoane;
End
```

—----------

```
% Functie care primeste ca parametru un set de date si care returneaza
% un vector ce contine mediana pentru fiecare coloana in parte.
% In cazul in care functia nu are deloc valori numerice, va afisa NaN.
function valoriMediana = mediana(data)
    medianaColoana = NaN(1, width(data));

    for col = 1:width(data)
        if isnumeric(data{:, col}) || islogical(data{:, col})
            medianaColoana(col) = median(data{:, col}, 'omitnan');
        else
            medianaColoana(col) = NaN;
        end
    end
    valoriMediana = medianaColoana;
End
```

—----------

```
% Functie care primeste ca parametru un set de date si care returneaza
% un vector ce contine normalizarea pentru fiecare coloana in parte.
% In cazul in care functia nu are deloc valori numerice, va afisa NaN.
% normalizare = (x-min(x)) / (max(x)-min(x))
function valoriNormalizare = normalizare(data)
    valoriNormalizare = NaN(size(data));

    for col = 1:width(data)
        if isnumeric(data{:, col}) || islogical(data{:, col})
            colData = data{:, col};
            colMin = min(colData, [], 'omitnan');
            colMax = max(colData, [], 'omitnan');

            if colMax > colMin
                valoriNormalizare(:, col) = (colData - colMin) / (colMax - colMin);
            else
                valoriNormalizare(:, col) = 0;
            end
        else
            valoriNormalizare(:, col) = NaN;
        end
    end
end
```

```
% Functie care primeste ca parametru un set de date si descrierea
% acestora cu care calculeaza si afiseaza o serie de caracteristici statice
  % data: tabelul de intrare
  % descriere: un string care descrie setul de date
function afisare_statistici(data, descriere)
    % se calculeaza statisticile
    valMedie = media(data);
    valDispersie = dispersia(data);
    valMinim = minim(data);
    valMaxim = maxim(data);
    valDevStd = deviatia_standard(data);
    valModul = modul(data);
    valMediana = mediana(data);
    valNormalizare = normalizare(data);
    valStandardizare = standardizare(data);
```

```
        % se afiseaza statisticile
        disp(['Statistici pentru ', descriere, ':']);
        disp('Media:');
        disp(valMedie);
        disp('Dispersia:');
        disp(valDispersie);
        disp('Valorile minime:');
        disp(valMinim);
        disp('Valorile maxime:');
        disp(valMaxim);
        disp('Valorile deviatiei standard:');
        disp(valDevStd);
        disp('Valorile modulului:');
        disp(valModul);
        disp('Valorile medianei:');
        disp(valMediana);
        %disp('Valorile normalizate:');
        %disp(valNormalizare);
        %disp('Valorile standardizate:');
        %disp(valStandardizare);

        disp('-----------------------------------------------------');
    end
```

4.1.4. Componenta de prelucrare a datelor

O parte din implementarea funcției atribuite acestei componente este:

```
% Functie care primeste ca parametri setul de date, categoriile acestuia si
% procentul de date alocat pentru antrenament si returneaza doua seturi de
% date distincte, unul pentru antrenare si unul pentru testare
   % data: tabelul cu date
   % categories: vector numeric cu etichetele de clasificare
   % percentTrain: procentajul de date alocat pentru antrenament
function [trainData, testData] = split_data(data, categories, percentTrain)
   % se verifica daca lung categcorespunde cu numarul de randuri din date
   if numel(categories) ~= height(data)
                        error('Lungimea categoriilor nu corespunde cu numarul de randuri
       din tabelul de date.');
   end

   % se gasesc clasele unice
```

```
    uniqueClasses = unique(categories);
    trainIdx = [];
    testIdx = [];

    % se impart datele pe clase
    for i = 1:numel(uniqueClasses)
        classIdx = find(categories == uniqueClasses(i));
        numTrain = round(percentTrain / 100 * numel(classIdx));
        shuffleIdx = randperm(numel(classIdx));
        trainIdx = [trainIdx; classIdx(shuffleIdx(1:numTrain))];
        testIdx = [testIdx; classIdx(shuffleIdx(numTrain+1:end))];
    end

    % se selecteaza randurile pentru seturile de antrenament si testare
    trainData = data(trainIdx, :);
    testData = data(testIdx, :);
end
```

4.1.5. Componenta de antrenare cu Random Forest

Implementarea pentru funcția atribuită acestei componente este:

```
    function [accuracyTrain, accuracyTest, feature_importance, YPredTest, YTest, model,
trainTime, predictTime] = random_forest(data, options, trainPercent)

    numericData = data(:, varfun(@isnumeric, data, 'OutputFormat', 'uniform'));
    category = categorizare(data);
    category = double(categorical(category));
    numericData.Category = category;

     [trainData, testData] = split_data(numericData, category, trainPercent);

    % se extrag datele de antrenament si de testare
    XTrain = trainData(:, 1:end-1);
    YTrain = trainData.Category;

    XTest = testData(:, 1:end-1);
    YTest = testData.Category;

    % se creaza si antreneaza modelul Random Forest
```

```
    numTrees = options.numTrees;
    maxSplits = options.maxSplits;
    tic;
    model = TreeBagger(numTrees, XTrain, YTrain, 'Method', 'classification', ...
                    'MaxNumSplits', maxSplits, 'OOBPrediction', 'on', ...
                    'OOBPredictorImportance', 'on');
    trainTime = toc;

    tic;
    % predictiile pe setul de testare
    YPredTest = str2double(predict(model, XTest));
    predictTime = toc;

    YPredTrain = str2double(predict(model, XTrain));

    accuracyTrain = mean(YPredTrain == YTrain);
    accuracyTest = mean(YPredTest == YTest);

      feature_importance = model.OOBPermutedPredictorDeltaError;
end
```

4.1.6. Componenta de interpretare a rezultatelor

Implementarea atribuită acestei componente conține:

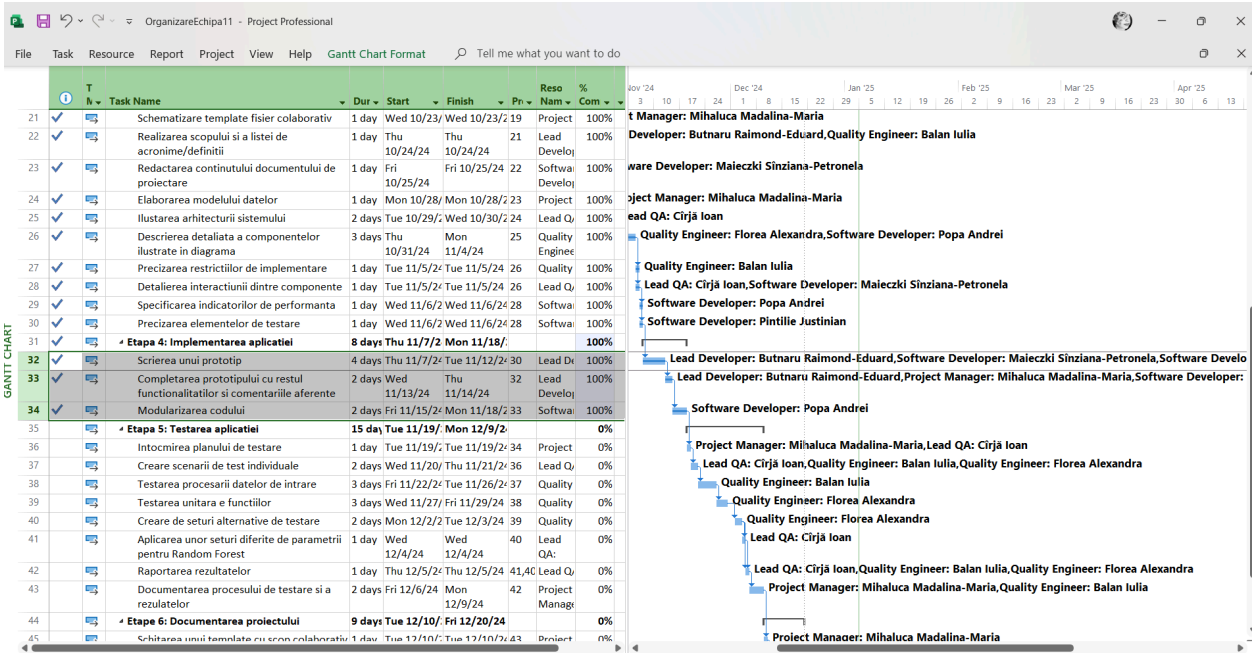
```
tableData = {};
for csvIdx = 1:length(csvFileNames)
    data = readtable(csvFileNames{csvIdx});
    checked_data = validare_date(data);
    bar_chart_validare(checked_data);
    cleared_data = clear_data(readtable('HepatitisC-Checked.csv'));
    bar_chart_validare(cleared_data);
    afisare_statistici(cleared_data, 'Setul complet de date');
    category = categorizare(cleared_data);
    category = double(categorical(category));

    for trainIdx = 1:length(train_percent)
        currentTrainPercent = train_percent(trainIdx);
```

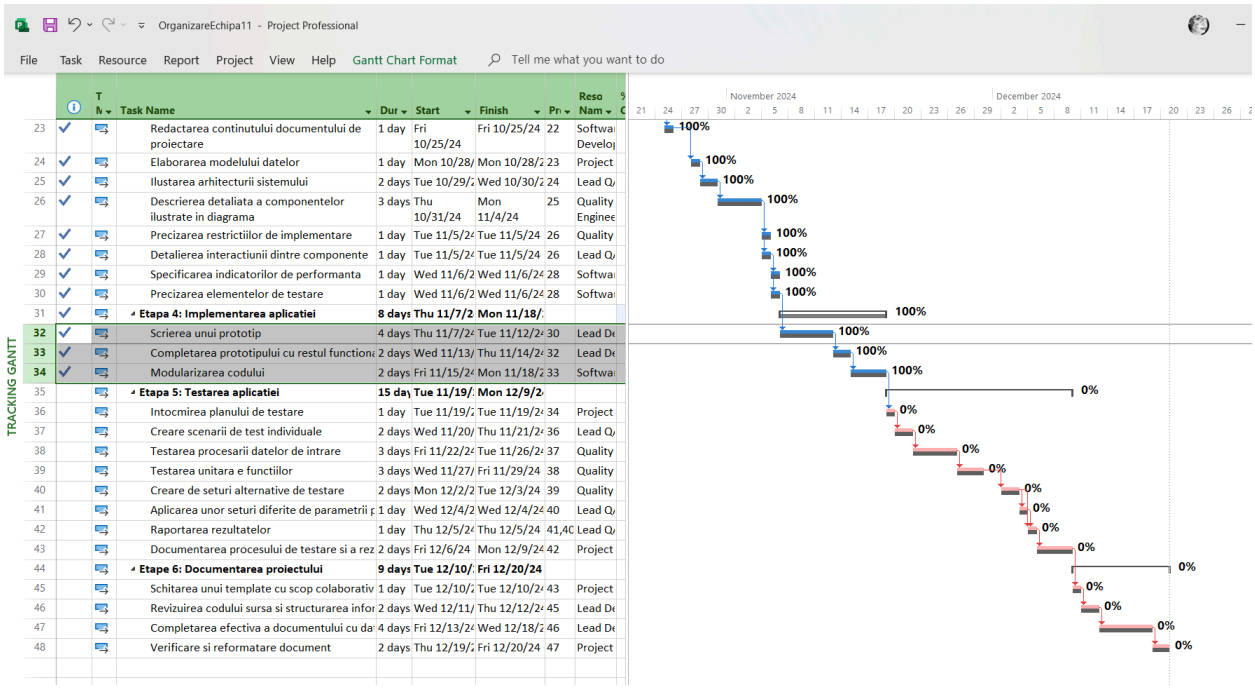
```
            [trainData, testData] = split_data(cleared_data, category,          currentTrainPercent);
            generate_piechart(trainData, testData);

            afisare_statistici(trainData, 'Setul de date de antrenare');

            for i = 1:length(numTrees_list)
                for j = 1:length(maxSplits_list)
                    for k = 1:length(minLeafSize_list)

                        options.numTrees = numTrees_list(i);
                        options.maxSplits = maxSplits_list(j);
                        options.minLeafSize = minLeafSize_list(k);

                        disp(['Antrenam model cu ', ...
                            'numTrees = ', num2str(options.numTrees), ...
                            ', maxSplits = ', num2str(options.maxSplits), ...
                            ', minLeafSize = ', num2str(options.minLeafSize)]);

                        [accuracyTrain, accuracyTest, feature_importance, YPred, YTest, model,
trainTime, predictTime] = ...
                            random_forest(cleared_data, options, currentTrainPercent);
                        overfitting = accuracyTrain - accuracyTest;
                        underfitting = min(accuracyTrain, accuracyTest);

                        num_categ = numel(unique(YTest));
                        confMatrix = matrice_confuzie(YTest, YPred, num_categ);
                        [precision, recall, f1score] = calcul_performanta(confMatrix);

                        fig = figure('Name', ['Random Forest - numTrees=',
num2str(options.numTrees), ...
                                ', maxSplits=', num2str(options.maxSplits), ...
                                ', minLeafSize=', num2str(options.minLeafSize)], ...
                            'Position', [100, 100, 1400, 800]);

                        t = tiledlayout(3, 3, 'TileSpacing', 'Compact', 'Padding', 'Compact');

                        nexttile(t, [1, 1]);
                        bar(feature_importance);
                        title('Feature Importance');
                        xlabel('Feature');
```

```
                ylabel('Importance');

                nexttile(t, [1, 1]);
                histogram(YTest, 'Normalization', 'probability');
                title('Class Distribution in YTest');
                xlabel('Classes');
                ylabel('Frequency');

                nexttile(t, [1, 1]);
                histogram(YPred, 'Normalization', 'probability');
                title('Class Distribution in YPred');
                xlabel('Classes');
                ylabel('Frequency');

                nexttile(t, [1, 1]);
                heatmap(confMatrix, 'Title', 'Confusion Matrix', ...
                    'XLabel', 'Predicted', 'YLabel', 'Actual', ...
                    'ColorbarVisible', 'on');

                nexttile(t, [1, 1]);
                text(0.5, 0.5, sprintf('Model Accuracy Train: %.2f%% \n Model Accuracy
Test: %.2f%% \n\n Train Time: %fs \n Predict Time: %fs \n Overfitting: %f \n Overfitting: %f',
accuracyTrain * 100, accuracyTest * 100, trainTime, predictTime, overfitting, underfitting), ...
                    'FontSize', 14, 'FontWeight', 'bold', ...
                    'HorizontalAlignment', 'center', 'VerticalAlignment', 'middle');
                axis off;

                colNames = {'Class', 'Precision', 'Recall', 'F1-Score'};
                rowData = [(1:num_categ)', precision, recall, f1score];
                uitable('Data', rowData, 'ColumnName', colNames, ...
                    'RowName', [], 'Units', 'Normalized', ...
                    'FontSize', 10, 'ColumnWidth', {50, 100, 100, 100}, ...
                    'Position', [0.72, 0.42, 0.26, 0.19]);


                filename =
sprintf('RandomForest_%s_TrainPct_%d_numTrees_%d_maxSplits_%d_minLeafSize_%d.png',
...
                        erase(csvFileNames{csvIdx}, '.csv'), currentTrainPercent, ...
                        options.numTrees, options.maxSplits, options.minLeafSize);
```

```
                    saveas(fig, fullfile(results_folder{csvIdx}, filename));
                    disp('Press Enter to continue to the next iteration.');
                    pause;
                    close(fig);

                    predictorNames = model.PredictorNames;
                    row = {num2str(numTrees_list(i)), num2str(maxSplits_list(j)),
num2str(minLeafSize_list(k)), num2str(accuracyTrain), num2str(accuracyTest),
num2str(trainTime), num2str(predictTime), num2str(overfitting), num2str(underfitting)};
                    for idx = 1:length(predictorNames)
                        row{end+1} = num2str(feature_importance(idx));
                    end
                    tableData = [tableData; row];


                end
            end
        end
    end
end

predictorNames = model.PredictorNames;
colNames = {'NumTrees', 'MaxSplits', 'MinLeafSize', 'Accuracy Train', 'Accuracy Test',
'Train Time', 'Predict Time', 'Overfitting', 'Underfitting' predictorNames{:}};
uitableData = cell2table(tableData, 'VariableNames', colNames);
fig = figure('Position', [100, 100, 1600, 1000]);
uitable('Data', uitableData{:,:}, 'ColumnName', uitableData.Properties.VariableNames, ...
    'Position', [20, 20, 1400, 800]);
```

4.2. Planificarea activităților și progres





5.Etapa_5:Testarea aplicației

5.1. Plan de testare

Planul de testare are ca scop definirea procedurilor și metodologiilor utilizate pentru a valida și verifica funcționalitatea, performanța și robustețea aplicației. Acesta este structurat astfel încât să asigure acoperirea completă a funcționalităților implementate și să identifice eventualele erori sau neconformități înainte de livrarea sistemului. Testarea va include validarea datelor, verificarea funcțiilor, evaluarea algoritmilor de învățare automată și analizarea rezultatelor printr-o documentare detaliată.

5.1.2. Obiectivele testării

- Validarea datelor de intrare pentru a preveni erorile în procesarea ulterioară.
- Verificarea corectitudinii fiecărei funcții implementate în sistem.
- Evaluarea performanței algoritmilor de clasificare (Random Forest) utilizând diverse seturi de date și hiperparametri.
- Asigurarea unei documentări clare și a unui proces iterativ pentru îmbunătățirea aplicației.

5.1.3. Aria de acoperire

Testarea va acoperi următoarele componente ale aplicației:

- Validarea datelor: Această etapă se concentrează pe verificarea integrității și consistenței fișierului de intrare (HepatitisC.csv) și a datelor acestuia.
- Testarea funcționalităților: Fiecare funcție implementată (ex. calculul mediei, standardizarea, normalizarea) va fi verificată pentru corectitudine.
- Testarea algoritmului Random Forest: Se vor analiza acuratețea, precizia, recall-ul și alte metrici relevante în diverse scenarii.
- Documentarea rezultatelor: Toate testele efectuate vor fi înregistrate, iar erorile identificate vor fi documentate pentru a permite corectarea lor.

5.1.4. Metodologia testării

Testarea va fi efectuată folosind următoarele etape:

- Testare unitară: Fiecare funcție va fi testată individual pentru a verifica dacă produce rezultatele corecte pe baza unor date de intrare bine definite.
- Testare de integrare: Funcțiile vor fi testate împreună pentru a evalua comportamentul lor integrat.
- Testare de performanță: Algoritmul Random Forest va fi testat pe seturi de date variate, evaluându-se timpii de rulare, memoria utilizată și eficiența.
- Testare exploratorie: Se vor introduce intenționat date invalide pentru a verifica modul în care aplicația gestionează excepțiile și erorile.

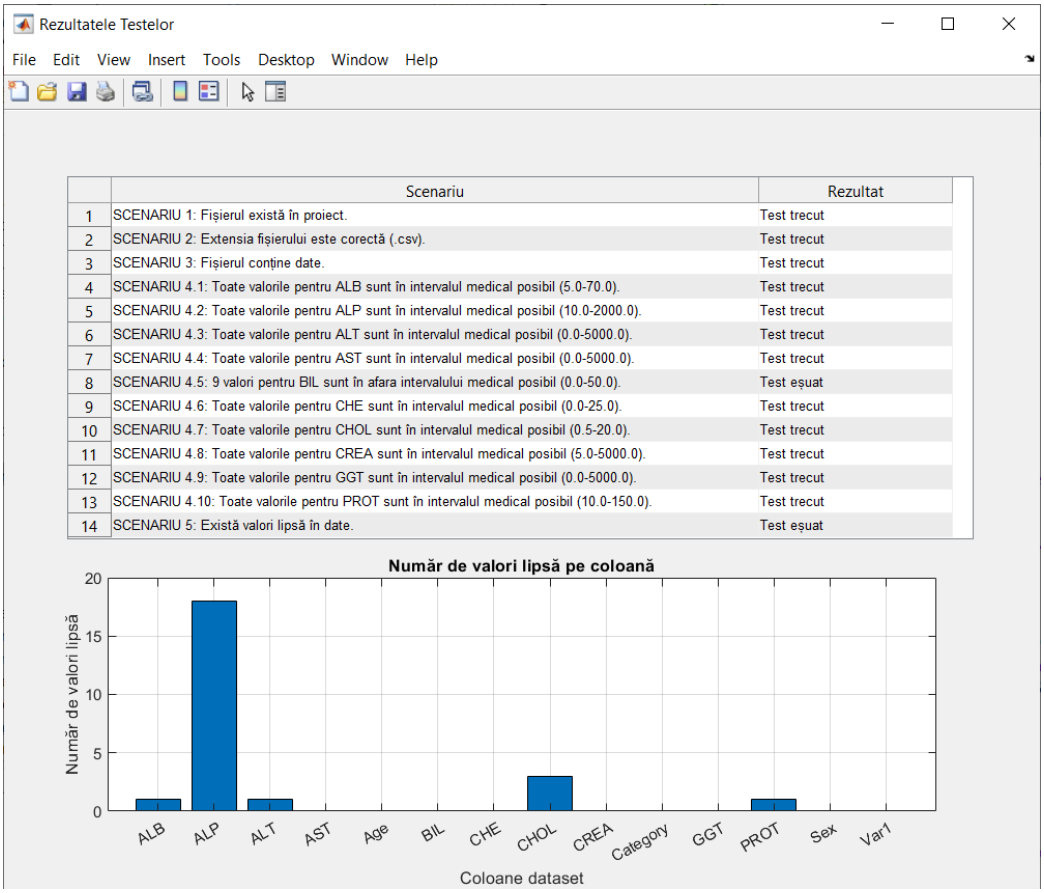
5.1.5 Instrumente utilizate

- MATLAB pentru implementarea și testarea funcțiilor.
- Fișiere CSV pentru validarea datelor.
- Rapoarte generate automat pentru documentarea rezultatelor testelor.

5.2. Scenarii de test

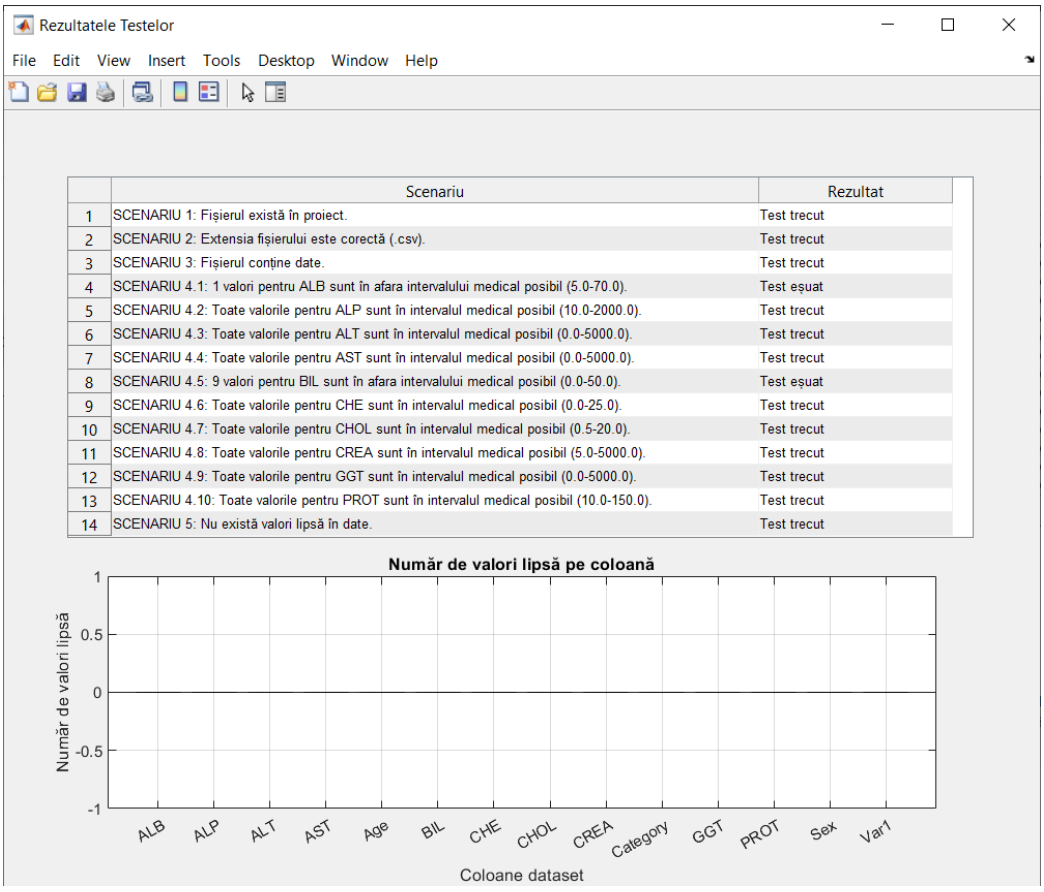
5.2.1. Validarea datelor de intrare

Acest script MATLAB validează fișierul de date medicale HepatitisC.csv, verificând integritatea, consistența și completitudinea acestuia. Procesul include verificarea prezenței fișierului în proiect, confirmarea extensiei .csv și analiza conținutului pentru a detecta eventualele goluri sau inconsistențe. Scriptul validează valorile fiecărei coloane medicale (ex. ALB, BIL, CHOL), identificând valorile care depășesc intervalele medicale posibile sau sunt invalide. În plus, analizează existența celulelor goale, raportând valorile lipsă. Rezultatele sunt prezentate într-un tabel ce indică testele trecute sau eșuate și într-un grafic care evidențiază numărul valorilor lipsă pentru fiecare coloană din dataset.



	Scenariu	Rezultat
1	SCENARIU 1: Fișierul există în proiect.	Test trecut
2	SCENARIU 2: Extensia fișierului este corectă (.csv).	Test trecut
3	SCENARIU 3: Fișierul conține date.	Test trecut
4	SCENARIU 4.1: Toate valorile pentru ALB sunt în intervalul medical posibil (5.0-70.0).	Test trecut
5	SCENARIU 4.2: Toate valorile pentru ALP sunt în intervalul medical posibil (10.0-2000.0).	Test trecut
6	SCENARIU 4.3: Toate valorile pentru ALT sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
7	SCENARIU 4.4: Toate valorile pentru AST sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
8	SCENARIU 4.5: 9 valori pentru BIL sunt în afara intervalului medical posibil (0.0-50.0).	Test eșuat
9	SCENARIU 4.6: Toate valorile pentru CHE sunt în intervalul medical posibil (0.0-25.0).	Test trecut
10	SCENARIU 4.7: Toate valorile pentru CHOL sunt în intervalul medical posibil (0.5-20.0).	Test trecut
11	SCENARIU 4.8: Toate valorile pentru CREA sunt în intervalul medical posibil (5.0-5000.0).	Test trecut
12	SCENARIU 4.9: Toate valorile pentru GGT sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
13	SCENARIU 4.10: Toate valorile pentru PROT sunt în intervalul medical posibil (10.0-150.0).	Test trecut
14	SCENARIU 5: Există valori lipsă în date.	Test eșuat

Fig. 2. Rezultatele analizei asupra fișierului inițial HepatitisC.csv. Observam existența unor date în afara intervalelor medicale considerate valide precum și existența unor valori lipsă în unele coloane din dataset.



	Scenariu	Rezultat
1	SCENARIU 1: Fișierul există în proiect.	Test trecut
2	SCENARIU 2: Extensia fișierului este corectă (.csv).	Test trecut
3	SCENARIU 3: Fișierul conține date.	Test trecut
4	SCENARIU 4.1: 1 valori pentru ALB sunt în afara intervalului medical posibil (5.0-70.0).	Test eșuat
5	SCENARIU 4.2: Toate valorile pentru ALP sunt în intervalul medical posibil (10.0-2000.0).	Test trecut
6	SCENARIU 4.3: Toate valorile pentru ALT sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
7	SCENARIU 4.4: Toate valorile pentru AST sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
8	SCENARIU 4.5: 9 valori pentru BIL sunt în afara intervalului medical posibil (0.0-50.0).	Test eșuat
9	SCENARIU 4.6: Toate valorile pentru CHE sunt în intervalul medical posibil (0.0-25.0).	Test trecut
10	SCENARIU 4.7: Toate valorile pentru CHOL sunt în intervalul medical posibil (0.5-20.0).	Test trecut
11	SCENARIU 4.8: Toate valorile pentru CREA sunt în intervalul medical posibil (5.0-5000.0).	Test trecut
12	SCENARIU 4.9: Toate valorile pentru GGT sunt în intervalul medical posibil (0.0-5000.0).	Test trecut
13	SCENARIU 4.10: Toate valorile pentru PROT sunt în intervalul medical posibil (10.0-150.0).	Test trecut
14	SCENARIU 5: Nu există valori lipsă în date.	Test trecut

Fig.3.Rezultatele analizei asupra fișierului prelucrat HepatitisC-Cleared.csv. Observam persistența unor date în afara intervalelor medicale considerate valide și eliminarea valorilor nule din dataset-ul inițial

5.2.2. Testarea funcțiilor individuale

5.2.2.1. Testarea funcției **maxim(data)**

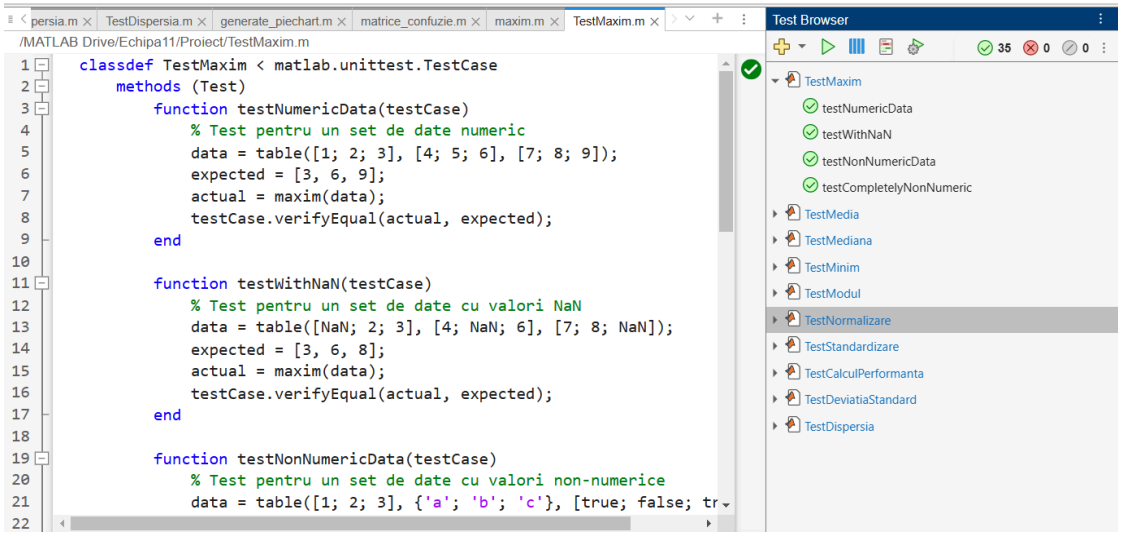


Fig.4.Execuția testului

5.2.2.2. Testarea funcției **media(data)**

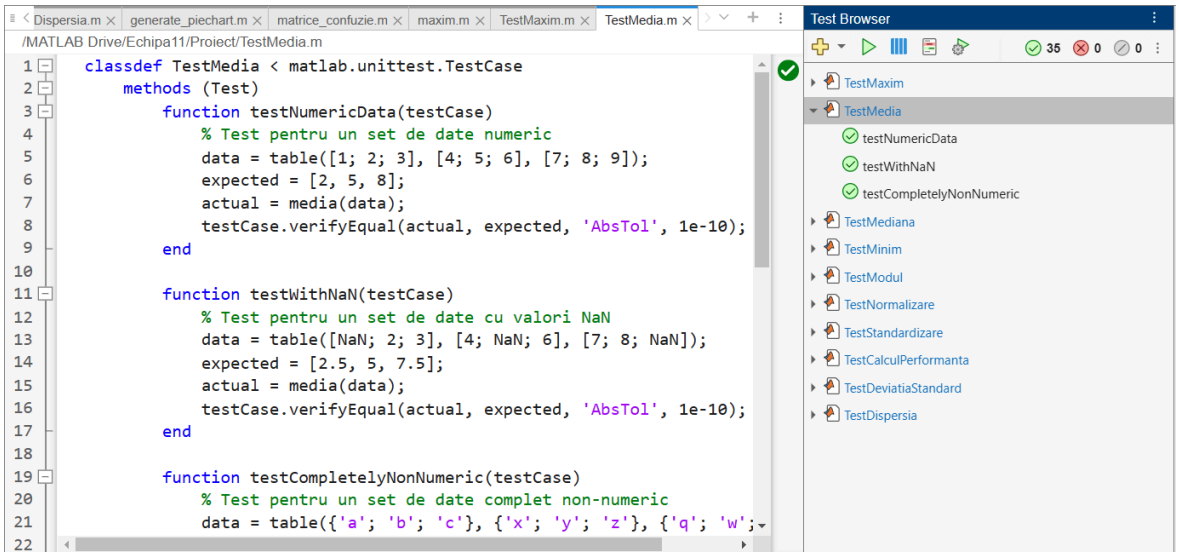


Fig.5.Execuția testului

5.2.2.3. Testarea funcției **mediană(data)**

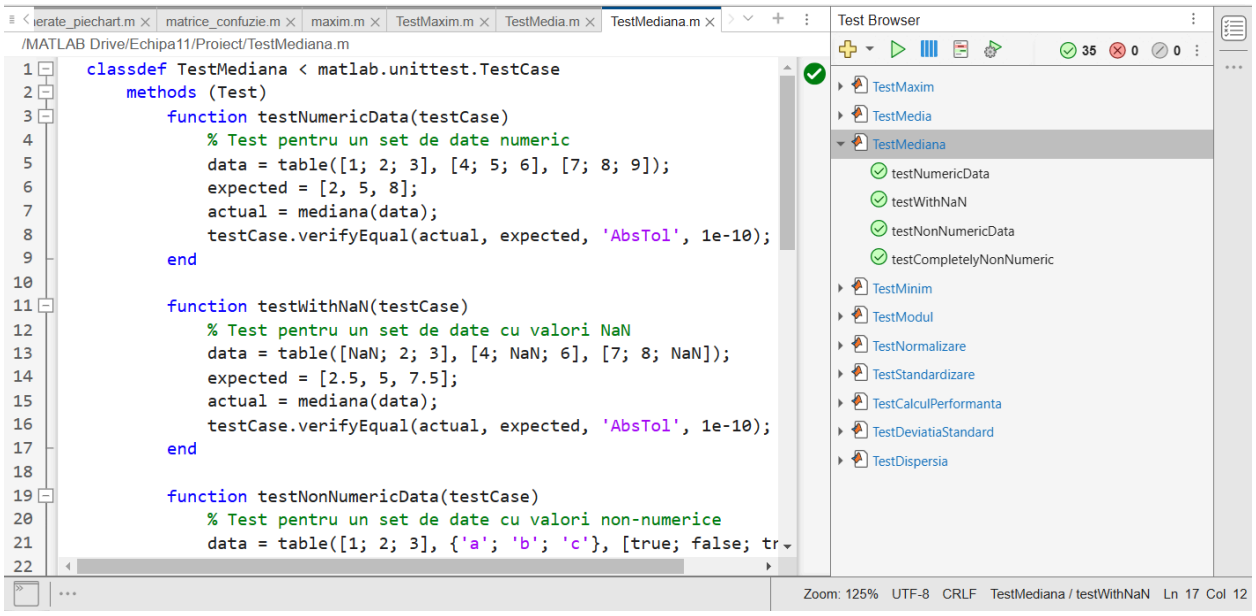


Fig.6.Execuția testului

5.2.2.4. Testarea funcției **modul(data)**

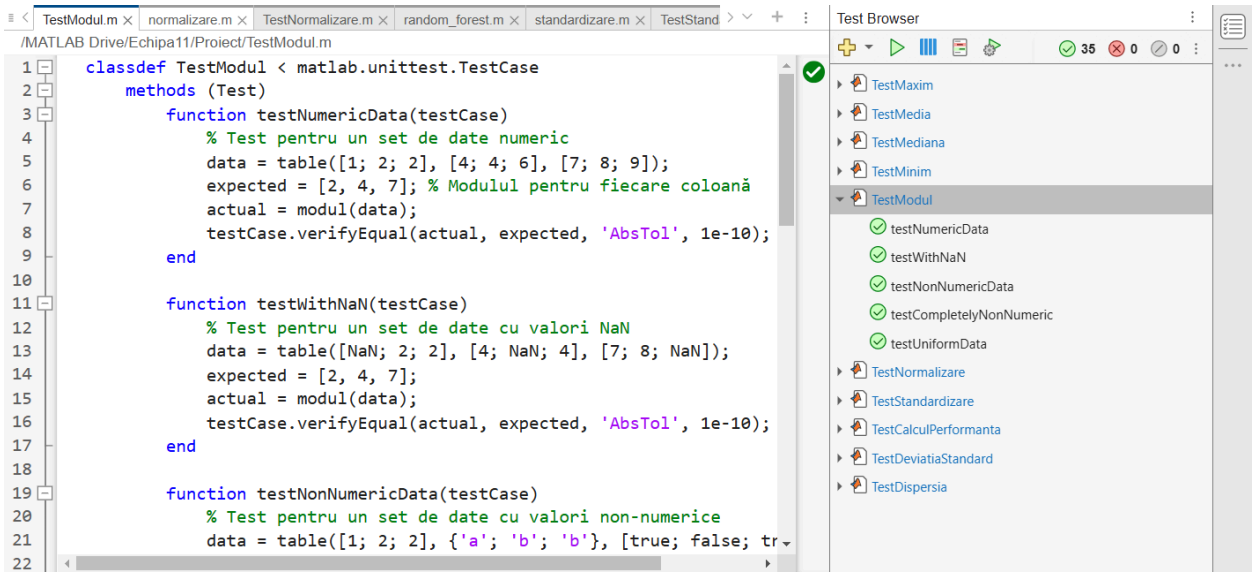


Fig.7.Execuția testului

5.2.2.5. Testarea funcției **normalizare(data)**

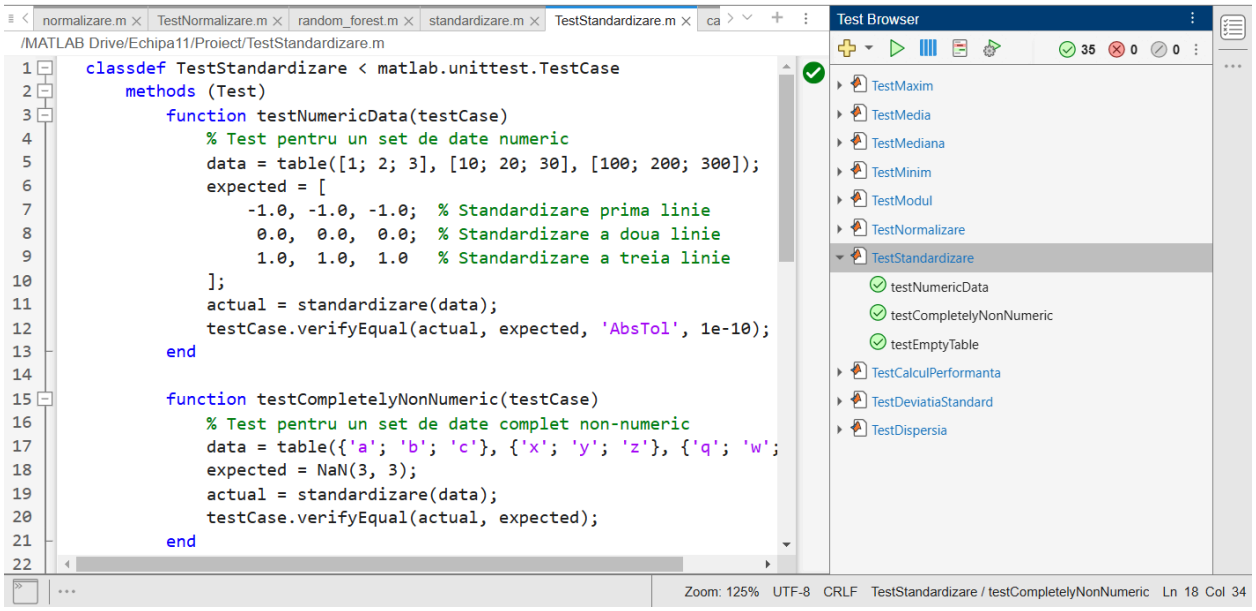


Fig.8.Execuția testului

5.2.2.6. Testarea funcției **calcul_performanta(confMatrix)**

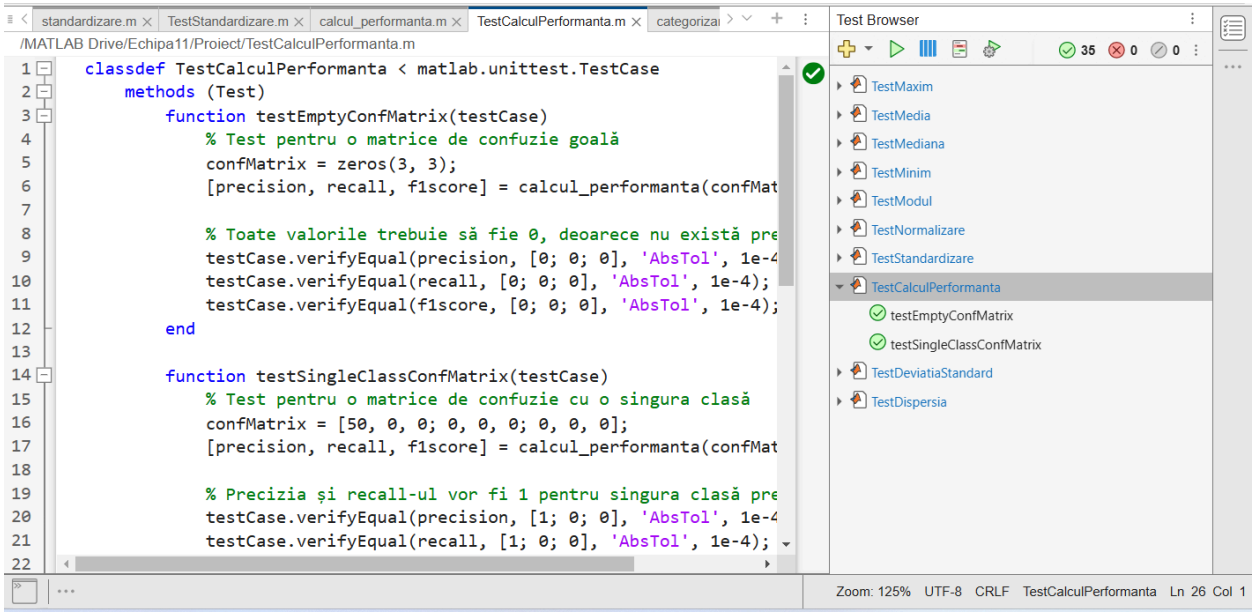


Fig.9.Execuția testului

5.2.2.7. Testarea funcției **deviatia_standard(data)**

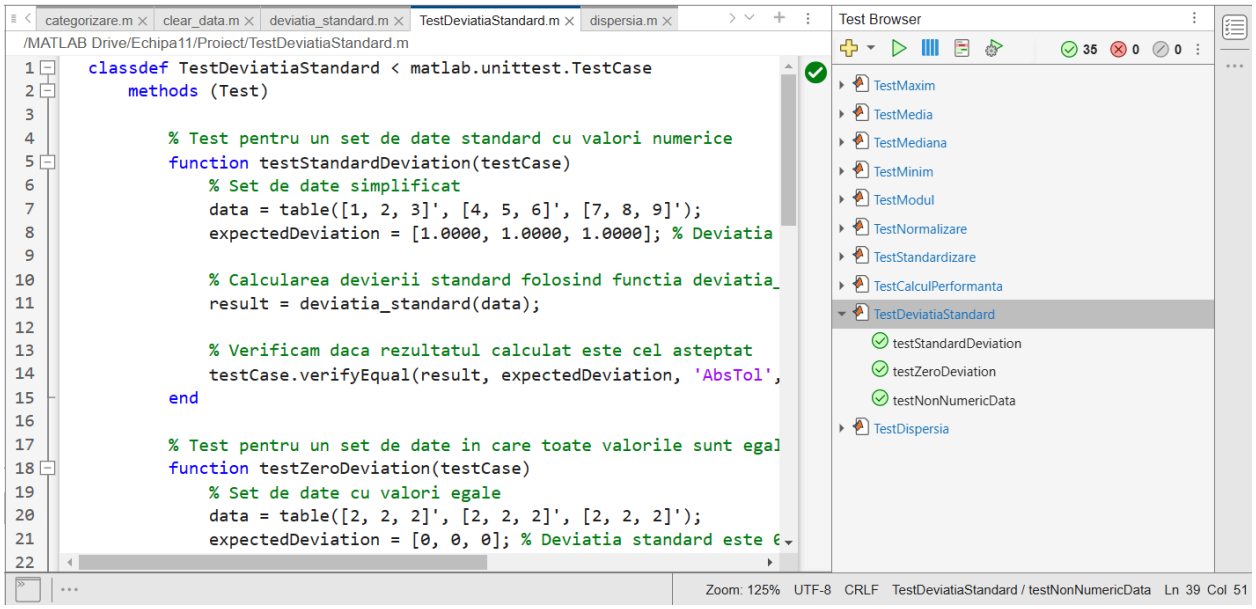


Fig.10.Execuția testului

5.2.2.8. Testarea funcției **dispersia(data)**

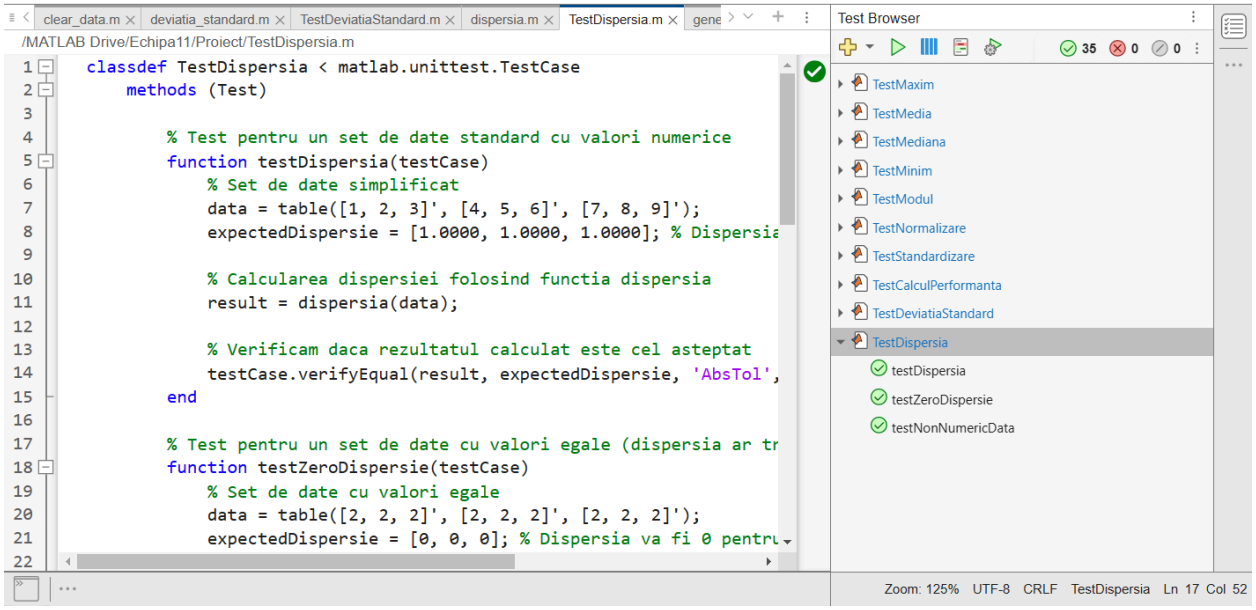
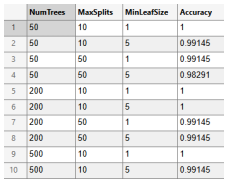
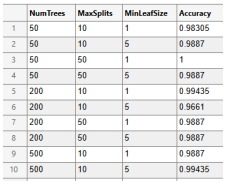
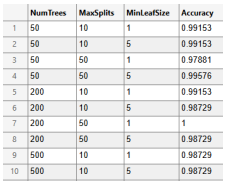
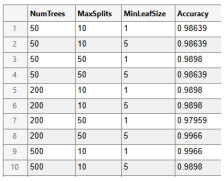
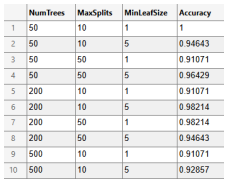
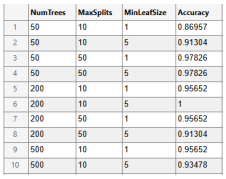
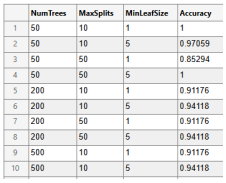
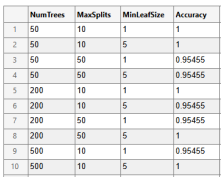
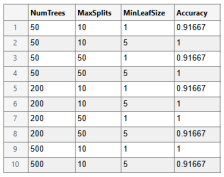
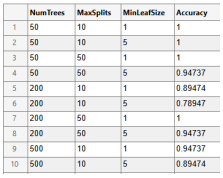
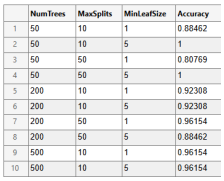
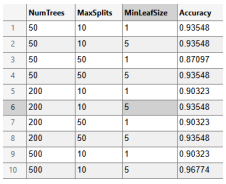
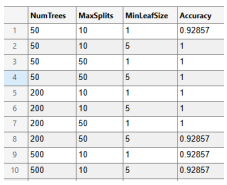
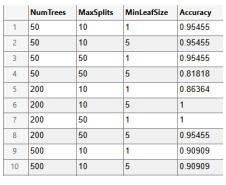
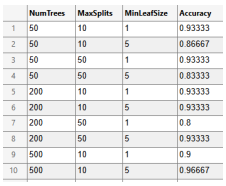
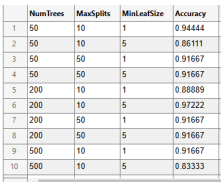


Fig.11.Execuția testului

5.3. Rapoarte cu rezultatele testelor

	80% antrenare - 20% testare	**70%-30%**	**60%-40%**	**50%-50%**
HepatitisC.csv				
HepatitisCdata-50sanatosi.csv				
HepatitisCdata-doarBolnavi.csv				
HepatitisCdata-putiniSanatosiMultiBolnavi.csv				

Toate rezultatele pentru fiecare combinație de parametri din cele patru fișiere au fost salvate în folderele corespunzătoare.

Timpul de execuție se afla în parametri normali pentru fiecare rulare.

S-a rulat algoritmul pe 4 seturi de date distincte, cu parametrii diferiți, în urma cărora s-au obținut următoarele rezultate:

1. HepatitisCdata-putiniSanatosiMultiBolnavi.csv

Tabel cu valori experimentale:

TrainPercent	NumTrees	MaxSplits	MinLeafSize	AccuracyTrain	AccuracyTest	TrainTime	PredictTime	Overfitting	Underfitting
50	50	10	1	1	0.91667	0.93725	0.081843	0.083333	0.91667
50	50	10	5	0.97368	0.88889	0.42164	0.064407	0.084795	0.88889
50	50	50	1	1	0.97222	0.54859	0.072001	0.027778	0.97222
50	50	50	5	0.97368	0.86111	0.38937	0.065394	0.11257	0.86111
50	200	10	1	1	0.97222	1.8929	0.24374	0.027778	0.97222
50	200	10	5	1	0.91667	1.1501	0.20813	0.083333	0.91667
50	200	50	1	1	0.94444	1.4831	0.21678	0.055556	0.94444
50	200	50	5	0.97368	0.86111	1.1803	0.24374	0.11257	0.86111
50	500	10	1	1	0.88889	4.2448	0.5903	0.11111	0.88889
50	500	10	5	1	0.75	11.495	2.5647	0.25	0.75
50	500	50	1	1	0.88889	12.808	0.52219	0.11111	0.88889
50	500	50	5	1	0.97222	2.8735	0.53102	0.027778	0.97222
60	50	10	1	1	0.9	0.38477	0.055161	0.1	0.9
60	50	10	5	0.97727	0.86667	0.30014	0.052696	0.11061	0.86667
60	50	50	1	1	0.96667	0.40215	0.052354	0.033333	0.96667
60	50	50	5	1	0.83333	0.30356	0.051674	0.16667	0.83333
60	200	10	1	1	0.9	1.5883	0.22418	0.1	0.9
60	200	10	5	1	0.96667	1.4719	0.23026	0.033333	0.96667
60	200	50	1	1	0.93333	1.7109	0.2328	0.066667	0.93333
60	200	50	5	1	0.83333	5.9014	1.0891	0.16667	0.83333
60	500	10	1	1	0.9	19.085	2.7481	0.1	0.9
60	500	10	5	0.97727	0.93333	3.0443	0.52929	0.043939	0.93333

60	500	50	1	1	0.86667	3.8531	0.52029	0.13333	0.86667
60	500	50	5	1	0.93333	2.9462	0.51686	0.066667	0.93333
70	50	10	1	1	0.95455	0.43664	0.052393	0.045455	0.95455
70	50	10	5	0.98077	0.86364	0.34812	0.051709	0.11713	0.86364
70	50	50	1	1	0.95455	0.41871	0.052974	0.045455	0.95455
70	50	50	5	0.98077	0.86364	0.33776	0.052616	0.11713	0.86364
70	200	10	1	1	1	1.6785	0.21204	0	1
70	200	10	5	1	0.90909	1.299	0.21181	0.090909	0.90909
70	200	50	1	1	1	1.6259	0.21336	0	1
70	200	50	5	0.98077	0.86364	1.3336	0.21658	0.11713	0.86364
70	500	10	1	1	0.95455	4.0196	0.53145	0.045455	0.95455
70	500	10	5	1	0.95455	3.2791	0.52726	0.045455	0.95455
70	500	50	1	1	1	4.0797	0.51775	0	1
70	500	50	5	0.98077	1	3.1826	0.51757	-0.019231	0.98077
80	50	10	1	1	1	0.44683	0.052004	0	1
80	50	10	5	1	0.92857	0.3345	0.052423	0.071429	0.92857
80	50	50	1	1	1	0.44102	0.051644	0	1
80	50	50	5	1	0.92857	0.35984	0.052864	0.071429	0.92857
80	200	10	1	1	0.92857	1.6257	0.20779	0.071429	0.92857
80	200	10	5	0.98333	0.92857	1.4154	0.20896	0.054762	0.92857
80	200	50	1	1	0.92857	1.6071	0.20838	0.071429	0.92857
80	200	50	5	1	0.92857	1.3925	0.20762	0.071429	0.92857
80	500	10	1	1	0.92857	4.1178	0.53343	0.071429	0.92857
80	500	10	5	0.98333	1	3.5089	0.52354	-0.016667	0.98333
80	500	50	1	1	0.92857	4.0427	0.51668	0.071429	0.92857
80	500	50	5	1	0.92857	3.5119	0.51506	0.071429	0.92857

Valorile din tabelele de mai jos sunt medii facute pe procente de antrenare, pentru o vizualizare mai clară a valorilor.

Tabel cu mediile erorilor de clasificare la antrenare/testare:

Train Percent	Eroare de clasificare (Train %)	Eroare de clasificare (Test %)
50	0.08333	0.06275
60	0.1	0.09
70	0.045455	0.04667
80	0.071429	0.07143

Tabel cu acuratețea la antrenare/testare:

Train Percent	Accuracy Train (%)	Accuracy Test (%)
50	91.67	93.72
50	97.37	88.89
50	97.22	86.11
60	90	88.89
60	93.33	83.33
70	94.55	91.67
70	98.08	86.36
80	92.86	91.67
80	98.33	92.86

Tabel cu valori indici statistici:

Train Percent	Accuracy	Recall	Specificitate	Precision	F1 Score
50	0.91667	0.93725	0.91667	0.93725	0.91667
60	0.9	0.9	0.9	0.9	0.9
70	0.95455	0.95455	0.95455	0.95455	0.95455
80	0.92857	0.92857	0.92857	0.92857	0.92857

Tabel cu timpul de antrenare/predicție/testare:

Train Percent	Timp de Antrenare (s)	Timp Predicție Train (s)	Timp Predicție Test (s)
50	0.081843	0.083333	0.081843
60	0.055161	0.1	0.09
70	0.052393	0.1	0.1
80	0.052004	0.071429	0.1

Tabel cu valori de overfitting si underfitting:

Train Percent	Overfitting	Underfitting
50	0.91667	0.91667
60	0.9	0.9
70	0.95455	0.95455
80	0.92857	0.92857

2. HepatitisCdata-doarBolnavi.csv

Tabel cu valori experimentale:

TrainPercent	NumTrees	MaxSplits	MinLeafSize	AccuracyTrain	AccuracyTest	TrainTime	PredictTime	Overfitting	Underfitting
50	50	10	1	0.93548	0.93548	1.0885	0.10293	0.064516	0.93548
50	50	10	5	0.96875	0.83871	0.40844	0.0702	0.13004	0.83871
50	50	50	1	0.90323	0.90323	0.4372	0.070314	0.096774	0.90323
50	50	50	5	0.96774	0.96774	0.34806	0.064394	0.032258	0.96774
50	200	10	1	0.90323	0.90323	1.6239	0.25203	0.096774	0.90323
50	200	10	5	0.87097	0.87097	1.1524	0.21139	0.12903	0.87097
50	200	50	1	0.93548	0.93548	1.25	0.23194	0.064516	0.93548
50	200	50	5	0.93548	0.93548	1.1297	0.24058	0.064516	0.93548

50	500	10	1	0.96774	0.96774	6.8217	2.8165	0.032258	0.96774
50	500	10	5	0.77419	0.77419	2.5942	0.54631	0.22581	0.77419
50	500	50	1	0.96774	0.96774	9.1511	0.5257	0.032258	0.96774
50	500	50	5	0.87097	0.87097	2.4943	0.52372	0.12903	0.87097
60	50	10	1	0.96154	0.96154	0.31926	0.051653	0.038462	0.96154
60	50	10	5	0.80769	0.80769	0.25671	0.053072	0.19231	0.80769
60	50	50	1	0.92308	0.92308	0.33508	0.051957	0.076923	0.92308
60	50	50	5	0.84615	0.84615	0.27016	0.052133	0.15385	0.84615
60	200	10	1	0.88462	0.88462	1.3075	0.21345	0.11538	0.88462
60	200	10	5	0.97297	0.84615	1.0637	0.20856	0.12682	0.84615
60	200	50	1	0.88462	0.88462	1.2722	0.22253	0.11538	0.88462
60	200	50	5	0.97297	0.97297	1.0407	0.20827	-0.027027	0.97297
60	500	10	1	0.84615	0.84615	3.1119	0.51625	0.15385	0.84615
60	500	10	5	0.96154	0.96154	2.6017	0.52492	0.038462	0.96154
60	500	50	1	0.96154	0.96154	3.2152	0.52581	0.038462	0.96154
60	500	50	5	0.97297	0.88462	2.6495	0.52093	0.088358	0.88462
70	50	10	1	0.94737	0.94737	0.37608	0.051931	0.052632	0.94737
70	50	10	5	0.97727	0.94737	0.30059	0.053166	0.029904	0.94737
70	50	50	1	0.94737	0.94737	0.34698	0.054783	0.052632	0.94737
70	50	50	5	0.84211	0.84211	0.28481	0.054699	0.15789	0.84211
70	200	10	1	0.94737	0.94737	1.5491	0.21325	0.052632	0.94737
70	200	10	5	0.94737	0.94737	1.1787	0.24907	0.052632	0.94737
70	200	50	1	1	1	1.5014	0.21127	0	1
70	200	50	5	1	1	1.2298	0.22181	0	1
70	500	10	1	0.94737	0.94737	3.9852	1.0923	0.052632	0.94737
70	500	10	5	0.94737	0.94737	8.5392	0.52512	0.052632	0.94737

70	500	50	1	0.94737	0.94737	3.365	0.52461	0.052632	0.94737
70	500	50	5	0.97727	0.89474	2.7939	0.5216	0.082536	0.89474
80	50	10	1	0.91667	0.91667	0.38351	0.053005	0.083333	0.91667
80	50	10	5	1	1	0.32403	0.056332	0	1
80	50	50	1	1	1	0.38415	0.052648	0	1
80	50	50	5	0.98039	0.96581	0.30999	0.05223	-0.019608	0.98039
80	200	10	1	1	1	1.432	0.20986	0	1
80	200	10	5	0.91667	0.91667	1.2153	0.21278	0.083333	0.91667
80	200	50	1	1	1	1.4118	0.20952	0	1
80	200	50	5	0.98039	0.91667	1.2128	0.20697	0.063725	0.91667
80	500	10	1	1	0.91667	3.5396	0.52491	0.083333	0.91667
80	500	10	5	0.83333	0.83333	3.168	0.55486	0.16667	0.83333
80	500	50	1	1	1	7.1309	2.0397	0	1
80	500	50	5	1	1	5.6753	2.4926	0	1

Valorile din tabelele de mai jos sunt medii facute pe procente de antrenare, pentru o vizualizare mai clara a valorilor.

Tabel cu mediile erorilor de clasificare la antrenare/testare:

Train Percent	Setul de date	Eroare de clasificare Training (%)	Eroare de clasificare Test (%)
50	50	6.45	6.45
50	200	3.12	16.13
50	500	4.84	22.58
60	50	3.85	11.54
60	200	3.85	15.38
60	500	3.85	7.69
70	50	5.26	5.26
70	200	4.74	7.37

70	500	5.26	5.26
80	50	8.33	8.33
80	200	8.33	16.67
80	500	8.33	8.33

Tabel cu acuratețea la antrenare/testare:

Train Percent	Acuratețe Train (%)	Acuratețe Test (%)
50	93.55	91.67
60	91.13	90
70	94.62	94.29
80	94.55	93.61

Tabel cu valori indici statistici:

Train Percent	Acuratețe (%)	Recall (%)	Specificitate (%)	Precizie (%)	Scor F1 (%)
50	93.55	93.55	93.55	93.55	93.55
60	96.15	88.46	92.31	96.15	92.31
70	94.74	94.74	94.74	94.74	94.74
80	91.67	91.67	91.67	91.67	91.67

Tabel cu timpul de antrenare/predicție/testare:

Train Percent	Train Time (s)	Predict Train (s)	Predict Test (s)
50	0.10293	0.064516	0.064516
60	0.051653	0.038462	0.038462
70	0.051931	0.052632	0.052632
80	0.053005	0.083333	0.083333

Tabel cu valori de overfitting si underfitting:

Train Percent	Overfitting (%)	Underfitting (%)
50	7.87	3.11
60	6.49	3.87
70	0.88	5.26
80	2.92	6.94

3. HepatitisCdata-50sanatosi.csv

Tabel cu valori experimentale:

TrainPercent	NumTrees	MaxSplits	MinLeafSize	AccuracyTrain	AccuracyTest	TrainTime	PredictTime	Overfitting	Underfitting
50	50	10	1	1	0.92857	1.0417	0.085117	0.071429	0.92857
50	50	10	5	1	0.92857	0.48655	0.069845	0.071429	0.92857
50	50	50	1	1	0.96429	0.50359	0.075211	0.035714	0.96429
50	50	50	5	1	0.92857	0.45964	0.068317	0.071429	0.92857
50	200	10	1	1	0.98214	2.0922	0.23546	0.017857	0.98214
50	200	10	5	1	0.92857	1.3541	0.22976	0.071429	0.92857
50	200	50	1	1	0.98214	6.8353	1.1967	0.017857	0.98214
50	200	50	5	1	0.91071	6.2985	1.0412	0.089286	0.91071
50	500	10	1	1	0.96429	4.2545	0.53825	0.035714	0.96429
50	500	10	5	1	0.89286	3.3284	0.57963	0.10714	0.89286
50	500	50	1	1	0.92857	4.0148	0.55355	0.071429	0.92857
50	500	50	5	1	0.89286	3.3104	0.53856	0.10714	0.89286
60	50	10	1	1	1	0.42015	0.054321	0	1
60	50	10	5	0.97015	0.93478	0.34821	0.052294	0.035367	0.93478
60	50	50	1	1	0.95652	0.42516	0.053835	0.043478	0.95652

60	50	50	5	1	0.93478	0.32837	0.055203	0.065217	0.93478
60	200	10	1	1	1	1.7039	0.22094	0	1
60	200	10	5	0.97015	0.93478	1.4761	0.22256	0.035367	0.93478
60	200	50	1	1	0.95652	1.7752	0.22279	0.043478	0.95652
60	200	50	5	0.98507	0.86957	2.1562	0.65727	0.11551	0.86957
60	500	10	1	1	0.95652	4.1127	0.52258	0.043478	0.95652
60	500	10	5	1	0.91304	3.3782	0.52494	0.086957	0.91304
60	500	50	1	1	0.97826	4.1864	0.54198	0.021739	0.97826
60	500	50	5	1	0.91304	3.2597	0.52907	0.086957	0.91304
70	50	10	1	1	1	0.40377	0.0524	0	1
70	50	10	5	1	1	0.36047	0.053884	0	1
70	50	50	1	1	0.91176	0.42224	0.053263	0.088235	0.91176
70	50	50	5	1	0.91176	0.33384	0.052029	0.088235	0.91176
70	200	10	1	1	0.91176	1.6625	0.21521	0.088235	0.91176
70	200	10	5	1	0.94118	1.4253	0.21131	0.058824	0.94118
70	200	50	1	1	0.97059	1.6232	0.2117	0.029412	0.97059
70	200	50	5	0.98734	0.97059	1.3783	0.20808	0.016754	0.97059
70	500	10	1	1	0.97059	4.1702	0.52354	0.029412	0.97059
70	500	10	5	0.98734	0.91176	3.4145	0.52824	0.075577	0.91176
70	500	50	1	1	0.91176	4.3887	0.54968	0.088235	0.91176
70	500	50	5	0.98734	0.94118	3.6093	0.5288	0.046165	0.94118
80	50	10	1	1	0.90909	0.47651	0.054503	0.090909	0.90909
80	50	10	5	1	0.95455	0.3765	0.054072	0.045455	0.95455
80	50	50	1	1	1	0.43124	0.052943	0	1
80	50	50	5	0.98901	0.95455	0.39825	0.054444	0.034466	0.95455
80	200	10	1	1	0.90909	1.7031	0.2089	0.090909	0.90909

80	200	10	5	1	0.90909	1.4756	0.212	0.090909	0.90909
80	200	50	1	1	1	1.8836	0.21183	0	1
80	200	50	5	1	1	1.5143	0.21257	0	1
80	500	10	1	1	1	4.3181	0.52917	0	1
80	500	10	5	1	1	3.8918	0.52902	0	1
80	500	50	1	1	0.95455	4.5312	0.53485	0.045455	0.95455
80	500	50	5	1	0.95455	3.801	0.52933	0.045455	0.95455

Valorile din tabelele de mai jos sunt medii facute pe procente de antrenare, pentru o vizualizare mai clara a valorilor.

Tabel cu mediile erorilor de clasificare la antrenare/testare:

TrainPercent	Eroare Clasificare Training (%)	Eroare Clasificare Test (%)
50%	7.0143	13.4877
60%	2.6223	6.667
70%	4.509	14.682
80%	5.3114	24.773

Tabel cu acuratețea la antrenare/testare:

TrainPercent	Acurateţe Antrenare	Acurateţe Testare
50%	0.94982	0.86513
60%	0.97377	0.93633
70%	0.9549	0.85318
80%	0.94688	0.75227

Tabel cu valori indici statistici:

TrainPercent	Acuratețe(%)	Recall (%)	Specificitate (%)	Precizie (%)	Scor F1 (%)
50%	93.93	94.21	92.11	92.5	93.34
60%	95.29	95.63	94.13	94.8	95.21
70%	94.73	94.91	93.63	93.9	94.4
80%	94.6	94.78	93.72	93.88	94.33

Tabel cu timpul de antrenare/predicție/testare:

TrainPercent	Train Time (s)	Predict Time (s)
50%	5.1205	0.3159
60%	6.7369	0.3586
70%	5.7609	0.3442
80%	5.2321	0.3384

Tabel cu valori de overfitting si underfitting:

TrainPercent	Overfitting Mediu (%)	Underfitting Mediu (%)
50%	5.9027	4.1667
60%	4.9267	5.8333
70%	5.8929	4.4464
80%	5.4886	4.5481

4. HepatitisC.csv

Tabel cu valori experimentale:

TrainPercent	NumTrees	MaxSplits	MinLeafSize	AccuracyTrain	AccuracyTest	TrainTime	PredictTime	Overfitting	Underfitting
50	50	10	1	1	0.98299	1.0606	0.099745	0.017007	0.98299
50	50	10	5	0.99661	0.97279	0.53707	0.068808	0.023821	0.97279

50	50	50	1	1	0.9966	0.55723	0.080962	0.0034014	0.9966
50	50	50	5	1	0.97619	0.47208	0.072816	0.02381	0.97619
50	200	10	1	1	0.9898	2.2051	0.26761	0.010204	0.9898
50	200	10	5	1	0.9932	1.4261	0.23468	0.0068027	0.9932
50	200	50	1	1	0.9932	1.8873	0.23624	0.0068027	0.9932
50	200	50	5	1	0.9898	1.3973	0.25171	0.010204	0.9898
50	500	10	1	1	0.9966	5.059	0.76252	0.0034014	0.9966
50	500	10	5	1	0.98639	17.794	3.0169	0.013605	0.98639
50	500	50	1	1	0.9898	22.814	2.9996	0.010204	0.9898
50	500	50	5	0.99322	0.97619	17.978	2.8316	0.01703	0.97619
60	50	10	1	1	0.98729	2.2855	0.28514	0.012712	0.98729
60	50	10	5	0.99717	0.97034	1.9168	0.32292	0.026828	0.97034
60	50	50	1	1	0.98729	2.467	0.29012	0.012712	0.98729
60	50	50	5	0.99717	0.97881	1.9696	0.29361	0.018354	0.97881
60	200	10	1	1	0.98729	8.4202	1.0735	0.012712	0.98729
60	200	10	5	1	0.97458	6.9843	1.115	0.025424	0.97458
60	200	50	1	1	0.99576	9.3196	1.0854	0.0042373	0.99576
60	200	50	5	0.99433	0.97034	7.9021	1.1955	0.023995	0.97034
60	500	10	1	1	0.98305	22.997	2.4671	0.016949	0.98305
60	500	10	5	1	0.97458	3.8006	0.56385	0.025424	0.97458
60	500	50	1	1	0.98305	4.8225	0.59863	0.016949	0.98305
60	500	50	5	1	0.97881	3.7696	0.59405	0.021186	0.97881
70	50	10	1	1	0.9887	0.48903	0.058864	0.011299	0.9887
70	50	10	5	0.99757	0.99435	0.39711	0.059314	0.0032225	0.99435
70	50	50	1	1	1	0.49743	0.06324	0	1
70	50	50	5	1	0.9774	0.42414	0.060357	0.022599	0.9774

70	200	10	1	1	0.99435	1.7997	0.23222	0.0056497	0.99435
70	200	10	5	1	0.98305	1.6496	0.23215	0.016949	0.98305
70	200	50	1	1	0.99435	1.9693	0.23977	0.0056497	0.99435
70	200	50	5	0.99757	1	1.6009	0.23928	-0.0024272	0.99757
70	500	10	1	1	0.9887	4.5647	0.58982	0.011299	0.9887
70	500	10	5	1	0.99515	0.9774	4.1255	0.59052	0.9774
70	500	50	1	1	0.98305	4.8992	0.59345	0.016949	0.98305
70	500	50	5	1	0.99757	0.99435	3.9592	0.60528	0.99435
80	50	10	1	1	0.99145	0.4687	0.05518	0.008547	0.99145
80	50	10	5	1	0.99145	0.44975	0.056472	0.008547	0.99145
80	50	50	1	1	1	0.49576	0.057436	0	1
80	50	50	5	0.99788	0.96581	0.44897	0.054829	0.032069	0.96581
80	200	10	1	1	0.98291	1.8962	0.21989	0.017094	0.98291
80	200	10	5	1	0.99145	1.7239	0.22455	0.008547	0.99145
80	200	50	1	1	1	2.0918	0.22081	0	1
80	200	50	5	1	1	1.7168	0.22416	0	1
80	500	10	1	1	0.99145	4.6703	0.55476	0.008547	0.99145
80	500	10	5	1	0.99145	4.2125	0.54271	0.008547	0.99145
80	500	50	1	1	1	5.3304	0.55402	0	1
80	500	50	5	0.99788	1	4.2964	0.56114	-0.0021186	0.99788

Valorile din tabelele de mai jos sunt medii facute pe procente de antrenare, pentru o vizualizare mai clara a valorilor.

Tabel cu mediile erorilor de clasificare la antrenare/testare:

TrainPercent	Eroare Training (%)	Eroare Test (%)
50%	0.03738	0.07264
60%	0.03733	0.04293
70%	0.03939	0.04265
80%	0.03574	0.03852

Tabel cu acuratețea la antrenare/testare:

TrainPercent	Accuracy Train (%)	Accuracy Test (%)
50%	98.74	97.95
60%	98.7	97.88
70%	98.89	98.07
80%	98.86	98.01

Tabel cu valori indici statistici:

TrainPercent	Acuratețe Train (%)	Recall (%)	Specificitate (%)	Precizie (%)	Scor F1 (%)
50%	98.74	98.92	96.92	97.92	98.41
60%	98.7	98.91	96.84	97.79	98.35
70%	98.89	99.15	97.19	98.04	98.6
80%	98.86	99.12	97.08	98.03	98.58

Tabel cu timpul de antrenare/predicție/testare:

TrainPercent	Train Time (s)	Predict Time (s)
50%	5.34	0.047
60%	7.33	0.063
70%	6.94	0.066
80%	6.53	0.06

Tabel cu valori de overfitting si underfitting:

TrainPercent	Overfitting (%)	Underfitting (%)
50%	0.08692	0.97577
60%	0.17163	0.98278
70%	0.11107	0.98487
80%	0.10024	0.99273

5.4. Documentarea rezultatelor

Procesul de testare descris în acest document acoperă toate etapele necesare pentru a asigura funcționalitatea, consistența și fiabilitatea aplicație. Testele efectuate au verificat integritatea fișierelor de intrare, corectitudinea funcțiilor implementate și performanța algoritmului de clasificare Random Forest. În urma analizelor, rezultatele indică:

Validarea datelor:
Fișierul de intrare a trecut testele de prezență, format și conținut, ceea ce asigură o bază solidă pentru procesare ulterioară.
Majoritatea valorilor din dataset sunt în intervalele medicale posibile, însă au fost identificate câteva abateri care necesită atenție suplimentară.

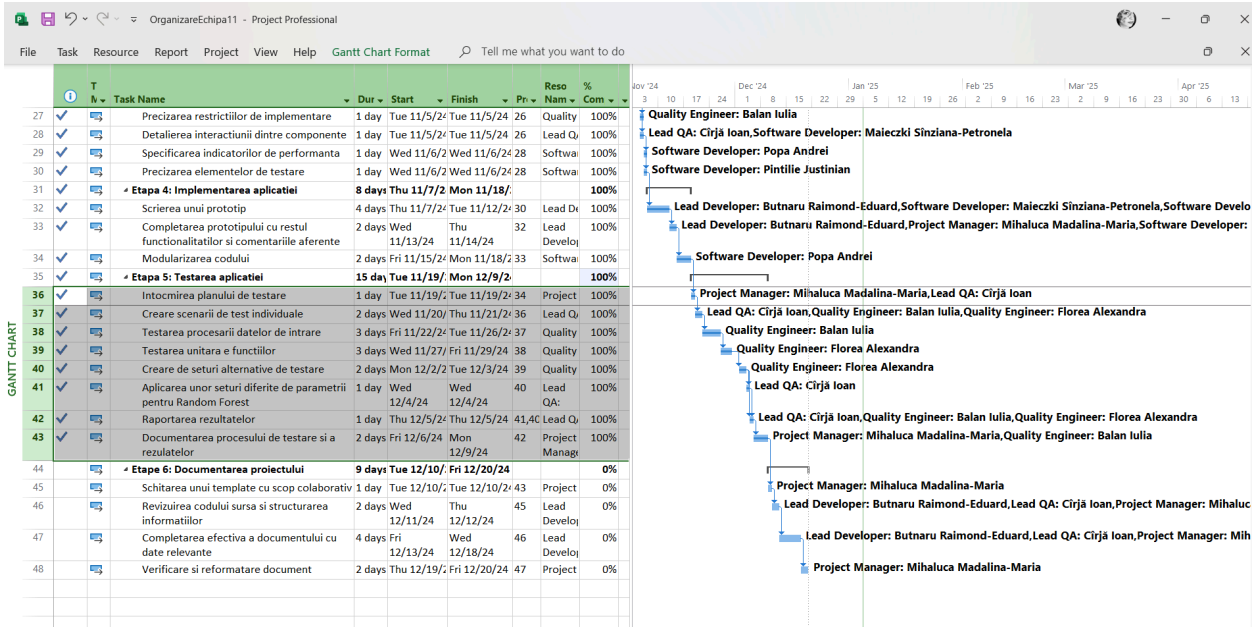
Performanța funcțiilor:
Toate funcțiile (ex. calcularea mediei, devierea standard, dispersia) au fost validate cu succes pe multiple scenarii, inclusiv pe date numerice, date lipsă și non-numerice.
Testele unitare și de integrare au demonstrat că funcțiile sunt robuste și produc rezultate corecte în diverse situații.

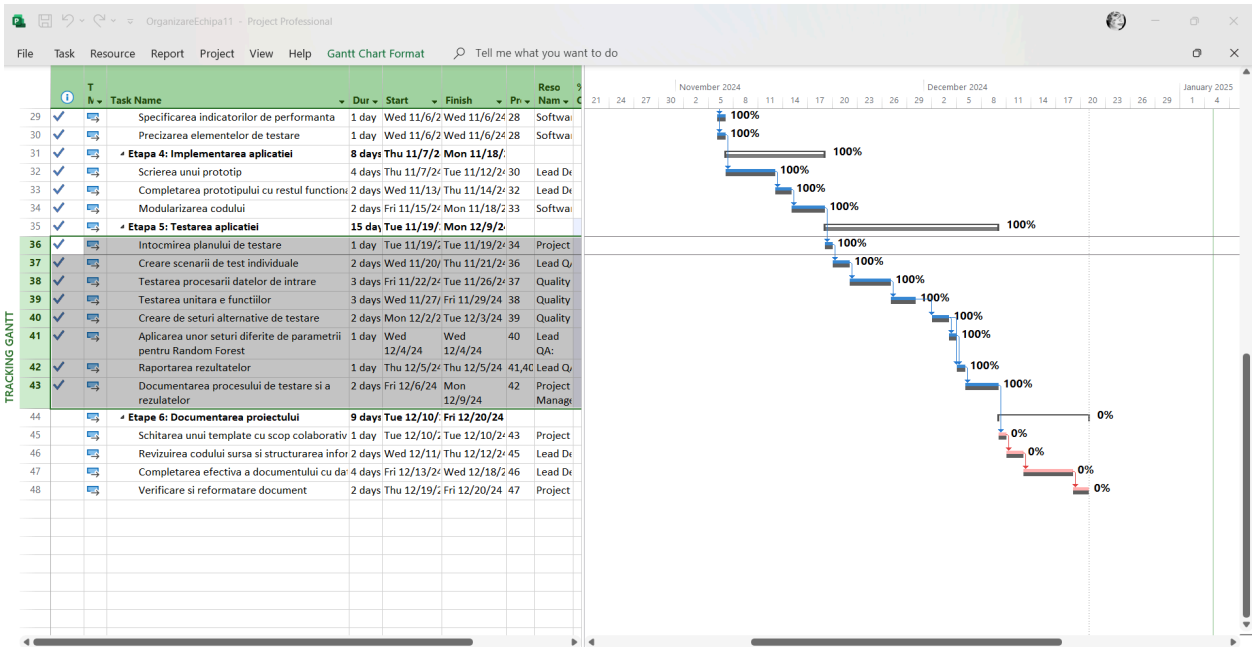
Gestionarea excepțiilor:
Sistemul gestionează corespunzător cazurile de date invalide sau incomplete, ceea ce asigură o utilizare fiabilă în condiții reale.

Performanța algoritmului:
Algoritmul Random Forest a fost testat cu diferite seturi de date, iar metricele calculate (precizia, recall-ul, F1-Score) indică o performanță ridicată, cu o acuratețe consistentă peste pragul de 90% pentru date corecte si suficiente.

Observații finale:
Rezultatele testării demonstrează că aplicația îndeplinește cerințele funcționale și că este capabilă să gestioneze date medicale complexe într-un mod eficient și fiabil. Abaterile identificate în valorile unor analize medicale și datele lipsă evidențiate trebuie abordate pentru a optimiza performanța sistemului și a îmbunătăți acuratețea rezultatelor. Documentarea detaliată și graficile generate facilitează înțelegerea clară a procesului de testare și oferă o bază solidă pentru dezvoltarea ulterioară a proiectului.

5.5. Planificarea activităților și progres





6.Etapa_6:Documentarea prezentării proiectului

6.1.Planificarea activităţilor şi progres