Introducere

Descrierea datelor si scopul cercetarii

Setul de date privind diagnosticul bolilor de inimă conține înregistrări medicale și informațiile asociate utilizate pentru a determina probabilitatea ca indivizii să sufere de o boala la inima. Acesta include factori precum vârsta, sexul, tipul durerii toracice, tensiunea arterială în repaus, colesterolul seric, glicemia la jeun, rezultatele electrocardiogramei de repaus, frecvența cardiacă maximă atinsă, angina indusă de efort, depresia ST indusă de efort relativ la repaus, panta segmentului ST de vârf în timpul efortului, numărul de vase majore colorate prin fluoroscopie și o variabilă categorică care indică prezența sau absența bolii la inima.

Scopul principal al acestui set de date este de a permite dezvoltarea și evaluarea modelelor de învățare automată pentru a prezice probabilitatea prezenței bolilor de inimă. Prin utilizarea acestui set de date, instituțiile medicale pot reduce riscul de diagnostic greșit și pot lua decizii mai informate în privința planurilor de tratament pentru pacienți. Analiza detaliată a datelor despre pacienți și indicatorii bolilor de inimă permite identificarea modelelor și tendințelor relevante, ceea ce este crucial pentru elaborarea protocoalelor de diagnostic și gestionarea riscului pacienților.

Proiectul oferă oportunitatea de a aplica diverse tehnici de machine learning pentru a îmbunătăți procesul de luare a deciziilor și pentru a dezvolta strategii de intervenție eficiente pentru pacienții cu risc ridicat. Acest lucru nu numai că ajută la menținerea sănătății și bunăstării indivizilor, dar și la sprijinirea furnizorilor de servicii medicale în oferirea unor evaluări mai precise și mai echitabile pentru pacienți.

Sursa datelor: <https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset/code>

VARIABILELE:

**Numerice (5):**

1. **age** – vârsta pacienților.
2. **trtbps** – tensiunea arterială în repaus (în mm Hg).
3. **chol** – nivelul total de colesterol din sânge. (mg/dl).
4. **thalachh** –valoarea maximă a frecvenței cardiace atinsă în timpul exercițiului fizic.
5. **oldpeak** – depresia segmentului ST indusă de exercițiu relativ la repaus.

**Calitative (8):**

1. **sex** – sexul pacienților.
   * Descriere: Sexul pacienților (1 = masculin, 0 = feminin).
2. **exng** – angina indusă de exercițiu.
   * Descriere: Angina indusă de exercițiu (1 = Da, 0 = Nu).
3. **caa** – numărul de vase majore colorate prin fluoroscopie.
   * Descriere: Numărul de vase majore ale inimii vizibile în timpul unei proceduri imagistice (0, 1, 2 sau 3).
4. **cp** – tipul durerii toracice.
   * Descriere: Tipul durerii toracice experimentate de pacienți (0 = angină, 1 = angină atipică, 2 = durere non-anginoasă, 3 = asimptomatic).
5. **fbs** – glicemia la jeun (mai mare de 120 mg/dl).
   * Descriere: Indicator dacă glicemia la jeun este peste 120 mg/dl (1 = Da, 0 = Nu).
6. **restecg** – rezultatele electrocardiogramei de repaus.
   * Descriere: Rezultatele ECG de repaus (0 = normal, 1 = anomalii ST-T, 2 = hipertrofie ventriculară stângă posibilă sau definitivă).
7. **slp** – panta segmentului ST de vârf în timpul exercițiului.
   * Descriere: Indică panta segmentului ST în timpul exercițiului fizic (0 = în sus, 1 = plat, 2 = în jos).
8. **thall** – defectul thalassemiei.
   * Descriere: Defectul thalassemiei (0 = normal, 1 = defect fix, 2 = defect reversibil).

**Variabilă țintă:**

1. **output** – prezența sau absența bolii de inimă.
   * Descriere: Statutul bolii de inimă (1 = prezență, 0 = absență).

Table 1: Setul de date importat in R

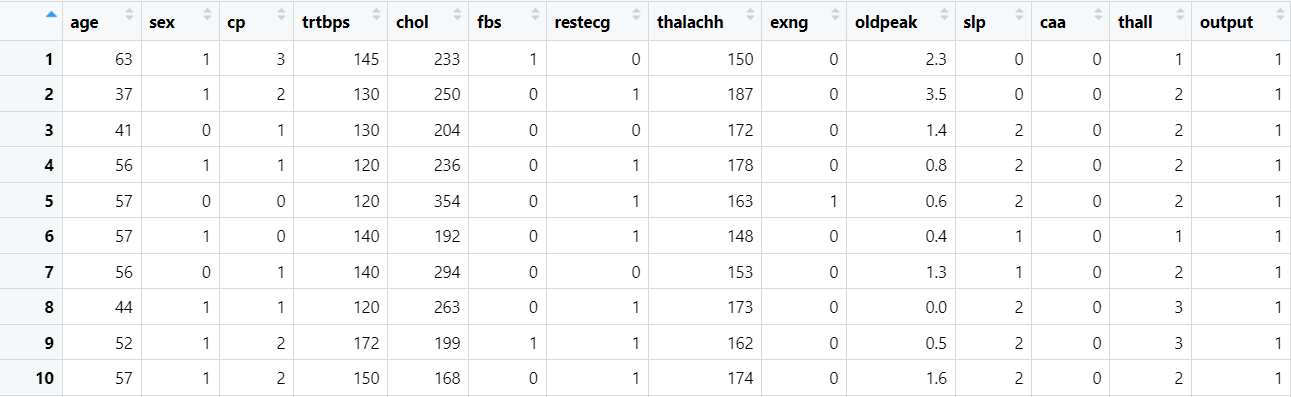
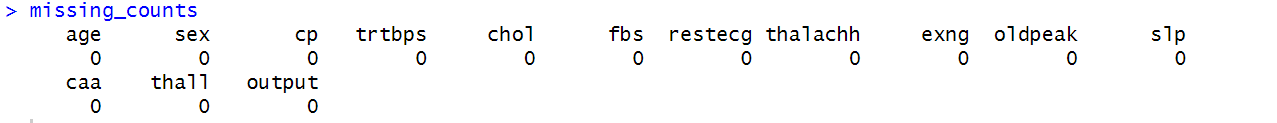
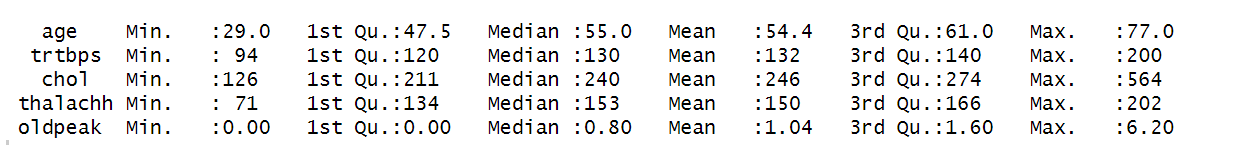


Table 2: Lipsa valorilor NA in setul meu de date



Statistici descriptive

Table 3: Summary - varibaile numerice



În ceea ce privește variabila **age** (vârsta), se observă că vârsta minimă a pacienților este de 29 de ani, în timp ce vârsta maximă ajunge la 77 de ani. Vârsta mediană este de 55 de ani, ceea ce indică faptul că jumătate dintre pacienți au vârsta sub 55 de ani. Vârsta medie a pacienților este de 54.4 ani, sugerând o concentrare a pacienților în jurul vârstei medii de 50 de ani.

Variabila **trtbps** (tensiunea arterială în repaus) arată că tensiunea arterială minimă în repaus este de 94 mm Hg, iar cea maximă este de 200 mm Hg. Tensiunea arterială mediană este de 130 mm Hg, cu o medie de 132 mm Hg. Aceasta indică o variabilitate considerabilă a tensiunii arteriale între pacienți, dar cu o tendință de a fi ușor peste valorile normale recomandate.

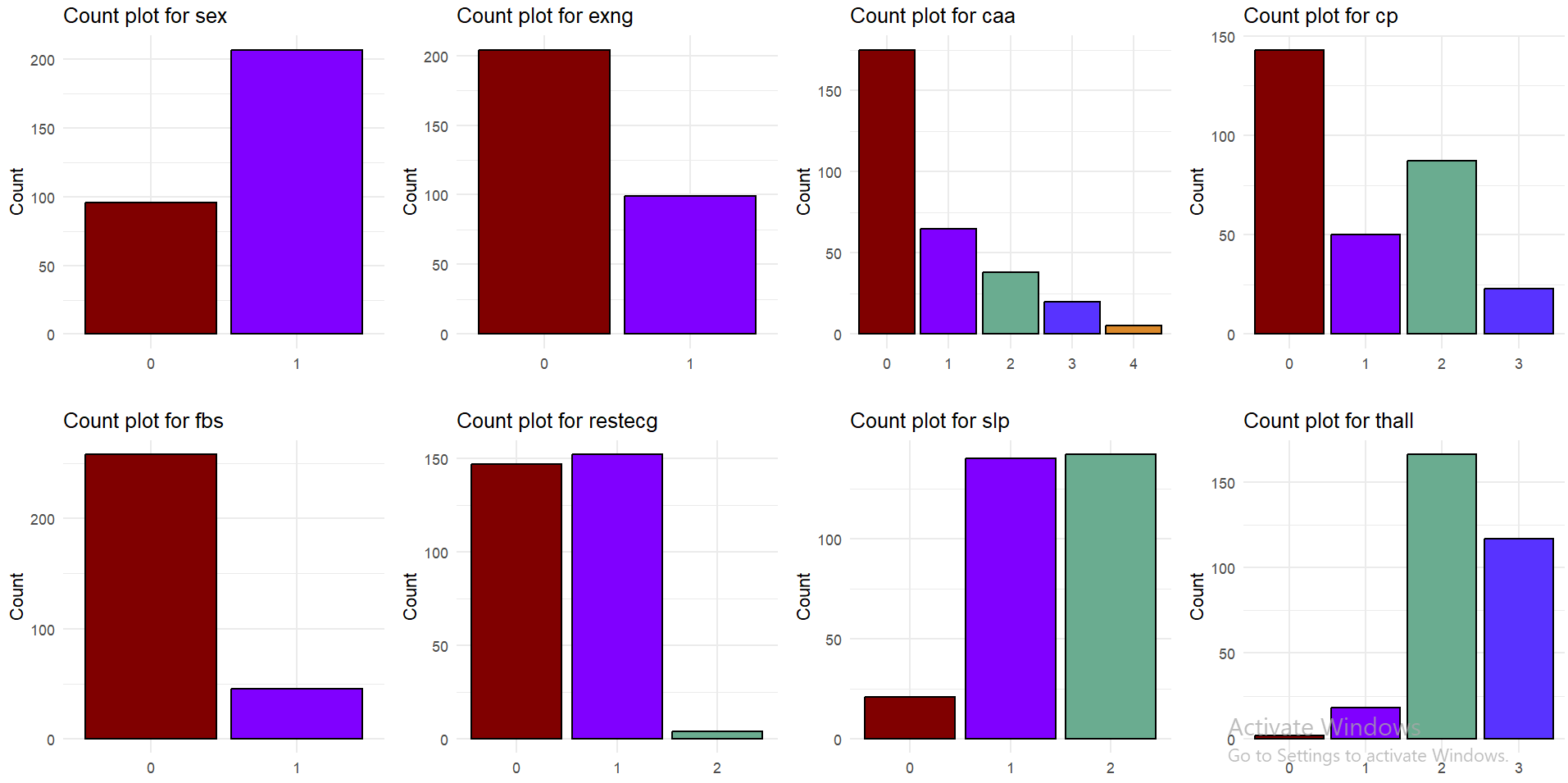
Pentru variabila **chol** (colesterol seric), nivelurile variază de la un minim de 126 mg/dl până la un maxim de 564 mg/dl. Nivelul median de colesterol este de 240 mg/dl, iar media este de 246 mg/dl. Aceste valori sugerează că mulți dintre pacienți au niveluri de colesterol ridicate, ceea ce este un factor de risc semnificativ pentru bolile de inimă.

Variabila **thalachh** (frecvența cardiacă maximă atinsă) indică o frecvență cardiacă minimă de 71 bpm și o maximă de 202 bpm. Frecvența cardiacă mediană este de 153 bpm, iar media este de 150 bpm. Aceste date sugerează că majoritatea pacienților ating frecvențe cardiace maxime ridicate în timpul exercițiilor fizice, ceea ce poate reflecta atât capacitatea lor cardiovasculară, cât și nivelul de stres indus.

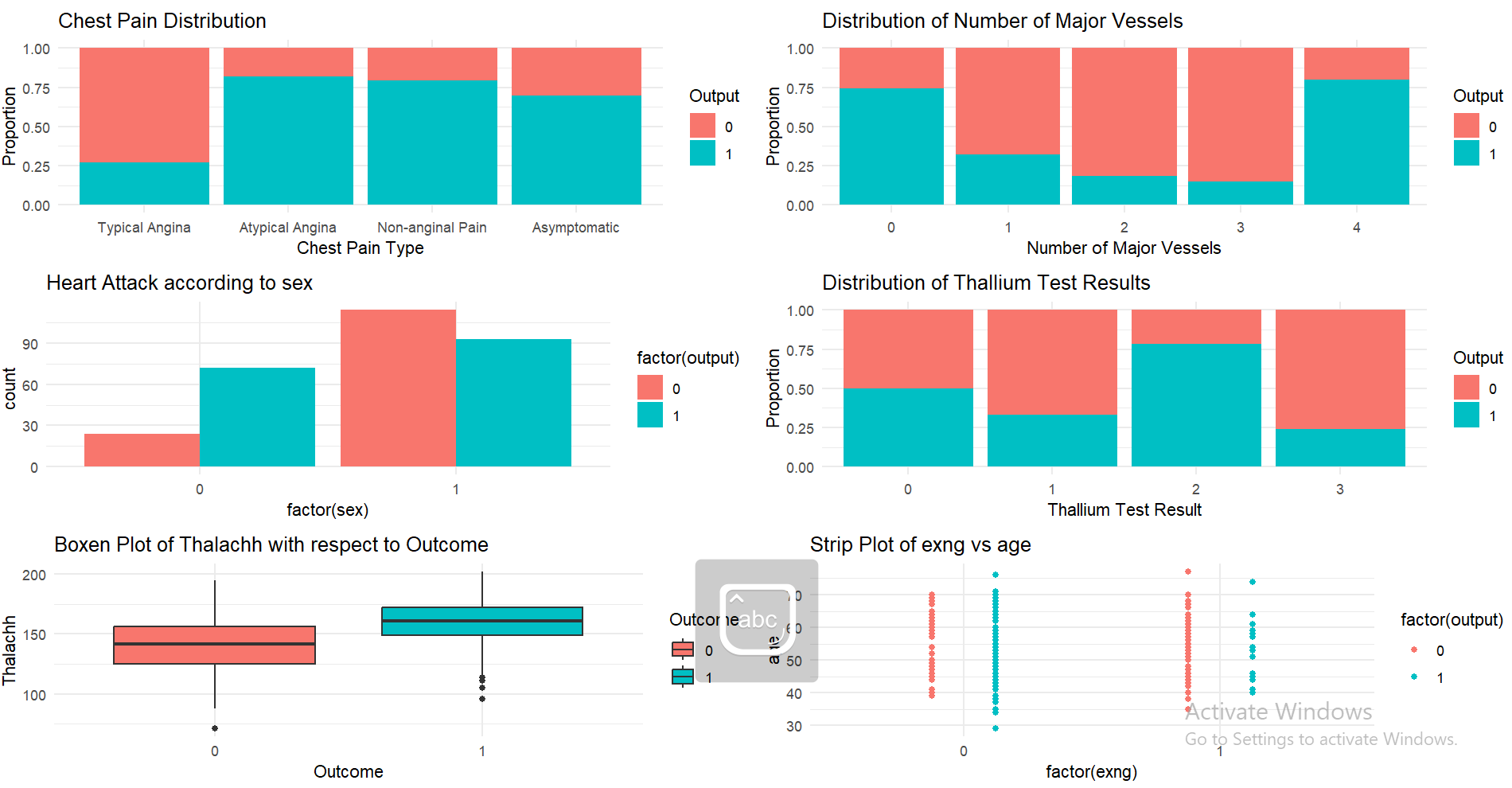
În ceea ce privește **oldpeak** (depresia segmentului ST indusă de exercițiu relativ la repaus), valorile variază de la 0.00 la 6.20. Valoarea mediană este de 0.80, iar media este de 1.04. Aceasta variabilă este esențială pentru evaluarea ischemiei miocardice, iar valorile mai ridicate pot indica probleme cardiace semnificative.

Aceste statistici oferă o bază solidă pentru înțelegerea distribuției și tendințelor în cadrul setului de date și sunt esențiale pentru dezvoltarea modelelor predictive de diagnostic al bolilor de inimă.

Table 4: Braplot-uri



* Distribuția pe sexe arată că majoritatea pacienților din setul de date sunt bărbați (codificați cu 1), iar un număr mai mic sunt femei (codificate cu 0). Această disproporție poate influența rezultatele analizei și ar trebui luată în considerare la interpretarea riscului asociat bolilor de inimă.
* Majoritatea pacienților nu prezintă angină indusă de exercițiu (codificați cu 0), în timp ce un număr semnificativ mai mic prezintă angină indusă de exercițiu (codificați cu 1). Acest lucru poate indica că angina indusă de exercițiu nu este foarte comună în acest set de date.
* Cele mai multe observații sunt pentru pacienți cu 0 vase majore colorate, urmate de pacienți cu 1, 2, și foarte puțini cu 3 vase colorate. Această variabilă este crucială pentru evaluarea severității bolii coronariene
* Tipul durerii toracice cel mai frecvent raportat este angina tipică (codificată cu 0), urmată de angină atipică (codificată cu 1), durere non-anginoasă (codificată cu 2), și asimptomatic (codificat cu 3). Această distribuție ajută la înțelegerea prevalenței diferitelor tipuri de dureri toracice în rândul pacienților.
* Majoritatea pacienților au glicemia la jeun sub 120 mg/dl (codificat cu 0), iar un număr mult mai mic au glicemia peste această valoare (codificat cu 1). Aceasta indică faptul că hiperglicemia nu este foarte frecventă în acest set de date.
* Distribuția arată că majoritatea pacienților au rezultate normale ale ECG-ului de repaus (codificat cu 0), urmate de anomalii ST-T (codificat cu 1), și foarte puțini cu hipertrofie ventriculară stângă (codificat cu 2). Aceste date sunt esențiale pentru evaluarea funcției cardiace în repaus
* Panta segmentului ST de vârf în timpul exercițiului este cel mai frecvent plată (codificat cu 1), urmată de în sus (codificat cu 2) și foarte puține cazuri în jos (codificat cu 0). Acest aspect este important pentru evaluarea ischemiei miocardice.
* Distribuția arată că majoritatea pacienților au un defect reversibil al thalassemiei (codificat cu 2), urmat de pacienți cu defect fix (codificat cu 1) și foarte puțini cu thalassemie normală (codificat cu 0). Acest lucru este relevant pentru evaluarea riscului asociat defectelor thalassemice în bolile de inimă.
* Aceste barploturi oferă o imagine clară a distribuției variabilelor calitative din setul de date și sunt utile pentru înțelegerea tendințelor și identificarea aspectelor critice în analiza riscului bolilor de inimă.



Analiza Exploratorie a Datelor (EDA)

În imagine sunt prezentate mai multe grafice rezultate în urma analizei exploratorii a datelor (EDA) pentru a înțelege relațiile dintre variabilele setului de date și riscul de atac de cord. Graficul care arată distribuția tipurilor de dureri toracice în funcție de rezultat (Output) sugerează că persoanele cu durere toracică non-anginoasă au șanse mai mari de atac de cord. De asemenea, graficul care ilustrează distribuția numărului de vase majore colorate în funcție de rezultat indică faptul că persoanele cu 0 vase majore colorate au un risc mai mare de atac de cord.

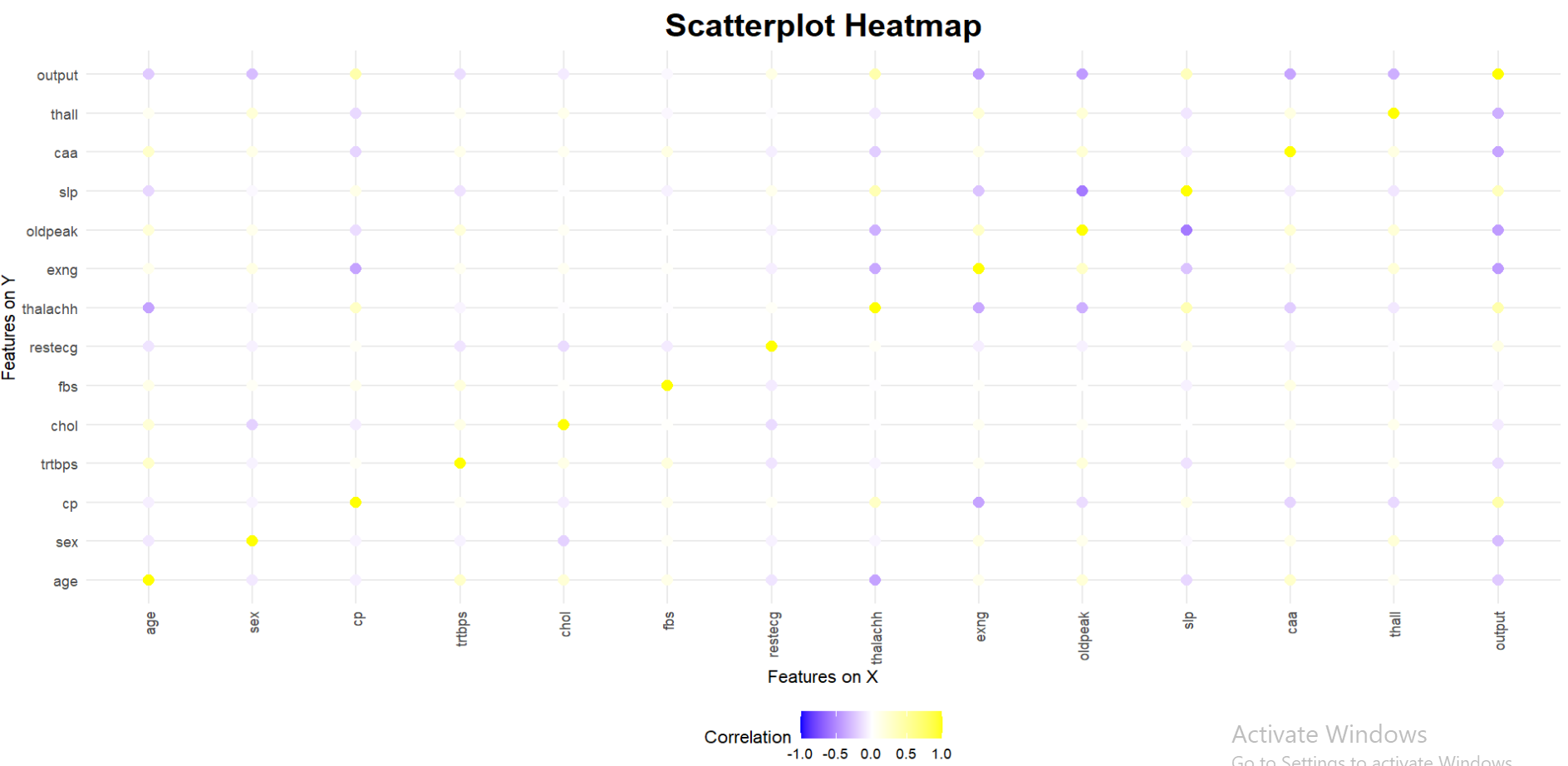
Distribuția atacurilor de cord în funcție de sex arată că persoanele de sex masculin (sex = 1) au un risc mai mare de atac de cord comparativ cu persoanele de sex feminin (sex = 0). În ceea ce privește rezultatele testului de thallium, persoanele cu un rezultat de 2 au un risc considerabil mai mare de atac de cord. Graficul boxplot arată că persoanele cu o frecvență cardiacă maximă mai mare (Thalachh) au șanse mai mari de atac de cord. Relația dintre angina indusă de exercițiu (Exng) și vârstă indică faptul că persoanele fără angină indusă de exercițiu (exng = 0) au un risc mai mare de atac de cord, indiferent de vârstă.

Concluzii din EDA

Din analiza exploratorie a datelor, am constatat că nu există valori lipsă (NaN) în setul de date. Totuși, există anomalii în toate caracteristicile continue. Setul de date conține de peste două ori mai multe persoane de sex masculin (sex = 1) decât de sex feminin (sex = 0). Nu există o corelație liniară evidentă între variabilele continue conform hărții de corelație (heatmap). Matricea scatterplot heatmap sugerează că ar putea exista o corelație între rezultat (output) și tipul de durere toracică (cp), frecvența cardiacă maximă atinsă (thalachh) și tensiunea arterială la repaus (slp).

Deși este intuitiv că persoanele în vârstă ar putea avea șanse mai mari de atac de cord, distribuția vârstei în funcție de rezultat arată că nu este neapărat cazul. Persoanele cu o frecvență cardiacă maximă atinsă mai mare (thalachh) au șanse mai mari de atac de cord. Persoanele cu un vârf anterior mai mic (oldpeak) au șanse mai mari de atac de cord. Persoanele cu durere toracică non-anginoasă (cp = 2) au șanse mai mari de atac de cord. Persoanele cu 0 vase majore (caa = 0) au un risc mai mare de atac de cord. Persoanele de sex masculin (sex = 1) au un risc mai mare de atac de cord. Persoanele cu un rezultat de 2 la testul de thallium au un risc mult mai mare de atac de cord. Persoanele fără angină indusă de exercițiu (exng = 0) au șanse mai mari de atac de cord.

Table 5: Heat-plot



1. **Corelații notabile între variabile**:
   * **age și caa**: Observăm o corelație pozitivă între vârstă și numărul de vase majore colorate, ceea ce sugerează că pacienții mai în vârstă tind să aibă mai multe vase afectate.
   * **thalachh și age**: Există o corelație negativă, ceea ce indică că pe măsură ce vârsta crește, frecvența cardiacă maximă tinde să fie mai mică.
   * **oldpeak și caa**: O corelație pozitivă sugerează că pacienții cu mai multe vase afectate tind să aibă și depresii ST mai mari în timpul exercițiului.
2. **Corelații negative puternice**:
   * **thalachh și age**: Frecvența cardiacă maximă scade odată cu vârsta, ceea ce este evidențiat de un punct albastru mai închis în grafic.
   * **exng și thalachh**: Există o corelație negativă, indicând că angina indusă de exercițiu tinde să fie asociată cu o frecvență cardiacă maximă mai mică.
3. **Relația cu variabila țintă (output)**:
   * **caa și output**: Corelație pozitivă, ceea ce sugerează că un număr mai mare de vase afectate este asociat cu o probabilitate mai mare de a avea o boală de inimă.
   * **thalachh și output**: Corelație negativă, indicând că o frecvență cardiacă maximă mai mică este asociată cu prezența bolii de inimă.

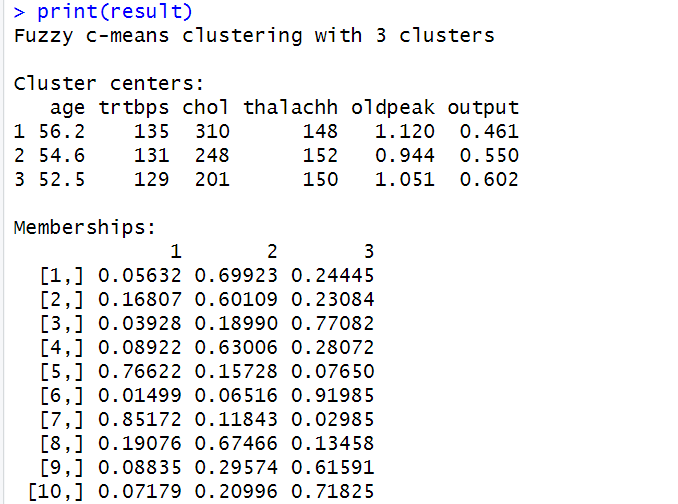
Aceste informații din heatmap sunt utile pentru identificarea relațiilor și tendințelor dintre variabile, ajutând la construirea și evaluarea modelelor predictive pentru diagnosticul bolilor de inimă. Corelațiile puternice pozitive sau negative pot sugera variabile importante care influențează starea de sănătate a inimii și care ar trebui incluse în modele de predicție.

1. UTILIZAREA METODEI “CLUSTERIZARE FUZZY”
2. Fuzzy clustering cu functia “cmeans”

Adaptam setul nostrum de date, astfel incat toate variabilele sa fie numerice, urmand sa alcatuim o custerizare cu 3 clustere, 100 iteratii si m=2 gradul de fuzzificare.

Centroizii (mediile) celor 3 clustere sunt :

Table 6: Rezultatele clusterizarii



**Interpretarea Centrelelor Clusterelor**

**Cluster 1**: Pacienți cu vârsta medie de 56.2 ani, cu valori ridicate ale colesterolului (310 mg/dl), tensiune arterială în repaus de 135 mm Hg și un nivel mai ridicat de depresie ST (1.120). Output-ul mediu (0.461) sugerează un risc intermediar de boală cardiacă.

**Cluster 2**: Pacienți cu vârsta medie de 54.6 ani, cu valori moderate ale colesterolului (248 mg/dl) și tensiune arterială în repaus de 131 mm Hg. Depresia ST este mai mică (0.944) și output-ul mediu (0.550) sugerează un risc moderat spre ridicat de boală cardiacă.

**Cluster 3**: Pacienți mai tineri, cu vârsta medie de 52.5 ani, cu valori mai scăzute ale colesterolului (201 mg/dl) și tensiune arterială în repaus de 129 mm Hg. Depresia ST (1.051) și output-ul mediu (0.602) sugerează un risc relativ ridicat de boală cardiacă.

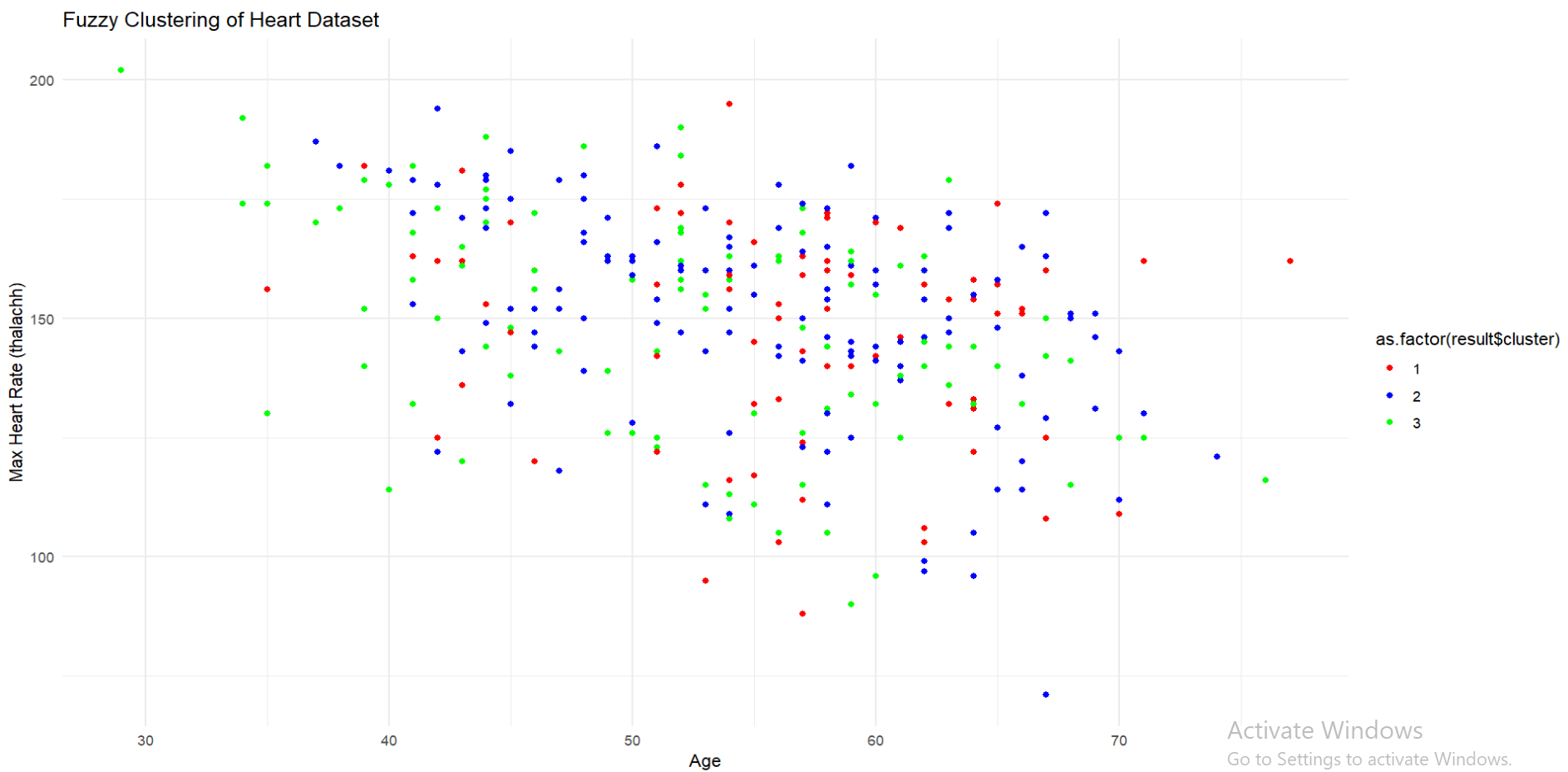
**Apartenența la Clustere**

Tabelul de "Memberships" arată gradul de apartenență al fiecărui punct de date la fiecare cluster. Fiecare rând din tabel reprezintă un punct de date, iar coloanele indică apartenența la fiecare dintre cele trei clustere.

Prima observație: 5.632% apartenență la Cluster 1, 69.923% apartenență la Cluster 2, și 24.445% apartenență la Cluster 3. Aceasta sugerează că această observație este cel mai probabil parte a Clusterului 2.

Analiza clustering-ului Fuzzy C-Means a identificat trei grupuri distincte de pacienți cu caracteristici diferite în ceea ce privește vârsta, tensiunea arterială, colesterolul, frecvența cardiacă maximă, depresia ST și riscul de boală cardiacă. Aceste clustere pot fi utilizate pentru a dezvolta strategii personalizate de diagnostic și tratament, având în vedere caracteristicile specifice fiecărui grup.

Table 7:Graficul clusterizarii in functie de varsta si recvența cardiacă maximă atinsă



1. **Distribuția Clusterelor**:

**Cluster 1 (roșu)**: Pacienții din acest cluster sunt distribuiți pe toată gama de vârstă, dar tind să aibă frecvențe cardiace maxime mai mici, de obicei sub 150 bpm.

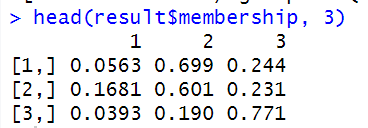
**Cluster 2 (verde)**: Acest cluster include pacienți de diferite vârste, dar se observă o concentrație mai mare în jurul vârstelor de 50-60 de ani și frecvențe cardiace maxime variate.

**Cluster 3 (albastru)**: Pacienții din acest cluster au o răspândire largă în ceea ce privește vârsta, dar tind să aibă frecvențe cardiace maxime mai mari, adesea peste 150 bpm.

Tendințe Generale: Observăm că, în general, pacienții mai tineri tind să aibă frecvențe cardiace maxime mai mari, în timp ce pacienții mai în vârstă au frecvențe cardiace maxime mai mici. Aceasta este o tendință așteptată, deoarece capacitatea cardiacă tinde să scadă odată cu vârsta.

1. **Interpretarea Clusterelor**:
   * **Cluster 1 (roșu)**: Pare să reprezinte pacienții cu un risc intermediar spre scăzut de boli de inimă, având în vedere frecvențele cardiace maxime mai mici.
   * **Cluster 2 (verde)**: Include pacienți cu riscuri moderate, cu o variabilitate mare în frecvențele cardiace maxime.
   * **Cluster 3 (albastru)**: Acest cluster ar putea reprezenta pacienți cu un risc mai mare de boli de inimă, având frecvențe cardiace maxime mai mari.

Table 8:Rezultatele apartenenței la clustere pentru primele trei observații

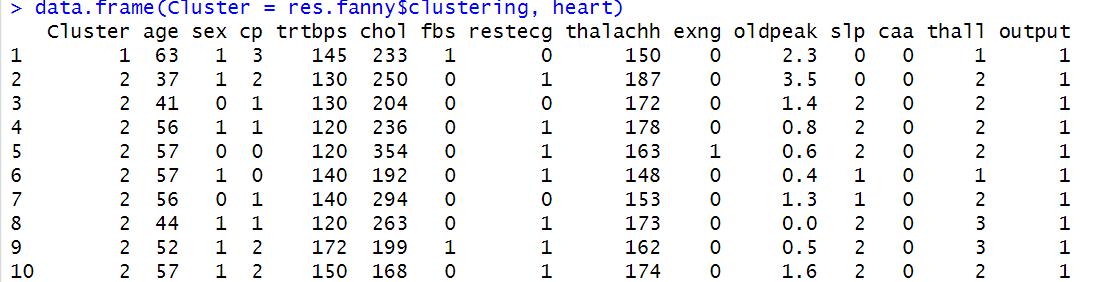


Prima observație are o apartenență majoritară la Clusterul 2 (69.9%), indicând că, dintre cele trei clustere, cel mai probabil face parte din Clusterul 2.

A doua observație are, de asemenea, o apartenență majoritară la Clusterul 2 (60.1%), ceea ce sugerează că cel mai probabil face parte din Clusterul 2, dar cu o probabilitate mai mică decât prima observație.

A treia observație are o apartenență majoritară la Clusterul 3 (77.1%), indicând că cel mai probabil face parte din Clusterul 3.

1. Fuzzy clustering cu biblioteca “cluster”



Suprapunerea Clusterelelor: Suprapunerea între cele două clustere sugerează că există caracteristici comune între observațiile din cele două grupuri, ceea ce poate fi util pentru identificarea caracteristicilor comune sau tranzitorii în setul de date.

Separarea Clusterelelor: Chiar dacă există suprapuneri, există o separare clară între cele două clustere, ceea ce indică faptul că clustering-ul fuzzy a reușit să identifice două grupuri distincte cu caracteristici diferite.

Table 9: Graficul apartenentelor la clustere dupa PCA

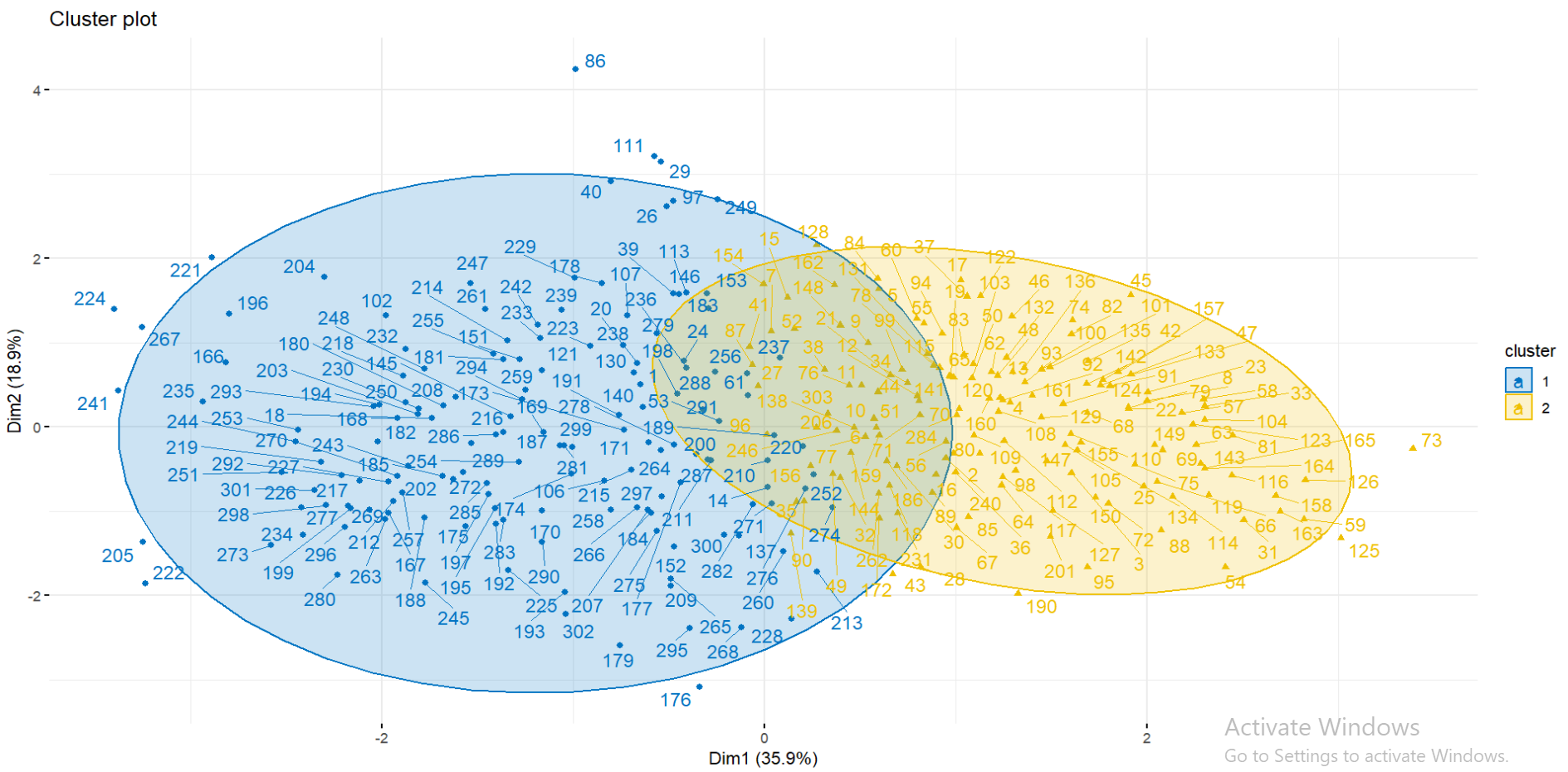


Table 10: Determinare coeficientii silueta

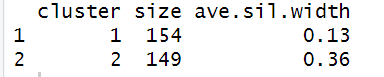
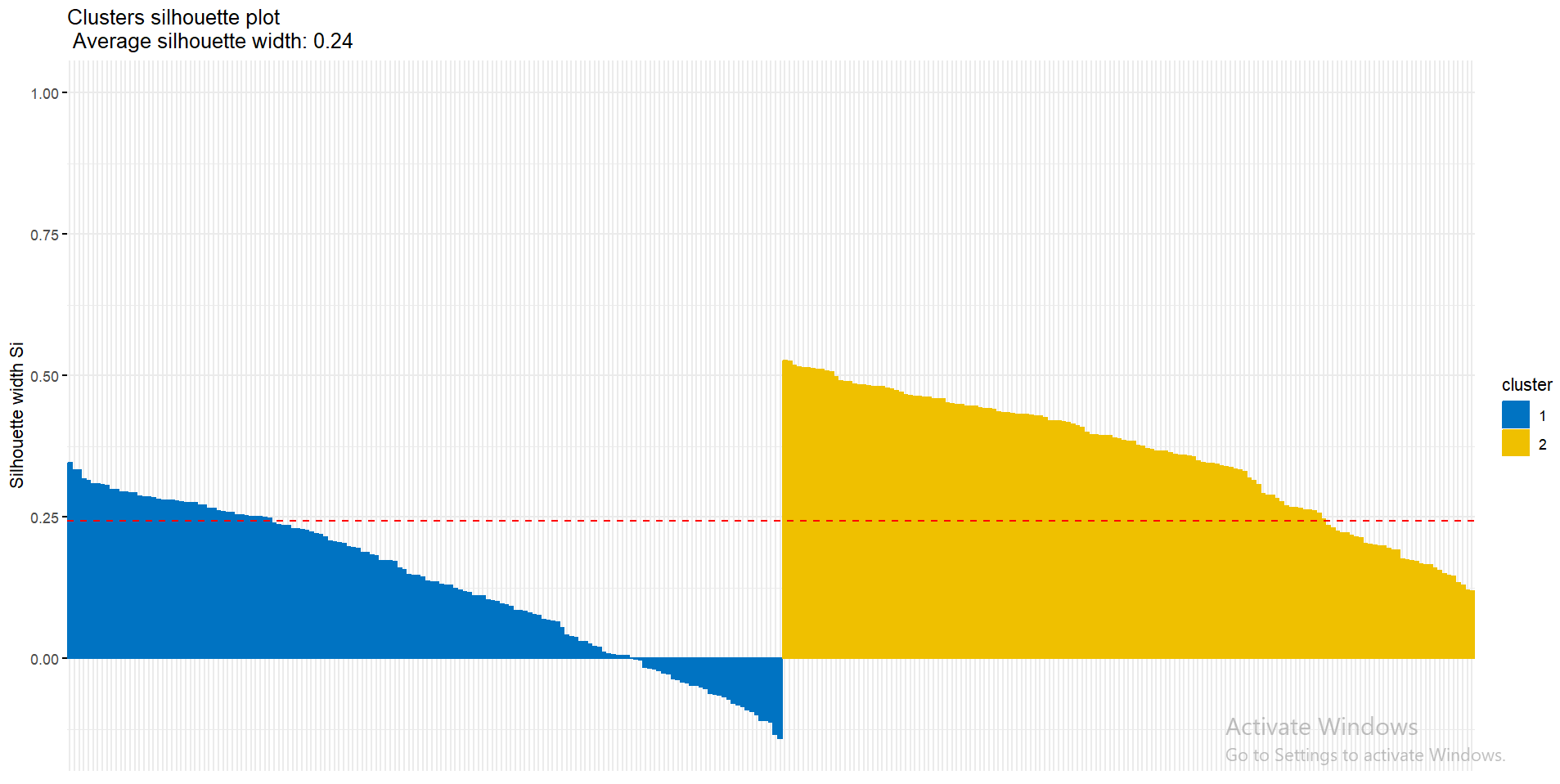


Table 11: Silhouette plot



1. **Silhouette Width (Lățimea Siluetei)**:

**Valori pozitive**: Indică faptul că observațiile sunt bine grupate în clusterele lor respective. Cu cât valoarea este mai aproape de 1, cu atât este mai bună separarea.

**Valori negative**: Sugerează că observațiile ar putea fi atribuite unui cluster greșit.

**Valori aproape de 0**: Indică că observațiile sunt pe granița dintre clustere.

Fiecare bară reprezintă o observație individuală.

**Cluster 1 (albastru)**: Observațiile din Clusterul 1 au valori de silhouette variabile, majoritatea fiind pozitive, ceea ce indică o separare relativ bună, deși există unele valori aproape de 0, sugerând o graniță neclară.

**Cluster 2 (galben)**: Observațiile din Clusterul 2 prezintă, de asemenea, valori pozitive pentru majoritatea, dar există o variație semnificativă, inclusiv unele valori negative, indicând că anumite observații ar putea fi greșit atribuite.

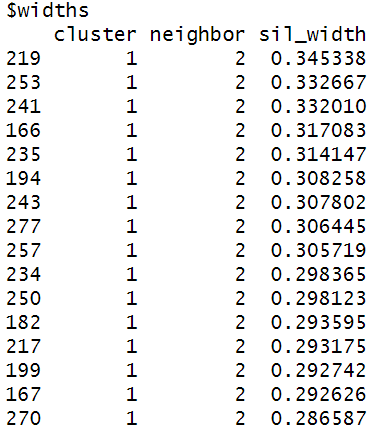
1. **Linia Punctată Roșie**:
   * Linia roșie reprezintă **lățimea medie a siluetei** (0.24).
   * O valoare medie a silhouette de 0.24 sugerează că clusterele sunt moderate spre slab separate. O valoare mai mare (aproape de 1) ar indica o separare mai clară între clustere.

**Cluster 1** (albastru) arată o calitate bună a clustering-ului pentru majoritatea observațiilor, dar există câteva puncte care sunt pe granița dintre clustere.

**Cluster 2** (galben) are o calitate mai variabilă a clustering-ului, cu unele observații bine grupate și altele care ar putea fi reatribuite.

**Lățimea medie a siluetei de 0.24** indică o separare generală slabă între clustere, sugerând că modelul de clustering ar putea beneficia de ajustări suplimentare sau de un număr diferit de clustere

Table 12: Determinam silueta individuala a fiecarui obiect



Observația 219 are o silhouette width de 0.345338, ceea ce este o valoare pozitivă moderată. Acest lucru sugerează că această observație este bine grupată în Clusterul 1, deși nu este foarte departe de Clusterul 2. Aceasta indică o bună apartenență la Clusterul 1, dar și o anumită proximitate față de Clusterul 2

1. Regresia logistică binomiala

Regresia binomială, cunoscută și sub denumirea de regresie logistică, este o metodă de analiză statistică utilizată pentru modelarea unei variabile de răspuns binare (adică variabile care au doar două valori posibile, cum ar fi "succes" și "eșec" sau "0" și "1"). Aceasta este o tehnică folosită în situațiile în care se dorește prezicerea probabilității unui anumit eveniment pe baza uneia sau mai multor variabile independente.

**Caracteristici Principale**

1. **Funcția Logit**:

Regresia binomială utilizează funcția logit pentru a modela relația dintre variabila dependentă binară și variabilele independente. Funcția logit este definită ca logaritmul raportului șanselor (log odds) ca funcție liniară a variabilelor independente.

1. **Modelul de Regressie**:
   * Formula generală a modelului de regresie logistică este:

logit(𝑝)=ln⁡(𝑝1−𝑝)=𝛽0+𝛽1𝑋1+𝛽2𝑋2+…+𝛽𝑛𝑋𝑛logit(*p*)=ln(1−*pp*​)=*β*0​+*β*1​*X*1​+*β*2​*X*2​+…+*βn*​*Xn*​

unde 𝑝*p* este probabilitatea de succes a variabilei dependente, 𝛽0*β*0​ este interceptul, iar 𝛽𝑖*βi*​ sunt coeficienții de regresie pentru variabilele independente 𝑋𝑖*Xi*​.

1. **Estimarea Coeficienților**:

Coeficienții de regresie (𝛽𝑖*βi*​) sunt estimați folosind metoda de maximum verosimilitate (maximum likelihood estimation), care găsește valorile coeficienților care maximizează probabilitatea de a observa datele date modelului.

1. **Interpretarea Coeficienților**:

Coeficienții de regresie (𝛽𝑖*βi*​) indică logaritmul raportului șanselor de schimbare a probabilității de succes pentru o unitate de schimbare în variabila independentă corespunzătoare, ținând cont de celelalte variabile din model. De exemplu, un coeficient pozitiv indică faptul că o creștere a variabilei independente este asociată cu o probabilitate mai mare de succes.

În scopul estimării riscului de atac de cord (**output**) pe baza celorlalte variabile din setul de date, folosim un model de regresie binomială . Acest model ne permite să prezicem probabilitatea unui atac de cord în funcție de caracteristicile pacienților. Iată pașii detaliați pentru pregătirea setului de date și estimarea modelului:

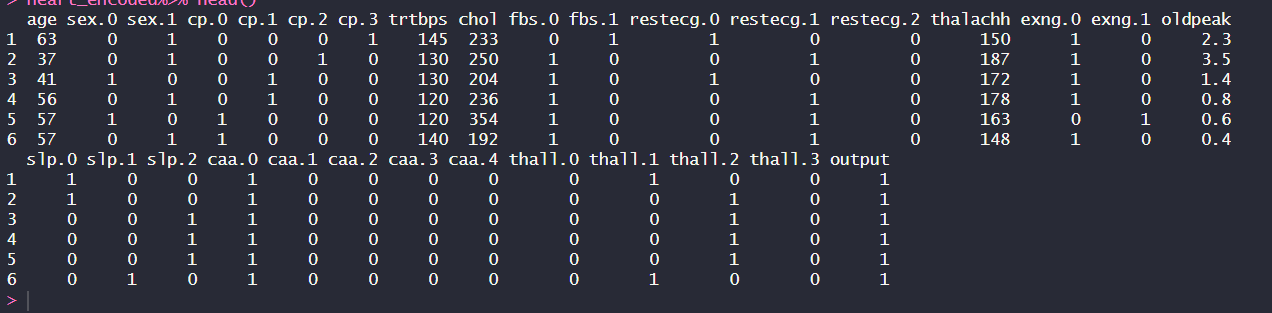
**Pregătirea Setului de Date**

**1. One Hot Encoding (Crearea Variabilelor Dummy)**

Variabilele categorice în setul de date nu sunt ierarhice (adică, valorile lor nu au un ordin natural). Pentru a le putea folosi în modelul de regresie logistică, este necesar să le codificăm astfel încât să fie utilizabile matematic.

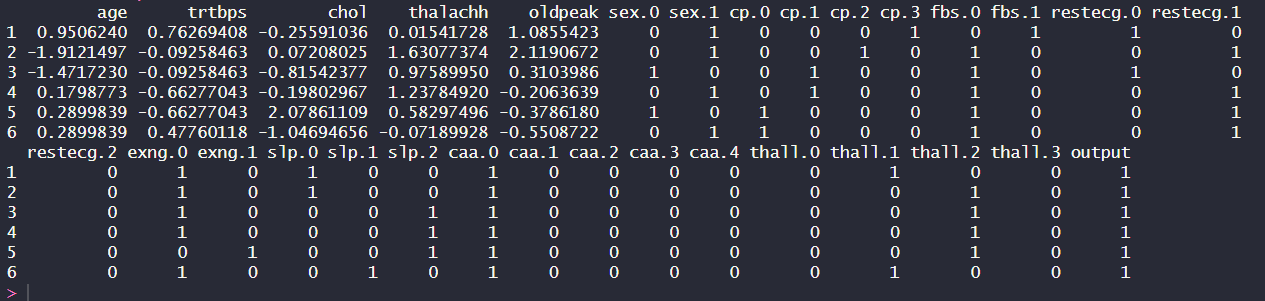
One hot encoding transformă variabilele categorice în coloane binare (dummy variables), unde fiecare coloană reprezintă o categorie distinctă a variabilei originale.

Să considerăm variabila sex, care poate avea valorile 0 (feminin) și 1 (masculin). One hot encoding va crea două coloane: sex\_0 și sex\_1. Fiecare coloană va avea valoarea 1 dacă observația aparține acelei categorii și 0 în caz contrar, astfel permite modelului să trateze fiecare categorie în mod distinct.



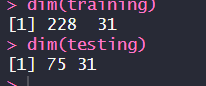
**2. Scalarea Variabilelor Numerice**

Variabilele numerice în setul de date pot avea scări și unități diferite, ceea ce poate afecta performanța modelului de regresie logistică.



**Împărțirea Datelor în Seturi de Antrenament și Test**

Pentru a evalua corect performanța modelului de regresie binomială, este esențial să împărțim setul de date în două subseturi: setul de antrenament și setul de test. Această metodă permite construirea modelului pe un subset al datelor (antrenament) și testarea performanței acestuia pe un alt subset (test) care nu a fost utilizat în timpul antrenamentului.

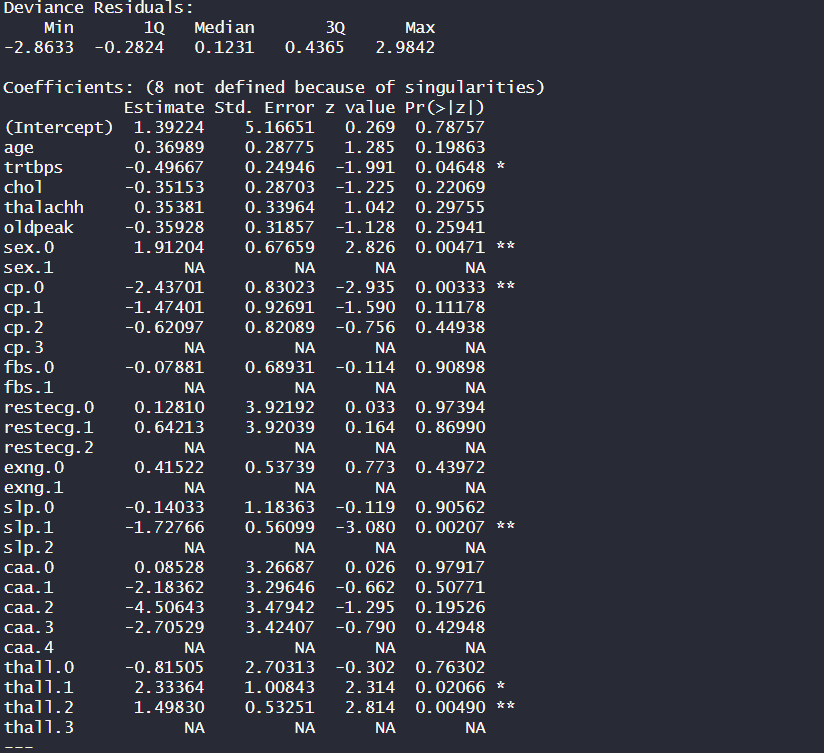


**Estimarea Modelului de Regresie Binomială**

După pregătirea setului de date, putem estima modelul de regresie binomială folosind toate variabilele independente pentru a prezice variabila dependentă output (risc de atac de cord).

**Formula Generală:**

logit(𝑝)=𝛽0+𝛽1age+𝛽2trtbps+𝛽3chol+…+𝛽𝑛thalllogit(*p*)=*β*0​+*β*1​age+*β*2​trtbps+*β*3​chol+…+*βn*​thall



**Coeficienți**

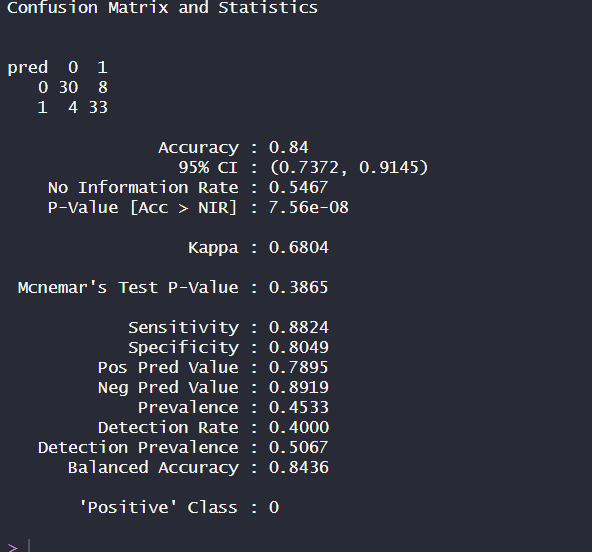
Fiecare coeficient (𝛽) reprezintă log-odds pentru variabila dependentă pentru o unitate de schimbare a variabilei independente, ținând cont de celelalte variabile din model. Semnificația coeficienților este dată de valorile Pr(>|z|) (valoarea p).

**age**: 0.36989 (p = 0.19863) – Vârsta nu este semnificativă în acest model pentru predicția riscului de atac de cord.

**trtbps**: -0.49667 (p = 0.04648) – Tensiunea arterială în repaus are un coeficient negativ semnificativ, sugerând că o creștere a tensiunii arteriale este asociată cu o scădere a riscului de atac de cord, ceea ce este neintuitiv și ar putea necesita o examinare suplimentară.

**sex** (sex.0): 1.91204 (p = 0.00471) – Sexul feminin (0) este asociat cu un risc semnificativ mai mic de atac de cord comparativ cu sexul masculin.

**cp** (cp.0): -2.43701 (p = 0.00333) – Pacienții cu angină tipică (cp = 0) au un risc semnificativ mai mic de atac de cord comparativ cu cei fără dureri toracice.



**Acuratețea (Accuracy)**: 0.84

Modelul prezice corect 84% din cazuri.

Interval de încredere de 95% (95% CI): (0.7372, 0.9145)

**No Information Rate**: 0.5467Proporția celei mai frecvente clase în setul de date de testare.

**P-Value [Acc > NIR]**: 7.56e-08

Probabilitatea ca acuratețea modelului să fie mai bună decât o predicție aleatorie bazată pe frecvența celei mai comune clase.

**Kappa**: 0.6804

Măsura acordului între predicțiile modelului și valorile reale, ajustată pentru acordul aleatoriu. O valoare de 0.6804 indică un acord bun.

**McNemar's Test P-Value**: 0.3865

Testează semnificația statistică a diferenței dintre predicțiile incorecte ale modelului.

**Sensitivitatea (Sensitivity)**: 0.8824

Rata de identificare corectă a cazurilor pozitive (atac de cord). Din cazurile reale de atac de cord, modelul identifică corect 88.24%.

**Specificitatea (Specificity)**: 0.8049

Rata de identificare corectă a cazurilor negative (fără atac de cord). Din cazurile reale fără atac de cord, modelul identifică corect 80.49%.

**Valorile Predictive**:

Pozitivă (Pos Pred Value): 0.7895 – Proporția predicțiilor pozitive corecte.

Negativă (Neg Pred Value): 0.8919 – Proporția predicțiilor negative corecte.

**Prevalența**: 0.4533

Proporția cazurilor de atac de cord în setul de date de testare.

**Detection Rate**: 0.4000

Proporția cazurilor corecte de atac de cord identificate din totalul cazurilor.

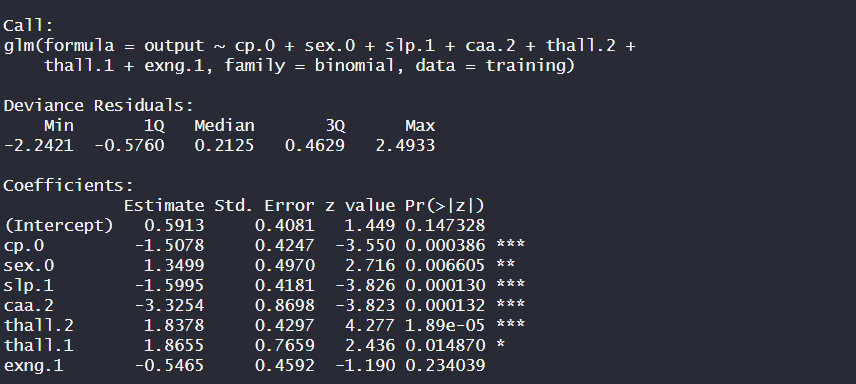
**Detection Prevalence**: 0.5067

Proporția cazurilor prezise ca fiind atac de cord din totalul cazurilor.

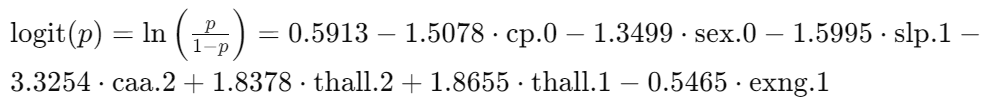
**Balanced Accuracy**: 0.8436

Media aritmetică a sensibilității și specificității, oferind o măsură echilibrată a performanței modelului.

Având în vedere că multe variabile din modelul anterior nu au fost semnificative, vom simplifica modelul bazându-ne pe variabilele care au prezentat corelații cu variabila output în heatmap. Aceste variabile sunt: cp.0, sex.0, slp.1, caa.2, thall.2, thall.1, și exng.1.



Ecuația de regresie logistică binomială:



unde:

p = probabilitatea ca evenimentul să se întâmple (output = 1)

1−p = probabilitatea ca evenimentul să nu se întâmple (output = 0)

Interpretarea coeficienților:

cp.0: Un coeficient negativ de -1.5078 indică faptul că o creștere a valorii pentru cp.0 scade logit-ul și, implicit, probabilitatea ca evenimentul să se întâmple (output = 1).

sex.0: Un coeficient negativ de -1.3499 sugerează că sex.0 are un efect negativ asupra probabilității ca evenimentul să se întâmple (output = 1).

slp.1: Coeficientul de -1.5995 arată că slp.1 reduce probabilitatea ca evenimentul să se întâmple.

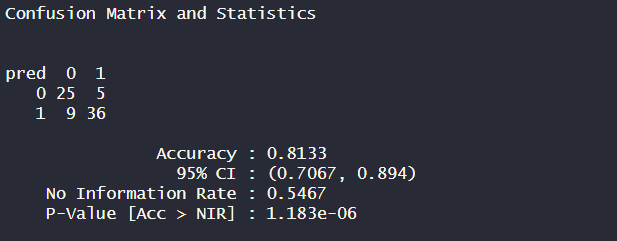
caa.2: Coeficientul de -3.3254 indică un efect negativ semnificativ asupra probabilității ca evenimentul să se întâmple.

thall.2: Coeficientul pozitiv de 1.8378 sugerează că o creștere a valorii pentru thall.2 crește probabilitatea ca evenimentul să se întâmple.

thall.1: Coeficientul pozitiv de 1.8655 arată că thall.1 crește probabilitatea ca evenimentul să se întâmple.

exng.1: Coeficientul de -0.5465 indică faptul că exng.1 scade probabilitatea ca evenimentul să se întâmple, deși acest coeficient nu este semnificativ statistic la un nivel de 0.05.

Fiecare dintre acești coeficienți reprezintă logaritmul natural al raportului șanselor (log odds) pentru fiecare unitate de schimbare în predictorul corespunzător, menținând toți ceilalți predictori constanți.



Matricea de confuzie arată performanța modelului de clasificare:

True Negatives (TN): 25 (predicție corectă de 0)

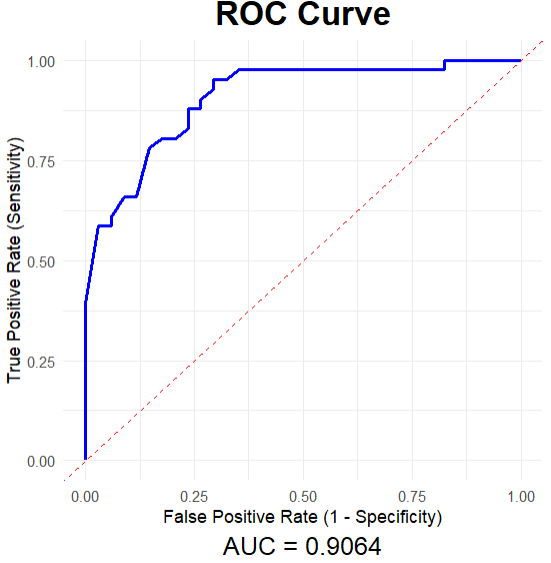
False Positives (FP): 5 (predicție incorectă de 1)

False Negatives (FN): 9 (predicție incorectă de 0)

True Positives (TP): 36 (predicție corectă de 1)

Acuratețea modelului este de 81.33%, cu un interval de încredere de 95% între 70.67% și 89.4%. Rata de informație (NIR) este 54.67%, iar valoarea p indică că acuratețea modelului este semnificativ mai bună decât NIR.

Graficul ROC este un instrument de evaluare a performanței unui model de clasificare binară. Curba ROC plasează rata pozitivă adevărată (True Positive Rate - Sensitivity) pe axa verticală și rata falsă pozitivă (False Positive Rate - 1 - Specificity) pe axa orizontală.



**Axa Verticală (True Positive Rate - Sensitivity)**:

Reprezintă proporția de cazuri pozitive corect identificate de model (TPR = TP / (TP + FN)).

Valori mai mari indică o capacitate mai bună a modelului de a detecta cazurile pozitive.

**Axa Orizontală (False Positive Rate - 1 - Specificity)**:

Reprezintă proporția de cazuri negative incorect identificate ca fiind pozitive (FPR = FP / (FP + TN)).

Valori mai mici indică o capacitate mai bună a modelului de a evita alarmele false.

**Linia Roșie Punctată**:

Reprezintă o clasă de referință, unde clasificarea este aleatorie (AUC = 0.5).

**Curba ROC**:

Curba albastră arată performanța modelului. Cu cât această curbă se apropie mai mult de colțul din stânga sus, cu atât modelul este mai performant.**AUC (Area Under the Curve)**:

* + AUC = 0.9064 pentru acest model.
  + Valori AUC mai apropiate de 1 indică un model mai performant. O valoare de 0.9064 sugerează că modelul are o performanță foarte bună în distingerea între clasele pozitive și negative.

Curba ROC și valoarea AUC oferă o evaluare comprehensivă a performanței modelului simplificat. Cu un AUC de 0.9064, modelul demonstrat este foarte eficient în clasificarea corectă a riscului de atac de cord, oferind o bază solidă pentru utilizarea practică în medii clinice. Performanța ridicată a modelului simplificat, combinată cu interpretabilitatea sa, îl face o opțiune valoroasă pentru predicția și gestionarea riscului de atac de cord.

1. ARBORI DE CLASIFICARE

Un arbore de clasificare este un model de învățare automată utilizat pentru clasificarea datelor. Acesta împarte datele în subseturi pe baza valorilor variabilelor explicative, construind un model de decizie sub formă de arbore. În acest context, vom utiliza tehnica de "pruning" pentru a îmbunătăți performanța modelului.

**Pruning în Arborii de Decizie**

Pruning este procesul de reducere a dimensiunii arborelui de decizie prin tăierea unor ramuri care nu aduc o îmbunătățire semnificativă a performanței modelului. Scopul principal este de a preveni overfitting-ul, adică adaptarea prea strânsă la datele de antrenament, ceea ce reduce capacitatea modelului de a generaliza pe date noi.

Parametrul cp

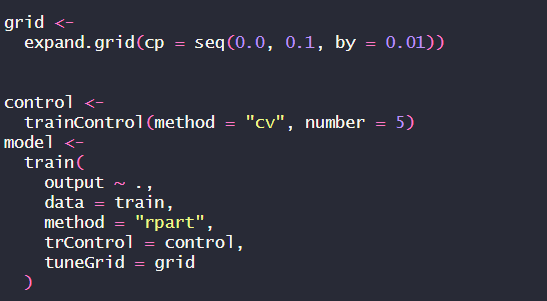
* cp (complexity parameter) este un parametru de regularizare utilizat în pruning-ul arborilor de decizie. Acesta controlează compromisurile între complexitatea modelului și acuratețea acestuia pe datele de antrenament.
* Valorile cp mai mici permit arbori mai complecși, în timp ce valorile cp mai mari favorizează arbori mai simpli.
* Prin optimizarea valorii cp, putem găsi cel mai bun echilibru între complexitate și performanță.

**Cross-Validation cu 5 Fold-uri**

Cross-validation (validarea încrucișată) este o tehnică de evaluare a performanței modelului prin împărțirea datelor în mai multe subseturi (fold-uri) și antrenarea și testarea modelului pe combinații diferite ale acestor subseturi. În 5-fold cross-validation, datele sunt împărțite în 5 fold-uri, iar modelul este antrenat pe 4 fold-uri și testat pe cel de-al 5-lea, repetând procesul de 5 ori pentru a obține o estimare robustă a performanței modelului.

**Estimarea Arborelui Pruned cu Grid Search**

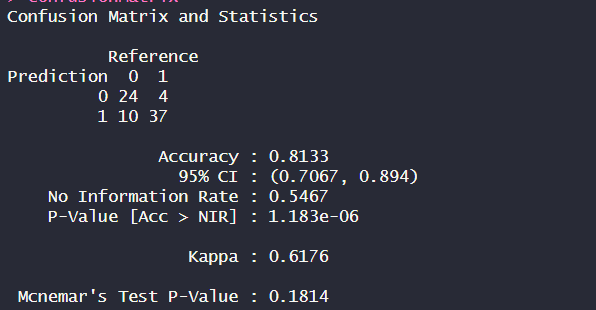
Vom utiliza un grid search pentru a testa valori ale parametrului cp între 0.01 și 0.1, și vom selecta cea mai bună valoare prin cross-validation cu 5 fold-uri.



**Estimarea parametrului cp**

După utilizarea grid search pentru a optimiza parametrul cp, valoarea optimă obținută este 0.03. Modelul de arbore de decizie antrenat cu acest parametru a produs următoarea matrice de confuzie pe setul de date de testare:

**Interpretarea Performanței Modelului de Arbore de Decizie**

****

Modelul de arbore de decizie a fost antrenat cu parametrul optim cp = 0.03, obținut prin grid search. Rezultatele pe setul de date de testare sunt prezentate în matricea de confuzie și statisticile asociate:

**Acuratețea**:

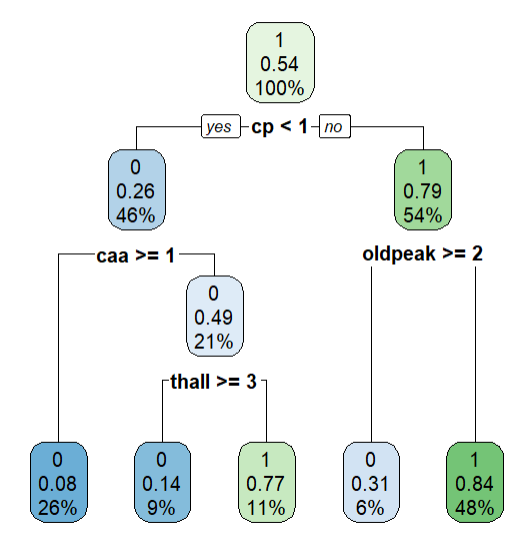
O acuratețe de 81.33% indică o performanță bună a modelului, similară cu cea obținută anterior cu modelul simplificat.

1. **Sensibilitatea și Specificitatea**:

Sensibilitatea (proporția de cazuri pozitive corect identificate) și specificitatea (proporția de cazuri negative corect identificate) sunt esențiale pentru evaluarea performanței modelului. Deși valorile exacte nu sunt date aici, matricea de confuzie indică un echilibru rezonabil între TP și TN.

1. **Erori de Clasificare**:

Modelul are mai multe erori de tip False Positives (10), ceea ce înseamnă că există cazuri în care modelul a prezis incorect un atac de cord. Totuși, numărul de False Negatives (4) este mai mic, ceea ce este crucial pentru un model de diagnosticare medicală deoarece este important să nu rateze cazurile de atac de cord.



**Interpretarea Deciziilor**

1. **cp < 1**: Dacă tipul de durere toracică este mai mic de 1, arborele continuă să investigheze **caa**.
2. **caa >= 1**: Dacă numărul de vase majore este cel puțin 1, se verifică **thall**.
3. **thall >= 3**: Dacă **thall** (thalassemia) este cel puțin 3, se prezice 1 (atac de cord) cu o probabilitate de 77%.
4. **cp >= 1**: Dacă tipul de durere toracică este cel puțin 1, se verifică **oldpeak**.
5. **oldpeak >= 2**: Dacă **oldpeak** (depresia segmentului ST) este cel puțin 2, se prezice 1 (atac de cord) cu o probabilitate de 84%.

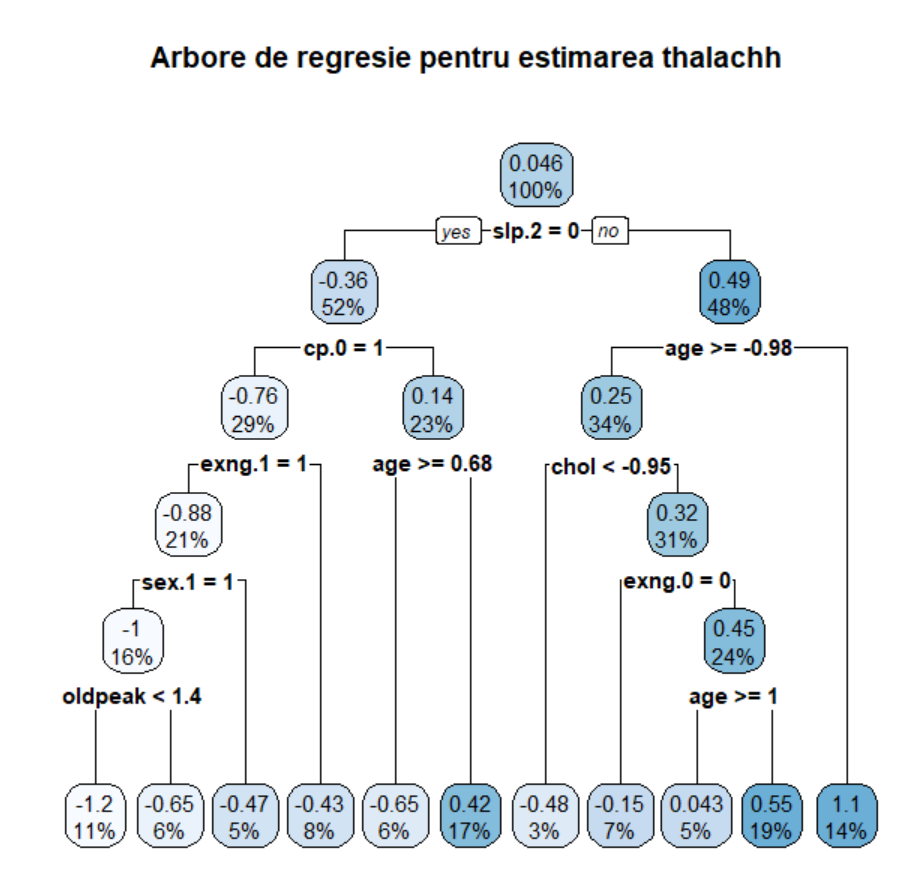
Importanța Variabilelor: cp, caa, thall, și oldpeak sunt variabilele cheie folosite în deciziile modelului.

Decizii de Predicție: Modelul simplificat oferă o structură clară de decizie bazată pe aceste variabile importante.

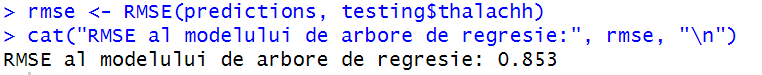
1. ARBORE DE REGRESIE

Construire și optimizare a unui arbore de regresie pentru a estima valoarea variabilei thalachh utilizând un set de date legat de sănătatea inimii. Primul pas a fost împărțirea setului de date heart\_processed în două subseturi: unul pentru antrenament (75%) și unul pentru testare (25%) utilizând funcția sample.split. Această împărțire asigură că modelul va fi antrenat pe un subset reprezentativ și testat pe un subset diferit pentru a evalua performanța sa.

În continuare, a fost construit un arbore de regresie inițial folosind funcția rpart, specificând metoda anova pentru a gestiona regresia

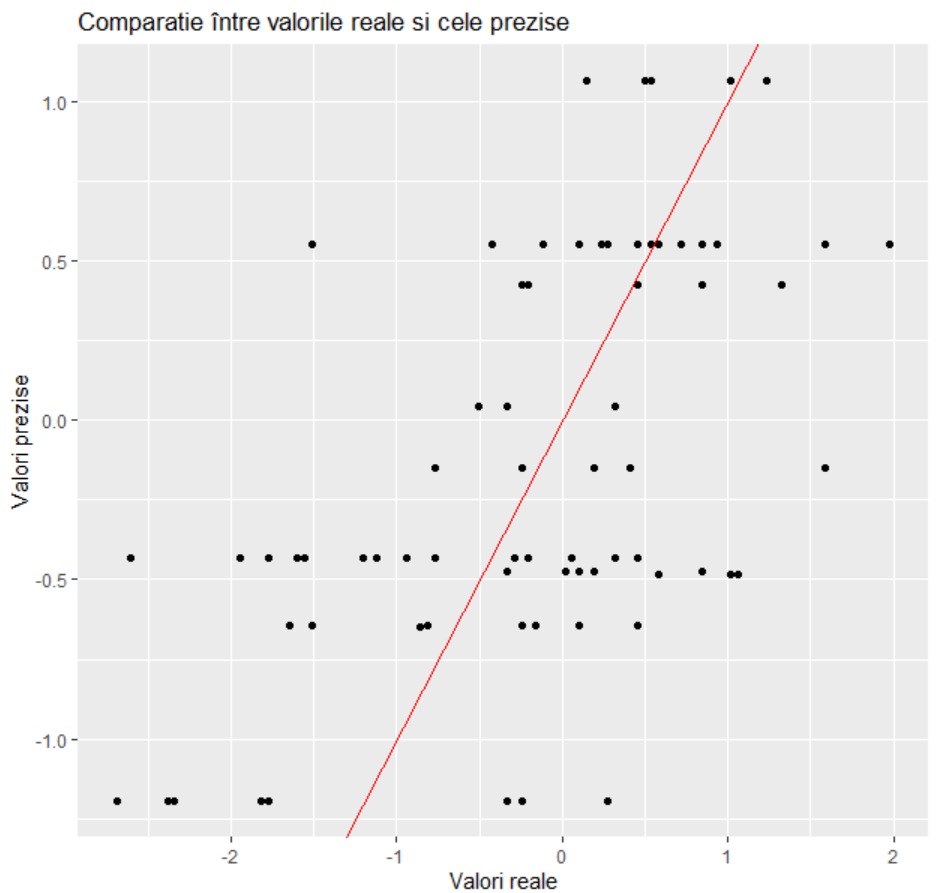


Modelul a fost apoi utilizat pentru a face predicții pe setul de testare, iar performanța acestuia a fost evaluată calculând eroarea medie pătratică rădăcină (RMSE) folosind funcția RMSE. O diagramă de comparație între valorile reale și cele prezise a fost generată utilizând ggplot2 pentru a vizualiza acuratețea predicțiilor modelului inițial.

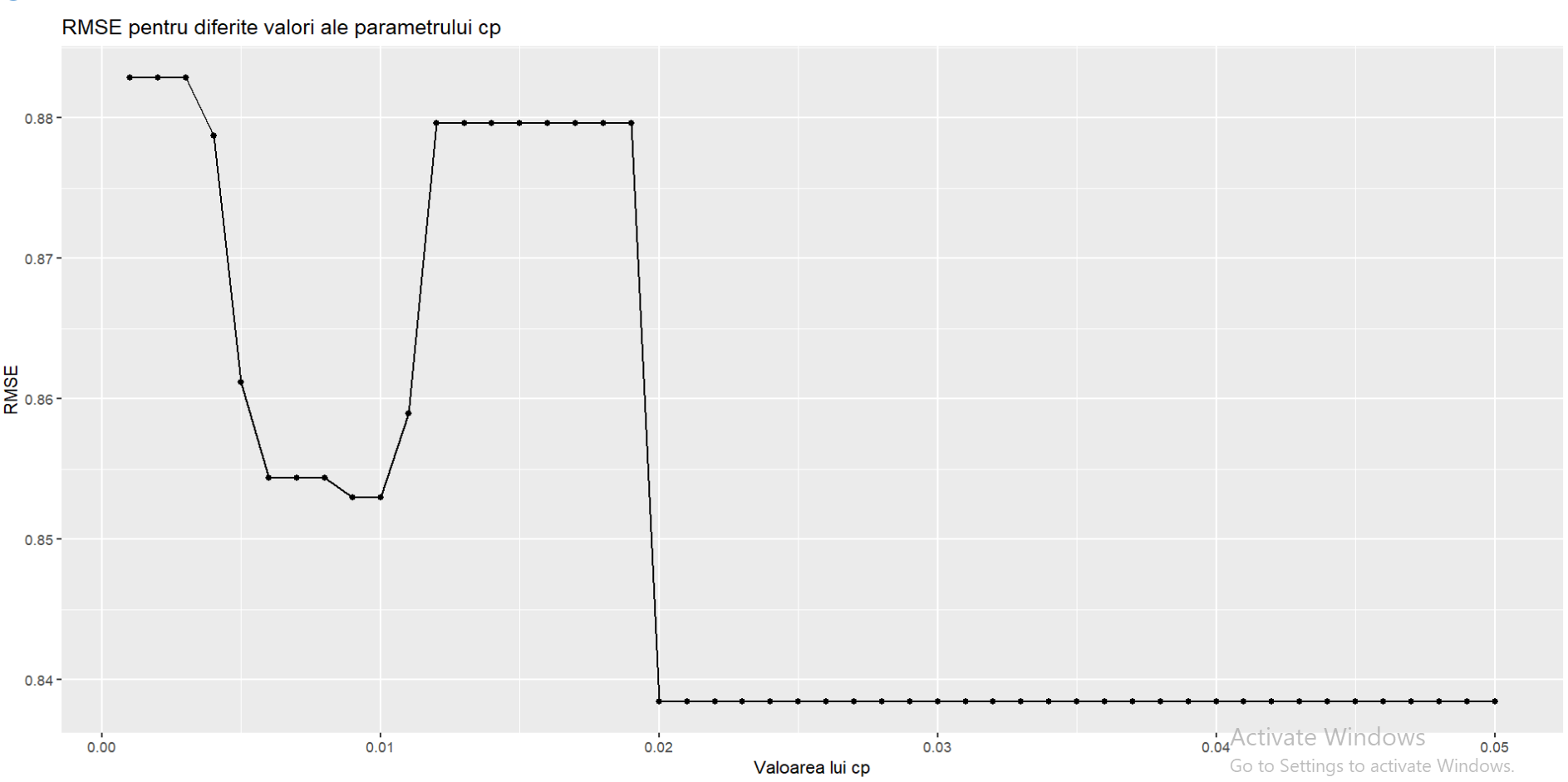


Observațiile din grafic arată că punctele sunt destul de bine aliniate de-a lungul liniei de identitate, ceea ce sugerează o corelație bună între valorile reale și cele prezise. Totuși, există și puncte care deviază semnificativ de la linia roșie, indicând erori de predicție mai mari pentru anumite observații. Această variație poate fi datorată zgomotului în date sau limitărilor modelului de regresie utilizat.

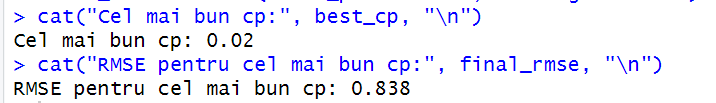
În concluzie, valoarea RMSE și graficul de comparație indică faptul că modelul de regresie construit este relativ precis în predicția valorilor thalachh, deși există anumite erori de predicție care ar putea fi îmbunătățite prin optimizarea modelului sau utilizarea unor metode alternative de modelare.

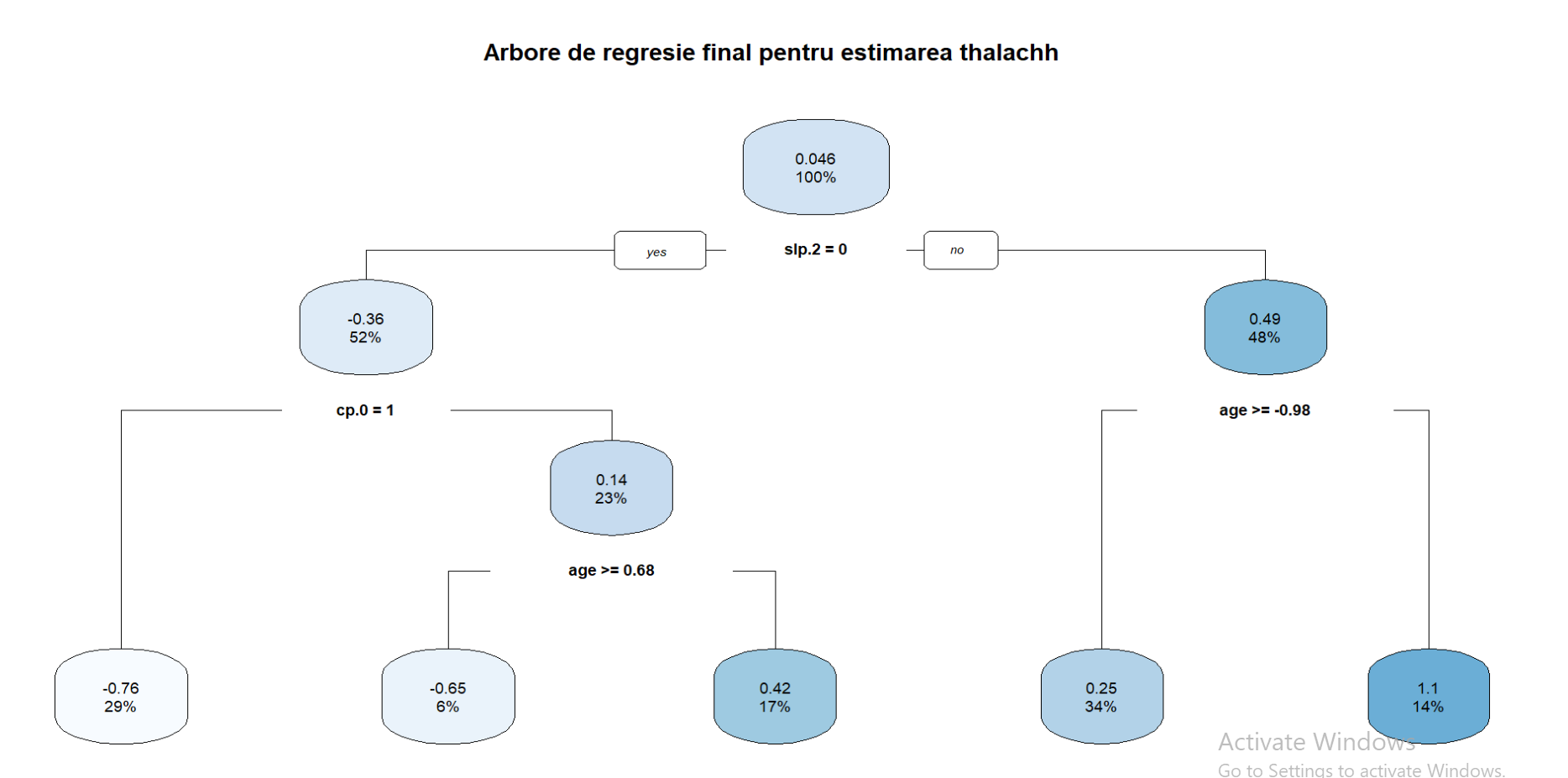
****

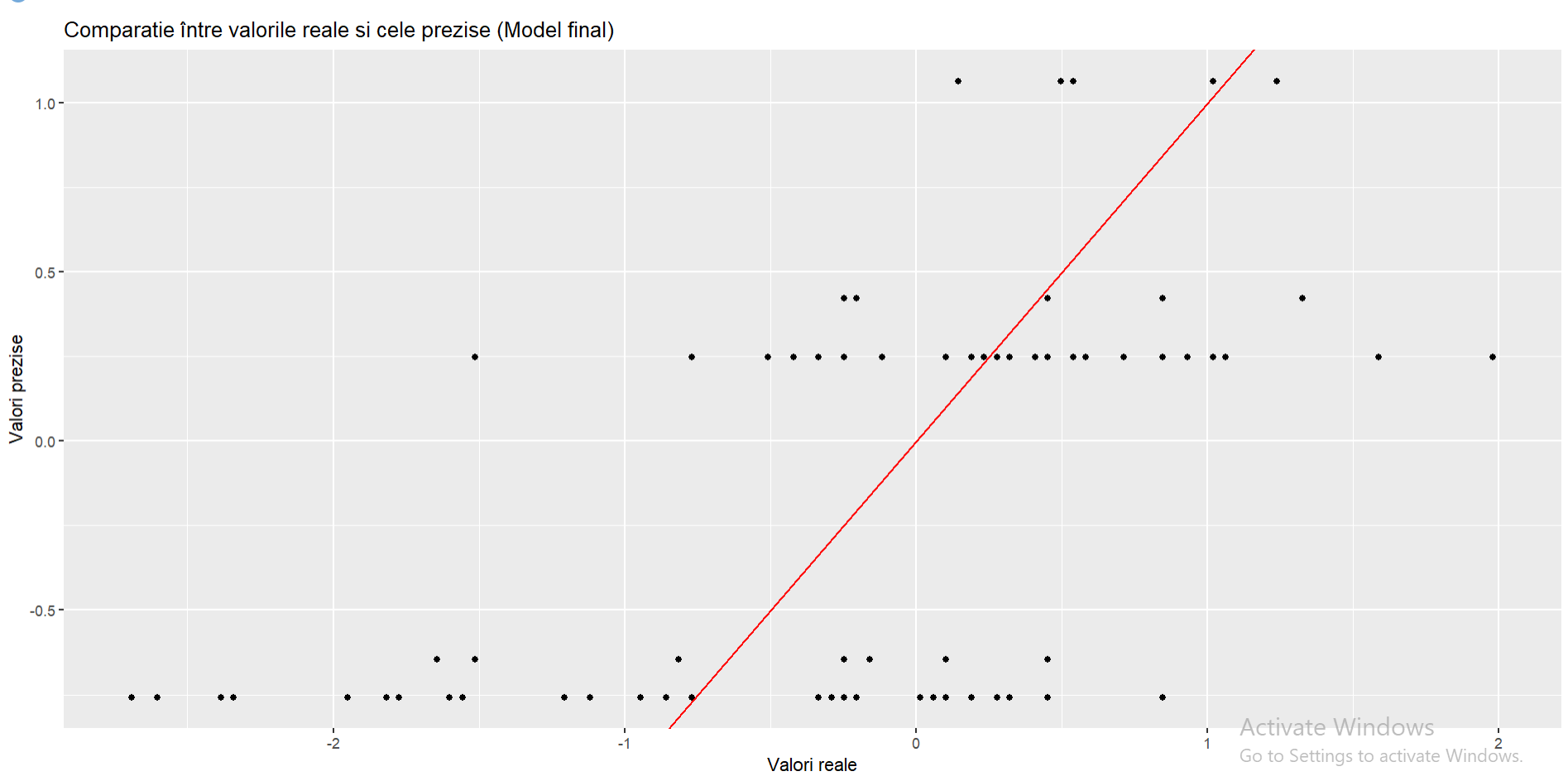
Pentru a optimiza performanța modelului, s-a realizat o căutare în grid pentru a determina valoarea optimă a parametrului de complexitate (cp). S-au testat diferite valori ale lui cp și s-a calculat RMSE pentru fiecare valoare utilizând un ciclu for. Rezultatele au fost apoi reprezentate grafic pentru a identifica vizual cea mai bună valoare a lui cp.

****

Valoarea optimă a lui cp a fost selectată ca fiind cea care minimizează RMSE, și a fost construit un model final utilizând această valoare optimă. Modelul final a fost vizualizat și utilizat pentru a face predicții pe setul de testare. Performanța modelului final a fost evaluată calculând RMSE pentru predicțiile finale și generând o diagramă de comparație între valorile reale și cele prezise pentru a vizualiza îmbunătățirile aduse de optimizare.





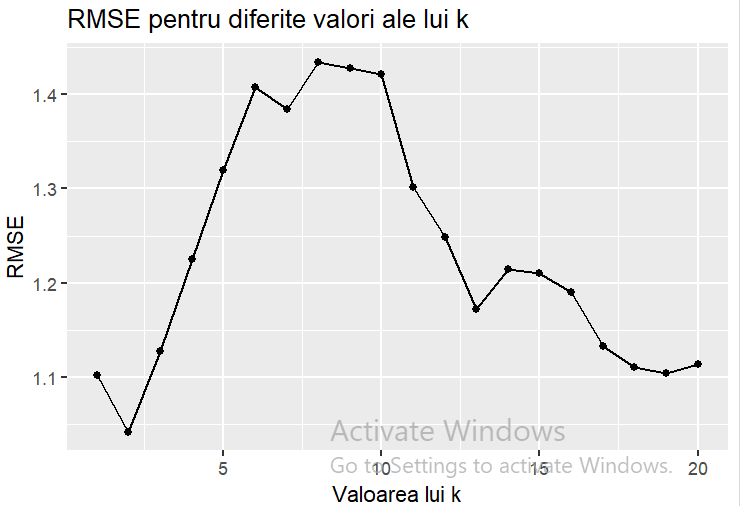


În concluzie, procesul descris în cod demonstrează cum un arbore de regresie poate fi construit și optimizat pentru a îmbunătăți acuratețea predicțiilor, folosind tehnici de validare încrucișată și căutare în grid pentru a găsi parametrii optimi. RMSE pentru modelul final a fost calculat la 0.838, indicând o performanță bună a modelului în estimarea variabilei thalachh.

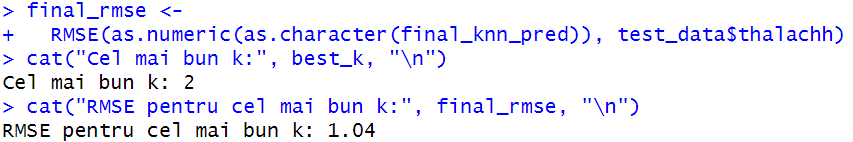
1. Folosirea KNN pentru Regresie și Optimizarea Parametrului k

Am implementat metoda KNN (K-Nearest Neighbors) pentru a estima variabila "thalachh" pe baza unui set de date de antrenament și testare. Inițial, setul de date a fost împărțit în două subseturi: un set de antrenament (80% din date) și un set de testare (20% din date), folosind funcția createDataPartition.

Pentru a determina valoarea optimă a lui k, am definit o secvență de valori pentru k, variind de la 1 la 20. Pentru fiecare valoare a lui k, am antrenat un model KNN și am calculat eroarea RMSE (Root Mean Square Error) pentru predicțiile realizate pe setul de testare. RMSE este utilizată pentru a evalua acuratețea modelului, valori mai mici ale RMSE indicând o performanță mai bună.

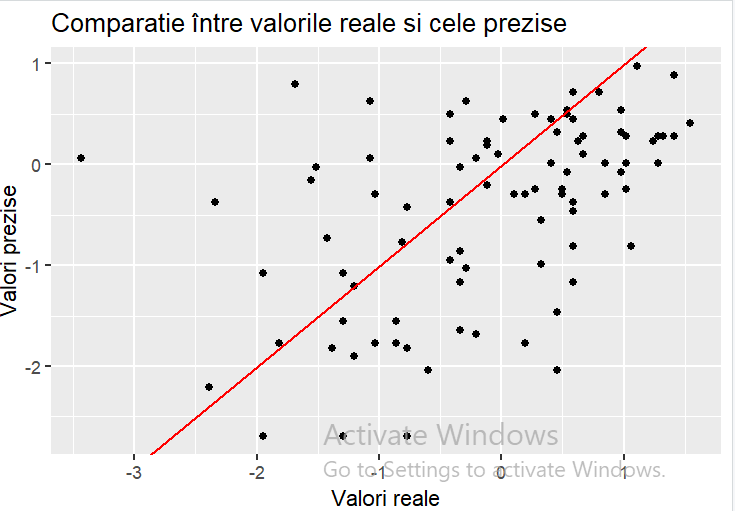


Rezultatele obținute au fost reprezentate grafic, astfel încât să putem vizualiza variația RMSE în funcție de numărul de vecini k. Din graficul "RMSE pentru diferite valori ale lui k", am observat că valoarea RMSE variază considerabil pe măsură ce k crește. Cel mai mic RMSE a fost obținut pentru k = 2, indicând că aceasta este valoarea optimă a lui k pentru modelul nostru.



După identificarea valorii optime a lui k, am antrenat din nou modelul KNN folosind această valoare și am evaluat performanța acestuia pe setul de testare. Eroarea RMSE pentru modelul final a fost de 1.04, ceea ce indică o performanță relativ bună a modelului nostru KNN.

În plus, am realizat o comparație între valorile reale și cele prezise pentru variabila "thalachh". În graficul de mai jos, punctele de pe grafic arată predicțiile modelului în comparație cu valorile reale. Linia roșie, care are panta de 1 și interceptul de 0, reprezintă o predicție perfectă. Observăm că majoritatea punctelor nu sunt aproape de această linie, indicând o performanță nu foarte buna a modelului KNN în estimarea valorilor reale ale variabilei "thalachh".

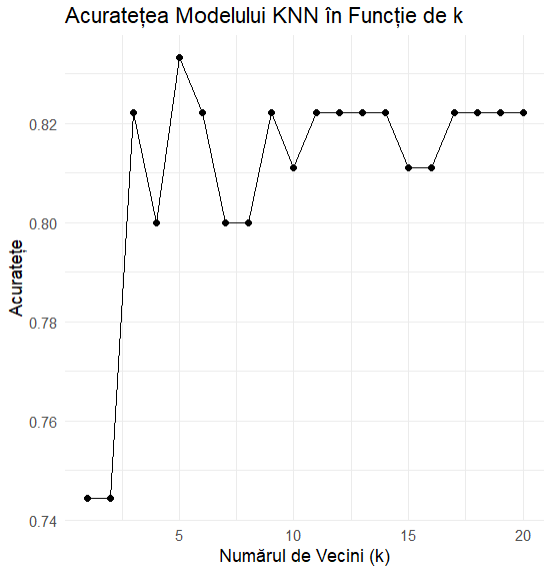


In aceasta situatie nici unul dintre algoritmii KNN sau arbore de regresie nu performeaza bine in prezicerea variabilei thalachh, insa RMSE este mai mica pentru arborele de regresie.

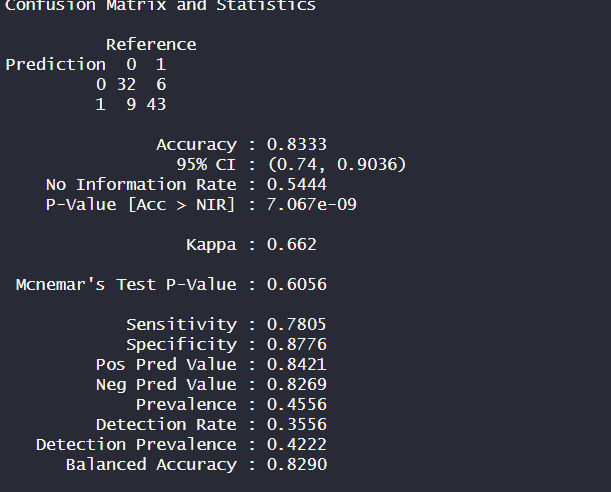
1. Modelul K-Nearest Neighbors pentru clasificare

Procesul Grid Search: Pentru a găsi cel mai bun parametru k, se testează diferite valori ale acestuia și se evaluează performanța modelului pentru fiecare valoare. În acest caz, s-a estimat cel mai bun k dupa accuracy-urile predictiilor pe setul de date.

La început, pentru k = 1, acuratețea modelului este cea mai mică, în jurul valorii de 0.74. Pe măsură ce creștem numărul de vecini, acuratețea crește rapid, atingând un maxim de aproximativ 0.82 pentru k = 5. Acest maxim indică faptul că, pentru k = 5, modelul are o performanță foarte bună în clasificare.

****

Matricea de confuzie obținută pentru modelul KNN pe setul de date de testare este următoarea:

****

Acuratețea (Accuracy): 0.8333

Modelul prezice corect 83.33% din cazuri.

**Performanță Generală:**

Modelul KNN cu k = 5 oferă o acuratețe de 83.33%, indicând o performanță foarte bună în clasificarea riscului de atac de cord.

**Sensibilitate și Specificitate:**

Sensibilitatea ridicată (78.05%) indică faptul că modelul detectează majoritatea cazurilor de atac de cord.

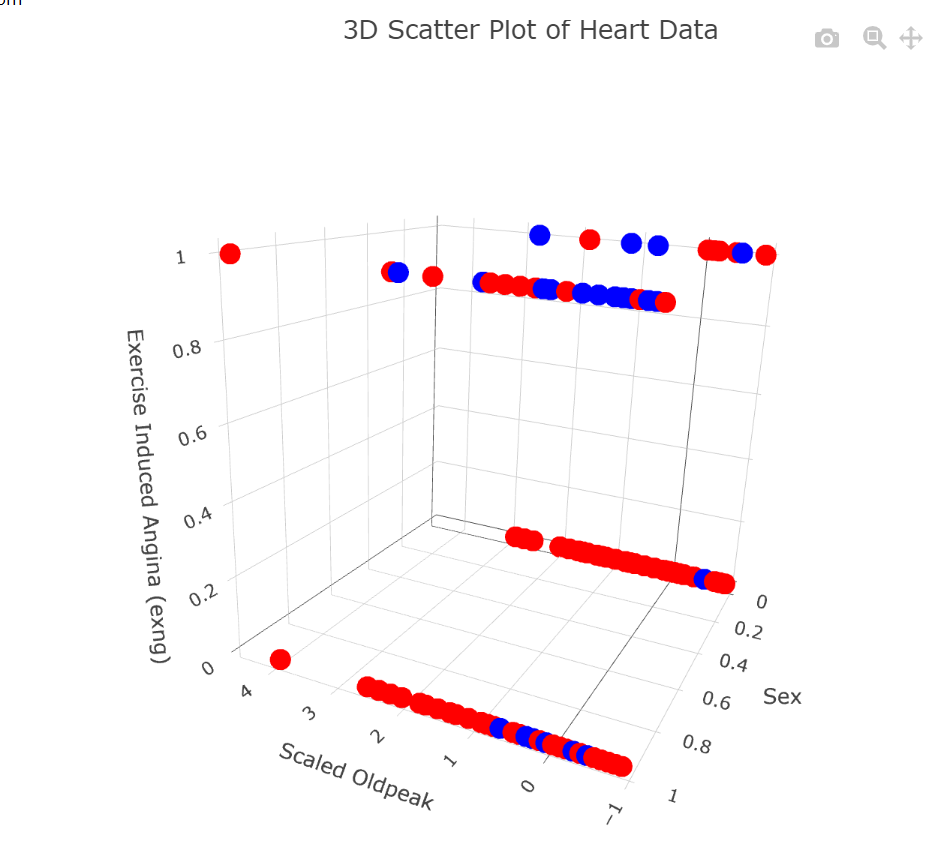
Specificitatea ridicată (87.76%) sugerează că modelul identifică corect majoritatea cazurilor fără atac de cord.

**Valori Predictive:**

Valorile predictive pozitive și negative ridicate indică faptul că modelul este de încredere în predicțiile sale.

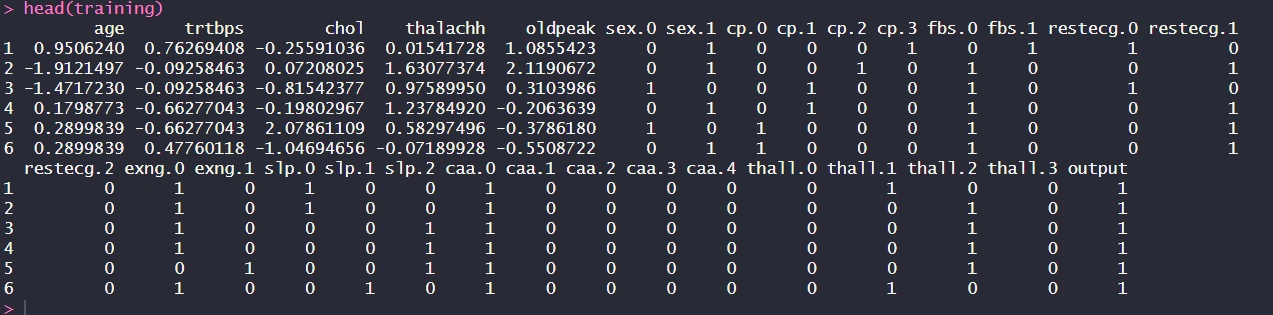
Graficul 3D prezentat oferă o reprezentare clară a relației dintre trei variabile esențiale din setul de date heart, și anume oldpeak (pe axa X, valorile scalate), sex (pe axa Y) și exng (pe axa Z), fiecare punct fiind colorat în funcție de variabila țintă output, care indică prezența unei boli de inimă (roșu pentru output = 1, adică cu boală, și albastru pentru output = 0, adică fără boală). Această vizualizare permite o analiză preliminară a distribuției pacienților, observându-se densități diferite ale punctelor în funcție de sex și de existența anginei provocate de efort.

Distribuția valorilor scalate ale oldpeak arată o variabilitate semnificativă, ceea ce sugerează că această variabilă ar putea fi un predictor important în clasificare. De asemenea, aglomerările punctelor în funcție de culoarea lor indică posibile separări clare între cazurile de boală și cele fără boală, evidențiind potențialul acestor variabile în a ajuta modelul KNN să identifice corect clasa fiecărui pacient.

****

1. Regresia Multinomială pentru Predicția Variabilei slp

Scopul: Utilizarea unui model de regresie multinomială pentru a prezice valoarea variabilei slp (slope of the peak exercise ST segment) pe baza tuturor celorlalte variabile disponibile. Datele au fost scalate și transformate folosind one-hot encoding pentru variabilele categorice.

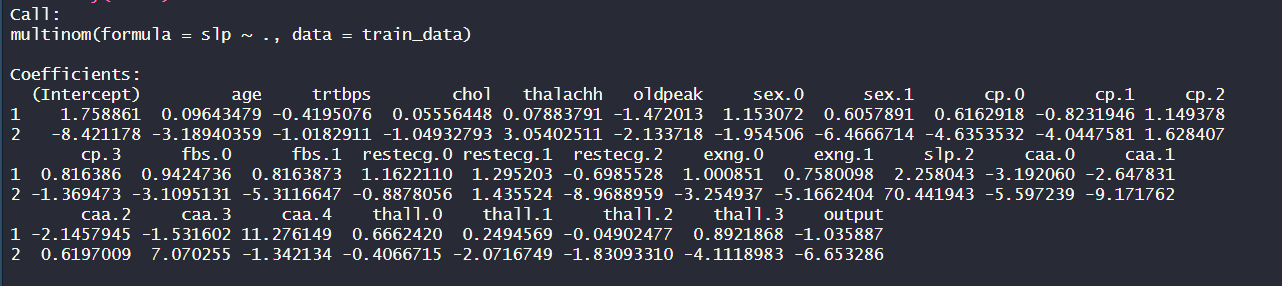


Antrenarea Modelului de Regresie Multinomială:

Modelul a fost antrenat utilizând toate variabilele disponibile ca predictori pentru valoarea slp.

S-au calculat coeficienții pentru fiecare clasă a variabilei țintă (slp), ceea ce permite înțelegerea

Coeficienții modelului indică magnitudinea și direcția impactului fiecărei variabile asupra probabilității fiecărei clase a variabilei slp. De exemplu, pentru clasa 1 a variabilei slp, coeficientul pentru age este 0.09643479, indicând o influență pozitivă a vârstei asupra probabilității de a avea această valoare a slp.



Interpretarea Coeficienților Modelului

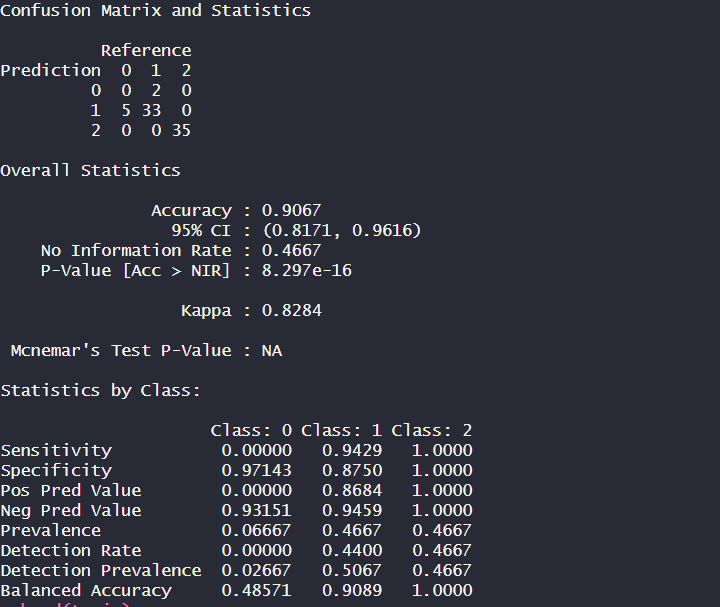
Variabile Predictive: Fiecare coeficient asociat unei variabile predictive indică impactul acelei variabile asupra probabilității de a aparține unei anumite clase a slp. De exemplu, un coeficient pozitiv pentru age în clasa 1 indică faptul că o creștere a vârstei mărește probabilitatea de a avea valoarea 1 pentru slp.

Matricea de Confuzie:

Matricea de confuzie arată cum se compară predicțiile modelului cu valorile reale ale variabilei slp.

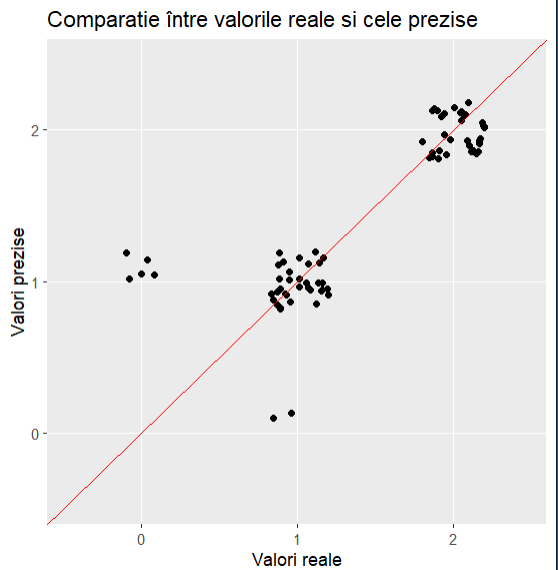
Acuratețea: Modelul are o acuratețe de 90.67%, ceea ce indică o performanță ridicată în predicțiile sale.

Kappa: Valoarea Kappa de 0.8284 sugerează un nivel ridicat de acord între predicțiile modelului și valorile reale, ajustat pentru acordul întâmplător.

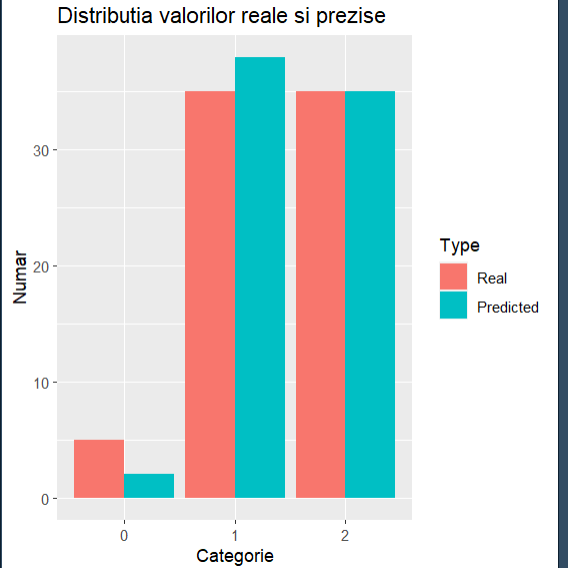


Grafic Comparativ:

Graficul comparativ al valorilor reale și prezise arată că majoritatea punctelor se aliniază pe linia roșie (y = x), indicând o bună precizie a modelului în predicțiile sale.



Graficul distribuției valorilor reale și prezise arată cum se distribuie predicțiile modelului în comparație cu valorile reale pentru fiecare clasă. Se observă o distribuție similară, confirmând performanța modelului.

****

1. Rețele Neuronale: Estimarea Valorii output

Rețele Neuronale: Rețelele neuronale sunt modele de învățare automată inspirate de creierul uman, utilizate pentru recunoașterea tiparelor și clasificarea datelor. Acestea sunt alcătuite din straturi de neuroni artificiali care sunt conectați între ei. Principalele componente ale unei rețele neuronale sunt:

1. Stratul de Intrare (Input Layer): Primul strat care primește datele de intrare. Fiecare neuron din acest strat corespunde unei caracteristici din setul de date.
2. Straturile Ascunse (Hidden Layers): Straturile intermediare între stratul de intrare și cel de ieșire. Acestea procesează informațiile și învață caracteristicile complexe ale datelor.
3. Stratul de Ieșire (Output Layer): Ultimul strat care produce predicțiile finale. Numărul de neuroni din acest strat depinde de numărul de clase în cazul clasificării sau de numărul de valori de ieșire în cazul regresiei.

Scopul: Am folosit o rețea neuronală pentru a prezice valoarea variabilei output pe baza celorlalte variabile din setul de date. Am utilizat tehnici de validare încrucișată pentru a optimiza hiperparametrii modelului, precum dimensiunea rețelei (size) și rata de decădere (decay).

* Împărțirea Datelor:

Datele au fost împărțite în seturi de antrenament și test utilizând o proporție de 75% pentru antrenament și 25% pentru testare.

* Pregătirea Datelor:

Variabila output a fost transformată într-o variabilă factor pentru a facilita antrenarea modelului de rețea neuronală.

* Definirea Gridului de Hiperparametri:

Un grid de hiperparametri a fost definit pentru a explora diferite combinații ale dimensiunii rețelei neuronale (size, numărul de neuroni din straturile ascunse) și ratei de decădere (decay, un parametru care controlează regularizarea și ajută la prevenirea overfitting-ului).

Dimensiunea Rețelei (size): Variază de la 1 la 15 neuroni.

Rata de Decădere (decay): Valori de 0.01, 0.1 și 0.5.

1. Validare Încrucișată:

Am folosit validarea încrucișată cu 5 fold-uri pentru a evalua performanța diferitelor combinații de hiperparametri din gridul definit.

1. Antrenarea Modelului:

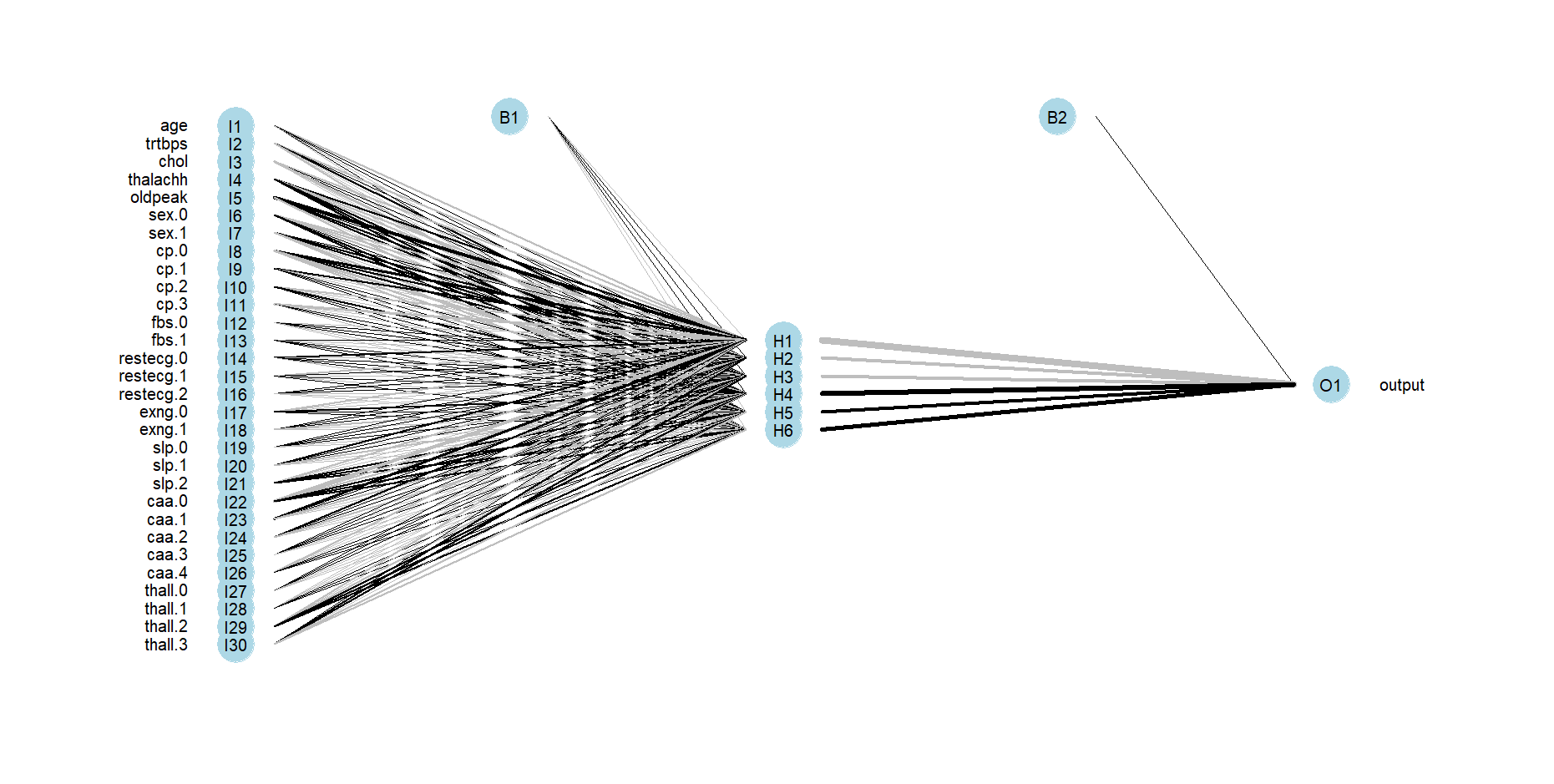
Modelul de rețea neuronală a fost antrenat folosind cele mai bune combinații de hiperparametri identificate în timpul validării încrucișate.

1. Parametrii Optimi:

Parametrii optimi găsiți au fost size = 6 și decay = 0.5.

1. Vizualizarea Rețelei Neuronale:

A fost creată o vizualizare a rețelei neuronale pentru a ilustra structura acesteia, incluzând layerele de input, hidden și output.



1. Predicții pe Setul de Test:

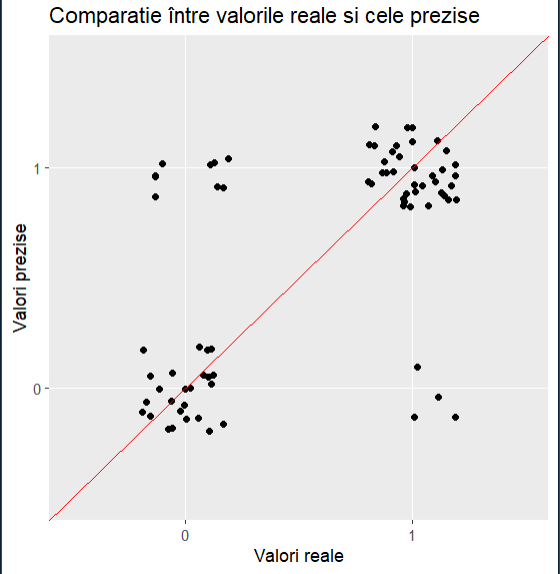
Am utilizat modelul antrenat pentru a face predicții pe setul de date de test și am evaluat performanța acestuia.

1. Matricea de Confuzie:

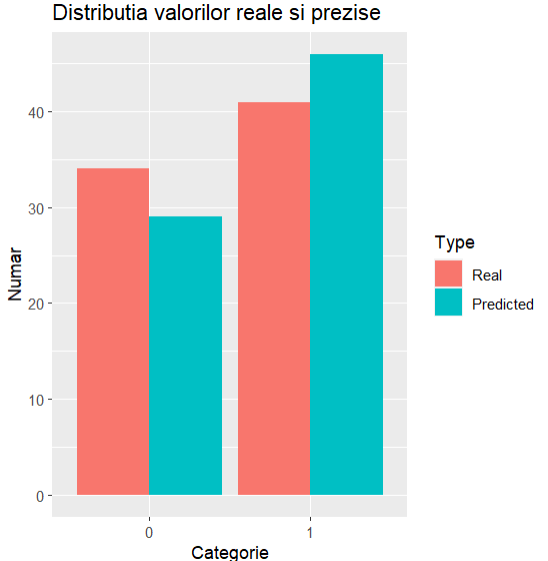
Matricea de confuzie a arătat performanța modelului, cu o acuratețe generală de 82.67%.

Graficul Comparativ:

Graficul comparativ între valorile reale și cele prezise arată că majoritatea punctelor se aliniază pe linia roșie (y = x), indicând o bună precizie a modelului.



Distribuția Valorilor Reale și Prezise:Graficul distribuției valorilor reale și prezise arată cum se distribuie predicțiile modelului în comparație cu valorile reale pentru fiecare categorie.



Performanță Generală: Modelul de rețea neuronală a oferit o performanță bună în predicția variabilei output, cu o acuratețe generală de 82.67% și valori bune de specificitate și sensibilitate.

Optimizarea Hiperparametrilor: Validarea încrucișată a fost esențială pentru identificarea celor mai buni hiperparametri (size = 6 și decay = 0.5), asigurând astfel o performanță optimă a modelului.

1. SVM cu Kernel Radial: Estimarea Valorii output

Support Vector Machine (SVM): SVM este un algoritm utilizat atât pentru clasificare, cât și pentru regresie. SVM funcționează prin găsirea unui hiperplan optim care separă diferitele clase din setul de date. În cazul datelor liniar inseparabile, SVM folosește kerneluri pentru a transforma datele într-un spațiu de dimensiuni mai mari unde acestea pot fi separate liniar.

Kernel Radial (RBF): Kernelul radial funcționează prin maparea datelor într-un spațiu de dimensiuni mai mari utilizând funcția de bază radială (RBF). Acest kernel este util pentru problemele în care frontiera de decizie dintre clase nu este liniară.

Kernel Trick: Kernel trick permite SVM-ului să funcționeze într-un spațiu de dimensiuni mari fără a calcula explicit coordonatele datelor în acel spațiu. În schimb, se folosește o funcție kernel pentru a calcula produsul scalar dintre perechile de puncte din spațiul original.

Pașii Urmați

* Împărțirea Datelor:

Datele au fost împărțite în seturi de antrenament (75%) și test (25%) pentru a asigura că modelul este antrenat pe un subset reprezentativ și testat pe un alt subset pentru a evalua performanța.

* Pregătirea Datelor:

Variabila output a fost transformată într-o variabilă factor pentru a facilita antrenarea modelului de SVM.

* Definirea Gridului de Hiperparametri:

A fost definit un grid de hiperparametri pentru C și sigma:

C: Parametru de regularizare care controlează echilibrul dintre maximizarea marginilor și minimizarea erorilor de clasificare. Valorile testate au fost 0.1, 0.001, 0.005, 0.01, 0.05, 0.5 și 1.

Sigma: Parametru pentru kernelul radial care controlează influența punctelor de date individuale. Valorile testate au fost 0.001, 0.01 și 0.1.

Validare Încrucișată:

A fost utilizată validarea încrucișată cu 5 fold-uri pentru a evalua performanța diferitelor combinații de hiperparametri. Validarea încrucișată ajută la estimarea performanței modelului și la evitarea overfitting-ului, asigurându-se că modelul generalizează bine pe date noi.

Antrenarea Modelului:

Modelul SVM cu kernel radial a fost antrenat pe setul de date de antrenament folosind combinațiile de hiperparametri din grid. Cel mai bun model a fost selectat pe baza performanței în validarea încrucișată.

Parametrii Optimi: Cei mai buni parametri identificați au fost C = 1 și sigma = 0.001.

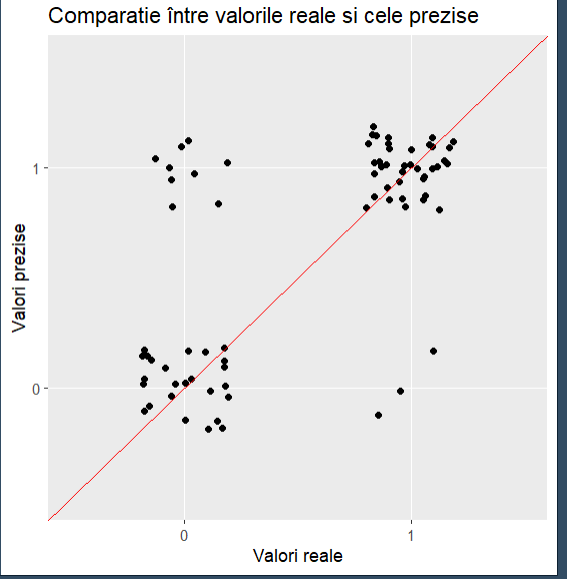
Predicții pe Setul de Test:

Modelul antrenat a fost utilizat pentru a face predicții pe setul de date de test și pentru a evalua performanța acestuia.

Matricea de Confuzie:

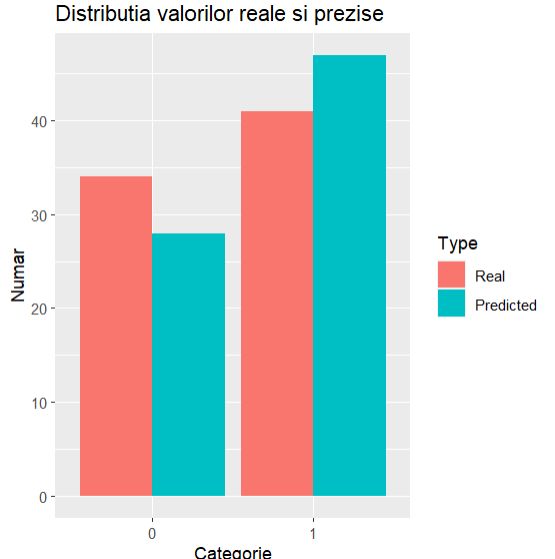
Matricea de confuzie a arătat performanța modelului, cu o acuratețe generală de 84%. Aceasta a furnizat statistici detaliate despre sensibilitate, specificitate și valori predictive.

Graficul comparativ între valorile reale și cele prezise arată că majoritatea punctelor se aliniază pe linia roșie (y = x), indicând o bună precizie a modelului.



**Distribuția Valorilor Reale și Prezise**:

Graficul distribuției valorilor reale și prezise arată cum se distribuie predicțiile modelului în comparație cu valorile reale pentru fiecare categorie.



**Performanță Generală**: Modelul SVM cu kernel radial a oferit o performanță bună în predicția variabilei **output**, cu o acuratețe generală de 84% și valori bune de specificitate și sensibilitate.

**Optimizarea Hiperparametrilor**: Validarea încrucișată a fost esențială pentru identificarea celor mai buni hiperparametri (C = 1 și sigma = 0.001), asigurând astfel o performanță optimă a modelului.

1. **Random Forest: Estimarea Valorii output**

**Random Forest**: Random Forest este un algoritm de învățare automată utilizat atât pentru clasificare, cât și pentru regresie. Este un model bazat pe arbori de decizie care creează o "pădure" de arbori decizionali prin bootstrap sampling și agregarea rezultatelor acestora. Random Forest ajută la reducerea overfitting-ului și la îmbunătățirea preciziei prin combinarea predicțiilor mai multor arbori decizionali.

**Împărțirea Datelor**:

Datele au fost împărțite în seturi de antrenament (75%) și test (25%) pentru a asigura că modelul este antrenat pe un subset reprezentativ și testat pe un alt subset pentru a evalua performanța.

**Pregătirea Datelor**:

Variabilele categorice au fost transformate în factori, iar datele au fost preprocesate folosind one-hot encoding pentru a crea variabile dummy pentru fiecare categorie. Variabilele numerice au fost scalate pentru a asigura uniformitatea.

**Definirea Gridului de Hiperparametri**:

Parametrul **mtry** în contextul algoritmului Random Forest reprezintă numărul de variabile (predictori) alese aleatoriu din setul total de variabile pentru a fi utilizate la fiecare divizare (split) a nodurilor în arborele de decizie. Acest parametru controlează diversitatea arborilor de decizie individuali din pădurea aleatorie.

Am utilizat un grid de hiperparametri pentru **mtry** (numărul de variabile utilizate pentru a împărți fiecare nod în arborii decizionali), testând valorile 2, 16 și 30.

**Validare Încrucișată**:

Am folosit validarea încrucișată cu 5 fold-uri pentru a evalua performanța diferitelor combinații de hiperparametri. Validarea încrucișată ajută la estimarea performanței modelului și la evitarea overfitting-ului, asigurându-se că modelul generalizează bine pe date noi.

**Antrenarea Modelului**:

Modelul Random Forest a fost antrenat pe setul de date de antrenament folosind combinațiile de hiperparametri din grid. Cel mai bun model a fost selectat pe baza performanței în validarea încrucișată.

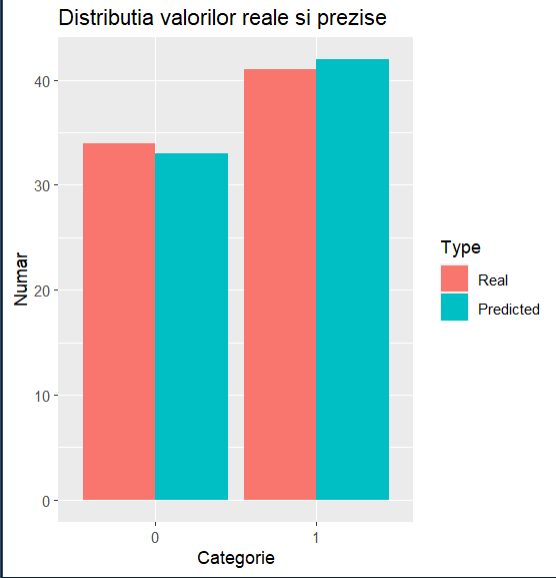
**Predicții pe Setul de Test**:

Modelul antrenat a fost utilizat pentru a face predicții pe setul de date de test și pentru a evalua performanța acestuia.

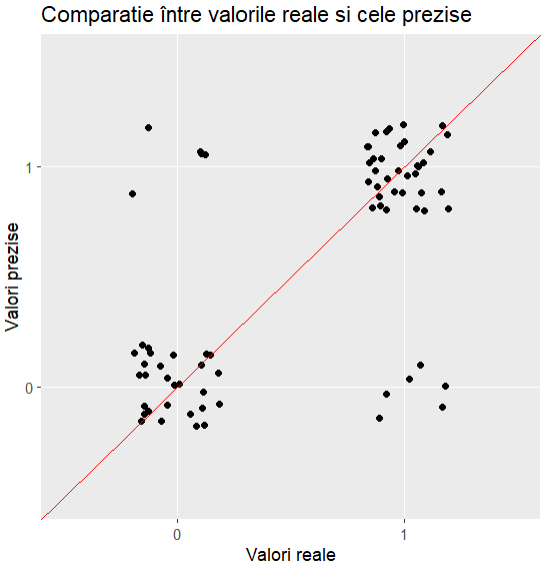
**Matricea de Confuzie**:

Matricea de confuzie a arătat performanța modelului, cu o acuratețe generală de 82.67%. Aceasta a furnizat statistici detaliate despre sensibilitate, specificitate și valori predictive.

Graficul comparativ între valorile reale și cele prezise arată că majoritatea punctelor se aliniază pe linia roșie (y = x), indicând o bună precizie a modelului.



Graficul distribuției valorilor reale și prezise arată cum se distribuie predicțiile modelului în comparație cu valorile reale pentru fiecare categorie.



**Concluzii**

**Performanță Generală**: Modelul Random Forest a oferit o performanță bună în predicția variabilei output, cu o acuratețe generală de 82.67% și valori bune de specificitate și sensibilitate.

**Optimizarea Hiperparametrilor**: Validarea încrucișată a fost esențială pentru identificarea celor mai buni hiperparametri (mtry = 2), asigurând astfel o performanță optimă a modelului.

1. ANEXE

install.packages("caret")

install.packages("rpart")

install.packages("rpart.plot")

install.packages("ggplot2")

install.packages("dplyr")

install.packages("e1071")

install.packages("cluster")

install.packages("factoextra")

install.packages("readxl")

install.packages("readr")

install.packages("caTools")

install.packages("ROCR")

install.packages("nnet")

install.packages("reshape2")

library(caret)

library(rpart)

library(rpart.plot)

library(ggplot2)

library(dplyr)

library(e1071)

library(cluster)

library(factoextra)

library(readxl)

library(readr)

library(ROCR)

library(nnet)

library(ggplot2)

library(dplyr)

library(reshape2)

library(ggplot2)

library(gridExtra)

library(readxl)

library(FNN)

library(NeuralNetTools)

library(caTools)

df <- read.csv("heart.csv")

str(df)

cat("The shape of the dataset is :", dim(df), "\n")

categorical\_cols <-

c('sex', 'exng', 'caa', 'cp', 'fbs', 'restecg', 'slp', 'thall')

for (col in categorical\_cols) {

df[[col]] <- factor(df[[col]])

}

categorical\_cols <-

c('sex', 'exng', 'caa', 'cp', 'fbs', 'restecg', 'slp', 'thall')

continuous\_cols <- c('age', 'trtbps', 'chol', 'thalachh', 'oldpeak')

target\_var <- 'output'

unique\_counts <-

sapply(df[categorical\_cols], function(x)

length(unique(x)))

unique\_counts\_df <- as.data.frame(t(unique\_counts))

colnames(unique\_counts\_df) <- "unique count"

continuous\_summary <- summary(df[continuous\_cols])

list(unique\_counts\_df = unique\_counts\_df,

continuous\_summary = continuous\_summary)

continuous\_summary <- summary(df[continuous\_cols])

continuous\_summary\_transposed <- t(continuous\_summary)

continuous\_summary\_transposed

missing\_counts <- colSums(is.na(df))

missing\_counts

categorical\_cols <-

c('sex', 'exng', 'caa', 'cp', 'fbs', 'restecg', 'slp', 'thall')

color\_palette <-

c("#800000", "#8000ff", "#6aac90", "#5833ff", "#da8829")

plot\_list <- list()

for (col in categorical\_cols) {

plot <- ggplot(df, aes\_string(x = col, fill = col)) +

geom\_bar(color = "black") +

scale\_fill\_manual(values = color\_palette) +

labs(x = "", y = "Count", fill = col) +

theme\_minimal() +

theme(legend.position = "none") +

ggtitle(paste("Count plot for", col))

plot\_list[[col]] <- plot

}

multiplot <- grid.arrange(grobs = plot\_list, nrow = 2)

multiplot

df\_numeric <- df

df\_numeric[] <-

lapply(df\_numeric, function(x)

as.numeric(as.character(x)))

df\_numeric <- df\_numeric[complete.cases(df\_numeric),]

corr\_mat <- as.data.frame(as.table(cor(df\_numeric)))

names(corr\_mat) <- c("feature1", "feature2", "correlation")

heatmap\_plot <-

ggplot(corr\_mat, aes(x = feature1, y = feature2, color = correlation)) +

geom\_point(size = 3) +

scale\_color\_gradient2(

low = "blue",

high = "yellow",

mid = "white",

midpoint = 0,

limits = c(-1, 1),

space = "Lab",

name = "Correlation"

) +

theme\_minimal() +

theme(

axis.text.x = element\_text(

angle = 90,

vjust = 0.5,

hjust = 1

),

legend.position = "bottom",

plot.title = element\_text(hjust = 0.5, size = 20, face = "bold")

) +

labs(x = "Features on X", y = "Features on Y",

title = "Scatterplot Heatmap")

print(heatmap\_plot)

plot1 <- ggplot(df, aes(x = cp, fill = as.factor(output))) +

geom\_bar(position = "fill") +

labs(title = "Chest Pain Distribution",

x = "Chest Pain Type",

y = "Proportion",

fill = "Output") +

scale\_x\_discrete(labels = c(

"Typical Angina",

"Atypical Angina",

"Non-anginal Pain",

"Asymptomatic"

)) +

theme\_minimal()

plot2 <-

ggplot(df, aes(x = as.factor(caa), fill = as.factor(output))) +

geom\_bar(position = "fill") +

labs(title = "Distribution of Number of Major Vessels",

x = "Number of Major Vessels",

y = "Proportion",

fill = "Output") +

theme\_minimal()

plot3 <-

ggplot(df, aes(x = factor(sex), fill = factor(output))) +

geom\_bar(position = "dodge") +

labs(title = "Heart Attack according to sex") +

theme\_minimal()

plot4 <-

ggplot(df, aes(x = as.factor(thall), fill = as.factor(output))) +

geom\_bar(position = "fill") +

labs(title = "Distribution of Thallium Test Results",

x = "Thallium Test Result",

y = "Proportion",

fill = "Output") +

theme\_minimal()

plot5 <-

ggplot(df, aes(

x = as.factor(output),

y = thalachh,

fill = as.factor(output)

)) +

geom\_boxplot() +

labs(title = "Boxen Plot of Thalachh with respect to Outcome",

x = "Outcome",

y = "Thalachh",

fill = "Outcome") +

theme\_minimal()

plot6 <-

ggplot(df, aes(

x = factor(exng),

y = age,

color = factor(output)

)) +

geom\_point(position = position\_dodge(width = 0.5)) +

labs(title = "Strip Plot of exng vs age") +

theme\_minimal()

plot(plot1)

plot(plot2)

plot(plot3)

plot(plot4)

plot(plot5)

plot(plot6)

heart <- df

heart\_numeric <- heart[, sapply(heart, is.numeric)]

result <-

cmeans(

heart\_numeric,

centers = 3,

iter.max = 100,

m = 2,

method = "cmeans"

)

print(result)

ggplot(data = heart, aes(

x = age,

y = thalachh,

color = as.factor(result$cluster)

)) +

geom\_point() +

scale\_color\_manual(values = c("red", "blue", "green")) +

labs(title = "Fuzzy Clustering of Heart Dataset", x = "Age", y = "Max Heart Rate (thalachh)") +

theme\_minimal()

o <- order(result$cluster)

data.frame(heart[o,], Cluster = result$cluster[o])

head(result$membership, 3)

clustered\_heart <- data.frame(heart, Cluster = result$cluster)

heart\_scaled <- scale(heart\_numeric)

res.fanny <- fanny(heart\_scaled, 3)

data.frame(Cluster = res.fanny$clustering, heart)

fviz\_cluster(

res.fanny,

ellipse.type = "norm",

repel = TRUE,

palette = "jco",

ggtheme = theme\_minimal(),

legend = "right"

)

fviz\_silhouette(res.fanny, palette = "jco", ggtheme = theme\_minimal())

print(res.fanny$silinfo)

colnames(heart)

heart <- read\_csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_encoded%>% head()

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

heart\_processed$output <- heart$output

??sample.split

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

training <- subset(heart\_processed, split == TRUE)

testing <- subset(heart\_processed, split == FALSE)

dim(training)

dim(testing)

logistic\_model <-

glm(output ~ ., data = training, family = binomial)

summary(logistic\_model)

prob <-

predict(logistic\_model, newdata = testing, type = "response")

pred <- ifelse(prob > 0.5, 1, 0)

conf\_matrix<-confusionMatrix(table(pred,testing$output))

conf\_matrix

accuracy <- sum(diag(conf\_matrix)) / sum(conf\_matrix)

accuracy

logistic\_model <-

glm(

output ~ cp.0 + sex.0 + slp.1 + caa.2 + thall.2 + thall.1 + exng.1,

data = training,

family = binomial

)

summary(logistic\_model)

prob <-

predict(logistic\_model, newdata = testing, type = "response")

pred <- ifelse(prob > 0.5, 1, 0)

conf\_matrix<-confusionMatrix(table(pred,testing$output))

conf\_matrix

conf\_matrix <- table(Predicted = pred, Actual = testing$output)

conf\_matrix

accuracy <- sum(diag(conf\_matrix)) / sum(conf\_matrix)

accuracy

sensitivity <- conf\_matrix[2, 2] / sum(conf\_matrix[2, ])

specificity <- conf\_matrix[1, 1] / sum(conf\_matrix[1, ])

sensitivity

specificity

p <-

predict(logistic\_model, newdata = testing, type = "response")

pr <- prediction(p, testing$output)

prf <- performance(pr, measure = "tpr", x.measure = "fpr")

roc\_data <- data.frame(FPR = prf@x.values[[1]],

TPR = prf@y.values[[1]])

auc <- performance(pr, measure = "auc")

auc\_value <- auc@y.values[[1]]

ggplot(roc\_data, aes(x = FPR, y = TPR)) +

geom\_line(color = "blue", size = 1) +

geom\_abline(

slope = 1,

intercept = 0,

linetype = "dashed",

color = "red"

) +

theme\_minimal() +

labs(

title = "ROC Curve",

x = "False Positive Rate (1 - Specificity)",

y = "True Positive Rate (Sensitivity)",

caption = paste("AUC =", round(auc\_value, 4))

) +

theme(

plot.title = element\_text(hjust = 0.5, size = 20, face = "bold"),

plot.caption = element\_text(hjust = 0.5, size = 15)

)

##################################################################

set.seed(1)

###ARBORE DE CLASIFICARE

heart <- read.csv("heart.csv")

heart <- na.omit(heart)

heart$output <- as.factor(heart$output)

split <-

createDataPartition(heart$output, p = 0.75, list = FALSE)

train <- heart[split, ]

test <- heart[-split, ]

grid <-

expand.grid(cp = seq(0.0, 0.1, by = 0.01))

control <-

trainControl(method = "cv", number = 5)

model <-

train(

output ~ .,

data = train,

method = "rpart",

trControl = control,

tuneGrid = grid

)

plot(model)

best\_cp <- model$bestTune$cp

best\_cp

final\_model <-

rpart(

output ~ .,

data = train,

method = "class",

control = rpart.control(cp = best\_cp)

)

prp(final\_model, extra = 1)

predictions <-

predict(final\_model, newdata = test, type = "class")

accuracy <- mean(predictions == test$output)

cat("Accuracy:", accuracy, "\n")

confusionMatrix <- confusionMatrix(predictions, test$output)

confusionMatrix

rpart.plot(final\_model)

heatmap\_plot

head(heart)

cor(cbind(heart$oldpeak,heart$output,heart$chol))

heart <- read.csv("heart.csv")

str(heart)

heart\_scaled <-

as.data.frame(scale(heart))

split <-

createDataPartition(heart\_scaled$oldpeak, p = 0.7, list = FALSE)

heart\_train <- heart\_scaled[split, ]

heart\_test <- heart\_scaled[-split, ]

head(heart\_train)

tree\_model\_oldpeak <-

rpart(oldpeak ~ slp, data = heart\_train, method = "anova")

rpart.plot(tree\_model\_oldpeak, main = "Arbore de regresie initial pentru 'oldpeak' în functie de 'output' si 'slp'")

param\_grid\_oldpeak <- expand.grid(cp = seq(0.0, 0.1, by = 0.01))

cv <- trainControl(method = "cv",

number = 5,

verboseIter = TRUE)

grid\_search\_oldpeak <-

train(

oldpeak ~ slp + thall,

data = heart\_train,

method = "rpart",

trControl = cv,

tuneGrid = param\_grid\_oldpeak

)

best\_cp\_oldpeak <- grid\_search\_oldpeak$bestTune$cp

best\_cp\_oldpeak

tree\_model\_oldpeak\_pruned <-

rpart(

oldpeak ~ output + slp,

data = heart\_train,

method = "anova",

control = rpart.control(cp = best\_cp\_oldpeak)

)

predictions\_oldpeak <-

predict(tree\_model\_oldpeak\_pruned, newdata = heart\_test)

mse\_oldpeak <-

mean((predictions\_oldpeak - heart\_test$oldpeak) ^ 2)

mse\_oldpeak

plot(

y=heart\_test$oldpeak,

x=predictions\_oldpeak,

main = "Actual vs Predicted 'oldpeak'",

xlab = "Actual oldpeak",

ylab = "Predicted oldpeak",

col = "blue"

)

abline(0, 1, col = "red")

rmse <-

sqrt(mean((predictions\_oldpeak - heart\_test$oldpeak) ^ 2))

rmse

rpart.plot(tree\_model\_oldpeak\_pruned, main = "Arbore de regresie pruned pentru 'oldpeak' în functie de 'output' si 'slp'")

#######################################################################

set.seed(9902)

heart <- read.csv("heart.csv")

heart\_numeric <- heart[, sapply(heart, is.numeric)]

heart\_numeric <- na.omit(heart\_numeric)

heart\_numeric$output <- as.factor(heart\_numeric$output)

split <-

createDataPartition(heart\_numeric$output, p = 0.7, list = FALSE)

train <- heart\_numeric[split, ]

test <- heart\_numeric[-split, ]

x\_train <- train[, -which(names(train) == "output")]

y\_train <- train$output

x\_test <- test[, -which(names(test) == "output")]

y\_test <- test$output

preProcValues <-

preProcess(x\_train, method = c("center", "scale"))

x\_train <- predict(preProcValues, x\_train)

x\_test <- predict(preProcValues, x\_test)

k\_values <- seq(1, 20, by = 1)

accuracy\_values <- sapply(k\_values, function(k) {

predictions <-

knn(

train = x\_train,

test = x\_test,

cl = y\_train,

k = k

)

confusion\_matrix <- confusionMatrix(predictions, y\_test)

accuracy <- confusion\_matrix$overall['Accuracy']

return(accuracy)

})

heatmap\_plot

best\_k <- k\_values[which.max(accuracy\_values)]

cat("Cel mai bun k:", best\_k, "\n")

calculate\_accuracy <- function(k, x\_train, y\_train, x\_test, y\_test) {

model <- knn(train = x\_train, test = x\_test, cl = y\_train, k = k)

mean(model == y\_test)

}

accuracy\_values <- sapply(k\_values, calculate\_accuracy, x\_train = x\_train, y\_train = y\_train, x\_test = x\_test, y\_test = y\_test)

accuracy\_df <- data.frame(k = k\_values, accuracy = accuracy\_values)

ggplot(accuracy\_df, aes(x = k, y = accuracy)) +

geom\_line() +

geom\_point() +

labs(title = "Acuratețea Modelului KNN în Funcție de k",

x = "Numărul de Vecini (k)",

y = "Acuratețe") +

theme\_minimal()

confusion\_matrix <- confusionMatrix(model, y\_test)

cat("Matricea de confuzie:\n")

print(confusion\_matrix)

cat("Acuratete:", confusion\_matrix$overall['Accuracy'], "\n")

conf\_matrix <- as.data.frame(confusion\_matrix$table)

conf\_matrix\_melted <- melt(conf\_matrix)

ggplot(data = conf\_matrix, aes(x = Prediction, y = Reference, fill = Freq)) +

geom\_tile() +

geom\_text(aes(label = Freq), color = "white") +

scale\_fill\_gradient(low = "blue", high = "red") +

theme\_minimal() +

ggtitle("Matricea de confuzie") +

xlab("Valori prezise") +

ylab("Valori reale")

accuracy\_df <-

data.frame(k = k\_values, accuracy = accuracy\_values)

ggplot(data = accuracy\_df, aes(x = k, y = accuracy)) +

geom\_line(color = "blue") +

geom\_point(color = "red") +

theme\_minimal() +

ggtitle("Acuratetea pentru diferite valori k") +

xlab("Valoarea k") +

ylab("Acuratete") +

scale\_x\_continuous(breaks = k\_values)

heatmap\_plot

#######################################################################

set.seed(123)

heatmap\_plot

data <- read.csv("heart.csv")

heart <- read.csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

split <-

createDataPartition(heart\_processed$output, p = 0.7, list = FALSE)

train\_data <- heart\_processed[split, ]

test\_data <- heart\_processed[-split, ]

str(train\_data)

train\_data$output <- as.numeric(train\_data$output)

test\_data$output <- as.numeric(test\_data$output)

k\_values <- seq(1, 20)

rmse\_values <- numeric(length(k\_values))

for (i in k\_values) {

knn\_pred <- knn(

train = train\_data[,-which(names(train\_data) == "thalachh")],

test = test\_data[,-which(names(test\_data) == "thalachh")],

cl = train\_data$thalachh,

k = i

)

rmse\_values[i] <-

RMSE(as.numeric(as.character(knn\_pred)), test\_data$thalachh)

}

plot\_df <- data.frame(k = k\_values, RMSE = rmse\_values)

ggplot(plot\_df, aes(x = k, y = RMSE)) +

geom\_point() +

geom\_line() +

labs(title = "RMSE pentru diferite valori ale lui k", x = "Valoarea lui k", y = "RMSE")

best\_k <- k\_values[which.min(rmse\_values)]

final\_knn\_pred <- knn(

train = train\_data[,-which(names(train\_data) == "thalachh")],

test = test\_data[,-which(names(test\_data) == "thalachh")],

cl = train\_data$thalachh,

k = best\_k

)

final\_rmse <-

RMSE(as.numeric(as.character(final\_knn\_pred)), test\_data$thalachh)

cat("Cel mai bun k:", best\_k, "\n")

cat("RMSE pentru cel mai bun k:", final\_rmse, "\n")

comparison\_df <-

data.frame(Real = test\_data$thalachh,

Predicted = as.numeric(as.character(final\_knn\_pred)))

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_point() +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

########################################################################

set.seed(9902)

data <- read.csv("heart.csv")

heart <- read.csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

(88)

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

training <- subset(heart\_processed, split == TRUE)

testing <- subset(heart\_processed, split == FALSE)

model <- rpart(thalachh ~ ., data = training, method = "anova")

rpart.plot(model, main = "Arbore de regresie pentru estimarea thalachh")

predictions <- predict(model, testing)

rmse <- RMSE(predictions, testing$thalachh)

cat("RMSE al modelului de arbore de regresie:", rmse, "\n")

comparison\_df <-

data.frame(Real = testing$thalachh, Predicted = predictions)

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_point() +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

cp\_values <- seq(0.001, 0.05, by = 0.001)

rmse\_values <- numeric(length(cp\_values))

for (i in 1:length(cp\_values)) {

model <-

rpart(

thalachh ~ .,

data = training,

method = "anova",

control = rpart.control(cp = cp\_values[i])

)

predictions <- predict(model, testing)

rmse\_values[i] <- RMSE(predictions, testing$thalachh)

}

plot\_df <- data.frame(cp = cp\_values, RMSE = rmse\_values)

ggplot(plot\_df, aes(x = cp, y = RMSE)) +

geom\_point() +

geom\_line() +

labs(title = "RMSE pentru diferite valori ale parametrului cp", x = "Valoarea lui cp", y = "RMSE")

best\_cp <- cp\_values[which.min(rmse\_values)]

final\_model <-

rpart(

thalachh ~ .,

data = training,

method = "anova",

control = rpart.control(cp = best\_cp)

)

rpart.plot(final\_model, main = "Arbore de regresie final pentru estimarea thalachh")

final\_predictions <- predict(final\_model, testing)

final\_rmse <- RMSE(final\_predictions, testing$thalachh)

cat("Cel mai bun cp:", best\_cp, "\n")

cat("RMSE pentru cel mai bun cp:", final\_rmse, "\n")

final\_comparison\_df <-

data.frame(Real = testing$thalachh, Predicted = final\_predictions)

ggplot(final\_comparison\_df, aes(x = Real, y = Predicted)) +

geom\_point() +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise (Model final)", x = "Valori reale", y = "Valori prezise")

######################################################################

heart <- read\_csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

heart\_processed$slp <- heart\_numeric$slp

heart\_processed <- heart\_processed[, -19]

heart\_processed <- heart\_processed[, -19]

head(heart\_processed)

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

train\_data <- subset(heart\_processed, split == TRUE)

test\_data <- subset(heart\_processed, split == FALSE)

train\_data$slp <- as.factor(train\_data$slp)

test\_data$slp <- as.factor(test\_data$slp)

model <- multinom(slp ~ ., data = train\_data)

summary(model)

head(training)

predictions <- predict(model, test\_data)

conf\_matrix <- confusionMatrix(predictions, test\_data$slp)

cat("Matricea de confuzie:\n")

print(conf\_matrix)

accuracy <- conf\_matrix$overall['Accuracy']

cat("Acuratetea modelului de regresie multinomiala:", accuracy, "\n")

comparison\_df <-

data.frame(Real = test\_data$slp, Predicted = predictions)

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_jitter(width = 0.2, height = 0.2) +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

test\_data\_prop <- prop.table(table(test\_data$slp))

cat("Proportiile categoriilor în datele de testare:\n")

print(test\_data\_prop)

predictions\_prop <- prop.table(table(predictions))

cat("Proportiile categoriilor în predictii:\n")

print(predictions\_prop)

comparison\_df <- data.frame(

Category = levels(test\_data$slp),

Real = as.numeric(table(test\_data$slp)),

Predicted = as.numeric(table(predictions))

)

comparison\_df <-

melt(

comparison\_df,

id.vars = "Category",

variable.name = "Type",

value.name = "Count"

)

ggplot(comparison\_df, aes(x = Category, y = Count, fill = Type)) +

geom\_bar(stat = "identity", position = "dodge") +

labs(title = "Distributia valorilor reale si prezise", x = "Categorie", y = "Numar")

##########################################################

data <- read.csv("heart.csv")

heart <- read\_csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

train\_data <- subset(heart\_processed, split == TRUE)

test\_data <- subset(heart\_processed, split == FALSE)

train\_data$output <- as.factor(train\_data$output)

test\_data$output <- as.factor(test\_data$output)

tune\_grid <-

expand.grid(

size = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15),

decay = c(0.01, 0.1, 0.5)

)

train\_control <- trainControl(method = "cv", number = 5)

nn\_model <-

train(

output ~ .,

data = train\_data,

method = "nnet",

tuneGrid = tune\_grid,

trControl = train\_control,

linout = FALSE,

trace = FALSE

)

print(nn\_model)

best\_params <- nn\_model$bestTune

cat("Cei mai buni parametri:\n")

print(best\_params)

plotnet(nn\_model, bias = TRUE)

predictions <- predict(nn\_model, test\_data)

predictions

table(test\_data$output, predictions)

conf\_matrix <- confusionMatrix(predictions, test\_data$output)

cat("Matricea de confuzie:\n")

print(conf\_matrix)

accuracy <- conf\_matrix$overall['Accuracy']

cat("Acuratetea modelului de retea neuronala:", accuracy, "\n")

comparison\_df <-

data.frame(Real = test\_data$output, Predicted = predictions)

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_jitter(width = 0.2, height = 0.2) +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

comparison\_df <- data.frame(

Category = levels(test\_data$output),

Real = as.numeric(table(test\_data$output)),

Predicted = as.numeric(table(predictions))

)

comparison\_df <-

melt(

comparison\_df,

id.vars = "Category",

variable.name = "Type",

value.name = "Count"

)

ggplot(comparison\_df, aes(x = Category, y = Count, fill = Type)) +

geom\_bar(stat = "identity", position = "dodge") +

labs(title = "Distributia valorilor reale si prezise", x = "Categorie", y = "Numar")

#######################################################

set.seed(1234)

heart <- read\_csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

train\_data <- subset(heart\_processed, split == TRUE)

test\_data <- subset(heart\_processed, split == FALSE)

train\_data$output <- as.factor(train\_data$output)

test\_data$output <- as.factor(test\_data$output)

cv <-

trainControl(method = "cv",

number = 5,

verboseIter = TRUE)

tune\_grid <- expand.grid(C = c(0.1, 0.001,0.005,0.01,0.05,0.5, 1),

sigma = c(0.001, 0.01, 0.1))

svm\_model <-

train(

output ~ .,

data = train\_data,

method = "svmRadial",

tuneGrid = tune\_grid,

trControl = cv,

)

print(svm\_model)

best\_params <- svm\_model$bestTune

cat("Cei mai buni parametri:\n")

print(best\_params)

predictions <- predict(svm\_model, test\_data)

conf\_matrix <- confusionMatrix(predictions, test\_data$output)

cat("Matricea de confuzie:\n")

print(conf\_matrix)

accuracy <- conf\_matrix$overall['Accuracy']

cat("Acuratetea modelului SVM:", accuracy, "\n")

comparison\_df <-

data.frame(Real = test\_data$output, Predicted = predictions)

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_jitter(width = 0.2, height = 0.2) +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

comparison\_df <- data.frame(

Category = levels(test\_data$output),

Real = as.numeric(table(test\_data$output)),

Predicted = as.numeric(table(predictions))

)

comparison\_df <-

melt(

comparison\_df,

id.vars = "Category",

variable.name = "Type",

value.name = "Count"

)

ggplot(comparison\_df, aes(x = Category, y = Count, fill = Type)) +

geom\_bar(stat = "identity", position = "dodge") +

labs(title = "Distributia valorilor reale si prezise", x = "Categorie", y = "Numar")

set.seed(1234)

heart <- read\_csv("heart.csv")

categorical\_vars <-

c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")

numerical\_vars <-

c("age", "trtbps", "chol", "thalachh", "oldpeak")

heart[categorical\_vars] <-

lapply(heart[categorical\_vars], as.factor)

heart\_encoded <- dummyVars(" ~ .", data = heart)

heart\_encoded <-

data.frame(predict(heart\_encoded, newdata = heart))

heart\_scaled <-

as.data.frame(scale(heart\_encoded[numerical\_vars]))

heart\_processed <-

cbind(heart\_scaled, heart\_encoded[setdiff(names(heart\_encoded), numerical\_vars)])

head(heart\_processed)

split <- sample.split(heart\_processed$output, SplitRatio = 0.75)

train\_data <- subset(heart\_processed, split == TRUE)

test\_data <- subset(heart\_processed, split == FALSE)

train\_data$output <- as.factor(train\_data$output)

test\_data$output <- as.factor(test\_data$output)

rf\_model <- train(

output ~ .,

data = train\_data,

method = "rf",

trControl = cv,

preProcess = c("center", "scale")

)

print(rf\_model)

predictions <- predict(rf\_model, test\_data)

conf\_matrix <- confusionMatrix(predictions, test\_data$output)

cat("Matricea de confuzie:\n")

print(conf\_matrix)

accuracy <- conf\_matrix$overall['Accuracy']

cat("Acuratetea modelului Random Forest:", accuracy, "\n")

comparison\_df <-

data.frame(Real = test\_data$output, Predicted = predictions)

ggplot(comparison\_df, aes(x = Real, y = Predicted)) +

geom\_jitter(width = 0.2, height = 0.2) +

geom\_abline(slope = 1,

intercept = 0,

color = "red") +

labs(title = "Comparatie între valorile reale si cele prezise", x = "Valori reale", y = "Valori prezise")

comparison\_df <- data.frame(

Category = levels(test\_data$output),

Real = as.numeric(table(test\_data$output)),

Predicted = as.numeric(table(predictions))

)

comparison\_df <-

melt(

comparison\_df,

id.vars = "Category",

variable.name = "Type",

value.name = "Count"

)

ggplot(comparison\_df, aes(x = Category, y = Count, fill = Type)) +

geom\_bar(stat = "identity", position = "dodge") +

labs(title = "Distributia valorilor reale si prezise", x = "Categorie", y = "Numar")

dim(training)