Universitatea Politehnica Bucureşti
Facultatea de Automatică si Calculatoare
Departamentul de Automatică şi Ingineria Sistemelor

Comparaţia algoritmilor K-NN şi Naive Bayes folosind un set de date de metaboliţi

Student
Cruceru Raluca-Elena, 331AA

Prof. Trandafir-Liviu Serghei

Bucureşti, 2024

Cuprins

1 Prezentare generala

Cei doi algoritmi pe care urmează să îi compar în această documentație prezintă două metode de învățare supervizată: k-Nearest Neighbours și Naive Bayes, care evidențiază caracteristici destul de diferite. Lucrul acesta se va putea observa și din interpretarea rezultatului. Setul de date folosit a fost ales astfel încât să aducă o perspectivă asupra modului în care datele sunt manipulate în fiecare dintre cazuri, fiind o analiză a metaboliților ce se află în urină în prezența unor cantități diferite de tratament medicamentos. Studiul implică participarea a 4 persoane de diverse vârste și cu diferite afecțiuni.

2 Introducere

Context si Motivatie - Setul de date:

Ipoteza experimentului pentru care datele din set au fost colectate va fi prezentată în continuare.

Prezența liganzilor acidului orotic, întâlniți în suplimentele alimentare specifice, este ipotezată să introducă o interferență în acuratețea diagnosticului cu debut tardiv al tulburărilor metabolice. Această interferență poate perturba rezultatele așteptate ale testelor de diagnostic, generând rezultate fals pozitive și complicând interpretarea acestor evaluări în contextul clinic.

Astfel, setul de date caută să extragă informații utile pentru a atesta corectitudinea ipotezei acestei teze, care susține că un anumit supliment organic de magneziu disponibil pe piață, chiar și la doza minimă, sub cea recomandată zilnic, ar putea genera fals pozitive pentru anumite tulburări metabolice implicate în calea de sinteză a pirimidinei, atunci când sunt testate în urină în laboratoare medicale, având efecte secundare nebenefice asupra metabolismului.

Cu toate acestea, deoarece această teză este încă în desfășurare, iar datele nu au fost complet colectate sau interpretate, am ales să folosesc acest set pentru a observa dacă există anumite conexiuni între metabolitii aleși și doza medicamentoasă (indiferent de persoana a căror analize aparțin) sau dacă analizele colectate prezintă anumite caracteristici pentru fiecare persoană care a luat parte la studiu. Astfel, am folosit doi vectori de clasă diferiți: primul are două clase - prima parte a studiului, în care doza de medicament este de 6 pastile pe zi, și cea de-a doua parte a studiului, în care doza este injumatățită. Cel de-al doilea vector de clasă este alcătuit din cele 4 persoane care au luat parte la studiu: 1-MIL, 2-DLCL, 3-FIL și 4-KL.

Context si Motivatie - Algoritmi:

Clasificatorul k-Nearest Neighbours (K-NN) este o metodă de învățare supervizată care se bazează pe principiul că instanțele similare se găsesc în proximitatea spațiului caracteristic. Fiind un algoritm leneș, K-NN nu construiește un model explicit, ci stochează setul de antrenament în memorie și efectuează predicții pe baza similarității cu instanțele cunoscute. În cazul nostru, unde analizăm metaboliții din urină în funcție de doza de medicament și participanți, distanța euclidiană este utilizată pentru a determina cei mai apropiați k vecini. Acest algoritm este robust în identificarea structurilor non-lineare în date și poate oferi rezultate relevante în contextul analizei metabolice.

Pe de altă parte, clasificatorul Naive Bayes se bazează pe Teorema lui Bayes, o teoremă probabilistică, pentru a estima probabilitățile asociate cu fiecare clasă. Acest algoritm presupune că caracteristicile sunt independente între ele, deși această asumpție "naivă" poate să nu se potrivească întotdeauna cu datele reale. În cazul nostru, unde valorile analizelor urinei sunt rareori aceleași, modelul probabilistic al Naive Bayes ar putea întâmpina dificultăți. Cu toate acestea, avantajul său constă în antrenarea rapidă și eficientă pe seturi de date mari, iar rezultatele obținute în cadrul studiului nostru vor oferi o perspectivă asupra potențialelor limitări și eficienței acestui algoritm în analiza datelor metabolice.

Pentru o evaluare cuprinzătoare, vom analiza atât performanța numerică a fiecărui algoritm, cât și interpretarea rezultatelor în contextul specific al studiului nostru privind metabolitii urinari. Astfel, putem obține o înțelegere detaliată a potențialului fiecărui algoritm în abordarea particularităților acestui set de date.

3 Algoritmi de invatare supervizata

Definirea modului de functionare al algoritmilor:

Clasificatorul k-Nearest Neighbours (K-NN) este o metodă de învățare supervizată care se bazează pe principiul că instanțele similare se găsesc în proximitatea spațiului caracteristic. Fiind un algoritm leneș, K-NN nu construiește un model explicit, ci stochează setul de antrenament în memorie și efectuează predicții pe baza similarității cu instanțele cunoscute. În cazul nostru, unde analizăm un set de data de poze si se doreste clasificarea lor bazat pe culoarea predominanta din acea poza - red sau blue.

Fiecare fotografie RGB este reprezentată ca un vector într-un spațiu multidimensional, unde fiecare dimensiune corespunde unui canal de culoare (Roșu, Verde, Albastru). De exemplu, un pixel cu valorile RGB (R, G, B) este reprezentat ca un punct în spațiu tridimensional: (R, G, B).

KNN se bazează pe o metrică de distanță pentru a măsura similaritatea între punctele de date. Metricile de distanță comune includ distanța euclidiană și distanța Manhattan. Distanța euclidiană este adesea folosită în acest context și se calculează astfel:

$$\text{Distanța Euclidiană} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

unde n este numărul de dimensiuni (canale de culoare), iar x_i și y_i sunt valorile dimensiunii i pentru două puncte.

Având o nouă fotografie RGB, algoritmul calculează distanța acesteia față de toate celelalte fotografii din setul de antrenament. Identifică cele K fotografii din setul de antrenament cu cele mai scurte distanțe față de fotografia de test.

Pentru clasificare, KNN utilizează un mecanism de votare în majoritate. Se atribuie culoarea predominantă în funcție de clasa majoritară dintre cei K vecini apropiați.

Pe de altă parte, **clasificatorul Naive Bayes** se bazează pe Teorema lui Bayes, o teoremă probabilistică, pentru a estima probabilitățile asociate cu fiecare clasă. Acest algoritm presupune că caracteristicile sunt independente între ele, deși această asumpție "naivă" poate să nu se potrivească întotdeauna cu datele reale.

Fiecare imagine RGB este reprezentată printr-un set de caracteristici, notat $X = \{X_1, X_2, X_3\}$, unde X_i reprezintă intensitatea pentru canalul i.

Probabilitățile a priori pentru fiecare clasă (culoare predominantă) sunt notate $P(Y)$, unde Y reprezintă clasa (culoarea). Aceste probabilități pot fi estimate pe baza frecvenței claselor în setul de antrenament.

Pentru fiecare clasă Y și fiecare caracteristică X_i, se calculează probabilitatea condiționată $P(X_i|Y)$, reprezentând probabilitatea ca o anumită valoare a caracteristicii să aparțină unei anumite clase. Ipoteza "naivă" presupune independența condiționată între caracteristici:

$$P(X_i|Y) = \frac{\text{Numărul de exemple cu clasa } Y \text{ și valoarea caracteristicii } X_i}{\text{Numărul total de exemple cu clasa } Y}$$

Cu ajutorul teoremei Bayes, se calculează probabilitatea posterioară pentru fiecare clasă, dată o imagine X:

$$P(Y|X) = \frac{P(X|Y) \cdot P(Y)}{P(X)}$$

Factorul de scalare $P(X)$ face ca toate probabilitățile să adune la 1, iar $P(X|Y)$ se obține prin înmulțirea probabilităților condiționate $P(X_i|Y)$ pentru toate caracteristicile.

Se alege clasa cu cea mai mare probabilitate posterioară ca rezultat al clasificării.

4 Date

- **Proveniența datelor:**: Datele folosite în acest referat provin dintr-un studiu în curs de finalizare, realizat de o studentă la IMC University of Applied Sciences Krems. Analiza acestor date a fost efectuată în laboratorul Institutului de Biologie București.

- **Descrierea datelor:**: Setul de date este format din 116 de intrări și cuprinde 4 coloane. Acestea sunt denumite **Ort** (concentrația de acid orotic), **Hip** (concentrația de acid hipuric), **Gly** (concentrația de glicină) și **Name** (numele persoanei a cărei analize aparțin, iar mai tarziu, etapa studiului în a doua parte a proiectului).

- **Curatarea Datelor:** Procesul de curățare a datelor a fost simplu. Datele au fost extrase dintr-un document master care includea analiza a 15 metaboliți. Am ales 3 metaboliți relevanți conform recomandărilor autorului tezei, pe care mai tarziu i-am etichetat manual in functie de vectorul de clase pe care l-am folosit. Deoarece analizele au fost completate manual, nu există intrări nule în set, eliminând necesitatea unui pas suplimentar de preprocesare. Unitatea de măsură pentru toate datele în coloanele ce reprezintă metaboliții este procentul, evitând astfel necesitatea schimbării unității de măsură. Pentru coloana de nume, am folosit numere întregi pentru a facilita calculul corelațiilor dintre date. La final, am aplicat o funcție de normalizare standard pentru uniformizarea valorilor.

5 Descriere structurala

Desigur, iată o variantă revizuită pentru a clarifica și adăuga informații:

În cadrul acestei abordări, am analizat mostrele de urină provenind de la 4 persoane, cu vârste cuprinse între 21 și 63 de ani, pe o perioadă de 3 săptămâni. În prima săptămână, aceștia au primit o doză medicamentoasă de 6 pastile pe zi, iar în cea de-a doua săptămână, doza a fost redusă la 3 pastile pe zi.

Procesul de clasificare s-a concentrat pe două aspecte importante:

- Am dorit să determinăm la cine anume aparține fiecare analiză de urină înregistrată, luând în considerare variațiile în funcție de vârstă și alte caracteristici individuale.
- Am investigat modul în care analizele pot indica doza administrată în fiecare săptămână, comparând rezultatele în funcție de dozajul de 6 pastile sau 3 pastile pe zi.

```
"Ort","Hip","Gly","Val","Name"
"3.450751","0.503707","3.450751","0.039752","1"
"3.101136","0.298618","3.101136","0.043566","1"
"3.287557","1.585503","3.287557","0.047762","1"
"0.560943","0.384216","0.560943","0.019574","1"
"0.946996","0.601827","0.946996","0.019937","1"
"4.168658","0.876611","4.168658","0.063098","1"
"2.526954","0.28722","2.526954","0.04264","1"
"0.575864","0.52026","2.187634","0.056503","2"
"0.219731","0.567896","1.784916","0.079872","2"
"1.215133","1.608328","2.513256","0.080144","2"
"0.164416","0.501079","3.134082","0.074136","2"
```

Figura 5.1: Structura setului de date etichetat cu numele

```
Ort,Hip,Gly,Val,Name
3.450751,0.503707,3.450751,0.039752,1
3.101136,0.298618,3.101136,0.043566,1
3.287557,1.585503,3.287557,0.047762,1
0.560943,0.384216,0.560943,0.019574,1
0.946996,0.601827,0.946996,0.019937,1
4.168658,0.876611,4.168658,0.063098,1
2.526954,0.28722,2.526954,0.04264,1
0.575864,0.52026,2.187634,0.056503,1
0.219731,0.567896,1.784916,0.079872,1
1.215133,1.608328,2.513256,0.080144,1
```

Figura 5.2: Structura setului de date etichetat cu etapele

- **Procesul de Antrenare și Validare:** Datele au fost împărțite în seturi distincte pentru antrenare (70%), validare (20%) și testare (10%). A fost aplicată și funcția de amestecare (shuffle), deoarece datele erau clasificate într-o anumită ordine. Alegerea celor doi algoritmi a avut drept scop compararea performanței lor în contextul setului de date specific ales. Această divizare și amestecare au fost esențiale pentru a asigura o evaluare corectă și obiectivă a abilităților predictive ale algoritmilor, evitând introducerea unor posibile biasuri legate de ordinea inițială a datelor.

```
# Split the data into training (70%), validation (20%), and testing (10%)
X_train, X_temp, y_train, y_temp = train_test_split( *arrays: X, y, test_size=0.3, shuffle=True, random_state=42)
X_val, X_test, y_val, y_test = train_test_split( *arrays: X_temp, y_temp, test_size=0.33, shuffle=True, random_state=42)
```

Figura 5.3: Divizia datelor

- **Rezultate și Evaluare:**

  Performanța modelului a fost evaluată prin utilizarea unor metrici specifice algoritmilor aleși. Testul de Acuratețe a furnizat o măsură fundamentală, reflectând proporția de predicții corecte în raport cu totalul acestora. Matricea de Confuzie a oferit o reprezentare detaliată, evidențiind numărul de predicții corecte și cele eronate pentru fiecare clasă în parte.

  De asemenea, s-a analizat corelația dintre variabilele de intrare (Ort, Hip, Gly, Val) și variabila țintă (Name), oferind o măsură a relației statistice între acestea. Pentru o înțelegere mai intuitivă a performanței, s-a creat un grafic de comparare între cei doi algoritmi, furnizând o perspectivă vizuală asupra calității predicțiilor.

- **Relevanța Algoritmilor:** KNN poate fi utilizat pentru a clasifica persoanele în categorii bazate pe profilul lor metabolic. Acesta poate fi util pentru identificarea similarităților între indivizi și pentru a prezice la ce categorie metabolică aparține o nouă persoană. În cazul 2, poate identifica similarități între profilurile metabolice ale persoanelor care au primit 6 pastile și ale celor care au primit 3 pastile. Acesta va încerca să clasifice un nou individ într-una dintre cele două categorii în funcție de similaritatea sa cu vecinii apropiați din setul de antrenare. O altă presupunere ar fi că algoritmul ar putea avea o acuratețe ridicată în cazul 1, unde se dorește identificarea persoanei care deține analizele în funcție de fluctuația de valori a metabolitilor pe care o prezintă pe parcursul studiului, dar având în vedere faptul că analizele sunt luate la intervale scurte de timp, algoritmul Naive Bayes prezintă un potențial ridicat în a putea clasifica datele corect.

  Gaussian Naive Bayes poate fi potrivit pentru clasificarea indivizilor în funcție de profilele metabolice. Este bun pentru modelele în care presupunerea de independență între caracteristici este rezonabilă. În cazul 2, poate evalua probabilitatea ca un profil metabolic să corespundă dozei de 6 pastile sau 3 pastile și să facă o clasificare bazată pe aceste probabilități.

6 Analiza codului

- **Biblioteci Utilizate**
  - NumPy: O bibliotecă puternică pentru operații matematice și manipulare de date.
  - Pandas: Folosit pentru manipularea și analiza datelor într-un format tabular.
  - Matplotlib și Seaborn: Pentru vizualizarea datelor și plotarea rezultatelor.
  - Scikit-learn (sklearn): Include metode pentru scalare și evaluarea modelelor de machine learning.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.naive_bayes import GaussianNB
import numpy as np
```

Figura 6.1: Bibliotecile utilizate

- **Procesul de Preprocesare a Datelor**
  - Citirea Datelor: Datele sunt încărcate din fișierul CSV într-un DataFrame folosind Pandas.
  - Conversia Tipurilor de Date și Separarea Datelor: Coloanele numerice sunt convertite la tipul de date float. Datele sunt separate în caracteristici (X) și variabila țintă (y).
  - Explorarea Datelor: Se explorează distribuțiile variabilelor, tipurile acestora, existența valorilor nule și se analizează numărul de intrări pentru fiecare clasă.
  - Histograma Datelor: Se afișează histograme pentru fiecare metabolit în setul de date.
  - Histograma Datelor pe Clase: Se afișează histograme pentru fiecare metabolit organizate pe clase (Nume).
  - Calcularea și Afișarea Corelației: Se calculează și afișează corelația dintre caracteristici și variabila țintă.

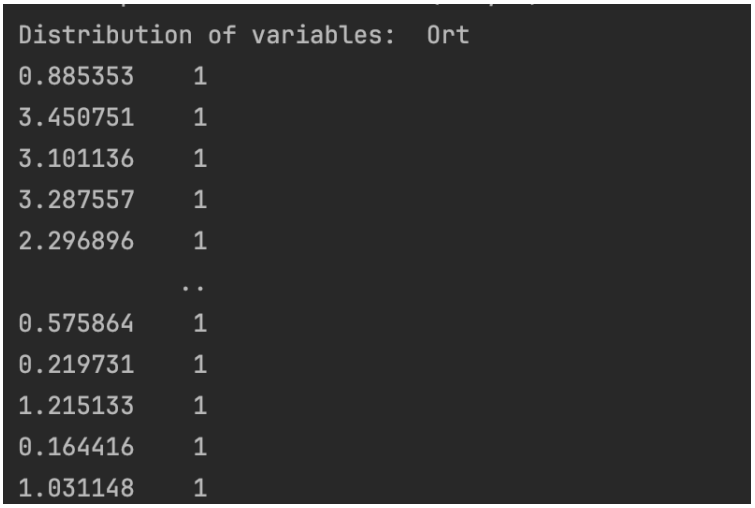

Figura 6.2: Distributia datelor

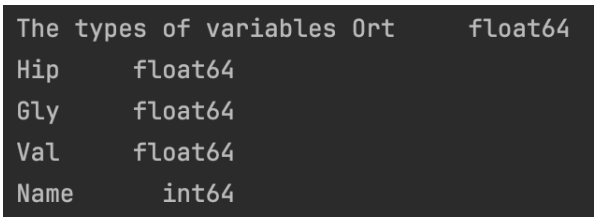

Figura 6.3: Tipul datelor

Figura 6.4: Forma datelor

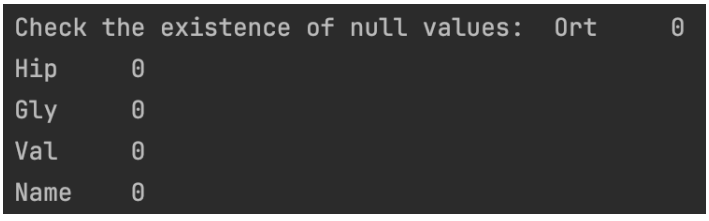

Figura 6.5: Valori nule

```
file_path = 'urinedata2.csv'
df = pd.read_csv(file_path)

print('The shape of the data set: ', df.shape)
# Convert numerical columns to float
df[['Ort', 'Hip', 'Gly', 'Val']] = df[['Ort', 'Hip', 'Gly', 'Val']].astype(float)

# Split the data into features (X) and target (y)
X = df[['Ort', 'Hip', 'Gly', 'Val']]
y = df['Name']

for var in df.columns:
    print('Distribution of variables: ', df[var].value_counts())

print('The types of variables', df.dtypes)

print('Check the existence of null values: ', df.isnull().sum())

print('The number of entries for each class: ', df['Name'].value_counts())

print('The percentage of entries for each class: ', df['Name'].value_counts()/float(len(df)))

df.hist(bins=20, figsize=(10, 6))
plt.suptitle( t: "Histograms for Each Metabolite", y=0.95)
plt.show()


grouped_data = df.groupby('Name')
```

Figura 6.6: Preprocesarea datelor

- **Generarea Seturilor de Antrenare, Validare și Testare**
  - Fereastră de Date: Se folosește o fereastră de date pentru a transforma datele în seturi de antrenare, validare și testare.
  - Normalizarea Datelor: Caracteristicile sunt standardizate pentru a avea aceeași scală.
- **Evaluarea Modelului**
  - Rularea Predictiilor: Modelul este folosit pentru a face predicții pe setul de testare.
  - Calculul Metricilor de Evaluare: Se calculează diferite metrici de evaluare precum matricea de confuzie si testul de acuratete.
- **Vizualizarea Rezultatelor**
  - Graficul Real vs. Predicted: Se folosesc graficele pentru a compara clasificările și performanțele celor doi algoritmi.
  - Matricea de Confuzie și Histograme

```
plt.bar(index, train_accuracies, bar_width, label='Train', color='skyblue')
plt.bar(index + bar_width, val_accuracies, bar_width, label='Validation', color='coral')
plt.bar(index + 2 * bar_width, test_accuracies, bar_width, label='Test', color='limegreen')

plt.xlabel('Classifier')
plt.ylabel('Accuracy')
plt.title('Comparison of Accuracy: KNN vs Gaussian NB')
plt.xticks(index + bar_width, classifiers)
plt.legend()
plt.show()
```

Figura 6.7: Plotarea rezultatelor

- **Modelarea algoritmului**
  - In cadrul algoritmului K Nearest Neighbours am utilizat un k = 5.

7 Rezultate si evaluare

Distributia datelor initiale

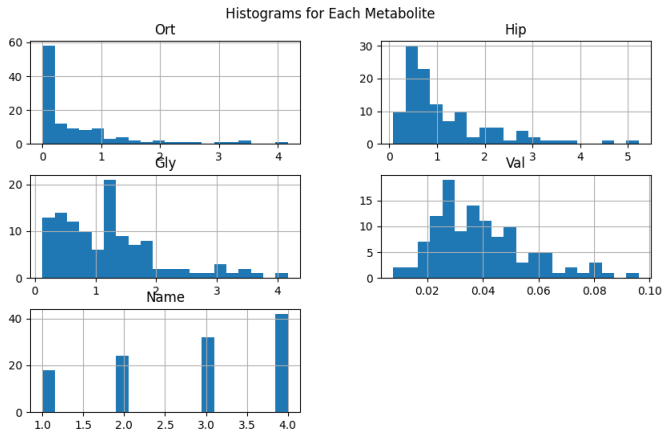

Figura 7.1: Distributia fiecarui metabolit

Din histograma prezentată în imaginea 7.1, se poate observa că cei patru metaboliți prezintă o distribuție destul de diversificată.

Primul metabolit, central în contextul tezei,acidul orotic, are o plajă relativ extinsă de valori, variind de la 0 la 4. Cu toate acestea, valorile prevalente se concentrează în proximitatea lui 0, având aproximativ 40 de apariții. Această observație sugerează că majoritatea datelor au valori scăzute, cu potențiale implicații pentru interpretarea rezultatelor în cadrul studiului.

Al doilea metabolit, Hip, prezintă, de asemenea, o tendință de a avea un număr semnificativ de valori concentrat în intervalul cuprins între 0 și 1, cu aproximativ 30 de apariții. Această concentrare a datelor într-o plajă specifică poate oferi indicii prețioase cu privire la comportamentul metabolitului în cadrul setului de studiu.

Al treilea metabolit manifestă o distribuție predominantă în intervalul 1 și 2, cu aproximativ 20 de apariții. Această distribuție mai uniformă sugerează o variabilitate mai echilibrată în valorile acestui metabolit comparativ cu ceilalți trei.

Ultimul metabolit prezintă valori concentrate în intervalul 0.02 și 0.03, cu aproximativ 25 de apariții. Această concentrare specifică poate reprezenta un punct de interes pentru investigarea posibilelor corelații sau influențe asupra rezultatelor analizelor.

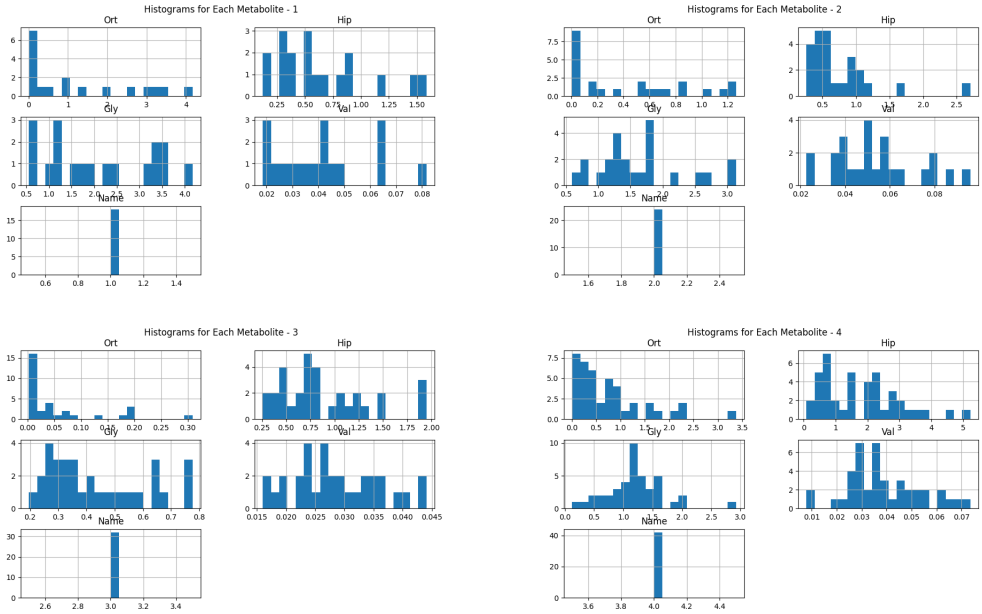

Figura 7.2: Distributia fiecarui metabolit la fiecare persoana

În cadrul analizei fluctuației valorilor metaboliților pentru fiecare persoană, se pot observa discontinuități în valorile acestora, dar acestea sunt relativ diferite în funcție de persoana căreia îi aparțin. De exemplu, în ciuda faptului că pentru persoanele 1, 2 și 4 valorile acidului orotic au ca valoare maximă aproximativ 3, persoana cu numărul 2 prezintă o valoare maximă a acidului orotic de 0,3. Astfel, se poate observa că distribuția datelor prezintă particularități care ar putea recomanda utilizarea algoritmului K-Nearest Neighbours.

Aceste diferențe semnificative între distribuțiile individuale ale metaboliților pot influența rezultatele algoritmului K-Nearest Neighbours, deoarece acesta se bazează pe identificarea celor mai apropiați vecini în funcție de similaritatea datelor. Alegerea acestui algoritm poate să ofere o abordare robustă, având în vedere variabilitatea semnificativă a datelor între indivizi.

```
The percentage of entries for each class:  Name
4    0.362069
3    0.275862
2    0.206897
1    0.155172
```

```
The number of entries for each class:  Name
4    42
3    32
2    24
1    18
```

Figura 7.3: Procentul si numarul fiecarei clase - persoane

Pentru evaluarea rezultatelor celor doi algoritmi, am calculat numărul și procentul de intrări pentru fiecare clasă. Astfel, se poate observa că în primul caz, în 7.3, clasa majoritară este 4, în timp ce în cel de-al doilea caz, în 7.4, clasa majoritară este 1.

```
The number of entries for each class:  Name
1    66
2    50
Name: count, dtype: int64
The percentage of entries for each class:  Name
1    0.568966
2    0.431034
```

Figura 7.4: Procentul si numarul fiecarei clase - etape

O altă observație relevantă în cadrul evaluării performanței algoritmului Naive Bayes este prezentată în 6.2. Deoarece analizele sunt date ce prezintă fluctuații destul de diferite, depinzând de persoană, alimentație și dozaj medicamentos, valorile nu se repetă pe parcursul setului de date, lucru ce poate crea dificultăți pentru aplicarea unui algoritm probabilistic.

Rezultatele primului vector de clase - Clasificarea analizelor in functie de persoana

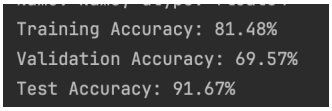

Figura 7.5: Acuratetea algoritmului k-Nearest Neighbours

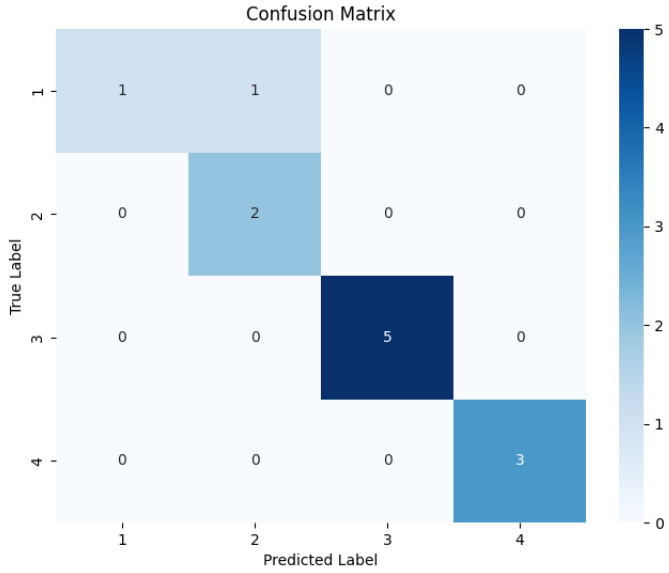

Figura 7.6: Matricea de confuzie pentru setul de testare - K-NN

Din figurile 7.5 și 7.6 se poate observa că presupunerile anterioare au fost relevante în clasificarea datelor. Distribuția datelor pentru fiecare persoană în parte este caracteristică și diversificată, iar valorile fiecărui metabolit prezintă o plajă diferită de variații, cu diferențe de valori semnificative. Astfel, clasificarea bazată pe calcularea distanței dintre acestea este relevantă în cadrul acestui set de date. Se poate observa că singura valoare calculată ca False Negative a fost atunci când clasa 2 a fost etichetată greșit ca fiind clasa 1. Luând în considerare faptul că intervalul de valori al celor două clase (7.2) este cel mai asemănător din punct de vedere al valorilor, se poate înțelege de ce a fost etichetat incorect.

Astfel, având în vedere acuratețea calculată pentru fiecare dintre cele trei seturi de date, este notabilă diferența între acuratețea setului de antrenare și cea a setului de testare. Totuși, având în vedere că setul de testare conține doar 12 valori, se poate înțelege de ce acuratețea este atât de ridicată sau există posibilitatea unui caz de overfitting având în vedere un set de date atât de mic. După împărțirea setului de date în 70% antrenare, 15% testare și 15% validare, acuratețea pentru setul de testare scade sub cea a setului de antrenare. De asemenea, este de luat în considerare faptul că am folosit un k = 5 în cadrul acestui algoritm, lucru care poate conduce la overfitting în cazul unui set de date atât de mic.

```
Gaussian NB Training Accuracy: 77.78%
Gaussian NB Validation Accuracy: 69.57%
Gaussian NB Test Accuracy: 66.67%
```

Figura 7.7: Acuratetea algoritmului Naive Bayes

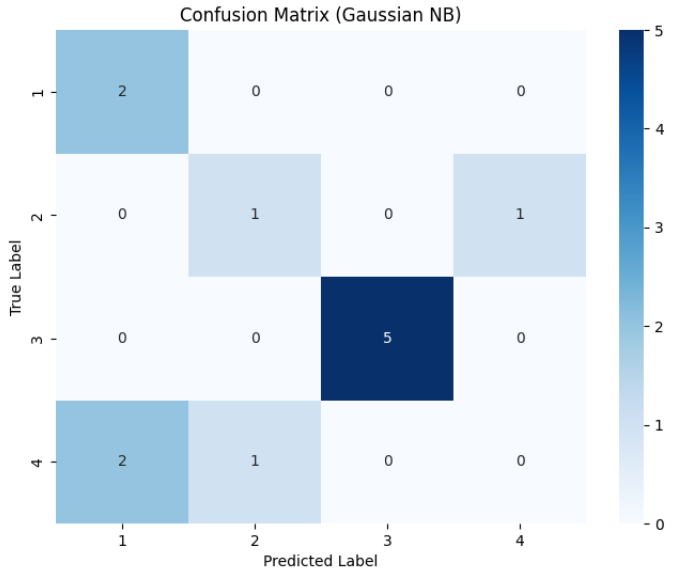

Figura 7.8: Matricea de confuzie pentru setul de testare - NB

Luând în considerare formatul setului de date, așa cum am prezis, algoritmul Naive Bayes are o acuratețe mult mai scăzută față de algoritmul prezentat anterior. Astfel, se poate observa că probabilitatea aparițiilor nu este relevantă pentru clasificarea acestui set de date, făcând algoritmul nepotrivit pentru analizarea acestuia.

Din păcate, matricea de confuzie a acestuia este dificil de interpretat din cauza distribuției datelor și nu se pot trage concluzii pe baza acesteia. Această dificultate poate apărea din cauza variației semnificative a datelor între persoane și a fluctuațiilor valorilor metabolitilor în funcție de diferiți factori precum alimentația și dozajul medicamentos. Acest aspect evidențiază importanța alegerii unui algoritm potrivit pentru caracteristicile specifice ale setului de date.

```
Check how each attribute correlates with the Name variable: Name    1.000000
Hip     0.461552
Ort    -0.122743
Val    -0.206956
Gly    -0.395010
```

Figura 7.9: Corelatia datelor fata de Y

Concentrația de acid orotic (Ort) prezintă o corelație pozitivă moderată (0.46) cu identitatea persoanei (Name). Acest rezultat sugerează că există o asociere semnificativă între concentrația de acid orotic și individul specific.

În cazul acidului hipuric (Hip), observăm o corelație negativă ușoară (-0.122) cu identitatea persoanei (Name). Această corelație indică o asociere slabă, dar negativă, între concentrația de acid hipuric și identitatea persoanei.

Pentru valină (Val), corelația negativă moderată (-0.21) cu identitatea persoanei (Name) indică o relație semnificativă între concentrația de valină și individ.

Glicina (Gly) prezintă o corelație negativă moderată (-0.39) cu identitatea persoanei (Name), sugerând că există o asociere semnificativă între concentrația de glicină și individul specific.

```
Null accuracy score: 0.2241
```

Figura 7.10: Acuratetea nula

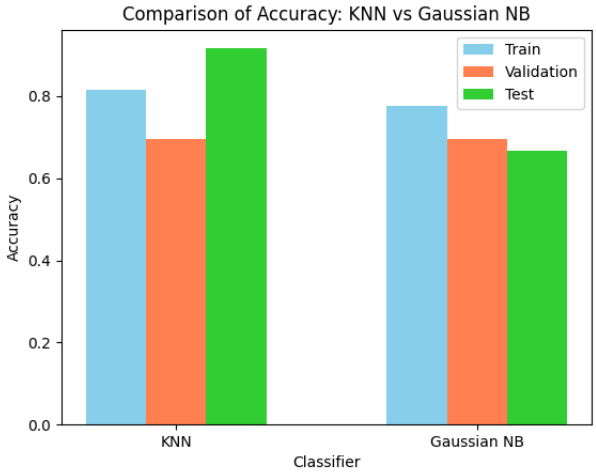

Figura 7.11: Acuratetea celor doi algoritmi comparata

Concluzionând, prin compararea celor doi algoritmi, se poate observa că pentru acest set de date algoritmul non-parametric K Nearest Neighbours este mult mai potrivit, având în vedere nu numai testul de acuratețe 7.10, dar și formatul particular al datelor. Acuratețea nulă, în acest context, este de aproximativ 22%, indicând că, în absența unui algoritm de clasificare, predicțiile aleatorii ar atinge o performanță de aproximativ 22%. Atât K Nearest Neighbours, cât și Naive Bayes depășesc semnificativ această acuratețe nulă, însă algoritmul K Nearest Neighbours prezintă o performanță mai bună în cadrul acestui studiu. Este important de menționat că interpretarea și aplicabilitatea rezultatelor depind în mare măsură de specificul setului de date și de obiectivele analizei.

Rezultatele celui de-al doilea vector de clase - Clasificarea analizelor in functie dozaj

```
Training Accuracy: 61.73%
Validation Accuracy: 60.87%
Test Accuracy: 66.67%
```

Figura 7.12: Acuratetea algoritmului k-Nearest Neighbours

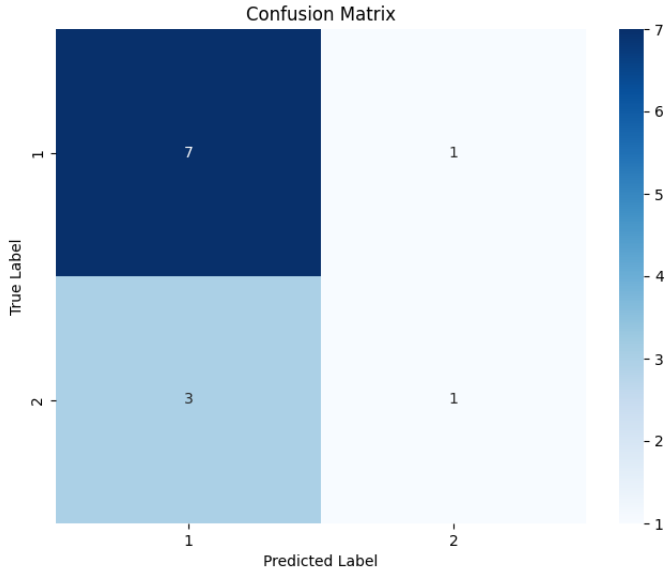

Figura 7.13: Matricea de confuzie pentru setul de testare - K-NN

Conform rezultatelor prezentate în 7.12 și 7.13, se poate observa că ipoteza conform căreia valorile sunt distribuite diferit în funcție de cele două dozaje nu este susținută în cadrul algoritmului K-Nearest Neighbours. Acest fapt arată că variațiile observate în setul de date nu sunt dictate de nivelul dozajului administrat, ci sunt mai degrabă influențate de particularitățile individuale ale participanților la studiu.

```
Gaussian NB Training Accuracy: 56.79%
Gaussian NB Validation Accuracy: 60.87%
Gaussian NB Test Accuracy: 50.00%
```

Figura 7.14: Acuratetea algoritmului Naive Bayes

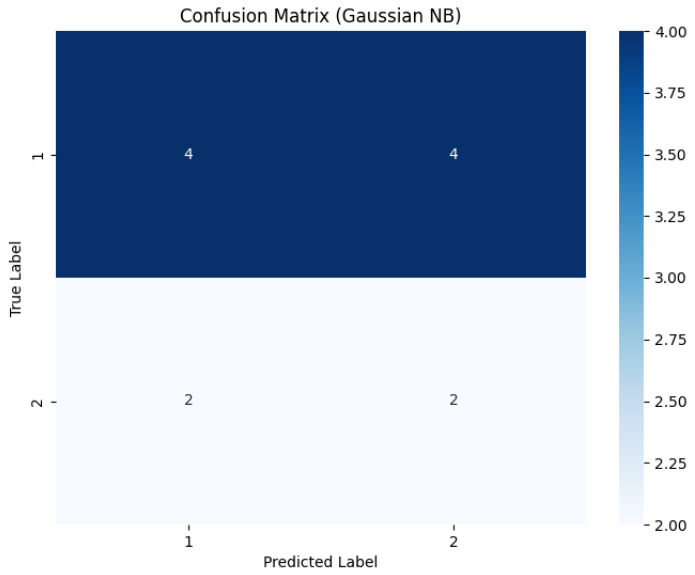

Figura 7.15: Matricea de confuzie pentru setul de testare - NB

Din analiza rezultatelor prezentate în cele două figuri care ilustrează acuratețea algoritmului Naive Bayes (7.14 și 7.15), se deduce aceeași concluzie ca și în cazul evaluării performanței algoritmului anterior.

```
Check how each attribute correlates with the Name variable: Name    1.000000
Ort    -0.084807
Gly    -0.109461
Hip    -0.148576
Val    -0.174887
```

Figura 7.16: Corelatia datelor fata de Y

```
Null accuracy score: 0.5690
```

Figura 7.17: Acuratetea nula

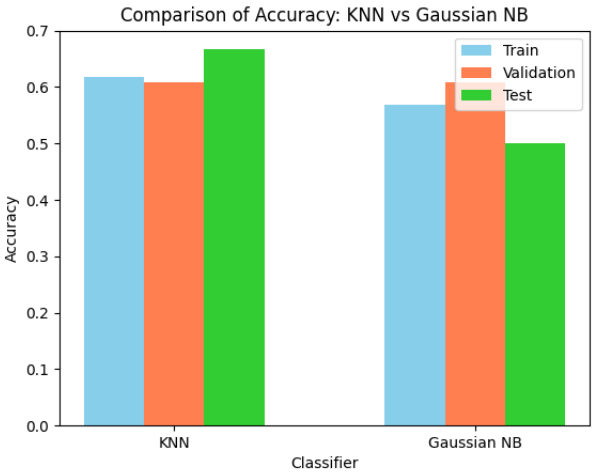

Figura 7.18: Acuratetea celor doi algoritmi comparata

Analizând cele trei figuri care reflectă performanța algoritmilor K-Nearest Neighbours și Naive Bayes, se constată că, în ciuda unei acurateți mai bune în cazul K-Nearest Neighbours comparativ cu Naive Bayes, ambii algoritmi prezintă o performanță sub așteptări pentru setul de date etichetat în acest mod.

Este important să se ia în considerare faptul că performanța algoritmilor de învățare supervizată poate varia în funcție de caracteristicile specifice ale setului de date și de contextul problemei, lucru confirmat in cadrul acestui referat.

De asemenea, menționarea acurateții nule arată că există o dificultate în clasificarea corectă a datelor în acest set, dar aceasta concluzie este relevanta dupa analizarea distributiei datelor pe cele doua clase: 1 si 2.