Antonio-Stefan Smarandescu

333AA

Analiza Performantei Algoritmilor KNN si Naive Bayes pentru Predictia Bolilor Cardiace

Cuprins:

1.Introducere

Motivatie :

Bolile cardiovasculare reprezintă principala cauză de deces la nivel global, conform statisticilor Organizației Mondiale a Sănătății. Datorită impactului major pe care aceste afecțiuni îl au asupra populației, este esențială identificarea timpurie a riscurilor pentru a reduce mortalitatea și a îmbunătăți calitatea vieții.

Proiectul de fata isi propune prin utilizarea tehnicilor de invatare automata sa analizeze rapid si eficient factorii de risc si de a prezice probabilitatea de aparitie a bolilor cardiace.

Descrierea generala a temei :

Acest proiect urmareste compararea perfomantelor a doi algoritmi cunoscuti : KNN(K-Nearest Neighborgs) si Naive Bayes(Gaussian NB) pentru a usura identificarea pacientilor cu risc crescut de boli cardiace.

Studiul este realizat pe baza unui set de date structurat, care conține factori medicali, dar si varsta sau genul pacientilor.

Obiectivele principale sunt:

1. **Antrenarea și testarea modelelor** pe seturi de date separate pentru a evalua performanța fiecărui algoritm.
2. **Comparația rezultatelor** între cei doi algoritmi pentru a determina avantajele și limitările fiecărei metode.
3. **Utilizarea indicatorilor de performanță**, precum matricea de confuzie, acuratețea, sensibilitatea, specificacitatea, precizia și F1-Score, pentru a cuantifica eficiența algoritmilor.

Tipuri de algoritmi folosiți și algoritmii aleși :

1. K-Nearest Neighbors (KNN)

KNN este un algoritm de clasificare supravegheat care atribuie o etichetă unei observații pe baza clasei majoritare a celor mai apropiați k vecini. Acesta este recunoscut pentru:

- Simplitatea implementării.
- Dependența de alegerea corectă a valorii k.
- Performanța redusă în prezența unui număr mare de variabile inutile sau a datelor nereduse dimensional.

2. Naive Bayes (GaussianNB)

Naive Bayes este un algoritm probabilistic care folosește teorema lui Bayes sub ipoteza de independență condiționată între variabilele caracteristice. Este potrivit pentru clasificări rapide și se comportă bine cu seturi de date mari. Avantajele includ:

- Eficiența în timp datorită calculului rapid al probabilităților.
- Robustetea în fața datelor zgomotoase sau lipsite de corelație puternică între caracteristici.

2. Metodologie

Preprocesarea datelor

1. Colectarea și curățarea datelor

Setul de date „heart_disease.csv” utilizat în acest proiect conține informații despre pacienți, incluzând factori precum:

- **Vârsta**
- **Genul**
- **Nivelul colesterolului seric**
- **Presiunea arterială**
- **Nivelul de glicemie**
- **Ritmul cardiac maxim**
- **Diagnosticul bolilor cardiace** (variabilă țintă)

Au fost efectuate următoarele etape pentru a asigura calitatea datelor:

- **Eliminarea valorilor lipsă** pentru a evita introducerea erorilor în modele.
- **Codificarea variabilelor categorice** folosind LabelEncoder, pentru a transforma textul în valori numerice.

2. Normalizarea datelor

S-a aplicat standardizarea caracteristicilor numerice folosind StandardScaler pentru a reduce diferențele dintre variabilele de intrare. Această tehnică este crucială pentru algoritmi precum KNN, care sunt sensibili la magnitudinea variabilelor.

3. Împărțirea setului de date

Setul de date a fost divizat astfel:

- **60% pentru antrenare**: utilizat pentru învățarea parametrilor modelului.
- **20% pentru validare**: destinat optimizării hiperparametrilor și prevenirii supraînvățării.
- **20% pentru testare**: folosit pentru evaluarea finală a performanței modelului

Implementarea algoritmilor

1. Modelul KNN

KNN a fost implementat utilizând biblioteca sklearn:

```
[29] #from sklearn.neighbors import KNeighborsClassifier
     # -------------------------------
     # KNN
     # -------------------------------
     knn = KNeighborsClassifier(n_neighbors=5)
     knn.fit(X_train, y_train)

     # Predicții KNN
     y_train_pred_knn = knn.predict(X_train)
     y_val_pred_knn = knn.predict(X_val)
     y_test_pred_knn = knn.predict(X_test)

     # Afișare rezultate KNN
     plot_confusion_matrix_with_metrics(y_train, y_train_pred_knn, "Training", "KNN")
     plot_confusion_matrix_with_metrics(y_val, y_val_pred_knn, "Validation", "KNN")
     plot_confusion_matrix_with_metrics(y_test, y_test_pred_knn, "Test", "KNN")
```

- Parametrul **k** a fost ales experimental ca 5.
- Distanța folosită: **Euclidiană**.

2. Modelul Naive Bayes

Modelul Naive Bayes Gaussian a fost implementat astfel:

```
#from sklearn.naive_bayes import GaussianNB
# --------------------------------
# Naive Bayes
# --------------------------------
nb = GaussianNB()
nb.fit(X_train, y_train)

# Predicții Naive Bayes
y_train_pred_nb = nb.predict(X_train)
y_val_pred_nb = nb.predict(X_val)
y_test_pred_nb = nb.predict(X_test)

# Afișare rezultate Naive Bayes
plot_confusion_matrix_with_metrics(y_train, y_train_pred_nb, "Training", "Naive B
plot_confusion_matrix_with_metrics(y_val, y_val_pred_nb, "Validation", "Naive Bay
plot_confusion_matrix_with_metrics(y_test, y_test_pred_nb, "Test", "Naive Bayes")
```

- Ipoteza principală: variabilele au o distribuție **gaussiană**.

Evaluarea modelelor

Performanța modelelor a fost analizată utilizând următorii indicatori:

- **Matricea de confuzie** pentru a evidenția distribuția predicțiilor corecte și greșite.
- **Acuratețea**, **precizia**, **sensibilitatea**, **specificitatea** și **F1-Score** pentru evaluarea cantitativă a eficienței.

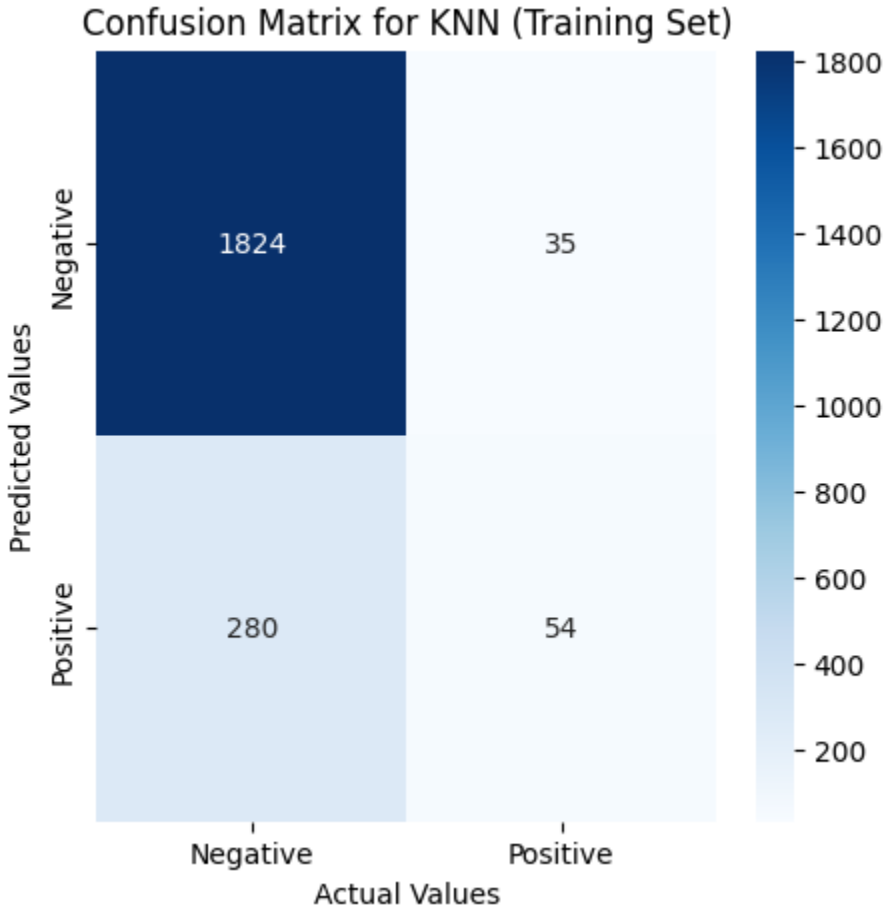
Această metodologie a fost concepută pentru a asigura validitatea și reproductibilitatea rezultatelor.

3. Rezultate obținute

Setul de Antrenare

KNN

- **Matricea de Confuzie**:
    - True Positives (TP): 54



Confusion Matrix for KNN (Training Set):

[[True Positive: 54 False Positive: 35]

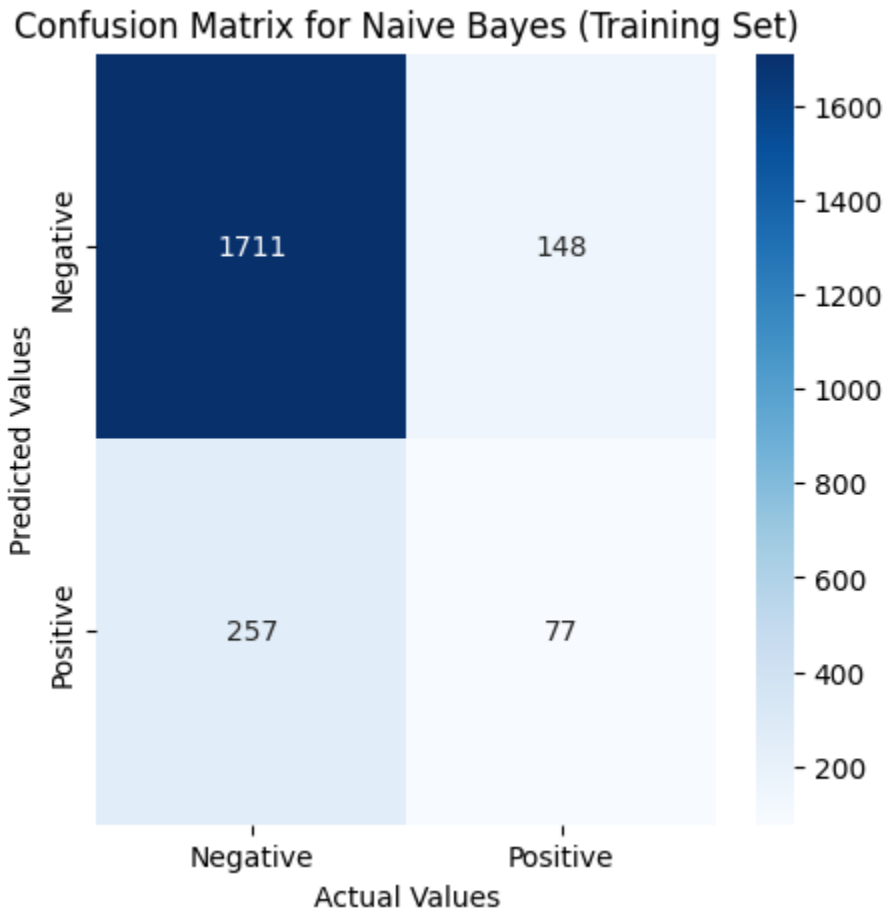
[False Negative: 280 True Negative: 1824]]

Performance Metrics for KNN (Training Set):

- Accuracy: 0.86
- Precision: 0.61
- Recall: 0.16
- Specificity: 0.98
- F1-Score: 0.26

Naive Bayes

- **Matricea de Confuzie**:



Confusion Matrix for Naive Bayes (Training Set):

[[True Positive: 77 False Positive: 148]

[False Negative: 257 True Negative: 1711]]

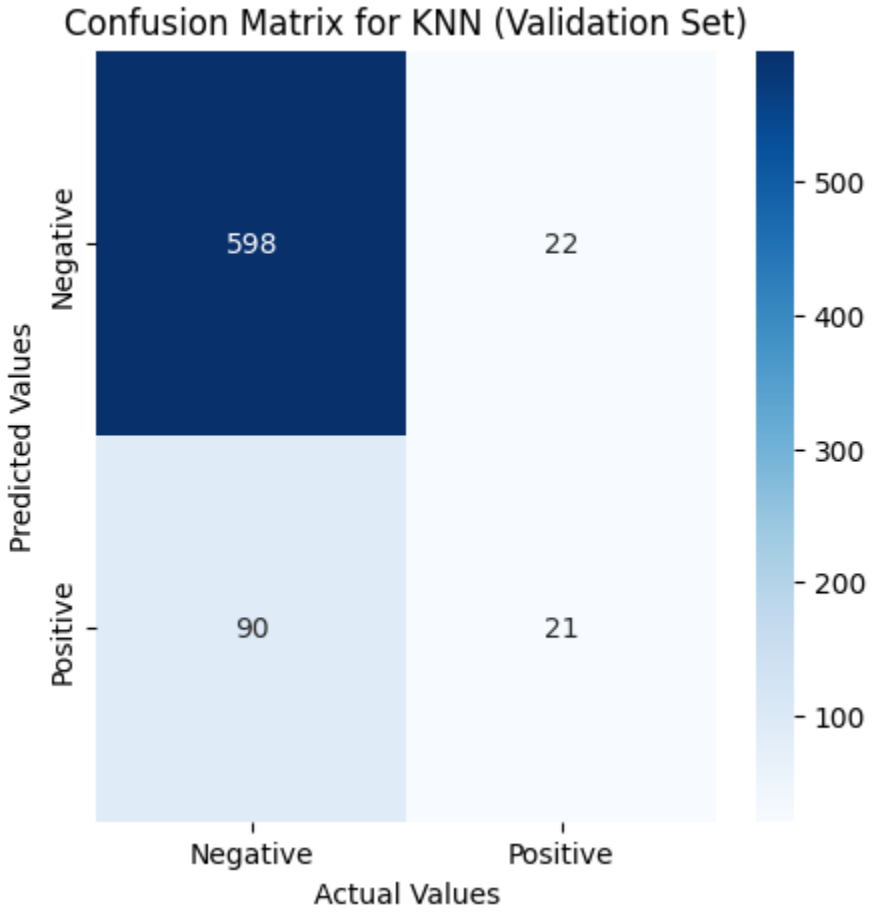
Performance Metrics for Naive Bayes (Training Set):

- Accuracy: 0.82
- Precision: 0.34
- Recall: 0.23
- Specificity: 0.92
- F1-Score: 0.28

Setul de Validare

KNN

- **Matricea de Confuzie**:



Confusion Matrix for KNN (Validation Set):

[[True Positive: 21 False Positive: 22]

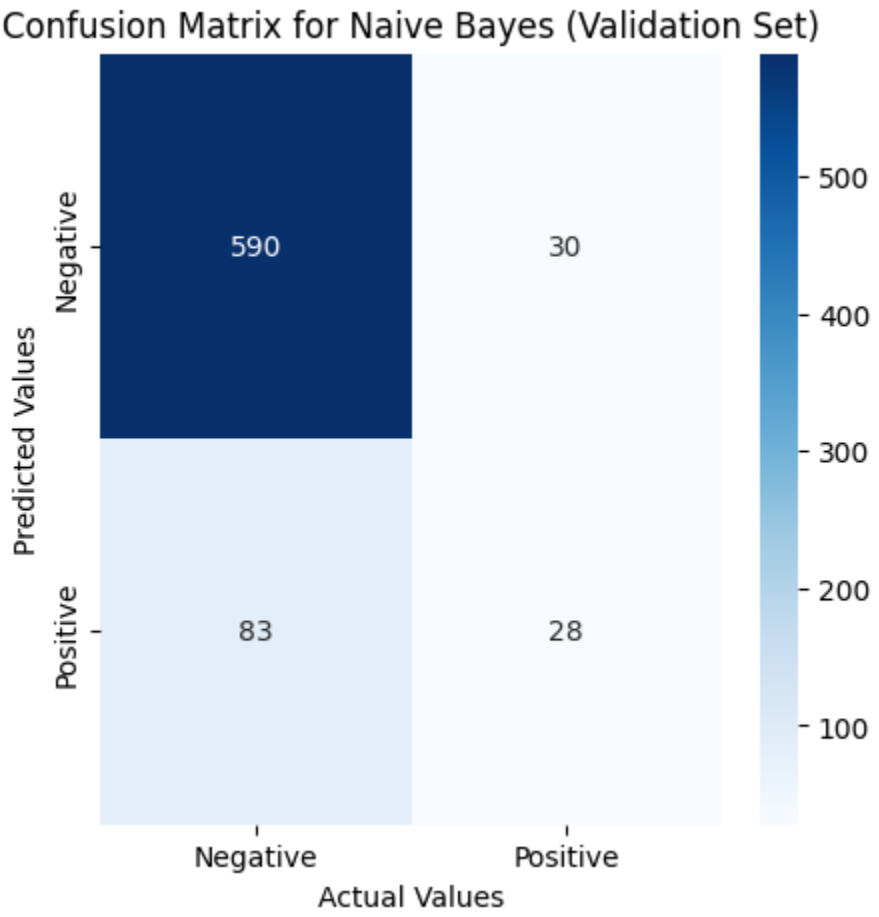
 [False Negative: 90 True Negative: 598]]

Performance Metrics for KNN (Validation Set):

- Accuracy: 0.85
- Precision: 0.49
- Recall: 0.19
- Specificity: 0.96
- F1-Score: 0.27

Naive Bayes

- **Matricea de Confuzie**:



Confusion Matrix for Naive Bayes (Validation Set):

[[True Positive: 28 False Positive: 30]

 [False Negative: 83 True Negative: 590]]

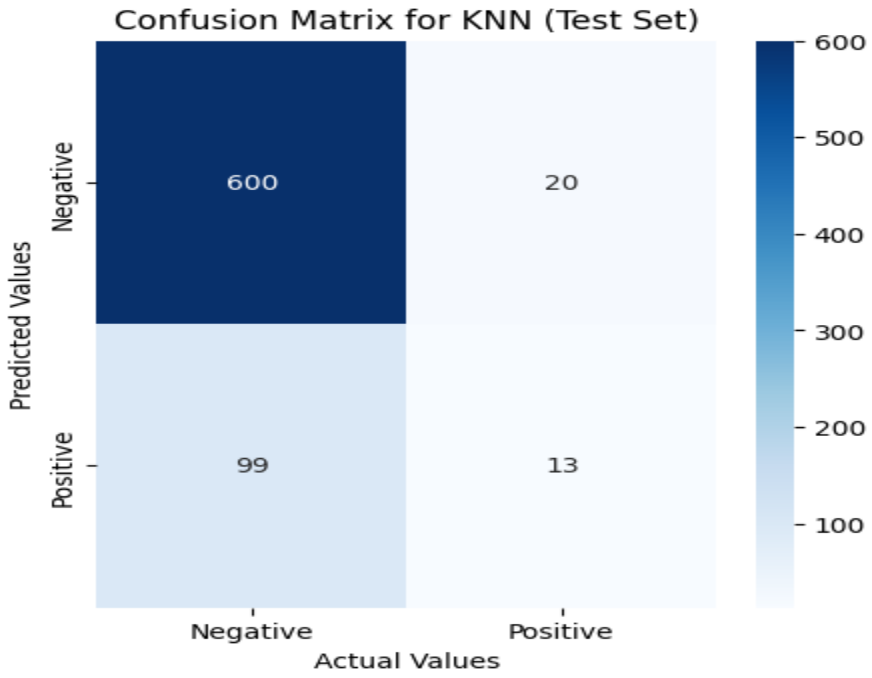
Performance Metrics for Naive Bayes (Validation Set):

- Accuracy: 0.85
- Precision: 0.48
- Recall: 0.25
- Specificity: 0.95
- F1-Score: 0.33

Setul de Testare

KNN

- **Matricea de Confuzie**:



Confusion Matrix for KNN (Test Set):

[[True Positive: 13 False Positive: 20]

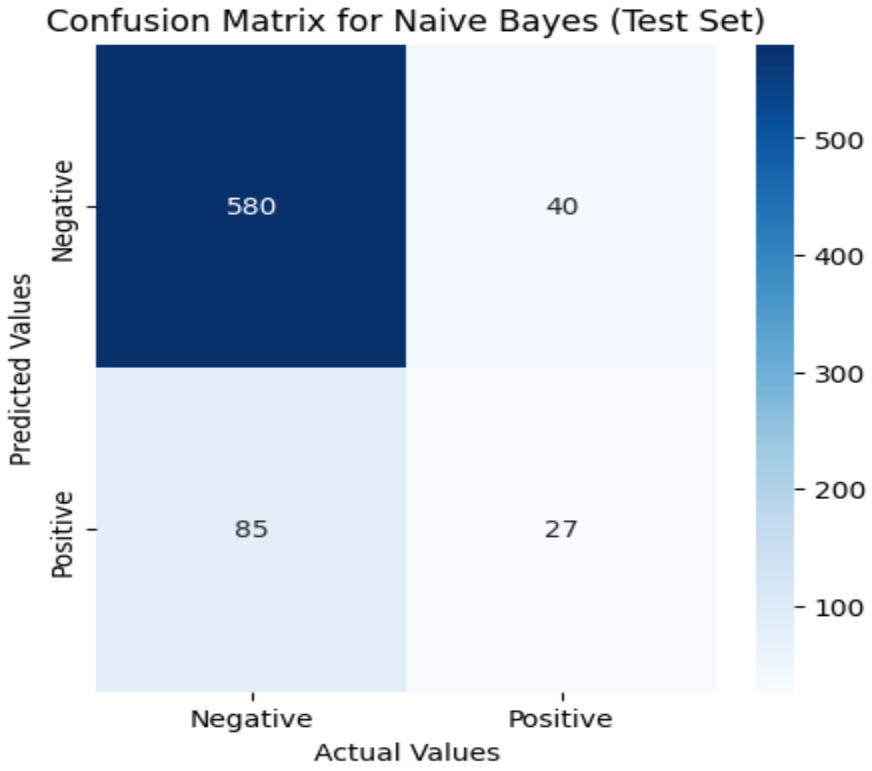
[False Negative: 99 True Negative: 600]]

Performance Metrics for KNN (Test Set):

- Accuracy: 0.84
- Precision: 0.39
- Recall: 0.12
- Specificity: 0.97
- F1-Score: 0.18

Naive Bayes

- **Matricea de Confuzie**:



Confusion Matrix for Naive Bayes (Test Set):

[[True Positive: 27 False Positive: 40]

[False Negative: 85 True Negative: 580]]

Performance Metrics for Naive Bayes (Test Set):

- Accuracy: 0.83
- Precision: 0.40
- Recall: 0.24
- Specificity: 0.94
- F1-Score: 0.30

Explicația valorilor din matricea de confuzie și indicatorii de performanță

1. **Matricea de Confuzie**
    - **True Positives (TP):** Numărul de cazuri pozitive corect clasificate de model (e.g., pacienți cu boli cardiace detectați corect).
    - **True Negatives (TN):** Numărul de cazuri negative corect clasificate de model (e.g., pacienți fără boli cardiace detectați corect).
    - **False Positives (FP):** Numărul de cazuri negative clasificate greșit ca pozitive (e.g., pacienți sănătoși clasificați ca având boli cardiace).
    - **False Negatives (FN):** Numărul de cazuri pozitive clasificate greșit ca negative (e.g., pacienți cu boli cardiace nedetectați).
    - **Interpretare:**
        - Un număr mare de **TP** și **TN** indică un model performant.
        - Un număr mare de **FP** sugerează alarme false, care pot crea probleme inutile.

- Un număr mare de **FN** este critic în acest context, deoarece pacienții bolnavi ar putea fi ratați, ceea ce reprezintă un risc major.

2. Indicatorii de Performanță

- **Acuratețe (Accuracy):** Proporția de predicții corecte realizate de model. O valoare ridicată indică faptul că modelul funcționează bine în ansamblu, dar poate fi influențată de clasele dominante.
- **Precizie (Precision):** Capacitatea modelului de a minimiza alarmele false. Este esențială în situațiile unde alarmele false pot avea consecințe negative (e.g., diagnostic inutil).
- **Sensibilitate (Recall):** Capacitatea modelului de a detecta toate cazurile pozitive. Este crucială în acest proiect, deoarece ratele fals negative trebuie minimizate.
- **Specificitate:** Capacitatea modelului de a clasifica corect cazurile negative. O specificitate ridicată indică faptul că modelul gestionează bine cazurile normale.
- **F1-Score:** Media armonică între precizie și sensibilitate. O valoare ridicată sugerează un echilibru între minimizarea fals pozitive și fals negative.

Comparație Naive Bayes vs. KNN

1. Acuratețe

Naive Bayes oferă o performanță mai constantă pe toate seturile, cu valori între 82% și 85%, ceea ce sugerează o capacitate de generalizare mai bună. Pe de altă parte, KNN prezintă o ușoară scădere a acurateței pe setul de testare (84%), ceea ce poate indica o tendință spre supraînvățare pe setul de antrenare.

2. Precizie

Precizia este mai ridicată pentru KNN pe setul de antrenare (61%), dar scade considerabil pe seturile de validare și testare (49% și 39%). Naive Bayes menține o precizie mai constantă (34%-40%), sugerând o performanță mai echilibrată în tratarea fals pozitive.

3. Sensibilitate (Recall)

Naive Bayes are o sensibilitate superioară pe toate seturile, variind între 23% și 25%, în timp ce KNN are valori mai scăzute (12%-19%). Acest lucru indică faptul că Naive Bayes este mai bun în identificarea cazurilor pozitive.

4. Specificitate

Ambele modele au specificitate ridicată, KNN având un ușor avantaj pe setul de antrenare și testare (97%-98%). Specificitatea ridicată arată că ambele modele clasifică bine cazurile negative, reducând alarmele false.

5. F1-Score

Naive Bayes prezintă un F1-Score mai ridicat pe seturile de validare și testare (33% și 30%), comparativ cu KNN (27% și 18%), ceea ce indică un echilibru mai bun între precizie și sensibilitate.

4. Concluzii

În baza analizei efectuate, se poate observa că:

1. **KNN** are o acuratețe mai mare pe setul de antrenare, dar performanța sa scade pe date noi, ceea ce indică o posibilă tendință de supraînvățare. Algoritmul este mai precis în reducerea fals pozitive, dar întâmpină dificultăți în identificarea cazurilor pozitive (sensibilitate scăzută).
2. **Naive Bayes** oferă o performanță mai echilibrată și consistentă pe toate seturile de date. Este mai potrivit pentru detectarea cazurilor pozitive, având o sensibilitate superioară, dar este mai predispus la clasificări fals pozitive, având o precizie mai scăzută decât KNN.
3. Alegerea algoritmului depinde de priorități: dacă evitarea fals pozitive este crucială, **KNN** ar putea fi preferat. Dacă identificarea cazurilor pozitive este mai importantă, atunci **Naive Bayes** este recomandat.

În concluzie, pentru acest set de date, **Naive Bayes** este algoritmul preferat datorită robusteții, consistenței și performanței sale echilibrate.