Raport: Clasificarea Spam-ului folosind Naive Bayes

Bibirea Mihnea-Constantin, Tăutu Iulia-Magda

January 12, 2024

1 Introducere

În această lucrare practică, am abordat problemă clasificării mailurilor în două categorii distincte: spam și non-spam. Această problemă este destul de importantă în domeniul filtrării e-mailurilor, unde se dorește eliminarea mesajelor nedorite. Pentru rezolvarea acesteia, am ales să implementăm algoritmul Bayes Naive, un model probabilistic care poate oferi performanțe bune în astfel de scenarii.

2 Algoritmul Bayes Naive

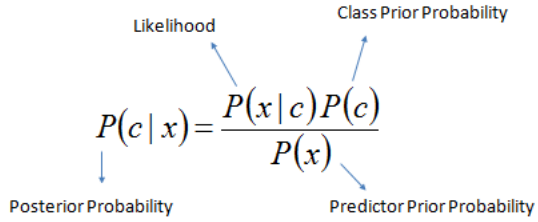
Algoritmul Bayes Naive se bazează pe principiul că fiecare caracteristică a datelor de intrare este independentă față de celelalte, făcând presupunerea "naivă" că aceste caracteristici nu sunt corelate. Pentru clasificarea spam-ului, acest algoritm utilizează probabilități condiționate pentru a determina probabilitatea ca un mesaj să fie spam sau non-spam, dată fiind observația caracteristicilor mesajului. Acest algoritm este construit pe două concepte principale:

1. **Independența Condiționată:**

   Algoritmul presupune că toate variabilele utilizate pentru clasificare sunt independente condiționat față de clasa de ieșire. Această presupunere este cunoscută sub numele de "naive" sau "naivitatea" algoritmului. Această ipoteză poate să nu fie întotdeauna valabilă în practică, dar face algoritmul ușor de implementat și eficient în multe situații.

2. **Teorema lui Bayes:**

   Algoritmul Naive Bayes se folosește de teorema lui Bayes pentru a calcula probabilitățile a posteriori ale claselor date observațiile. Formula generală a teoremei lui Bayes este:



$$P(c \mid \mathbf{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

=

3 Potrivirea pentru problema data

- **Eficiență în spații mari de caracteristici:** Datorită naivității, algoritmul funcționează bine în spații mari de variabile, cum ar fi reprezentarea bag-of-words a documentelor, ceea ce îl face potrivit pentru problemele cu un număr mare de cuvinte cheie posibile.
- **Eficient și ușor de implementat:** Algoritmul este computațional eficient și ușor de implementat, ceea ce îl face o alegere potrivită în situații în care resursele sunt limitate.

3.1 Metodologia Experimentului

Împărțirea setului de date: Setul de date este împărțit în 10 părți, numerotate de la 1 la 10. Primele 9 părți sunt utilizate pentru antrenament, iar partea 10 este rezervată pentru testare. Această abordare este cunoscută sub numele de "cross-validation", unde fiecare parte servește alternativ ca set de testare, iar restul ca set de antrenare.
Evaluare performanță: Se utilizează o metrică simplă, precum acuratețea (accuracy), pentru a evalua performanțele modelului. Acuratețea este calculată ca raportul dintre numărul de predicții corecte și numărul total de exemple de testare.

4 Comparare cu Alți Algoritmi

1. K-Nearest Neighbors - KNN:
   De ce: KNN este non-parametric și poate fi eficient pentru seturi de date mici sau când relațiile dintre puncte sunt complexe.
   Comparare: KNN poate să devină computațional costisitor și să aibă o performanță mai slabă pe seturi de date cu multe caracteristici.

2. Arbori de Decizie:
   De ce: Arborii de decizie sunt ușor de înțeles, interpretat și vizualizat.

Sunt potriviți pentru seturi de date cu caracteristici categorice sau numerice.
Comparare: Arborii de decizie pot fi vulnerabili la overfitting și pot avea o performanță mai scăzută pe seturi de date cu multe caracteristici sau date zgomotoase.

5 Concluzii

Algoritmul Naive Bayes este o alegere potrivită pentru problemele de clasificare a textului, cum ar fi filtrarea mesajelor spam, datorită eficienței sale în gestionarea datelor textuale. În cazul nostru, rezultatele bune ale metricilor sugerează că Naive Bayes este o alegere solidă pentru clasificarea e-mailurilor în spam și non-spam.
Dezavantaje ale algoritmului Bayes Naive:

1. Ipoteza de independență condiționată:
   Presupunerea că toate caracteristicile sunt independente condiționat poate să nu fie întotdeauna valabilă în practică, mai ales în probleme complexe.

2. Performanță redusă pe caracteristici corelate:
   Dacă caracteristicile sunt puternic corelate, algoritmul poate să ofere performanțe mai slabe.