Universitatea ,,Alexandru Ioan Cuza”

Studiu asupra Clasificării Email-urilor Spam

Student name: *Bondar Florina-Alina, Ungurean Ana-Maria 3A4*

Course: *Machine Learning* – Professor: *Dimitriu Corina*
Due date: *January 12th, 2024*

1 Introducere

Email-urile spam reprezintă o problemă persistentă în mediul online, afectând eficiența comunicațiilor și generând disconfort utilizatorilor. Pentru a contracara această problemă, numeroși algoritmi de clasificare au fost dezvoltați și studiați. În acest document, ne propunem să analizăm și să evaluăm gradul de adaptare/adecvare al unor astfel de algoritmi în contextul clasificării email-urilor spam, utilizând setul de date Ling-Spam.

Setul de date Ling-Spam, disponibil la adresa `http://www.aueb.gr/users/ion/data/lingspam_public.tar.gz`, oferă o colecție diversificată de email-uri, cuprinzând atât mesaje spam, cât și non-spam (ham). Scopul principal al acestui studiu este să analizeze, din punct de vedere teoretic și experimental, performanța algoritmilor existenți în abordarea acestei probleme specifice.

2 Înțelegerea setului de date

2.1. Analiza setului de date. Observăm că setul de date este complex, compus din patru foldere denumite ***lemm, bare, stop, lemm_stop***, fiecare conținând zece subfoldere de la ***part1*** la ***part10***. Pentru antrenarea clasificatorilor, vom utiliza primele nouă subfoldere din cele patru foldere menționate anterior. Celelalte subfoldere, mai precis cele numite ***part10***, vor fi folosite pentru testarea clasificatorilor. De asemenea, observăm că mail-urile considerate a fi spam conțin prefixul ***spm*** în titlul fișierului. Acest aspect va facilita etichetarea instanțelor din setul de date utilizat de clasificatori.

2.2. Identificarea atributelor și etichetelor. Un alt pas în procesul de clasificare este identificarea atributelor și etichetelor. Atributele sunt caracteristicile specifice ale fiecărei instanțe din setul de date, în timp ce etichetele indică clasificarea corectă a acestora.

În cazul nostru, pentru setul de date Ling-Spam, atributele ar putea include diverse caracteristici extrase din conținutul e-mail-urilor, cum ar fi frecvența cuvintelor cheie, lungimea mesajului, sau altele relevante pentru identificarea spam-ului. În clasificările ce urmează a fi implementate am considerat frecvența cuvintelor cheie ca a fiind atributele. Iar, etichetele indica dacă un e-mail este considerat spam sau nu în funcție de numele fișierului.

2.3. Preprocesarea datelor. Preprocesarea datelor este un alt pas esențial în analiza textului. Este necesară pentru uniformizarea, curățarea și simplificarea informațiilor,

contribuind la îmbunătățirea performanței modelelor și la generalizarea lor eficientă la noi date.

1. **Conversia la litere mici (`to_lowercase`):**

   Această etapă uniformizează textul, asigurând că literele mici și majuscule sunt tratate la fel în procesul de clasificare. Astfel, "Subject" și "subject" sunt considerate aceeași entitate.

2. **Eliminarea semnelor de punctuație (`remove_punctuation`):**

   Punctuația este eliminată pentru a reduce dimensiunea vocabularului și pentru a îmbunătăți eficiența algoritmului de clasificare. De obicei, semnele de punctuație nu aduc informații semantice semnificative.

3. **Tokenizarea (`tokenize`):**

   Tokenizarea descompune textul în unități semantice mai mici, cunoscute sub numele de "token-uri" (de obicei, cuvinte). Această etapă simplifică analiza ulterioară a textului.

4. **Eliminarea cuvintelor de oprire (`remove_stopwords`):**

   Cuvintele de oprire sunt eliminate pentru a reduce zgomotul și pentru a accelera procesul de clasificare. Acestea includ cuvinte comune, cum ar fi "și", "sau", "cu", care adesea nu aduc informații semantice semnificative.

5. **Eliminarea numerelor (`remove_numbers`):**

   Numerele sunt eliminate pentru a evita eventualele confuzii și pentru a spori generalizarea modelului. De cele mai multe ori, numerele nu aduc informații semantice relevante pentru clasificarea email-urilor.

Datele de antrenare preprocesate sunt stocate în fișierul numit `train_data_processed.csv`. Acest fișier conține două coloane: una pentru textul preprocesat și una pentru indicatorul dacă textul aparține sau nu unui email spam. În același format se găsesc și datele de test în fișierul `test_data_processed.csv`

2.4. Analiza cuvintelor. Analizând cuvintele frecvente în e-mailurile spam și non-spam, observăm diferențe semantice semnificative între cele două categorii.

În e-mailurile spam, se remarcă utilizarea cuvintelor precum "email," "order," "report," "mail," "address," sugerând o posibilă intenție de a atrage atenția asupra unor acțiuni specifice, cum ar fi plasarea unei comenzi sau raportarea de informații.

Pe de altă parte, în e-mailurile non-spam, cuvintele precum "university," "subject," "linguistic," "conference," reflectă un conținut mai academic și informativ, asociat limbajului universitar și cercetării științifice.

Această analiză indică faptul că conținutul cuvintelor cheie în e-mailurile spam și non-spam poate oferi indicii semantice asupra naturii și intențiilor mesajelor, facilitând astfel clasificarea lor automată.

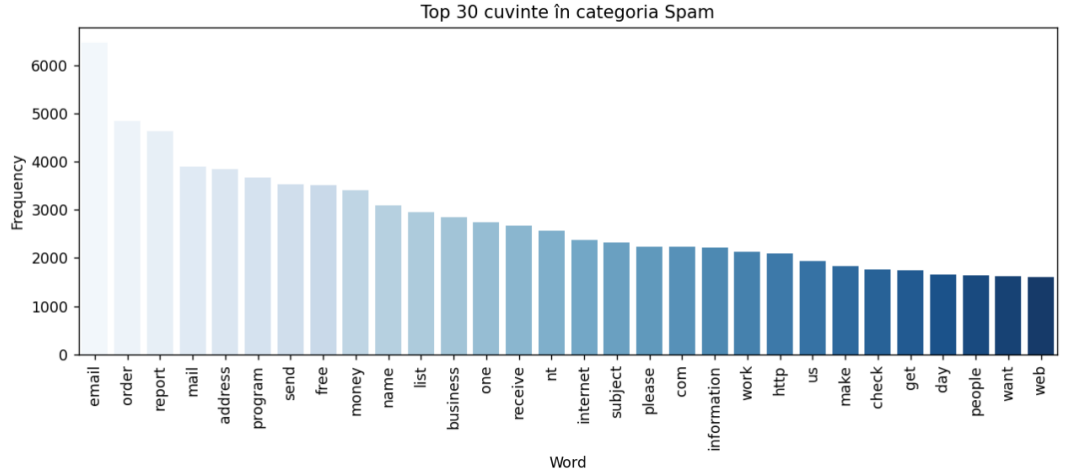

Figure 1: Cele mai frecvente 30 de cuvinte în e-mailurile spam

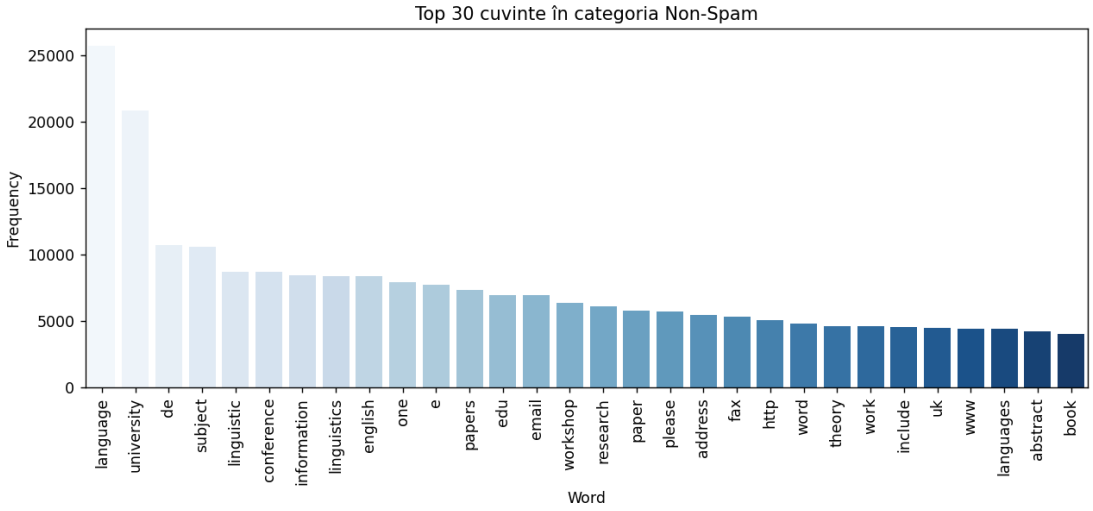

Figure 2: Cele mai frecvente 30 de cuvinte în e-mailurile non-spam

3 Implementarea algoritmului Bayes Naiv

3.1. Explicare implementare.

1. **Inițializare și Încărcare Date:** Inițializarea clasificatorului cu fișierele de antrenare și testare. `train_data_processed.csv` și `test_data_processed.csv` sunt încărcate în variabilele `train_data` și `test_data`.

2. **Antrenare:** Antrenarea modelului pe setul de date de antrenare. Frecvența cuvintele din e-mailurile spam și non-spam sunt înregistrate și numărate în dicționarele `spam_word_counts` și `non_spam_word_counts`.

3. **Predicție:** Implementarea funcției `predict(text)` are scopul de a calcula probabilitățile spam și non-spam pentru un text dat și de a face o predicție pe baza acestora. Aceasta utilizează probabilitățile calculate pentru cuvintele individuale din text și le multiplică pentru a obține o probabilitate globală pentru fiecare clasă.

Am realizat și o funcție `predict_with_laplace(text)` care aplică netezirea Laplace la calculul probabilităților, având în vedere cazul în care un cuvânt din textul dat nu a fost întâlnit în setul de antrenare, în cazul calculării acurateții la testare.

4. **Calcularea Ratei de Erori:** Funcția `calculate_error(test_data)` evaluează precizia modelului pe setul de date de testare și calculează rata de eroare.

5. **Cross-Validation Leave-One-Out:** Implementarea metodei `cross_validate()` are rolul de a evalua performanța modelului folosind tehnica de Cross-Validation Leave-One-Out. Acest proces presupune eliminarea, pe rând, a unei instanțe din setul de date de antrenare. Modelul este apoi antrenat și evaluat de mai multe ori, fiecare iterație având o altă instanță eliminată, iar instanța eliminată este utilizată ca set de testare. Restul instanțelor rămân în setul de antrenare. Având în vedere că rularea acestei funcții durează destul de mult, am realizat și o altă funcție `cross_validate_optimal()`, care optimizează procesul de cross-validation prin eliminarea repetării antrenării modelului pentru fiecare iterație, revenind la condițiile inițiale după fiecare evaluare a instanței de testare.

6. **Plotarea Rezultatelor Cross-Validation:** Funcția `plot_cross_validation_results()` generează un grafic pentru a vizualiza rezultatele obținute în timpul validării încrucișate.

7. **Evaluare pe Setul de Date de Testare și Antrenare:** Funcțiile `evaluate_on_test_data()` și `evaluate_on_train_data()` evaluează performanța modelului pe setul de date de testare, respectiv antrenare.

3.2. Motivare alegere din punct de vedere individual. În alegerea algoritmului Bayes Naiv pentru rezolvarea problemei clasificării e-mailurilor spam, s-au luat în considerare mai multe aspecte. Unul dintre motivele majore este simplitatea și eficiența acestui algoritm. Bayes Naiv este ușor de înțeles și implementat, ceea ce îl face o alegere potrivită pentru studii precum clasificarea email-urilor spam.

Alte motive includ versatilitatea sa și capacitatea de a gestiona eficient seturi de date cu un număr mare de caracteristici precum este setul nostru de date . În plus, presupunerea de independență a caracteristicilor poate să funcționeze surprinzător de bine în practică în ceea ce privește clasificarea e-mailurilor spam.

3.2.1. Bayes Naiv la nivel teoretic. La nivel teoretic, algoritmul Bayes Naiv se bazează pe Teorema lui Bayes și presupune independența condiționată a caracteristicilor, ceea ce înseamnă că probabilitatea unei anumite caracteristici este considerată independentă de celelalte caracteristici date clasa (spam sau non-spam). Chiar dacă această presupunere poate părea prea nesemnificativă, în realitate, în multe cazuri, Bayes Naiv se comportă surprinzător de bine și adesea depășește așteptările.

Modelul folosește probabilitățile a priori și a posteriori pentru a face predicții, oferind o abordare robustă pentru clasificarea e-mailurilor spam. Prin adaptarea frecvenței cuvintelor în setul de date de antrenare, Bayes Naiv estimează probabilitățile și face predicții bazate pe aceste estimări. Astfel, la nivel teoretic, algoritmul Bayes Naiv reprezintă o soluție eficientă pentru problema clasificării.

3.2.2. Bayes Naiv la nivel experimental. Am evaluat performanța algoritmului Bayes Naiv folosind setul de date Ling-Spam într-un context experimental. Mai jos sunt prezentate rezultatele obținute pentru diferite faze ale procesului:

1. **Evaluare pe setul de date de antrenare:** Am calculat eroarea de antrenare și am obținut o acuratețe de 99.78%, indicând faptul că modelul Bayes Naiv se adaptează bine la datele de antrenare și este capabil să învețe caracteristicile acestora.

2. **Leave-One-Out Cross-Validation (CVLOO):** Am efectuat validarea încrucișată utilizând tehnica Leave-One-Out. Eroarea medie la fiecare iterație a fost înregistrată, iar acuratețea medie a CVLOO a fost de 99.7%, care arată că modelul prezintă o bună generalizare și robustețe în fața schimbărilor în setul de date, oferind rezultate consistente în mai multe iterații ale modelului.

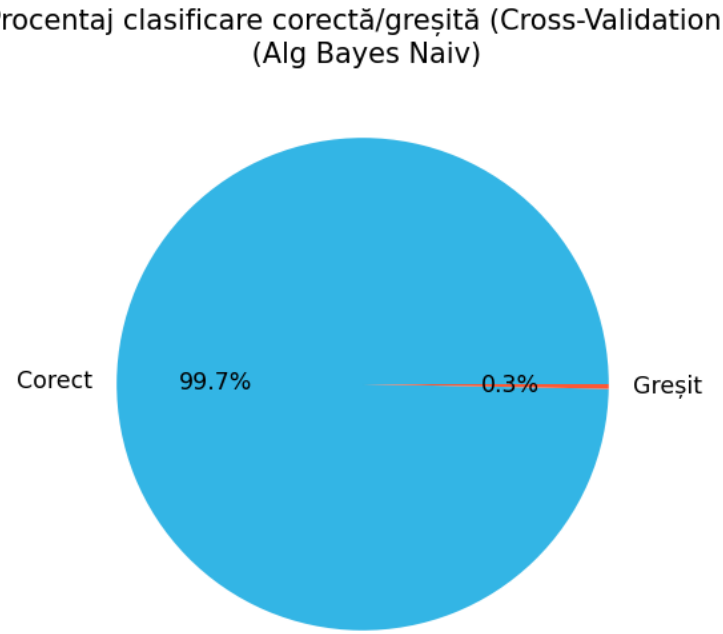

Figure 3: Grafic ilustrativ pentru vizualizarea procentajelor la Leave-One-Out Cross-Validation

3. **Evaluare pe setul de date de testare:** Am măsurat eroarea pe un set de date separat destinat testării modelului. Acuratețea obținută la acest nivel a fost de 93.13%. Chiar dacă acest rezultat este mai mic decât acuratețea obținută pe setul de antrenare și CVLOO, el indică o capacitate acceptabilă a modelului de a generaliza și de a face față unor date noi, nedând dovadă de overfitting.

   Figura 4 prezintă rezultatele instanțelor de test clasificate greșit folosind clasificatorul Bayes Naiv. Pe axa orizontală sunt reprezentate instanțele individuale, în timp ce pe axa verticală avem etichetele de clasificare atribuite de model (spam sau non-spam). Observăm că la testare toate email-urile No-Spam au fost clasificate corect iar doisprezece email-uri Spam au fost clasificate greșit.

   În Figura 5, se prezintă procentajele de clasificare a instanțelor de test. Observăm că clasificatorul Bayes Naiv demonstrează o performanță bună la testare, având un număr scăzut de instanțe clasificate greșit (aproximativ 1%).

În ansamblu, algoritmul Bayes Naiv se dovedește a fi un clasificator eficient în detecția spamului, oferind o combinație solidă de învățare, generalizare și capacitate de adaptare la date noi.

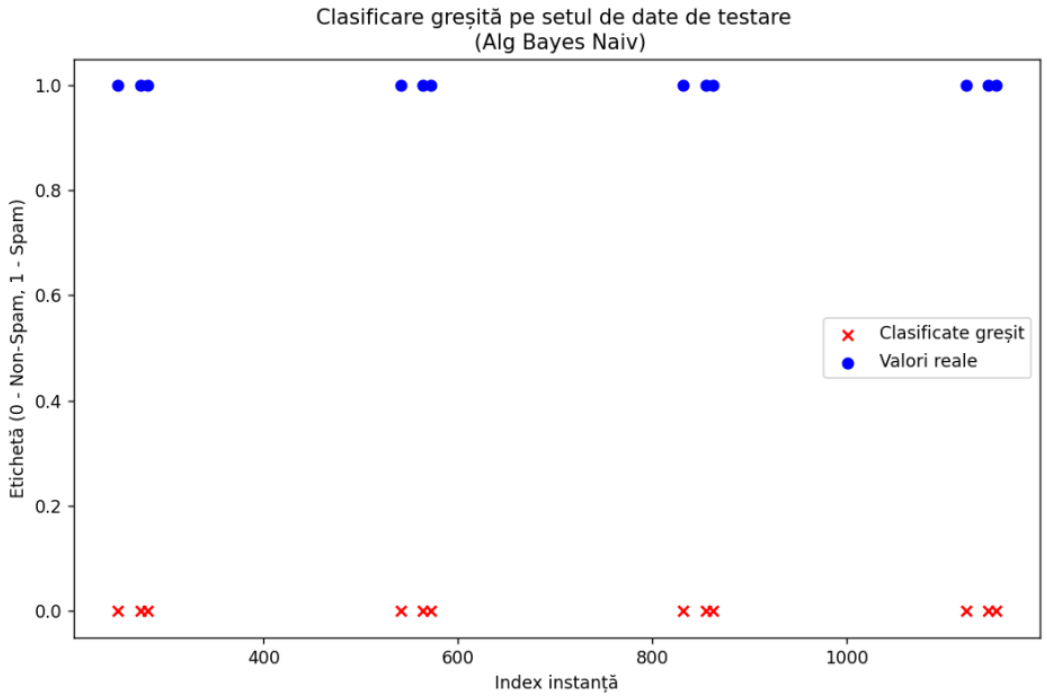

Figure 4: Grafic ilustrativ pentru vizualizarea instanțelor de test care au fost clasificate greșit

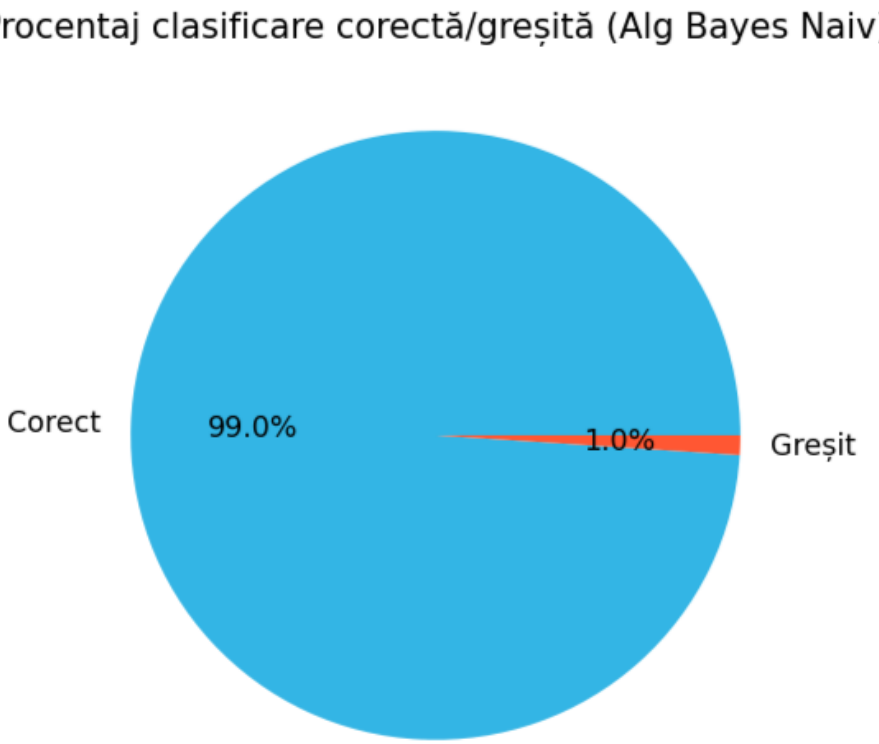

Figure 5: Grafic ilustrativ pentru vizualizarea instanțelor de test care au fost clasificate greșit

3.3. Motivare alegere prin comparație cu ceilalți algoritmi. Pentru rezolvarea acestei probleme de clasificare a emailurilor spam, am putea folosi și clasificatori precum ID3, Ada-Boost sau k-NN, deoarece:

1. **ID3:** Acest algoritm utilizează arbori de decizie pentru a organiza informațiile și lua decizii de clasificare pe baza caracteristicilor relevante.
2. **Ada-Boost:** Acest algoritm de învățare automată îmbunătățește performanța prin combinarea mai multor clasificatori slabi într-unul puternic.
3. **k-NN:** Acest algoritm utilizează distanța între exemplele de antrenament pentru a clasifica noi exemple în funcție de vecinătatea lor în spațiul caracteristicilor.

Alegerea dintre acești clasificatori poate depinde de specificul setului de date și de cerințele problemei. O analiză detaliată a performanței fiecărui algoritm în contextul particular al clasificării de e-mail spam ar putea orienta alegerea optimă a algoritmului potrivit pentru rezolvarea eficientă a problemei.

3.3.1. Comparație la nivel teoretic. În tabelul următor se poate observa o comparație teoretică între algoritmii Bayes Naiv, ID3, AdaBoost și k-NN.

Caracteristica	**Bayes Naiv (BN)**	**ID3**	**AdaBoost**	**k-NN**
Principiu de Lucru	Se bazează deciziile pe teorema lui Bayes și presupune independența atributelor.	Construiește un arbore de decizie pe baza informației câștigate în timpul construcției.	Combină multiple clasificatoare slabe pentru a forma unul puternic.	Clasifică bazându-se pe votul majoritar al vecinilor săi.
Gestionare a Variabilelor	Eficient în gestionarea variabilelor discrete.	Poate gestiona atât variabile discrete, cât și continue.	Necesită ajustări pentru a trata variabile discrete sau continue.	Poate întâmpina dificultăți în gestionarea unor dimensiuni ridicate ale datelor.
Adaptabilitate la Date Noi	Se poate să se adapteze bine la schimbările în distribuția cuvintelor cheie.	Poate fi sensibil la variația datelor și la cuvintele cheie noi.	Are potențial de adaptare prin adăugarea de clasificatoare slabe.	Poate întâmpina dificultăți cu cuvinte cheie rare sau variabilitatea datelor.
Interpretabilitate	Interpretare ușoară a rezultatelor bazate pe probabilități.	Arborele de decizie oferă o înțelegere clară și interpretabilă.	Complexitate crescută, interpretarea poate fi dificilă.	Simplu și ușor de înțeles, dar interpretarea poate fi dificilă în cazul unor seturi de date mari.
Rezistență la Zgomot	Rezistent la zgomot și la cuvinte cheie rare.	Poate fi sensibil la date zgomotoase sau la cuvinte cheie rare.	Rezistent la zgomot datorită combinației de clasificatoare slabe.	Sensibil la date zgomotoase sau la variabilitatea datelor.
Performanța Generală	Bine adaptat la clasificarea textelor și a e-mailurilor spam.	Bine adaptat la seturi de date omogene și cu variabile categorice.	Poate îmbunătăți performanța generală prin agregarea de clasificatoare slabe.	Performanța depinde de alegerea metricii de distanță și a valorii lui k.

Table 1: Comparatie Teoretica intre Algoritmi de Clasificare

3.3.2. Comparație la nivel experimental. Am evaluat performanța celorlalți algoritmi de clasficare folosind setul de date Ling-Spam și librăria `sklearn` într-un context experimental. Mai jos sunt prezentate rezultatele obținute:

Algoritm	**Acuratețe Antrenare (%)**	**Acuratețe CVLOO (%)**	**Acuratețe testare (%)**
Bayes Naiv (BN)	99.78	99.7	93.13
ID3	100	99.99	93.99
AdaBoost	99.77	99.76	98.11
k-NN	99.65	99.98	92.44

Table 2: Rezultate Experimentale pentru diferiți algoritmi de clasificare

Analiza Algoritmilor:.

1. **Bayes Naiv (BN):**
   - Rezultate consistente și competitive pe toate cele trei seturi de date.
   - Acuratețea la antrenare și CVLOO este similară, indicând o bună capacitate de generalizare.
   - Acuratețea la testare este cu puțin mai mică, dar rămâne la un nivel înalt, sugerând robustețe.
2. **ID3:**
   - Diferența semnificativă dintre acuratețea la antrenare și cea la testare indică un posibil scenariu de overfitting.
3. **AdaBoost:**
   - Diferența redusă între acuratețea la antrenare și CVLOO indică o bună generalizare, iar acuratețea la testare rămâne la un nivel excelent.
4. **k-NN:**
   - Diferența semnificativă dintre acuratețea la antrenare și cea la testare indică o posibilă problemă de overfitting.

Analiza Algoritmilor la Cross-Validation:. Graficele cu fold-uri (5-Fold Cross-Validation) ilustrează clar discrepanțe în performanța algoritmilor. ID3, AdaBoost și k-NN au înregistrat o frecvență semnificativă de clasificări eronate, sugerând sensibilitatea lor la variabilitatea datelor. În contrast, Bayes Naiv a demonstrat o consistență mai bună și o rată mai redusă a clasificărilor greșite, indicând o adaptabilitate superioară.

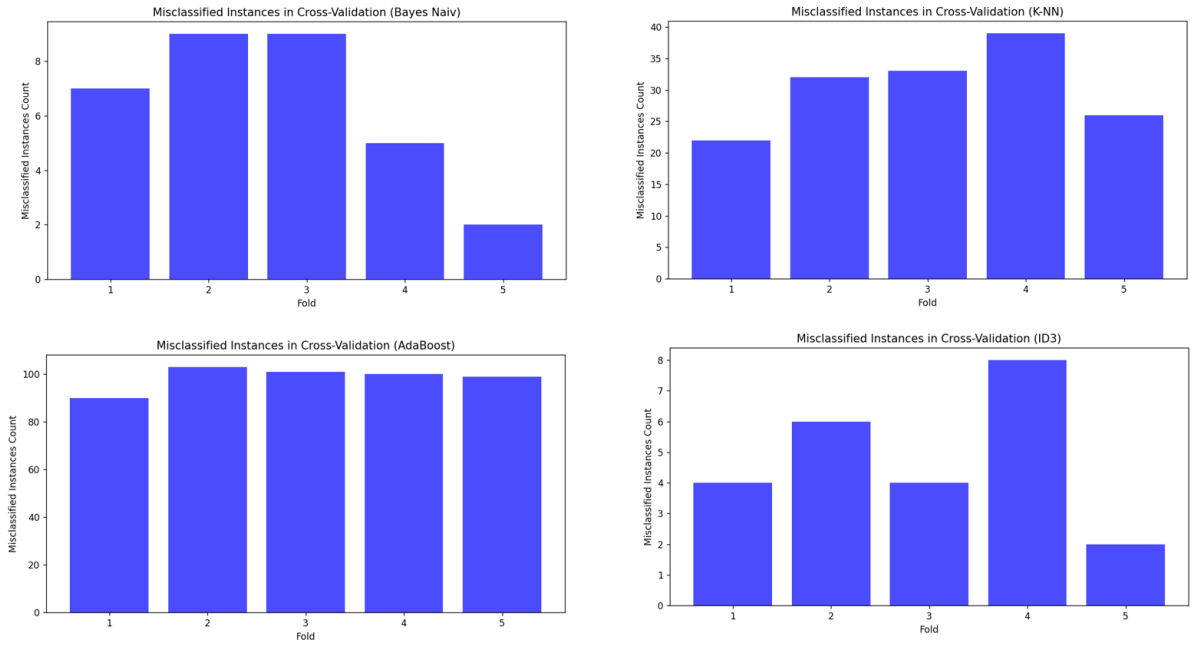

Figure 6: Ilustrare număr de instanțe clasificate greșit la 5-Fold Cross-Validation

Analiza Algoritmilor la Testare:. Observăm din grafice din Figura 7 și Figura 8 , indiferent de contextul de evaluare, algoritmul Bayes Naiv se distinge prin cel mai bun procentaj de clasificare corectă. Acest rezultat sugerează că, în diversele condiții testate, Bayes Naiv a fost cel mai robust și precis în identificarea corectă a instanțelor de date. În comparație, algoritmii ID3, AdaBoost și k-NN au prezentat procentaje mai scăzute de clasificare corectă, indicând o performanță relativ mai redusă în contextul evaluării.

4 Concluzii

În comparație cu alți algoritmi evaluați, precum k-NN, ID3 și AdaBoost, Bayes Naiv se evidențiază ca un clasificator deosebit în detectarea spamului. În timp ce k-NN și ID3 au prezentat semne de overfitting, Bayes Naiv a demonstrat o capacitate mai bună de generalizare și stabilitate în contextul diverselor seturi de date. AdaBoost, deși a obținut rezultate bune, a avut o diferență mai mare între acuratețea la antrenare și la CVLOO, sugerând o potențială sensibilitate la variabilitatea datelor.

În contrast, Bayes Naiv a prezentat consistență în performanță, indiferent de contextul evaluării. Cu o adaptabilitate superioară și precizie în clasificarea mesajelor de tip spam, Bayes Naiv se impune ca o alegere robustă și fiabilă în comparație cu k-NN, ID3 și AdaBoost pentru această sarcină specifică.

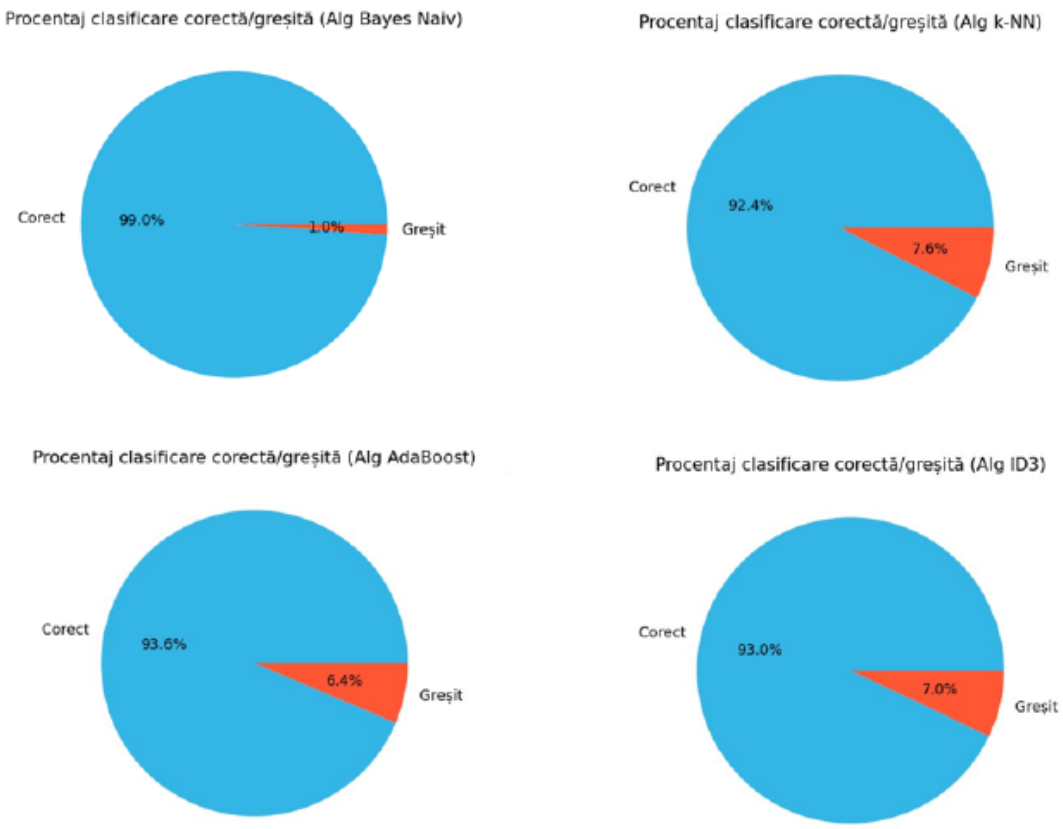

Figure 7: Ilustrare procentaj clasificare corectă/greșită la testare

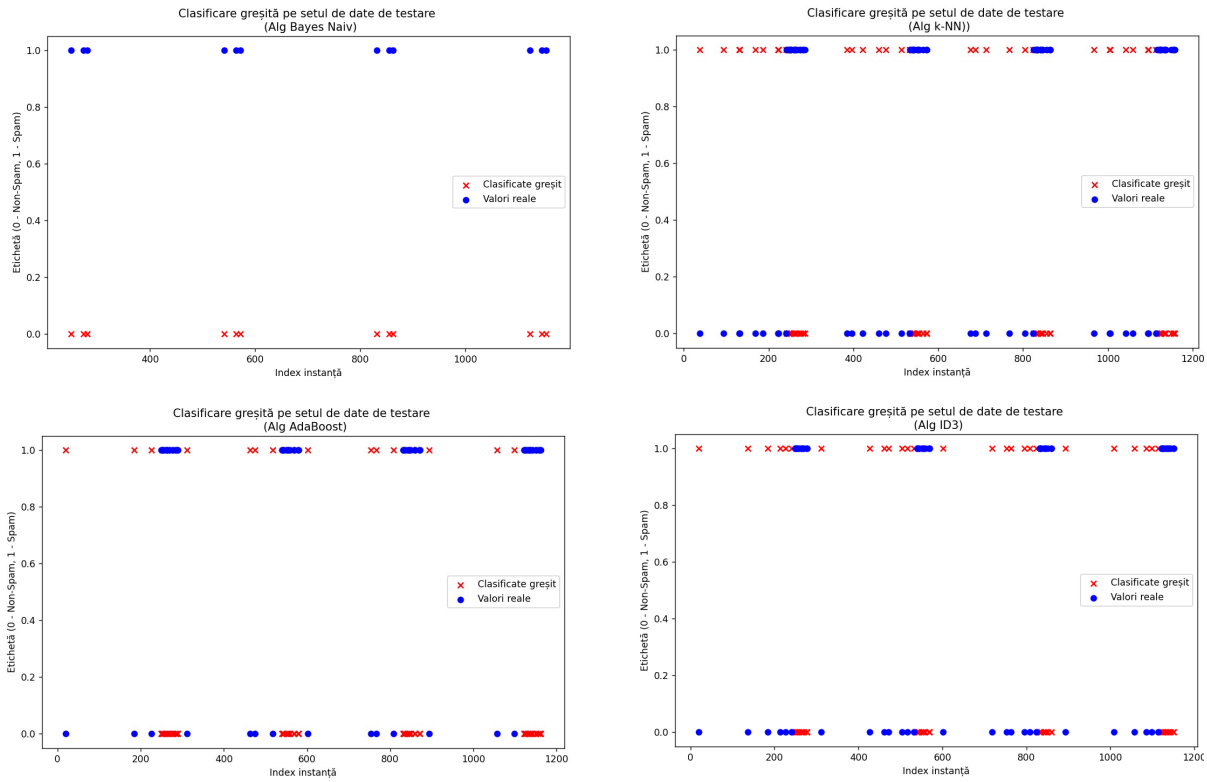

Figure 8: Ilustrare instanțe de test care au fost clasificate greșit