



UNIVERSITATEA DIN  
BUCUREȘTI

FACULTATEA DE  
MATEMATICĂ ȘI  
INFORMATICĂ



SPECIALIZAREA MATEMATICĂ-INFORMATICĂ

Lucrare de licență

# CLASIFICAREA POSTĂRIILOR SENZAȚIONALISTE PE REȚELELE SOCIALE

Absolvent

Dragoș-Ștefan Mihalcea

Coordonator științific

Lect. Dr. Sergiu Nisioi

București, iunie 2023

## Rezumat

În sfera rețelelor de socializare, se poate observa o prezență semnificativă a unor postări create strategic pentru a captiva cititorii și a-i redirecționa către anumite pagini web. Cunoscut sub termenul de "clickbait", această practică predominantă se bazează în mare măsură pe utilizarea formulărilor senzaționaliste, concepute cu pricepere pentru a stârni curiozitatea persoanelor și a le încuraja să exploreze mai departe. În mod frecvent, fluxul excesiv de informații prin intermediul postărilor de tip clickbait pe rețelele de socializare prezintă riscuri semnificative pentru sănătatea mentală, precum și riscuri suplimentare asociate cu răspândirea informațiilor înșelătoare în scopul unor câștiguri financiare sau al unor motivații politice îndoielnice. Contribuția mea la acest subiect a rezultat din participarea activă la sarcina 5 a competiției SemEval-2023: Clickbait Spoiling. Participarea mea s-a concentrat în mod specific pe sub-sarcina care implică identificarea diferitelor tipuri de spoilere a titlurilor clickbait, unde am realizat o analiză comparativă a mai multor abordări de învățare automată și învățare profundă.

## **Abstract**

Within the realm of social media, a significant presence of posts can be observed, strategically crafted to captivate readers and redirect their attention towards specific web pages. Termed as "clickbait", this prevailing practice relies heavily on the deployment of sensationalistic formulations, skillfully designed to pique the curiosity of individuals and entice them to explore further. Frequently, the excessive influx of information through clickbait posts on social media presents substantial hazards to mental well-being, as well as additional risks associated with the dissemination of misleading information for financial gain or questionable political motives. My contribution to this subject stemmed from my active participation at SemEval-2023 task 5: Clickbait Spoiling. My participation was specifically focused on the sub-task that involved identifying various types of clickbait spoiling, where I conducted a comparative analysis of several machine learning and deep learning approaches.

# Cuprins

<b>1</b>	<b>Introducere</b>	<b>5</b>
<b>2</b>	<b>Setul de date</b>	<b>8</b>
<b>3</b>	<b>Metodologie</b>	<b>12</b>
3.1	Metode de extragere a caracteristicilor . . . . .	12
3.2	Metode de învățare automată . . . . .	13
3.2.1	Mașini cu vectori suport . . . . .	13
3.2.2	Amplificarea gradientului . . . . .	15
3.2.3	Modele de transformeri pe text . . . . .	17
3.3	Metrici . . . . .	19
3.3.1	Matricea de confuzie . . . . .	19
3.3.2	Acuratețea balansată . . . . .	19
3.3.3	Coeficientul de corelație al lui Matthew . . . . .	19
3.4	Strategii de validare . . . . .	21
3.4.1	Validarea încrucișată . . . . .	21
3.4.2	Divizarea Antrenare-Validare . . . . .	22
<b>4</b>	<b>Rezultate și contribuții</b>	<b>23</b>
4.1	Experimente cu mașini cu vectori suport . . . . .	24
4.2	Experimente cu amplificarea gradientului . . . . .	26
4.3	Experimente cu transformeri pe text . . . . .	28
<b>5</b>	<b>Concluzii și activități viitoare</b>	<b>30</b>
	<b>Bibliografie</b>	<b>31</b>

# Capitolul 1

## Introducere

Clickbait-ul se referă la o tehnică utilizată în crearea conținutului online, în special în titluri sau antete, pentru a atrage atenția utilizatorilor și a-i încuraja să dea click pe un link sau să se angajeze cu conținutul. Termenul "clickbait" este o combinație a cuvintelor "click" (referindu-se la acțiunea de a apăsa pe un link) și "bait" (referindu-se la atragerea sau tentarea cuiva).

Deși clickbait-ul poate fi eficient în captarea atenției, este adesea criticat pentru că poate fi înșelător, manipulator sau poate să exagereze conținutul real. Stârnindu-le interesul prin acest format de titlu, utilizatorii accesează diverse site-uri așteptând un anumit tip de informație sau experiență, doar pentru a fi dezamăgiți sau înșelați atunci când conținutul nu îndeplinește promisiunea inițială. În general, se observă că titlurile senzaționaliste afectează cititorii în mod diferit față de titlurile tradiționale de știri. Potrivit modelului de comunicare cognitiv-afectiv, în timp ce calitatea cunoștințelor percepute este mai ridicată în titlurile tradiționale de știri decât în titlurile clickbait; nivelurile factorilor afectivi, cum ar fi stimularea și curiozitatea în titlurile clickbait, sunt de asemenea mai ridicate decât în titlurile tradiționale de știri [14].

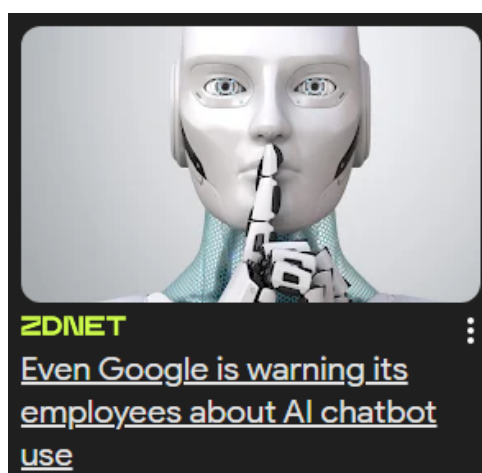


Figura 1.1: Exemplu de titlu senzaționalist [6]

În ultimii ani s-au făcut progrese semnificative în combaterea titlurilor clickbait prin crearea de detectoare folosind multiple abordări care au la bază tehnici de inteligență artificială, în deosebi modele de clasificare. Deși s-au înregistrat anumite progrese în această direcție, presa și creatorii de conținut își schimbă strategiile continuu și vor găsi o modalitate de a ocoli clasificatoarele. Prin urmare, ar trebui creată o metodă robustă care să poată prezice viitoare modificări ale procedurii de detectare [18].

O posibilă soluție pentru a preveni accesarea conținutului de tip înșelător ar putea apărea prin conceptul de "clickbait spoiling". Acesta implică reformularea titlurilor senzaționaliste originale sau introducerea de conținut suplimentar în postarea inițială cu scopul de a limita curiozitatea cititorilor.

Există o abordare potențială pentru implementarea spoiling-ului care poate fi divizată în doi pași:

1. identificarea tipului de informație necesară pentru a reduce curiozitatea articolului sau postării de pe social media.
2. folosirea tipului de informație aflat la pasul anterior și conținutului articolului sau postării pentru a genera un nou conținut ce va fi considerat drept spoiler


Clickbait tweet	Spoiler
 <b>Lifehacker</b>  @lifehacker How to keep your workout clothes from stinking: <a href="https://lifehack.kr/57Y0uEZ">lifehack.kr/57Y0uEZ</a>	"washing [them]"
 <b>New York Post</b>  @nypost Just how safe are NYC's water fountains? <a href="https://nyp.st/2yHSGnr">nyp.st/2yHSGnr</a>	"The Post independently tested eight water fountains in New York City's most frequented parks, and found that all met or exceeded the state's guidelines for water quality."
 <b>CNBC</b>  @CNBC A Harvard nutritionist and brain expert says she avoids these 5 foods that "weaken memory and focus." (via @CNBCMakeIt) <a href="https://cnb.cx/2TG6zeX">cnb.cx/2TG6zeX</a>	"1. Added sugar" [...] "2. Fried foods" [...] "3. High-glycemic-load carbohydrates" [...] "4. Alcohol" [...] "5. Nitrates" [...]

Figura 1.2: Exemple de spoiling pentru tweet-uri clickbait [4]

Recent, au fost investigate mai multe metode care implică identificarea mai multor tipuri de spoilere și, apoi, generarea unui conținut special în vederea reducerii curiozității induse de clickbait-uri de către *Hagen* și *Froebe* [12, 8] și în cadrul SemEval 2023 Task 5: *Clickbait Spoiling* [7].

Participarea mea [16] la această sarcină se rezumă doar la primul pas din abordarea mai sus menționată, mai precis clasificarea diferitelor tipuri de spoiler pentru articolele senzationaliste. În continuare, voi prezenta o analiză calitativă și cantitativă a datelor, precum și o comparație între diferite metodologii de clasificare.

# Capitolul 2

## Setul de date

În cadrul acestei lucrări voi clasifica postările senzaționaliste în funcție de diverse tipuri de spoilere din componența acestora. În conformitate cu organizatorii SemEval ([12, 8]) această tipologie cuprinde următoarele trei clase:

1. spoilere de tip *phrase* - constau într-un singur cuvânt sau o frază formată din mai multe cuvinte, extrase din documentului asociat.

Clickbait	Spoiler
THIS IS NOT A DRILL: you could work for Kate Middleton	housekeeper
Easily Stressed People Are Mentally Healthier After Writing This Down. See How The Trick Works.	own personal values

Tabela 2.1: Exemple de tip *phrase*

2. spoilere de tip *passage* - sunt alcătuite dintr-o secvență mai lungă de cuvinte sau dintr-un șir de câteva propoziții din documentul asociat.

Clickbait	Spoiler
Tom Hiddleston finally reveals the truth about his romance with Taylor Swift	"That's the truth. It's not a publicity stunt."
An unbearable provocation.	An Afghan migrant attacked a woman at an asylum centre in Austria because she was reading a bible

Tabela 2.2: Exemple de tip *passage*



3. spoilere de tip *multi* - cuprind mai multe fraze sau pasaje non-consecutive din document.

Clickbait	Spoiler
The Fastest Growing Economy in the World! Number 4 Will Shock You	1. India 2. Bangladesh 3. China 4. Indonesia 5. Pakistan
Unarmed teen shot dead by police on university campus	Tyler Comstock Iowa State University

Tabela 2.3: Exemple de tip *multi*

Setul de date este compus din 4000 de articole senzaționaliste, formate din 2 sub-seturi: cel de antrenare care cuprinde 3200 de exemple și cel validare care cuprinde 800 de exemple; procurate în urma participării la sarcina de *Clickbait Spoiling* în cadrul SemEval 2023 [7]. În tabelele 2.1, 2.2 și 2.3 sunt prezentate câteva exemple de titluri senzaționaliste împreună cu spoilerele corespunzătoare. Cele de tip *phrase* sunt caracterizate prin lungimea scurtă (de câteva cuvinte), comparativ cele de tip *passage* care se află la polul opus, remarcându-se prin fraze lungi. Spoilerele *multi* prezintă similarități cu ambele clase menționate anterior conținând mai multe *fraze* sau *pasaje*, dar se diferențiază de acestea prin lipsa de cursivitate.

class	#examples	#sentences	avg_sents	std_sents	#tokens	#unq_tokens
phrase	1702	37k	21.5	21.1	706k	350k
passage	1596	45k	27.9	89.1	814k	380k
multi	702	29k	41.9	43.0	529k	227k

Tabela 2.4: Statisticile setului de date compute pe datele publice de antrenare/validare din timpul competiției. Împărțirea în propoziții s-a făcut folosind spacy și au fost luate drept token doar șirurile de caractere alfa-numerice.

În tabelul 2.4 sunt prezentate câteva statistici ale setului de date obținute pe datele de antrenare/validare procurate în timpul competiției. Pentru a măsura numărul de propoziții, media propozițiilor per document și numărul de simboluri (cu spacy [13]) am folosit doar conținutul propriu-zis al articolelor. Scopul acestora este de a identifica diferențe structurale și dezechilibre între clase.

Se poate observa în tabelul 2.4 că în majoritatea exemplurilor este nevoie de o *frază* sau un *pasaj* drept spoiler, acestea fiind în număr de măcar doua ori mai mare decat cele de tip *multi*. Totodată, se poate remarca faptul că exemplele din clasa *passage* sunt mai lungi și conțin mai multe propoziții decât cele din clasa *phrase* prin prisma numărului mai mare de propoziții totale, propoziții per articol și caractere.

## Global Topic Representation

- 0\_trump\_donald\_presidential\_trumps
- 1\_restaurants\_restaurant\_meal\_food
- 2\_vogue\_fashion\_dress\_lingerie
- 3\_iphone\_iphones\_smartphone\_smartphones
- 4\_gta\_nintendo\_gaming\_wii
- 5\_criminal\_crimes\_arrest\_crime
- 6\_diet\_diabetes\_healthy\_obesity
- 7\_rappers\_rapper\_rap\_kanye
- 8\_marvel\_marvels\_spiderman\_avengers
- 9\_nfl\_inning\_baseball\_patriots
- 10\_clickbait\_facebook\_facebooks\_headline...
- 11\_pets\_dogs\_breeds\_vet
- 12\_cosmetics\_makeup\_skincare\_lipstick

Figura 2.1: Culoarea și cuvintele cheie fiecărui subiect.

O analiză realizată cu biblioteca BERT-topic [11] asupra celor mai comune subiecte din fiecare clasă ar putea arăta cum modelele percep distribuția fiecărui tip de conținut din fiecare document. În figura 2.1 sunt prezentate cuvintele-cheie pentru fiecare temă, în timp ce în figura 2.2 sunt prezentate distribuțiile acestor teme. Primele trei subiecte legate de Donald Trump, restaurante și modă au distribuții asemănătoare. Principalele diferențe sunt în clasa *multi* care sunt prezente mai multe subiecte legate de sănătate și din care lipsesc complet subiectele legate animale de companie (subiectul 11).

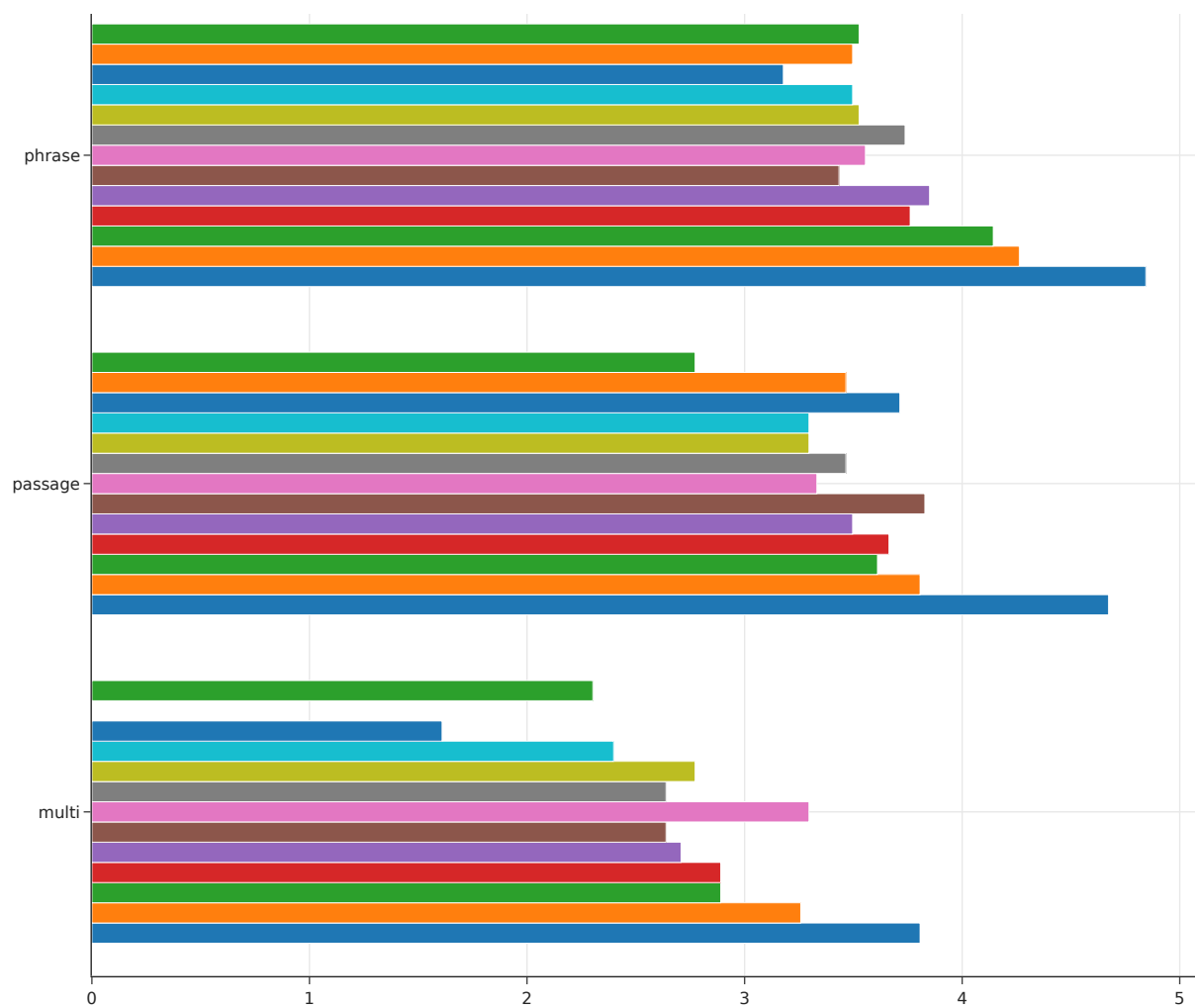


Figura 2.2: Distributia subiectelor în clase

# Capitolul 3

## Metodologie

### 3.1 Metode de extragere a caracteristicilor

În aceasta secțiune voi vorbi despre cele două metode extragere a caracteristicilor folosite:

1. spacy (*en\_core\_web\_lg*) [13]
2. TF-IDF [21]

Spacy este o bibliotecă pentru procesarea limbajului natural în Python ce conține modele pre-antrenate pentru a crea vectori de caracteristici din cuvinte sau documente. În experimentele încercate am folosit *en\_core\_web\_lg* ce conține 514 mii de vectori unici cu 300 de dimensiuni. Astfel, am extras pe rând caracteristici pentru fiecare text dintr-un document, post-document și titlu.

TF-IDF este un vectorizator de text care, după cum sugerează și numele, combină două concepte: frecvența termenilor și frecvența documentelor. Acesta reprezintă și o metodă de a măsura originalitatea unui termen, comparând numărul de apariții al unui cuvânt într-un document cu numărul de documente în care acel cuvânt apare.

$$TF(t, d) = \frac{\text{numărul de apariții al lui } t \text{ în } d}{\text{numărul total de termeni din } d}$$

$$IDF(t) = \log \frac{N}{1 + df}$$

$$TF - IDF(t, d) = TF(t, d) \cdot IDF(t)$$

$d$  - documentul ales,  $N$  - numărul total de documente,  $df$  - numărul de documente cu termenul  $t$

În experimentele încercate am ales (asemănător ca la spacy) să creez vectori de caracteristici pentru fiecare document, post-document și titlu al articolelor din setul de

date. Am folosit implementarea din *scikit-learn* [17] pentru *TfidfVectorizer*, alegând sa fie extrase unigrame si bigrame pentru calcularea frecvențelor.

## 3.2 Metode de învățare automată

### 3.2.1 Mașini cu vectori suport

Mașini cu vectori suport sau SVM (Support Vector Machines) sunt algoritmi de învățare supervizată al căror scop este găsirea unui hiperplan n-dimensional (n-numărul de caracteristici) optim care separă în două clase.

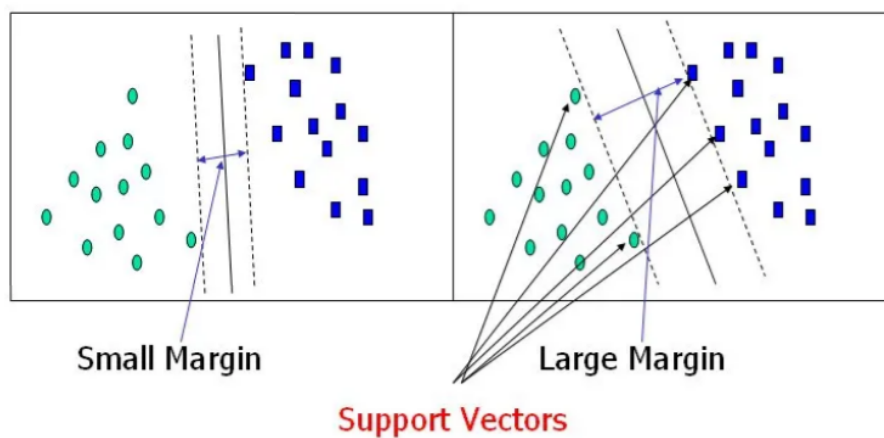


Figura 3.1: Găsirea hiperplanului ideal [9]

După cum se poate vedea și în figura 3.1, hiperplanul trebuie să fie plasat astfel încât distanța dintre el și vectorii suport să fie maximă. Vectorii suport sunt cele mai apropiate puncte de date față de hiperplan.

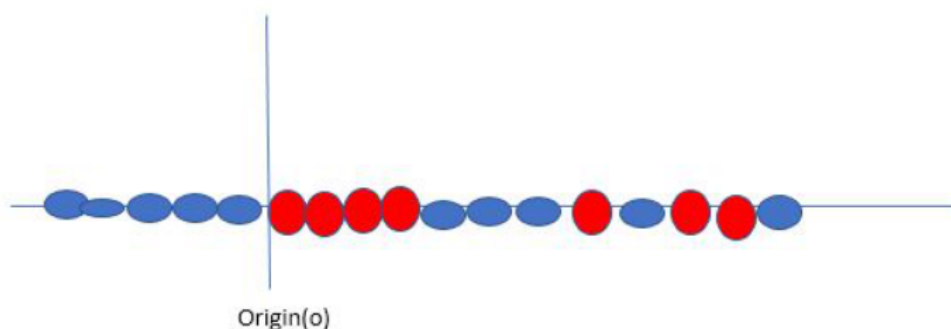


Figura 3.2: Plan în care punctele nu pot fi separate liniar [10]

În figura 3.2 este prezentat un scenariu poate fi rezolvat cu SVM. Toate punctele de date se află pe o singură axă (unidimensionale) și nu pot fi separate printr-o funcție liniară. În acest caz, nucleul generează pentru fiecare punct  $x_i$  un punct  $y_i$  prin intermediul unei

funcții (în cazul de față funcția distanță față de origine) și proiectează noua pereche  $(x_i, y_i)$  într-un nou plan bidimensional. Astfel, după cum se poate observa și în figura 3.3, acum se poate găsi un hiperplan liniar care să separe cele 2 clase.

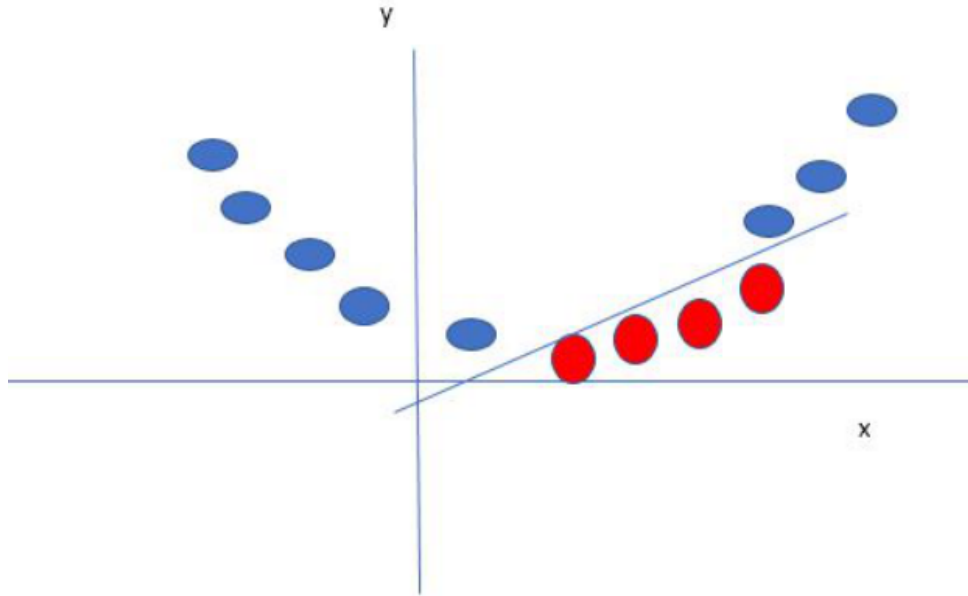


Figura 3.3: Plan 2D în care punctele pot fi separate [10]

În experimentele încercate am folosit implementare din *scikit-learn* [17] pentru algoritmi *SVC* și *LinearSVC* [23]. Acestea au fost în număr de douăsprezece și pot fi împărțite în două categorii în funcție de metoda de extragere a vectorilor de caracteristici folosită:

1. vectori generați cu *spacy*

Au avut loc șase experimente, fiecare folosind drept date de antrenare 3200 (datele de antrenare SemEval) de vectori cu 300 de dimensiuni extrase din document, post-document sau titlu.

2. vectori generați folosind unigrame și bigrame

În mod asemănător au avut loc șase experimente, însă de această dată fiecare din ele au folosit drept date de antrenare 3200 de vectori cu dimensiuni variabile pentru document (685k), post-document(32k) și titlu (28k).

Cu toate că inițial am folosit drept date de validare 800 de articole (datele de validare SemEval) care au fost vectorizate prin ambele metode descrise mai sus, am implementat validarea încrucișată pentru fiecare experiment în parte. Astfel, am folosit toate cele 4000 de articole pe rând atât ca date de antrenare, cât și de validare.

### 3.2.2 Amplificarea gradientului

Amplificarea gradientului este un algoritm de învățarea automată care combină modele mai slabe într-un singur model mai bun care are drept scop minimizarea funcției de pierdere (precum eroarea pătrată medie, entropia încrucișată sau verosimilitatea) a modelului anterior prin coborârea de gradient.

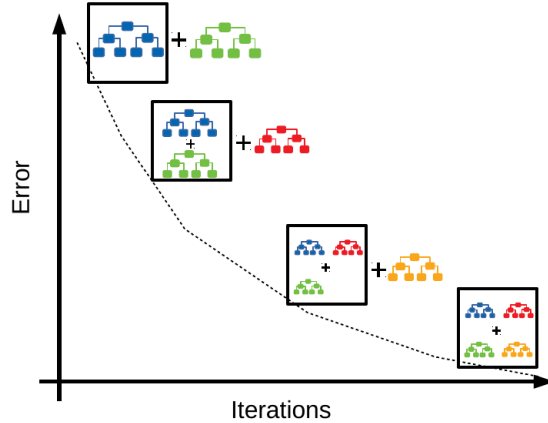


Figura 3.4: [22]

La fiecare iterație, algoritmul calculează gradientul funcției de pierdere având în vedere predicțiile ansamblului curent și apoi antrenează un nou model slab pentru a minimiza acest gradient. Predicțiile noului model sunt apoi adăugate în ansamblu, iar procesul este repetat până un anumit criteriu este îndeplinit. Cele mai importante avantaje ale acestei metode sunt:

1. are performanțe foarte bune și o acuratețe ridicată
2. nu necesită multă pre-procesare, descurcându-se bine atât cu date numerice, cât și cu date categorice
3. gestionează bine lipsa de date, nefiind nevoie de intervenție

În continuare voi prezenta un pseudocod al acestui algoritm. Pentru acesta voi considera drept date de antrenare un set  $\{(x_i, y_i)\}_{i=1}^n$ , o funcție de pierdere diferențibilă  $L(y, F(x))$  și un număr de iterații  $M$ .

La pasul 1, modelul este inițializat cu o valoare constantă.

$$F_0(x) = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, \gamma)$$

La pasul 2, se parcurg cu o structură repetitivă numerele de la  $m = 1$  la  $M$  și au loc următoarele:

1. Se calculează reziduurile:

$$r_{im} = - \left[ \frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)}$$

pentru  $i = 1, \dots, n$ .

2. Se antrenează un model slab sau de bază  $h_m(x)$  cu setul  $\{(x_i, r_{im})\}_{i=1}^n$
3. Se calculează coeficientul  $\gamma_m$  rezolvând următoarea ecuație:

$$\gamma_m = \arg \min_{\gamma} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

4. Se actualizează modelul:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

La ultimul pas, voi avea ca rezultat modelul final:  $F_M(x)$ .

În propriile experimente am folosit implementare din biblioteca *XGBoost* [3] pentru algoritmul *XGBClassifier*. Asemănător metodei cu mașini cu vectori suport, am desfășurat douăsprezece experimente care pot fi împărțite în același fel în funcție de metoda de extragere a vectorilor de caracteristici folosită, doar că înainte de a fi servite modelului, clasele au fost encodate cu cifre de la 0 la 2.



### 3.2.3 Modele de transformeri pe text

BERT (Bidirectional Encoder Representations from Transformers) [5] este un model bazat pe *Transformers* [24], un mecanism de atenție ce învață relațiile contextuale între cuvinte (sau grupuri de litere) dintr-un text. În varianta de bază, un *Transformer* conține două mecanisme separate:

- un *encoder* care citește textul introdus
- un *decoder* care produce o predicție pe o anumită temă

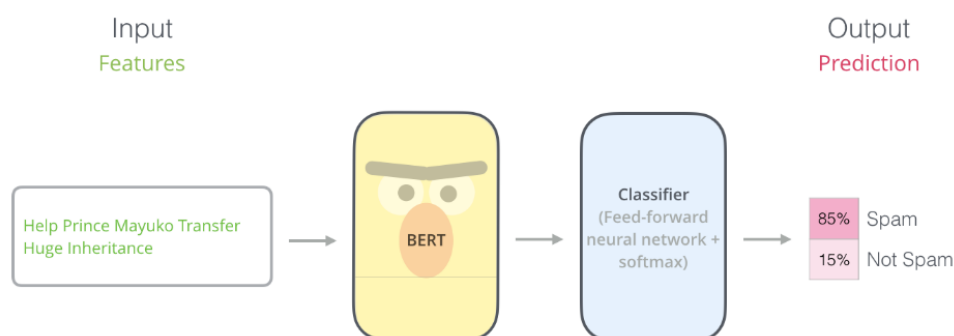


Figura 3.5: Exemplu de clasificare cu BERT [2]

Întrucât scopul lui BERT este să genereze un model de limbaj, doar *encoder-ul* este necesar. Inovația adusă de acest model constă în faptul că, textul introdus este citit de encoder deodată, într-o singură secvență de cuvinte. Astfel, modelul poate învăța contextul unui cuvânt bazându-se pe toate împrejurimile sale (la stânga și la dreapta cuvântului). Aceasta abordare este în mod contrar modelelor direcționale, care citesc textul introdus în mod secvențial (de la stânga la dreapta, sau invers).

Un mod simplist pentru a explica cum funcționează un *encoder* ar fi următorul:

- Conținutul introdus este o secvență de caractere, care sunt întâi încorporate în vectori și apoi procesate de rețeaua neuronală.
- Rezultatul produs este o secvență de vectori, în care fiecare corespunde unui caracter introdus ce are același index.

Multe modele lingvistice (în general direcționale) au drept scop prezicerea următorului cuvânt într-o secvență, o abordare ce restricționează învățarea bazată pe context. Pentru a depăși acest impediment, BERT folosește două tehnici de antrenare:

- MLM (Masked-Language Modeling)

Înainte ca datele să fie introduse în model, 15% din cuvintele din secvență sunt înlocuite cu un simbol pe post de mască. Apoi modelul încearcă să prezică valorile

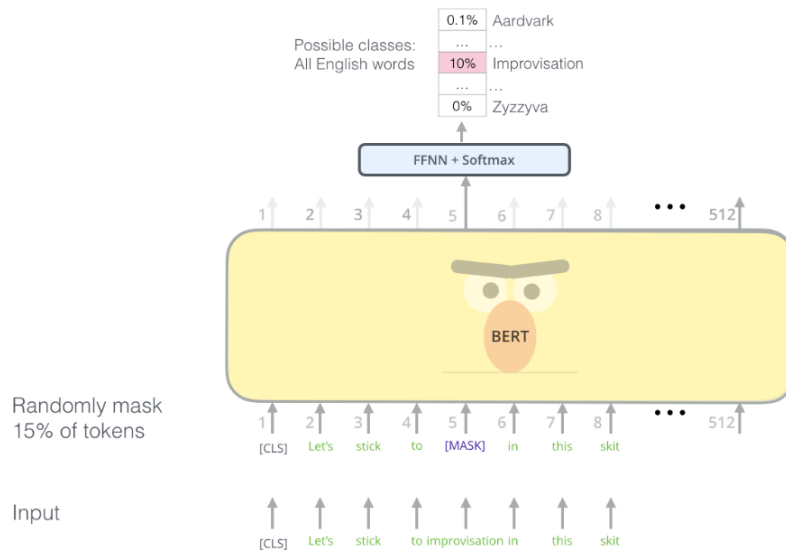


Figura 3.6: MLM [2]

originale ale cuvintelor mascate, bazându-se pe contextul oferit de cuvintele din împrejurimile sale.

- NSP (Next Sentence Prediction)

În procesul de antrenare, modelul primește perechi de propoziții drept date de intrare și încearcă să prezică dacă cea de a doua propoziție este consecutivă primeia în documentul original. În timpul antrenării jumătate din perechi vor conține două propoziții consecutive, în timp ce cealaltă jumătate va conține propoziții alese aleator. Presupunerea este că propozițiile alese aleator nu sunt conectate.

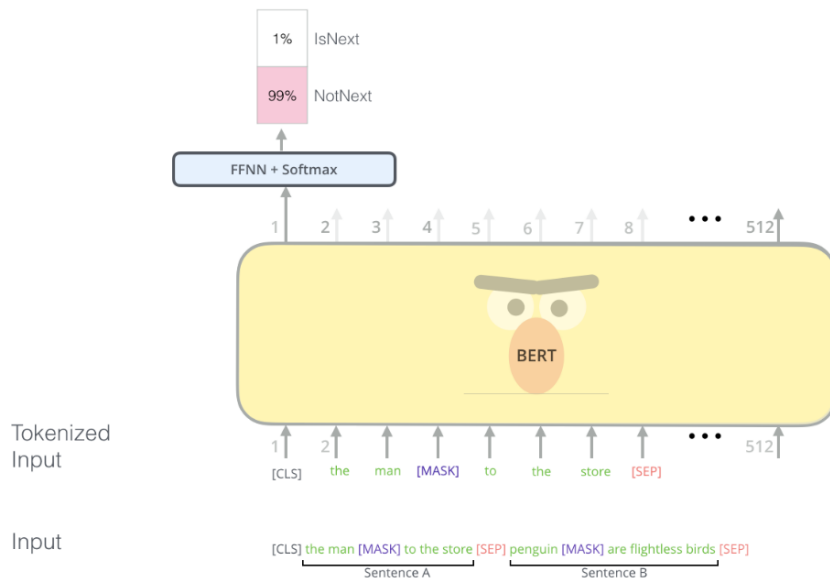


Figura 3.7: NSP [2]

## 3.3 Metrice

### 3.3.1 Matricea de confuzie

Matricea de confuzie este o matrice pătratică folosită pentru a măsura performanța unui algoritm de clasificare. Fiecare rând reprezintă instanțele dintr-o clasă reală, în timp ce fiecare coloană reprezintă instanțele dintr-o clasă prezisă. Pe prima diagonală a acesteia se află instanțele prezise corect în clase.



		Expected			
		1	2	3	4
Predicted	1	52	3	7	2
	2	2	28	2	0
	3	5	2	25	12
	4	1	1	9	40

Figura 3.8: Un exemplu de matrice de confuzie [15]

### 3.3.2 Acuratețea balansată

Acuratețea balansată este o metrică folosită pentru a măsura performanța unui model de clasificare. Aceasta este folosită când setul de date nu este balansat iar unele clase au substanțial mai multe instanțe decât altele.

Formula de calcul generalizată pentru o repartiție în  $K$  clase este:

$$bacc = \frac{\sum_k^K \frac{c_k}{t_k}}{K}$$

- $c_k$  - numărul de predicții corecte pentru clasa  $k$
- $t_k$  - numărul total de instanțe din clasa  $k$

### 3.3.3 Coeficientul de corelație al lui Matthew

Coeficientul de corelație al lui Matthew sau MCC(Matthew Correlation Coefficient) este o metrică statistică menită să măsoare mai bine performanța unui model în cazul unor clase dezechilibrate. Formula generalizată pentru calcularea coeficientului de corelație cu  $K$  clase este următoarea [1]:

$$MCC = \frac{c \times s - \sum_k^K p_k \times t_k}{\sqrt{(s^2 + \sum_k^K p_k^2) \times (s^2 + \sum_k^K t_k^2)}}$$

- $t_k$  - numărul de instanțe pentru clasa  $k$
- $p_k$  - numărul de predicții pentru clasa  $k$
- $c$  - numărul de instanțe prezise corect
- $s$  - numărul total de experimente

Acești valori sunt între  $[-1, 1]$  pentru cazul binar, dar dacă există mai mult de două clase, limita inferioară va fi între  $[-1, 0]$ . Un rezultat de 1 indică o predicție perfectă în timp ce un rezultat de 0 indică o precizie asemănătoare unor predicții aleatoare.

Pentru a ilustra avantajele acestei metode voi considera următorul exemplu binar de clasificare care are următoarea matrice de confuzie:

	<i>Clasa1</i>	<i>Clasa2</i>
<i>Clasa1</i>	87	3
<i>Clasa2</i>	7	3

După cum se poate vedea, proporția claselor este dezechilibrată: avem 90 de specimene din clasa 1 și doar 10 specimene din clasa 2. Voi calcula acuratețea acestui model de clasificare:

$$acc = \frac{87 + 3}{87 + 3 + 7 + 3} = \frac{90}{100} = 0.9$$

Acest model are o acuratețe de 90%, însă la o privire mai atentă se poate observa că pentru speciemenle din clasa 2 au fost prezise corect 3 și incorect 7. Voi calcula acum coeficientul de corelație pentru a putea face o comparație.

$$mcc = \frac{(87 + 3) \times (87 + 3 + 7 + 3) - [(87 + 7) \times (87 + 3) + (3 + 3) \times (7 + 3)]}{\sqrt{((87 + 3 + 7 + 3)^2 - (87 + 7)^2 - (3 + 3)^2) \times ((87 + 3 + 7 + 3)^2 - (87 + 3)^2 - (7 + 3)^2)}}$$

$$mcc = \frac{90 \times 100 - (94 \times 90 + 6 \times 10)}{\sqrt{(100^2 - 94^2 - 6^2) \times (100^2 - 90^2 - 10^2)}}$$

$$mcc = \frac{480}{\sqrt{1128 \times 1800}} \approx 0.33$$

Așadar, coeficientul de corelație are valoare aproape de 0 (aproximativ 0.33) ceea ce ne indică faptul că nu prezice clasele mai bine decât un model care le alege la întâmplare, în ciuda faptului că are o acuratețe optimistă de 90%.

## 3.4 Strategii de validare

### 3.4.1 Validarea încrucișată

Validarea încrucișată este o metodă de testare a performanței unui model de învățare automată în care datele sunt împărțite în două categorii: una folosită pentru a antrena modelul și cealaltă folosită pentru a-l valida. Cele două seturi de date trebuie interchimbate succesiv astfel încât fiecare dată să poată fi folosită drept validator. [20]

Una din cele mai folosite metode de acest tip este *k-fold cross-validation*, care funcționează în felul următor:

1. se alege un număr  $k$  (de obicei 5 sau 10) pentru a împărți setul de date în  $k$  părți egale
2. se iterează de  $k$  ori, iar la fiecare iterație:
  - $k - 1$  părți ale setului de date sunt considerate drept set de antrenare, iar o parte este considerată drept set de validare
  - se antrenează un model nou pe setul de antrenament din iterația curentă și apoi se verifică pe setul de validare și se salvează rezultatele
3. se calculează scorul final ca medie aritmetică a rezultatelor din fiecare iterație

În cazul de față, dat fiind faptul că autorii SemEval au furnizat un singur set de date am aplicat metoda de validare încrucișată folosind un număr de 5 sub-seturi, rezultând în a avea, la fiecare iterație, un set de antrenare cu 3200 de articole și un set de validare cu 800 de articole. Am folosit această abordare în cadrul experimentelor cu mașini cu vectori suport și amplificarea gradientului.

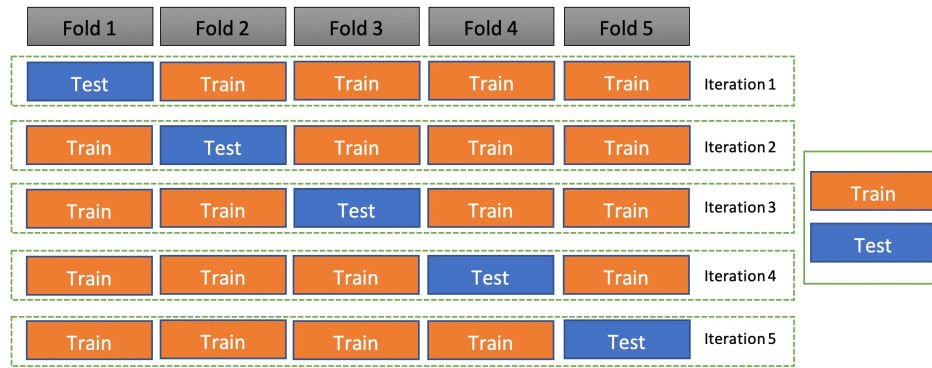


Figura 3.9: Exemplu  $k$ -fold *corss-validation* [19]

### 3.4.2 Divizarea Antrenare-Validare

Divizarea Antrenare-Validare este o metodă de testare a performanței unui model de învățare automată în care setul de date se împarte în două subseturi: unul pentru antrenare și altul pentru validare. În timpul fiecărui experiment (indiferent de metoda de învățare) am folosit setul total de 4000 de articole în modul sugerat de organizatorii SemEval: primele 3200 de articole au fost considerate drept date de antrenare, iar următoarele 800 au fost considerate drept date de validare.

# Capitolul 4

## Rezultate și contribuții

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
LinearSVC	tf-idf	47.8	0.2	26.6	59.9	57.0	45.2	0.2
LinearSVC	tf-idf (titles)	54.5	0.3	38.5	57.5	67.5	48.6	0.2
LinearSVC	tf-idf (post-texts)	56.7	0.4	38.5	58.7	72.8	54.5	0.3
LinearSVC	en_core_web_lg	38.3	0.1	9.8	92.5	12.5	40.4	0.1
LinearSVC	en_core_web_lg (titles)	50.1	0.2	53.9	21.4	74.9	46.2	0.2
LinearSVC	en_core_web_lg (post-texts)	49.7	0.2	54.5	29.2	65.4	49.6	0.3
SVC-rbf	spacy + docs_post_text	55.9	0.3	38.5	62.1	67.2	54.0	0.3
SVC-rbf	spacy + docs_title	52.5	0.3	38.5	57.8	61.2	50.7	0.3
SVC-rbf	spacy + docs	43.9	0.2	10.5	66.2	54.9	43.0	0.2
SVC-rbf	tf-idf + docs_post_text	53.7	0.3	25.9	60.9	74.3	51.2	0.3
SVC-rbf	tf-idf + docs_title	51.1	0.3	25.2	57.5	70.8	46.3	0.2
SVC-rbf	tf-idf + docs	43.1	0.2	12.6	58.1	58.5	46.3	0.2
XGBoost	tf-idf	39.5	0.1	11.2	43.5	63.9	42.5	0.1
XGBoost	tf-idf (titles)	48.0	0.2	27.3	53.4	63.3	47.0	0.2
XGBoost	tf-idf (post-texts)	53.1	0.3	34.3	62.1	63.0	52.6	0.3
XGBoost	en_core_web_lg	44.9	0.2	34.3	62.1	63.0	43.7	0.2
XGBoost	en_core_web_lg (titles)	49.2	0.2	30.1	56.5	60.9	47.6	0.2
XGBoost	en_core_web_lg (post-texts)	50.9	0.3	27.3	59.6	65.7	50.2	0.3
XGBoost (tuned)	tf-idf (post-texts)	54.2	0.3	35.7	61.5	65.4	51.7	0.3
XGBoost (tuned)	tf-idf (title)	49.7	0.2	31.5	52.2	65.4	47.5	0.2
XGBoost (tuned)	tf-idf (doc)	41.5	0.1	14.0	47.8	62.7	42.6	0.1
XGBoost (tuned)	spacy (docs_post_text)	50.0	0.3	27.3	59.0	63.9	50.2	0.3
XGBoost (tuned)	spacy (docs_title)	45.4	0.2	25.9	53.4	57.0	47.5	0.2
XGBoost (tuned)	spacy (docs)	42.8	0.2	14.7	55.6	58.2	44.5	0.2
BERT	docs	47.6	0.2	29.3	50.3	63.2	-	-
BERT	docs_title	58.4	0.4	49.6	60.2	65.3	-	-
BERT	docs_post_text	67.1	0.5	61.5	67.4	72.2	-	-

Tabela 4.1: Rezultate

În acest capitol voi discuta rezultatele prezentate în tabelul 4.1 precum și alte experimente sau încercări notabile. Pentru a păstra coerența voi împărți rezultatele în funcție de metoda de învățare folosită și le voi comenta pe fiecare în parte.

## 4.1 Experimente cu mașini cu vectori suport

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
LinearSVC	tf-idf	47.8	0.2	26.6	59.9	57.0	45.2	0.2
LinearSVC	tf-idf (titles)	54.5	0.3	38.5	57.5	67.5	48.6	0.2
LinearSVC	tf-idf (post-texts)	56.7	0.4	38.5	58.7	72.8	54.5	0.3
LinearSVC	en_core_web_lg	38.3	0.1	9.8	92.5	12.5	40.4	0.1
LinearSVC	en_core_web_lg (titles)	50.1	0.2	53.9	21.4	74.9	46.2	0.2
LinearSVC	en_core_web_lg (post-texts)	49.7	0.2	54.5	29.2	65.4	49.6	0.3

Tabela 4.2: Rezultate Linear SVC

Inițial am folosit algoritmul *LinearSVC* cu parametri de bază atât cu *spacy*, cât și cu *TF-IDF* pentru a clasifica tipul de spoiler. După cum se poate observa în tabelul 4.2 abordarea cu TF-IDF este superioară celei cu *spacy*. Cel mai bun rezultat are o acuratețe balansată de 56.7 % și un coeficient de corelație de 0.4 pe împărțirea inițială a datelor (primele 3200 de articole au fost folosite pentru antrenare, iar restul de 800 pentru validare). Acestea scad puțin în urma validării încrucișate cum este de așteptat.

O altă observație importantă ar fi că modelul obține rezultate asemănătoare folosind fie titluri, fie post-texte cu un mic avantaj pentru cea din urmă. Totuși nu se descurcă la fel de bine cu întregul text al articolului unde acuratețea scade chiar și cu 12 procente, iar coeficientul de corelație se înjumătățește .

Următorul experiment a constatat în utilizarea modelului *SVC* variind valorile parametrilor săi. Prima schimbare încercată a fost schimbare nucleului (*rbf* și *poly*, iar în urma experimentelor am ales să folosesc funcția *rbf* pentru a obține performanța maximă.

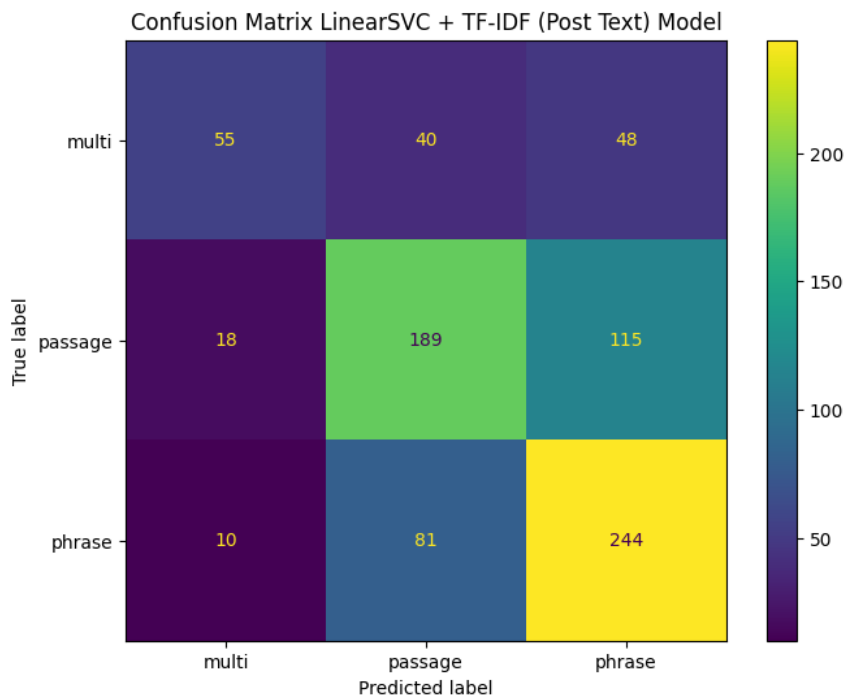


Figura 4.1: Matricea de confuzie pentru LinearSVC



Apoi, am probat numere de la  $10^{-2}$  la  $10^2$  multiplicând cu 10 de fiecare dată ca valori pentru parametrul de regularizare. Rezultatele au fost asemănătoare pentru  $C = 1$  și  $C = 10$ , iar în rest s-a dovedit că rezultatele se înrăutățeau. Astfel, după câteva încercări în plus am ajuns la performanță maximă pentru  $C = 5$ .

În final, am mai experimentat cu *gamma* și *decision\_function\_shape*. Pentru *gamma* valorile alese diminuau performanța modelului și am decis, așadar, să păstrez valoarea de bază. Pentru *decision\_function\_shape*, cele două variante(*ovo* și *ovr*) nu prezintă un impact semnificativ asupra performanței. Astfel am decis să păstrez valoarea de bază pentru micul avantaj pe care îl prezintă rezultatele în această configurație.

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
SVC-rbf	spacy + docs_post_text	55.9	0.3	38.5	62.1	67.2	54.0	0.3
SVC-rbf	spacy + docs_title	52.5	0.3	38.5	57.8	61.2	50.7	0.3
SVC-rbf	spacy + docs	43.9	0.2	10.5	66.2	54.9	43.0	0.2
SVC-rbf	tf-idf + docs_post_text	53.7	0.3	25.9	60.9	74.3	51.2	0.3
SVC-rbf	tf-idf + docs_title	51.1	0.3	25.2	57.5	70.8	46.3	0.2
SVC-rbf	tf-idf + docs_post_text	43.1	0.2	12.6	58.1	58.5	46.3	0.2

Tabela 4.3: Rezultate SVC

Rezultatele din tabelul 4.3 sunt obținute folosind configurația prezentată mai sus și obțin performanțe asemănătoare cu cele din tabelul 4.2. De această dată se pare că modelele procesate cu *spacy* învață mai bine decât cele cu *TF-IDF*. Cel mai bun model din acest set a fost antrenat cu post-texte, la fel ca în cel precedent, iar rezultate sunt aproximativ egale acela.

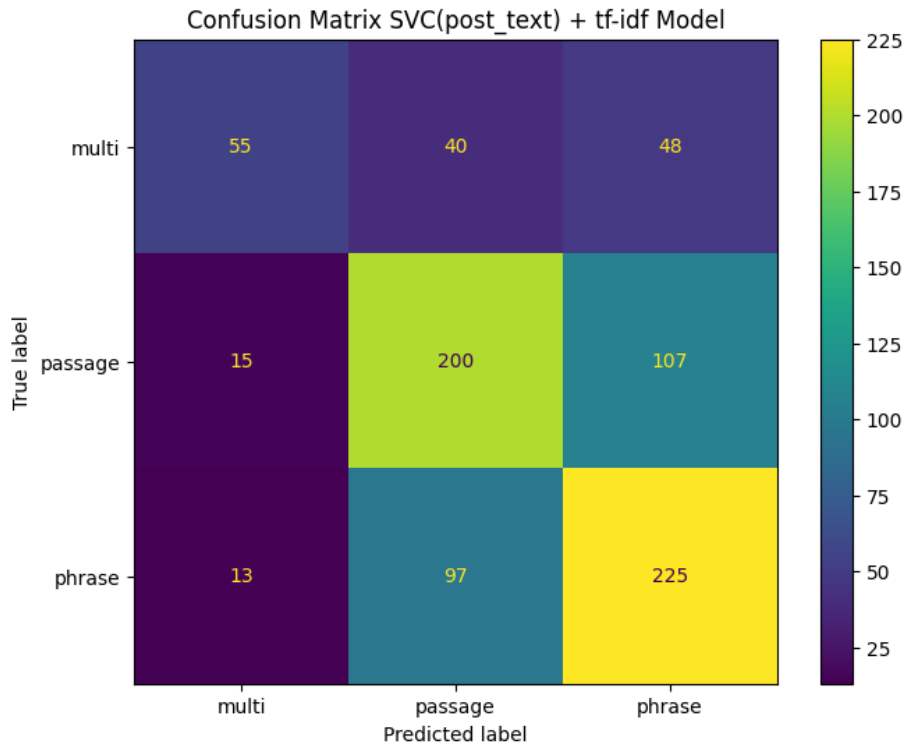


Figura 4.2: Matricea de confuzie pentru SVC modificat

Cele două matrici de confuzie din figurile 4.1 și 4.2 sunt similare și ne ajută să înțelegem mai bine problemele pe care le întâmpină modelele. O primă observație clară este că modelele nu se descurcă deloc bine în a clasifica spoilerurile de tip *multi*, având în vedere distribuția aproximativ egală de pe primul rând. Cea de-a doua observație este că în cazul spoilerurilor de tip *passage* și *phrase* modelele prezic corect în proporție de 60-70% și nu le încurcă cu cele de tip *multi*, însă au încă destul de multe probleme în a se distinge între ele.

## 4.2 Experimente cu amplificarea gradientului

La început, am folosit pentru acest tip de experimente *XGBClassifier* din biblioteca *XGBoost* cu parametrul "booster" setat ca gbtree. Cel mai performant model are o acuratețe balansată de 53.1% și un coeficient de corelație de 0.3, iar în urma validării încrucișate performanța rămâne în mare parte neschimbată. Observația făcută anterior legat de faptul că modele au rezultate mai bune atunci când sunt antrenate cu post-texte rămâne valabilă și în acest caz. Un lucru puțin surprinzător ce reiese din tabelul 4.4 este că în urma procesării cu *TF-IDF* diferența de performanță între modelul antrenat cu titluri și cel antrenat cu post-texte este una importantă, de mai mult de 5 procente cand ne uităm la acuratețe și de mai mult de 0.1 când ne uităm la coeficientul de corelație (și cu și fără validare încrucișată). Până acum cele doua aveau performanțe similare, MCC-ul a rămas de obicei cam la fel, iar acuratețea nu a diferit cu mai mult de 3%.

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
XGBoost	tf-idf	39.5	0.1	11.2	43.5	63.9	42.5	0.1
XGBoost	tf-idf (titles)	48.0	0.2	27.3	53.4	63.3	47.0	0.2
XGBoost	tf-idf (post-texts)	53.1	0.3	34.3	62.1	63.0	52.6	0.3
XGBoost	en_core_web_lg	44.9	0.2	34.3	62.1	63.0	43.7	0.2
XGBoost	en_core_web_lg (titles)	49.2	0.2	30.1	56.5	60.9	47.6	0.2
XGBoost	en_core_web_lg (post_texts)	50.9	0.3	27.3	59.6	65.7	50.2	0.3

Tabela 4.4: Rezultate XGBClassifier

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
XGBoost (tuned)	tf-idf (post-texts)	54.2	0.3	35.7	61.5	65.4	51.7	0.3
XGBoost (tuned)	tf-idf (title)	49.7	0.2	31.5	52.2	65.4	47.5	0.2
XGBoost (tuned)	tf-idf (doc)	41.5	0.1	14.0	47.8	62.7	42.6	0.1
XGBoost (tuned)	spacy (docs_post_text)	50.0	0.3	27.3	59.0	63.9	50.2	0.3
XGBoost (tuned)	spacy (docs_title)	45.4	0.2	25.9	53.4	57.0	47.5	0.2
XGBoost (tuned)	spacy (docs)	42.8	0.2	14.7	55.6	58.2	44.5	0.2

Tabela 4.5: Rezultate XGBClassifier tunat

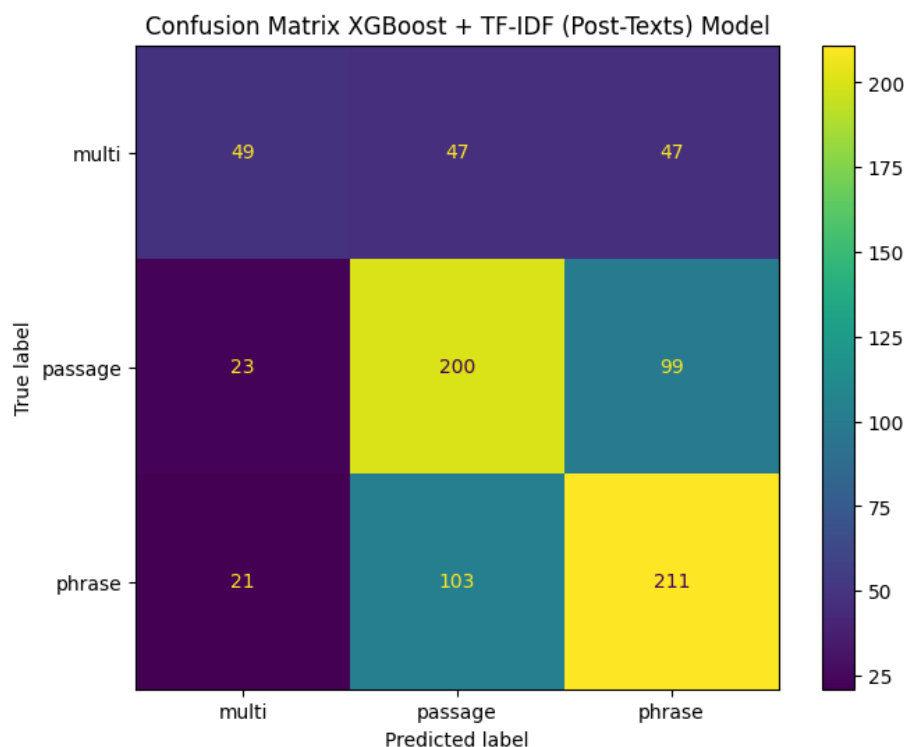


Figura 4.3: Matricea de confuzie pentru XGBClassifier

În următorul grup de experimente am experimentat cu mai multi parametrii ai *XGB-Classifier*. Primul parametru schimbat a fost *booster*, schimbând din "gbtree" în "dart". Această schimbare nu a avut aproape niciun impact asupra rezultatelor, astfel am decis să îl las la fel ca în setul precedent de antrenare.

Următorul parametru cu care am experimentat a fost *learning\_rate*. Inițial pentru a avea o imagine de ansamblu am încercat valori cuprinse între  $10^{-2}$  și  $10^2$ , multiplicând cu 10 la fiecare iterație. Am obținut rezultate apropiate de cele din setul anterior cu 0 și 1. Ulterior, din aproape în aproape am găsit valoarea 0.315 ca fiind cea în care performanța este maximă și o depășește pe cea din experimentele anterioare.

Cel de-al treilea parametru pe care am încercat să-l modific este *n\_estimators*. Am pornit cu valoarea 100 care nu afecta în niciun mod succesul modelelor. Încercând apoi 90 și 110 și ulterior 1, 10, 200, 500, 1000 am observat o scădere majoră a performanței la cele din urmă și creșteri minore pentru primele. Am ales să folosesc valoarea 95, intrucât (deși infimă) creșterea succesului modelului era maximă.

Matricele de confuzie din figurile 4.3 și 4.4 sunt similare cu cele ale experimentelor cu mașini cu vectori suport și scot la iveală aceleași limitări ale modelelor:

- confuzia totală în clasificarea spoilerelor *multi*
- confuzia parțială în distingerea unor exemple din clasele *phrase* și *passage*

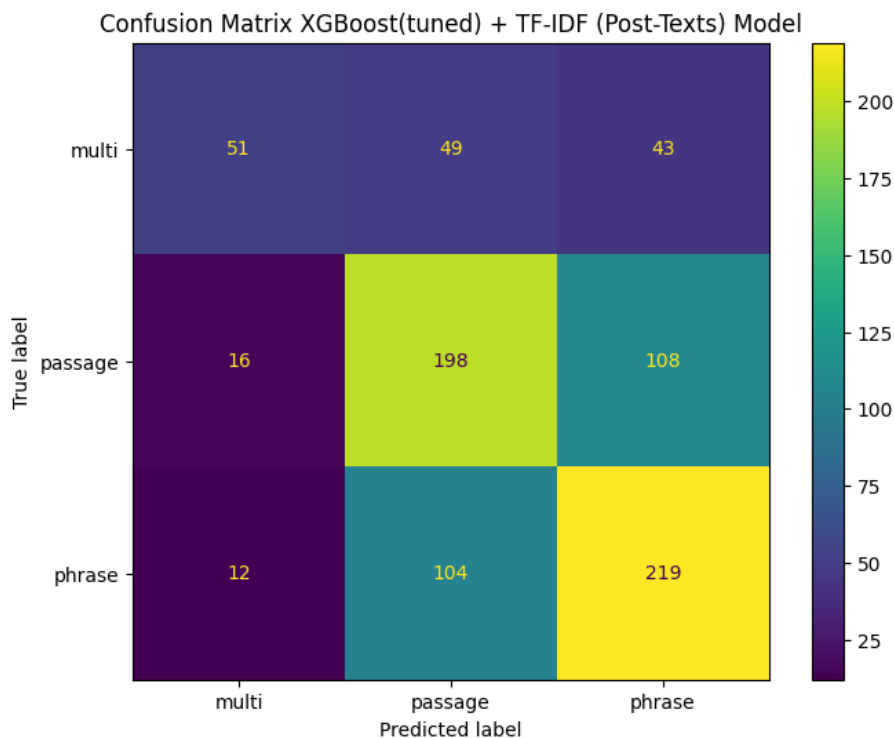


Figura 4.4: Matricea de confuzie pentru XGBClassifier modificat

### 4.3 Experimente cu transformeri pe text

Model	Feature	Bal. Acc.	MCC	%multi	%passage	%phrase	ACC 5CV	MCC 5CV
BERT	docs	47.6	0.2	29.3	50.3	63.2	-	-
BERT	docs_post_text	67.1	0.5	61.5	67.4	72.2	-	-
BERT	docs_title	58.4	0.4	49.6	60.2	65.3	-	-
BERT preprocess	docs_post_text	55.3	0.3	40.5	52.1	73.1	-	-

Tabela 4.6: Rezultate BERT

În ultima serie de experimente am antrenat cu BERT drept model cu transformeri pe text. Modele au fost antrenate fara o preprocesare a textului timp de 10 epoci cu *batch\_size* de 12, iar din considerente de timp și resurse nu au fost evaluate folosind validarea încrucișată. Din tabelul 4.6 se observă că această abordare este superioară celor prezentate anterior, întrucât sunt 2 experimente care prezintă o performanță mai mare decât oricare din abordările anterioare.

Așadar, cel mai bun model antrenat are o acuratețe balansată de 67.1% și un coeficient de corelație de 0.5, rezulate mai bune decât toate cele obținute până acum.

În ultima linie a tabelului 4.6 este un ultim experiment ce are rezultate mai slabe, dar în care am încercat o preprocesare a datelor prin mai multe metode:

- extendirea cuvintelor sau expresilor scrise contractat (de exemplu: "we'll" în loc de "we will")
- ștergerea spațiilor libere
- ștergerea de *stopwords*
- lematizare

Am concluzionat că o astfel de preprocesare diminuează caracterul senzationalist al textelor, făcând astfel contextul mai dificil de înțeles. Astfel se pot explica rezultatele slabe ale acelui model.

În figura 4.5 se poate observa cum BERT depășește câteva din limitările celorlalte modele și clasifică în mod corect 61.5% din exemplele din clasa *multi* și având acuratețe crescută în prezicerea celorlalte două clase.

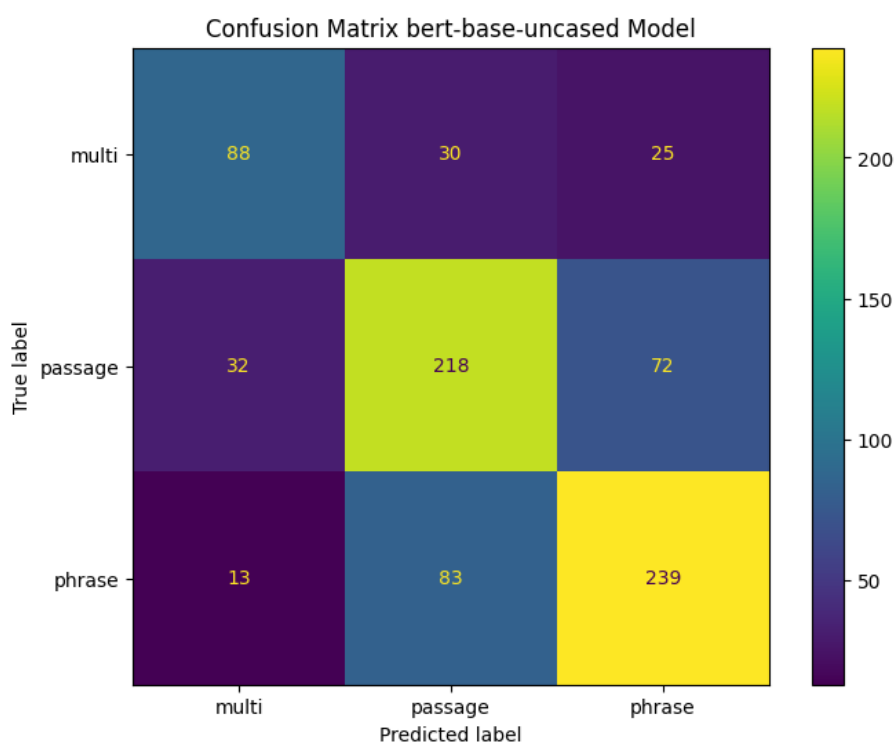


Figura 4.5: Matricea de confuzie pentru BERT

# Capitolul 5

## Concluzii și activități viitoare

În demersul experimentelor efectuate, am descoperit că modelele cu transformeri pe text ar putea fi cheia clasificării spoilerelor, fiind mult superioare celor cu mașini cu vectori suport și cu amplificarea gradientului.

Experimentele mele cu transformeri au avut la bază modelul BERT, fără preprocesare sau modificări făcute parametrilor, ajungând la o acuratețe de aproximativ 70%. Acest lucru mă face să cred că folosirea altor modele cu transformeri pe text, împreună cu modificări potrivite asupra parametrilor de antrenare, ar putea crește performanța acestui tip de clasificator. De aceea mi-aș dori ca în cercetările următoare să experimentez cu mai multe modele precum: RoBERTa, DeBERTa, DistilBERT sau ALBERT.

Cu toate acestea, majoritatea confuziilor apar în clasa *multi*, caracterizată prin mai multe pasaje și fraze discontinue. În urma examinării setului de date, am observat că o mare porțiune din exemplele de tip *multi* conțin fraze sau pasaje foarte scurte, făcând astfel clasificarea dificilă chiar și pentru oameni.

Consider că acest impediment nu este cel mai mare obstacol în fața minimizării numărului de postări senzaționaliste de pe rețelele sociale. În cele din urmă, acest clasificator doar va servi unui generator ce format să aibă spoilerul. Sunt de părere că deși inițial această metoda va diminua numărul click-uri pe care utilizatorul de rând le va face pe articole senzaționaliste, în cele din urmă companiile și creatorii de conținut vor găsi alte metode pentru a ocolii acest obstacol.

# Bibliografie

- [1] 3.3. Metrics and scoring: quantifying the quality of predictions, [Online; accesat pe 14 Iunie 2023], URL: [https://scikit-learn.org/stable/modules/model\\_evaluation.html#matthews-corrcoef](https://scikit-learn.org/stable/modules/model_evaluation.html#matthews-corrcoef).
- [2] Jay Alamar, *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*, [Online; accesat pe 15 Iunie 2023], URL: <http://jalamar.github.io/illustrated-bert/>.
- [3] Tianqi Chen și Carlos Guestrin, „XGBoost”, în Aug. 2016, DOI: [10.1145/2939672.2939785](https://doi.org/10.1145/2939672.2939785), URL: <https://doi.org/10.1145/2939672.2939785>.
- [4] *Clickbait Challenge at SemEval 2023 - Clickbait Spoiling*, URL: <https://pan.webis.de/semeval23/pan23-web/clickbait-challenge.html>.
- [5] Jacob Devlin, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, Oct. 2018, URL: <https://arxiv.org/abs/1810.04805>.
- [6] Maria Diaz, „Even Google is warning its employees about AI chatbot use”, în (Iun. 2023), [Online; accesat pe 17 Iunie 2023], URL: <https://www.zdnet.com/article/even-google-is-warning-its-employees-about-ai-chatbot-use/>.
- [7] Maik Fröbe, Tim Gollub, Matthias Hagen și Martin Potthast, „SemEval-2023 Task 5: Clickbait Spoiling”, în *17th International Workshop on Semantic Evaluation (SemEval-2023)*, 2023.
- [8] Maik Fröbe, Matti Wiegmann, Nikolay Kolyada, Bastian Grahm, Theresa Elstner, Frank Loebe, Matthias Hagen, Benno Stein și Martin Potthast, „Continuous Integration for Reproducible Shared Tasks with TIRA.io”, în *Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023)*, Lecture Notes in Computer Science, Berlin Heidelberg New York: Springer, Apr. 2023.
- [9] Rohith Gandhi, „Support Vector Machine — Introduction to Machine Learning Algorithms”, în (Nov. 2022), [Online; accesat pe 12 Iunie 2023], URL: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>.

- [10] GeeksforGeeks, „Support Vector Machine SVM Algorithm”, în *GeeksforGeeks* (Iun. 2023), [Online; accesat pe 12 Iunie 2023], URL: <https://www.geeksforgeeks.org/support-vector-machine-algorithm/>.
- [11] Maarten Grootendorst, „BERTopic: Neural topic modeling with a class-based TF-IDF procedure”, în *arXiv preprint arXiv:2203.05794* (2022).
- [12] Matthias Hagen, Maik Fröbe, Artur Jurk și Martin Potthast, „Clickbait Spoiling via Question Answering and Passage Retrieval”, în *60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, ed. de Smaranda Muresan, Preslav Nakov și Aline Villavicencio, Association for Computational Linguistics, Mai 2022, pp. 7025–7036.
- [13] Matthew Honnibal, Ines Montani, Sofie Van Landeghem și Adriane Boyd, „spaCy: Industrial-strength Natural Language Processing in Python”, în (2020), DOI: [10.5281/zenodo.1212303](https://doi.org/10.5281/zenodo.1212303).
- [14] Juwon Hwang, Porismita Borah, Dhavan Shah și Markus Brauer, „The Relationship among COVID-19 Information Seeking, News Media Use, and Emotional Distress at the Onset of the Pandemic”, în *International Journal of Environmental Research and Public Health* 18.24 (2021), ISSN: 1660-4601, DOI: [10.3390/ijerph182413198](https://doi.org/10.3390/ijerph182413198), URL: <https://www.mdpi.com/1660-4601/18/24/13198>.
- [15] Rohit Kundu, *Confusion Matrix: How To Use It and Interpret Results [Examples]*, [Online; accesat pe 13 Iunie 2023], Mai 2023, URL: <https://www.v7labs.com/blog/confusion-matrix-guide>.
- [16] Dragoș-Stefan Mihalcea și Sergiu Nisioi, *Clark Kent at SemEval-2023 Task 5: SVMs, Transformers, and Pixels for Clickbait Spoiling*, 2023.
- [17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot și E. Duchesnay, „Scikit-learn: Machine Learning in Python”, în *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [18] Abinash Pujahari și Dilip Singh Sisodia, „Clickbait detection using multiple categorisation techniques”, în *Journal of Information Science* 47.1 (Feb. 2021), pp. 118–128, DOI: [10.1177/0165551519871822](https://doi.org/10.1177/0165551519871822), URL: <https://doi.org/10.1177/0165551519871822>.
- [19] Gopal Krishna Ranjan, „Introduction to k-fold Cross-Validation in Python”, în *SQLRelease* (Iul. 2021), [Online; accesat pe 14 Iunie 2023], URL: <https://sqlrelease.com/introduction-to-k-fold-cross-validation-in-python>.
- [20] Payam Refaeilzadeh, Lei Tang și Hongfang Liu, *Cross-Validation*, Ian. 2009, pp. 532–538, DOI: [10.1007/978-0-387-39940-9\\_565](https://doi.org/10.1007/978-0-387-39940-9_565), URL: [https://doi.org/10.1007/978-0-387-39940-9\\_565](https://doi.org/10.1007/978-0-387-39940-9_565).



- [21] Stephen Robertson, „Understanding Inverse Document Frequency: On Theoretical Arguments for IDF”, în *Journal of Documentation - J DOC* 60 (Oct. 2004), pp. 503–520, DOI: [10.1108/00220410410560582](https://doi.org/10.1108/00220410410560582).
- [22] Parth Shukla, „Top 10 Interview Questions on Gradient Boosting Algorithms”, în *Analytics Vidhya* (Nov. 2022), [Online; accesat pe 13 Iunie 2023], URL: <https://www.analyticsvidhya.com/blog/2022/11/top-10-interview-questions-on-gradient-boosting/>.
- [23] Vladimir Vapnik, *The nature of statistical learning theory*, Springer science & business media, 1999.
- [24] Ashish Vaswani, *Attention Is All You Need*, Iun. 2017, URL: <https://arxiv.org/abs/1706.03762>.