UNIVERSITATEA „ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

Aplicație web pentru detecția știrilor false prin intermediul învățării automate

propusă de

Prenume Nume

Sesiunea: iunie, 2024

Coordonator științific

Titlu Prenume Nume

UNIVERSITATEA „ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

Aplicație web pentru detecția știrilor false prin intermediul învățării automate

propusă de

Prenume Nume

Sesiunea: iunie, 2024

Coordonator științific

Titlu Prenume Nume

Cuprins

Introducere

Motivație

Obiective generale

Scurtă descriere a soluției

Contribuții

1. Descrierea problemei

Situația politică la nivel mondial din ultima decadă a amplificat dezinformarea populației și manipularea opiniei sale prin diverse mijloace, printre care se regăsește și diseminarea informațiilor cu caracter înșelător prin intermediul site-urilor web ce publică articole de încredere îndoielnică, sau a mesajelor publice sau private împrăștiate de conturi false înregistrate pe anumite rețele de socializare.

Un sistem de detecție a știrilor false reprezintă o soluție de clasificare a unei colecții de afirmații provenite dintr-o sursă pentru stabilirea veridicității acestora. În scopul acestei lucrări, o grupare de fraze ce pot proveni dintr-un articol jurnalistic este considerată o știre. De asemenea, numele site-ului web (afișat alături de sigla sa, sau făcând parte din adresa web) unde a fost găsită o știre este considerat sursa acesteia.

Veridicitatea unei știri este reprezentată printr-o valoare de adevăr, astfel încât știrile ce provin dintr-o sursă de încredere și care prezintă afirmații verosimile vor fi considerate adevărate, în timp ce știrile provenind dintr-o sursă despre care se cunoaște că a mai publicat în trecut știri false sau în care informația nu este verosimilă, vor fi considerate false.

În continuarea acestui capitol, vor fi prezentate sumar o selecție de soluții existente formate din sisteme/aplicații comerciale ce furnizează o rezolvare la problema detectării știrilor false.

1.1 Soluții comerciale existente pentru detecția știrilor false

1.1.1 TrustServista

„TrustServista” este o platformă de analiză și verificare a știrilor ce folosește inteligența artificială devoltată de o companie start-up din Cluj-Napoca, „Zetta Cloud”, dezvoltată în anul 2018 cu ajutorul unei sponsorizări din partea „Google Digital Initiative”, un program pentru combaterea fenomenului știrilor false.

TrustServista folosește algoritmi avansați de inteligență artificială pentru a determina gradul de încredere în articolele jurnalistice. Clienții furnizează un URL sau text direct, algoritmii analizează articolul, identificând similarități semantice, legături și referințe între acesta și alt conținut disponibil pe web, urmând apoi a genera un set de măsurători ale gradului de încredere. [1]

Acestea includ un scor „TrustLevel” - indicând gradul de încredere al conținutului și, de asemenea, „Pacientul Zero”, prin care se identifică sursa originară a articolului. Aceasta poate reprezenta mai multe surse, printre care se numără postări pe rețele de socializare, pe bloguri, pagini web sau publicații jurnalistice. [1]

2. Abordări anterioare

2.1 Istorie

3. Descrierea soluției

Scopul aplicației web este de a prezice valoarea de adevăr al unui text furnizat prin diferite metode, printre care se numără:

- Introducerea directă într-o casetă text;
- Selectarea textului prezent pe o pagină web a unui articol și transferul acestuia către caseta text;
- Extragerea în caseta text dintr-un fișier încărcat de utilizator din pagina aplicației.

folosind diverși clasificatori binari.

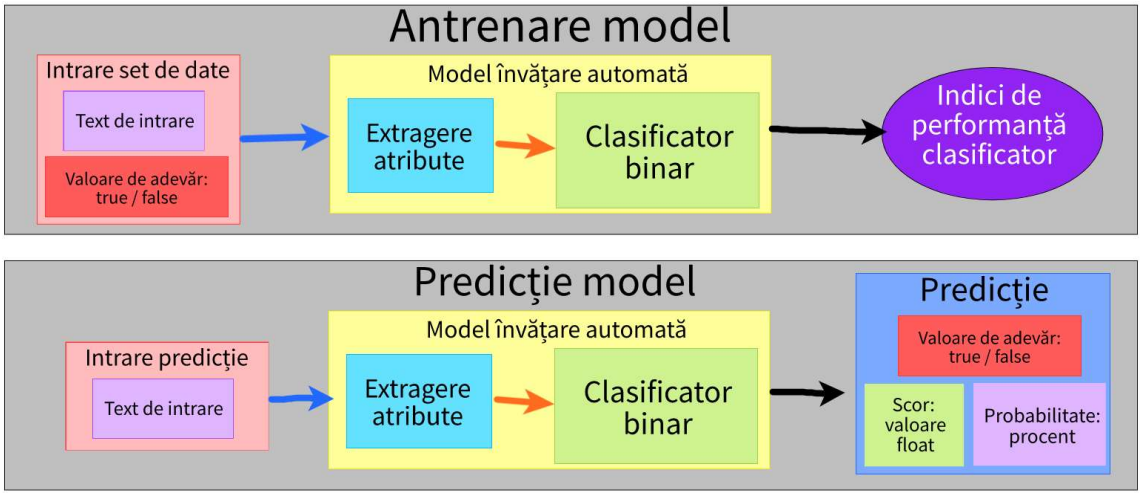

Figura 1. Diagrama de funcționare a sistemului

3.1 Tehnologii utilizate în dezvoltarea aplicației

3.1.1 .NET 8

.NET reprezintă o platformă gratuită, cu sursă deschisă, multiplatformă de dezvoltare pentru construirea mai multor tipuri de aplicații. Aceasta poate rula programe scrise în mai multe limbaje de programare, C# fiind cel mai popular dintre ele. .NET este bazat pe un runtime de înaltă performanță și este utilizat în producție de multe aplicații cu nevoie de scalabilitate înaltă. [2] Versiunea 8 a platformei .NET este ultima lansată la momentul actual și este o versiune LTS. („long term support”, suportată pe termen îndelungat)

Limbajul C# folosit împreună cu .NET este un limbaj de programare modern, sigur, orientat pe obiecte, puternic tipat și de uz general care prezintă atât caracteristici de nivel

înalt, precum tipul de date „înregistrare” (record), cât și caracteristici low-level, precum pointeri către funcții.

3.1.2 ML.NET

„Machine Learning for .NET” este o bibliotecă pentru .NET dedicată învățării automate. Aceasta permite dezvoltatorilor să construiască, antreneze, lanseze și să consume modele specializate în aplicații .NET existente fără a necesita expertiză anterioară în dezvoltarea modelelor de învățare automată sau experiență în alte limbaje de programare precum Python sau R. Framework-ul furnizează posibilitatea de a încărca date din fișiere sau baze de date, permite transformări de date și include mulți algoritmi de învățare automată. Folosind ML.NET, se pot construi modele pentru diverse scenarii, printre care se numără:

- Clasificare binară;
- Clasificare multiclass;
- Previziune;
- Detecția anomaliilor. [3]

3.1.3 Blazor

Blazor este un framework de dezvoltare web gratuit și cu sursă liberă ce permite dezvoltatorilor să creeze interfețe web pentru utilizatori, bazat pe componente, folosind C# și HTML. Acest framework este dezvoltat de Microsoft, ca parte din framework-ul web ASP.NET Core. Blazor poate fi folosit pentru dezvoltarea aplicațiilor de tip single-page, aplicații mobile sau de tip server side rendering, folosind tehnologii .NET. [4]

3.2 Obținerea unui set de date pentru antrenare

Un set de date aliniat obiectivelor lucrării a fost selectat în urma căutării pe platforma Kaggle, ce furnizează o comunitate pentru pasionații de știința datelor și a învățării automate. Acest set se numește „Misinformation & Fake News text dataset 79k” și a fost publicat de către un contributor al platformei ce activează sub pseudonimul „stevenpeutz” în urmă cu doi ani. Setul este împărțit în două fișiere: un fișier ce conține 34975 de articole considerate adevărate și 43642 de articole considerate false (incluzând informații false, dezinformare sau propagandă).

Știrile reale provin din surse credibile, precum „Reuters”, „New York Times”, „Washington Post” și altele. Știrile false au fost colectate de pe site-uri cu alinierea politică „extremism de dreapta”, precum „Redflag Newsdesk”, „Breitbart”, „Truth Broadcast Network”, dintr-un set de date public elaborat pentru un articol științific din 2017 și din cazurile de propagandă și dezinformare descoperite în cadrul proiectului de „fact checking” „EUvsDisinfo”. [5]

Pentru a face setul de date potrivit pentru antrenarea modelelor de învățare automată s-au efectuat următorii pași:

- Fiecărui fișier i s-a adăugat o coloană reprezentând valoarea de adevăr în format numeric (0 pentru fals, 1 pentru adevărat);
- Fișierele au fost unite printr-o operație de concatenare;

- Liniile fișierului au fost distribuite în mod aleatoriu pentru a întrețese știri false cu știri adevărate.

3.3 Modelul de învățare automată

Pentru predicția valorii de adevăr a unei știri, s-a optat pentru utilizarea clasificatorilor binari în detrimentul clasificatorilor de tip multiclass, deoarece, din punct de vedere psihologic, un utilizator poate discerne facil între două valori propoziționale, în defavoarea a mai multor categorii de știri false (dezinformare, parodie, clickbait, ș.a.)

3.3.1 Clasificarea binară

Clasificarea statistică reprezintă un tip de învățare supervizată, reprezentând o metodă de învățare automată pentru care categoriile sunt predefinite și care este folosită pentru a categorisi observații probabilistice noi în categoriile predefinite. Tipul de învățare supervizată în care sunt prezente doar două categorii predefinite este cunoscută ca clasificare statistică binară. Printre metodele des întâlnite de clasificare binară se numără:

- Arbori de decizie;
- Păduri de arbori de decizie;
- Rețele Bayes;
- Mașini cu vectori de suport;
- Rețele neuronale;
- Regresie logistică. [6]

Pentru antrenarea modelului au fost selectați mai mulți clasificatori binari furnizați de biblioteca ML.NET:

- Averaged Perceptron;
- Fast Forest;
- Fast Tree;
- Field Aware Factorization Machine;
- Lbfgs Logistic Regression;
- Light Gbm;
- Linear Svm;
- Sdca Logistic Regression;
- Sdca Non Calibrated.

3.3.2 Prelucrarea textului pentru clasificatorul binar

Datele de intrare, atât pentru antrenarea clasificatorului binar, cât și în cazul predicției, conțin text. Acest text nu poate fi furnizat în starea actuală către clasificator, întrucât acesta necesită un vector de numere în virgulă mobilă ca intrare.

Transformarea dintr-un text într-un vector de numere se poate realiza prin extragerea atributelor. În contextul ML.NET, extragerea atributelor este efectuată prin efectuarea unor operații:

- Detecția limbii textului;
- Analiză lexicală (extragerea cuvintelor relevante);

- Normalizarea textului (eliminarea diacriticelor);
- Eliminarea cuvintelor nesemnificative (stopwords);
- TF-IDF (lb. eng. „term frequency-inverse document frequency”, măsurarea importanței unui termen, comparativ cu restul documentului).

4. Implementarea soluției

Soluția C# este alcătuită dintr-un proiect pentru „frontend” (conținând interfața cu utilizatorul) și unul pentru „backend” (conținând logica de business).

4.1 Proiectul de backend

În cadrul proiectului backend sunt create mai multe clase publice pentru folosința din contextul frontend-ului.

Clasa Utils conține câteva funcții utilizate frecvent în celelalte clase. Proprietatea Context furnizează o referință singleton la o instanță a clasei MLContext, necesară majorității operațiilor ML.NET. Funcția SaveMetrics salvează măsurătorile asociate antrenării unui clasificator în format XML în directorul curent, iar LoadMetrics încarcă, în funcție de numele clasificatorului furnizat ca parametru, ceea ce s-a salvat anterior.

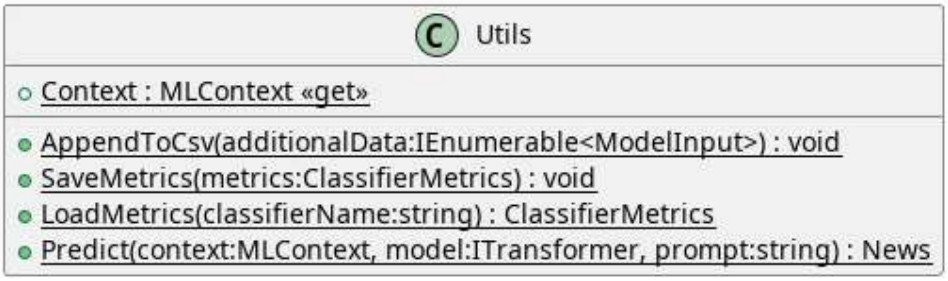

Figura 2. Clasa Utils

4.1.1 Clase model

Clasa ClassifierMetrics conține măsurători obținute în urma antrenării unui clasificator binar. Această clasă este serializată în format XML și scrisă în directorul curent pentru a salva rezultatele ultimei antrenări.

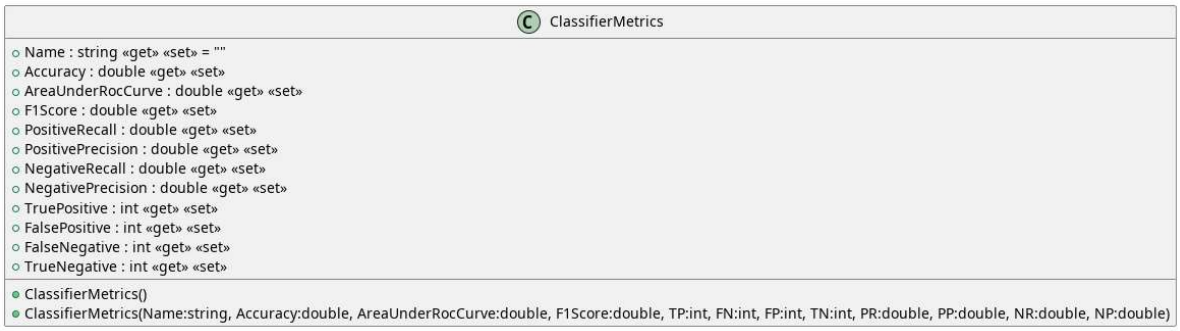

Figura 3. Clasa ClassifierMetrics

Recordul News conține rezultatul unei predicții (output-ul sistemului).

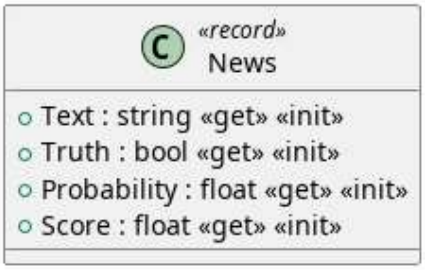

Figura 4. Record-ul News

Clasa ModelInput conține input-ul sistemului, format dintr-un text și o valoare de adevăr. Se construiește o instanță pentru fiecare înregistrare din setul de date.

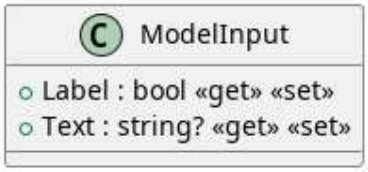

Figura 5. Clasa ModelInput

Clasa ModelOutput extinde clasa ModelInput și conține rezultatul predicției modelului. Recordul News se instanțiază cu valorile furnizate de această clasă.

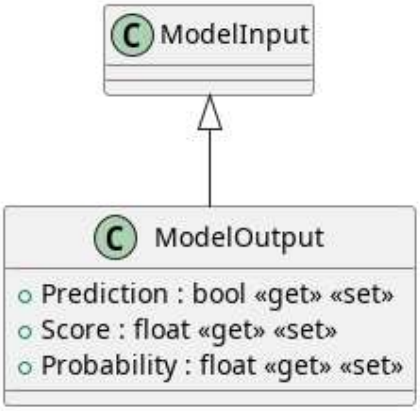

Figura 6. Clasa ModelOutput

4.1.2 Clase clasificator

Fiecare clasificator implementează interfața INewsBinaryClassifier, care conține o metodă pentru predicția unei știri, o metodă pentru antrenarea clasificatorilor și o proprietate pentru măsurătorile obținute în urma antrenării.

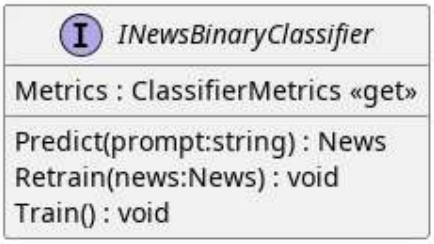

Figura 7. Interfața INewsBinaryClassifier

Pe baza clasificatorilor din biblioteca ML.NET, au fost create următoarele clase clasificator:

- AveragedPerceptronNewsBinaryClassifier;
- FastForestNewsBinaryClassifier;
- FastTreeNewsBinaryClassifier;
- FieldAwareFactorizationMachineNewsBinaryClassifier;
- LbfgsLogisticRegressionNewsBinaryClassifier;
- LightGbmNewsBinaryClassifier;
- LinearSvmNewsBinaryClassifier;
- SdcaLogisticRegressionNewsBinaryClassifier;
- SdcaNonCalibratedNewsBinaryClassifier.

Constructorul acestor clase încearcă să încarce modelul și măsurătorile obținute anterior și salvate în directorul curent, iar în cazul în care acestea nu pot fi încărcate, se reantrenează modelul.

```
13      public AveragedPerceptronNewsBinaryClassifier()
14      {
15          try
16          {
17              string classifierName = GetType().Name.Split("NewsBinaryClassifier")[0];
18              Model = Utils.Context.Model.Load(Path.Combine(Environment.CurrentDirectory,
19                  $"{classifierName}-pretrained-model.zip"), out _);
20              Metrics = Utils.LoadMetrics(classifierName);
21          }
22          catch
23          {
24              Console.WriteLine("Failed to load pretrained model, training...");
25              Train();
26          }
27          if (Model == null || Metrics == null)
28              throw new Exception("Failed to train model");
29      }
```

Figura 8. Constructorul clasei AveragedPerceptronNewsBinaryClassifier

Metoda Predict apelează metoda cu același nume din clasa Utils, furnizând ca parametri modelul cu care se face predicția și textul știrii. Metoda Train este cea mai amplă din cadrul claselor clasificator și are rolul de a antrena modelul folosind un clasificator specific fiecărei clase. Această metodă realizează următorii pași:

- Se încarcă setul de date din fișierul train.csv în variabila dataView;

- Se construiește variabila estimator în care este stocat algoritmul modelului;
- Se împarte setul de date în seturi de antrenare și de test (acesta din urmă reprezentând 20% din totalul setului de date);
- Se obține modelul în urma aplicării algoritmului pe setul de antrenare;
- Se obțin măsurătorile în urma predicției setului de date de test și comparării acestor predicții cu valorile de adevăr actuale;
- Se salvează modelul în format ZIP în directorul curent;
- Se inițializează proprietatea Metrics a clasei, pe baza măsurătorilor obținute;
- Se salvează măsurătorile în format XML în directorul curent.

```
public void Train()
{
    var dataView = Utils.Context.Data.LoadFromTextFile<ModelInput>(
        Path.Combine(Environment.CurrentDirectory, "train.csv"),
        separatorChar: ',',
        hasHeader: true
    );
    var estimator = Utils.Context.Transforms.Text.FeaturizeText(
        outputColumnName: "Features",
        inputColumnName: nameof(ModelInput.Text))
        .Append(Utils.Context.BinaryClassification.Trainers.AveragedPerceptron(
            labelColumnName: "Label",
            featureColumnName: "Features"
        ))
        .Append(Utils.Context.BinaryClassification.Calibrators.Platt());
    var trainTestSplit = Utils.Context.Data.TrainTestSplit(dataView, testFraction: 0.2);
    var model = estimator.Fit(trainTestSplit.TrainSet);
    IDataView predictions = model.Transform(trainTestSplit.TestSet);
    BinaryClassificationMetrics metrics = Utils.Context.BinaryClassification.Evaluate(predictions);
    Console.WriteLine();
    Console.WriteLine("Model quality metrics evaluation");
    Console.WriteLine("--------------------------------");
    Console.WriteLine($"Accuracy: {metrics.Accuracy:P2}");
    Console.WriteLine($"Auc: {metrics.AreaUnderRocCurve:P2}");
    Console.WriteLine($"F1Score: {metrics.F1Score:P2}");
    Console.WriteLine(metrics.ConfusionMatrix.GetFormattedConfusionTable());
    Console.WriteLine("=============== End of model evaluation ===============");
    Utils.Context.Model.Save(
        model,
        dataView.Schema,
        Path.Combine(Environment.CurrentDirectory,
            $"{GetType().Name.Split("NewsBinaryClassifier")[0]}-pretrained-model.zip")
    );
    Model = model;
    Metrics = new ClassifierMetrics(
        GetType().Name.Split("NewsBinaryClassifier")[0],
        metrics.Accuracy,
        metrics.AreaUnderRocCurve,
        metrics.F1Score,
        (int)metrics.ConfusionMatrix.Counts[0][0],
        (int)metrics.ConfusionMatrix.Counts[0][1],
        (int)metrics.ConfusionMatrix.Counts[1][0],
        (int)metrics.ConfusionMatrix.Counts[1][1],
        metrics.PositiveRecall,
        metrics.PositivePrecision,
        metrics.NegativeRecall,
        metrics.NegativePrecision
    );
    Utils.SaveMetrics(Metrics);
}
```

Figura 9. Metoda Train, comună claselor de tip NewsBinaryClassifier

Din cauză că unii clasificatori nu furnizează și probabilitatea unei predicții, s-a utilizat calibratorul Platt.

În contextul învățării automate, „calibratorul Platt” este o tehnică folosită pentru a calibra probabilitățile de ieșire a unui clasificator binar, în sensul că acesta ajustează scorul probabilităților, astfel încât acestea sunt mai precise.

Calibratorul Platt este des întâlnit pentru a remedia situațiile în care probabilitățile de ieșire nu sunt destul de precise. Acest calibrator poate fi reprezentat matematic astfel:

$$P(y = 1 \mid f(x)) = \frac{1}{1 + \exp\left(a * f(x) + b\right)}$$

Formula 1. Probabilitatea calibratorului Platt

5. Rezultate statistice și interpretare

Concluzii

Referințe

[1] „TrustServista: A question of trust | Digital News Initiative,” [Interactiv]. Available: https://newsinitiative.withgoogle.com/dnifund/report/battling-misinformation/trustservista-question-trust/. [Accesat 13 Mai 2024].

[2] „Introduction to .NET - .NET | Microsoft Learn,” [Interactiv]. Available: https://learn.microsoft.com/en-us/dotnet/core/introduction. [Accesat 15 Mai 2024].

[3] „GitHub - Machine Learning for .NET,” [Interactiv]. Available: https://github.com/dotnet/machinelearning. [Accesat 16 Mai 2024].

[4] „Blazor - Wikipedia,” [Interactiv]. Available: https://en.wikipedia.org/wiki/Blazor. [Accesat 16 Mai 2024].

[5] „Misinformation & Fake News text dataset 79k,” [Interactiv]. Available: https://www.kaggle.com/datasets/stevenpeutz/misinformation-fake-news-text-dataset-79k. [Accesat 14 Mai 2024].

[6] „Binary classification - Wikipedia,” [Interactiv]. Available: https://en.wikipedia.org/wiki/Binary_classification. [Accesat 15 Mai 2024].

Tabel de figuri