Descrierea Aplicației **Alemia**

Apostolescu Ștefan
Băjan Ionuț-Mihăiță
Iosif George-Andrei

04/02/2021

Tabelă de Conținut

1 Introducere

Alemia este o aplicație ce folosește algoritmi de învățare automată cu scopul de a automatiza procesul de oferire a unor note pentru proiectele realizate de studenți, pentru cursul de "*Programare Orientată pe Obiecte*". Oferă profesorilor posibilitatea de a contrui seturi de date cu acest tip de proiecte, de a antrena propriile modele de învățare automată, respectiv de a le ajusta în situațiile în care notele prezise nu sunt unele de încredere.

2 Noțiuni Teoretice

2.1 Algoritmi de Învățare Automată

2.1.1 Regresia Liniară

Regresia liniară este folosită pentru a prezice relația dintre unul sau mai mulți factori sau variabile. Pentru doua variabile (una de intrare și una de iesire) va fi generată o linie care va fi folosită pentru prezicerea unor noi valori. Daca numărul de variabile crește la 3, lina va deveni un plan, iar pentru oricare număr mai mare de 3, aceasta va deveni un hiperplan. Scopul regresiei liniare este de a găsi hiperplanul care se afla la distanță minimă față de punctele planurilor componente.

2.1.2 SVR

Față de regresia simplă, unde scopul este de a minimiza rata de eroare, în SVR se încearcă încadrarea erorii intr-un anumit prag, ceea ce inseamnă că scopul SVR este de a aproxima cea mai buna valoare într-o marjă data, denumita *epsilon-tube*. Spre deosebire de regresia liniară, unde valorile mari produc rezultate direct proporționale, SVR minimizează impactul acestora și prognozează rezultate centrate în jurul unei valori.

2.1.3 Ridge

Ridge Regression este o versiune liniarizată a regresiei liniare. La funcția de cost originală adăugăm un termen regularizat al cărui scop este de a forță algoritmul de învățare să se potrivească mai bine pe setul de date și să micșoreze variațele estimărilor.

2.2 Eroare Rădăcină Medie Pătratică

Eroare Rădăcină Medie Pătratică (engl. "*root mean square error*", abreviat RMSE) este o metrică folosită pentru a calcula diferența dintre valorile estimate și cele corecte. Calcularea acestuia se face extrăgând radicalul din eroarea medie pătratică. Eroarea medie pătratică reprezintă diferența dintre valoarea prezisă și cea corectă, ridicată la pătrat. Aceasta surprinde abaterea standard a diferenței dintre valorile prezise și cele corecte. Cu cat metrica RMSE este mai mică, cu

atât modelul antrenat are o predicție mai corectă și mai apropiată de adevăr. RMSE este folosită pe parcursul acestui proiect pentru a compara diferențele dintre algoritmi.

3 Funcționarea Aplicației

3.1 Extragerea Atributelor

Aplicația dezvoltată se bazează pe extargerea unor caracteristici speciale:

- numărul de clase;
- numărul de erori de *codying style* (se utilizează framework-ul Cpplint);
- numărul de moșteniri;
- numărul de metodelor virutale;
- numărul membrilor statici;
- numărul variabilelor globale;
- numărul tipului de accesibilitatea a datelor: `public`, `private` și `protected`;
- numărul `define`-urilor;
- numărul datelor de tipul `template` , *Standard Template Library*, `struct`, `enum`;
- numărul de comentarii;
- numărul de funcții; și
- dimensiunea fișierelor antet (cu extensia `.h`) și sursă (cu extensia `.cpp`).

Întrucât setul de date inițial nu avea o formă bine definită, datele inițiale au fost parcurse recursiv și au fost copiate în directorul `data/preprocess` ce respectă formă standard.

Având o structură standardizată, următoarea etapă a constituit extragerea atributelor utile pentru sistem. Atributele menționate mai sus, sunt extrase cu ajutorul unei funcției ce folosește anumite expresii regulate pentru deteminarea atributelor. După parcurgerea tuturor datelor, se generează fișierele de tip CSV `data/features.csv` și `data/test_features.csv`.

Este de menționat că acestă etapă durează proporțional cu setul de date de antrenare, deoarece sunt extrase atibutele menționatele mai sus, iar framework-ul Cpplint necesită timp suplimentar pentru verificarea corectitudinii *codying style*-ului.

În cadrul procesului de reantreanare, sistemul, pe baza fișierului încărcat din pagina web, realizează mutarea fișierelor din directorul `data/raw/train` în directorul standardizat. După acestă operațiune se generează un nou fișier `data/featuresRetrained.csv`. Ulterior, conținutul acestui fișier este adăugat la în fișierul `data/features.csv`.

3.2 Preprocesarea Datelor

Preprocesarea datelor constă în acele operații aplicate asupra atributelor brute, extrase în etapa anterioara. Ea este alcătuită din următorii pași:

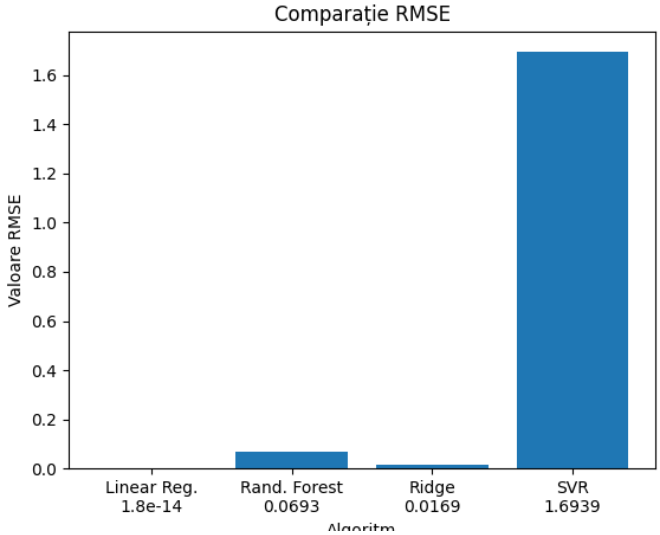
- citirea datelor din fișierele de tip CSV în cadre de date, ce permit o bună manipulare a lor;
- generarea automată de histograme, care permit vizualizarea datelor de pe fiecare coloană a cadrelor;
- renunțarea la anumite coloane care sunt uniforme și care nu oferă niciun plus de informație modelelor de învățare automată; și
- completarea notelor pentru acele întregistrări din setul de date de antrenare care nu le au setate, printr-o abordare bazată pe algoritmul celor mai apropiați K vecini; și
- selectarea celor mai relevante 15 coloane, pe baza unei metrici stabilite.

Preprocesarea apare în două momente ale rulării aplicației: la antrenarea modelelor de învățare automată, când atributele extrase din seturile de date sunt preprocesare înainte de pasarea lor către algoritmii de învățare, și la analiza unui exemplu nou, mai concret la atribuirea unui exemplar nou trimis prin intermediu interfeței grafice.

3.3 Antrenarea Modelelor de Învățare Automată

Problema actuală a fost abordată prin mai multe metode, cu scopul de a vedea diferențele dintre acestea. Astfel, în urma antrenării unui model de random forest regression s-a obținut un RSME de 0.58.

Graficul de mai jos surprinde diferențele (eroarea) dintre datele prezise si cele corecte.



Imagine 1: Diagrama de Componente a Aplicației

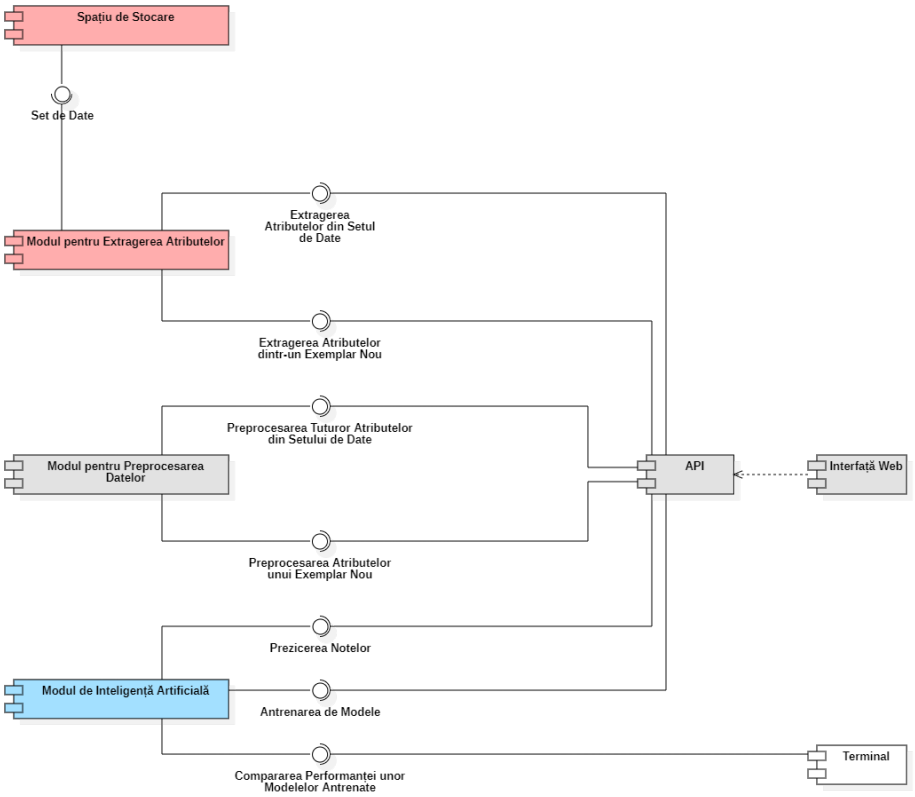
Datele au fost obținute experimental și pot suferi variații în urma repetării testelor. Experimentul a fost efectuat pe același set de date.

Din experimentele efectuate, vedem că pentru setul de date dat, modelul *Random Forest* are cel mai mic RMSE. Acest lucru ne demonstrează performanță acestui algoritm față de cele clasice, precum și că este cel mai potrivit dintre toate cele de mai sus pentru a rezolva problema dată.

4 Arhitectura Aplicației

De menționat este faptul că, în diagrama de mai jos, componentele sunt colorate în funcție de membrul echipei care s-a ocupat de dezvoltarea ei, astfel:

- albastru pentru Apostolescu Ștefan;
- roșu pentru Băjan Ionuț-Mihăiță; și
- gri pentru Iosif George-Andrei.



Imagine 2: Diagrama de Componente a Aplicației

5 Modul de Utilizare al Aplicației

Interfața grafică îi oferă utilizatorului posibilitatea de a genera o predicție pentru un fișier de intrare, precum și de a corecta și reantrena modelul dacă predicția făcută de acesta nu este corectă.

5.1 Prezicerea Notei

Pentru a genera o predicție se incarcă în câmpul "*Archive*" o arhivă ce conține fișierele cu tema studentului. Tema încărcată este apoi procesată, sunt extrase caracteristicile relevante, trecute prin modelul actual și prezisa notă.

5.2 Ajustarea Modelului

Ajustarea modelului presupune inserarea în setul de date actual a informațiilor extrase din fișierele încărcate și a notei considerate corectă, oferită de utilizator. Pe baza acestor informații, parametrii modelului sunt reantrenați și rezultă un model îmbunătățit. Pentru a ajuta modelul, utilizatorul introduce nota corectă in câmpul "*Manually adjusted grade*" și reantrenează modelul pentru noul set de date, apăsând butonul "*Retrain the model*". La apăsarea butonului "*Restart the grading process*" se poate prezice o noua notă pentru aceeași intrare și se poate observa o îmbunătățire a rezultatuli modelului antrenat.

6 Concluzii

În concluzie, aplicația dezvoltată demonstrează utilitatea algoritmilor de învățare automată, în contextul ușurării muncii profesorilor pentru corectarea unor proiecte.