Construirea unei clasificator

Daniela Palcu
Gabriel Matasariu
Grupa 30238

25/05/18

Abstract

Vă vom prezenta construirea unui clasificator capabil sa clasifice semnul de punctuatie "punct" din limba romana. Fiindu-i furnizate anumite caracteristici ale punctului respectiv(feature-uri), acesta trebuie sa fie capabil sa indice clasa din care face parte punctul: punct terminal sau punct neterminal.Acest raport contine 4 sectiuni care descriu problema propusa, generatorul de feature-uri, clasificatorii utilizati si concluziile obtinute in urma acestui proces.

1 Specificatia problemei

Problema propusa in cadrul acestui articol este aceea de a identifica in cadrul unei propozitii tipurile de punct care apar, dupa cum urmeaza:

- punct terminal - un punct care se afla la sfarsitul unei propozitii

  Exp: Ana are mere.

- punct neterminal - un punct care nu se afla la sfarsitul unei propozitii ci face parte dintr-o prescurtare sau apare din alte motive

  Exp: Ana locuieste pe str. Stefan cel Mare.(primul punct nu este un punct terminal)

Datele folosite pentru training-ul clasificatorului respectiv pentru testarea aceastuia se constituie din diverse propozitii/paragrafe luate de pe internet sau construite de noi in incercarea de a acoperi toate cazurile posibile.

2 Generatorul de feature-uri

In incercarea de a stabili cu precizie cat mai mare tipul unui punct, am identificat 6 caracteristici(feature-uri) posibile care ne-ar putea ajuta sa clasificam punctul. Cele 6 feature-uri sunt:

- **NewLine feature(NL)** - acest feature indica daca urmatorul caracter dupa punct este new line `'\n'`
- **NextIsSpace feature(NIS)** - aceste feature indica daca urmatorul caracter dupa punct este un spatiu ' '
  - justificarea utilizarii acestui feature ar fi aceea ca in general, dupa orice semn de punctuatie(deci implicit si dupa punct), regulile de scriere spun ca se lasa un spatiu liber inainte de a incepe o noua propozitie
- **OneLetterBefore feature(OLB)** - acest feature indica daca inainte de punct se afla o singura litera si nu un cuvant
  -justificarea utilizarii acestui feature e aceea ca unele puncte pot aparea in cadrul prescurtarii unui nume (*exp : I. L. Caragiale*),caz in care de cele mai multe ori, punctul nu este unul terminal
- **AbbreviationBefore feature(AB)** - acest feature indica daca inainte de punct se gaseste o prescurtare (generatorului ii este furnizata o lista cu cele mai cunoscute prescurtari)
  -justificarea utilizarii acestui feature este ca in general dupa prescurtari urmeaza un punct
- **SusspensionPoint feature(SP)** - acest feature va indica daca punctul este un punct de suspensie "..."
- **NumberPoint feature(NP)** - acest feature va indica daca punctul face parte din reprezentarea unui anumit numar

Toate feature-urile prezentate mai sus vor returna 'y' in cazul in care punctul respecta conditiile impuse de feature-ul respectiv si 'n' in caz contrar.

Exemplu:

- vom considera propozita : *"Dorel locuieste pe str. Stefan cel Mare, nr. 275 si are nr. de tel. 03423516."*
- in urma rularii generatorului de feature-uri asupra acestei propozitii vom rezultatele din Tabelul 1!

Table 1: Features

	NL	NIS	OLB	AB	SP	NP
1	n	y	n	y	n	n
2	n	y	n	y	n	n
3	n	y	n	y	n	n
4	n	y	n	y	n	n
5	y	n	n	n	n	n

3 Clasificatori

In aceasta sectiune se vor prezenta pasii parcursi in partea a doua a proiectului propus, partea de construire a unui model si testare a acestuia.

Setul de date creat in prima parte a proiectului a fost impartit in doua: un set pentru partea de antrenare(training) si un alt set pentru partea de test. Pentru partea de training, clasificatorii au construit un model care este evaluat utilizand metrica "precision", iar pentru partea de test, modelul construit este testat si evaluat pe un alt set de date. Metrica "precision" a fost aleasa deoarece obiectivul proiectului este acela de a identifica punctele care reprezinta delimitatori de propozitie, iar dorinta este de a identifica acel model care invata sa identifice cel mai bine acest fapt.

Deoarece setul de date este unul relativ mic, cross validation este utilizat in etapa de evaluare, astfel setul de date este impartit in k parti, fiecare parte avand atat rol de parte de antrenare, cat si rol de parte de testare pentru a asigura o precizie cat mai buna a rezultatelor.

Pentru gasirea celui mai bun model, metoda "grid search" a fost utilizata pentru a asigura o performanta cat mai ridicata. Grid Search reprezinta o

metoda prin care sunt cautati acei parametrii(hyperparameters) ai modelului prin care se obtine cel mai bun scor.

Prima parte a fisierului notebook reprezinta partea de pregatire a datelor. In aceasta parte, sunt analizate valorile si tipul datelor care apartin fiecarei trasaturi in parte. Cum toate trasaturile prezinta valori nominale, acestea sunt transformate in valori numerice. Astfel, valoarea y va deveni 1, iar valoarea n va deveni valoarea 0. Aceleasi fapt este aplicat si pentru clasele fiecarui exemplu in parte. Aceasta transformare are loc deoarece unii clasificatori nu utilizeaza valori nominale.

Etapa urmatoare reprezinta gasirea celui mai potrivit model de catre un clasificator in parte. Clasificatorii utilizati, caracteristicile acestora, cat si comparatia lor cu alti clasificatori sunt prezentate in subsectiunile urmatoare. Trebuie mentionat faptul ca pentru fiecare clasificator in parte a fost construit un pipeline utilizand clasa "Pipeline" din sklearn cu scopul de a aplica o transformare si un estimator final. Aceasta transformare este utilizata pentru trasaturi cu scopul de a putea permite selectarea trasaturilor dorite pentru partea de training. In proiect, sunt folosite toate trasaturile, dar pe viitor s-ar putea incerca si o alta abordare.

- **Logistic Regression**

  Primul clasificator folosit este unul dintre cei mai simplii clasificatori, nefiind setati niciunul dintre parametri(hyperparameter). Acesta este utilizat, pentru a putea fi comparat cu ceilalti clasificatori din punctul de vedere al simplitatii.

  Pentru partea de training, scorul obtinut este de : 97,3%.

- **Support Vector Machines**

  Pentru acest clasificator au fost incercati urmatorii parametrii: pentru kernel a fost aleasa atat o varianta lineara cat si o varianta nonlineara. De asemenea, fiecarui kernel in parte ii sunt specifici anumiti parametrii, fiind si acestia utilizati pentru clasificare, de exemplu pt rbf kernel, este specific parametrul "gamma".

  Scorul obtinut este de : 100%.

In proiect a fost utilizata si clasa LinearSVC din sklearn, care se caracterizeaza printr-un kernel liniar.
Scorul obtinut de LinearSVC este de : 97,2%.

- **Decision Tree**

  Parametrii care au fost incercati pentru acest clasificator sunt reprezentati de catre numarul maxim al adancimii copacului, cat si criteriul de impartire care este utilizat pentru algoritm. In urma aplicarii Grid Search, cei mai buni parametrii sunt : max_depth = 5 si entropy = "gini".
  De asemenea, utilizand biblioteca graphviz se poate vizualiza mai bine, procesul de antrenare.

  Scorul obtinut este de: 97,99%

- **Ensemble Learning**

  Clasificatorul folosit este Decision Tree, fiind utilizata clasa din sklearn RandomForest.
  Si pentru Ensemble Learning au fost utilizati aceeasi parametrii care au fost folositi pentru un singur Decision Tree.

  Scorul obtinut este de: 97,99%

- **Neural Network**

  Pentru reteaua neuronala au fost incercati mai multi parametrii, dintre acestia remarcandu-se, numarul de hidden layers si numarul de neuroni din fiecare layer. Astfel au fost abordate mai multe posibilitati, plecand de la o retea simpla, la o retea densa.

  Scorul obtinut este de: 97,99%

Dupa gasirea celor mai buni parametrii ai modelului, modelele au fost testate folosind un alt set de date. In urma testarii, au fost obtinute urmatoarele rezultate:
Logistic Regression : 94,6%
Support Vector Machines : 59,2%

LinearSVM : 93,1%
Decision Tree : 95,09%
Ensemble Learning : 94,47%
Neural Network : 95,04%

Astfel se pot prezenta urmatoarele concluzii:

- Dintre cei 5 clasificatori folositi, cele mai bune rezultate sunt obtinute de catre : Neural Network, Decision Tree, Ensemble Learning, Logistic Regression.

- Despre Support Vector Machine putem spune ca in partea de training realizeaza overfitting, acest lucru fiind vizibil in partea de test, deoarece diferenta de scor obtinuta este foarte mare. In schimb, LinearSVC nu realizeaza overfitting, dar scorul pe care il obtine este mai mic decat cel al celorlalte. Acest lucru poate fi cauzat de faptul ca atat numarul de exemple, cat si numarul de trasaturi este unul relativ de mic sau modelul cel mai potrivit pentru acest set de date nu este unul linear.

- Logistic Regression comparat cu ceilalti, are un scor apropiat fata de ei, deoarece numarul de trasaturi este unul foarte mic, numarul de exemple fiind la fel unul mic, favorizand astfel clasificatorul sa evolueze foarte bine.

- Datele fiind foarte bine structurate, clasificatorii mentionati mai in sus reusesc sa obtina un scor destul de mare.

- Decision Tree comparat cu reteaua neuronala, ar fi mai avantajat de faptul ca putem vizualiza drumul pe care il parcurge pe parcursul clasificarii, astfel putem oberva trasaturile care sunt alese ca puncte de reper, in timp ce reteaua neuronala este ca o cutie neagra in care la intrare introducem date si vedem doar rezultatul. Tot ceea ce se poate spune despre reteaua neuronala este faptul ca scorul cel mai mare obtinut in partea de training este pentru o retea simpla, cu un singur layer si cativa neuroni.

- Ensemble Learning nu evolueaza atat de bine ca si un simplu Decision Tree, datele pe care le-am generat fiind suficiente doar pentru un singur Decision Tree ca sa obtina un rezultat destul de bun.

4 Concluzii

Una din limitarile acestui clasificator consta in dificultatea gasirii unor feature-uri care sa fie cu adevarat decisive in stabilirea tipului de punct. Pentru ca rezultat sa fie cat mai precis ar trebui facuta o analiza asupra intregii propozitii pentru a putea stabili imprejurarile in care apare acel punct(e nevoie de un context mult mai amplu nu doar de niste simple feature-uri).

Cateva imbunatatiri posibile care ar putea fi aduse asupra generatorului ar fi identificarea altor feature-uri pe langa cele deja existente precum si imbunatatirea setului de date de training cu exemple cat mai diverse.