Detectarea efectelor adverse ale medicamentelor în mediul online

Ciocan Dragoș

Coordonator Științific: Conf. Dr. Mădălina Răschip

Cuprins

- problema efectelor adverse ale medicamentelor;
- setul de date;
- preprocesarea textului;
- transformarea textului în vectori;
- antrenarea folosind diferiți clasificatori;
- compararea rezultatelor;
- concluzii;

Problema efectelor adverse ale medicamentelor

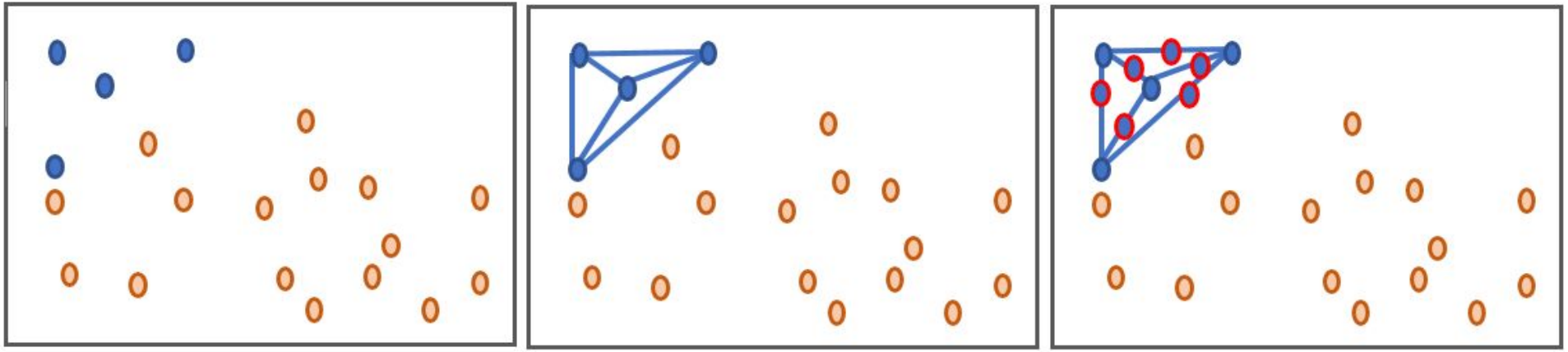
- acestea apar și după lansarea medicamentelor;
- este dificilă colectarea informațiilor, mai ales în mod automat;
- oamenii expun uneori aceste probleme în mediul online, dar aceștia folosesc exprimări colocviale.

Setul de date

- date puține
- foarte nebalansat;
- format în urma căutării de tweet-uri după prezența numelor de medicamente.

Algoritmul SMOTE

- datele nefiind balansate, a trebuit să aplic tehnici de balansare;
- algoritmul SMOTE generează date artificiale între perechi de puncte vecine și apropiate;



Undersampling

- pentru a balansa datele am folosit și metoda „undersampling”;
- aceasta constă în micșorarea numărului de date din clasa majoritară în setul de antrenament (restul folosindu-se în setul de test)
- am folosit proporțiile: 1:2, 1:3, 1:4, 1:5

Prepocesarea textului

- înlocuirea numelor de medicamente și ale efectelor adverse cu simboluri specifice: MED, respectiv ADR;

- înlocuirea link-ului cu simbolul LNK;

- înlocuirea referințelor (cu @) cu REF;

- eliminarea tuturor caracterelor non-alfabetice;

- curățare (spații multiple).

Exemplificare preprocesare

Următorul tweet

@MarkMcGahan80 does the Tysabri make you tired at first after infusion, not that you need an excuse to be tired! Sleep well xx :o) xx

se va transforma în:

REF does the MED make you ADR at first after infusion not that you need an excuse to be ADR sleep well xx o xx

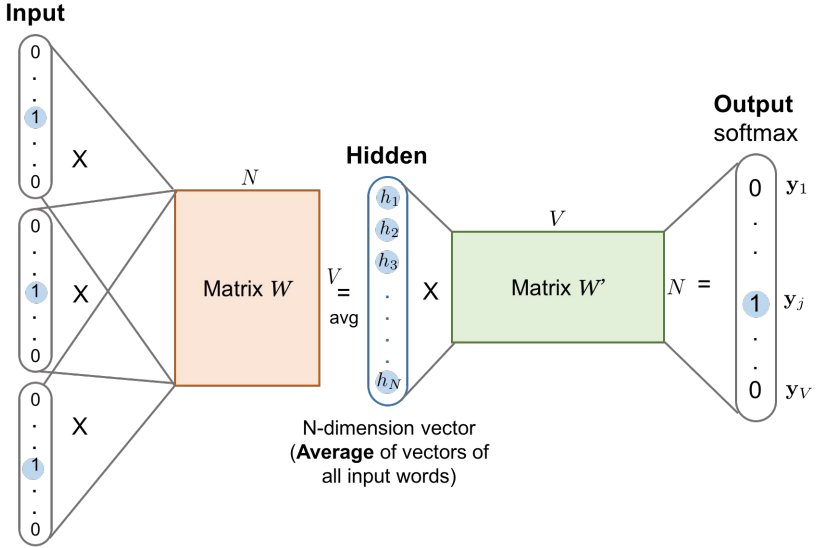
După cum se observă, rămân litere din cadrul emoticoanelor, însă pentru word2vec sau fasttext nu este o problemă.

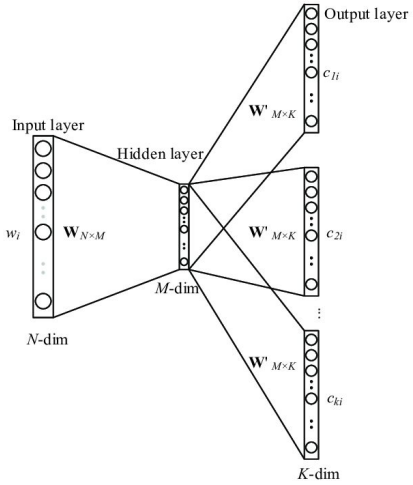
Transformarea textului în vectori

- n-grame;
- word2vec;
- fasttext;

Word2Vec

- în general, este de 2 feluri: Skip-Gram și CBOW;
- se bazează pe antrenarea nesupervizată a unei rețele neuronale;
- Skip-Gram prezice probabilitatea unui context, în timp ce CBOW prezice probabilitatea unui cuvânt, fiind dat un context;





Clasificatorii folosiți

- SVC;
- regresie logistică;
- bayes naiv;
- rețea neuronală.

Support Vector Classifier

- se comportă ca un clasificator liniar;
- dacă nu poate clasifica datele din hiperspațiul curent printr-un hiperplan, „proiectează” datele într-un hiperplan superior;
- această proiectare nu se face propriu-zis, ci doar se calculează relația dintre puncte într-un spațiu superior;
- functiile nucleu (funcțiile de „proiectare”) cele mai folosite sunt funcția polinomială: $(a \times b + r)^{d}$, și rbf (radial basis function): $e^{-y * (a - b) * (a - b)}$, unde a și b reprezintă coordonatele punctelor pentru care vrem să calculăm relația.

Mai multe despre SVC

- funcția polinomială este o funcție poate fi dusă până într-un număr finit de dimensiuni, prin parametrul d
- rbf este o funcție care calculează relațiile dintre puncte într-un spațiu infinit dimensional, astfel: alegem gamma ½ => $e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2-2ab)} = e^{-\frac{1}{2}(a^2+b^2)}e^{ab}$

  după care, printr-o serie Taylor, e^{ab} devine: $e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$

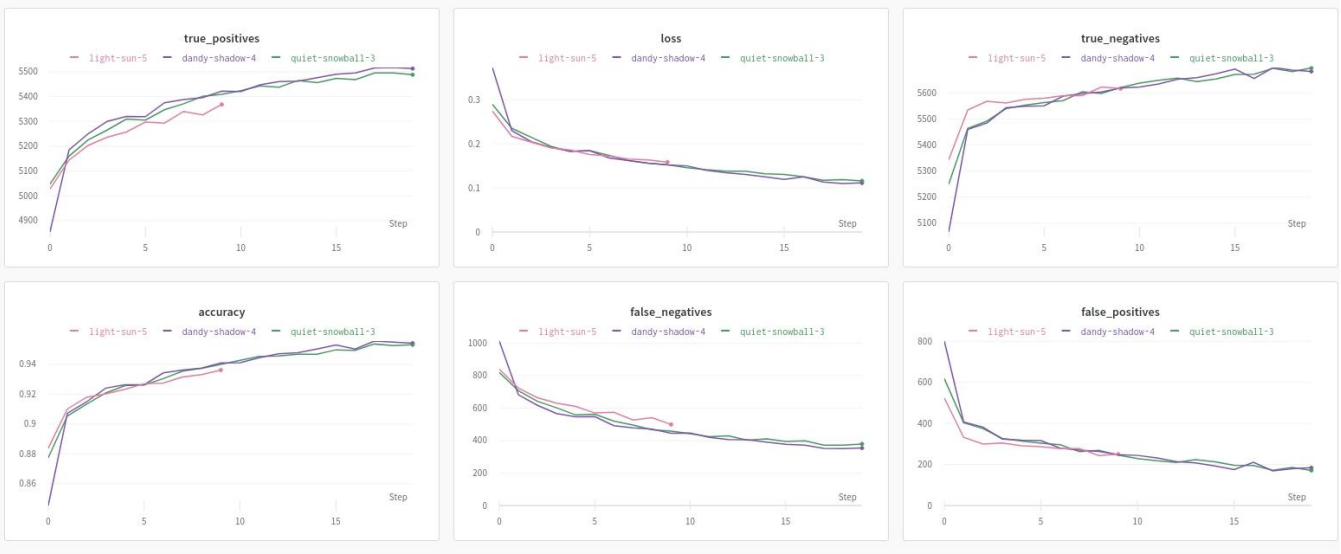
  , iar în final, funcția inițială devine: $e^{-\frac{1}{2}(a-b)^2} = (s, s\sqrt{\frac{1}{1!}}a, s\sqrt{\frac{1}{2!}}a^2, \ldots, s\sqrt{\frac{1}{\infty!}}a^\infty) \cdot (s, s\sqrt{\frac{1}{1!}}b, s\sqrt{\frac{1}{2!}}b^2, \ldots, s\sqrt{\frac{1}{\infty!}}b^\infty)$

  deci este un produs de vectori într-un spațiu infinit.

  precizare: s = $e^{-\frac{1}{2}(a-b)^2}$

Rețea neuronală

- am folosit o rețea neuronală „deep” (DFF - deep feed forward)
- pe primul strat ascuns am folosit 80 de neuroni, iar pe al 2-lea 20
- funcțiile de activare folosite au fost RELU și Sigmoid (pe ultimul strat)
-

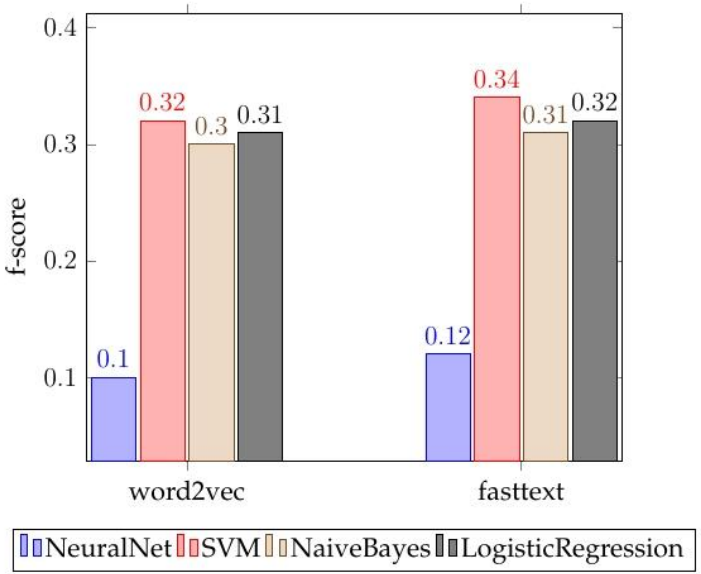


Regresie logistică

- este potrivit pentru clasificare binară;
- se bazează pe funcția logistică: $1 / (1 + e^{-x})$;
- mai exact, se calculează următoarea probabilitate: P(X) = P(Y=1|X), folosind formula $p(X) = e^{b0 + b1*X} / (1 + e^{b0 + b1*X})$
- după o logaritmare și o exponențiere, formula va deveni:
  odds = e^(b0 + b1 * X)
- de aici și vine denumirea de regresie, clasificatorul calculând probabilitatea, nu prezicând exact clasa unei intrări
- pentru a clasifica o intrare X, se setează pragul 0.5

Rezultate

- un grafic care compară cele mai bune rezultate;



Concluzii

- problema rămâne deschisă;
- poate fi îmbunătățit rezultatul cu un set de date mai mare;
- metoda BERT.

Bibliografie

- https://www.researchgate.net/publication/322905432/figure/fig1/AS:614314310373461@1523475353979/The-architecture-of-Skip-gram-model-20.png
- https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/
- https://machinelearningmastery.com/logistic-regression-for-machine-learning/
- https://app.wandb.ai
- https://datasciencecampus.github.io/images/blog/data-from-smote/smote1.png
- https://www.youtube.com/watch?v=Qc5IyLW_hns