IART 2020/21

Relatório Final do 2ºProjeto

Irony detection in tweets - Natural Language Processing

Elementos da Turma 4 Grupo 11:
José Rodrigues - up201708806
Marina Dias - up201806787
Pedro Vale - up201806083

Maio 2021

Especificação do projeto

O tema do nosso trabalho é o “Irony detection in english tweets”

O problema que nos é apresentado é dividido em duas fases:

- **Fase A:** Classificação binária. Dado um tweet, detectar se ele é irónico ou não;
- **Fase B:** Classificação com vários rótulos. Dado um tweet, detectar a que categoria pertence:
  - Ironia verbal com contraste de polaridade
  - Ironia verbal sem contraste de polaridade
  - Ironia situacional
  - Não irónico

Ferramentas e algoritmos

Algoritmos: Na resolução do problema que nos é apresentado, iremos explorar a implementação dos seguintes algoritmos: *Naïve Bayes*, *Árvores de Decisão*, *Redes Neuronais*, *K-ésimo vizinho mais próximo* e *Support Vector Machines*.

Ferramentas: Para este segundo projeto, tomamos a decisão de alterar a nossa língua de programação escolhida para Python, para podermos fazer uso das variadas bibliotecas que nos irão facilitar o desenvolvimento do mesmo. Iremos explorar maioritariamente o *Jupyter Notebook*, mas iremos também explorar outras ferramentas recomendadas como *IPython*, *NLTK*, *Stanza* e *Scikit-learn*.

Detalhes na Preparação dos Dados

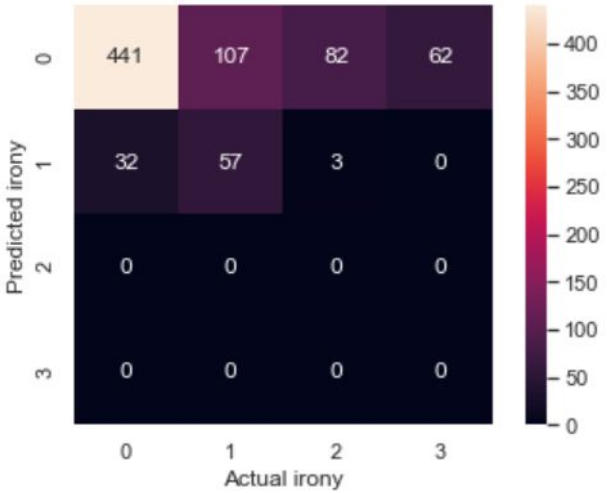
Para tratar dos dados começamos por definir funções para dar parse dos tweets que nos são fornecidos no ficheiro de aprendizagem, tendo os separado por tweets irónicos e não irónicos (task A) e por tweets não irónicos e vários tipos de ironias (task B).

Fizemos uma “limpeza” nos tweets, o que incluiu tirar as identificações (que começavam com @ para identificar alguém), os links, convertemos todas as palavras para minúsculas, retiramos pontuação, apóstrofes, caracteres isolados e as stop words. Depois fizemos uso também de uma função de stemming e duma função de lemmatization, ambas reduzem as palavras à sua palavra raiz, tendo como única diferença que o stemming pode resultar não resultar numa palavra real.

Naïve-Bayes

Naïve-Bayes é um algoritmo que analisa uma dada *dataset*, usando as estatísticas que vai colectando para fazer uma *prediction*. Verifica a frequência de cada palavra e a sua classificação em cada caso. Depois de processar toda a *dataset*, armazena a probabilidade de cada *tweet* abrangendo cada classificação possível para cada palavra analisada. Quando faz uma prediction, verifica a probabilidade para cada classificação e multiplica a probabilidade de todas as classificações de cada palavra. No final, o que resta fazer é escolher a maior probabilidade de todas as opções.

Notamos também que, na *Task B*, devido ao *dataset* não ser equilibrado (um número muito baixo de ironias de classe 2 e 3 em relação aos outros), o algoritmo não fez *predictions* dessas duas classes.

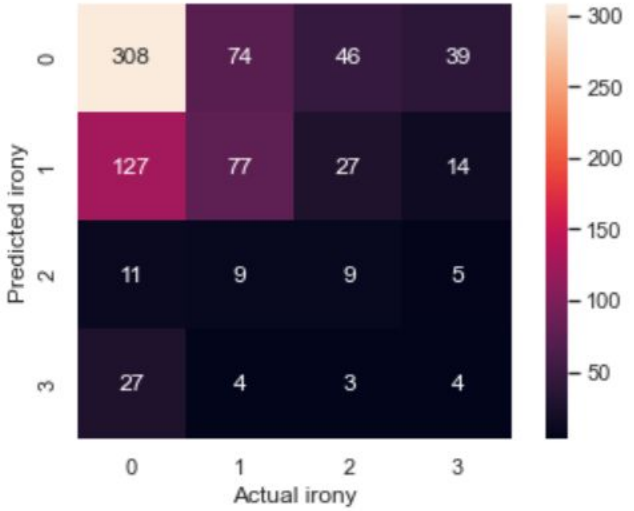



```
The accuracy is 0.63520%
The precision is 0.51409%
The recall is 0.63520%
The f1 is 0.54991%
```

Decision Trees

Para prever o classificação de um dado *dataset*, o algoritmo começa pelo nó inicial (*root node*). Assim, compara os valores do nó inicial com os valores do dataset real e, baseado nessa comparação, segue a ramificação e salta para o próximo nó.

No próximo nó, o algoritmo volta a comparar os valores e continua até encontrar o *leaf node*.

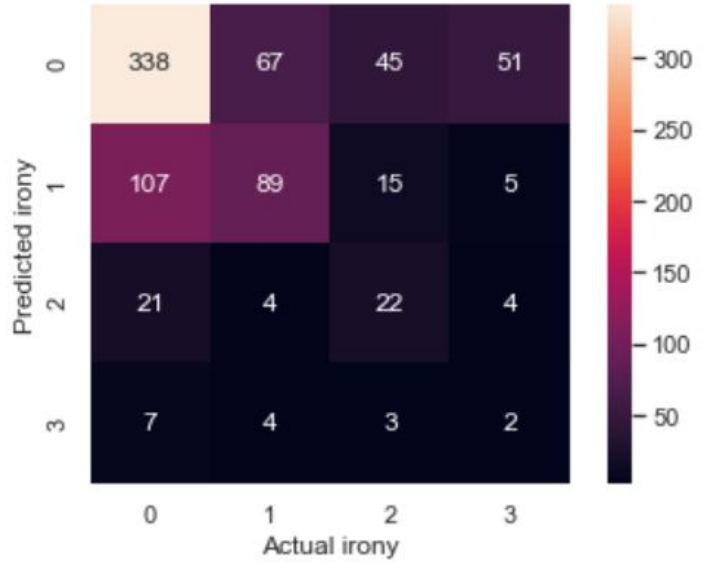



```
The accuracy is 0.50765%
The precision is 0.50067%
The recall is 0.50765%
The f1 is 0.49685%
```

Neural Networks

Neural Networks são inspirado no cérebro humano, pois possuem camadas ocultas que representam as células cerebrais. Estes sistemas "aprendem" a executar tarefas considerando exemplos fornecidos, geralmente sem serem programados com regras específicas. Não reconhecem o objeto que está a ser analisado, mas geram características de identificação a partir dos exemplos que processam.

De notar que este algoritmo, em relação aos outros dois, tem um tempo de treino muito superior.




```
The accuracy is 0.57526%
The precision is 0.54987%
The recall is 0.57526%
The f1 is 0.55585%
```

Comparação

Avaliando os resultados obtidos através da aplicação dos três algoritmos em ambas as *Tasks*, chegámos à conclusão de que o algoritmo *Naive Bayes* produz os melhores resultados nestas tarefas.

Notamos também que as cleaning features aplicadas não resultaram numa diferença muito grande de resultados, mas tal também poderá ser explicado por o *dataset* de teste não ser muito grande.

	Task A			Task B		
	Naive Bayes	Decision Trees	Neural Networks	Naive Bayes	Decision Trees	Neural Networks
Accuracy (%)	0.64158	0.54974	0.62372	0.61862	0.53699	0.54847
Precision (%)	0.67169	0.59135	0.64377	0.49162	0.57305	0.54733
Recall (%)	0.64158	0.54974	0.62372	0.61862	0.53699	0.54847
F1 (%)	0.64509	0.55174	0.62778	0.54305	0.53597	0.53932
Training time (s)	0.03600	0.73903	77.41127	0.03800	1.00751	80.75165
Testing time (s)	0.00600	0.00601	0.00800	0.00601	0.00800	0.00899

1. All tweet cleaning features used combined with only stemming

	Task A			Task B		
	Naive Bayes	Decision Trees	Neural Networks	Naive Bayes	Decision Trees	Neural Networks
Accuracy (%)	0.64413	0.53699	0.60842	0.63520	0.49235	0.58036
Precision (%)	0.67508	0.55120	0.61611	0.51409	0.48129	0.54868
Recall (%)	0.64413	0.53699	0.60842	0.63520	0.49235	0.58036
F1 (%)	0.64756	0.54151	0.61121	0.54991	0.48190	0.55566
Training time (s)	0.05422	0.45407	74.88217	0.05800	0.58800	74.48304
Testing time (s)	0.00900	0.01004	0.01600	0.00900	0.01000	0.01200

2. All tweet cleaning features used combined with only lemmatization

	Task A			Task B		
	Naive Bayes	Decision Trees	Neural Networks	Naive Bayes	Decision Trees	Neural Networks
Accuracy (%)	0.64413	0.56250	0.61607	0.63520	0.50383	0.57526
Precision (%)	0.67508	0.57593	0.62734	0.51409	0.48724	0.54929
Recall (%)	0.64413	0.56250	0.61607	0.63520	0.50383	0.57526
F1 (%)	0.64756	0.56669	0.61952	0.54991	0.49070	0.55471
Training time (s)	0.06100	0.50460	73.88814	0.05600	0.60823	73.53518
Testing time (s)	0.00900	0.01000	0.01300	0.01000	0.01197	0.01500

3. No tweet cleaning features used

Conclusão

- De acordo com os resultados de cada um dos três algoritmos explorados, os tweets não irónicos são os que têm resultados mais elevados (exactidão, precisão e recordação). Isto deveu-se a ter mais tweets para o programa treinar os tweets sem ironia do que os outros tipos de ironia. Isto prova que quanto mais o programa testar, independentemente do algoritmo, melhor será a tomada de decisões de quanto mais treino receber.

- Em geral, é possível dizer que estamos satisfeitos com os resultados. O programa é capaz de fazer o que foi pedido. Este projecto permitiu-nos aprender mais sobre aprendizagem supervisionada e inteligência artificial no contexto do processamento de linguagem natural.

Referências

- Irony Detection in Tweets (medium.com)
- Irony Detection Machine Learning (github)
- Exploring Machine Learning Techniques for Irony Detection
- Naive Bayes classifier
- Slides das Aulas de IART 2020/2021