Irony Detection In English Tweets

André Malheiro up201706280@fe.up.pt
Diogo Gomes up201806572@fe.up.pt
Rúben Almeida up201704618@fe.up.pt
FEUP MIEIC IART Grupo 43 2020/2021

Especificação do Problema e Referências Bibliográficas

Especificação do problema:

Com Irony Detection in English Tweets é pretendido criar diferentes modelos usando técnicas de Machine Learning capazes de detetar tweets em língua inglesa com caraterísticas irónicas.
Este problema foi inicialmente apresentado numa competição de Machine Learning, a SemEval 2018.
Este problema insere-se na categoria de problemas de aprendizagem supervisionada. Para além desta classificação, trata-se de um clássico problema de NLP. Neste tipo de problema a dificuldade reside em dois elementos:

- Filtrar o dataset de treino de tokens inúteis
- Encontrar e escolher os parâmetros corretos entre os diferentes algoritmos de ML.

Referências Bibliográficas:

Apesar da temática da deteção de ironia ser muito específica, ela insere-se num tipo de problemas para o qual existe imensa literatura associada. A análise de sentimentos. São inúmeros os recursos sobre a temática, incluindo de colegas de anos mais avançados.

Os materiais confiáveis e utilizados por nós que merecem realce são os livros:

- Igual, Dra. Laura- Introduction DataScience Python Book
- Bird, Steven - Natural Language Processing with Python

O Dataset

O dataset usado no projeto é o original providenciado pela organização da SemEval aos participantes.
A organização disponibiliza publicamente o dataset para fins educativos. No repositório com os dados a própria fez uma divisão clara entre o train e o test dataset. Como a competição já está finalizada temos acesso ao set de treino com os dados devidamente etiquetados.

Os dados de treino possuem **3833 tweets. Só existe uma categoria de classificação binária, a "label"** em que 0 é não irónico e 1 é irónico.

Os dados de teste possuem **783 tweets. Também só com a categoria "label"**

O dataset está balanceado e não requer pré-processamento quanto à procura de valores null, pois todos estão preenchidos.

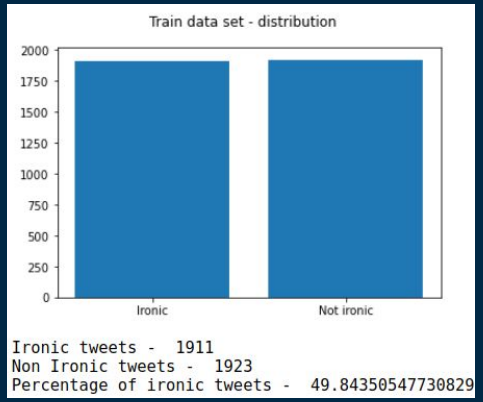

Fig.1: Distribuição dos tweets de train por labels.

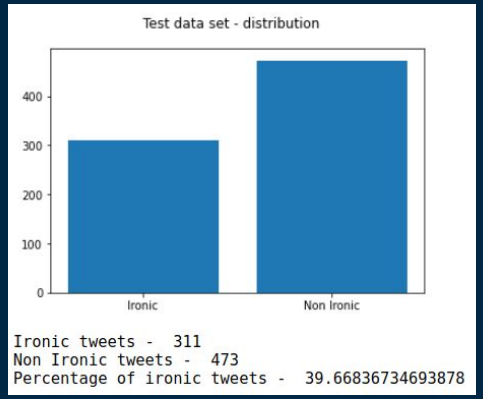

Fig.2: Distribuição dos tweets de test por labels.

Bibliotecas e Algoritmos a Utilizar

Fazendo uso da biblioteca scikit learn podemos executar uma gama muito variada e completa de algoritmos de ML. Para além das 3 bibliotecas requeridas no trabalho, **iremos introduzir tantos algoritmos quantos forem relevantes para obter bons resultados**.

Os três que iremos utilizar/já fazemos utilização são os seguintes:

Naïve Bayes **Neural Networks** **Decision Trees**

No caso de algoritmos como as decision trees que permitem usar a técnica de Grid Search, ela será explorada, de forma a encontrar os melhores parâmetros para executar o modelo.

A linguagem de programação utilizado será Python3 e é usado o ambiente Jupyter Notebook usando como suporte o anaconda para gestão de dependências de Python.

Para além de utilizar o Scikit learn, iremos utilizar outras bibliotecas como:

- Pandas: Para carregar e manipular os dados de uma forma estruturada
- Numpy: Requisitada pelo Pandas para operações sobre arrays
- NLTK: Bibliotecas de NLP que introduzem os Stemmers e os Lemmatizers
- **contractions:** Biblioteca de Python que expandem os vocábulos ingleses que recebem apóstrofe para serem encurtados
- Seaborn e Matplotlib: Para poder fazer gráficos da matriz de confusão de forma user-friendly

Trabalho já realizado

Dado o nosso projeto derivar de uma competição de ML já concretizada, podemos utilizar o ranking da mesma para manter um nível de comparação da qualidade do nosso trabalho e da utilidade do nosso pré processamento a cada estádio.

O vencedor da SemEval2018 ChuhanWu obteve, para o mesmo dataset que nós, os seguintes resultados:

Accuracy:0.7374 **Precision:0.6304** **Recall:0.8006** **F1-Score:0.7054**

De forma a analisar logo à priori a qualidade dos dados, aplicamos um Naïve Bayes aos dados sem qualquer pré-processamento, os resultado foram muito fracos em comparação com os acima mencionados.

	Our Results	Winners Result
Accuracy	0.514031	0.7347
Precision	0.550978	0.6304
F1-Value	0.516916	0.8006

Fig.3 Resultados preliminares

Até ao momento o nosso grupo já conseguiu aplicar algumas estratégias de pré processamento textual aos tweets, bem como técnicas de Lemmatization e Stemming. **Com esse esforço, conseguimos melhorar os resultados obtidos para:**

	Our Results	Winners Result
Accuracy	0.647959	0.7347
Precision	0.643146	0.6304
F1-Value	0.644722	0.8006

Fig.4 Melhores Resultados Atuais obtidos

Pré processamento e trabalho futuro

De forma a ter sido possível obter os melhoramentos apresentados foi necessário aplicar uma cadeia de pré processamentos nos tweets de forma a normalizar os tweets. Para além dessas modificações foi aplicado a biblioteca contractions, bem como Stemming e Lemmatization utilizando a biblioteca NLTK.

Pré processamentos introduzidos:

- Conversão para lowercase
- Remoção das mentions de twitter. Palavra que começa por "@"
- Remoção do "#" nas Hashtags
- Remoção de algarismos
- Remoção de links HTTP

Todo o trabalho posterior irá orbitar em torno quer de mais pré processamento, quer de *tunning* dos algoritmos utilizados.

No que diz respeito ao pré processamento, iremos ainda introduzir:

- Normalizar os emojis para keywords chave --> Poderá ser necessário introduzir biblioteca especializada.
- Unir os dados de teste e treino providenciados de forma a ter mais dados disponível para treinar o modelo, usando técnicas de cross validation para garantir que o fenômeno de overfitting é mitigado.

No que diz respeito a tunning de algoritmos, iremos:

- Introduzir Grid Search,
- Configurar a Neural Network e forma a obtermos os melhores resultados possíveis de forma consistente.