Laboratório Nº 3: Text Mining - Métricas e Início do Processamento do Conjunto de Treino

Extração Automática de Informação

Prof. Joaquim Filipe
Eng. Filipe Mariano

Objetivos

- Continuação da implementação de vários passos de pré-processamento de modo a limpar o conteúdo dos documentos
- Introdução ao conceito de Term Frequency (TF), Inverse Document Frequency (IDF) e Term Frequency-Inverse Document Frequency (TF-IDF).

1. Contagens

Uma das *features* mais básicas que podem ser extraídas é o número de palavras ou caracteres de um texto. Estas *features* básicas podem ser tidas em conta, visto que, na generalidade dos casos, os sentimentos expressos de uma forma negativa podem conter um menor número de palavras do que os positivos. Contudo, esta afirmação pode também depender do domínio de aplicação do problema estudado.

2. Term Frequency

O conceito de *Term Frequency* (TF) é simplesmente o rácio entre a contagem do termo presente num texto e o número total de termos do texto. Desta forma, pode ser generalizado da seguinte forma:

```
TF = (Número de vezes que o termo T ocorre num documento) / (Número de termos do documento)
```

3. Inverse Document Frequency

Por sua vez, o *Inverse Document Frequency* (IDF) corresponde a uma métrica para determinar o quão raro um termo é entre os diversos documentos. O valor calculado tem o efeito de destacar palavras que são distintas (contêm informação útil) em um determinado documento. Assim, o IDF de um termo raro é alto, enquanto o de um termo frequente é provavelmente baixo.

O cálculo do IDF é feito através da seguinte fórmula:

$IDF_{(t)} = \log (N / d_{(t)})$

Em que `t` representa o termo, `N` o número de documentos do corpus e `d` o número de documentos em que o termo `t` ocorre.

4. TF-IDF

O *Term Frequency-Inverse Document Frequency* (TF-IDF) é uma estatística que tem como objetivo refletir o quão importante um termo é num documento, relativamente ao *corpus*. Ou seja, um valor alto de TF-IDF é alcançado através de uma frequência elevada do termo num documento e uma frequência baixa do mesmo termo em toda a coleção de documentos.

O cálculo do TF-IDF é feito através da seguinte fórmula:

$TF\text{-}IDF_{(t)} = TF_{(t)} * IDF_{(t)}$

Em que `t` representa o termo, `TF` a frequência do termo no documento e o `IDF` a frequência inversa do documento para esse termo.

5. Exercícios

1. Proceder à contagem de n-gramas de palavras existentes num texto. Crie um módulo na diretoria `classification` chamado `counting.js` que deverá exportar duas funções: `countByNGram` e `countBySize`.
a. A função `countByNGram` tem dois parâmetros que são um texto e um inteiro `n`. A partir do texto deverá ser retornado o nº total de tokens de acordo com o `n` recebido. Exemplo: Se o `n` for 2 será o nº total de sequências de 2 palavras seguidas existentes no texto.
b. A função `countBySize` tem um parâmetro que é já o array de tokens e deverá devolver o nº total de tokens existente.

2. No mesmo módulo `counting.js` criar uma função `numberOfOccurrences` que receberá um array de tokens e um token e irá devolver o número de ocorrências desse token.

3. Exporte uma função `exists` no módulo `counting.js` para que verifique a existência de um token num array de tokens, devolvendo um valor de verdadeiro ou falso.

4. Ainda no módulo `counting.js` criar uma função `tf` (term frequency) que receberá um token e um array de tokens e irá devolver a frequência desse token. Devem contar o número de ocorrências do token (já calculado a partir do `numberOfOccurrences`), assim como deverão calcular o nº total de tokens existente no array.

5. No módulo `counting.js` criar uma função `idf` que receberá um valor `N` (nº de documentos) e o valor de $d_{(t)}$ e irá devolver o cálculo do *inverse document frequency* (ver secção 3).

6. No módulo `counting.js` criar uma função `tfidf` que receberá o valor de `tf` e o valor `idf` e calculará o *term frequency - inverse document frequency* (ver secção 4).

7. No módulo `train.js` criar uma função `process`, incorporando o módulo de pré-processamento anteriormente criado `preprocessing.js`, de modo a realizar o seguinte:
a. Chamar a função `getTrainingSet` para obter todos os textos que estão marcados para serem utilizados como conjunto de treino. Deve dividir o processamento por classes, pelo que deverá processar todos os textos de cada classe, ou seja, primeiro os da classe `Positive` e depois os textos da classe `Negative`.
b. Para cada texto deverá utilizar a função definida no laboratório anterior para aplicar todas as técnicas referidas de pré-processamento abordadas para `n=1` e `n=2`, ou seja, para sequências de apenas 1 palavra e sequências de 2 palavras. Relembrar que este pré-processamento (feito na aula passada) deveria de:
i. Remover as *stopwords* existentes em cada texto.
ii. Remover pontuação, números e espaços a mais existentes em cada texto.
iii. Normalizar as palavras existentes de acordo com o algoritmo de *Porter Stemming*.

8. No módulo criado na diretoria `./test` juntar uma propriedade `tf` com o cálculo do tf para cada token encontrado em cada texto. Gravar o conteúdo do array de objetos criado anteriormente e agora com a nova propriedade `TF` para um ficheiro `preprocessing_tf.json`.

Nota: Este ficheiro `preprocessing_tf.json` poderá ser submetido através do Moodle para validar se o pré-processamento se encontra a funcionar corretamente, assim como, o cálculo do TF.