Instituto Superior de Engenharia do Porto

Modelo de representação de texto mais adequado à classificação

Alexandra Isabel Magalhães Alves

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Área de especialização em

Tecnologias do Conhecimento e Decisão

Orientador: Doutor Nuno Filipe Fonseca Vasconcelos Escudeiro

JÚRI:

Presidente:

Doutora Maria de Fátima Coutinho Rodrigues, Professora Coordenadora, Instituto Superior de Engenharia do Porto

Vogais:

Doutor Nuno Alexandre Pinto da Silva, Professor Coordenador, Instituto Superior de Engenharia do Porto

Doutor Nuno Filipe Fonseca Vasconcelos Escudeiro, Equiparado Assistente 2ºTrienio D/M, Instituto Superior de Engenharia do Porto

Porto, Novembro 2010

Dedico este projecto ao meu namorado e aos meus pais por todo o apoio e carinho.

Agradecimentos

Agradeço antes de tudo aos meus pais que sempre se sacrificaram para que os seus filhos tivessem um futuro melhor, sempre nos aconselharam e apoiaram nas decisões e obstáculos mais difíceis.

Ao meu orientador o Eng. Nuno Escudeiro, um obrigado muito especial por ter sido incansável na ajuda e coordenação prestada ao longo da realização deste trabalho. A ajuda dele foi fulcral para a conclusão deste trabalho. Muito obrigado por tudo.

A todos os meus amigos, colegas de curso, familiares e professores que sempre me ajudaram a ultrapassar as dificuldades que foram surgindo, ensinaram e apoiaram ao longo de toda a minha vida académica.

Ao meu irmão pela ajuda fornecida na revisão deste relatório.

E por último, o meu muito obrigado, ao meu namorado, Daniel Gonçalves, por toda a ajuda prestada, disponibilidade, conselhos, apoio e paciência ao longo da realização deste projecto.

A todos o meu muito obrigado.

Resumo

A área de *text mining*, mais especificamente a classificação de texto, é alvo de muito trabalho e avanços nos últimos anos. Esta área tornou-se cada vez mais importante com a evolução da tecnologia e assume grande relevância na actual sociedade de informação.

Um dos problemas ainda presente nesta área baseia-se na classificação de texto para categorias que representam conceitos muito próximos e difíceis de distinguir quando se considera o modelo tradicional do "saco de palavras" (*bag-of-word*). Estes problemas surgem sobretudo quando se classifica texto referente a um mesmo tema, como por exemplo: respostas em texto livre dadas por alunos ao responder a perguntas abertas, comentários a um mesmo filme, etc.

Neste trabalho, é apresentado um estudo sobre todo o processo de classificação de texto que permite avaliar as tarefas e fases mais importantes para a definição de uma metodologia útil para o problema enunciado.

A abordagem adoptada neste trabalho baseou-se na ideia de que os resultados da classificação podem melhorar caso se considerem representações de texto mais elaboradas que o simples modelo *bag-of-words*. Foram então criados diversos modelos de representação dos documentos - envolvendo os modelos de *bag-of-words*, *NGrams* e *Pos-Tag* - todos eles baseados em diversas combinações de tarefas de pré-processamento. Os classificadores usados para a classificação dos documentos foram o *support vector machine e k-nearest neighbour*. Por fim, para a avaliação da classificação foi aplicada a técnica de validação cruzada para reduzir a variabilidade das estimativas das medidas de desempenho analisadas (abrangência e precisão).

Foi possível concluir que os modelos de representação que parecem mais adequados, para a resolução do problema proposto, são os modelos *bag-of-words* construídos com base em nomes. E, que os classificadores *support vector machine* apresentam melhor desempenho que o classificadores *k-nearest neighbour*.

Palavras-chave: Classificação de texto, modelo de representação.

Abstract

The Text Mining area, specifically the text classification is the subject of recent research advances. This area has become increasingly important with the evolution of technology and is highly relevant in today's information society.

One of the problems still present in this area is based on the text classification into categories that represent concepts very close and difficult to distinguish when one considers the traditional model of "bag of words". These problems arise especially when classifying text referring to the same theme, for example: free text responses given by students to answer open questions, comments to the same movie, etc.

In this work, we present a study on the whole process of text classification that allows to assess the tasks and the most important stages for the definition of a useful approach to the problem stated.

The approach adopted in this study was based on the idea that classification results may improve when considering more elaborated text representations than the simple bag-of-words model. Afterwards we created several representation models of documents involving bag-of-words models, NGrams and POS-tags. All of them were based on various combinations of pre-processing tasks that is considered one of the most important stages of classification. The classifiers used for document classification were the support vector machine and k-nearest neighbor. Finally, we applied cross-validation to estimate classification performance measures (precision and recall).

We concluded that the representation models that seem most appropriate for solving the proposed problem are the bag-of-words models based on names. Furthermore, it was also conclude that the support vector machine classifiers outperform the k-nearest neighbor classifiers.

Keywords: Text classification, representation models.

Índice

Índice de Fíguras

Índice de Tabelas

Acrónimos

BOW	Bag-of-words, saco de palavras
CLAWS	Constituent Likelihood Automatic Word-tagging System
HTML	Hypertext Markup Language
IDF	Inverse Document Frequency
IE	Information Extration
IR	Information Retrieval
JVM	Java Virtual Machine
KM	Kernel Methods
KNN	K-Nearest Neighbour
LSI	Latent Semantic Indexing
NN	Neural Networks
POS	Part-of-speech
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TF	Term Frequency
TFxIDF	Term Frequency – Inverse Document Frequency
TM	Text Mining

1. Introdução

1.1. Apresentação do problema

Cada vez mais, o *text mining* e mais particularmente a área de classificação de texto têm uma grande importância nas sociedades de informação de hoje. Com o crescimento das tecnologias e do armazenamento digital torna-se importante a manipulação de informação não estruturada.

O problema abordado neste trabalho baseia-se na classificação de texto em categorias que representam conceitos muito próximos e difíceis de distinguir quando se considera o modelo tradicional de "saco de palavras", em inglês, *bag-of-words* (BOW).

O modelo BOW é bastante popular, pois é facilmente aplicável a texto proveniente de diversas fontes. As palavras são consideradas individualmente, como tal tem a desvantagem de ignorar a informação posicional e não reconhecer a semântica do texto. Através duma contagem das palavras que aparecem nos documentos é possível identificar os tópicos principais focados nos textos. Este modelo é bastante eficaz em situações em que a ordem das palavras no texto não interessa e quando se pretende que as categorias estejam relacionadas com tópicos [Santos, 2006]. Mas, quando se pretende classificar textos em função, por exemplo, das emoções que eles transmitem ou do facto do texto fornecer uma opinião positiva ou negativa sobre o mesmo tema, esta abordagem já não é tão adequada.

1.2. Motivações

A classificação de texto tem evoluído bastante com a evolução das tecnologias e hoje em dia, com o crescimento da Web 2.0, torna-se fundamental conseguir interpretar aquilo que o utilizador transmite.

Com o aparecimento da Web 2.0, o utilizador passou a ter um papel activo na Web. Agora o utilizador pode interagir na Web e não só ver informação. O utilizador passou a poder escrever para a Web, a criar os seus próprios sites (*blogs*) e a opinar em diversos sites.

Outra grande evolução foi o aparecimento das redes sociais e fóruns em que é possível a partilha de informação e ligação com outras pessoas.
O crescimento das redes sociais, dos fóruns e dos sites que permitam ao utilizador dar a sua opinião sobre algo, torna a classificação das suas opiniões e comentários cada vez mais importante. Esta foi uma das motivações que levou ao desenvolvimento deste trabalho.

Outra das motivações que deu origem ao problema proposto surgiu da ideia de correcções semi-automáticas de respostas em texto livre dadas por alunos ao responder a perguntas abertas. Este problema reflecte um conjunto de características particulares, uma vez que todas as respostas se referem ao mesmo tema e torna-se difícil distinguir automaticamente a qualidade de duas respostas.

Os inquéritos com respostas abertas são outro dos exemplos que reflectem o problema aqui apresentado.

Nestes casos torna-se útil distinguir a qualidade das respostas para as conseguir categorizar correctamente.

A hipótese idealizada, para a resolução do problema apresentado, é considerar representações dos textos mais elaboradas do que as do simples modelo BOW, para melhorar os resultados com a precisão da classificação.

1.3. Objectivos e abordagem do problema

Propõe-se neste trabalho, estudar e analisar várias alternativas para a representação de texto a fim de avaliar a hipótese colocada.

É apresentado um estudo de todo o processo de classificação de texto, desde o pré-processamento. A etapa inicial de pré-processamento é considerada como uma das etapas mais importantes visto que terá impacto em todas as fases posteriores. Para a representação dos documentos, onde aqui serão usados os modelos de BOW, *NGrams* e *Pos-tag*, sendo estes também combinados entre si. Para a classificação dos documentos, foram usados os classificadores *support vector machine* (SVM) *e k-nearest neighbour* (KNN). Por fim, são apresentados os métodos de avaliação dos resultados obtidos, tais como a precisão e a abrangência.

Como era de esperar, a fase mais envolvente de todo o processo consta do pré-processamento e representação dos documentos. Foi decidido criar vários modelos de representação de documentos que permitam encontrar a melhor forma de resolução do problema proposto. Estes vários modelos resultam de todas as combinações possíveis entre as várias fases de pré-processamento, isto é, a remoção de pontuação, a remoção de *stop-words* e o *stemming* e de modelos de representação: BOW, *NGrams* e *Pos-Tag*. Para os *NGrams* foram considerados valores para n=2,3,4 e 5 e para o *pos-tag* foram examinadas as categorias gramaticais: nomes, verbos e adjectivos.

Para cada modelo de representação foi criada a respectiva matriz TFxIDF – *Term frequency x Inverse document frequency*, que foi posteriormente utilizada como input para os classificadores referidos SVM e KNN (neste caso considerando 3, 7 e 11 vizinhos).

Finalmente, para a avaliação dos resultados foi aplicada a técnica de validação cruzada para reduzir a variabilidade das estimativas das medidas de desempenho analisadas (precisão e abrangência).

A análise dos resultados fornece indicações sobre os modelos de representação a adoptar, assim como sobre os algoritmos de classificação mais adequados.

1.4. Contributos deste trabalho

Este trabalho foi motivado pela dificuldade inerente a problemas de classificação de texto, em que as classes não estão relacionadas directamente com o tópico/tema a que o texto se refere mas com outros aspectos. No caso concreto do trabalho presente, as classes respeitam a juízos de valor e a opiniões pessoais sobre um determinado filme. O problema de classificação estudado pretende distinguir entre opiniões favoráveis, opiniões desfavoráveis e opiniões irrelevantes.
O problema de classificação em si é ainda mais difícil pelo facto de termos usado um corpus constituído por críticas a filmes diversos (as críticas não dizem todas respeito ao mesmo filme).
Em concreto, o problema de classificação consiste em aprender um classificador para o conceito crítica positiva, negativa e imparcial a partir de um corpus constituído por críticas referentes a vários filmes.

Este trabalho focou-se na avaliação de vários modelos de representação dos textos com o objectivo de perceber qual a melhor forma de representação que produz melhores resultados de classificação.

Podemos referir, no âmbito do presente trabalho, os seguintes contributos:

1. O principal contributo é o estudo comparativo dos vários modelos de representação de textos, considerando combinações de tarefas típicas de pré-processamento (pontuação, *stop-words*, *stemming*) e de tipos de atributos (palavras, *Ngrams*, *Pos-Tag*).
2. A avaliação de dois algoritmos de classificação (SVM e KNN) para os vários modelos de representação é outro contributo gerado.
3. O código produzido em R, que pode ser reutilizável, é também uma mais-valia do trabalho efectuado.
4. Por fim, a compilação do corpus em si é mais um contributo deste trabalho.

1.5. Tecnologias utilizadas

A ferramenta usada neste trabalho foi o **R** - *Project for Statistical Computing*.

R é uma linguagem e um ambiente de desenvolvimento integrado para cálculos estatísticos e gráficos.

O código fonte do R está disponível sob a licença GNU GPL e as versões binárias pré-compiladas são fornecidas para Windows, Macintosh, e muitos sistemas operacionais Unix/Linux.

R é também altamente expansível com o uso dos pacotes, que são bibliotecas para funções específicas ou áreas de estudo específicas.

Um conjunto de pacotes é incluído com a instalação de R, com muitos outros disponíveis na rede de distribuição do R (em inglês CRAN). [REF11]

O R possui uma livraria específica para o *text mining* (library TM) entre outras que possibilitaram a realização de experiencias práticas.

O site oficial desta ferramenta é: www.r-project.org.

1.6. Estrutura do relatório

Este trabalho encontra-se estruturado por capítulos para melhor compreensão do leitor. Sendo no primeiro capítulo introduzido o problema e sucintamente apresentada a abordagem do problema, assim como os resultados obtidos. Também, são descritos os contributos deste trabalho, a tecnologia usada e a estrutura do relatório.

Num segundo capítulo é exposto um estudo de arte sobre todo o processo de classificação de texto. Sendo num terceiro capítulo sublinhados alguns dos modelos de representação de documentos mais frequentes.

Posteriormente, é definida num quarto capítulo, a abordagem seguida para a resolução do problema, assim como a análise dos problemas que foram surgindo ao longo do trabalho. Foi também neste capítulo que foi definido o corpus, conjunto de documentos em *text mining*, a ser utilizado e as características que ele deveria apresentar.

No quinto capítulo, descreve-se a avaliação dos resultados, isto é, apresenta-se o plano experimental implementado, os resultados obtidos e a análise/discussão dos resultados.

Por fim, são apresentadas as conclusões obtidas da realização deste projecto, assim como as recomendações e linhas para trabalho futuro.

2. Classificação de texto

2.1. Text Mining

O *Text Mining* (TM) é o processo de extracção de conhecimento interessante, útil, novo e relevante a partir de dados não estruturados ou semi-estruturados.
É geralmente definido como um processo que utiliza métodos para navegar, organizar, encontrar e descobrir informação em bases textuais escritas em linguagem natural [REF1].

O *Text Mining* surgiu da necessidade de descobrir, de forma automática, informações (padrões e anomalias) em textos. Estima-se que em média, 80% da informação nas empresas é não estruturada e está armazenada em formato textual, tais como documentos, manuais, livros, emails, etc [Rodrigues, 2008]. Devido à sobrecarga de informação torna-se impossível tratá-la de forma manual, daí a relevância e crescimento da área do TM.

Esta área também ganhou importância com o crescimento da internet e dos mecanismos de procura, que hoje em dia fazem parte do dia-a-dia de muitas pessoas.

O *Text Mining* pode ser visto como uma extensão do *Data Mining* [Tan, 1999] mas aplicado a dados textuais.

O uso desta tecnologia permite sobretudo:

- Recuperação de informação (*Information Retrieval* – IR)
- Extracção de informação (*Information Extration* – IE)
- Classificação de textos
- Resumo de documentos
- Realizar análises qualitativas e quantitativas em documentos de texto
- etc

Através das ferramentas e métodos típicos do TM é possível:

- Classificar documentos automaticamente
- Ter uma visão do conteúdo de um documento sem o ler
- Alimentar automaticamente bases de dados
- Melhorar o índice de um motor de pesquisa para melhorar a consulta de documentos

- etc

Para a resolução do problema proposto, vamos recorrer ao uso do TM, mais particularmente às técnicas de classificação de texto, uma vez que se pretende classificar respostas em texto livre. Como tal, nos seguintes capítulos é focada e analisada a área de classificação de texto.

2.2. Definição da classificação de texto

A classificação de texto refere-se à problemática de atribuir automaticamente categorias pré-definidas a documentos de texto livre. [Yang, 1999; Lewis, 1991]

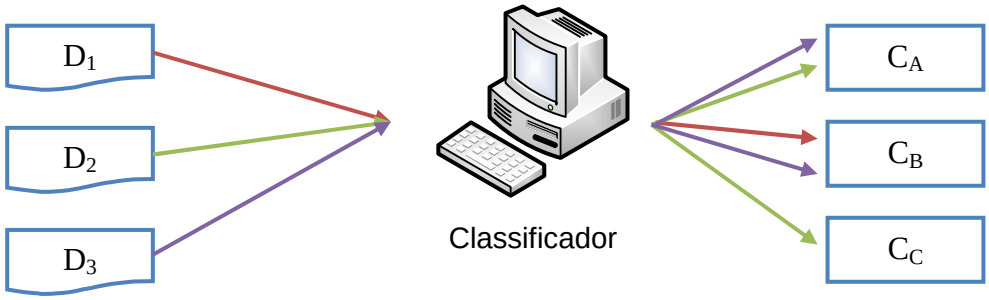


Figura 1 - *Text Classification*

Em geral, os trabalhos de classificação de textos procuram encontrar o tema central de um texto (ou temas, se houver mais de um).

Para além do termo **classificação de textos** (*text classification*) é comum na literatura encontrar-se termos como **categorização de textos** (*text categorization*), **classificação de documentos** (*document classification*), **categorização de documentos** (*document categorization*), **descobrimento de tópicos** (*topic spotting*) [F. Sebastiani, 2002].

A primeira tarefa da classificação de texto baseia-se em transformar os documentos, que normalmente são representados por sequências de caracteres, numa representação adequada para a tarefa de classificação ou melhor dizendo, para os algoritmos de aprendizagem que, na maioria dos casos, precisam dos dados de entrada estruturados.

Um número elevado de termos pode diminuir significativamente o processo de aprendizagem do classificador, assim como um conjunto demasiado pequeno [Dos Santos, 2008]. Torna-se importante, no processo de classificação de texto, reduzir a dimensão dos documentos e vectores para facilitar a classificação. O método da selecção de características (*feature selection*) é uma das tarefas mais importante na redução da dimensionalidade.

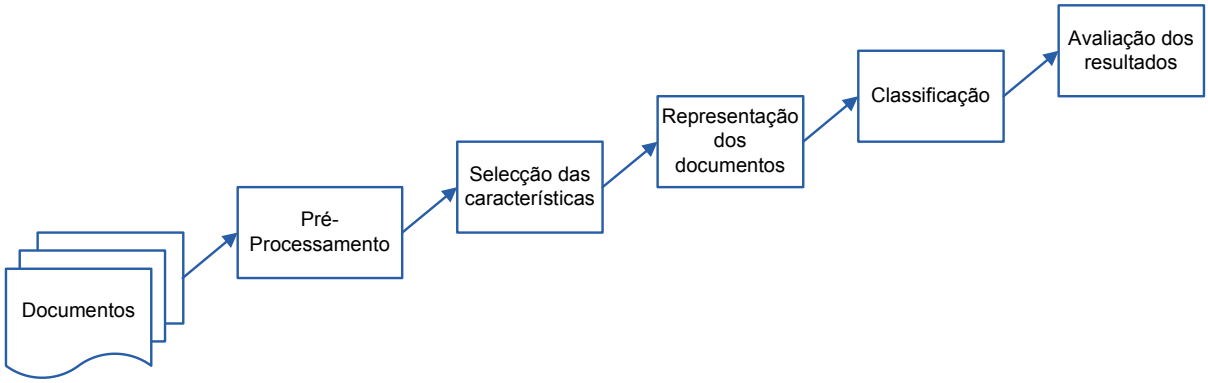


Figura 2 - Etapas da classificação de texto

Resumidamente, os passos que normalmente fazem parte do processo de classificação de texto são:

- Pré-processamento: permite tratar o documento inicial, transformando-o de forma a facilitar o processo de classificação através de métodos computacionais.
- Classificação: são criados classificadores que permitem atribuir categorias a documentos. É necessário avaliar vários métodos/algoritmos a fim de descobrir a melhor forma de resolver o problema proposto neste trabalho.
- Avaliação dos resultados: por fim é necessário analisar e avaliar os resultados obtidos.

2.3. Pré - processamento

O pré-processamento pode ser definido como um processo transformativo, aplicado normalmente para reduzir o número de termos de um documento, de forma a obter uma representação dos documentos mais adequada para as fases seguintes. As tarefas de pré-processamento, desempenham um papel importante na classificação de textos, visto poderem afectar de forma determinante as fases posteriores. [Dos Santos, 2008]

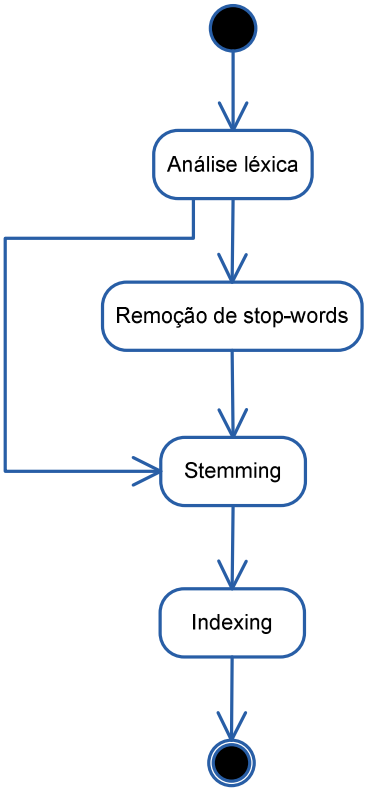


Figura 3 – Tarefas de pré-processamento

Como demonstra a Figura 3, existem diversas tarefas que podem ser realizadas na fase de pré-processamento do texto. No entanto, estes são dependentes dos modelos e das técnicas que vão ser posteriormente aplicados. Esta fase pode ser considerada a mais importante num processo de classificação de texto.

- Análise léxica

Eliminar pontuação, acentos, espaços em excesso, converter o texto todo para maiúsculas ou minúsculas.

- Remoção de *Stop-words*

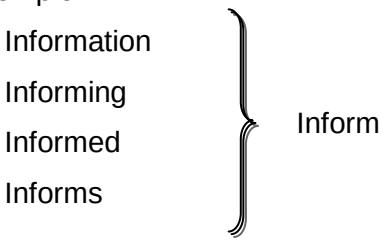
As *stop-words* são os termos considerados irrelevantes e somente desempenham um papel funcional no texto. São palavras que dependem do idioma e podem também depender do tópico de interesse. São normalmente removidas para melhorar os métodos de processamento. Em vários casos a remoção das *stop-words* não traz consequências nefastas, palavras como: de, o, a, para, não são úteis dado que têm uma semântica fraca. Tal não é verdade se em vez de palavras forem armazenadas frases, por exemplo. Se em

vez de se armazenar a frase “Voar para a Madeira”, que contém duas *stop-words*, se armazenar “Voar Madeira”, perde-se algum significado. [Dos Santos, 2008]

Estas listas, obviamente dependentes da língua em que o texto está escrito, podem ser encontradas com facilidade na internet, como por exemplo nos seguintes sites: http://www.ranks.nl/stopwords/portugese.html ou então http://members.unine.ch/jacques.savoy/clef/index.html.

- *Stemming*

O processo de *stemming* reduz as palavras ao mesmo *stem,* ou seja, reduz as palavras pela sua raiz semântica, retirando os afixos das palavras (sufixos e prefixos). O algoritmo de Porter [Porter, 1980] é provavelmente o algoritmo de “*stemming*” mais conhecido.
Por exemplo:



Durante o processo de *stemming* costumam ocorrer dois tipos de erros [Chaves, 2003]:

- *Over-stemming*:
  - ocorre quando é removido parte do *stem* e não só um sufixo. Por exemplo: gramática é reduzida a grama em vez de gramát.
  - termos com significados diferentes podem ser reduzidos ao mesmo *stem*, por exemplo: barato e barata serão reduzidos a barat.
- *Under-stemming*:
  - ocorre quando um sufixo não é removido completamente. Por exemplo: referência é reduzida a refêrenc em vez de refer.
  - termos com significados iguais são reduzidos a *stems* diferentes. Por exemplo: viagem e viajar podem ter *stems* diferentes viag e viaj, respectivamente

Existe outro tipo de erro que pode acontecer:

- *Mis-stemming*:
  - O erro consiste em retirar o final da palavra, considerando que é o sufixo da palavra. Por exemplo: a palavra lápis poderia ser reduzida ao *stem* lapi,

dependendo de como o plural das palavras são tratados pelo algoritmo de *stemming*. [Martins, 2003]:

É também uma fase do pré-processamento que é dependente da língua em que o texto está escrito. Diferentes línguas requerem diferentes algoritmos de *stemming*.

- *Indexing*

Definir o índice dos termos, as características que serão utilizadas para a modelação do documento.

Este processo de pré-processamento pode ser enriquecido com recurso a outras técnicas.
O thesaurus ou enciclopédia de termos são um exemplo dessas técnicas que se podem revelar bastantes úteis.
Consistem numa base de dados lexical, ou seja, um dicionário que contém palavras e informação sobre estas. Normalmente a informação armazenada consiste: no significado das palavras, sinónimos, categoria lexical, assim como relações semânticas entre as palavras diferentes ou conjuntos de palavras.
Assim, é possível encontrar ligações entre palavras que dificilmente seriam detectadas por outro meio.
A WordNet é a maior e mais usada enciclopédia de termos para a língua inglesa. Ela agrupa as palavras Inglesas em conjuntos de sinónimos chamados *synsets*, fornece definições curtas e genéricas e regista as várias relações semânticas entre estes conjuntos de sinónimos.
A WordNet faz a distinção entre substantivos, verbos, adjectivos e advérbios porque eles seguem um conjunto de regras gramaticais diferente.
Diferente de outros dicionários, a WordNet não inclui informação sobre etimologia, pronunciação e as formas dos verbos irregulares e contém apenas informação limitada sobre a sua utilização.
A WordNet já foi usado para diferentes propósitos em sistemas de informação, tal como, desambiguação do sentido das palavras, recuperação de informação, classificação automática de texto, sumarização automática de texto e até gerar puzzles de cruzamentos de palavras automaticamente. [REF7]

Esta base de dados encontra-se disponível em: http://wordnet.princeton.edu/.

A aplicação das tarefas de pré-processamento deve ser cuidadosa porque o poder de predição de palavras é altamente dependente do tópico de interesse [Chakrabarti et al, 1998].
Outro aspecto essencial a considerar é a língua em que o documento está escrito, uma vez que determina a lista de *stop-words* e o algoritmo de *stemming* que se vai utilizar.

2.4. Selecção de características

A selecção de características, em inglês, *feature selection*, tenta identificar palavras mais informativas dos documentos para melhorar a eficiência da categorização e reduzir a complexidade computacional. [Yang, 1998]

A selecção de características é desejável não somente para evitar *overffiting* mas também para melhorar a precisão da classificação.
O *overffiting* consiste na hiper-especialização, ou seja, quando se constrói um modelo que se adequa bem ao conjunto de treino mas falha na generalização dos dados. Assim, o modelo fica apenas eficiente para a resolução do problema sobre o conjunto de dados sobre o qual foi treinado ou conjuntos de dados com características semelhantes.
Para manter a precisão é necessário descartar o maior número de características possível, diminuindo assim os tempos de treino e classificação. Deste modo, ficamos com um modelo mais limpo, tornando a interpretação dos resultados mais eficiente.

As técnicas de selecção de atributos, em classificação de texto, são aplicadas normalmente seguindo o processo [Chakrabarti, 2003]:

1. Calcular, para cada característica, a medida que permite descriminar categorias;
2. Listar as características em ordem decrescente por essa medida;
3. Manter um subconjunto de características com um maior poder discriminativo possível.

Métodos de selecção de características

- *Document Frequency Threshold* – usa a frequência do documento inversa (*IDF – inverse document frequency*), ou seja, o número de documentos onde a característica está presente, e elimina as características cuja frequência do documento inversa fica fora de algum limite pré-definido. A aplicação desta técnica é

simples, barata e tem fornecido bons resultados, embora requeira algum cuidado na especificação dos níveis de limiar. [Escudeiro, 2004]

- *Information Gain* (ganho de informação): é um critério que define a qualidade de cada termo. Mede o número de bits de informação obtidos para o prognóstico da categoria sabendo a presença ou ausência de um termo no documento. A categorização de textos possui um espaço dimensional muito grande e é preciso calcular a qualidade do termo de maneira global. A partir de um conjunto de textos de treino, para cada termo único, é calculado o ganho de informação. Os termos que não alcançarem um limiar pré-definido serão excluídos. [Galho, 2003]

- *Mutual Information* (informação mútua): mede a associação entre cada característica e cada categoria, baseando-se numa tabela de contingência nos dois sentidos. As características com a medida de maior informação mútua são seleccionadas. [Escudeiro, 2004]

- X^2 *statistic*: utiliza a mesma tabela de contingência que a informação mútua, mas realiza um teste estatístico do X^2 para inferir sobre a independência entre cada característica e categoria. A principal vantagem deste método, em comparação com a informação mútua, é que neste caso, a estatística do teste é um valor normalizado que permite comparações entre as características para a mesma categoria. [Escudeiro, 2004]

- *Term strength* (força do termo): significativamente diferente do anterior, este método calcula a força do termo, independentemente da categoria do documento.
  Assume-se que os documentos que compartilham muitas palavras comuns são semelhantes, e ainda, que as palavras comuns são informativas. Este método estima a importância de um termo, baseando-se na probabilidade condicional de um termo aparecer num determinado documento dado que aparece noutro documento similar. [Escudeiro, 2004]

- *Terms frequency* (Frequência dos termos): calcula o número de vezes que a característica aparece num documento, só as palavras que ocorrem com mais frequência é que são guardadas. [Gonçalves e Quaresma, 2005]

É importante referir que uma redução agressiva de atributos pode conduzir à eliminação de demasiados atributos significativos, resultando numa representação pobre do documento. Desta representação pobre podem resultar deficiências irrecuperáveis no processo de classificação, o que pode levar a uma classificação errada de documentos. Por esta razão, alguns sistemas de recuperação nem sequer removem as *stop-words*. Como por exemplo, o sistema Thesus [Halkidi et al, 2003; Escudeiro, 2004] que permite que os utilizadores procurem documentos dentro de uma colecção de documentes que fora previamente pesquisada e classificada.

2.5. Representação do documento

Após o pré-processamento dos dados é necessário proceder à sua representação mais adequada, com o intuito de facilitar a sua manipulação nas fases seguintes.

Existem vários modelos de representação dos documentos. Os mais vulgares e úteis para a resolução do nosso problema parecem ser os seguintes:

Modelo Booleano: este modelo considera os documentos como sendo conjuntos de palavras que manipula e descreve através dos operadores booleanos: *and*, *or* e *not*. As expressões booleanas são flexíveis, permitem unir conjuntos, descrever intersecções e retirar partes de um conjunto. [Rodrigues, 2008]
Não existe um resultado parcial; cada documento ou é relevante ou não relevante para uma determinada consulta.
As principais vantagens deste modelo são a sua simplicidade e o formalismo claro. As desvantagens provêm do facto da correspondência exacta poder levar ao retorno de demasiados ou poucos resultados, de pouca precisão, ou de uma abrangência (ver capítulo 2.7) pobre, dos documentos.

Modelo Vectorial (*Vector space model*): cada documento é representado por um vector de termos. Normalmente, os termos são palavras simples, palavras-chave ou frases longas, mas isto depende da aplicação. Por exemplo, caso se usem as palavras como termos, a dimensão do vector é o número de palavras distintas que ocorrem no conjunto de textos onde se inserem o documento em análise.
A cada termo é atribuído um peso, ou seja, é associado um valor que indica o grau de importância do termo no documento. [Rodrigues, 2006]

Este peso pode ser binário, isto é, indica se existe ou não um determinado termo no documento; inteiro, indicando o número de vezes que o termo aparece no documento, ou real. Um cálculo muito frequente para a atribuição dos pesos é o TFxIDF. No capítulo 3.1 é descrito mais em pormenor os cálculos do peso de cada termo.

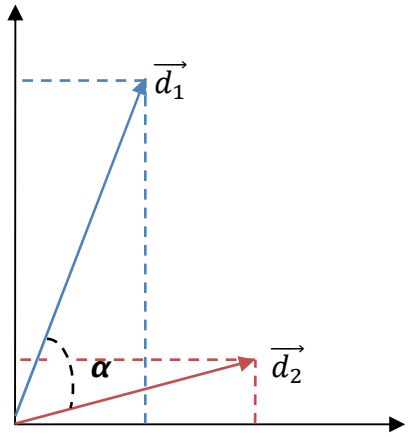


Figura 4 - Modelo Vectorial

A Figura 4 ilustra o modelo vectorial, onde:

$\overrightarrow{d_j} = (w_{1,j}, w_{i,2}, \ldots, w_{i,j})$

d_i corresponde ao documento i

$w_{i,j}$ corresponde ao peso de cada termo j para o documento i.

As coordenadas do documento são dadas pelo seu peso, que pode ser obtido de várias formas. A similaridade entre documentos é medida com base nos vectores que os representam.

A medida mais usada para determinar a similaridade entre dois vectores é a do co-seno do ângulo entre os vectores:

(1)

$$Sim(D_1, D_2) = \frac{\sum_{K=1}^{n}(w_{d1,k} \times w_{d2,k})}{\sqrt{\sum_{K=1}^{n}(w_{d1,k})^2} \times \sum_{K=1}^{n}(w_{d2,k})^2}$$

Onde:

D_1 é o vector de termos do documento 1;

D_2 é o vector de termos do documento 2;

$w_{d1,k}$ são os pesos dos termos do documento 1;

$w_{d2,k}$ são os pesos dos termos do documento 2.

Uma das desvantagens deste modelo reside no facto de se assumir que os termos são independentes, perdendo-se assim o contexto em que os termos aparecem.

Algumas propostas, distintas dos modelos vectoriais tradicionais, tentam explorar sequências de caracteres - o texto é representado como um conjunto de sequências de caracteres - usando funções de Kernel para medir a similaridade entre documentos; o modelo *string kernel*. [Escudeiro, 2004]
O texto pode ainda ser representado como sequência de palavras, que é linguisticamente mais representativo do que caracteres, adaptando a função *string kernel* a funções *word kernel*, reduzindo significantemente o problema de dimensão. [Nicola et al, 2003]

Modelo Probabilístico: este modelo, como o nome indica, trabalha com conceitos provenientes da área de probabilidade e estatística.
Os termos indexados dos documentos não possuem pesos pré-definidos, sendo a ordenação dos documentos calculada pesando dinamicamente os termos dos documentos entre si.
O objectivo deste modelo é saber a probabilidade de um documento ser ou não relevante comparando com outro documento.

Este modelo avalia os documentos em ordem decrescente da sua probabilidade de serem relevantes, o que é uma vantagem. As suas principais desvantagens são: a necessidade de adivinhar a separação inicial dos documentos em conjuntos relevantes e não-relevantes; tal como no modelo Booleano, os pesos são binários; tal como no modelo vectorial, os termos indexados são assumidos como independentes. [Escudeiro, 2004]

Os modelos abordados são na maioria dos casos aplicados a palavras. Neste trabalho são analisadas palavras isoladas, como também conjuntos de palavras adjacentes, através dos *N-grams* e ainda grupos gramaticais, aplicando o *pos-tagging*.

No capítulo 3 são analisados os vários modelos e formas de representação de documentos, referidos anteriormente, a fim de encontrar o melhor método para a resolução do problema proposto neste trabalho.

2.6. Classificadores

Um classificador é um algoritmo que, dada uma ou mais classes (ou etiquetas) de entrada, automaticamente decide a que classe, ou classes, pertence um dado documento, com base numa análise do seu conteúdo. [Sebastiani, 2005]

Vários classificadores/métodos já foram criados para a resolução de problemas de categorização de texto. Alguns dos classificadores mais comuns para a área de *text mining* são:

- Naïve Bayes

O método Naïve Bayes é um modelo matemático simples que se baseia no teorema de Bayes.
Usa a probabilidade conjunta das palavras/termos e categorias, para estimar probabilidades de uma determinada categoria de documentos (BOW). [Escudeiro, 2004]
Como o seu nome indica é um classificador ingénuo (naïve), isto porque assume que os atributos são independentes.
Apesar de ser um classificador "ingénuo", simples e rápido, possui uma precisão alta e reporta um óptimo desempenho em várias tarefas de classificação.
A probabilidade de um documento *d* pertencer a uma classe *c* é calculada através da seguinte fórmula:

(2)

$$P(c|d)\ \alpha\ P(c) \prod_{1 \le k \le n_d} P(t_k|c)$$

Na classificação de texto, o objectivo é encontrar a melhor classe para o documento. A melhor classe na classificação Naïve Bayes é a classe mais provável ou a classe do máximo a posteriori (*maximum a posteriori – MAP*):

(3)

$$c_{map} = \arg max_{c \in C}\ \hat{P}(c|d) = \arg max_{c \in C}\ \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}\,(t_k|c)$$

Coloca-se $\hat{P}$ para P porque desconhece-se os verdadeiros valores dos parâmetros $P(c)$ e $P(t_k|c)$. Mas estes são estimados a partir do conjunto de treino. [Cambridge University Press, 2009]

- Árvores de decisão (*Decision tree*)

Uma árvore de decisão é um diagrama de fluxo que apresenta uma estrutura em árvore, como o nome indica. Os nós internos são para testes e as folhas representam as categorias. Um nó interno inclui um teste a um atributo enquanto que o ramo representa o resultado do teste.
Uma vez que o atributo que é utilizado para fazer a decisão não está definido é possível usar o atributo que dá o máximo de informação.

Potencialmente existem várias árvores de decisão que podem ser inferidas a partir de um mesmo conjunto de dados. Mas, diferentes árvores apresentam diferentes taxas de exactidão.
Existem diversos algoritmos de construção de árvores de decisão, baseados em optimizações locais que escolhem os atributos que melhor particionam o conjunto de dados. [Rodrigues, 2008]

Algumas das vantagens deste modelo centram-se no facto de este não assumir nenhuma distruibuição particular para os dados, ou seja, pode construir modelos para qualquer função desde que o número de exemplos de treino seja suficiente. Também possui uma estrutura da árvore de decisão independente da escala das variáveis, um elevado grau de interepretabilidade e é eficiente na construção de modelos.

No entanto as árvores de decisão, como qualquer método, possuem alguns inconvenientes: mesmo sendo fácil de aplicar é muito dificil de construir e não é possível visualizá-lo numa só árvore, possui replicação de subárvores. Para além disso é um modelo instável uma vez que pequenas pertubarções do conjunto de treino podem provocar grandes alterações no modelo aprendido.

- Redes neuronais (*Neural Networks - NN*)

Uma rede neuronal é uma máquina de processamento altamente paralelo e distribuído composto por unidades de cálculo simples que tem a capacidade inata de aprender, guardar e utilizar conhecimento. [Rodrigues, 2008]

As redes neuronais assemelham-se ao cérebro humano em dois pontos principais: [Rodrigues, 2008]

- O conhecimento é adquirido pela rede a partir do ambiente envolvente através de um processo de aprendizagem;
- Os nós da rede (neurónios) estão ligados através de interligações com pesos (pesos sinápticos), usados para guardar o conhecimento.

As redes neuronais, para a recuperação de informação, são compostas por três camadas: uma para os termos de consultas, uma para os termos índice e outra para os próprios documentos.

O objectivo das redes neuronais é obter um vector de pesos que classifica correctamente quase todos os exemplos de treino.

As redes neuronais conseguem uma precisão da classificação geralmente elevada, mesmo para problemas complexos. Outra das suas vantagens é o processamento distribuído, ou seja, o conhecimento é distribuído através dos pesos das ligações. É um modelo robusto no tratamento de exemplo mesmo que contenham erros e permite uma avaliação rápida da aprendizagem da função alvo.
Além disso, as redes neuronais são difíceis de usar, uma vez que possuem muitos parâmetros a definir e também é difícil determinar a topologia óptima da rede para um problema. Por fim, além de requerem pré-processamento específico de dados, o conhecimento não é legível e não são fornecidas explicações dos resultados.

- Rocchio

O algoritmo de Rocchio é um método clássico do modelo-espaço-vectorial para categorização de documentos na recuperação de informação. A ideia base deste modelo, aplicado à classificação de texto, é usar um conjunto de treino de documentos para construir um vector protótipo por categoria.
É um método simples de implementar e eficiente em computação, tendo já sido utilizado como base para várias avaliações [Lewis et al., 1996; Cohen e Singer, 1996].
A sua principal falha é a suposição de apenas um centróide por categoria, o que limita a obtenção de bons resultados quando os documentos de uma dada categoria formam clusters diferentes.

Vários autores [Joachims, 1998; Yang e Liu, 1999] referem que o SVM e o KNN são os classificadores mais indicados para problemas de *text mining*. Estes classificadores são introduzidos mais em detalhe nos subcapítulos seguintes.

2.6.1. K- Nearest Neighbour

O método KNN, também conhecido por método do vizinho mais próximo, representa documentos no espaço como um vector, em função dos seus pesos.
Para classificar um novo documento, o sistema localiza os k vizinhos mais próximos entre os documentos de treino, e usa as categorias dos k vizinhos com melhor ranking para prever a categoria do novo documento.
Durante a fase de treino é feita uma simples memorização dos exemplos, é a chamada aprendizagem preguiçosa – "*lazy learning method*". [Rodrigues, 2008]

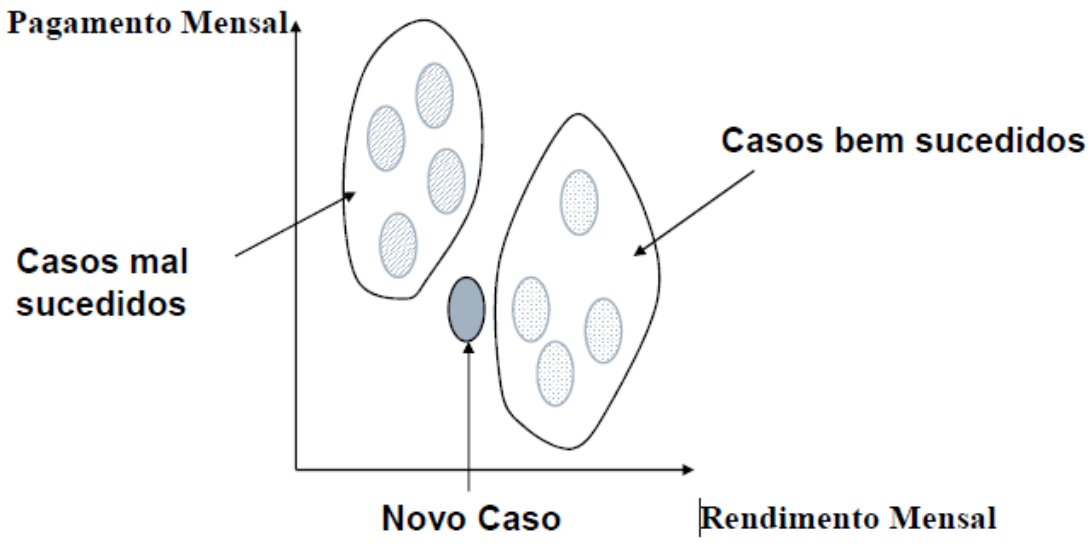


Figura 5 - Método K vizinhos mais próximos

O resultado da similaridade de cada documento vizinho para um novo documento que está a ser classificado é usado como peso de cada uma das suas categorias, e a soma dos pesos das categorias sobre os k vizinhos mais próximos são usados para o ranking da categoria. [Yang, 2009]

Os principais factores que influenciam o desempenho do algoritmo KNN são: o valor de k (número de vizinhos a considerar), a função de semelhança usada para encontrar os

vizinhos mais próximos e a regra de decisão para identificar a classe para o documento teste.

Um dos inconvenientes deste algoritmo é a sua eficácia, uma vez que ele precisa de comparar o documento teste com todas as amostras do conjunto de treino para poder classificá-lo. Além disso, o bom desempenho deste algoritmo depende do valor atribuído a k, que deve ser avaliado caso a caso.

A melhor escolha do valor de k depende dos dados. Em geral, valores maiores de k reduzem o efeito do ruído sobre a classificação, mas criam limites entre as classes menos distintas. [Miah, 2009]

É um modelo bastante usado em problemas de classificação de texto, uma vez que é bastante eficaz quando possui um conjunto de treino grande. Além disso, é fácil de implementar, usa qualquer função de distância e os resultados finais são simples de entender, entre outras vantagens.
No entanto, o cálculo das distâncias entre casos é computacionalmente intensivo, uma vez que analisa o documento original com todos os dados de treino, quando apenas alguns é que podem ser importantes. E, como já foi referido anteriormente, é necessário um grande conjunto de treino.
Outra possível desvantagem pode derivar da função de distância usada e do número de vizinhos, uma vez que os resultados dependem destas escolhas.

Como referido anteriormente, são usadas medidas de semelhança entre vizinhos para decidir a categoria a que pertence o novo documento. As medidas de similaridade mais usadas na classificação de texto são: a distância euclidiana e a similaridade cosseno.

Distância Euclidiana

A distância euclidiana é o cálculo mais usado para obter a distância entre dois pontos.

(4)

$$d\,(p,q) = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}$$

Calcula a distância entre dois documentos, em que, *n* é o número de atributos e p_k e q_k são, respectivamente, os k^{th} atributos *p* e *q*.

Similaridade Cosseno

A similaridade é medida através do cosseno entre os vectores representativos dos documentos a comparar.

(5)

$$sim(X, D_j) = \frac{\sum_{t_i \in (X \cap D_j)} x_i \times d_{ij}}{\|X\|_2 \times \|D_j\|_2}$$

Onde:

- X é o documento de teste, representado como um vector;
- D_j é o j^{th} documento de treino;
- t_i é a palavra comum entre X e D_j;
- x_i é o peso da palavra t_i em X;
- d_{ij} é o peso da palavra t_i no documento D_j;
- $||X||^2$ é a norma de X;
- $||D_j||^2$ é a norma de D_j.

2.6.2. Support Vector Machine

A técnica de SVM, baseada na teoria estatística de aprendizagem, foi desenvolvida por Vladimir Vapnik em 1979, com o intuito de resolver problemas de classificação de padrões.
Desde a década de 90 é muito usada em diversas áreas tais como: reconhecimento de assinaturas, classificação de texto e imagens, identificação de *spams*, reconhecimento de padrões diversos (por exemplo: detecção de face em imagens), identificação de dados replicados, etc.
Tem grande sucesso na resolução de problemas de classificação de texto, área na qual estamos a trabalhar.

Os métodos de SVM usam o espaço vectorial para representar os documentos.

SVMs Lineares

Um classificador SVM linear encontra um separador (**hiperplano**) entre duas classes com uma margem máxima, que separa os exemplos positivos e negativos para cada classe.
A **margem** é a distância entre o hiperplano e o ponto mais próximo de cada uma das classes.
Os pontos mais próximos do hiperplano de separação são definidos como **vectores de suporte** (*support vectors*).
A separação óptima é definida pela maior distância entre cada classe, devendo-se então maximizar a margem, uma vez que os pontos mais próximos são aqueles onde existe maior incerteza sobre a sua classificação.

No caso de duas classes linearmente separáveis, um classificador SVM encontra o hiperplano que está mais afastado dos pontos mais próximos, não tendo em conta os restantes.

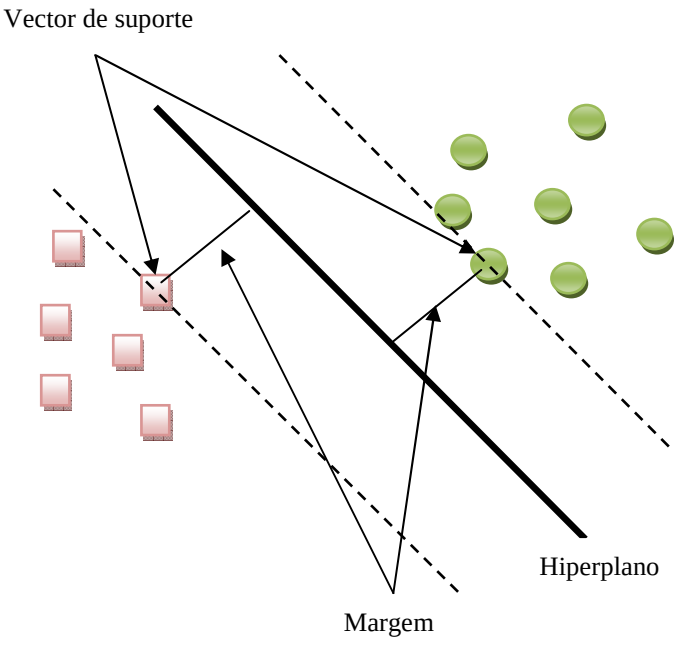


Figura 6 – SVM linear

SVMs não lineares

As SVMs são eficazes na classificação de conjuntos de dados linearmente separáveis ou que possuam uma distribuição aproximadamente linear. Porém, há muitos casos em que não é possível dividir satisfatoriamente os dados de treino por um hiperplano. [Lorena e De Carvalho, 2008]
Nesses casos o uso da fronteira curva é o mais adequado para a separação das classes, são as chamadas SVMs não lineares.

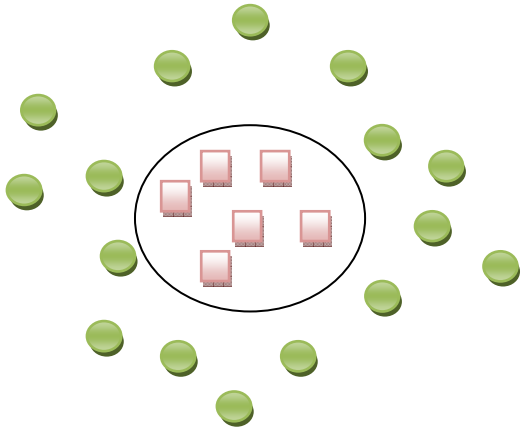


Figura 7 – SVM não linear

Quando as classes não são linearmente separáveis, torna-se possível utilizar diferentes funções matemáticas, designadas *kernels*, que têm como objectivo mapeá-las num espaço em que já o sejam, sendo assim possível definir hiperplanos de decisão para o novo espaço. [Santos, 2008]

Classificação de texto com SVM

Segundo a análise teórica de Thorsten Joachims, as SVMs reconhecem as propriedades particulares do texto e atingem um bom desempenho em tarefas de classificação de texto. Obtêm substancial e significantemente melhores resultados que outros métodos existentes, nomeadamente devido à sua robustez em trabalhar perante um conjunto elevado de termos. Uma vez que é hábil em generalizar bem espaços multidimensionais de grande dimensão, as SVMs eliminam a necessidade de selecção de características, fazendo a aplicação de classificação de texto consideravelmente mais simples.

Como principais fraquezas, apontam-se a sua sensibilidade a escolhas de valores de parâmetros e a dificuldade de interpretação do modelo gerado. [Dos Santos, 2008; Lorena e De Carvalho, 2008]

2.7. Métodos de avaliação

As medidas mais comuns para a avaliação dos resultados da classificação de texto são: abrangência (*recall*), precisão (*precision*), fallout, exactidão (*acurracy*) e error. Sendo as medidas de precisão e abrangência as mais usuais e indicadas para a classificação de texto.

Precisão

Como o nome indica, mede a precisão das categorias correctamente encontradas. Ou seja, a habilidade do sistema manter os documentos irrelevantes fora do resultado de uma consulta. A sua taxa máxima é 1.

(6)

$$\text{Precision} = \frac{\text{Categorias encontradas e correctas}}{\text{Total de categorias encontradas}}$$

Abrangência

A abrangência permite obter a relação entre o número de documentos correctamente classificados numa dada categoria sobre o total de documentos dessa categoria. Ou seja, mede a habilidade do sistema encontrar os documentos mais relevantes. A sua taxa máxima é 1.

(7)

$$\text{Recall} = \frac{\text{Categorias encontradas e correctas}}{\text{Total de categorias correctas}}$$

Constata-se que quando a precisão aumenta a abrangência baixa e vice-versa.

O modelo ideal é aquele que maximiza ambas as medidas: precisão e abrangência. Estas medidas são relacionadas através da medida *f-measure* (ver descrição mais abaixo neste capítulo) – um valor elevado do *f-measure* garante valores elevados de precisão e abrangência. [Rodrigues, 2008]

Fallout

Mede a quantidade de documentos recuperados irrelevantes. A melhor taxa é 0, ou seja, significa que nenhum dos documentos encontrados é irrelevante. Mas, este resultado terá de ser bem analisado porque pode significar que não foi recuperado nenhum documento, logo, não tem nenhum em comum com os documentos irrelevantes.

(8)

$$\text{Fallout} = \frac{\text{Categorias encontradas e incorrectas}}{\text{Total de categorias incorrectas}}$$

Exactidão

Permite obter a relação dos exemplos correctamente classificados sobre o número total de exemplos avaliados.

Error

Define o valor dos exemplos incorrectamente classificados sobre o número total de exemplos avaliados.

F-measure

A *f-measure*, também designada por *F-Score* ou F_1 *score*, representa a média harmónica de precisão e abrangência.

(9)

$$F - measure = 2 * \frac{\text{precisão} * \text{abrangência}}{\text{precisão} + \text{abrangência}}$$

3. Modelos de representação

Os algoritmos de aprendizagem não interpretam directamente os documentos digitais, sendo necessário transformar cada texto, que seja processável por meios computacionais, numa representação compacta do seu conteúdo. [Gonçalves, 2007]

3.1. Bag-of-words (saco de palavras)

O BOW é o modelo mais utilizado em aplicações de classificação de texto.
Este modelo transforma a cadeia de caracteres de um documento num conjunto de palavras, registando além da presença de uma palavra, a sua frequência. Ignora toda a estrutura do documento. Propriedades básicas do texto, como a ordem em que as palavras ocorrem e a pontuação, são ignoradas.
Outra desvantagem deste modelo é a sua incapacidade em capturar a semântica do texto, isto é, há palavras com significados distintos que apesar de serem exactamente iguais têm significados diferentes, dependendo do contexto em que são utilizadas. Com este modelo, essas palavras serão consideradas como iguais.

Contudo, este modelo tem como vantagens ter um custo em termos de processamento muito menor do que as aproximações alternativas, especialmente no caso de colecções grandes e dinâmicas de texto. [Santos, 2006]

No modelo do BOW qualquer texto – frase, documento, colecção de documentos – é visto como um grupo de elementos. Dependendo da aplicação, estes elementos podem ser: palavras simples, n-gramas, estruturas compostas, etc. [Sarmento, 2006]

Na Tabela 1, w_i representa uma palavra, d_j representa um documento e a_{ij} o peso de cada palavra no documento.

Tabela 1 - Modelo de BOW

	w_1	w_2	w_3	⋯	w_n
d_1	a_{11}	a_{12}	a_{12}	⋯	a_{1n}
d_2	a_{21}	a_{22}	a_{23}	⋯	a_{2n}
⋯	⋮	⋯	⋯	⋱	⋮
d_n	a_{n1}	a_{n2}	a_{n3}	⋯	a_{nn}

Existem várias medidas para calcular os valores dos pesos de a_{ij}. Essas medidas podem ser classificadas em dois tipos: binárias e baseadas em frequências.

Os pesos binários indicam a ocorrência ou não de um dado termo num determinado documento (1 – verdadeiro e 0 – falso), podendo ser utilizados, por exemplo, para extrair informações relativas à semelhança de documentos a partir do número de termos em comum.

Os pesos baseados em frequência visam contabilizar o número de ocorrências de um dado termo num determinado documento, servindo como base para diversas medidas estatísticas [Magalhães, 2009]. A sua medida é a frequência do termo – TF(w_i, d_j) definida como o número de vezes que uma palavra w_i aparece num documento d_j.

Estes dois pesos (binário, frequência) não têm em conta a frequência do termo sobre todos os documentos no corpus não reflectindo o carácter discriminativo do termo dado o corpus. Tendo em conta este ponto, o método mais conhecido para calcular o peso dos termos é o TFxIDF.

Em primeiro lugar, o TFxIDF tem em conta a frequência do termo (TF), isto porque quanto maior for a ocorrência de um termo num documento maior é o seu contributo para a importância do documento. De seguida, o IDF mede o quão infrequente é uma palavra em todo o conjunto de documentos.

De acordo com este método, se uma palavra é muito frequente em todo o conjunto de documentos, então não é considerada como sendo particularmente representativa do documento em si (pois ocorre na maior parte dos documentos, tal como acontece, por exemplo, com as *stop-words*). Em contraste, se uma palavra aparece poucas vezes em toda a colecção de textos conclui-se que tem uma grande importância para definir o documento em causa. [Santos, 2006]

Existem várias fórmulas para a representação do cálculo do TFxIDF. A seguinte fórmula é uma delas:

(10)

$$W_{i,j} = tf_{i,j} * \log\left(\frac{N}{df_{i,j}}\right)$$

Onde:

$Tf_{i,j}$ = frequência do termo i no documento j;

$Df_{i,j}$ = número de documentos que contêm i;

N = número total de documentos.

É comum utilizar-se algoritmos como o "*Pos-tagger*" e dicionários léxicos para fornecer informação adicional sobre os termos, a fim de melhor identificar quais os termos relevantes e quais os termos irrelevantes para a classificação do texto.

3.2. N-Grams

Um *N-Gram* é uma sequência de n palavras consecutivas: *unigram* para n=1, *bigram* para n=2, *trigram* para n=2, etc. Por exemplo: "*Text Classification*" tem 2 palavras é o chamado *bigram*.

O modelo de representação do *N-Gram* possibilita manter a ordem das palavras permitindo assim conservar o contexto, ao contrário do modelo do BOW.
Por exemplo: no modelo do BOW, um texto contendo a expressão "grande homem" e outro contento "homem grande" geram exactamente a mesma distribuição. Porém, adoptando o modelo 2-*grams* (*bigram)*, evita-se que isto ocorra.

Em setembro de 2006 a Google tornou público o seu repositório de *n-grams*:
http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13

3.3. Latent Semantic Indexing (Indexação semântica latente)

A indexação semântica latente, do inglês Latent Semantic Indexing (LSI), é um método de indexação e recuperação que usa a técnica matemática *Singular Value Decomposition* (SVD) para encontrar padrões nas relações entre os termos e os conceitos contidos numa colecção de texto não estruturada. O LSI é baseado no princípio de que as palavras que são usadas no mesmo contexto tendem a ter significados semelhantes. A característica essencial do LSI é a sua capacidade de extrair o conteúdo conceptual do corpo do texto através da criação de associações entre os termos que ocorrem em contextos similares. [REF14]
Este método também é designado de *Latent Semantic Analysis,* ou seja, análise semântica latente.

De seguida é apresentada a evolução do LSI ao longo dos anos: [REF14]

Mid-1960s *– Factor analysis technique first described and tested (H. Borko and M. Bernick)*

1988 *– Seminal paper on LSI technique published (Deerwester et al.)*

1989 *– Original patent granted (Deerwester et al.)*

1992 *– First use of LSI to assign articles to reviewers (Dumais and Nielsen)*

1994 *– Patent granted for the cross-lingual application of LSI (Landauer et al.)*

1995 *– First use of LSI for grading essays (Foltz, et al., Landauer et al.)*

1999 *– First implementation of LSI technology for intelligence community for analyzing unstructured text (SAIC).*

2002 *– LSI-based product offering to intelligence-based government agencies (SAIC)*

2005 *– First vertical-specific application – publishing – EDB (EBSCO, Content Analyst Company)*

Segundo Zukas e Price, que realizaram um estudo sobre categorização de documentos usando LSI, o LSI é uma técnica automática para processamento de material textual. Produz capacidades de vanguarda tais como: classificação automática de documentos, recuperação de informação conceptual e recuperação de informação *cross-lingual*.

Estes dois autores concluem que a tecnologia LSI amadureceu ao ponto de ser uma abordagem particularmente atractiva para a categorização de texto. Os resultados da categorização de texto com LSI são competitivos, em conjuntos de testes comparáveis, com os melhores resultados reportados na literatura. A grande vantagem da categorização de texto usando LSI é o suporte nativo para as linguagens internacionais.

3.4. Part-of-speech tagging (POS Tagging)

Uma mesma palavra pode ter significados diferentes dependendo do contexto em que é usada. Por exemplo, a palavra *love* na frase "*I love this movie*" tem um significado diferente do que quando é utilizada na frase "*It's a love story*". Ao extrair a categoria gramatical das palavras (adjectivo e substantivo respectivamente) este conflito é reduzido. [Santos, 2006]

O POS tagging é o processo de atribuir a cada palavra a sua categoria lexical correspondente, como nome, verbo, preposição, etc, baseado tanto na definição da palavra

como no contexto onde ela se encontra. Por exemplo, a frase “I love this movie” é transformada em “I/PRP love/VBR this/PP movie/NN” onde PRP = pronome, VBR = verbo, PP = preposição e NN = substantivo.
Os classificadores passaram a identificar o par “palavra/POS” como uma palavra.
É importante referir que as *tags* atribuídas dependem do programa utilizado. Para o exemplo acima, foi utilizado o *Stanford Tagger*, disponível na página: http://nlp.stanford.edu/software/tagger.shtml.

O *POS Tagging* é um passo importante na análise da linguagem natural, é robusto, rápido e trabalha com 95-97% de precisão. [Volk, 2005]

É importante referir que este processo apresenta um problema que consiste no facto de, por vezes, algumas palavras poderem pertencer a mais do que uma categoria lexical.

A norma CLAWS (*the Constituent Likelihood Automatic Word-tagging System*) é um sistema de *part-of-speech* utilizado para a língua inglesa, que continua a ser desenvolvido desde o inicio dos anos 80 e consegue atingir 96-97% de precisão.

É possível encontrar uma lista de vários *POS taggers* disponíveis pela Stanford em: http://www-nlp.stanford.edu/links/statnlp.html#Taggers.

3.5. String Kernels

As *string kernels* permitem comparar documentos de texto através de *substrings* contidas nos documentos e não exclusivamente por palavras. Estas *substrings* não precisam de ser próximas, mas elas recebem um peso diferente conforme o grau de proximidade [Fortuna, 2004]. Por exemplo: a *substring* “b-a-r” está presente tanto na palavra “barco” como na palavra “bilhar” mas com diferentes pesos. O peso depende do comprimento da *substring* e do factor de declínio λ. No exemplo anterior, a *substring* “bar” receberia um peso de λ^5 como parte de “barco” e um peso de λ^6 como parte de “bilhar”.

As vantagens desta abordagem, comparando com o BOW, são que ela pode detectar palavras com diferentes sufixos ou prefixos: as palavras “microcomputador”, “computadores” e “computadorizado” partilham a mesma *substring*. [Fortuna, 2004]

Os *Kernels* são sistemas de aprendizagem padrão (como redes neuronais ou árvores de decisão). Estes sistemas operam sobre dados de entrada depois de serem transformados em vectores de características. Em alguns casos, os dados não são facilmente descritos por vectores de características explícitos, tais como, gráficos ou imagens, por exemplo.

Os métodos de Kernel (KM – *Kernel Methods*) são uma alternativa eficaz para a extracção de características explícitas. [Lodhi et all, 2002]
A função de Kernel retorna o produto interno entre documentos, mapeados num grande espaço de características dimensionais. Em muitos casos, o produto interno pode ser calculado sem calcular explicitamente os vectores de características.
Vários algoritmos de aprendizagem podem ser reescritos para que os vectores de características só apareçam no interior do produto interno, então não há necessidade de os calcular explicitamente.
O exemplo mais conhecido de um sistema baseado em kernel é o SVM, mas também existem outros: Kernel PCA, Kernel KCCA, KNN, etc. [Fortuna, 2004]

4. Abordagem ao problema

Após o estudo do processo de classificação de texto em si, pretende-se neste capítulo descrever a metodologia usada para a resolução do problema proposto, incluindo todos os passos desenvolvidos, bem como a caracterização e resolução dos problemas encontrados.

4.1. Descrição de alto nível da metodologia

O problema apresentado é um problema difícil de abordar, isto porque a máquina não entende o significado das palavras ou frases recebidas. A ideia desenvolvida neste trabalho permitiu analisar várias alternativas possíveis para a representação dos textos, de forma a potenciar uma melhor performance dos processos de classificação automática de textos, no que respeita à precisão e abrangência. Paralelamente pretendia-se perceber qual a influência dos vários elementos que compõem um texto (palavra, frases, pontuação, etc). Também foram ensaiados vários algoritmos de classificação.

Foram criados diversos modelos para representação de textos, variando as tarefas executadas ao nível do pré-processamento e o tipo de atributos considerados relevantes. Estes modelos foram sendo adaptados e melhorados ao longo dos desenvolvimentos práticos.

Os modelos de representação implementados foram baseados na representação dos textos com diferentes tipos de atributos: palavras (BOW), *ngrams* (para n=2,3,4,5) e *pos-tag* (para *tag* = Verbos, nomes e adjectivos). Para cada modelo procedeu-se ao respectivo pré-processamento dos dados, combinando as diversas tarefas para avaliar o impacto dos passos mais relevantes (remoção de pontuação, remoção de *stop-words* e aplicação do *stemming*).

Após obter todas as instâncias de representação do corpus, resultantes das várias combinações possíveis, com base nestes dois aspectos (atributos e tarefas de pré-processamento), procede-se à sua classificação.

Como resultado do estudo prévio do processo de classificação de texto, constatou-se que os classificadores mais eficazes e mais usados são o SVM e o KNN. Por isso, estes foram os classificadores usados para proceder à avaliação dos vários modelos de representação.

Por fim, os resultados obtidos são avaliados à luz dos critérios de avaliação típicos em classificação de texto. Os critérios mais importantes são a precisão e a abrangência.

Após a definição de alto nível da abordagem a adoptar, foi necessário definir as características necessárias para o corpus a ser trabalhado. Foi essencial obter um conjunto de textos atraente e sobretudo pertinente para o problema proposto.
Decidiu-se que o corpus a usar teria de conter opiniões distintas sobre um mesmo tema ou tópico. Após várias pesquisas de mercado e análises optou-se por um corpus constituído por críticas cinematográficas. As críticas cinematográficas exprimem opiniões positivas e negativas sobre um mesmo filme, exibindo portanto, as características que se pretende estudar. O corpus efectivamente utilizado contém críticas referentes a vários filmes distintos, aumentado ainda mais o grau de dificuldade do problema de classificação.

Uma vez definida a abordagem seguida na parte prática deste trabalho, é importante descrever todos os passos implementados assim como as dificuldades e problemas que tiveram de ser ultrapassados.

4.2. Interpretação dos dados no R

O R separa as palavras de uma frase pelo carácter espaço. Como tal a frase “Óptimo filme, aconselho a quem gostaria de, por vezes, alterar o seu passado!”, quando dividida em palavras fica (cada palavra será separada por []): [Óptimo] [filme,] [aconselho] [a] [quem] [gostaria] [de,] [por] [vezes,] [alterar] [o] [seu] [passado!].
De notar que quando existe uma pontuação, essa pontuação fica junta com a palavra que a precede, isto se não existir um espaço entre a palavra e a pontuação. Para o R a palavra “filme,” e “filme” são consideradas diferentes, daí a importância da pontuação e da necessidade de analisar a sua presença ou a falta dela.

Por outro lado a pontuação é usada também para exprimir sentimentos (ponto de exclamação, por exemplo) e a sua análise pode ser relevante para a tarefa de classificação que está a ser tratada neste trabalho.

4.3. Representação de documentos

Nesta secção são apresentados todos os passos e análises desenvolvidos para a criação e instanciação dos modelos de representação do corpus.
Esta é a fase mais importante e morosa de todo o processamento, uma vez que é aqui que os dados vão sofrer alterações fulcrais e irreversíveis para as fases seguintes.

4.3.1. Definição dos modelos de representação

As Figuras 8 e 9 ilustram os modelos de representação implementados, apresentando todas as combinações efectuadas com as diversas tarefas de pré-processamento para avaliar o impacto dos passos mais relevantes (remoção de pontuação, remoção de *stop-words* e aplicação do *stemming*) e com os diferentes tipos de atributos (palavras (BOW), *ngrams* e *pos-tag*).

No diagrama da Figura 8 apresentam-se os passos aplicados para os modelos do BOW e dos *NGrams*.

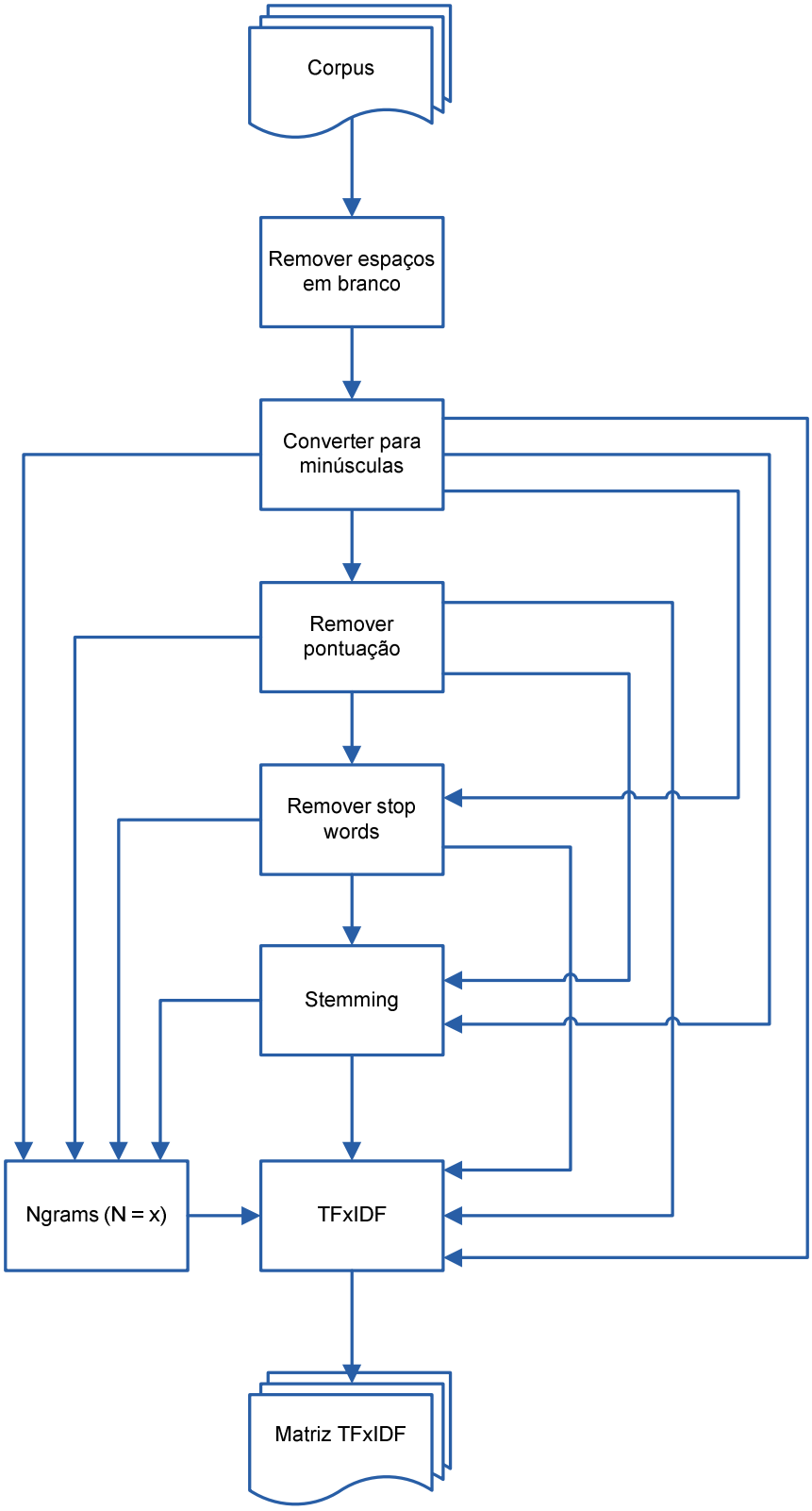


Figura 8 – Modelo de representação BOW e *NGrams*

Neste caso, poderão ser criados vários modelos dependendo de se executar ou não cada uma das funções:

- remoção de pontuação;
- remoção de *stop-words*;
- *stemming*.

  Podemos, por exemplo, obter uma representação do corpus com pontuação, com *stop-words*, com *stemming* ou então sem pontuação, com *stop-words*, com *stemming*, etc.

- Por fim, pode-se implementar *NGrams* considerando vários valores para n. Optou-se por considerar n=2,3,4 e 5.

É possível combinar as várias opções para posteriormente avaliar o impacto de cada passo, e decidir qual a forma de representação dos textos garante o melhor desempenho, na correcta identificação do teor das opiniões expressas no texto.

Posteriormente é apresentado o diagrama que introduz o conceito de *Pos-Tag*. Com o *Pos-Tag* é possível relacionar as palavras em função da sua categoria gramatical, o que pode ser bastante significativo.

Neste diagrama, para além das funções opcionais referenciadas anteriormente, introduz-se a possibilidade de realizar *Pos-Tagging*.
O *Pos-Tagging* deve ser aplicado antes de ser efectuada qualquer alteração ao texto. Toda a informação é importante, desde a pontuação, às *stop-words,* etc, para identificar a categoria gramatical de uma palavra.

Para a implementação do modelo *POS-Tag* foi usado o método tagPOS da libraria OpenNPL disponível no R. As *tags* usadas por este método podem ser consultadas em: http://bulba.sdsu.edu/jeanette/thesis/PennTags.html

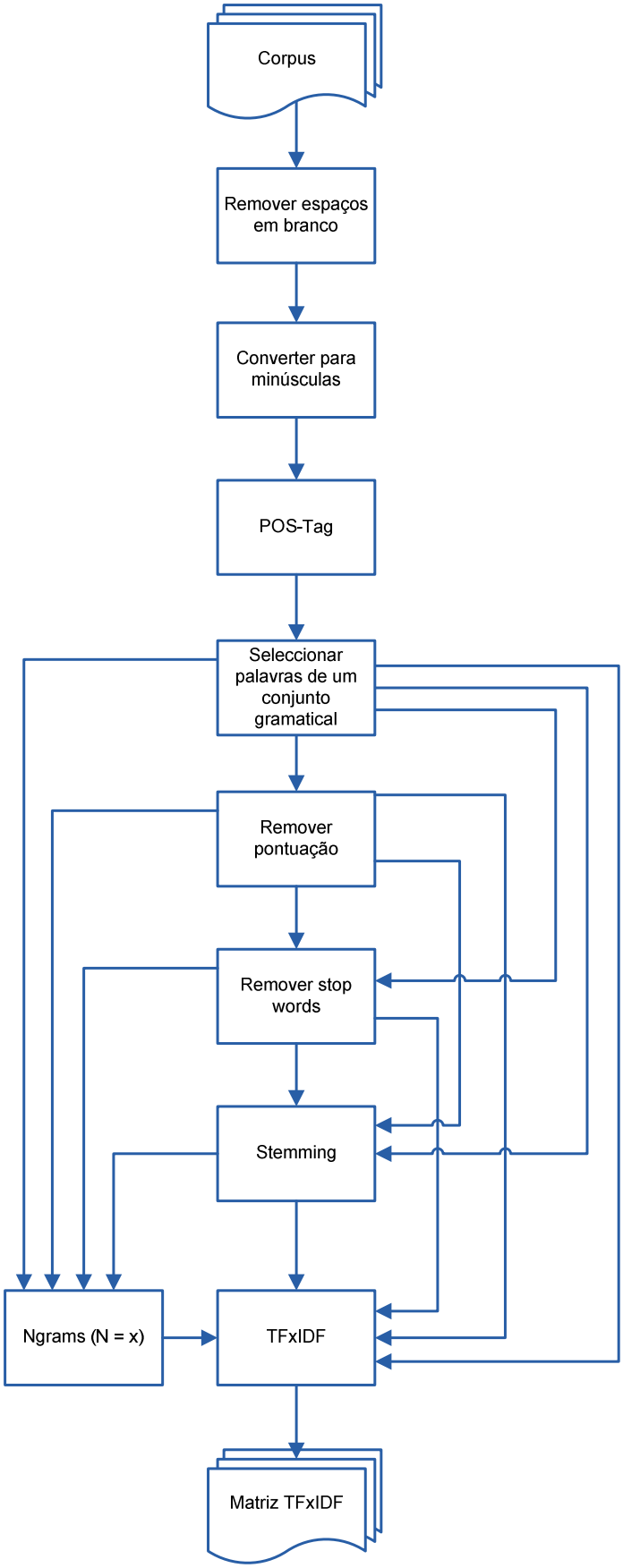


Figura 9 – Modelo de representação POS-Tag

4.3.2. Conjunto de modelos

As Figuras 8 e 9, na secção anterior, ilustram os diversos modelos de representação de texto criados. Foram criadas várias alternativas, para os modelos BOW, *NGrams* e *POS-Tag* combinado as diversas tarefas de pré-processamento.
Ao todo são construídos conjuntos de oito combinações para cada tipo de modelo.

No caso do *NGrams* foram escolhidos alguns valores distintos para n, a fim de poder avaliar qual o efeito que este parâmetro pode ter. Foram criados modelos para n=2,3,4,5 sendo estes os valores que pareceram mais adequados. Como tal, foram criados 8 combinações * 4 valores de n logo trinta e dois modelos para *NGrams*.

Por fim, para o *POS-Tag* as categorias gramaticais seleccionadas foram: verbos, nomes, adjectivos e algumas combinações entre elas: verbos e nomes, nomes e adjectivos e nomes, adjectivos e verbos.
Para cada uma destas categorias gramaticais, para além de aplicar os modelos simples, onde somente se retira ou não a pontuação, *stop-words* e *stemming*, também foi aplicado o conceito de *NGrams* para os valores de n já indicados anteriormente. Assim sendo, foram criados para cada categoria gramatical 8 combinações + (8 combinações * 4 valores de n), ou seja, 40 modelos. Tendo em conta que foram aplicados seis grupos de categorias gramaticais totalizam-se 40 * 6 = 240 modelos.

No total, foram criados 8 modelos de BOW + 32 modelos *NGrams* + 240 modelos *Pos-Tag* (sendo alguns com *NGrams* também) totalizando 280 modelos de representação de texto.

Como era de esperar, durante a criação dos diversos modelos surgiram vários problemas, tanto com a ferramenta R como com a criação dos modelos em si. Foi necessário efectuar vários testes para se perceber a influência das várias etapas e a relação existente entre elas.

Após alguns resultados falhados, observou-se que as etapas que exigem mais atenção e cuidado são a remoção das stop-words e a fase de *POS-Tag*.
Mais à frente neste relatório são apresentadas as análises e os problemas/obstáculos detectados ao longo da criação dos modelos de representação de texto.

Para o desenvolvimento destes modelos foi preciso arranjar forma de criar funções genéricas que permitam executar somente as fases do pré-processamento necessárias para cada modelo a conceber.

4.3.3. Função genérica de criação dos modelos no R

Após definir os vários modelos a serem construídos, foi desenvolvida uma função no R que permite criar cada um dos modelos, para posterior classificação.

PreProcessingFunction (dataCorpus, stringModels, valueNGram)

- dataCorpus: corpus
- stringModels: é uma *string* de três dígitos binários que define o que vai ou não ser aplicado. A primeira posição corresponde à remoção ou não da pontuação, a segunda posição corresponde à remoção ou não das *stop-words* e por fim, a terceira posição à aplicação ou não do *stemming*. Por exemplo, a combinação 101 corresponde a: sem pontuação, com *stop-words* e com *stemming*.
- valueNGram: indica o valor de n para o modelo de *NGrams*. Caso este valor seja igual a 0 indica que não é aplicado o modelo de *NGrams*. Se for superior a 0 indica que é aplicado o modelo de *NGrams* e qual o valor a usar para n.

No caso dos modelos *POS-Tag* foi construída uma função prévia que efectua a atribuição de *tags* e posteriormente selecciona somente as palavras das categorias gramaticais pretendidas e guarda os resultados em ficheiro.

SavePosTagFiles(dataCorpus, GramaticalCategory)

- dataCorpus: corpus
- GramaticalCategory: indica a ou as categoriais gramaticais para o modelo pretendido.

Depois, ao usar a função PreProcessingFunction, o corpus enviado no campo dataCorpus corresponderá aos comentários guardados previamente, ou seja, só as palavras das categorias gramaticais solicitadas.

O resultado final de cada modelo é uma matriz TFxIDF que será posteriormente usada para o processo de aprendizagem e classificação.

Foram encontradas algumas dificuldades para a criação destes modelos, nomeadamente para a metodologia de *POS-Tag*. De seguida são descritas as análises e obstáculos encontrados ao longo desta fase.

4.3.4. Instanciação dos modelos de representação

Nesta área, estão expostas todas as análises necessárias para entender o comportamento de certas funções no R e as influências que têm para a concepção dos vários modelos. São também descritos os vários problemas que foram surgindo e a forma como foram resolvidos ou contornados.

4.3.4.1. Análise da remoção de *stop-words*

Como já foi referido anteriormente neste relatório, as *stop-words* são termos considerados irrelevantes e somente desempenham um papel funcional. Para cada língua existe uma lista desses termos. Neste caso foi usada a lista disponível na livraria tm do R, acessível através do método *stopwords*(língua).
De seguida, são apresentadas algumas das *stop-words* usadas na lista de *stop-words* disponível no R.

Tabela 2 - Alguns exemplos de *stop-words* (inglês)

a	*are*	*do*	*how's*	*gets*
large	*last*	*its*	*is*	*from*
here	*full*	*gets*	*kind*	*many*
possible	*shan't*	*Than*	*to*	*we*
until	*wants*	*your*	*Yes*	*you're*

Pode-se constatar que esta lista é, como se esperava, composta por palavras simples sem qualquer pontuação associada. Como tal, um dos problemas detectados na remoção das *stop-words* baseou-se no facto do R considerar a palavra e a pontuação como uma palavra só, como já foi explicado anteriormente no capítulo 4.2. Logo, ao remover as *stop-words* algumas delas não eram removidas. Por exemplo: a palavra "are" consta da lista de *stop-*

words, no entanto quando temos “are,” a *stop-word* “are” não é removida, isto porque a palavra considerada é “are,” e não “are”.
Isto acontece somente nos modelos em que a pontuação não é removida antes de remover as *stop-words*. Então, nestes casos é necessário remover provisoriamente a pontuação, de seguida remover as *stop-words* e por fim voltar a colocar a pontuação, nas palavras que ainda se mantêm após a remoção das *stop-words*.

4.3.4.2. Análise da representação POS-Tag

Para a construção dos modelos *POS-Tag* foi necessário efectuar uma análise do comportamento da atribuição das categorias gramaticais ou *tags* para perceber qual a melhor metodologia a seguir.

As *tags do part-of-speech* usadas no R estão disponíveis em anexo neste relatório.

O objectivo desta análise foi perceber a influência das *stop-words* e da pontuação para a fase da atribuição das *tags*, deduzindo logo que o *stemming*, uma vez que altera a palavra, seria sempre realizado após qualquer remoção ou escolha de palavras a reter.

- Exemplo prático

Usando a frase: “This kind of film requires us to be very forgiving, and if we are, it promises to entertain. Angels & Demons succeeds.”

É usada a função tagPOS, existente na livraria *openNLP* do R, para atribuição das categorias gramaticais.

Após aplicação da função à frase original obtém-se o seguinte resultado:
"This/DT kind/NN of/IN film/NN requires/VBZ us/PRP to/TO be/VB very/RB forgiving,/RB and/CC if/IN we/PRP are,/VBP it/PRP promises/VBZ to/TO entertain./VB Angels/NNPS &/CC Demons/NNPS succeeds./."

De seguida, efectuou-se o teste removendo os espaços em branco extra e convertendo a frase para minúsculas. Verificou-se que não existe qualquer influência na atribuição das *tags*. Como tal, o primeiro passo consiste em remover estes dois itens.

Posteriormente, procedeu-se à remoção das *stop-words* antes de aplicar a função tagPOS, adquirindo o seguinte resultado: "film/NN requires/VBZ forgiving,/JJ are,/NN promises/VBZ entertain./DT angels/NNS &/CC demons/NNS succeeds./."
Como se pode ver, certas palavras foram categorizadas de maneira diferente. Por exemplo, a palavra “forgiving,”, inicialmente foi categorizada como adverbio (RB) e aqui é categorizada como adjectivo (JJ). Também se constata que certas *stop-words* não foram removidas uma vez que estão agregadas a uma pontuação. Mas este caso já foi abordado no subcapítulo anterior (4.3.4.1).

Por fim, averigua-se novamente que removendo a pontuação antes de aplicar a função tagPOS adquire-se diferentes *tags* para as mesmas palavras: "this/DT kind/NN of/IN film/NN requires/VBZ us/PRP to/TO be/VB very/RB forgiving/JJ and/CC if/IN we/PRP are/VBP it/PRP promises/VBZ to/TO entertain/VB angels/NNS demons/NNS succeeds/VBZ".

O mesmo acontece removendo os dois, ou seja, a pontuação e depois a *stop-words*: "This/DT film/NN requires/VBZ forgiving/VBG promises/NNS entertain/VB Angels/NNPS Demons/NNPS succeeds/VBZ".

Foi possível também observar, que quando se aplica as categorias gramaticais à frase original, não é atribuída nenhuma categoria gramatical à última palavra.
Após alguns testes em diversos comentários constatou-se que por vezes é colocada a *tag* e que por vezes não é. Verificou-se também que estes casos só acontecem com a última palavra do documento.
Foram analisados vários documentos com diversos comentários e todos continham o mesmo modo de terminar. Mesmo após pegar em certas frases isoladamente e aplicar o método *tagPOS*, ele continuava a ter o mesmo comportamento, ou seja, tanto colocava a *tag* como não colocava.
Não conseguindo determinar o que se passava e para não se perderem palavras, optou-se por remover a pontuação final antes de atribuir a *tag*, repondo-a novamente após a atribuição da *tag*. A pontuação será colocada antes da *tag* dada à palavra em questão, mesmo sabendo que esta tem um papel importante na atribuição da *tag*. Por exemplo: na frase apresentada para os testes, removendo o ponto final fica somente a palavra “succeeds”, de seguida é atribuída a categoria gramatical e fica “succeeds/VBZ”, por fim volta-se a colocar a pontuação antes da categoria gramatical para que a palavra correcta se mantenha sempre, ou seja, “succeeds.”, obtendo então “succeeds./VBZ”. Conseguindo assim contornar o problema enunciado.

- Conclusões

Como era de esperar, a remoção de uma qualquer palavra ou pontuação tem influência na definição da categoria gramatical das palavras.

O primeiro fluxo de dados criado consistia em, remover os espaços em branco extra, de seguida converter o corpus para minúsculas e posteriormente atribuir a categoria gramatical. No entanto, depois procedia-se à remoção da pontuação e *stop-words* antes de seleccionar as palavras das categorias gramaticais desejadas. Porém, como o R considera as palavras separadas por espaço, após a atribuição da *tag* ele vai considerar a palavra e a *tag* como uma palavra só. Por exemplo: “are,” é considerado “are,/VBP”, e ao tentar remover a pontuação ou a *stop-word* dá erro nos métodos usados devido à má formação da frase. Foi então necessário encontrar uma alternativa para permitir aplicar os vários processos consoante os modelos a criar (com ou sem pontuação, com ou sem *stop-words*, etc).

Surgiu a ideia de ter dois *dataset’s* em paralelo, um para o *POS-Tag*, que servia para indicar a categoria gramatical das palavras e o outro, usando novamente o texto original, para o restante processamento. Depois, pegando nas palavras que sobravam no segundo *dataset* e procurando no *output* do *POS-Tag*, para identificar a sua categoria gramatical, guardava-se somente as palavras que correspondiam às categorias gramaticais pretendidas. Isto, para ser possível fazer os vários modelos e não ter erros aquando das alterações ao texto (remoção de pontuação, etc).

Mas, após alguns testes e análises das várias hipóteses, conclui-se que é mais fácil e obtém-se os mesmos resultados aplicando a categoria gramatical, depois seleccionado as palavras das categorias gramaticais desejadas e por fim, aplicando os restantes passos, ou seja, remoção de pontuação, *stop-words,* etc.
No caso das *stop-words*, é necessário ter em conta todos os passos já referidos anteriormente no subcapítulo 4.3.4.1.

4.3.4.3. Problemas encontrados

Um dos problemas mais difíceis de contornar é um problema existente no R, visto que existe um limite de memória para armazenar os objectos criados. Este problema surgiu por diversas vezes na criação dos modelos de representação de dados através do erro: “*Error: allocate vector of size XX Mb*”, onde XX é o tamanho que ele não conseguia alocar.

Para contornar este problema, uma das soluções encontradas foi ter sempre o cuidado de eliminar os objectos criados no R à medida que estes não iam sendo necessários.
Outra das soluções usadas, consistiu em acrescentar o comando *--max-mem-size=2G* no atalho do executável do R para permitir utilizar o máximo de memória possível. No entanto, o máximo que ele permite é 247Mb, o que por vezes pode não chegar quando é necessário usar objectos muito grandes.

Porém, mesmo após todas estas soluções, foi impossível criar todos os modelos planeados. Inicialmente, o corpus utilizado para a construção dos modelos continha 1127 comentários, mas após os vários problemas de memória e a impossibilidade de criação de alguns modelos, foi necessário ponderar uma redução do corpus. Esta opção também surgiu pelo facto de que ainda seria necessário manipular os modelos para classifica-los e avalia-los, o que poderia levar a mais problemas de memória irreversíveis.
Optou-se então pela redução do corpus, ficando somente 500 comentários, após alguns testes com a criação dos modelos que eram impossíveis de criar. Posteriormente, voltou-se a passar por todos os processos necessários à criação dos diversos modelos.

Outro dos problemas de memória que surgiu, menos problemático, foi no *Java Virtual Machine* (JVM), necessário para utilizar os métodos existentes na livraria rJava do R. Neste caso, é possível ao iniciar o JVM no R enviar a memória útil através do comando .jinit(parameters="-Xmx1g"). No entanto, o limite que ele permitiu colocar foi de 1giga, neste caso sendo suficiente para os modelos necessários.

Também foi necessário rever a formatação de alguns comentários, uma vez que o método tagPOS (já referenciado anteriormente no subcapítulo 4.3.4.2) deixava de atribuir as categorias gramaticais aos comentários. Após observação dos comentários, para se perceber onde deixava de atribuir as categorias gramaticais, constatou-se que existiam alguns caracteres (*enters*) que impediam a continuação da atribuição das categorias gramaticais.

Posteriormente à conclusão dos vários modelos, isto é, da obtenção das matrizes TFxIDF, procede-se à fase de classificação dos mesmos.

4.4. Classificação

Para efectuar a classificação dos modelos criados, os classificadores usados foram: SVM e KNN, uma vez que já foi referido anteriormente que são os mais indicados para problemas de classificação de texto (capitulo 2.6).

No caso do classificador KNN foi preciso definir um valor para k, correspondendo ao número de vizinhos mais próximos a considerar. Para abranger grupos de tamanhos variados os valores escolhidos foram k=3, 7 e 11.

Para cada modelo, isto é, para cada matriz TFxIDF obtida na fase de representação de documentos, foi aplicado tanto o classificador SVM como o classificador KNN para cada valor de k definido.

No R, já existem funções que implementam a classificação tanto para os classificadores SVM como KNN, como tal, foi criado um programa que implementasse estes classificadores para aplicar às diversas matrizes TFxIDF.

Para proceder à classificação dos documentos é essencial primeiro treinar o classificador e só depois é que se procede à classificação. Os dados de treino devem conter a respectiva classificação para permitir ao classificador aprender como classificar os dados.

5. Avaliação dos resultados

Por fim, é apresentada a metodologia da avaliação dos resultados, assim como, os resultados obtidos e a análise dos mesmos.

5.1. Constituição do corpus

Como já foi referido em capítulos anteriores, o corpus será constituído por críticas cinematográficas. Além da selecção do tema que compõe o corpus, a escolha da linguagem foi outro dos passos importantes a validar.

A selecção da linguagem centralizou-se entre português ou inglês. Após análise dos algoritmos e métodos já existentes para trabalhar com cada uma das línguas, optou-se por efectuar o corpus em inglês. Isto porque foram encontrados todos os métodos necessários para esta linguagem e sendo uma língua universal, sobretudo na área tecnológica, encontra-se mais fiabilidade nas ferramentas e métodos usados.
Na língua portuguesa já se encontra bastante trabalho desenvolvido e certos métodos/ferramentas existentes permitem seleccionar a língua que pretendemos. No entanto, muitos dos métodos desenvolvidos para português são para português do Brasil como tal, este foi um dos outros factores que nos levou à selecção da língua inglesa.

A etapa mais complicada desta fase centrou-se na pesquisa e recolha de dados válidos para a constituição do corpus. Mesmo seleccionando como tema críticas cinematográficas, nem todos os sites correspondiam ao que se pretendia.
Após várias pesquisas encontrou-se um site (http://www.metacritic.com/) que contém comentários tanto de críticos como do público em geral a filmes. Tendo bastante informação útil para o problema enunciado, optou-se por retirar o conjunto de dados necessário daqui.
Foram seleccionados vários filmes polémicos que obtiveram cotação negativa, positiva e imparcial para se poder abranger todo o tipo de opiniões. Após análise dos diversos comentários, somente foram extraídos os comentários dos críticos, uma vez que estão melhor escritos ortograficamente.
Para constituir o corpus extraíram-se do site da metacritic, os seguintes atributos:

- Nome do filme;
- Comentário/crítica;
- E cotação do crítico (valor inteiro entre 0 – mínimo e 100 – máximo).

Assim, para cada filme, o objectivo é classificar a opinião dos críticos como: **Negativo** (pontuação de 0-20), **Positivo** (pontuação de 80-100) e **Imparcial** (pontuação de 21-79).

Para esta fase, foi criado um programa em R para extracção dos dados necessários das páginas HTML contendo os comentários (cotação do crítico, o comentário e o nome do filme). Foi necessário analisar a página HTML e proceder a alguns testes e ajustes, até se conseguir retirar o conteúdo desejado.

Inicialmente, foram extraídos 1127 comentários de diversos filmes diferentes (num total de 34 filmes), no entanto devido a problemas já explicados no capítulo 4.3.4.3 foi necessário reduzir o corpus para 500 comentários (extraídos agora de 16 filmes distintos).
Sendo um corpus constituído por críticas relativas a diversos filmes distintos, o grau de dificuldade da tarefa de classificação sobre este corpus foi aumentado. Temos então um corpus referente a um mesmo tema (críticas cinematográficas), no entanto são críticas não só de um filme mas de vários.

Pode concluir-se que esta fase foi muito morosa e importante, uma vez que o corpus é a base de todo o trabalho.
A cotação das 500 críticas que constituem o corpus utilizado para a avaliação, distribuem-se como representado nas Figuras 10 e 11.

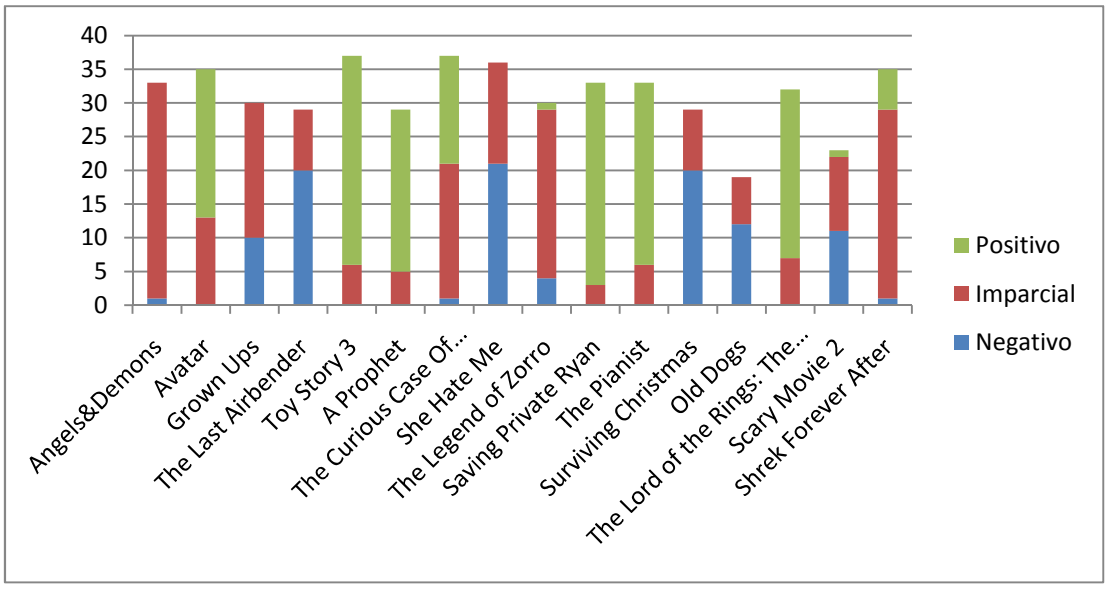


Figura 10 - Distribuição da cotação das críticas no corpus

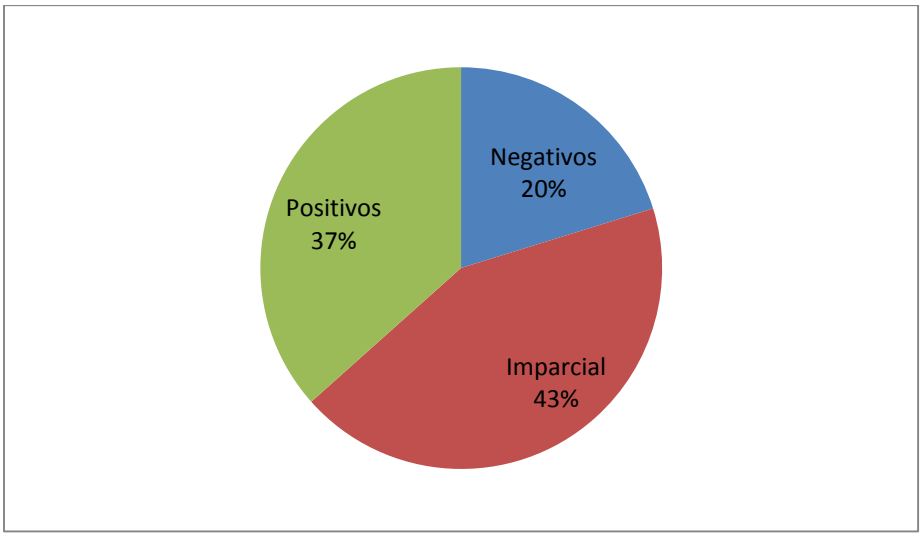


Figura 11 - Distribuição da cotação no corpus

5.2. Plano experimental

Como já foi referido na secção anterior (5.1), foram recolhidas críticas cinematográficas para a constituição do corpus. O objectivo é avaliar e classificar a opinião expressa pelos críticos sobre um determinado filme. No entanto, o corpus efectivamente utilizado contém críticas referentes a vários filmes distintos, aumentado ainda mais o grau de dificuldade do problema de classificação.

Após a criação dos diversos modelos de representação de documentos e a obtenção das matrizes de TFxIDF para cada modelo, procedeu-se a classificação e avaliação dos documentos.
Como já foi referido anteriormente (capitulo 4.4), os classificadores usados foram o SVM e KNN com k=3,7 e 11.

Para cada matriz e classificador, foi usada a técnica de validação cruzada (*cross-validation*).
A técnica de validação cruzada é uma técnica para avaliar como os resultados de uma análise estatística generalizará um conjunto de dados independentes. É principalmente usado em casos onde o objectivo é a previsão, e se pretende estimar com que precisão um modelo de previsão vai executar. [REF17]
O conjunto inicial dos dados foi dividido em k subconjuntos de igual tamanho. Em cada iteração foram usados: 1 subconjunto de treino e k-1 subconjuntos para teste.

10-fold cross validation - A avaliação é realizada 10 vezes. Porquê 10? Experimentação intensiva mostrou que 10 é a melhor escolha para se ter uma estimativa fiável. [Rodrigues, 2008]

Como tal, foi necessário dividir os dados em 10 conjuntos. Uma vez que o corpus é composto por um total de 500 documentos, foram então criados conjuntos de 50 documentos.

Para repartir os comentários aleatoriamente entre os conjuntos de teste e treino, foi criado um vector com índices aleatórios entre 1 e 500. Este vector foi guardado e usado para todas as experiências realizadas, a fim de manter uma coerência nas experiências.
Para cada matriz e para cada classificador, foram corridas as 10 experiências e para cada uma delas calculou-se: a matriz de confusão, a precisão e a abrangência.
A precisão e abrangência, de cada matriz, foram calculadas através da média da precisão e abrangência das dez experiências. Usando os valores da precisão e da abrangência é possível calcular o valor do *f-measure*.
Os valores obtidos foram exportados para uma folha .csv a fim de serem analisados para posterior interpretação dos resultados.

A matriz de confusão apresenta o número de classificações correctas sobre o número de classificações previstas para cada classe. As classes representam as categorias que queremos classificar, sendo neste caso: positivo, negativo ou imparcial. A informação contida nas matrizes de confusão permitiu calcular a precisão e abrangência.

Para melhor compreensão, foi atribuído a cada modelo um nome correspondente às suas características. Cada nome tem a seguinte nomenclatura: Mxyz_n_t, onde cada parâmetro tem o seguinte significado:

- x – pontuação (se 0 removeu-se pontuação se 1 não se removeu)
- y – *stop-words* (se 0 removeram-se as *stop-words* se 1 não se removeram)
- z – *stemming* (se 0 não se fez *stemming*, se 1 fez-se)
- n – *NGrams* (se 0 BOW, caso contrário número de termos (N) em cada *gram*)
- t – tipo de POS tagging (*null* se não se fez *Pos-Tag*; VB caso se tenha aplicado a *tag* verbos, etc)

5.3. Apresentação dos resultados

O modelo ideal é aquele que maximiza as medidas de precisão e abrangência [Rodrigues, 2008]. Estas medidas são relacionadas através do cálculo do *f-measure*, logo um valor elevado do *f-measure* assegura valores elevados de precisão e abrangência.

Os gráficos e tabelas apresentados reflectem os resultados obtidos, recorrendo à técnica da validação cruzada para 10-fold, para os 280 modelos ensaiados (ver capítulo 4.3.2) aplicados aos classificadores automáticos utilizados (SVM e três classificadores KNN, para 3, 7 e 11 vizinhos).

5.3.1. Precisão, abrangência e *f-measure*

A análise da precisão, da abrangência e do *f-measure* (Figura 12) mostra que a abrangência, quando comparada com a precisão, é praticamente constante e independente do modelo de representação utilizado. Assim, a medida *f-measure* segue, naturalmente, as variações verificadas na precisão.

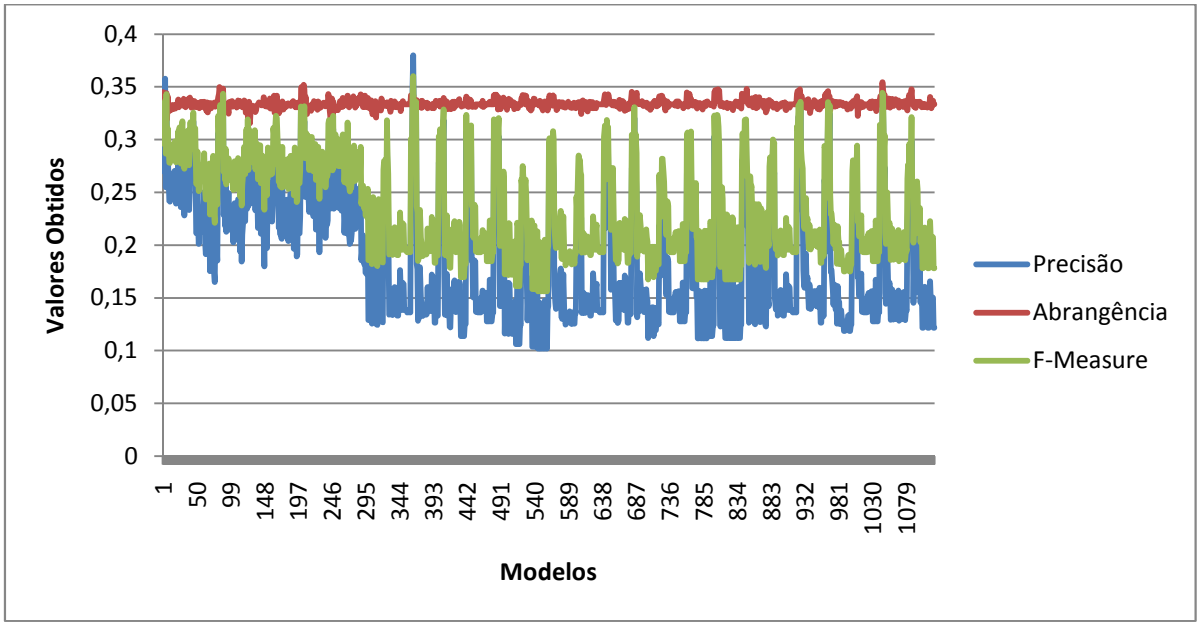


Figura 12 - Valores da precisão, abrangência, *f-measure* observados nos vários modelos ensaiados

A distribuição dos valores da precisão, da abrangência e do *f-measure* pode observar-se de uma forma mais explícita através dos gráficos box-and-whiskers (Figura 13), onde é evidente a elevada concentração dos valores da abrangência em torno da sua média, com alguns *outliers* na cauda direita da distribuição. Observa-se um enviesamento à direita o

que, aliás não surpreende dado que todas estas distribuições apresentam uma assimetria (*skewness*) positiva (ver mais abaixo a Tabela 3).

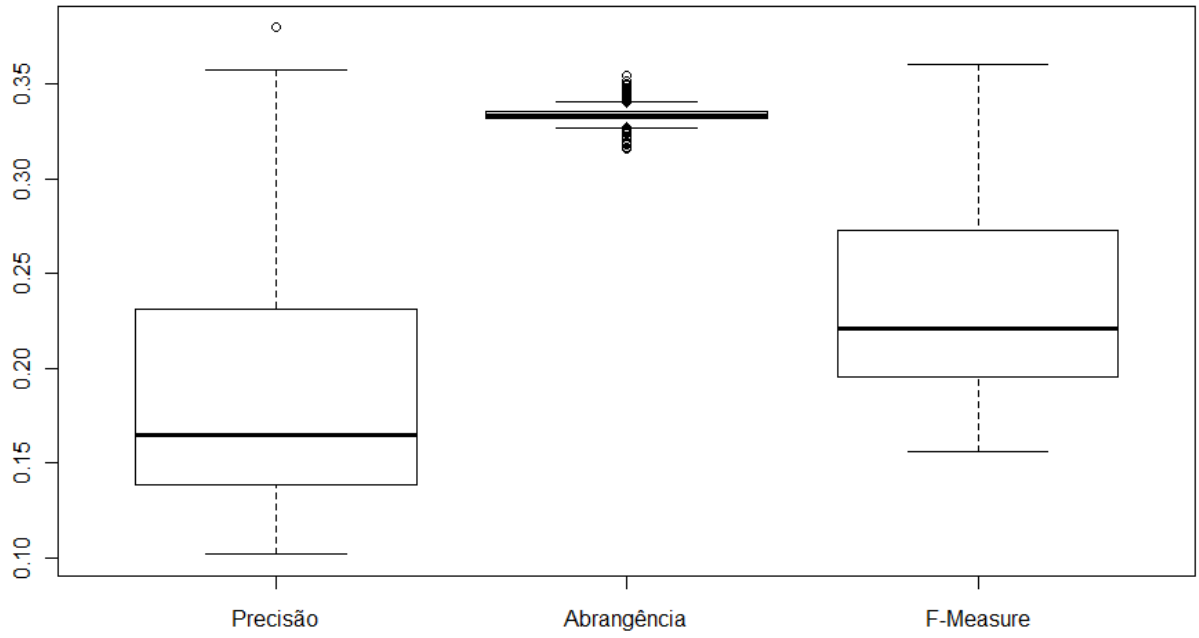


Figura 13 - Distribuição da precisão, abrangência e *f-measure*

Ao analisar as estatísticas descritivas destas três medidas (Tabela 3) confirma-se, como era de esperar, o comportamento da precisão e da abrangência observados na Figura 10.

Como se pode constatar, a variância da amostra (*sample variance*) da abrangência é praticamente nula. A abrangência apresenta uma amplitude de 0,039 e uma média de 0,334 enquanto que a precisão apresenta uma amplitude de 0,278 e uma média de 0,187.

Tabela 3 - Estatísticas descritivas para a precisão, abrangência e *f-measure*

	Precisão	*Abrangência*	*F-Measure*
Mean	0,187	0,334	0,235
Standard Error	0,002	0,000	0,001
Median	0,165	0,333	0,221
Mode	0,136	0,333	0,193
Standard Deviation	0,057	0,004	0,046
Sample Variance	0,003	0,000	0,002
Kurtosis	-0,655	3,462	-1,034
Skewness	0,633	0,567	0,384
Range	0,278	0,039	0,204
Minimum	0,102	0,316	0,156
Maximum	0,380	0,354	0,360
Sum	209,087	373,937	262,797
Count	1120	1120	1120

O valor máximo obtido para a precisão é de 0,380, adquirido para o modelo com as seguintes características: com pontuação, sem *stop-words*, com aplicação do *stemming*, sem *NGrams*, para o modelo de representação *POS-Tag* para a *tag* nomes (M011_0_NN), obtido através do classificador KNN para K=3. E o valor mínimo obtido é de 0,10 para o modelo: com pontuação, com *stop-words*, sem aplicar *stemming* para o modelo *POS-Tag* para as *tags*: nomes, adjectivos e verbos e para *NGrams* com n =5 (M000_5_NNJJVB), usando o classificador KNN para K=3.

No caso da abrangência, o valor máximo obtido foi de 0,354 e o valor mais baixo foi de 0,316. O modelo com melhor resultado tem as seguintes características: sem pontuação, com *stop-words*, sem aplicar *stemming*, sem *NGrams* e para o modelo *POS-Tag* para as categorias gramaticais: nome e adjectivo (M100_0_NNJJ), obtido através do classificador KNN para k=11. No caso do modelo que apresentou menor valor, as características são: sem pontuação, sem *stop-words*, sem aplicar *stemming*, sem *NGrams*, para o modelo *POS-Tag* para as *tags*: verbos e nomes (M110_0_VBNN), resultando do classificador SVM.

Para o *f-measure* o valor máximo obtido é de 0,360 e o valor mínimo é de 0,156.

Uma vez que o valor do *f-measure* é uma medida que agrega a precisão e a abrangência, vamo-nos focar nos seus resultados.

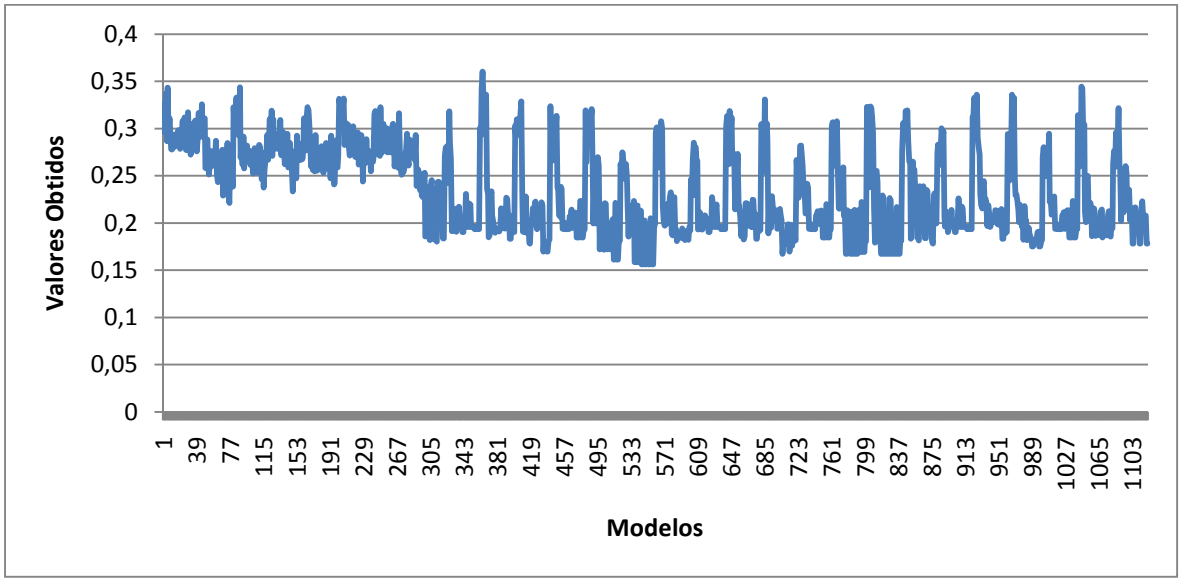


Figura 14 - Valor do *f-measure* nos vários modelos ensaiados

Através da análise dos valores do *f-measure* (Figura 14) é possível observar um padrão que se repete periodicamente.

É também notória a dominância exercida pelos modelos correspondentes aos primeiros 280 pontos do gráfico da Figura 14.

Como já foi referido no início deste capítulo, foram ensaiados 280 modelos de representação de documentos aos quais foram aplicados quatro classificadores (SVM e KNN para k=3, 7 e 11). Os gráficos apresentados nas Figuras 12 e 14 possuem 1120 pontos que correspondem aos 280 modelos de representação ensaiados multiplicados por estes 4 classificadores. Os primeiros 280 pontos correspondem aos valores medidos para o classificador SVM, os seguintes correspondem ao classificador KNN com k = 3, os seguintes para k = 7 e os últimos para k = 11.

Em média, os melhores resultados foram obtidos através do classificador SVM (Figura 15).

5.3.1. Classificadores

É possível perceber que existe um padrão regular presente nos quatro classificadores.

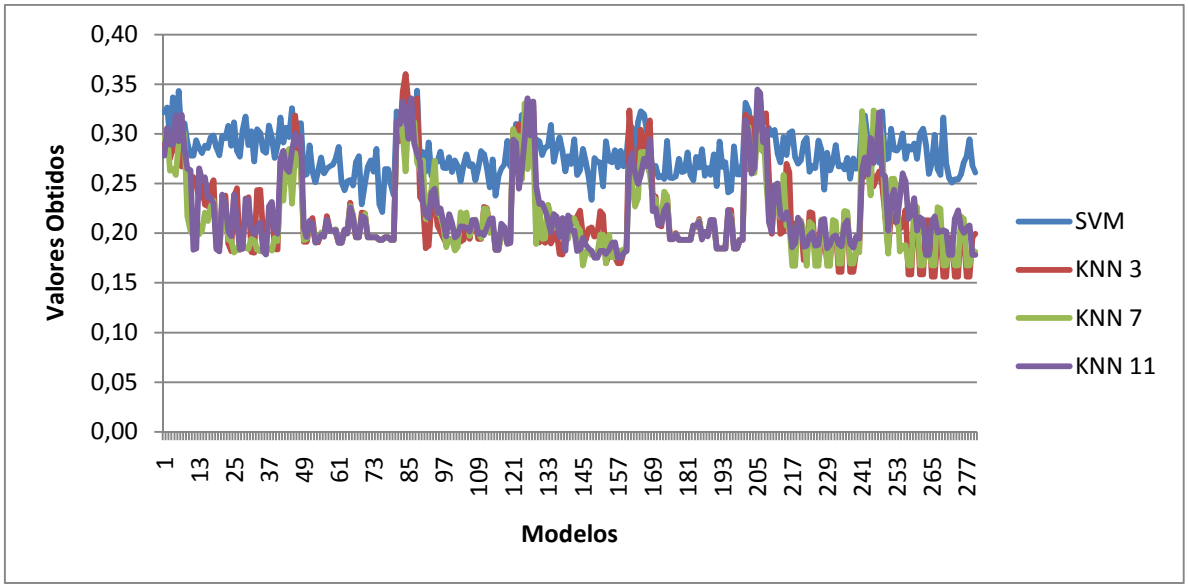


Figura 15 - Valores do *f-measure* para os vários modelos e classificadores

Foi criado um gráfico (Figura 15) que representa as linhas empilhadas, a fim de mostrar a tendência de cada um dos classificadores. Neste gráfico vê-se claramente que existe um padrão que se repete e que é mais acentuado para os classificadores KNN.

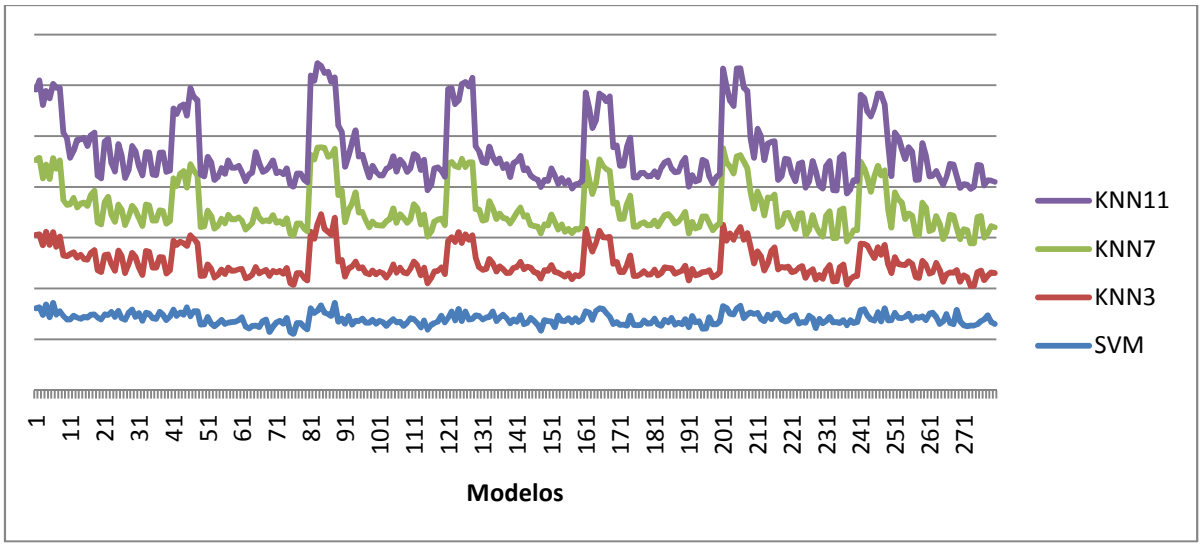


Figura 16 - Valores do f-measure para os vários modelos e classificadores (linhas empilhadas)

O padrão visível parece repetir-se mais ou menos de 40 em 40 modelos de representação. Verifica-se que o número de vizinhos considerados para o KNN não parece ser relevante, uma vez que não se observa nenhum destaque das estratégias (k = 3, 7 e 11) face às restantes.

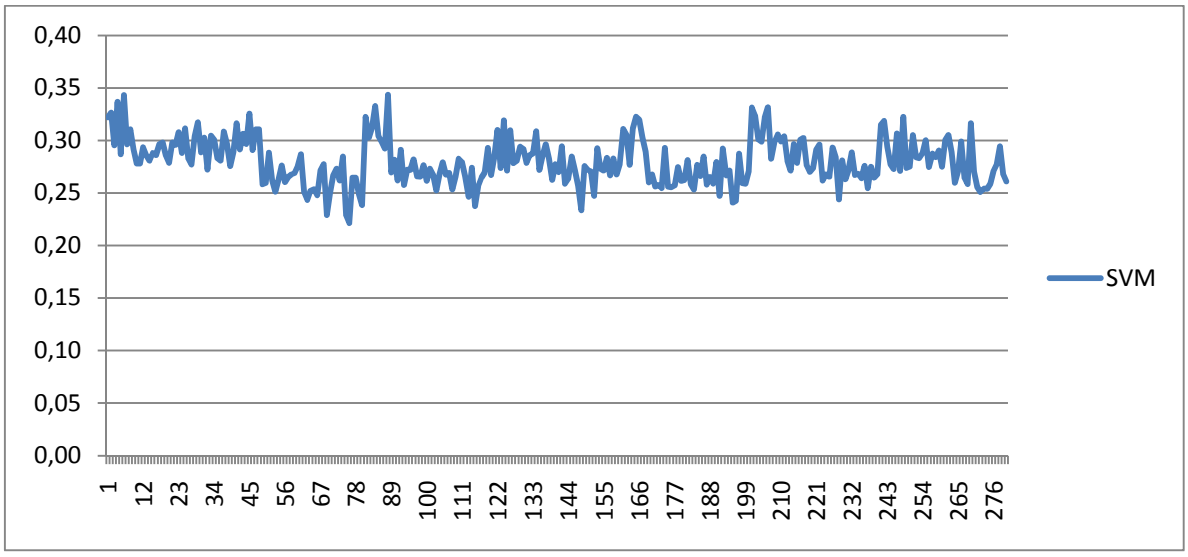


Figura 17 - Valores do *f-measure* para todos os modelos de representação com o classificador SVM

Se nos focarmos na performance do classificador SVM, sobre os vários modelos de representação (Figura 17), é possível observar o padrão que se repete sensivelmente a cada 40 modelos.

O corpus utilizado para estas experiências é somente composto por 500 comentários. Sendo um corpus de pequena dimensão, ao utilizar o método 10-fold *cross validation* só

estamos a usar 50 comentários para treinar o classificador (1/10 de 500). Optou-se então por realizar novamente a parte de classificação considerando somente 3-fold, esperando assim aumentar os valores das medidas de avaliação considerados e demonstrar com mais exactidão quais as melhores características para os modelos de representação.

5.3.1. 3-fold *cross validation*

Através da análise aos resultados obtidos para a precisão, a abrangência e o *f-measure* (Figura 18), continua a ser visível que a abrangência apresenta valores constantes e com poucas variações ao longo de todos os modelos de representação usados.
O valor máximo da precisão é de 61% enquanto que o valor máximo da abrangência é de 49%. O *f-measure* máximo é de 51%.

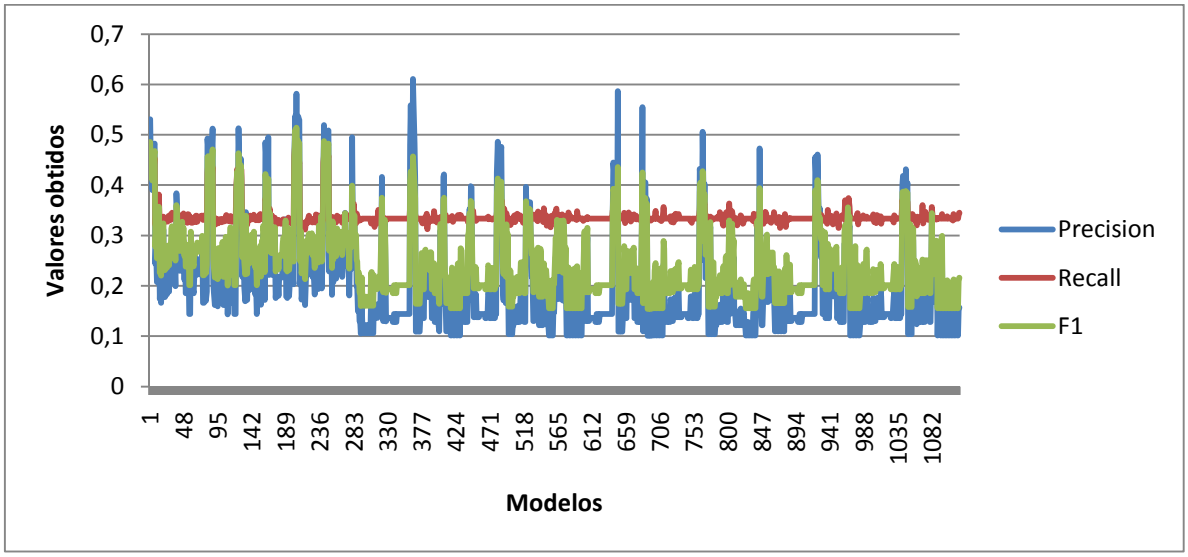


Figura 18 - Valores da precisão, abrangência, *f-measure* observados nos vários modelos ensaiados (3-fold)

Ao observar a representação dos valores do *f-measure*, na Figura 18, é perceptível um padrão regular ao longo dos modelos ensaiados.
Nota-se, como anteriormente com os valores obtidos através de 10-fold *cross validation*, que os primeiros 280 modelos apresentam, consideravelmente, melhores resultados que os restantes. Os primeiros 280 modelos correspondem aos valores obtidos para o classificador SVM (como já foi explicado anteriormente).

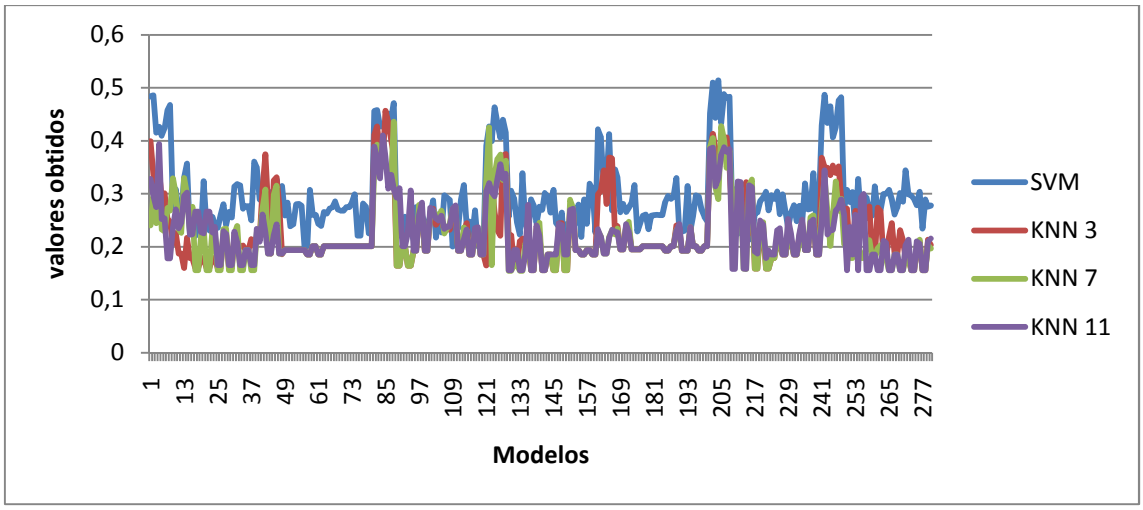


Figura 19 - Valores do *f-measure* para os vários modelos e classificadores (3-fold)

Salvo raras excepções, aqui também é visível, que os melhores resultados foram obtidos através do classificador SVM (Figura 19).

Se nos focarmos somente na performance do classificador SVM sobre os vários modelos de representação (Figura 20), observamos um padrão que se repete sensivelmente a cada 40 modelos.
Esta percepção é muito mais acentuada através dos resultados obtidos usando a técnica da validação para 3-fold do que para 10-fold.

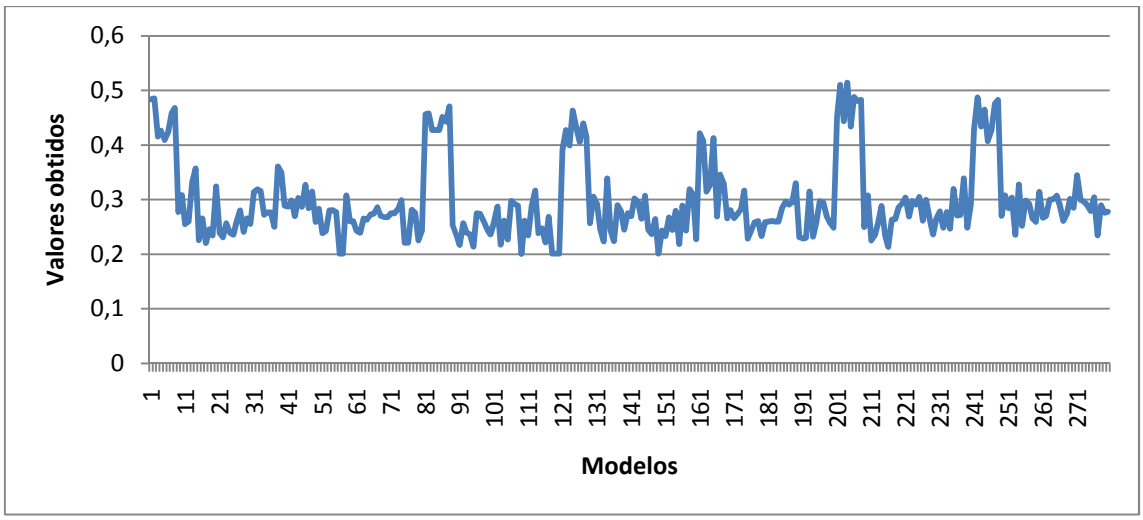


Figura 20 - Valores do *f-measure* para todos os modelos de representação com o classificador SVM (3-fold)

Nota-se que o segundo conjunto de 40 modelos e o quarto conjuntos detêm os valores mais baixos e em cada conjunto existe um pico sensivelmente nos primeiros modelos do conjunto.

5.4. Análise/Discussão dos resultados

Existem sete ciclos de 40 observações, cada ciclo corresponde às sete combinações de categorias gramaticais (*Pos-tagging*) usadas, ou seja, nenhuma, verbos (VB), nomes (NN), verbos e nomes (VBNN), adjectivos (JJ), nomes e adjectivos (NNJJ) e por fim nomes, adjectivos e verbos (NNJJVB).

Em cada um destes sete ciclos verifica-se que os valores mais elevados do *f-measure* correspondem de uma forma geral às 8 primeiras observações, tanto para os resultados obtidos com o 10-fold ou com o 3-fold. Estes valores correspondem aos modelos BOW. Os modelos que recorrem aos *NGrams* (foram analisados *NGrams* para 2, 3, 4 e 5 termos), isto é, aqueles que correspondem aos pontos 9 a 40 em cada ciclo apresentam sistematicamente valores do *f-measure* piores que os 8 primeiros pontos de cada ciclo (correspondentes aos modelos BOW).

Aparentemente, e contrariamente ao que estávamos à espera, os modelos baseados em *NGrams* têm um desempenho inferior aos modelos BOW. É importante agora avaliar quais os factores que determinam oscilações do *f-measure* neste grupo.

Para tal, procedeu-se a uma análise 2k factorial para determinar os efeitos principais dos factores considerados.

5.4.1. Análise 2k factorial

A análise 2k factorial consiste num modelo que tenha k factores e onde pretendemos obter uma percepção da influência de cada um na resposta [Escudeiro, 2002]. Desenvolvendo-se nos seguintes passos:

1. Escolher 2 níveis representativos de cada factor (associar o sinal – a um deles e o sinal + ao outro);
2. Realizar corridas de simulação para cada uma das 2^k combinações possíveis (*design points*);

3. Com base nos resultados obtidos preencher a matriz de desenho (*design matrix*).

O efeito principal do factor j, e_j, é a variação média da resposta devida à mudança do factor j do seu nível "–" para o nível "+", quando se mantêm todos os outros factores fixos. [Escudeiro, 2002]

Os factores considerados para esta análise foram: pontuação, *stop-words*, *stemming*, *Pos-Tag* VB, *Pos-Tag* NN e *Pos-Tag* JJ.

Neste caso, o efeito principal de um factor mede a variação média do *f-measure* quando esse factor é activado. A média foi calculada para o conjunto de todos os modelos de BOW, ou seja, para 56 modelos correspondentes a oito modelos vezes sete configurações.

De seguida, são apresentados os efeitos principais, relativos ao *f-measure*, para os resultados obtidos com o classificador SVM (uma vez que já concluímos que este é melhor) para a validação cruzada 10-fold e 3-fold. Os resultados são apresentados nas tabelas 4 e 5.

Tabela 4 - Efeitos principais sobre o *f-measure* nos modelos de representação com melhor desempenho (10-fold)

Factor	pontuação	stopwords	stemming	postagVB	postagNN	postagJJ
Efeito principal	-0,004	-0,009	0,011	-0,100	0,082	-0,089
%	-0,42	-0,94	1,10	-10,00	8,20	-8,92

Tabela 5 - Efeitos principais sobre o f-measure nos modelos de representação com melhor desempenho (3-fold)

Factor	pontuação	stopwords	stemming	postagVB	postagNN	postagJJ
Efeito principal	-0,005	-0,003	0,014	-0,158	0,199	-0,094
%	-0,49	-0,27	1,39	-15,75	19,92	-9,41

Através destas tabelas (4 e 5) podemos ver que os factores que melhor contribuem para melhorar o desempenho dos classificadores são, por ordem de importância:

1. Considerar somente nomes
2. Aplicar *stemming*

E que, reduzir a representação de um documento aos verbos ou aos adjectivos faz piorar os resultados dos classificadores.

5.4.1. Conclusões da análise

Através dos resultados apresentados é possível concluir que o modelo de representação dos textos não é um factor determinante para a abrangência, que parece não ser muito influenciada pela escolha dos atributos que descrevem os textos. Isto considerando um corpus que apresente características semelhantes àquelas que foram estudadas.

Após comparação dos classificadores usados, conclui-se que o SVM apresenta melhor desempenho que os classificadores KNN. O número de vizinhos considerados, para o classificador KNN, não parece ser relevante isto porque nenhum dos classificadores predomina (k = 3, 7 e 11) face aos restantes.

Relativamente aos modelos de representação de texto, é possível concluir que os modelos *NGrams* não parecem ser os mais adequados para o nosso problema. No entanto, os modelos BOW construídos com base em nomes parecem ser os mais adequados. Podemos também acrescentar que a aplicação de *stemming* também apresentou bons resultados.
Pelo contrário, a construção de modelos com base em verbos e adjectivos não parecem ser adequados para a resolução do problema proposto, uma vez que pioram os resultados dos classificadores.

6. Conclusões

O principal objectivo deste trabalho consistiu em definir uma metodologia adequada para a classificação de texto em categorias que representam conceitos muito próximos e difíceis de distinguir quando se considera o modelo tradicional de BOW. Como já foi referido, estes problemas surgem sobretudo quando se classifica texto referente a um mesmo tema.
Para a resolução deste trabalho, a abordagem adoptada baseou-se na ideia de que os resultados podem melhorar caso se considerem representações de texto mais elaboradas do que as do simples modelo BOW.

Inicialmente foi feito um levantamento do estado de arte relativamente ao processo de classificação de texto. Foram estudadas todas as etapas que compõem este processo para definir a metodologia, mais apropriada, a seguir. A fase de pré-processamento é considerada como uma das mais importantes, visto que é a base de todo o processo e tem impacto em todas as fases posteriores. Foi uma fase que exigiu muita atenção e análise para todas as tarefas realizadas e sobretudo na ordem em que eram executadas.
Optou-se pelos modelos de representação BOW, *NGrams* com n = 2, 3, 4 e 5 e Pos-Tag para as *tags*: verbos, nomes, verbos e nomes, adjectivos, nomes e adjectivos e por fim nomes, adjectivos e verbos. Os classificadores avaliados foram o SVM e KNN para k=3, 7 e 11.

Após o estudo do estado de arte, foram criados vários modelos de representação dos documentos que permitam avaliar a influência das várias tarefas de pré-processamento (remoção de pontuação, remoção de *stop-words*, aplicação do *stemming*) aplicadas aos modelos de representação escolhidos.
Foram criados dois diagramas (Figura 8 e 9) que ilustram as várias alternativas a avaliar. Ao todo foram avaliadas duzentas e oitenta opções.

Ao longo de todo o processo de criação das matrizes TFxIDF, que resultam dos modelos de representação criados, foram surgindo vários problemas que permitiram estruturar e perceber melhor as várias alternativas estudadas.

Para a fase experimental foi utilizada a ferramenta R. Tratando-se de uma ferramenta que desconhecia, foi necessário um período de adaptação à mesma.
Um dos passos mais morosos e difíceis consistiu na escolha e recolha de um corpus que fosse pertinente para o problema proposto. O corpus que foi seleccionado é composto por

críticas cinematográficas, uma vez que estas exprimem opiniões positivas e negativas sobre um mesmo tema, neste caso um filme. O corpus efectivamente utilizado contém críticas referentes a vários filmes distintos, aumentado ainda mais o grau de dificuldade do problema de classificação a tratar.
Depois da criação das matrizes estas foram submetidas ao processo de classificação usando o método do *cross-validation* aplicado a 10-fold *cross validation*. No entanto, como o corpus é de pequena dimensão, foi efectuada também a classificação para 3-fold esperando assim aumentar os valores das medidas de avaliação usados.

Por fim, avaliamos os resultados obtidos para definir qual a melhor estratégia a usar.
Concluímos que, contrariamente ao que se podia esperar, o método de *NGrams* é pouco benéfico nestes casos, no entanto os métodos BOW juntamente com o *Pos-Tag* com base em nomes podem-se tornar bastante interessantes, assim como, a aplicação de *stemming* ao texto na fase de pré-processamento. Porém, o uso de *Pos-Tag* para as categorias gramaticais verbos e adjectivos fazem piorar os resultados dos classificadores, como tal não são adequados ao problema.

Quanto ao classificador, foi possível avaliar que o SVM apresenta melhor desempenho que os classificadores KNN.

6.1. Limitações e trabalho futuro

A grande limitação deste trabalho consistiu no facto da ferramenta R operar com uma área de memória muito limitada. Foi necessário efectuar uma longa pesquisa sobre os problemas de memória, tanto derivados do R como do Java, e ter diversos cuidados com os objectos que estavam a ser usados e somente usar os objectos realmente necessários. Mesmo assim, não conseguia criar todos os modelos pretendidos para a avaliação, tendo de optar por reduzir o corpus que estava a ser utilizado para conseguir criar todos os modelos desenhados.

O facto da ferramenta R, utilizada para a fase experimental, ser nova para mim obrigou-me a um estudo prévio sobre a mesma para tirar melhor partido das suas funcionalidades e aplicar de forma correcta as funções necessárias.

Um dos pontos que seria interessante avaliar e que, de certeza, conduziria a mais conclusões decisivas para este problema, seria aplicar a metodologia descrita a um corpus

que se referisse somente a um tema. Neste trabalho aumentou-se o grau de dificuldade do problema de classificação ao considerar comentários relativos a diversos filmes.

Outro aspecto importante seria efectuar testes para variados corpus, sempre exprimindo opiniões relativas a um mesmo tema, a fim de ter uma avaliação mais rica e variada para obter mais certezas nas conclusões tiradas.

Uma vez que a categoria gramatical revelou-se importante, seria interessante efectuar testes considerando outras *tags* para os modelos de representação *POS-Tag*, a fim de perceber qual a melhor combinação a usar. É importante detectar qual a melhor ou as melhores categorias gramaticais para interpretar emoções, etc.

O classificador SVM é mais adequado para estes casos, porém o classificador KNN também nos ajudou a perceber a existência de um padrão repetitivo. No entanto, as experiências efectuadas (para k=3, 7 e 11) deram-nos valores muito semelhantes não tendo existido nenhum destacamento. Talvez conseguíssemos melhores resultados ao considerar valores para k mais altos e/ou mais espaçados.

7. Bibliografia

Biskri, Ismaïl e Delisle, Sylvain. "*Text Classification and Multilinguism:Getting at Words via N-grams of Characters*". Université du Québec à Trois Rivières Département de mathématiques et d'informatique, 2002

Boali, Li et all. "*An Improved k-Nearest Neighbor Algorithm for Text Categorization*". Institute of Computational Linguistics Department of Computer Science and Technology Peking University, Beijing, P.R. China, 100871, 2003.

Cazella, Sílvio César. "Introdução a Mineração de Textos (KDT)". Unisinos, 2007.

Cardoso, Olinda. "Recuperação de Informação". UNIVERSIDADE FEDERAL DE LAVRAS DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO, 2002.

Chaves, Marcirio Silveira. "Um estudo e apreciação sobre algoritmos de stemming para a Língua Portuguesa." IX Jornadas Iberoamericanas de Informática. Cartagena de Indias, Colômbia, 2003.

Colas Fabrice e Brazdil Pavel. "*Comparison of SVM and Some Older Classification Algorithms in Text Classification Tasks*", 2006.

Dantas, Suzana. "Estrutura de indexação Modelos de RI", 2002.

Deerwester, Scott et all. "*Indexing by Latent Semantic Analysis*", 1996.

Dos Santos, António Paulo Gomes. "Classificação Multi-Etiqueta Hierárquica De Textos Segundo A Taxonomia ACM", Tese em Tecnologia do Conhecimento e Decisão, 2008.

Escudeiro, Nuno Filipe Fonseca Vasconcelos. "*AUTOMATIC WEB RESOURCE COMPILATION USING DATA MINING*". MSc Thesis on Data Analysis and Decision Support Systems, 2004.

Escudeiro, Nuno. "MOSI – Modelação e Simulação". Acetatos de apoio teórico á disciplina de MOSI, 2002.

Even-Zohar, Yair. “*Introduction to Text Mining*”. Automated Learning Group, National Centrer for Supercomputing Applications, University of Illinois, 2002.

Fan, Weiguo et all. "*Tapping the power of text mining*". Communications of the ACM, vol. 49, 2006.

Feinerer, Ingo. “*Introduction to the tm Package Text Mining in R*”, 9 de Fevereiro de 2010.

Fortuna, Blaz. “*String Kernels*”. Department of Knowledge Technologies, 2004.

Galho, Thaís Silva. “Categorização Automática de Documentos de Texto Utilizando lógica Difusa”, 2003.

Garside, Roger. “*The CLAS7S word-tagging system*”, 2004.

Gonçalves, Teresa e Quaresma, Paulo. “*Evaluating preprocessing techniques in Text Classification problem*”. Departamento de Informática, Universidade de Évora, 2005.

Halkidi, M. et all “*Thesus: Organizing Web document collections based on link semantics*”, The VLDB Journal, 12, pp 320-332, 2003

Jaju, Ravindra. “*An Introduction to Text Mining*”, 2004.

Joachims Thorsten. “*Text Categorization with Support Vector Machines: Learning with Many Relevant Features*”. Proceedings of the European Conference on Machine Learning, 1998 .

Lodhi, Huma et all. “*Text Classification using String Kernels*”. Department of Computer Science,Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK, 2002.

Lorena, Ana Carolina e De Carvalho, André C. P. L. F. “Uma introdução às Support Vector Machines”, 2008.

Magalhães, Bruno Nogueira. “Avaliação de métodos não-supervisionados de seleção de atributos para Mineração de Textos”. Fevereiro 2009.

Martins, Cláudia Aparecida. “Uma abordagem para pré-processamento de dados textuais em algoritmos de aprendizado”. USP – São Carlos, 2003.

Miah Muhammed. “*Improved k-NN Algorithm for Text Classification*”. Department of Computer Science and Engineering University of Texas at Arlington, TX, USA, 2009.

Oguri, Pedro. “APRENDIZADO DE MÁQUINA PARA O PROBLEMA DE SENTIMENT CLASSIFICATION”, 2007.

Ramaswamy, Srinivasan. “*Multiclass Text Classification A Decision Tree based SVM Approach*”. School of Information, Univeristy of California, Berkeley, 2006.

Rodrigues, Fátima. “Descoberta do Conhecimento – Data Mining”. Departamento de Engenharia de Informática (DEI/ISEP), 2008.

Rodrigues, Fátima. “Descoberta do Conhecimento – Text Mining”. Departamento de Engenharia de Informática (DEI/ISEP), 2008.

Santos, Bruno. “Níveis de processamento de texto”. 2006

Sarmento, Luís. “Medidas de Associação”. Simpósio Doutoral Linguateca, 3, 4 Outubro 2006.

Sebastiani, Fabrizio. “*Machine learning in automated text categorization*.” ACM Computing Surveys (CSUR). 2002. 1-47.

Sebastiani, Fabrizio. “*Text Categorization*”. Istituto di Scienza e Tecnologie dell'Informazione, Italy, 2005.

Sharp, Mark. “*Text Mining*”. Rutgers University, School of Communication, Information and Library Studies, 2002.

Sholom M. Weiss et all. “*Text Mining: Predictive Methods for Analyzing Unstructured Information*”, 2005.

Sunil, Samatha Gagan. “*openNLP*”. 2006.

Tan, Ah-Hwee "*Text Mining: The state of the art and the challenges*". Proceedings, PAKDD'99 workshop on Knowledge Discovery from Advanced Databases, Beijing, April, 1999.

Venables W. N., Smith D. M. and the R Development Core Team. "*An Introduction to R*". Notes on R: A Programming Environment for Data Analysis and Graphics Version 2.10.0, 26 de Outubro de 2009.

Volk, Martin. "*An introduction to PoS-Tagging and Syntax*". Stockholm University, 2005

Yang, Yiming. "*An Evaluation of Statistical Approaches to Text Categorization*". *Journal of Information Retrieval*, vol. 1, nos. 1/2, pp 67-88. 1999

Yang Yiming e Liu Xin. "*A Re-examination of Text Categorization Methods*". In: Proceedings of 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999.

Yang, Y. e Pederson, J. "*A Comparative Study of Feature Selection in Text Categorization*", International Conference on Machine Learning, 1997.

Zukas, Anthony e Price, Robert J. "*Document Categorization using Latent Semantic Indexing*". Publicado em: *Proceedings, Symposium on Document Image Understanding Technologies, Greenbelt, Maryland*, Abril, 2003

Referências a sites:

[REF1] Cortex Intelligence. "O que é Text Mining?". http://www.cortex-intelligence.com/html/tecnologia/index.html, último acesso em 07 de Fevereiro de 2010.

[REF2] Fauré, Christian. "*Introduction au Text-mining*". http://www.christian-faure.net/2007/05/30/introduction-au-text-mining/, último acesso em 09 de Fevereiro de 2010.

[REF3] Cambridge University Press. "*Naive Bayes text classification*". http://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html, último acesso em 26 de Junho de 2010.

[REF4] Porter, Martim. "*The Porter Stemming Algorithm*". http://tartarus.org/~martin/PorterStemmer/, último acesso em 21 de Agosto de 2010.

[REF5] Yihua, Liao. "Revisão do Método K Nearest Neighbor-categorização de texto". http://www.usenix.org/events/sec02/full_papers/liao/liao_html/node4.html, último acesso em 06 de Julho de 2010

[REF6] Tecnomo, kardi. "*Eucladean Distance*". http://people.revoledu.com/kardi/tutorial/Similarity/EuclideanDistance.html, último acesso em 06 de Julho de 2010.

[REF7] http://en.wikipedia.org/wiki/WordNet, último acesso em 28 de Agosto de 2010.

[REF8] http://www.tlab.it/en/allegati/help_en_online/gtfidf.htm, último acesso em 02 de Setembro de 2010.

[REF9] http://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html, último acesso em 02 de Setembro de 2010.

[REF10] http://en.wikipedia.org/wiki/String_Kernel, último acesso em 05 de Setembro de 2010.

[REF11] http://www.sti.fea.usp.br/conteudo.php?i=564, último acesso em 14 de Setembro de 2010.

[REF12] http://www.r-project.org/, último acesso em 29 de Outubro de 2010.

[REF13] http://bulba.sdsu.edu/jeanette/thesis/PennTags.html, último acesso em 30 de Setembro de 2010.

[REF14] http://en.wikipedia.org/wiki/Latent_semantic_indexing, último acesso em 20 de Setembro de 2010.

[REF15] http://www.stanford.edu/~maureenh/quals/html/ml/node131.html, último acesso em 16 de Outubro de 2010.

[REF16] http://paginas.fe.up.pt/~jcard/publicacoes/tese_html/node80.html, último acesso em 17 de Outubro de 2010.

[REF17] http://www.cs.cmu.edu/~schneide/tut5/node42.html, último acesso em 17 de Outubro de 2010.

[REF18] http://en.wikipedia.org/wiki/P_value, último acesso em 23 de Outubro de 2010.

[REF19] http://reference.findtarget.com/search/vector%20space%20model/, último acesso em 30 de Outubro de 2010.

8. Glossário

Corpus: Conjunto de documentos que servem de base para a descrição ou o estudo de um fenómeno.

Etimologia: Parte da Gramática que trata da origem e formação das palavras.

Latente: Algo que se encontra oculto, dissimulado, disfarçado.

Semântica: Ramo da linguística que estuda o significado das palavras.

Anexo 1 - *Part-of-speech*: *tags* usadas no R

Na seguinte tabela (Tabela 6) [REF13] são apresentadas as *tags* usadas, ao nível das palavras, no R para a utilização do *Pos-Tag*.

Tabela 6 - *tags* usadas no R para aplicação do POS-Tag

Tag	Descrição	Tag	Descrição
CC	Coordenating conjunction	**RB**	Adverb
CD	Cardinal number	**RBR**	Adverb, comparative
EX	Existential there	**RBS**	Adverb, superlative
FW	Foreign Word	**RP**	Particle
IN	Preposition or subordinating conjunction	**SYM**	Symbol
JJ	Adjective	**TO**	to
JJR	Adjective, comparative	**UH**	Interjection
JJS	Adjective, superlative	**VB**	Verb, base form
LS	List item marker	**VBD**	Verb, past tense
MD	Modal	**VBG**	Verb, gerund or present participle
NN	Noun, singular or mass	**VBN**	Verb, past participle
NNS	Noun, plural	**VBP**	Verb, non-3rd person singular present
NNP	Proper noun, singular	**VBZ**	Verb, 3rd person singular present
NNPS	Proper noun, plural	**WDT**	Wh-determiner
PDT	Predeterminer	**WP**	Wh-pronoun
POS	Possessive ending	**WP$**	Possessive wh-pronoun (prolog version WP-S)
PRP	Personal pronoun	**WRB**	Wh-adverb
PRP$	Possessive pronoun (prolog version PRP-S)		