FEP FACULDADE DE ECONOMIA
UNIVERSIDADE DO PORTO

Ano Letivo de 2015/2016

Mestrado em Modelação, Análise de Dados e Sistemas de Apoio à Decisão

Unidade Curricular de Extração de Conhecimento de Dados II

RELATÓRIO DO TRABALHO PRÁTICO I

"Problema de Classificação em contexto de Text Mining"

Docentes: Professor Dr. João Gama
Professor Dr. Pavel Bradzil
Discentes: Rui Pedro Machado, 201300292
Hélder Filipe Russa, 201508409

INDICE

1. Introdução....3
1.1 Contexto....3
2. Desenvolvimento....4
2.1 Processo de classificação de documentos....4
2.2 Conjunto de Dados utilizado....5
2.2.1 Extração do conjunto de dados....5
2.2.2 Tecnologia utilizada....6
2.3 Pré-processamento....6
2.4 Treino do Modelo....11
2.5 Fluxos desenvolvidos em KNIME....15
2.6 Conclusão....17
3. Bibliografia....18
4. ANEXOS....19
4.1 Anexo I – Código R para extração dos dados de treino e teste....19
4.2 Anexo II – Detalhe das experiências realizadas....20

1. INTRODUÇÃO

No âmbito da disciplina de Extração de Conhecimento de Dados I, do Mestrado em Modelação, Análise de Dados e Sistemas de Apoio à Decisão da Faculdade de Economia da Universidade do Porto 2015/2016, foi desenvolvido este trabalho, cujo tema central é o *Text Mining* pretendendo-se desenvolver um modelo de classificação de categorias com base no estudo de padrões em documentos de texto.

1.1 Contexto

Quando navegamos entre páginas online, é comum verificar que grande parte dos artigos, cometários ou opiniões sobre um determinado tema estão organizados de tal forma que é possível associar um determinado texto a um produto, filme ou personalidade. Tomemos por exemplo o caso da plataforma IMDB onde conseguimos navegar entre filmes organizados por categorias e onde para cada filme, temos uma listagem de comentários e opiniões de utilizadores.

Do ponto de vista do consumir de informação, que no caso dos filmes será alguém com interesse em verificar a opinião acerca dos mesmos, estes textos são meramente consultivos e a sua importância muito individualizada e orientada ao tópico que o utilizador está a consultar. No entanto, se tomarmos estes textos como documentos, que podem ser organizados em função da categoria dos filmes e assim utilizá-los como fonte de um modelo de *data mining*, podemos encontrar padrões de utilização de palavras, invisíveis a olho humano que permitam "prever" a categoria de um filme em função das palavras utilizadas para descrever o mesmo.

Da mesma forma que este tipo de técnicas se pode aplicar a filmes, podemos igualmente utilizá-la para livros, classificação de personagens históricas e até mesmo personagens de filmes e videojogos. Tal aplicação pode ser útil para classificar um conjunto de documentos de forma automatizada e assim permitir poupar tempo com classificações manuais. Quando nos referimos a técnicas, é de denotar que as mais utilizadas em *Text mining* são a classificação de documentos (selecionada para este trabalho), *sentiment analysis* ou *clustering*.

Este trabalho consiste assim na classificação de dois grupos de documentos de texto, avaliando por fim a eficácia utilizando alguns classificadores já conhecidos. Os dados escolhidos dizem respeito a descrições de personagens de videojogos e o objetivo será treinar um classificador para que consiga prever se um dado documento diz respeito a um "Herói" ou a um "Vilão" de um videojogo.

2. DESENVOLVIMENTO

2.1 Processo de classificação de documentos

A tarefa de classificação de documentos de uma forma simples, consiste na associação de um documento a uma ou mais classes, utilizando um dicionário de termos encontrados nesses mesmos documentos. Esta tarefa tem bastantes aplicações tais como *spam filtering* onde os emails são classificados como spam ou non-spam, indexação para recuperação de documentos, ou até organização e manutenção de vastos catálogos disponíveis na internet.

A categorização é feita considerando as palavras que aparecem nos documentos e a sua frequência (*Bag of Words*). A coleção de documentos é transformada numa matriz onde as colunas são os termos/palavras extraídas desses mesmos documentos e onde é armazenada informação sobre a sua frequência para cada uma das linhas/documentos. Posteriormente, depois de treinado o algoritmo de classificação podemos utilizá-lo na classificação de novos documentos.

Dado que diversas palavras ou até mesmo a pontuação podem adicionar entropia e assim piorar a *performance* dos classificadores, existem várias técnicas de pré-processamento que permitem melhorar a classificação. Neste trabalho foram utilizadas diversas destas técnicas e os resultados da sua aplicação serão demonstrados nos capítulos seguintes.

2.2 Conjunto de Dados utilizado

O conjunto de dados de treino é constituído por 100 documentos relativos a descrições de heróis de videojogos assim como 100 documentos relativos a vilões, totalizando assim 200 documentos para o treino do classificador. Como temos o mesmo número de "categorias", podemos afirmar que os dados são balanceados.

Relativamente ao formato dos dados, estes não se encontravam estruturados numa base de dados relacional, nem em ficheiros, dado que estes estavam embebidos numa página web, pelo que foi necessário desenvolver um método por código, que permitisse automatizar esta extração. Como nota de interesse os endereços para obtenção dos dados foram os seguintes:

- Vilões http://www.gamesradar.com/top-100-villains-video-games/
- Heróis http://www.gamesradar.com/top-100-video-game-heroes/

2.2.1 Extração do conjunto de dados

O método escolhido para extrair os dados foi através da linguagem R, utilizando os conceitos de *web mining* aprendidos durante as aulas desta mesma disciplina. De seguida uma pequena descrição do pseudocódigo utilizado para a obtenção dos documentos (O código completo pode ser encontrado no anexo I).

I. Obter a página HTML através da função `htmlTreeParse`
II. Construção do código XPath para extrair de todas as "div" html com as descrições da personagem - o nome do mesmo.
    - `'//article[@class="gallery-image" and @id!="article-item1"]/h3'`
III. Construção do código XPath para extrair de todas as "div" html com as descrições da personagem - a descrição do mesmo.
    - `'//article[@class="gallery-image" and @id!="article-item1"]/div[@class="gallery-text"]'`

IV. Através de uma função própria, `getHtmlData`, combinamos os diferentes textos num só objeto data.table de R.

V. Iterar sobre a data.table obtida no passo anterior e gravar os textos para uma pasta, separando-os por ficheiros.

- Eg: `write(fileBody,paste('heroes_', fileName,'.txt',sep=''))`

2.2.2 Tecnologia utilizada

Para a realização deste estudo, devido a um nível de experiência considerável de outros trabalhos realizados, foi utilizada a plataforma KNIME dado permitir, devido ao seu ambiente de desenvolvimento visual, testar diferentes alternativas e métodos de uma forma rápida, intuitiva e fácil.

2.3 Pré-processamento

Como forma de redução do conjunto de palavras a utilizar na classificação, um conjunto de técnicas de pré-processamento disponíveis no KNIME foram utilizadas. Caso fossem utilizadas todas as palavras a classificação seria bastante mais demorada e provavelmente menos eficaz pois algumas das palavras seriam bastante frequentes, tais como verbos ou proposições, mas levariam a classificações erradas. É importante relevar que estas técnicas têm uma dupla função, nomeadamente servir para seleção de atributos relevantes (*Feature Selection*) e igualmente de limpeza de dados, sendo que como complemento da primeira foi ainda utilizada na fase de treino do modelo uma técnica baseada em *Information Gain* para selecionar os termos com maior importância para a "previsão" da classe.

Na imagem seguinte podemos visualizar através da componente de *tagcloud* da libraria de *Text mining* da plataforma KNIME, as frequências de termos extraídas dos documentos relativos a vilões, utilizando como base os documentos originais sem qualquer transformação. Tal como é visível, sem qualquer técnica de pré-processamento, os termos mais frequentes são pontuações e palavras auxiliares tais como "*the*" ou "*a*", sem qualquer relevância para a tarefa de classificação e que causam decréscimo de *performance* na execução da fase de treino, com o aumento de termos a utilizar pelos classificadores e assim devem ser retirados. É interessante comparar esta imagem obtida com os documentos originais com uma outra resultante dos documentos transformados pelas técnicas de pré-processamento.

Figure 1 - Tagcloud inicial para a categoria vilão

Tal como previamente descrito, foram tomados como fonte de dados, o total de documentos extraídos, na sua versão original e aplicado um conjunto de técnicas de pré-processamento. Estas, disponíveis no conjunto de nós de *Preprocessing* do laboratório de *Text processing* do KNIME, são descritas na lista seguinte.

- ***Punctuation Erasure***
  - Componente que permite apagar os sinais de pontuação da lista de termos. De acordo com a *tagcloud* previamente ilustrada, é de esperar que para vilões mude devido à aplicação desta técnica.
- ***N Chars Filter***
  - Componente que filtra todas as palavras com menos do que N caracteres. – No nosso trabalho foi escolhido ***(N=3) ,*** o que significa que apenas termos com mais de três caracteres serão utilizados.
- ***Number Filter***
  - Componente que filtra todos os termos numéricos.
- ***Stop word filter***
  - Componente que filtra todos os termos que estão presentes numa dada lista de palavras proibidas.
- ***Porter Stemmer***
  - Componente que seleciona uma forma representativa para cada palavra.
- ***TF (Term Frequency)***
  - Componente que calcula o número de vezes que cada termo ocorre (absoluto ou relativo) – No nosso trabalho foram utilizadas as frequências absolutas.
- ***Row Filter***
  - Filtra as linhas de acordo com um certo critério. – No nosso trabalho foi selecionado ***(TF absoluta > 3)***, para que apenas termos com mais de 3 ocorrências sejam utilizados.

De seguida podemos visualizar a *tagcloud* da categoria vilão após a aplicação das técnicas de pré-processamento. Através de uma rápida análise podemos verificar que a palavra *villain* é central, no entanto, com maior interesse, termos como "*kill*", "beat" ou "*phsyco*", são centrais, o que mostra alguma coerência dada a natureza deste tipo de personagem.

Figure 2 - Tagcloud para categoria vilão após pré-processsamento

De seguida podemos analisar, por questões informativas a mesma *tagcloud* para a categoria herói, onde é visível que o termo mais utilizado neste tipo de documentos é "*hero"* mas também "*save*" ou "*make*" têm altas frequências.

Figure 3 - Tagcloud para categoria herói após pré-processamento

Como forma de resumo, a implementação de todas estas técnicas descritas em KNIME, podem ser visualizadas na ilustração seguinte, correspondente a um *metanode* que agrega os diferentes nós de pré processamento, identificados pelo seu nome. De realçar que para as diferentes experiencias de treino realizadas, estes nós foram sendo adicionados e removidos, pelo que nesta ilustração temos apenas a última experiencia realizada, com todas as técnicas em utilização.

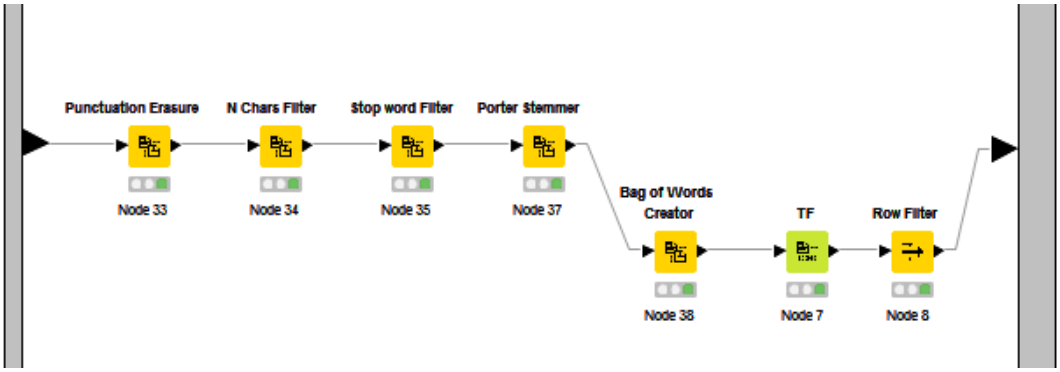

Figure 4 - Pré processamento em KNIME

A redução da dimensionalidade, ou seja, a existência de um grande número de termos/variáveis é um dos grandes problemas em *Text mining* pois afeta a performance e a qualidade dos classificadores.

A aplicação das técnicas previamente apresentadas, permitiram reduzir o número de termos, dos 5489 iniciais para os 168 termos finais, tal como é possível visualizar no seguinte gráfico resumo.

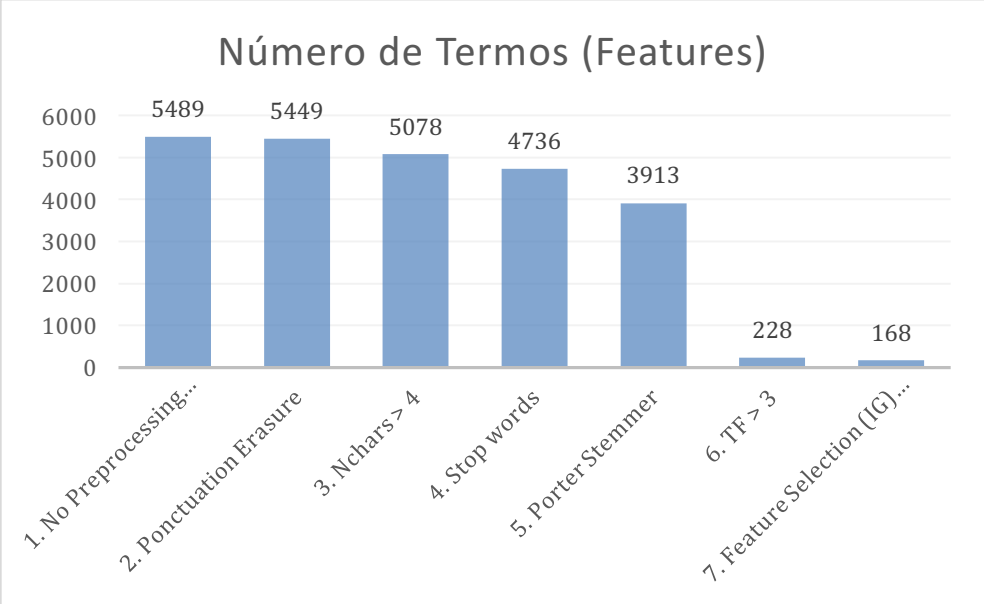

Figure 5 - Evolução do número de termos com aplicação das técnicas de pré-processamento

2.4 Treino do Modelo

Terminada a fase de pré-processamento passamos ao treino e respetiva classificação do conjunto de teste e a avaliação dos diferentes classificadores utilizados. Para a elaboração deste trabalho optamos por avaliar os seguintes classificadores:

- Árvores de decisão (DT);
- 3-vizinho mais próximo (3-NN);
- *Naive Bayes* (NB);
- *Support Vector Machines* (SVM);

Durante a execução deste trabalho não foram estudados os efeitos da variação nos parâmetros dos classificadores sendo que os utilizados foram os presentes no KNIME por defeito. Para a avaliação dos classificadores será utilizada a taxa de erro, ou seja, a percentagem de instâncias incorretamente classificadas.

No gráfico seguinte encontram-se representadas as taxas de erro dos classificadores ao longo das experiências realizadas. De denotar que as experiências são cumulativas relativamente às técnicas utilizadas. Assim o gráfico deve ser lida como tendo 7 experiências e onde na primeira não foi utilizada qualquer técnica de pré-processamento, na segunda utilizou-se *Ponctuation eraser*, na terceira além da segunda utilizou-se remoção de palavras com menos de 4 caracteres e assim sucessivamente. Neste gráfico a percentagem de dados de treino utilizada foi de 80%.

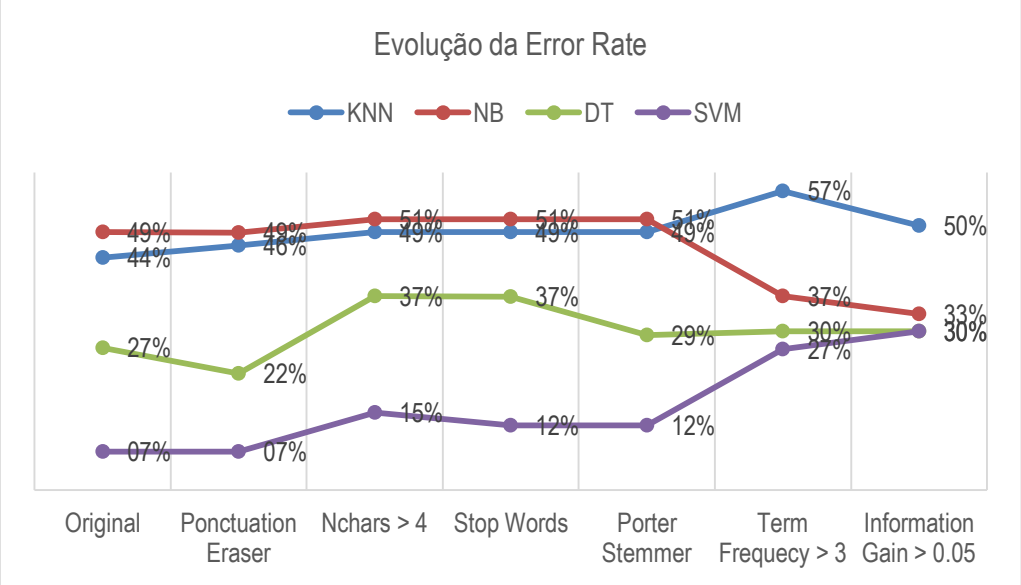

Figure 6 - Evolução da error rate ao longo das experiências

Através do estudo do gráfico anterior concluímos que o classificador SVM apresenta a menor taxa de erro ao longo das experiências e chega inclusive a ter uma taxa bastante interessante de apenas 7% de instâncias classificadas erradamente. Desta forma o classificador SVM seria o melhor de entre os testados. Por outro lado, 3NN (KNN com 3 Vizinhos) a par do

Naive Bayes são os que apresentam uma taxa de erro superior. Os resultados detalhados das experiências podem ser encontrados no Anexo II.

A titulo de resumo acerca da performance dos classificadores, na tabela seguinte é possível observar a média das taxas de erro por classificador, onde se valida a constatação anterior de que o SVM é no geral o que tem melhores valores enquanto que o 3NN e NB são os piores.

	% Treino	KNN	NB	DT	SVM
Taxa de Erro	70%	**45,8%**	**47,2%**	28,6%	**19,3%**
	80%	**49,7%**	**46,2%**	30,6%	**16,8%**

Figure 7 - Taxa de erro média entre classificadores em todas as experiências

Considerando apenas a melhor experiencia obtida, nomeadamente com SVM a 80% dos dados para treino e 20% para teste, foi realizada uma análise mais aprofundada da performance obtida, utilizando todas as técnicas de pré-processamento identificadas. A matriz de confusão obtida foi a seguinte:

	Villain^	**Hero^**
Villain	14	2
Hero	7	7

Table 1 Matriz confusão melhor resultado

De seguida por forma a estudar em maior detalhe o comportamento do classificador é representada uma tabela de avaliação do modelo preditivo com o classificador SVM.

	TP	**FP**	**TN**	**FN**	**Recall**	**Precision**	**Sensitivity**	**Specifity**	**F-Measure**	**Accuracy**
Villain	14	7	7	2	0,88	0,67	0,88	0,5	0,76	?
Hero	7	2	14	7	0,50	0,78	0,5	0,875	0,61	?
Overall	?	?	?	?	?	?	?	?	?	70%

Table 2 Medidas de avaliação de modelos preditivos com SVM

Analisando os valores da tabela anterior podemos verificar que o que mais se destaca em 30 casos classificados é que o nosso classificador é sensivelmente melhor a prever casos da classe "Villain" quando comparado com a classe "Hero". Este facto pode ser comprovado olhando para a medida F-Measure, onde o valor da classe "Villain" se aproxima mais de 1 (Valor desejável), tendo um valor observado de 0,76, enquando que a classe "Hero" se fica pelos 0,61. Se olharmos para os valores da TPR e TNR podemos validar tal afirmação. TPR = Sensitivity é de 88% para a classe "Villain" enquanto que se fica pelos 50% na classe "Hero" (Para esta classe fica-se pela mesma performance que um classificador aleatório).

Por último demonstramos o resultado da aplicação de uma análise baseada em curvas ROC, para comparar os classificadores e de uma forma mais cientifica validar a teoria de que o SVM é o melhor classificador. Esta análise teve em conta apenas as probabilidades da classe "**Villain"**.

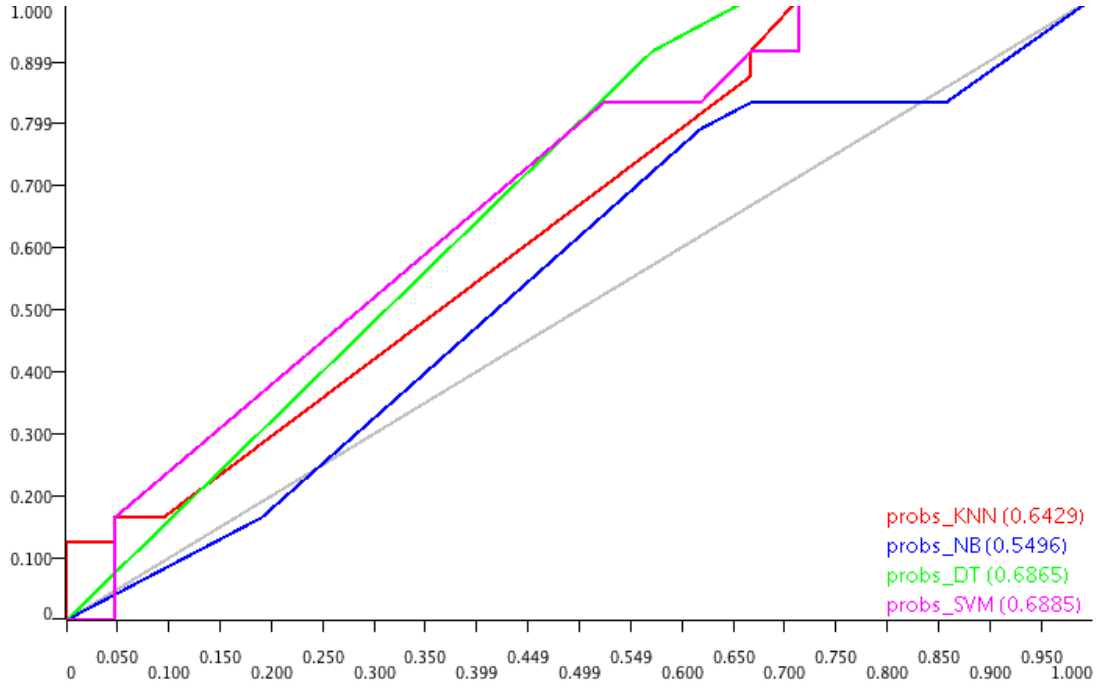

Figure 8 - ROC Curve

Como podemos verificar o classificador com uma maior AUC (Area under curve) é o SVM com 0,6885, no entanto as DT estão muito próximo, o que pode levar a crer que a classificar "*Villains*" estes são muito semelhantes e a diferença estará na classificação dos "*Heroes*"

2.5 Fluxos desenvolvidos em KNIME

Nas ilustrações seguintes podemos visualizar os desenvolvimentos realizados em KNIME. De uma forma resumida existem três secções principais, o fluxo geral onde se pode ver o conjunto de passos necessário para concluir este problema de classificação, nomeadamente a leitura dos dados, a concatenação dos documentos num só objecto de KNIME, o pré-processamento, depois em paraleleo a geração da tagcloud e da matriz de termos do documento e por último a fase de treino dos classificadores. A Figura 8 demonstra o enunciado.

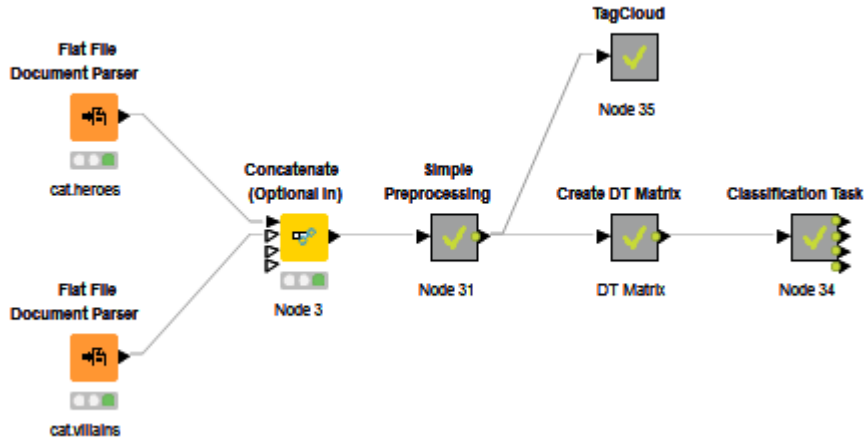

Figure 9 - Fluxo global em KNIME

Dado que o meta nó de pré-processamento já foi apresentado no capítulo dedicado a esta fase, de seguida fazemos um zoom in ao meta nó de criação da matriz de termos do documento onde inicialmente criamos um vetor de documentos, adicionamos o atributo alvo a este vetor e por último filtramos a coluna de documentos para ter apenas o conjunto de termos e o atributo da classe (Atributo alvo).

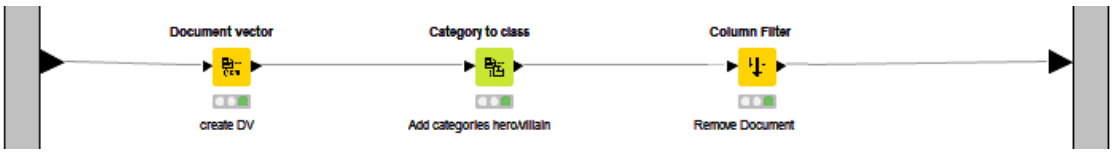

Figure 10 - Criação do Vetor de Documentos

Por último na figura seguinte podemos ver a fase de treino e availiação de resultados dos diferentes classificadores, onde inicialmente se faz um particionamento dos dados em dados de treino e teste, se treinam os modelos e por último de analisa a matriz de confusão através do objeto Scorer.

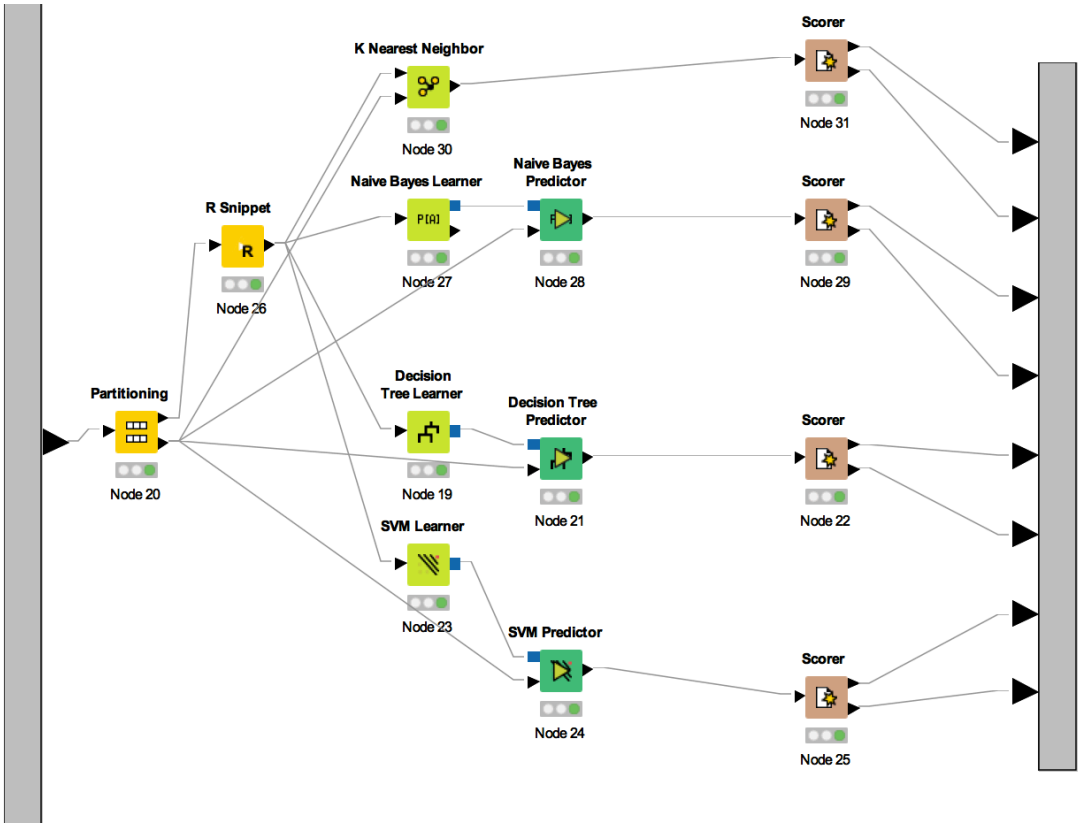

Figure 11 Treino e análise dos diferentes classificadores

2.6 Conclusão

Após todo o trabalho realizado, é importante salientar que o processo mais complicado é certamente o pré-processamento dos dados. Isto porque muito embora o objetivo seja sempre obter a melhor performance com o melhor classificador, em tarefas de *Text Mining* temos de ter em conta o número de termos a utilizar, caso contrário podemos não ter recursos computacionais para terminar a tarefa.

Assim sendo o equilíbrio entre a melhor performance e o número de termos a utilizar, diretamente relacionado com as técnicas de pré-processamento a utilizar, são o grande esforço neste tipo de trabalhos.

Relativamente aos resultados obtidos, podemos afirmar que foram bons para certas experiencias (SVM e DT em cercos casos) e aquém das expectativas noutros casos (NB). Gostaríamos ainda de em trabalhos futuros explorar um pouco mais as potencialidades dos classificadores e fazer variar as suas diversas opções como forma se verificar se a performance melhoraria.

3. BIBLIOGRAFIA

- Slides da Disciplina de Extração do Conhecimento de Dados I, da Faculdade de Economia da Universidade do Porto, 2015
- Slides da Disciplina de Extração do Conhecimento de Dados II, da Faculdade de Economia da Universidade do Porto, 2016
    - *Web Mining*
    - *Text Mining*
- Gama, J., Carvalho, A., Facelli, K., Lorena, A., & Oliveira, M. (2012). *Extração de Conhecimento de Dados*. (E. Silabo, Ed.) (1st ed.). Lisboa: Silabo, Edições.

4. ANEXOS

4.1 Anexo I – Código R para extração dos dados de treino e teste

```
#install.packages("XML")
#install.packages("file")
#install.packages("data.table")
#library(XML)
#library("file")
#library(data.table)

#Function that allow us to remove any Html tag, providing just the text
 inside it.
cleanFun <- function(htmlString) {
  return(gsub("<.*?>", "", htmlString))
}

getHtmlData <- function(inHomePage,inXpathName,inXpathDesc,inNbCases){
  getNames <- as.array(getNodeSet(homePage,inXpathName))
  getData <- as.array(getNodeSet(homePage,inXpathDesc))
  output <- data.table()
  for(i in 1:inNbCases){
    name <- cleanFun(xmlValue(getNames[[i]]))
    desc <- cleanFun(xmlValue(getData[[i]]))
    row <- cbind(Name=name,Desc=desc)
    output <- rbind(output,as.data.table(row))
  }
  return(output)
}

## Get Heroes Data
homePage <- htmlTreeParse("http://www.gamesradar.com/top-100-video-
 game-heroes/",useInternal = TRUE)
heroesXPathNames <-'//article[@class="gallery-image" and @id!="article-
 item1"]/h3'
heroesXPathData <- '//article[@class="gallery-image" and @id!="article-
 item1"]/div[@class="gallery-text"]'
heroes <- getHtmlData(homePage,heroesXPathNames,heroesXPathData,101)
heroes[, Category:='Hero']
write.csv(heroes,"c:\\Users\\ruima\\Desktop\\heroes.csv",row.names=F)
```

4.2 Anexo II – Detalhe das experiências realizadas

			KNN		NB		DT		SVM	
Step	**Training %**	**Pre Process Strategy**	***Accur.***	***Error***	***Accur.***	***Error***	***Accur.***	***Error***	***Accur.***	***Error***
0	70%	-	50,8%	49,2%	50,8%	49,2%	73,4%	26,6%	90,1%	9,8%
1		1	54,0%	46,0%	50,8%	49,2%	75,4%	24,6%	**91,8%**	**8,2%**
2		[1,2]	50,8%	49,2%	49,2%	50,8%	68,9%	31,1%	90,2%	9,9%
3		[1,2,3]	50,8%	49,2%	49,2%	50,8%	73,3%	26,7%	81,9%	18,1%
4		[1,2,3,4]	50,8%	49,2%	49,2%	50,8%	72,1%	27,9%	83,6%	16,4%
5		[1,2,3,4,5]	68,9%	31,1%	62,2%	37,7%	68,9%	31,1%	68,9%	31,1%
6		[1,2,3,4,5,6]	53,3%	46,6%	60,0%	40,0%	68,8%	31,1%	66,6%	33,3%
7		[1,2,3,4,6]	50,8%	49,2%	49,2%	50,8%	72,1%	27,8%	81,9%	18,1%
0	80%	-	56,0%	44,0%	51,2%	48,8%	73,1%	26,9%	92,7%	7,3%
1		1	53,7%	46,3%	51,3%	48,7%	78,0%	22,0%	**92,7%**	**7,3%**
2		[1,2]	51,2%	48,8%	48,8%	51,2%	68,3%	36,7%	85,4%	14,6%
3		[1,2,3]	51,2%	48,8%	48,8%	51,2%	63,4%	36,6%	87,8%	12,2%
4		[1,2,3,4]	51,2%	48,8%	48,8%	51,2%	70,7%	29,3%	87,8%	12,2%
5		[1,2,3,4,5]	43,4%	56,6%	63,3%	36,7%	70,0%	30,0%	73,3%	26,7%
6		[1,2,3,4,5,6]	50,0%	50,0%	66,7%	33,3%	70,0%	30,0%	70,0%	30,0%
7		[1,2,3,4,6]	51,2%	48,8%	48,8%	51,2%	70,7%	29,3%	85,4%	14,6%

Figure 12 - Tabela resumo dos principais resultados obtidos

Legenda: (1) "*Ponctuation Eraser*" (2) "*Nchars > 4*" (3) "*Stop Words*" (4) "*Porter Stemmer*" (5) "*Term Frequecy > 3*" (6) "*Information Gain > 0,05*