Língua Natural

Grupo 7 - Relatório do Mini-Projeto 2

83567 - Tiago Gonçalves
83576 - Vítor Nunes

Introdução

O projecto consiste na classificação, atribuição de categorias, de questões sobre cinema. É fornecido a um classificador um set de questões com categorias previamente conhecidas e depois é possível colocar novas questões e o mesmo atribuir a categoria da mesma.

Proposta de Solução

A primeira abordagem consistiu em utilizar a biblioteca nltk [1] para realizar:

- *Parsing*: ler as questões conhecidas do ficheiro e criar uma estrutura de dados compatível com o classificador.
- *Tokenizing*: separar as palavras da frase e, posteriormente, utilizar os lemas.
- *Stemming*: ignorar palavras que aparecem em todas as frases e não são relevantes. Articuladores e pronomes, designados de *stop words*.
- Naive Bayes: os classificadores utilizados foram baseados em Naive Bayes, que fazem uso da definição de probabilidade condicionada entre lemas.

A métrica utilizada, para atribuir pesos a determinados lemas, correponde a atribuir um valor boleano (True ou False) consoante tenha sido observado anteriormente ou não.

Numa segunda abordagem, após realizada uma pesquisa e análise de um tutorial [8], decidiu-se utilizar:

- *Parsing* e *Tokenizing*: CountVectorizer, que constroi uma matriz com contagens dos tokens, isto é, a frequência com que aparecem nas frases de treino.
- TF-IDF[2]: permite retirar importância às palavras mais comuns entre categorias.
- SGD [3]: é um classificador baseado em *SGD Supervised Learning*. Foi o escolhido por apresentar melhores taxas de *accuracy* como se pode constatar na **Tabela 2**.

[1] Disponível em http://www.nltk.org/
[2] Term-Frequency Times Inverse Document-Frequency
[3] Stochastic Gradient Descent

Resultados experimentais

A primeira abordagem revelou os seguintes dados:

Classificador	**Accuracy**
Multinomial Naive Bayes	$\approx 9.5238\%$
Bernoli Naive Bayes	$\approx 2.3810\%$
Complement Naive Bayes	$\approx 9.5238\%$

Tabela 1: *Accuracy* usando Naive Bayes sobre o ficheiro NovasQuestoes.txt

Os resultados não foram bons, isto porque os algoritmos Naive Bayes utilizam a definição de probabilidade condicionada para determinar a probabilidade de dois lemas aparecerem seguidos.

Porém, o que se pretende é classificar frases em categorias específicas logo foi necessário seguir outra abordagem.

A segunda abordagem, usando a biblioteca SKLEARN revelou os seguintes dados:

Classificador	**Kernel**	**Accuracy**
SGD	$hinge$	$\approx 92,8571\%$
SGD	log	$\approx 80,9524\%$
SGD	$modified_huber$	$\approx 61,9048\%$
SGD	$squared_hinge$	$\approx 73,8095\%$
SGD	$perceptron$	$\approx 73,8095\%$
SGD	$epsilon\_insensitive - l2$	$\approx 95,2381\%$
SGD	$epsilon\_insensitive - l1$	$\approx 78,5714\%$
SGD	$epsilon\_insensitive - l2 - invscalling$	$\approx 88,0952\%$
SGD	$epsilon\_insensitive - l2 - constant$	$\approx 78,5714\%$
KNeighbors	$K = 3$	$\approx 83,3333\%$
SVC	$linear$	$\approx 00,0000\%$
DecisionTree		$\approx 90,4762\%$
RandomForest		$\approx 78,5714\%$

Tabela 2: Comparação de vários classificadores usando o ficheiro de teste NovasQuestoes.txt

O SGD ($epsilon\_insensitive$) apresentou o melhor resultado, porém após realizar mais testes, mudando o *training set* e o *testing set*, obtemos melhor resultados usando o SGD ($hinge$). Observámos também, que o nível de falha reside em categorias que sejam bastante próximas. Por exemplo, detetámos alguns erros em distinguir a categoria *budget* da categoria *revenue*.

Decidimos realizar mais testes para despistar eventuais padrões nos ficheiros de teste. Criou-se um *script* (CORPUS.SH) onde juntámos as perguntas e respostas de treino e de teste. O *script* separa, aleatoriamente, as frases da seguinte forma:

- $70\%(175)$ número de questões de treino
- $30\%(75)$ número de questões de teste

Com base nos resultados apresentados pelo *script* concluimos que o classificador com mais taxa de *accuracy* em casos gerais foi o SGD ($hinge$) ainda que o $epsilon\_insensitive$ apresente na **Tabela 2** melhores resultados.

Conclusão e trabalho futuro

Por forma a resolver erros em categorias próximas, uma possível solução seria criar uma lista de palavras mais comuns por categoria por forma a aperfeiçoar o critério. Conclui-se então, que não existe nenhum classificador pré-determinado, isto é, o classificador deve ser obtido à custa de testes com casos reais.

Referências

[1] Text Classification using Algorithms - chatbot

[2] dk_, Soul of the Machine: How Chatbots Work

[3] PythonProgramming.net, Text Classification with NLTK

[4] nltk.org Probability in NLTK

[5] Syed Sadat Nazrul, Multinomial Naive Bayes Classifier for Text Analysis

[6] Olli Huang, Applying Multinomial Naive Bayes to NLP Problems: A Practical Explanation

[7] scikit-learn.org, 1.9. Naive Bayes

[8] scikit-learn.org, Working With Text Data