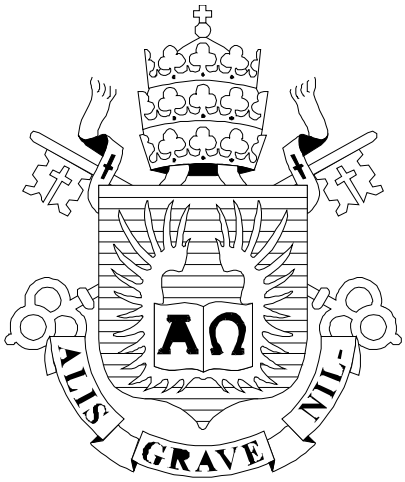


PUC

INF2390 – Tópicos de Banco de Dados IV

Aplicação do algorítimo de Naive bayes para classificação automática de artigos do portal de jornalismo G1

Demetrius Costa Rapello

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900

RIO DE JANEIRO - BRASIL

Aplicação do algorítimo de Naive bayes para classificação automática de artigos do portal de jornalismo G1

Demetrius Costa Rapello
demetrius.rapello@gmail.com

Sumário

1 Introdução

Classificação automática de texto é definida pela atividade de assinalar uma nome de classes predefinido a um novo documento através do aprendizado prévio realizado pelo classificador que é baseado na análise de usos das palavras no corpus de exemplo [1]. Vários métodos de classificação tem sido estudados entre eles: Decision Tree, Naive Bayes, Neural Networks e Support Vector Machines(SVM). Entre estes métodos, Naive Bayes apresenta o modelo mais simples por que dispõem de um classificador probabilístico baseado no teorema de Bayes que utiliza um modelo de features independentes para aferição das estatística.

Basicamente o modelo funciona assim, imaginado que queiramos descobrir nomes que são masculinos e femininos, treinamos o classificador do Naive Bayes com um corpus de nomes próprios e passamos a feature de avaliar a última letra do nome próprio. Com essas informações o classificador avalia as probabilidades dos nomes terminarem com uma determinada letra como na tabela abaixo:

Tabela extraída do nltk cookbook on-line [7].

Última letra	Feminino	Masculino
A	38.3	1
K	1.0	31.4
F	1.0	15.3
P	1.0	10.6
W	1.0	10.6

Como pode ser visto na tabela acima existe 38.3 vezes mais chance de um nome terminado com a letra A ser Feminino e não Masculino.

Apesar da aparente simplicidade, o classificador do Naive Bayes tem sido usado em muitas pesquisas para classificação de texto onde destaco a pesquisa de [3] sobre a classificação de artigos de noticia entre um conjunto de categorias.

2 Estado da Arte

Este trabalho é baseado no trabalho de [3] onde é apresentado um modelo para classificação automática de artigos usando o método de Naive Bayes. Neste trabalho os autores descrevem as principais etapas do processo de classificação de texto apresentando uma proposta de algoritmo para classificação de artigos em categorias.

2 1 Pré processamento

A tarefa de pré processamento dos documentos consiste na diminuição do ruído presente no corpus de treinamento. Para isso são empregadas tarefas de padronização de palavras(case folding), exclusão de stopwords, exclusão de repetição de caracteres, traduções, exclusão de caracteres especiais como markups html e outros marcadores de texto como pontuações e quebra de texto (Tokenization). No trabalho de [3] foram usadas as atividades de: case folding, exclusão de stopwords, parsing ou tokenization.

O pré processamento é importante pois tende a reduzir o corpus de trabalho garantindo uma melhor performance na execução do treinamento do classificador, além de maximizar a classificação dos documentos.

2 1 1 Case Folding

Case folding é a atividade de unificar as palavras dos documentos colocando-as em um mesmo padrão. Assim se num dado documento existir a palavra AMOR escrita em diversas formas como: Amor, amor e AMOR, elas representarão uma única palavra para o classificador.

2 1 2 Parsing

Nesta fase o corpus de treino é quebrado em palavras e armazenado em um Array, este processo também é conhecido como **Bag of Words**.

Existem diversas formas de **Tokenizar** um texto, a mais comum é quebrar o texto em palavras, usando como regra a presença de um espaço em branco entre elas. Há também formas mais elaboradas descritas no NLTK-COOKBOOK [6] são elas: PunktWordTokenizer, WordPunctTokenizer, RegexpTokenizer, PunktSentenceTokenizer e TreebankWordTokenizer.

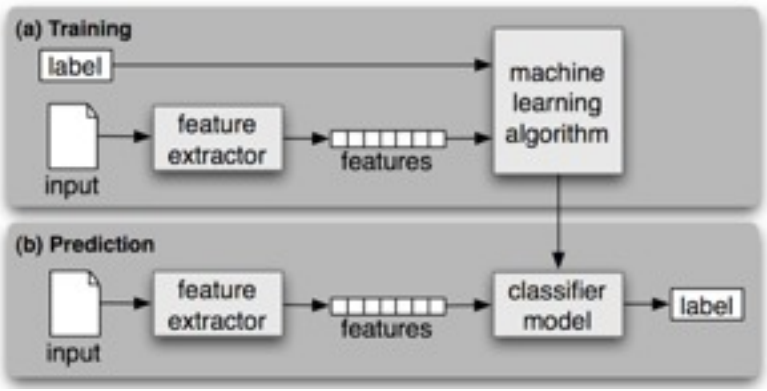
2 1 3 Stopwords Elimination

Stopwords são palavras comuns que não contribuem para o significado do texto. Geralmente são tratados como stopwords os conectores de sentenças, preposições, verbos etc. Nesta fase o conjunto de stopword é filtrado do corpus de treino.

2 2 Classificação automática de Texto Supervisionada

Segundo [7] Classificação é a tarefa de atribuir um determinado rótulo a um dado texto de entrada. Na classificação cada rótulo é considerado isoladamente uns dos outros e o conjunto dos rótulos disponíveis é conhecido a priori. São Exemplos de classificação: Decidir se um email é ou não um spam, Escolher uma determinada categoria para um novo artigo.

Na classificação supervisionada, um conjunto de dados é separada para servir de base de aprendizado pelo classificador. De acordo com o entendimento desta base, o classificador consegue inferir qual a categoria correta para uma nova instancia.



2 2 1 Feature Extraction

É a tarefa de tentar extrair do texto, padrões de informações que possibilitem ao classificador o entendimento do corpus. Em outras palavras é ajudar o classificador a entender o saco de palavras que entregamos para ele. Assim, quanto melhor é o conjunto de features extraídas, é esperado que informações mais relevantes sejam passadas para o classificador melhorando a sua capacidade de aprendizado e classificação [2].

Em uma forma mais simples, podemos definir que: Dado um documento D composto de um vetor de palavras P pode ser representado em features por:

Features = Array[p=True] para todo p pertencente a P

Desta forma estamos dizendo que cada palavra existem no documento é representada por uma feature que atribui True para a presença da palavra no documento.

A seleção de melhores critérios para formação de features tendem a facilitar e potencializar o trabalho dos classificadores. Portanto faz-se necessário um estudo de padrões do corpus para a identificação das melhores features.

2 2 2 Naive Bayes Classifier

O Naive Bayes é uma método de classificação que usa um classificador probabilístico baseado no teorema de Bayes. Baseado nas features recolhida na etapa de extração de features, ele tenta maximizar a probabilidade do documento conter ou não um determinado rótulo.

Exemplificando, cada documento do corpus de treino é representado como um saco de palavras (f1,f2...fn) onde f1 é a primeira palavra do documento, f2 a segunda e assim sucessivamente. Seja C o conjunto de categorias de classificação existente. Durante a classificação, o abordagem de Bayes tentará encontrar o a categoria com a maior probabilidade para as palavras em (f1,f2...fn) seguindo a formula.

$$\operatorname{classify}(f_1, \ldots, f_n) = \operatorname*{argmax}_c \, p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c).$$

O Naive Bayes tem sido usado em várias pesquisas [1,2,3,4] e tem apresentado resultados satisfatórios na tarefa de classificação de texto.

3 Implementação

O objetivo do trabalho é estudar a viabilidade de uma alternativa automática para auxiliar os jornalistas na classificação das matérias entre as categorias existentes. O sistema de publicação de matéria já está desenvolvido e será sobre ele que o sistema especialista deverá atuar. Para entender como o sistema especialista vai atuar é importante explicar o funcionamento do sistema de publicação de Matérias do G1.

3 1 Sistema de Publicação de Matérias

A produção de matérias do G1 conta com cerca de 70 jornalistas divididos em aproximadamente 27 categorias com uma produção diária de 1200 novas matérias sendo 400 escritas e 800 importadas por agências externas.

O sistema foi concebido sob plataforma web de modo que pode ser acessível pela intranet corporativa.

Os editores tem acesso uma interface de publicação de matéria que apresenta um formulário com os campos da estrutura de uma matéria (Título, Subtítulo e corpo) e após editada a matéria o Editor seleciona a categoria a qual a matéria pertence em um campo de seleção.

Uma vez publicada a matéria, ela fica disponível na página da categoria a qual foi associada.

Ex: uma matéria da categoria Mundo:

http://g1.globo.com/mundo/noticia/2011/02/obama-pede-que-presidente-do-egito-escute-o-apelo-por-mudancas.html

3 2 Sistema especialista

O sistema especialista deve ser capaz de sugerir, após a edição da matéria, a categoria correta evitando assim a seleção manual, para isso, será necessário que o sistema especialista já esteja com o seu classificador treinado para que possa em tempo realizar a operação de classificação:

O tempo de classificação não poderá ultrapassar 2 segundos pois isso prejudicaria a produção geral de matérias do portal G1

3 3 Verificação

Para treinar o classificador foram separadas matérias de acordo com a distribuição por categoria. Conforme exposto na tabela abaixo:

Categoria	5%	Total
Auto Esporte	36	1827
Bom Dia Brasil	80	4028
Brasil	389	19496
Ciência e Saúde	46	2346
Cinema	33	1656
Concursos e Emprego	79	3954
Crime e Justiça	168	8449
Economia	115	5756
Economia e Negócios	670	33514
Eleições 2010	76	3847
G1	771	38557
Jornal da Globo	34	1709
Jornal Hoje	44	2243
Jornal Nacional	102	5140
Minas Gerais	55	2789
Mundo	1234	61743
Música	39	1985
Planeta Bizarro	42	2125
Política	179	8956
Pop & Arte	203	10165
Rio de Janeiro	201	10067
São Paulo	215	10781
Tecnologia e Games	150	7508
Vestibular e Educação	51	2585
	5012	251226

Tabela - Total de artigos por categoria

Para o aprendizado, 75% do corpus será utilizado. O restante, 25%, será utilizado para fazer o teste e validação da eficácia do algoritmo.

Para avaliação do algoritmo, serão usadas as métricas: Accuracy, Precision e Recall.

Accuracy é o percentual de acertos que o classificador consegue contra o dataset de testes.

Para melhor entender as métricas Precision e Recall precisamos observar a tabela abaixo e explicar o conceito de Falso Positivo, Falso Negativo, Verdadeiro Positivo e Verdadeiro Negativo:

	Brasil	Economia
Categoria Correta	Sim	Não
Julgamento do Classificador	Sim	Não
	Não	Sim

Falso positivo segundo [6] é quando o classificador classifica um rótulo contrário ao seu real valor, no Exemplo acima quando o classificador diz sim para categoria é Economia e não é a categoria Economia.

Falso negativo segundo [6] é quando o classificador classifica um rótulo contrário ao seu real valor, no Exemplo acima quando o classificador diz não para categoria é BRASIL e é a categoria BRASIL.

Verdadeiro positivo segundo [6] é quando o classificador classifica um rótulo contrário ao seu real valor, no Exemplo acima quando o classificador diz sim para categoria Brasil e está certo.

Verdadeiro negativo segundo [6] é quando o classificador classifica um rótulo contrário ao seu real valor, no Exemplo acima, quando o classificador diz não para categoria Economia e está certo.

Com isso podemos entender Precision e Recall onde:

Precision é a ausência de falsos positivos e Recall é a ausência de falsos negativos. Desta forma elas são métricas inversas, quanto maior a precisão menor e o recall e vice-versa.

4 Resultados

Já foram feitos alguns experimentos com o corpus que será utilizado. A implementação inicial com o Naive Bayes chegou nos seguintes resultados:

1 - Porções balanceadas de documentos por categorias.

Utilizando a abordagem de cada palavra presente no documento é uma feature os resultados foram:

```
def document_features_single(self, document):
    features = {}
    for word in document:
        features[word] = True
    return features
```

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavr
2	28	0.872340425532	0.94019138756	0.846982758621	2005	549611
8	146	0.636520538488	0.884083044983	0.548283261803	3862	1063654
15	327	0.378733572282	0.875	0.221857923497	4464	1288493
27	707	0.30993669144	0.878378378378	0.137711864407	4844	1394865

Utilizando a abordagem de remover as stopwords e tratar cada palavra unicamente no documento.

```
def document_features_single_set_sem_stop(self, document):
    features = {}
    for word in set(document):
        if word not in self.stopwords:
            features[word] = True
    return features
```

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavr
2	26	0.873670212766	0.936067551267	0.849945235487	2005	549611
8	134	0.649637556093	0.878130217028	0.5613660619	3862	1063654
15	306	0.438470728793	0.864864864865	0.303157894737	4464	1288493
27	670	0.264244426094	0.837209302326	0.114649681529	4844	1394865

Utilizando a abordagem de usar somente as melhores palavras que qualificam o documento.

```
categorias=set()
total_word_count=0
word_scores = {}

for d in corpus:
   words = set(d[0])
   categoria = d[1]
   for word in words:
      if word not in stopwords:
         word_fd.inc(word.lower())
         label_word_fd[categoria].inc(word.lower())
         total_word_count += 1
   categorias.add(categoria)

for word, freq in word_fd.iteritems():
   word_scores[word]=0
   for c in categorias:
      word_scores[word] += BigramAssocMeasures.chi_sq(label_word_fd[c][word],(freq,
label_word_fd[c].N()), total_word_count)

best = sorted(word_scores.iteritems(), key=lambda (w,s): s, reverse=True)[:10000]
bestwords = set([w for w, s in best])

def best_word_feats(self, document):
   features = dict([(word, True) for word in set(document) if word in self.bestwords])
   return features
```

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavra
2	19	0.797207446809	0.899871630295	0.755387931034	2005	549611
8	69	0.589575422851	0.763458401305	0.515418502203	3662	1063654
15	124	0.333034647551	0.76404494382	0.222707423581	4464	1288493
27	206	0.31544178365	0.768181818182	0.179215270414	4844	1394865

2 – Mudando a estratégia para passar uma porção igual de documentos por categoria os resultados melhoram para os mesmos 3 algorítmos apresentados.

Utilizando a abordagem de cada palavra presente no documento é uma feature os resultados foram:

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavra
2	8	0.936666666667	0.992592592593	0.881578947368	400	106205
8	68	0.873333333333	0.990740740741	0.718120805369	1600	443164
15	234	0.817142857143	0.945945945946	0.686274509804	2800	850329
27	651	0.824761904762	0.943396226415	0.662251655629	4200	1263630

Utilizando a abordagem de remover as stopwords e tratar cada palavra unicamente no documento.

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavra
2	7	0.95	0.992753623188	0.907284768212	400	106205
8	65	0.885	0.974137931034	0.743421052632	1600	443164
15	218	0.841428571429	0.940170940171	0.733333333333	2800	850329
27	602	0.833650793651	0.923076923077	0.734693877551	4200	1263630

Utilizando a abordagem de usar somente as melhores palavras que qualificam o documento.

Categorias	Tempo	Accuracy	Precision	Recall	Total de documentos	Total de palavra
2	6	0.89	1.0	0.791139240506	400	106205
8	33	0.83	1.0	0.656050955414	1600	443164
15	90	0.713333333333	0.936708860759	0.46835443038	2800	850329
27	185	0.711746031746	0.885714285714	0.446043165468	4200	1263630

5 Conclusão

O dataset de treinamento é responsável direto pela corretude do classificador, nos estudos aplicados, percebemos que quando porções de documentos eram colocados de acordo com o percentual de 2% do total da categoria, ou seja, gerando uma distribuição balanceada, o classificador apresentava um queda de aproximadamente 65% entre os testes com 2 categorias (87%) e 27 categorias (30%). Confrontando com o dataset formado por porções iguais de documentos por categoria, configurando um distribuição equalitária, os resultados são bem melhores. A queda é de aproximadamente 11% entre os testes com 2 categorias (93%) e 27 categorias (82%).

Entre os algoritmos para extração de features, os desempenho ruim do algoritmo que descobre as palavras mais relevantes para o texto pois em ambos os testes este algoritmo piorou o resultado de corretude, precisão e recall a medida em que se aumentava o corpus de treinamento com novas categorias.

A primeira hipótese para trabalhos futuros é estudar outras alternativas para extração de features que melhorem o desempenho da classificação dos artigos em nas categorias, talvez o uso de dicionários específicos com palavras contextualizadas as categorias, para que durante a seleção as features tenham um peso maior.

A segunda hipótese é trabalhar com mais de um classificador binário como visto em [1,2] e confrontar os resultados para descobrir aquele que maximiza a classificação.

Referências Bibliográficas

[1] Bofeng Zhang, Xin Xu, Jinshu Su - 2007. **An Ensemble Method for Multi-class and Multi-label Text Categorization:** National University of Defense Technology, China;

[2] Min-Ling Zhang, Jose M. Penã and Victor Robles - 2009. **Feature Selection for Multi-Label Naive Bayes Classification**: College of Computer and Information Engineering, Hohai University, Nanjing 210098, China;

[3] Arni Darliani Asy'arie, Adi Wahyu Pribadi - 2009. **Automatic News Articles Classification in Indonesian Language by Using Naive Bayes Classifier Method**: Universitas Islam Negeri Syarif Hidayatullah Jakarta, Universitas Pancasila Jl. Srengseng Sawah, Jagakarsa Jakarta.

[4] Ioannis Katakis, Grigorios Tsoumakas, Ioannis Vlahavas - 2008. **Multilabel Text Classification for Automated Tag Suggestion**: Department of Informatics, Aristotle University of Thessaloniki, Greece;

[5] Tom M. Mitchell. **Machine Learning: Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression**.

[6] Jacob Perkins. **Python Text Processing with NLTK 2.0 Cookbook**.

[7] NLTK COOKBOOK . **Learning to Classify Text**
http://nltk.googlecode.com/svn/trunk/doc/book/ch06.html