UNIVERSIDADE ESTADUAL DE MARINGÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE INFORMÁTICA

MÉTODOS DE CLASSIFICAÇÃO E CLUSTERIZAÇÃO

ACADÊMICOS: RA: 65261 - ADRIANO KATAYAMA GROFF
RA: 62367 - EVANDRO MATHIAS FRIEDRICHSEN
RA: 60022 - GUSTAVO BENNEMANN DE MOURA
RA: 65048 - MATHEUS URATAKI ALVES DA SILVA

MARINGÁ, 17 de Junho de 2013.

1 INTRODUÇÃO

Na área de classificação de dados, um dos dominós pesquisados é a classificação de textos, ela é amplamente utilizada pelas ferramentas de busca para gerar um bom resultado a cada termo pesquisado. Uma ferramenta capaz de predizer e classificar ou clusterizar grupos de forma correta é de suma importância, pois essa é uma das coisas que faz um site de busca se diferenciar de outros. Porém essa tarefa não é tão simples.

Neste trabalho serão apresentados alguns métodos de aprendizagem de maquina para classificação e clusterização. Cada método apresenta características diferentes e serão apresentadas nas sessões abaixo. Foram utilizada duas técnicas de classificação de dados: Naive Bayes e Support Vector Machine (SVM). E duas para clusterização de dados: K-means e Hierarchical Clustering.

2 NAIVE BAYES

O classificador Naive Bayes é um dos classificadores mais utilizados em aprendizagem de maquina. É denominada ingênua (Naive) por assumir que os atributos são condicionalmente independentes, a informação de um evento não é informativa sobre nenhum outro.

É aplicado a tarefas de aprendizagem onde: cada instância x é descrita por uma conjunção de valores de atributos; função alvo f(x) pode assumir qualquer valor de um conjunto V; um conjunto de exemplos de treinamento da função alvo é fornecido; uma nova instância é descrita pela tupla de valores de atributos <a1,a2,...,an>. A tarefa é predizer o valor alvo ou classe, para esta nova instância.

Ao treinar o classificador calculamos uma distribuição geradora Pr(d|c) para cada classe c pertencente a {-1.1}. Na fase de classificação, calculamos qual distribuição tem a maior probabilidade de ter gerado cada documento.

Quando usar esse classificador: quando estiver disponível um conjunto de treinamento grande ou moderado; os atributos que descrevem as instâncias forem condicionalmente independente dada a classe.

Aplicações bem sucedidas: diagnóstico médico; classificação de documentos textuais.

No modelo multinominal, cada documento é representado por um vetor de atributos caracterizando o número de vezes que cada atributo ocorre no documento.

Pode ser de forma que cada gerador controle uma “roleta” com W faixa. Ao gerar um documento, o gerador primeiro escolhe um comprimento L para um documento e depois gira a roleta L vezes para definir quais palavras colocará no texto.

3 SVM

As SVM são baseadas na teoria de aprendizado estatístico, sendo um método de aprendizagem supervisionado. Diferentemente de métodos de clusterização, a ideia envolvida é de separar linearmente ambas as classes, de forma a elementos pertencentes a classes diferentes permanecerem em lados diferentes.

A SVM, a partir dos dados de treinamento, define um hiperplano n-dimensional, de forma que dos dados de treinamento, representados na forma de pontos nesse ambiente n-dimensional, sejam divididos de forma a maximizar a distância entre os pontos e o hiperplano. Dessa forma, esse método consegue classificar apenas duas classes distintas (FRADKIN et al, 2006).

Elas podem obter fronteiras lineares, as quais são chamadas de SVMs lineares, e fronteiras não lineares que são extensões das SVM lineares. Onde as SVMs lineares podem ser de margens rígidas ou suaves (LORENA; CARVALHO, 2007).

As SMVs usam como base a ideia de que se os dados forem mapeados em um espaço de dimensão suficientemente grande eles serão linearmente separáveis. Para não correr risco de superadaptação dos dados as SVM encontram sempre o separador linear ótimo.

Encontrar o separador é um problema de otimização de programação quadrática (Russel), assim tem-se que encontrar os valores dos parâmetros αi que maximizem a equação:

$$\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

Onde $\alpha_i \geq 0$ e $\Sigma_i \alpha_i y_i = 0$

Os dados entram na expressão apenas em forma de produtos pontuais de pares de pontos, no caso da equação correspondente ao próprio separador, e uma vez que os valores ótimos de αi tenham sido calculados a expressão é dada por:

$$h(x) = sign\left(\sum_i \alpha_i y_i (x \cdot x_i) \right)$$

Uma ultima propriedade importante do separador ótimo definido por essa equação é que os pesos αi associados a cada ponto de dado são zero, com exceção dos pontos mais próximos ao separador, que são também chamados de vetores de suporte.

É possível encontrar separadores lineares no espaço de características de elevado numero de dimensões F(x) apenas substituindo na equação (1) por uma função de núcleo.

Ao mapear novamente os separadores lineares resultantes ao espaço de entrada original, é possível obter limites não lineares entre os exemplos positivos e negativos (Russel).

4 K-MEANS

O algoritmo K-means de clusters de dados tem a finalidade de separar as amostras em N grupos de igual variância, minimizando um critério conhecido como “inércia” dos grupos. Este algoritmo requer que o número de cluster seja especificado.

Ele se adapta bem ao grande número de amostras e tem sido usado em uma ampla gama de áreas de aplicação. Também é equivalente ao algoritmo de expectatica-maximização quando é definida a matriz de covariância para ser diagonal, igual ou pequena. O K-Means escolhe centroides que minimizam o conjunto da soma dos quadrados das funções obejtivos utilizando um conjunto de dados X com N amostras.

$$J(X, C) = \sum_{i=0}^{n} \min_{\mu_j \in C}(||x_j - \mu_i||^2)$$

O algoritmo K-Means é basicamente composto por três etapas, a primeira etapa o algoritmo escolhe os centroides iniciais utilizando o método mais básico sendo escolhido K amostras do conjunto de dados X; após a inicialização o algoritmo consiste de um looping entre os dois passos principais, o primeiro passo atribui cada amostra para seu centroide mais próximo, a segunda etapa cria novos centroides tomando o valor médio de todas as amostras, que foram atribuída a cada centroide anterior. A diferença entre os centroides anteriores e os novos centroides representa a inercia, o algoritmo então continua a repetir estes dois passos até que o valor desta diferença seja inferior a um limiar, ou seja, o ciclo de etapas se repete até que os centroides não se movam significativamente.

O algoritmo de K-Means sempre converge, porém ele pode convergir para um mínimo local, isto depende muito da escolha dos centroides iniciais. Para solucionar este problema o calculo é realizado diversas vezes com diferentes inicializações de centroides.

5 HIERARCHICAL CLUSTERING

Hierarchical Clustering é uma família de algoritmos de clusterização que geram conjuntos aninhados fundindo-os sucessivamente. A hierarquia de clusters é representada por uma arvore. A raiz da arvore é composta por um conjunto *unique* que contém todas as amostras, e as folhas contem os conjuntos com apenas uma amostra.

O algoritmo Ward realiza um Hierarchical Clustering fazendo a minimização das variâncias, ou seja, a cada passo o algoritmo de Ward minimiza a soma das diferenças de quadrados em todos os clusters.

6 ARQUITETURA

6.1 Pre-processamento

Para o pre-processamento foi utilizada as seguintes funções da biblioteca do NLTK.

- wordpunct_tokenize: Função responsável por dividir as strings em listas de substrings. wordpunct_tokenize pode ser usado, por exemplo, palavras em uma string ou para encontrar listas de frases. Deve - se tomar atenção quando em utilizar os caracteres Unicode.

- Stopwords: São palavras que podem ser consideradas irrelevantes para o conjunto.
- Lancaster Stemmer: Interface usada para remover afixos morfológicos das palavras, deixando apenas a palavra primitiva. Algoritmos Stemming, visam remover também os afixos em função gramatical, morfologia derivativa. Porém existe um problema devido as palavras irregulares, regras morfológicas complicadas e palavras com sentido ambíguo.

O pre-processamento pega todos os textos, tokeniza (usando o wordpunct_tokenize) e filtra os tokens com tamanho menor que 4 e maior que 20, se forem stopwords, após isso é feita o stemming.

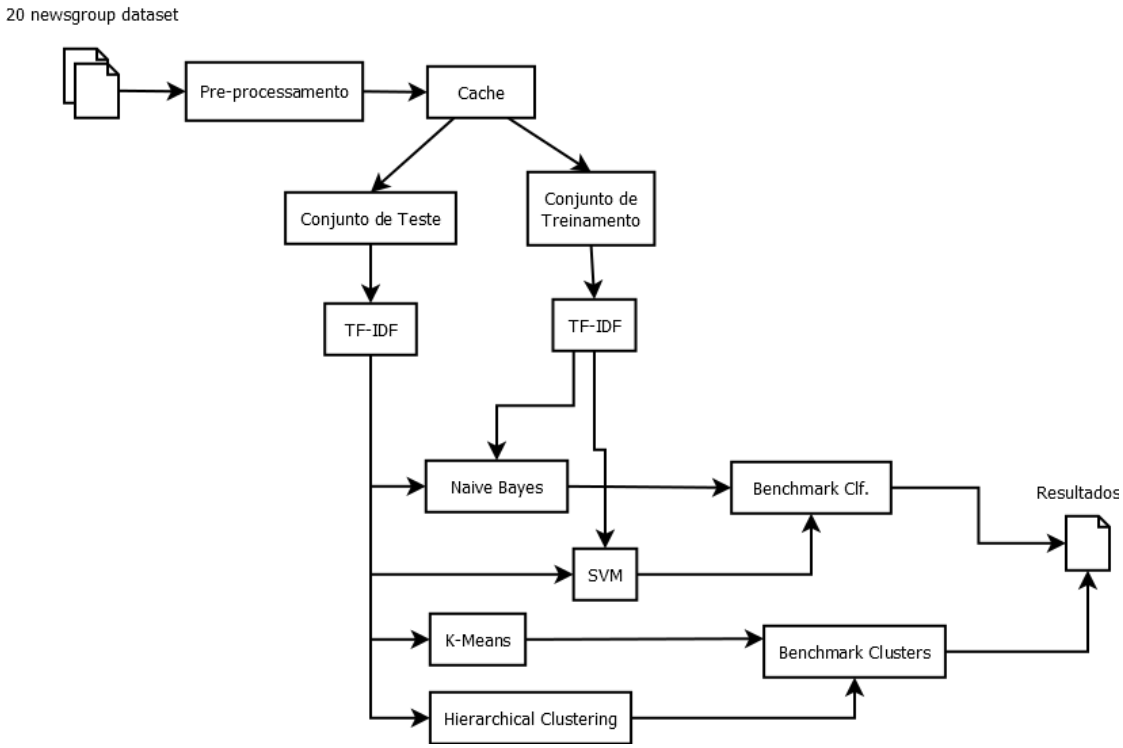

Figura 1 – Pipeline de execução

O conjunto de teste e treinamento é separado no pré-processamento (sabe-se qual é qual porque o de teste vem do mini-newsgroup).

O TF-IDF é executado pelo *CustomTfidfVectorizer* que é uma abstração da classe *TfidfVectorizer* do scikit-learn, que já implementa geração do bag-of-words e retornando o TF-IDF.

7 PROBLEMAS ENCONTRADOS

O principal problema ocorreu devido a grande quantidade de dados disponíveis, ocasionando em erros de memória. Esse problema foi resolvido parcialmente aplicando uma melhor filtragem no pré-processamento. Mesmo diminuindo a quantidade de tokens gerados (dimensão da matriz) o algoritmo de Warp (Hierarchical Clustering) não funcionava devido a sua entrada só poder ser uma *array*. Como a matriz gerada é muito grande, não foi possível converte-la para *array*, sendo assim, adotamos o conjunto melhor para a avaliação dos resultados dos algoritmos de clusterização.

Também não foi possível gerar os gráficos dos algoritmos de clusterização, pois dimensão dos dados é muito maior que 2 ou 3. Era possível utilizar uma decomposição (PCA) fazendo com que a dimensão fosse diminuída para gerar os gráficos, porém isso afetava drasticamente os resultados dos algoritmos.

8 RESULTADOS

Nesta sessão serão descritos os resultados obtidos pelo nosso programa para cada algoritmo proposto. Para os algoritmos de classificação, o conjunto de treinamento utilizado possui 17997 arquivos e o de teste 2000 arquivos. Já para o algoritmo de classificação foi utilizado o conjunto de teste, sendo explicado anteriormente o motivo.

8.1 Naive Bayes

Segue abaixo a tabela contendo a precisão e recall, f1-score da execução do Naive Bayes.

	precision	recall	f1-score	support
alt.atheism	0.70	0.81	0.75	100
comp.graphics	0.72	0.83	0.77	100
comp.os.ms-windows.misc	0.83	0.75	0.79	100
comp.sys.ibm.pc.hardware	0.72	0.78	0.75	100
comp.sys.mac.hardware	0.83	0.82	0.82	100
comp.windows.x	0.90	0.84	0.87	100
misc.forsale	0.94	0.74	0.83	100
rec.autos	0.91	0.87	0.89	100
rec.motorcycles	0.99	0.92	0.95	100
rec.sport.baseball	0.99	0.93	0.96	100
rec.sport.hockey	0.96	0.96	0.96	100
sci.crypt	0.83	0.95	0.88	100
sci.electronics	0.90	0.81	0.85	100
sci.med	1.00	0.94	0.97	100
sci.space	0.91	0.96	0.94	100
soc.religion.christian	0.80	0.97	0.87	100
talk.politics.guns	0.72	0.84	0.77	100
talk.politics.mideast	0.87	0.98	0.92	100

Abaixo é a imagem da matriz de confusão gerada, sendo 0 alt.atheism e 20 talk.politics.mideast

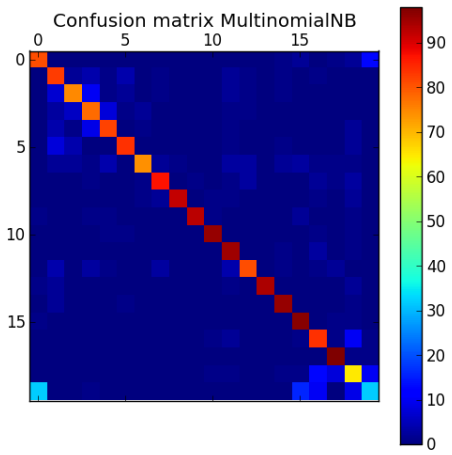

Figura 2 – Matriz de confusão (Naive Bayes) - Colorido

8.2 SVM

Segue abaixo a tabela contendo a precisão e recall, f1-score da execução do SVM.

	precision	recall	f1-score	support
alt.atheism	0.72	0.81	0.76	100
comp.graphics	0.79	0.84	0.82	100
comp.os.ms-windows.misc	0.84	0.90	0.87	100
comp.sys.ibm.pc.hardware	0.76	0.75	0.75	100
comp.sys.mac.hardware	0.86	0.83	0.84	100
comp.windows.x	0.88	0.84	0.86	100
misc.forsale	0.86	0.86	0.86	100
rec.autos	0.91	0.89	0.90	100
rec.motorcycles	0.93	0.96	0.95	100
rec.sport.baseball	0.96	0.95	0.95	100
rec.sport.hockey	0.99	0.95	0.97	100
sci.crypt	0.97	0.94	0.95	100
sci.electronics	0.84	0.92	0.88	100
sci.med	0.95	0.95	0.95	100
sci.space	0.99	0.96	0.97	100
soc.religion.christian	0.92	0.93	0.93	100
talk.politics.guns	0.83	0.88	0.85	100
talk.politics.mideast	0.90	0.96	0.93	100
talk.politics.misc	0.70	0.62	0.66	100
talk.religion.misc	0.54	0.44	0.49	100
avg / total	0.86	0.86	0.86	2000

Abaixo é a imagem da matriz de confusão gerada, sendo 0 alt.atheism e 20 talk.politics.mideast

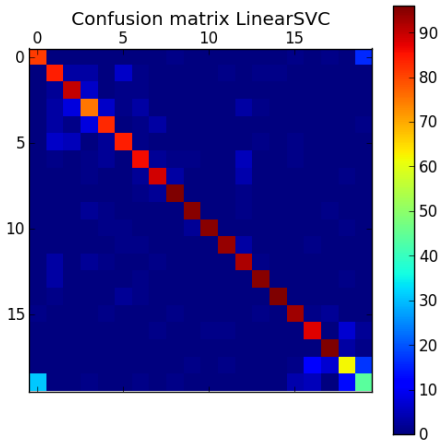

Figura 3 – Matriz de confusão (SVM) - Colorido

8.3 K-means

SilhouetteCoefficient: 0.0036979587376316748

V-measure: 0.32765275810047834

Completeness: 0.3361247981008012

Homogeneity: 0.31959729453865438

8.4 Hierarchical Clustering

SilhouetteCoefficient: 0.0037259055901819773

V-measure: 0.42489425279750703

Completeness: 0.50392808371119124

Homogeneity: 0.36729011144737594

O índice de completude é a taxa de que todos os membros de uma dada classe são atribuídos ao mesmo cluster, já a homogeneidade é a taxa de que cada cluster possui membros de uma única classe. O coeficiente de silhueta é uma medida que combina a coesão e separação, já o V-measure é medida que mede a concordância entre duas entre as classes reais e as classes encontradas, ignorando as permutações possíveis.

9 CÓDIGO

Devido a numeração, segue em anexado ao trabalho impresso o código do programa.

10 INSTRUÇÕES DE USO

Para utilizar o programa basta ter todas as bibliotecas instaladas (NLTK, scipy, numpy, scikit-learn) e executar o arquivo *main.py* digitando no terminal *python main.py*.

11 REFERÊNCIAS

Lorena, A. C.; Carvalho, A. C. P. L. F.. Uma Introdução às Support Vector Machines. Revista de Informática Teórica e Aplicada, vol.14, no2, p.43-67, 2007.

Disponível em <http://seer.ufrgs.br/rita/article/download/rita_v14_n2_p43-67/3543>. Acesso em 09 de Jun. de 2013

FRADKIN, D.; MUCHNIK, I.. Support Vector Machines for Classication, 2006.

Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.76.5272&rep=rep1&type=pdf>. Acesso em 09 de Jun. de 2013

http://www.mestreseo.com.br/keyword/stop-words-como-funcionam-palavras-de-parada

http://nltk.org/api/nltk.stem.html

http://www.eletrica.ufpr.br/ufpr2/professor/36/TE808/5-NaiveBayes-AM.pdf

http://scikit-learn.org/stable/modules/clustering.html