3. Aprendizado de Máquina

O processo de pesquisa em aprendizado de máquina consiste em examinar e experimentar as estratégias mais eficazes para a construção de programas que aprendem a partir da experiência adquirindo conhecimento de forma automática. Um programa aprende um conjunto de tarefas *T* com uma medida de desempenho *D* a partir de uma experiência *E*, se seu desempenho de aprendizado *D* aumenta com a experiência *E*, ou seja, se é capaz de tomar decisões baseado em experiências acumuladas por meio da solução bem sucedida de problemas anteriores (Mitchell, 1997).

Existem diversas aplicações práticas deste processo em diversas áreas e um exemplo é a classificação de textos. A tarefa de classificação de textos se favorece em muito do aprendizado de máquina, pois esta técnica permite que o aprendizado ocorra apenas com a disponibilização de exemplos. Através da análise feita nestes exemplos o algoritmo consegue perceber o que diferencia um texto de outro e cria os parâmetros necessários que possibilitarão a classificação.

Então partimos de uma situação em que não é conhecida nenhuma relação à priori entre os dados de entrada e a saída desejada e, apenas através de exemplos, conseguimos criar esta relação.

O aprendizado de Máquina faz parte de uma área muito mais ampla chamada de Inteligência Artificial (Mitchell, 1997). Esta área procura estudar e compreender o fenômeno da inteligência e paralelamente desenvolve instrumentos para apoiar a inteligência humana. Outra área que apoia em muito o desenvolvimento de pesquisas em aprendizado de máquina é a Filosofia. Dela emprestamos metodologias básicas de desenvolvimento de conhecimento para aplicação em algoritmos de aprendizado de máquina.

A Filosofia é a primeira das ciências e foi ela que originou todas as outras existentes hoje. As metodologias desenvolvidas são muitas e dentre elas

podemos destacar o método de Aristóteles que consistia nas formas indutivas e dedutivas de se raciocinar. Basicamente o raciocínio dedutivo consiste em argumentar do geral para o particular, por exemplo: Todos os gatos miam, Mimi é um gato, logo Mimi mia. O indutivo por outro lado consiste em argumentar do particular para o geral, por exemplo: Mimi mia, Mimi é um gato, logo todos os gatos miam.

A dificuldade com o método dedutivo consiste na falta de premissas universalmente verdadeiras, pondo em cheque a eficácia do método de Aristóteles para descobrir a verdade. O raciocínio indutivo é mais característico do mundo moderno, pois está associado com a metodologia científica. Alguns opositores a este método argumentam que nunca se pode ter certeza de que se chegou a qualquer verdade através do método indutivo a não ser que se tenha observação completa ou universal, o que é impossível.

3.1.Tipos de Aprendizado

Utilizando o método de raciocínio indutivo existem vários tipos de aprendizado dentre eles: Aprendizado Supervisionado, Semi-Supervisionado e Não supervisionado. Para cada uma destas técnicas foram desenvolvidos diversos algoritmos. Para o aprendizado Supervisionado podemos encontrar: Transformation-Based Learning (TBL) e Naive Bayes. Para o Não supervisionado Hidden Markov Model (HMM). Alguns exemplos de algoritmos que utilizam o aprendizado semi-supervisionado são: EM (Expectation-Maximization) Modelo Bayesiano de Misturas Multinomiais e HMM. Muitos destes algoritmos possuem versões em mais de um tipo de aprendizado.

O aprendizado supervisionado utiliza pares associados de textos com seus atributos e a sua classificação, como exemplos para a criação de um modelo. No aprendizado não supervisionado, os exemplos não possuem valor de classificação associado e por isso são agrupados pelo algoritmo em clusters ou classes.

No aprendizado semi-supervisionado temos como objetivo aprender a classificação de textos utilizando um número limitado de exemplos etiquetados e acrescentar documentos não etiquetados. O processo de etiquetagem é

extremamente mais “caro” do que simplesmente a coleta de documentos não etiquetados.

Mas podemos nos perguntar como documentos não etiquetados (amostras) podem aumentar a precisão de uma classificação. Em (Nigam, Mccallum, Theun e Mitchell, 2000) muito é discutido sobre esta possibilidade. Este artigo diz que, a princípio, podemos pensar que não é possível obter nenhum ganho com isso. Mas as amostras efetivamente nos fornecem informações valiosas sobre a distribuição da probabilidade conjunta de n-gramas. Suponha, por exemplo, que utilizando apenas os exemplos podemos determinar que os documentos que possuam a palavra “é” são pertencentes à classe positiva. Se nós utilizarmos este fato para estimar a classificação de várias amostras podemos chegar à conclusão que a palavra “definição” ocorre freqüentemente nas amostras que agora se acredita serem da classe positiva. Esta coexistência das palavras “definição” e “é” numa quantidade muito grande de amostras pode nos fornecer informação muito útil na construção de um classificador mais preciso.

3.2.Algorítimo EM Bayesiano de Misturas Multinomiais

Neste trabalho utilizamos um algoritmo EM para treinar um classificador a partir de textos etiquetados e não etiquetados configurando um aprendizado do tipo semi-supervisionado, recurso extremamente útil quando temos acesso a poucos exemplos para o domínio do problema. Este algoritmo EM foi combinado conforme sugerido em (Nigam, Mccallum, Thrun e Mitchell, 2000) com o classificador naive-Bayes, um modelo de misturas de multinomiais que é bastante utilizado na tarefa de classificação de textos.

A seguir estaremos detalhando os conceitos envolvidos na definição deste Algoritmo EM Bayesiano de Misturas Multinomiais especificando cada uma de suas características.

3.2.1.Modelo Bayesiano

O modelo Bayesiano, assim como qualquer classificador probabilístico define um modelo probabilístico gerador para os dados e assume duas

premissas: (1) os dados são produzidos por um modelo de misturas e (2) existe uma correspondência de um para um entre as componentes desta mistura e as classes.

Desta forma todo o documento d_i é gerado de acordo com uma distribuição probabilística definida por um grupo de parâmetros denominado Θ. Esta distribuição consiste de uma mistura de componentes $c_j \in C = \{c_1,...,c_{|C|}\}$. Cada componente é parametrizada por um subconjunto de Θ. Desta forma um documento d_i é criado em dois passos: (1) selecionar uma componente da mistura de acordo com a sua probabilidade a priori $P(c_j \mid \Theta)$ e (2) gerar o documento através desta componente selecionada, de acordo com os seus parâmetros pela distribuição $P(d_i \mid c_j;\Theta)$. Então podemos caracterizar a probabilidade de um documento como a soma da probabilidade de todas as componentes da mistura.

$$P(d_i \mid \Theta) = \sum_{j=1}^{|C|} P(c_j \mid \Theta) P(d_i \mid c_j;\Theta) \quad (1)$$

3.2.2.Classificador naive-Bayes

Naive Bayes considera um modelo gerador probabilístico para textos. Este modelo é uma especialização do modelo de misturas descrito anteriormente e por isso assume as duas premissas já apresentadas. O termo ingênuo (naive) se referencia exatamente ao fato deste modelo adicionalmente assumir que existe uma independência entre os termos, ou seja, a distribuição conjunta dos termos é igual ao produtório da distribuição de cada um deles.

Um texto pode ser descrito pelo seu tamanho $|d_i|$ e por uma lista ordenada de termos $X = \{x_{i1}, x_{i2}...\}$. Então a componente selecionada gera uma seqüência de termos de tamanho especificado. Podemos então expressar a probabilidade de um documento dado a componente da mistura e Θ como sendo:

$$P(d_i \mid c_j;\Theta) = P(|d_i|)\prod_n P(x_{in} \mid c_j;\Theta;X)$$
$$P(d_i \mid c_j;\Theta) \propto \prod_n P(x_{in} \mid c_j;\Theta) \quad (2)$$

E chegamos a (2) utilizando a premissa naive-Bayes que os termos de um texto são gerados de forma independente do contexto, ou seja, independente dos outros termos existentes no mesmo texto. Isto quer dizer que assumimos que a probabilidade de um termo é independente da posição que ela aparece no documento. Assumimos também que para todas as classes o tamanho do documento é distribuído igualmente simplificando ainda mais o cálculo.

Assim, os parâmetros que definem cada uma das componentes associadas às classes é uma distribuição sobre os termos. Estas podem ser definidas utilizando várias distribuições tais como: a binomial ou a multinomial.

No modelo binário, assumimos que cada documento é representado por um vetor de atributos binários de modo que cada atributo indica a ocorrência ou não de um evento no documento. No modelo multinomial, assumimos que cada documento é representado por um vetor de atributos inteiros caracterizando o número de vezes que cada evento ocorre no documento. No nosso caso estaremos utilizando o modelo multinomial.

3.2.3. Distribuição Multinomial

A distribuição multinomial pode ser empregada na determinação da probabilidade quando, no evento especificado, se deseja calcular a probabilidade de um acontecimento composto, estabelecido por vários eventos. Neste caso, os eventos que constituem o acontecimento devem ser independentes e a ordem dos eventos, dentro do acontecimento, não influencia o cálculo da probabilidade.

Dado um grupo de variáveis $X_1, X_2, ..., X_n$ que possuem uma função probabilística:

$$P(X_1 = x_1, ..., X_n = x_n) = \frac{N!}{\prod_{i=1}^{n} x_i!} \prod_{i=1}^{n} \phi_i^{x_i}$$

onde x_i são inteiros positivos tais que $\sum_{i=1}^{n} x_i = N$, e ϕ_i são constantes com $\phi_i > 0$ e $\sum_{i=1}^{n} \phi_i = 1$.

Então a distribuição conjunta de $X_1, X_2, ..., X_n$ é uma distribuição multinomial e $P(X_1 = x_1, ..., X_n = x_n)$ é dado pelo coeficiente correspondente da série multinomial $(\phi_1 + \phi_2 + ... + \phi_n)^N$.

Em outras palavras, se $X_1, X_2, ..., X_n$ são eventos mutuamente exclusivos com $P(X_1 = x_1) = \phi_1, ..., P(X_n = x_n) = \phi_n$. Então a probabilidade de X_1 ocorrer x_1 vezes, ..., X_n ocorrer x_n vezes é dada por (Papoulis, 1984):

$$P_N(x_1, x_2, ..., x_n) = \frac{N!}{x_1! ... x_n!} \phi_1^{x_1} ... \phi_n^{x_n} . \quad (3)$$

3.2.4.Modelo Bayesiano de Misturas Multinomiais

O modelo Bayesiano de Misturas Multinomiais então considera quatro premissas relativas à geração de documentos: (1) existe um modelo de misturas, (2) existe uma correspondência de um para um entre as componentes da mistura e as classes, (3) a ocorrência dos termos de um texto são eventos independentes e (4) o tamanho do documento e distribuído igualmente entre as classes. Todas estas premissas são violadas em textos reais: documentos podem comumente ser associados a mais de uma classe, os termos em um texto não são independentes, mas apesar disto empiricamente o classificador apresenta consistentemente bons resultados (McCallum e Nigam, 1998).

À medida que podemos estimar os parâmetros de Θ, obtendo θ, a partir dos documentos de treinamento, exemplos, é possível inverter o modelo gerador e calcular a probabilidade de um componente da mistura ter gerado um dado documento. Retiramos isto do teorema de Bayes e depois por substituição utilizando as eq. (1) e eq. (2) obtemos eq. (4).

$$P(c_j \mid d_i; \theta) = \frac{P(c_j \mid \theta) P(d_i \mid c_j; \theta)}{P(d_i \mid \theta)}$$

$$P(c_j \mid d_i; \theta) = \frac{P(c_j \mid \theta) \prod_n P(x_{in} \mid c_j; \theta)}{\sum_j P(c_j \mid \theta) \prod_n P(x_{in} \mid c_j; \theta)} \quad (4)$$

O documento será considerado pertencente à classe que tiver o maior valor calculado: $\arg\max_j P(c_j \mid d_i;\theta)$

Utilizando uma notação mais simples seguimos para as fórmulas que serão utilizadas no algoritmo.

$$\pi_j = P(c_j \mid \theta) \qquad \sum_j \pi_j = 1$$
$$\phi_{jp} = P(x_p \mid c_j;\theta) \qquad \sum_p \phi_{jp} = 1$$
$$\pi_{ij} = P(c_j \mid d_i)$$

Para um conjunto de textos etiquetados, o nosso conjunto de treinamento *D*, teremos $\pi_{ij} \in \{0,1\}$ e podemos afirmar a partir da eq. (1) que:

$$P(D \mid \theta) = \pi_1 P_1(D \mid \phi_{11},...,\phi_{1p}) + ... + \pi_j P_j(D \mid \phi_{j1},...,\phi_{jp}) \qquad (5)$$

E finalmente utilizando as distribuições multinomiais temos:

$$P_j(D_i \mid \theta) = \frac{N!}{n_1!...n_p!}\phi_{j1}^{n_1}...\phi_{jp}^{n_p} \qquad (6)$$

Quando estimamos θ queremos encontrar o Θ que maximize $P(\Theta \mid D)$. Utilizando o teorema de Bayes podemos quebrar a expressão $P(\Theta \mid D)$ em $P(D \mid \Theta)P(\Theta)$ e derivá-la em Θ. Este processo descrito em (Nigam, Mccallum, Thrun e Mitchell, 2000) nos levará às formulas que serão utilizadas no algoritmo EM (Expectation-Maximization) descrito a seguir para a estimativa de $\theta = (\pi_1, \pi_2,..., \pi_j, \phi_{11},..., \phi_{1p},..., \phi_{j1},..., \phi_{jp})$.

3.2.5.O Algoritmo EM

O algorítmo EM Expectation Maximization (Dempster, Laird e Rubin, 1977) é um processo iterativo eficiente para calcular a máxima verossimilhança estimada, quando existem dados faltando ou escondidos. No nosso caso as amostras são consideradas incompletas, pois a sua classificação não é utilizada para o aprendizado. No cálculo da máxima verossimilhança nós queremos estimar os parâmetros do modelo para os quais os dados observados são os

mais prováveis. A convergência é assegurada dado que o algoritmo garante aumentar a verossimilhança a cada iteração.

Cada iteração do algoritmo executa dois processos: o Expectation e o Maximization. No passo Expectation, ou E-step, os dados que estão faltando são estimados de acordo com os dados observados que geraram o modelo e seus parâmetros iniciais. No passo Maximization ou M-step, a verossimilhança é maximizada assumindo que os dados escondidos são conhecidos. A estimativa calculada para os dados faltantes no E-step são utilizadas para este fim.

Desta forma o E-step pode ser interpretado como sendo um passo construtor de um mínimo local para a distribuição a posteriori enquanto que o M-step aperfeiçoa este valor, melhorando a estimativa dos dados faltantes. Uma segunda referência sobre o Algoritmo EM é (McLachlan e Krishnan,1997).

O algoritmo EM foi combinado com o modelo Bayesiano de Misturas Multinomiais conforme sugerido em (Nigam, Maccallum, Thrun e Mitchell, 2000). Este processo permite que possamos utilizar documentos não etiquetados para a melhora do aprendizado diminuindo o trabalho manual na etiquetagem de exemplos. Estaremos assim transformando o nosso processo em um aprendizado semi-supervisionado. Note que a aproximação teórica utilizada depende das premissas já descritas: (1) os dados são produzidos por um modelo de misturas e (2) existe uma correspondência de um para um entre as componentes da mistura e as classes. Quando estas premissas não são satisfeitas, e na maioria dos casos reais isso acontece, aprimoramentos se fazem necessários como mostrado em (Nigam, Maccallum, Thrun e Mitchell, 2000). Em vários experimentos a adição de amostras, textos não etiquetados, piorou o desempenho do classificador.

A seguir descreveremos mais detalhadamente os passos do algoritmo EM e como ele calcula os parâmetros do modelo de misturas multinomiais:

3.2.6.Pseudocódigo

Faremos aqui uma breve descrição do processo como um todo incluindo cada um de seus passos. Aqui já utilizaremos a formalização próxima ao utilizado no algoritmo que foi implementado.

- Entrada: Conjunto de textos exemplo e textos amostra
- Criação de um modelo de mistruras multinomial inicial com a estimativa de Θ a partir apenas dos exemplos.
- Repetir até que não ocorra uma variação da melhora dos parâmetros maior que um fator especificado
    - E-Step – Utilizando o classificador atual Θ, estimar a probabilidade de cada documento amostra ter sido gerado por cada componente da mistura.

    - M-Step – Recriação do classificador Θ, utilizando os exemplos e amostras estimadas.
- Saída: Um classificador, que recebe um texto sem classificação e define a classe mais provável.

Para o E-Step temos o seguinte pseudo-código:

Loop i = 1 até a quantidade total de sentenças
Loop j = 1 até a quantidade total de classes

$$\pi_{ij} = \frac{\pi_j f_j(X_i \mid \phi_{j1},...,\phi_{jp})}{\sum_l \pi_l f_l(X_i \mid \phi_{l1},...,\phi_{lp})} \quad (7)$$

sendo que, para o cálculo de $f_j(X_i \mid \phi_{j1},...,\phi_{jp})$ fizemos algumas simplificações e utilizando a eq. (6) na eq. (7) obtemos:

$$\pi_{ij} = \frac{\pi_j \phi_{j1}^{n_{i1}} ... \phi_{jp}^{n_{ip}}}{\sum_l \pi_l \phi_{l1}^{n_{i1}} ... \phi_{lp}^{n_{ip}}} \quad (8)$$

E para o M-Step:

Loop j = 0 até a quantidade total de classes

$$\pi_j = \frac{\sum_i \pi_{ij}}{i} \quad (9)$$

Loop p = 0 até a quantidade total de palavras

$$\phi_{jp} = \frac{\sum_i \pi_{ij} n_{ip}}{\sum_l \sum_i \pi_{ij} n_{il}} \quad (10)$$

3.2.7.Suavização de Laplace

Com o propósito de evitar que o cálculo da probabilidade seja igual a zero pelo fato de uma das palavras não ocorrer em nenhum exemplo da categoria, podemos utilizar a suavização de Laplace que é muito utilizada em trabalhos encontrados na literatura.

Somamos então 1 ao numerador e o número de classes é somado no denominador no primeiro caso ficando assim:

Loop j = 0 até a quantidade total de classes

$$\pi_j = \frac{1 + \sum_i \pi_{ij}}{i + j} \quad (11)$$

Para o cálculo de ϕ somamos 1 no numerador e no denominador somamos o valor do tamanho do léxico obtendo:

Loop p = 0 até a quantidade total de palavras

$$\phi_{jp} = \frac{1 + \sum_i \pi_{ij} n_{ip}}{\sum_l \sum_i \pi_{ij} n_{il} + p} \quad (12)$$

3.2.8.Métricas *precision-recall* e F1

As métricas de $precision - recall$ são as mais utilizadas numa classificação binária. A tarefa de uma classificação binária muitas vezes se assemelha mais a uma filtragem do que a uma clusterização, pois encontramos poucos casos positivos em uma enorme quantidade de casos negativos. Imaginando a tarefa de obtenção de LOs podemos claramente observar esta situação. Em textos longos estamos interessados em separar apenas as partes que poderão ser utilizadas como objetos de aprendizagem. *Precision* e *recall* avaliam

corretamente esta tarefa e dada uma categoria c_i, $precision$ e $recall$ associados a esta categoria são definidos como:

Recall = Número de predições positivas corretas
Número de exemplos positivos

$$recall_i = \frac{TP_i}{N_i} \tag{13}$$

Precision = Número de predições positivas corretas
Número de predições positivas

$$precision_i = \frac{TP_i}{TP_i + FP_i} \tag{14}$$

Quando trabalhamos com diversas classes estamos calculando as mesmas métricas para cada classe separadamente e podemos sumarizar estes valores obtendo métricas globais da seguinte forma:

$$Macrorecall = \frac{\sum_i^{|C|} recall_i}{|C|} \qquad Macroprecision = \frac{\sum_i^{|C|} precision_i}{|C|} \tag{15}$$

Estas duas medidas globais apresentam normalmente resultados bem diferentes devido à distribuição de exemplos entre as classes. Caso esta distribuição seja muito disforme, o que ocorrerá com muita freqüência, as medidas por classe serão importantes assim como as medidas globais.

Para um classificador ser considerado bom não é suficiente que ele tenha uma destas medidas alta isoladamente. Por isso também calculamos uma terceira medida que faz uma avaliação conjunta delas da seguinte forma:

$$F_\beta = \frac{(\beta^2 + 1)\, precision \times recall}{\beta^2\, precision + recall} \tag{16}$$

Utilizamos β = 1 dando a mesma importância para as duas medidas e desta forma calculando a métrica chamada de F_1.

Adicionalmente em alguns experimentos utilizamos também a métrica Accuracy. Ela sozinha não é uma boa métrica de desempenho, pois poderíamos alcançar valores altos sempre classificando os textos na classe negativa, mas para os experimentos relativos ao aprendizado semi-supervisionado nos foi bastante útil. Esta métrica é na verdade a *precision* considerando todas as classes.

$$Accurrancy = \frac{TP}{TotaldeAmostras} \tag{17}$$

Finalizados os capítulos conceituais que apresentaram as bases teóricas do presente trabalho podemos seguir para a apresentação do processo que foi desenvolvido utilizando esta base. Apresentamos a seguir o sistema de mineração de LOs que foi desenvolvido para concretizar este processo.