Trabalho Final - Aprendizado de Máquina e Mineração de Dados

Diorge Brognara
Gabriel Silva Trevisan
Thiago Miranda
Wilton Vicente Gonçalves da Cruz
DC -Departamento de Computação
UFSCar - Universidade Federal de São Carlos

Abstract

Este texto foi desenvolvido para o trabalho final da disciplina de Aprendizado de Máquina e Mineração de Dados, ministrada pelo professor Dr. Estevam Rafael Hruschka Junior no segundo semestre de 2016. Esse trabalho tem o objetivo consolidar os conceitos vistos durante a disciplina, assim como verificar a aplicação dos algoritmos vistos em aula para as tarefas de aprendizado supervisionado e não supervisionado, levantando-se observações sobre os resultados obtidos. Os algoritmos contemplados por esse trabalho foram os seguintes: Árvore de Decisão (ID3 ou C4.5), *Naive Bayes*, Regressão (linear ou logística), *k-Means* e *Expectation Maximization*(para a tarefa de agrupamento).

1 Introdução

Nessa seção, será feita uma breve descrição da disciplina de Aprendizado de Máquina, apresentando-se uma definição do problema estudado por ela. A partir dessa definição, será mostrado como diferentes tipos de problema levam ao uso de diferentes abordagens.

1.1 Definição do Aprendizado de Máquina

Na literatura, é possível encontrar diferentes definições para a disciplina de Aprendizado de Máquina. Uma definição bastante comum para a tarefa de Aprendizado de máquina pode ser encontrada em Mitchell[2006]:

> [...] [A] machine learns with respect to a particular task T, performance metric P, and type of experience E, if the system reliably improves its performance P at task T, following experience E. [Mitchell, 2006]

1.2 Abordagens do Aprendizado de Máquina

Há diversas categorias de problemas que podem ser abordados por aprendizado de máquina. Exemplos incluem classificação de texto, detecção de objetos em imagens, sistemas de recomendação automática, detecção automática de fraudes em cartão de crédito, mecanismos de pesquisa, entre outros. Cada um desses problemas é possui características particulares, que devem ser consideradas em sua abordagem.

A variedade da natureza dos problemas também leva ao surgimento de diferentes métricas de desempenho, adequadas a cada tipo de problema.

Por exemplo, um aspecto muito importante a ser considerado em algoritmo de aprendizado de máquina é o *overfitting*, isto é, a dependência muito forte aos dados para a geração dos modelos ou das hipóteses. Isso faz com que, muitas vezes, o algoritmo não tenha uma boa generalização para dados além dos dados de treinamento. Busca-se evitar esse aspecto quando a tarefa de aprendizado é preditiva, pois o algoritmo não generalizaria bem para os dados que se deseja prever, porém, para tarefas descritivas, esse aspecto não é negativo, pois deseja-se de fato que o algoritmo gere o modelo mais adequado para os dados de treinamento.

Além das diferentes medidas de desempenho, deve-se também levar em consideração os dados de treinamento e de teste do algoritmo, já que eles serão usados para gerar as saídas desejadas.

Dessa forma, o conhecimento das características das tarefas a serem realizadas e dos diferentes algoritmos de aprendizado de máquina é de grande importância para determinar qual algoritmo terá melhores resultados na tarefa a ser realizada.

2 Fundamentos teóricos

Nessa seção, serão discutidos alguns aspectos teóricos dos algoritmos que serão examinados nesse trabalho. Os algoritmos utilizados podem ser divididos em duas categorias: a de aprendizado supervisionado e de aprendizado não supervisionado. Dentre os algoritmos utilizados, árvore de decisão, *Naive Bayes* e regressão são algoritmos de aprendizado supervisionado e os algoritmos de agrupamento (k-Means e *Expectation Maximization*) são de aprendizado não supervisionado.

Alguns dos aspectos que serão analisados nesse trabalho para cada algoritmo são séries temporais, o viés indutivo do algoritmo e sua sensibilidade a hiperparâmetros.

O processo de aprendizado de máquina geralmente envolve a extração de informações mais gerais acerca de um evento com base em um conjunto de instâncias conhecidas desse evento. Em lógica, o processo de se obter conhecimentos gerais a partir de exemplos específicos é denominado indução. Como, em geral, um conjunto de instâncias não é suficiente para generalizar todos os casos possíveis, deve-se haver um conjunto de pressuposições acerca do problema que

permitam a generalização. Em aprendizado de máquina, esse conjunto de pressuposições é chamado de viés indutivo do algoritmo.

Há vários tipos de viés indutivo para os algoritmos de aprendizado de máquina. Exemplos são o viés de busca, ou de preferência e o viés de linguagem, ou restrição. O viés indutivo de cada algoritmo apresentado será discutido nessa seção.

Na terminologia de aprendizado de máquina, hiperparâmetros são características do algoritmo de aprendizado de máquina, não inferidos do conjunto de dados de treinamento, que influenciam diretamente no desempenho do algoritmo de aprendizado de máquina. Quanto mais sensível a hiperparâmetros for um algoritmo, mais seu desempenho será afetado pelos valores desses hiperparâmetros.

É comum que se faça uma otimização desses hiperparâmetros para cada problema que se deseja resolver, já que a configuração ótima geralmente se altera de um problema para o outro.

2.1 Algoritmos supervisionados

Em algoritmos de dados supervisionado, é dado um conjunto de dados de treinamento rotulados para o algoritmo. As variáveis não rótulo são denominadas atributos e a variável rótulo é comumente chamada de variável alvo, ou variável resposta. O objetivo dos algoritmos de aprendizado supervisionado é determinar a relação existente entre os atributos e a variável alvo, de forma a predizer o valor da variável alvo para um dado não rotulado. A função que relaciona os atributos e a variável alvo é denominada função hipótese.

Os elementos do conjunto de dados de treinamento são também chamados de instâncias e são comumente representados por vetores de atributos. É comum denominar-se a tarefa de aprendizado supervisionado de classificação, quando a variável alvo é discreta, e de regressão, quando a variável alvo é contínua.

Árvore de Decisão

O algoritmo de árvore de decisão é um algoritmo simbólico de classificação, em que a variável alvo geralmente possui duas classes. Nesse algoritmo, cada valor de atributo ou da variável alvo é visto como uma expressão lógica, de forma que cada instância é vista como uma conjunção de expressões lógicas.

A saída do algoritmo é uma árvore de decisão, que pode ser vista como um conjunto de regras que determinam associam um valor da saída para cada conjunção de entradas. Essa árvore é obtida a partir do conjunto de dados de treinamento, verificando-se as frequências dos valores dos atributos com relação aos valores da variável alvo.

Os algoritmos de árvore de decisão vistos durante a disciplina foram o ID3 e o C4.5. Ambos os algoritmos utilizam métricas com base na entropia para a determinação da ordem da colocação dos atributos na árvore. No caso do ID3, utiliza-se o ganho de informação, enquanto no caso do C4.5, utiliza-se a razão de informação.

A entropia pode ser vista como uma medida da quantidade de informação sobre a classe que pode ser obtida a partir de um conjunto de dados. Seja S um conjunto de amostras, p^+ a proporção de exemplos positivos de um evento nessa amostra e p^- a proporção de eventos negativos dessa amostra. A entropia desse conjunto é definida por:

$$\text{Entropia}(S) = -(p^+ \log_2(p^+)) - (p^- \log_2(p^-)) \quad (1)$$

O valor da entropia varia entre 0 e 1 e é menor quanto menos igualmente distribuído for o conjunto de dados com relação à classe. O ganho de informação e a razão de informação são métricas que buscam indicar o atributo que fornece mais informação sobre a classe, ou seja, o atributo que, em média, quando se dividir a árvore em ramos, mais diminuirá a entropia do conjunto. A razão de informação leva também em conta atributos com muitos valores, dividindo-se o ganho pela informação intrínseca(*split information*) de cada atributo.

Seja S um conjunto de dados, A um atributo, $V(A)$ o conjunto de valores possíveis desse atributo e S_{Av} o subconjunto de S em que atributo A possui valor $v \in V(A)$. Então, o ganho de informação e a razão de informação do atributo A, com relação ao conjunto S, são definidos por:

$$\text{Ganho}(S, A) = \text{Entropia}(S) - \sum_{v \in V(A)} \frac{|S_{Av}|}{|S|} \text{Entropia}(S_{Av}) \quad (2)$$

$$\text{Razao}(S, A) = \frac{\text{Ganho}(S, A)}{\text{SplitInformation}(S, A)} \quad (3)$$

$$\text{SplitInformation}(S, A) = -\sum_{v \in V(A)} \frac{|S_{Av}|}{|S|} \log_2 \left(\frac{|S_{Av}|}{|S|} \right) \quad (4)$$

Tanto o algoritmo ID3 como o algoritmo C4.5 utilizam uma estratégia de subida de encosta para a determinação dos atributos dos nós, em que, uma vez determinado o atributo de um nó, o algoritmo continua para suas subárvores, sem retornar para esse nó.

O algoritmo C4.5 também possui um mecanismo de poda, que busca eliminar ramos da árvore que fornecem menos informações para a classificação das instâncias.

Há diversas maneiras de realizar a poda em árvores de decisão, uma delas consiste em separar um conjunto de dados de teste, representar a árvore de decisão como um conjunto de regras do tipo se então, em seguida, para cada antecedente de cada regra, comparar o desempenho da árvore no conjunto de teste com o desempenho dela, retirando-se esse antecedente. Caso o desempenho da árvore melhore, esse antecedente é retirado da regra. Isso diminui o tamanho dos ramos da árvore de decisão, tornando-a menos suscetível ao *overfitting*.

Tanto no algoritmo ID3 quanto no C4.5, verifica-se uma tendência de escolha de árvores mais curtas. Isso se à escolha dos atributos dos nós, que é feita com base nos nós que fornecem mais informação sobre a classe. Os algoritmos seguem a estratégia de subida de encosta, já que, após escolhido um nó, o algoritmo segue para suas subárvores, não retornando mais a esse nó.

Enquanto essa estratégia não necessariamente retorna a menor árvore possível, a árvore encontrada é geralmente uma

das menores possíveis. Dessa forma, diz-se que os algoritmos ID3 e C4.5 possuem um viés de busca, ou preferência, já que dão preferência a árvores mais curtas, ou seja, hipóteses mais simples.

A preferência por hipóteses mais curtas (comumente chamada na literatura de navalha de Occam) permite que o algoritmo generalize melhor para dados não vistos, já que hipóteses mais longas tendem a ser mais específicas para o conjunto de dados.

Um hiperparâmetro de importância para a árvore de decisão é a utilização ou não da poda e qual forma de poda é utilizada; outro hiperparâmetro importante quando lidando com atributos contínuos ou com um domínio muito grande é a discretização feita para esses atributos, já que ela determina a ramificação da árvore.

Naive Bayes

O algoritmo de *Naive Bayes* é um algoritmo probabilístico de classificação. Ele utiliza o teorema de Bayes e a pressuposição de independência dos atributos, dada a classe, para estimar a probabilidade de cada classe a partir do conjunto de treinamento, dos atributos do dado observado e de algum conhecimento a priori sobre a distribuição da classe.

O *Naive Bayes* é geralmente é considerado um algoritmo *lazy* por não ser necessária a construção de um modelo, ainda que um modelo probabilístico seja gerado. Em contrapartida, o algoritmo de árvore de decisão mencionado acima é considerado um algoritmo *eager*, pois gera um modelo, sendo a árvore o modelo gerado.

Dessa forma, o algoritmo de *Naive Bayes* deve ser generalizado toda vez que um dado de entrada é recebido. Isso faz com que o tempo de treinamento seja reduzido, porém gasta-se mais tempo na classificação.

Para se determinar a probabilidade da classe, dados os atributos observados (posteriori), sabendo-se, de antemão, a distribuição da classe (priori), utiliza-se o teorema de Bayes. Seja $h \in H$ uma hipótese possível em um espaço de hipóteses H, $D \in \mathcal{D}$ os atributos do dado observado, pertencentes ao espaço de atributos $\mathcal{D}$, então, a probabilidade da hipótese condicionada pelos atributos é dada por:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \tag{5}$$

Considerando-se a hipótese como sendo a classe $c_j \in C$ atribuída ao dado observado, pertencente ao conjunto de classes C, e D um vetor de atributos $D = (a_1, a_2, \cdots, a_n)$, tem-se:

$$P(c_j|a_1, a_2, \cdots, a_n) = \frac{P(a_1, a_2, \cdots, a_n|c_j)P(c_j)}{P(a_1, a_2, \cdots, a_n)} \tag{6}$$

Dessa forma, a hipótese de máximo a posteriori pode ser determinada obtendo-se o valor de c_j que maximize a expressão acima. Como o denominador é constante com relação a c_j, encontra-se:

$$c_{j_{MAP}} = \arg\max_{c_j \in C} P(a_1, a_2, \cdots, a_n|c_j)P(c_j) \tag{7}$$

O termo $P(a_1, a_2, \cdots, a_n|c_j)$ pode ser expandido da seguinte forma:

$$\begin{aligned} &P(a_1, a_2, \cdots, a_n|c_j) = \\ &= P(a_1|a_2, a_3, \cdots, a_n, c_j)P(a_2|a_3, \cdots, a_n, c_j) \cdots P(a_n|c_j) \end{aligned} \tag{8}$$

O número de operações necessárias para determinar a probabilidade acima ou a quantidade de memória necessária para armazenar essas probabilidades de antemão cresce muito rapidamente em relação à quantidade de variáveis, o que torna a determinação do máximo a posteriori inviável computacionalmente.

O *Naive Bayes* resolve esse problema assumindo independência condicional entre os atributos, dada a classe, ou seja, a probabilidade da conjunção de n atributos, dada a classe é igual ao produto das probabilidades de cada atributo, dada a classe. Dessa forma:

$$\begin{aligned} P(x_k|x_{k+1}, \cdots, x_n|c_j) &= \frac{P(x_k, \cdots, x_n, c_j)}{P(x_{k+1}, \cdots, x_n, c_j)} \\ = \frac{P(x_k, \cdots, x_n|c_j)P(c_j)}{P(x_{k+1}, \cdots, x_n|c_j)P(c_j)} &= \frac{P(x_k, \cdots, x_n|c_j)}{P(x_{k+1}, \cdots, x_n|c_j)} \\ &= P(x_k|c_j) \end{aligned} \tag{9}$$

Assim, segue que:

$$\begin{aligned} &P(a_1|a_2, a_3, \cdots, a_n, c_j)P(a_2|a_3, \cdots, a_n, c_j) \cdots P(a_n|c_j) \\ &= P(a_1|c_j)P(a_2|c_j) \cdots P(a_n|c_j) = \prod_{i=1}^{n} P(a_i, c_j) \end{aligned} \tag{10}$$

Dessa forma, o classificador *Naive Bayes*, $c_{j_{NB}}$ é dado por:

$$c_{j_{NB}} = \arg\max_{c_j \in C} \prod_{i=1}^{n} P(a_i|c_j)\, P(c_j) \tag{11}$$

$$P(a_i|c_j) = \frac{\text{frequencia}(a_i, c_j)}{\text{frequencia}(c_j)} \tag{12}$$

Apesar da pressuposição de independência do *Naive Bayes* (que pode ser considerada um viés indutivo do algoritmo) aparentar ser bastante restritiva, o algoritmo ainda apresenta bom desempenho em diversas classes de problemas. A independência condicional dada a classe permite a determinação mais rápida da posteriori.

Pela expressão do classificador Naive Bayes, observa-se que, se um atributo não foi observado para alguma classe, sua probabilidade será igual a zero. Isso torna a resposta do algoritmo bastante dependente dos dados, já que, caso um atributo não seja observado no conjunto, desconsidera-se todos os outros.

Para se evitar isso, é comum que se utilize algum tipo de suavização, por exemplo, adicionando-se uma observação em cada atributo de cada classe, o que equivale a somar $\frac{1}{|C|}$ no numerador e 1 no denominador na determinação das probabilidades condicionais.

Regressão
2.2 Algoritmos não Supervisionados
k-Means
Expectation Maximization
3 Experimentos e Resultados
3.1 Algoritmos Supervisionados
Árvore de Decisão
Naive Bayes
Regressão
3.2 Algoritmos não Supervisionados
k-Means
Expectation Maximization
4 Conclusões

References

[Mitchell, 2006] Tom M. Mitchell. *The Discipline of Machine Learning*. Carnegie Mellon University, School of Computer Science, Machine Learning Department, 2006.