

Inteligência Artificial

Lógica Fuzzy, Perceptron e Aprendizagem de Máquina

Prof. Flávio Vinícius Cruzeiro Martins

VALOR: 25 pontos

- A política de pontos descontados por dia de atraso será de $2^n - 1$ ponto(s), onde n representa o número de dias em atraso.

PARTE 1 - LÓGICA FUZZY (7 pontos)

Escolha uma aplicação da vida real que possa ser aplicado lógica Fuzzy (duas se tiver fazendo em dupla), modele o problema e o conjunto de regras. Testes sua modelagem para diferentes tipos de entrada e mostre também o gráfico (surface) das soluções possíveis. Faça uma pequena análise dos seus resultados, principalmente interpretando o gráfico dado pelo surface.

PARTE 2 – PERCEPTRON (7 pontos)

Deseja-se implementar as funções “E” e “OU” de 3 entradas, com emprego de um *Perceptron* de 2 saídas. Implemente o algoritmo de treinamento de “Regra Delta” para obter os pesos e limiares (*biases*) para o modelo em questão, usando a “*toolbox*” de RNA do MATLAB.

PARTE 2 – DETECÇÃO DE SPAM COM APRENDIZAGEM DE MÁQUINA (11 pontos)

Introdução

O objetivo deste trabalho é utilizar a ferramenta Weka para comparar alguns algoritmos de aprendizagem de máquina em um problema de detecção de spam. Os algoritmos a serem comparados são: árvores de decisão, bayes ingênuo e redes neurais. Cada grupo deve baixar os dados e instalar a ferramenta Weka conforme as instruções abaixo. Depois deve executar as Partes 1, 2 e 3 e responder as perguntas em um relatório que deve ser entregue em anexo. Como parte do trabalho deve ser elaborado um artigo científico relatando o estado da arte do problema, as propostas de solução e os resultados obtidos com os experimentos. O formato do artigo deverá seguir ao modelo postado no grupo. Serão disponibilizados modelo em Word (.doc) e em LaTeX (.tex), lembrando que para documentações entregues em LaTeX receberão pontos extras.

Dados

A base de dados que será utilizada nesse trabalho é conhecida como Spambase e foi produzida por um grupo de pesquisa do Hewlett-Packard Labs. Ela está disponível publicamente no UCI Machine Learning Repository e contém 4601 mensagens de e-mail (sendo que 1813 são spam e 2788 não são spam). Cada mensagem é representada por 57 atributos numéricos sendo que a maioria desses atributos representa a frequência de uma determinada palavra na mensagem. A base de dados pode ser baixada aqui, já no formato ARFF que pode ser lido pela ferramenta WEKA.

Weka

O pacote Weka é um conjunto de implementações de algoritmos de aprendizagem de máquina, desenvolvido na Universidade de Waikato na Nova Zelândia.

O Weka foi implementado na linguagem Java, que tem como principal característica ser portátil, desta forma pode rodar nas mais variadas plataformas. Além disso é um software de domínio público estando disponível em <http://www.cs.waikato.ac.nz/ml/weka/>.

O Weka lê arquivos de exemplos para aprendizagem no formato ARFF que consiste basicamente de duas partes. A primeira contém uma lista de todos os atributos, onde devemos definir o tipo do atributo ou, no caso de atributos discretos, os valores que ele pode ter. Quando utilizamos valores estes devem estar entre "{ }", separados por vírgulas. A segunda parte consiste dos exemplos propriamente ditos, um por linha, com os valores de cada atributo para cada exemplo separados por vírgulas. Caso o valor de um determinado atributo seja desconhecido para um exemplo, devemos representá-lo com o símbolo "?".

Usando o Weka

1. Baixe e instale o Weka, seguindo as instruções disponíveis aqui.
2. Execute o Weka. Você deve ver uma GUI com algumas opções. Escolha 'Explorer'.
3. A janela 'Weka Explorer' deve aparecer. No tab inicial você pode selecionar o arquivo de dados. Escolha 'Open file...' e selecione o arquivo ARFF que você deve ter baixado com os dados (spambase.arff). Nesse tab você ainda pode examinar algumas informações sobre os

dados(frequências de valores de atributos) e remover atributos, mas isso não é necessário para esse trabalho.

4. Vá para o tab 'Classify'. Este será o tab utilizado para rodar os algoritmos de aprendizagem e visualizar os resultados.
5. Abaixo de 'Classifier', selecione 'Choose'. Você deve ver uma lista de algoritmos disponíveis, separados em pastas. Os três que você usará serão 'bayes > NaiveBayes' (Bayes Ingênuo), 'trees > J48' (Árvore de Decisão) e 'functions > MultiLayerPerceptron' (Rede Neural Multi-camada).
6. Quando você tiver escolhido um algoritmo, pode também escolher opções clicando como botão esquerdo em cima da caixa de texto que fica ao lado do botão 'Choose'. Uma janela será aberta para que você possa escolher opções que são diferentes para cada algoritmo.
7. Neste trabalho faremos a avaliação separando a base em duas partes uma para treinamento (66%) e outra para teste (33%). Isso pode ser feito utilizando a opção 'Test Options > Percentage split'.
8. Clique 'Start' pra começar a execução do algoritmo. Os resultados aparecerão na janela à direita.

Se aparecer uma mensagem de "out of memory", reinicie o Weka a partir da linha de comando com mais memória virtual, usando o seguinte comando (a partir do diretório onde o arquivo weka.jar estiver instalado): `java -Xmx1000m -jar weka.jar`

Parte 1 - Árvores de Decisão

Nesta parte, o objetivo é aprender uma árvore de decisão. Para executar o algoritmo de árvore de decisão no Weka, abaixo de 'Classifier' clique em 'Choose' e escolha 'trees > J48'. Não esqueça de marcar o 'Test Options' como sendo 'Percentage Split' (66%).

1. Execute o algoritmo com as opções padrão. Qual é o tamanho da árvore obtida? Qual é a sua taxa de acerto?
2. Agora, clicando com o botão esquerdo na caixa de texto ao lado do botão 'Choose', abra a janela de opções. Modifique a opção 'reducedErrorPruning' para 'True'. Essa opção faz uma poda na árvore até reduzir o erro em uma amostra de validação. Execute o algoritmo e reporte o tamanho da árvore e a taxa de acerto obtida com essa opção. A mudança foi significativa? Por que isso aconteceu?

Parte 2 - Bayes Ingênuo

Nesta parte, o objetivo é aprender um modelo bayesiano simples. Para executar o algoritmo bayesiano simples no Weka, abaixo de 'Classifier' clique em 'Choose' e escolha 'bayes > Naive Bayes'. Não esqueça de marcar o 'Test Options' como sendo 'Percentage Split' (66%).

1. Execute o algoritmo com as opções padrão. Qual foi a taxa de acerto obtida?
2. Agora, clicando com o botão esquerdo na caixa de texto ao lado do botão 'Choose', abra a janela de opções. Modifique a opção 'UseSupervisedDiscretization' para 'true'. Nessa opção, ao invés de tratar cada atributo como uma variável real gaussiana, o algoritmo primeiro discretiza os atributos e depois utiliza contagens para estimar probabilidades. Execute o algoritmo e reporte a taxa de acerto obtida com essa opção. A mudança foi significativa? Por que isso aconteceu?

Parte 3 - Redes Neurais

Nesta parte, o objetivo é aprender uma rede neural com uma camada de entrada, uma camada interna e uma camada de saída. Para executar o algoritmo de redes neurais no Weka, abaixo de 'Classifier' clique em 'Choose' e escolha 'functions > MultiLayerPerceptron'. Não esqueça de marcar o 'Test Options' como sendo 'Percentage Split' (66%).

1. Clicando com o botão esquerdo na caixa de texto ao lado do botão 'Choose', abra a janela de opções. Coloque o valor de 'Reset' como sendo 'false' e de 'Training Time' como sendo 100. Treine 6 redes neurais diferentes considerando todas as combinações dos seguintes valores para as opções 'learning rate': 0.1 e 0.3, e 'hidden layers' (número de unidades na camada oculta): 5, 10 e 20.
 - a. Para cada execução, qual foi o tempo para treinar a rede e qual a sua taxa de acerto?
 - b. Qual rede teve a melhor taxa de acerto? Qual seria uma explicação plausível para isso?
2. Mude o parâmetro 'Training Time' para 300 e o 'hidden layers' para 10. Treine duas redes neurais usando como 'learning rate': 0.1 e 0.3.
 - a. Para cada execução, qual foi o tempo para treinar a rede e qual a sua taxa de acerto?
 - b. Compare os resultados com os da pergunta 1(a) com 10 unidades na camada interna. Aumentar o tempo de treinamento melhorou a taxa de acerto? Isso sempre irá acontecer?
 - c. Crie outra rede com 'Training Time'=300, 'hidden layers'=5 e 'learning rate'=0.3. Compare esse resultado com o resultado análogo em 1(a). A taxa de acerto foi melhor ou pior?
3. Faça seus próprios experimentos variando 'Training Time', 'hidden layers' e 'learning rate'. Varie um parâmetro de cada vez, mantendo os outros constantes. A variação na taxa de acerto ocorre conforme você esperava?

*** Parte 3: retirado do site da professora Bianca Zadrozny da Universidade Federal Fluminense ***