19/11/2020

Segundo Trabalho de Programação I

Prof. Flávio Miguel Varejão

I. Descrição do Problema

Classificação de dados é o problema mais comum na área de aprendizado de máquina. Esse problema consiste em prever a classe de um objeto dentre um conjunto pré-determinado de possíveis classes com base nas características deste objeto.

Formalmente, dado um conjunto de dados X com N objetos $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, sendo que cada ponto $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^t$ possui d características e uma classe C_i, pertencente ao conjunto de classes $\{C_1, \ldots, C_K\}$, deseja-se criar um classificador $H(\mathbf{x}_j)$ capaz de prever a classe C_j do objeto $\mathbf{x}_j$.

A idéia deste segundo trabalho é fazer variações sobre o estudo de classificadores realizado no primeiro trabalho. A primeira variação consiste em utilizar a técnica de validação cruzada (ver seção I.1) para avaliar os classificadores. A segunda variação consiste em realizar um pré-processamento na base de dados para padronizar as características dos objetos e evitar que uma característica tenha maior influência na decisão do que outra simplesmente porque seus valores são de ordem de magnitude superior. A técnica de padronização utilizada no trabalho será a z-score (ver seção I.2). A terceira variação consiste em tornar o classificador vizinho mais próximo menos suscetível a presença de ruídos. Para resolver esse problema se costuma escolher os k vizinhos mais próximos para decidir a classificação do objeto por votação (ver seção I.3).

As técnicas de classificação utilizadas no primeiro trabalho (centróide e vizinho mais próximo) também serão usadas no segundo. No tradicional algoritmo do vizinho mais próximo, dado o objeto $\mathbf{x}_j$ que se deseja classificar, o algoritmo busca no conjunto de treinamento o exemplo $\mathbf{x}_i$ mais semelhante a $\mathbf{x}_j$ e atribui a $\mathbf{x}_j$ a classe de $\mathbf{x}_i$. Neste algoritmo será usada a distância Euclideana $|| x_i - x_j ||$ como métrica de distância. Ela é calculada pela expressão:

$$|| x_i - x_j || = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{id} - x_{jd})^2}$$

No algoritmo do classificador por centróides usa-se todos os exemplos de uma classe do conjunto de treinamento para determinar o centróide desta classe. Determina-se o centróide de cada uma das classes. Dado o objeto $\mathbf{x}_j$ que se deseja classificar, o algoritmo busca no conjunto de centróides aquele mais próximo do exemplo $\mathbf{x}_j$ e atribui a ele a classe do centróide mais próximo. Também aqui será usada a distância Euclideana.

O centróide $\mu_j = [\mu_{j1}, \mu_{j2}, \ldots, \mu_{jd}]^t$ é o ponto representativo da classe C_j e é calculado como o centro de massa do grupo de exemplos da classe C_j:

$$\mu_j = \frac{1}{n_j} \sum_{\mathbf{x}_i \in C_j} \mathbf{x}_i$$

onde n_j é o total de objetos pertencentes a classe C_j no conjunto de treinamento.

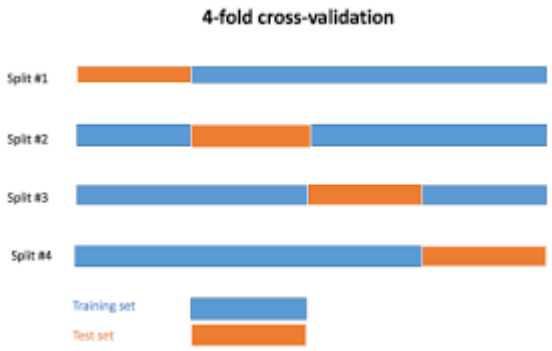
A base de dados será lida de um arquivo no formato csv (comma separated values) na qual cada linha representa um objeto da base. O nome do arquivo será lido da entrada padrão. Todas as características do objeto são representadas por números ponto flutuante. A classe do objeto é uma string e corresponde ao último dado da linha.

Os resultados da avaliação dos classificadores será apresentado em termos de acurácia e da matriz de confusão (ver https://pt.wikipedia.org/wiki/Matriz_de_confusão).

I.1. Validação Cruzada

A validação cruzada é um método de reamostragem e tem como objetivo avaliar a capacidade de generalização do seu modelo. Em outras palavras, verificar o quão pronto seu modelo está para receber novos dados. Na validação cruzada os dados de entrada são subdivididos em *nf* subconjuntos de dados (também chamados de folds). Você treina um modelo em todos os subconjuntos, exceto um (*nf-1*) e depois avalia o modelo no subconjunto que não foi usado para o treinamento. Esse processo é repetido *nf* vezes, com um subconjunto diferente reservado para avaliação (e excluído do treinamento) a cada vez.

O diagrama a seguir mostra um exemplo de subconjuntos de treinamento e subconjuntos de avaliação complementar gerados para cada um dos quatro modelos que são criados e treinados durante uma validação cruzada 4-fold. O modelo um usa os primeiros 25% dos dados para avaliação e os 75% restantes para treinamento. O modelo dois usa o segundo subconjunto de 25 por cento (25 a 50 por cento) para avaliação, e os três subconjuntos restantes de dados para treinamento e assim por diante.



A escolha dos objetos para compor os subconjuntos (folds) de dados será feita de maneira aleatória a partir de uma semente também lida da entrada padrão durante a execução do programa.

Note que cada exemplo da base só aparece em um único fold, isto é, não existe repetição de exemplo em folds, e todos exemplos devem ser alocados (não deve sobrar nenhum exemplo da base).

No caso da divisão do total de exemplos da base pelo número de folds não ser exata, haverá sobra de exemplos a qual deve ser distribuída exemplo a exemplo para os folds seguindo a sequência definida de folds. Por exemplo, suponha que desejamos dividir uma base com 10 exemplos em 4 folds. Suponha que os exemplos são gerados

randomicamente na seguinte sequência: [9, 3, 2, 4, 1, 5, 7, 10, 6, 8]. Os 4 folds resultantes serão [9, 3, 6], [2, 4, 8], [1, 5], [7, 10].

Note também que para cada fold de teste teremos um valor de acurácia e uma matriz de confusão. Em vez de reportar o valor do acurácia para cada fold, neste trabalho será reportada a média das acurácias dos folds e seu desvio padrão. Já com relação a matriz de confusão, será reportada uma matriz de confusão unificada com a soma das matrizes de confusão de cada fold.

A acurácia e o desvio padrão dos classificadores será apresentada na saída padrão sempre com duas casas decimais com arredondamento na segunda casa. A matriz de confusão será apresentada em um arquivo texto de saída cujo nome será lido da entrada padrão.

O desvio padrão é calculado pela seguinte fórmula:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Onde:

x_i é o valor observado.
μ é a média dos valores observados.
N é o total de valores observados utilizados para calcular a média.

I.2. Padronização

Padronizar significa calcular o número de desvios padrão pelos quais um determinado valor observado está acima ou abaixo do valor médio do que está sendo observado ou medido. Os valores padronizados acima da média são positivos, enquanto aqueles abaixo da média são negativos.

O cálculo é realizado subtraindo a média do valor individual e dividindo a diferença pelo desvio padrão. Esse processo de conversão de valores é chamado de **padronização**. Um valor padronizado é normalmente chamado de z-score. Ele é calculado da seguinte forma:

$$z = \frac{x - \mu}{\sigma}$$

Onde:

x é o valor a ser padronizado.
μ é a média dos valores padronizados.
σ é o desvio padrão dos valores padronizados.

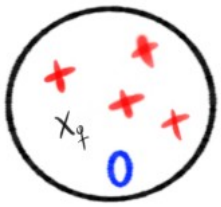
A padronização de base é uma importante transformação quando o método de classificação não compensa as diferenças de ordem de magnitude das diferentes características da base. Classificadores como centróide e vizinho mais próximo não fazem isso e, portanto, a padronização da base é indicada antes do uso desses classificadores.

Note que a padronização deve ser feita em todas as características da base e somente levar em conta os exemplos de treinamento. Fique atento que as médias e desvios padrão das características devem ser armazenados para fazer a mesma transformação nos exemplos de teste antes da consulta. Lembre ainda que para cada fold de teste será necessária uma padronização dos exemplos de treinamento.

I.3. K Vizinhos Mais Próximos

O algoritmo *k*-vizinhos mais próximos funciona da seguinte forma. Dado um objeto de teste *xq*, o algoritmo encontra os *k* vizinhos mais próximos (kNN) de *xq* no conjunto de treinamento. Em seguida, a classe de *xq* é dada pela classe que ocorrer com maior frequência entre os *k* vizinhos. A figura seguinte mostra um exemplo de aplicação

Na figura abaixo, são mostrados os cinco vizinhos mais próximos do objeto de teste *xq*. Desses cinco vizinhos, 4 são da classe "+" (vermelha) e 1 da classe "0" (azul). Ao aplicar o kNN, com k=5, o objeto *xq* é classificado como sendo da classe vermelha, pois essa classe possui mais representantes na vizinhança de *xq*.



Em caso de empate na votação, deve ser escolhida a classe cuja distância média dos vizinhos para o objeto for menor.

O valor de *k* será fornecido como entrada do programa.

II. Especificação do Sistema

Funcionalidades a serem implementadas:

*1.*Leitura do nome do arquivo de entrada, do nome do arquivo de saída, do número de folds, do número de vizinhos e da semente aleatória da entrada padrão.
*2.*Leitura da base de dados do arquivo csv de entrada.
*3.*Apresentação na saída padrão da acurácia média e o desvio padrão dos classificadores vizinho mais próximo, centróide e k vizinhos mais próximos.
*4.*Gravação da matriz de confusão dos classificador vizinho mais próximo, centróide e k vizinhos mais próximos no arquivo de saída.

Formato dos Dados do Entrada:

```
número de folds:              inteiro positivo
número de vizinhos:           inteiro positivo
semente:                      inteiro positivo
características dos objetos:  ponto flutuante
classe:                       string
```

Funções de Haskell para escolha randomizada de exemplos de teste:

```
import System.Random
mkStdGen semente
randomRs (0, ntotal-1) gerador

onde
      semente é um valor inteiro lido da entrada padrão
      ntotal é o número total de exemplos da base
      randomRIO gera um número inteiro aleatório no intervalo (1,
ntotal) e retorna como um IO(Int)
```

Os exemplos seguintes são apenas ilustrativos dos formatos de entrada e saída e não existe correspondência entre os seus dados.

Exemplo de formato de arquivo de entrada:

```
7,5.4,6.32,9,classe1
17,32.3,5,9.99,classe2
33,54,5.6,65.8,classe2
77.7,33.4,98,7.56,classe1
8.9,5.8,6,9,classe1
```

Exemplo de formato de arquivo de saida:

```
vizinho mais próximo:
72,28
15,35

centroides:
80,20
30,20

k-vizinhos mais próximos:
70,30
25,25
```

Exemplo de formato de interação do programa com o usuário:

```
Forneca o nome do arquivo de entrada: base.csv
Forneca o nome do arquivo de saida: confusao.txt
Forneca o número de folds: 10
Forneca o número de vizinhos: 5
Forneca o valor da semente para geracao randomizada: 42
Acuracia(vizinho): 71.33%
Desvio-Padrao(vizinho): 1.03%
Acuracia(centroide): 66.67%
Desvio-Padrao(centroide): 2.61%
Acuracia(k-vizinhos): 71.33%
Desvio-Padrao(k-vizinhos): 0.34%
```

III. Requisitos da implementação

- Modularize seu código adequadamente. Utilize os mecanismos de criação e importação de módulos de Haskell.

- Crie códigos claros e organizados. Utilize um estilo de programação consistente, dando preferência a criação de **tipos abstratos de dados**. Comente seu código.

- Os arquivos do programa devem ser lidos e gerados na mesma pasta onde se encontram os arquivos fonte do seu programa.

IV. Condições de Entrega

O trabalho deve ser feito individualmente e submetido até as 23:59 horas da data limite especificada na Atividade Segundo Trabalho Computacional na nossa sala de aula virtual. O trabalho deve ser submetido em um arquivo zip com o nome PG_1_TRABALHO_2_NomedoAluno_SobrenomedoAluno.zip. O arquivo principal (o que contém o main do trabalho) obrigatoriamente deve estar com o nome "main". Note que a data limite já leva em conta um dia adicional de tolerância para o caso de problemas de submissão via rede. Isso significa que o aluno deve submeter seu trabalho até no máximo um dia antes da data limite. Se o aluno resolver submeter o trabalho na data limite, estará fazendo isso assumindo o risco do trabalho ser cadastrado no sistema após o prazo. Em caso de recebimento do trabalho após a data limite, o trabalho não será avaliado e a nota será ZERO. Aluno que receber zero por este motivo e vier pedir para o professor considerar o trabalho estará cometendo um ato de DESRESPEITO ao professor e estará sujeito a perda adicional de pontos na média.

V. Data de Entrega: 06/12/2020

A correção/revisão dos trabalhos será marcada posteriormente.

VI. Avaliação

Os trabalhos terão nota zero se:

A data de entrega for fora do prazo estabelecido;

O trabalho não compilar;

O trabalho não gerar o arquivo com o resultado e formato esperado;

For detectada a ocorrência de plágio.

Observação importante

Caso haja algum erro neste documento, serão publicadas novas versões e divulgadas erratas em sala de aula. É responsabilidade do aluno manter-se informado, freqüentando as aulas ou acompanhando as novidades na página da disciplina na Internet.