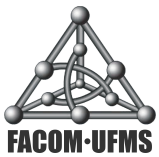

Algoritmos e Programação Orientada a Objetos I

Prof. Bruno M. Nogueira

Faculdade de Computação - UFMS

Segundo Trabalho - T2

1 Descrição do problema

Neste trabalho, você será responsável por desenvolver um sistema de apoio a pesquisadores da área de Aprendizado de Máquina (AM). Seu sistema deve permitir aos pesquisadores a gestão completa de seus experimentos, de maneira a facilitar a obtenção dos resultados.

Usualmente, pesquisadores em Aprendizado de Máquina têm o interesse de comparar o desempenho de diferentes algoritmos em cenários distintos. Para isso, utilizam-se diferentes conjuntos de dados (*datasets*), com diferentes características, sob os quais são aplicados os algoritmos.

Algoritmos de Aprendizado de Máquina têm parâmetros que definem seu comportamento. Por exemplo, uma rede neural tem o número de neurônios e o número de camadas ocultas a serem utilizadas; o algoritmo k-vizinhos mais próximos tem o valor de vizinhos a ser considerado, dentre outros. Diferentes valores destes parâmetros definem diferentes comportamentos dos algoritmos, levando-os a desempenhar melhor - ou pior - em diferentes conjuntos de dados.

Por fim, um mesmo algoritmo, com um mesmo conjunto de parâmetros, pode ser avaliado por diferentes medidas de avaliação. Cada medida de avaliação fornece uma "visão" diferente do desempenho do algoritmo. Por exemplo, usualmente deseja-se saber quais foram os valores de precisão, revocação e acurácia dos algoritmos em um mesmo conjunto de dados.

O diagrama das classes a serem utilizadas neste seu sistema está disponível na Figura 1. Nas seções a seguir, essas classes serão brevemente descritas.

Classe Algoritmo

Esta é a classe responsável por fornecer a abstração simplificada de um algoritmo de Aprendizado de Máquina. Nesta abstração, temos os seguintes atributos:

- nome: variável textual que armazena o nome do algoritmo.
- parametros: vetor com três posições de variáveis em ponto flutuante que armazena os parâmetros do algoritmo utilizado. Um algoritmo pode ter de 0 a 3 parâmetros. Assim, nem todas as posições do vetor poderão ser utilizadas. Posições não utilizadas deverão ser preenchidas com o valor curinga $Float.MAX\_VALUE$.

Essa classe conta com os seguintes métodos:

- Método construtor: deve receber o nome e o conjunto de parâmetros do algoritmo, inicializando, da maneira apropriada, as variáveis de instância.
- getNome: deve retornar o nome do algoritmo.
- getParametros: deve retornar o conjunto de parâmetros do algoritmo.

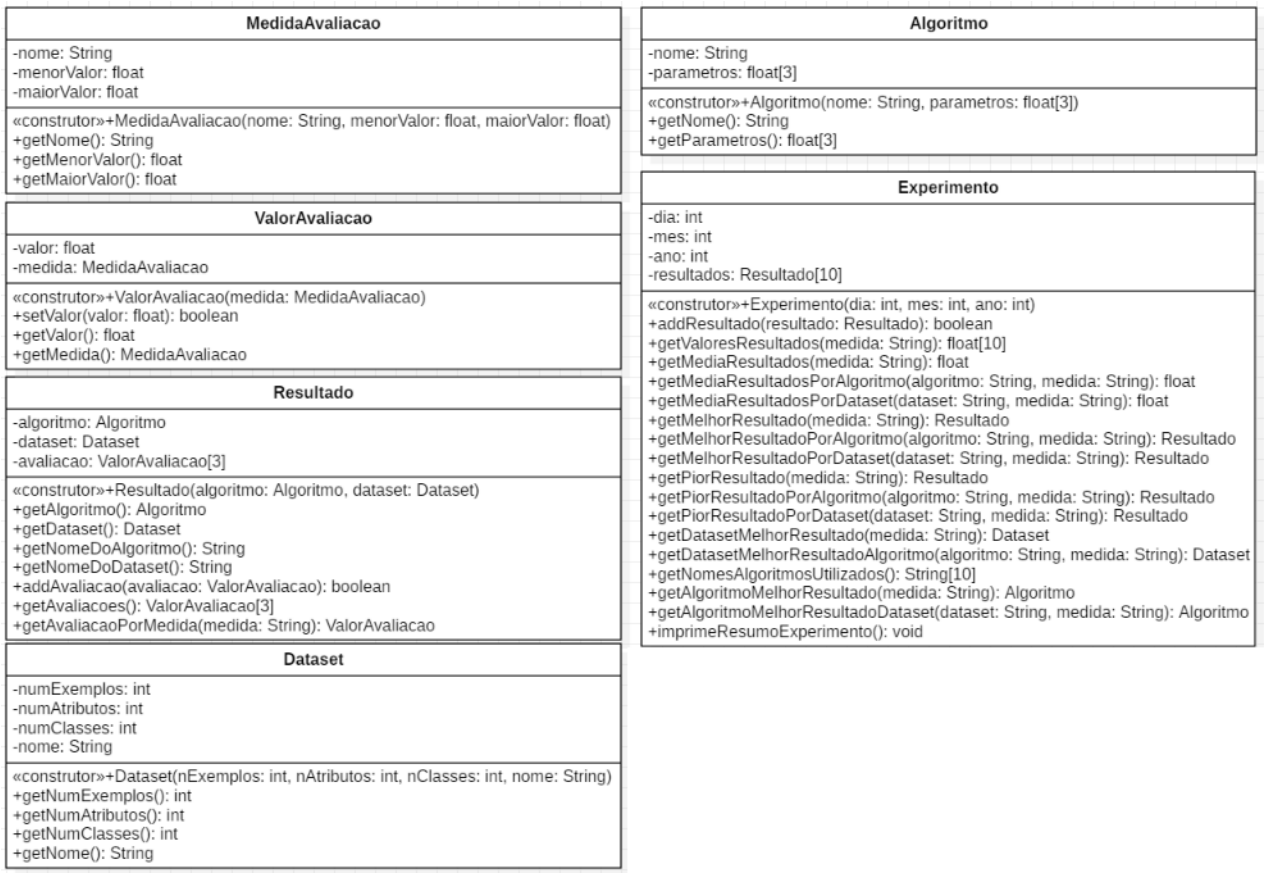

Figura 1: Diagrama de classes para o sistema de apoio a pesquisadores de Aprendizado de Máquina.

Classe MedidaAvaliacao

Neste classe, temos a abstração de uma medida de avaliação que poderá ser utilizada pelos pesquisadores para avaliar seus algoritmos. São atributos desta classe:

- nome: variável textual que armazena o nome da medida de avaliação.
- menorValor: variável em ponto flutuante que armazena o menor valor que a medida de avaliação pode assumir.
- maiorValor: variável em ponto flutuante que armazena o maior valor que a medida de avaliação pode assumir.

Além disso, a classe tem os seguintes métodos:

- método construtor: deve receber como parâmetros o nome da medida de avaliação, seu menor valor e seu maior valor, respectivamente. Deve inicializar, de maneira adequada, as variáveis de instância.
- getNome: deve retornar o nome da medida de avaliação.
- getMenorValor: deve retornar o menor valor que a medida de avaliação pode assumir.
- getMaiorValor: deve retornar o maior valor que a medida de avaliação pode assumir.

Classe ValorAvaliacao

Classe responsável por representar um valor de avaliação obtida por um algoritmo em uma determinada medida de avaliação. São seus atributos:

- valor: variável de ponto flutuante que armazena o score obtido pelo algoritmo na medida de avaliação utilizada.
- medida: variável do tipo MedidaAvaliacao que armazena a medida de avaliação à qual pertence este valor obtido.

São métodos desta classe:

- Método construtor: deve receber uma instância da classe MedidaAvaliação, que representa a medida de avaliação à qual pertence este valor. Deve, também, inicializar, de maneira adequada, as variáveis de instância.
- setValor: método responsável por armazenar o valor obtido pelo algoritmo na medida de avaliação. Deve receber como parâmetro o valor da medida. Este valor deve ser validado, de acordo com os limites aceitos pela medida utilizada. Caso o valor seja aceito, deve-se guardá-lo na variável de instância correspondente e retornar o valor *true*, assinalando que a atribuição foi bem sucedida. Caso contrário, deve imprimir uma mensagem na tela informando da impossibilidade de atribuição e retornar o valor *false*.
- getMedida: deve retornar a medida de avaliação utilizada.
- getValor: deve retornar o valor da medida de avaliação armazenada no objeto.

Classe Dataset

Esta classe fornece uma abstração de um conjunto de dados utilizado no experimento. Contem os seguintes atributos:

- numExemplos: variável inteira que armazena o número de exemplos contidos no conjunto de dados.
- numAtributos: variável inteira que armazena o número de atributos contidos no conjunto de dados.
- numClasses: variável inteira que armazena o número de classes contidas no conjunto de dados.
- nome: variável textual que armazena o nome do conjunto de dados.

São métodos desta classe:

- Método construtor: deve receber como parâmetros o número de exemplos, o número de atributos, o número de classes e o nome do conjunto de dados, inicializando, de maneira adequada, as variáveis de instância.
- getNumExemplos: deve retornar o número de exemplos do conjunto de dados.
- getNumAtributos: deve retornar o número de atributos do conjunto de dados.
- getNumClasses: deve retornar o número de classes do conjunto de dados.
- getNome: deve retornar o nome do conjunto de dados.

Classe Resultado

Esta classe fornece uma abstração de um resultado obtido nos experimentos realizados de um algoritmo em um conjunto de dados, de acordo com algumas medidas de avaliação. São atributos desta classe:

- algoritmo: variável do tipo Algoritmo que armazena o algoritmo e suas configurações que foram utilizadas para obtenção deste resultado.
- dataset: variável do tipo Dataset que armazena o conjunto de dados utilizado na obtenção deste resultado.
- avaliacao: vetor de três posições do tipo ValorAvaliacao que armazenam os valores de avaliação obtidas pelo algoritmo no conjunto de dados, de acordo com até três medidas de avaliação. Um algoritmo pode ser avaliado utilizando de 1 a 3 medidas de avaliação. Assim, nem todas as posições do vetor estarão preenchidas. Posições vazias devem conter o valor *null*.

São métodos desta classe:

- Método construtor: deve receber como parâmetros o algoritmo e o conjunto de dados utilizado na obtenção deste resultado. Deve inicializar as variáveis de instância da maneira adequada.
- getAlgoritmo: deve retornar o algoritmo utilizado neste resultado.
- getDataset: deve retornar o conjunto de dados utilizado neste resultado.
- getNomeDoAlgoritmo: deve retornar o nome do algoritmo utilizado neste resultado.
- getNomeDoDataset: deve retornar o nome do conjunto de dados utilizado neste resultado.
- addAvaliacao: método utilizado para adicionar um resultado de avaliação do algoritmo no conjunto de dados. Uma avaliação só poderá ser adicionada se houver espaço no vetor de avaliações. Além disso, só é possível haver uma avaliação para cada medida de avaliação. Assim, não é possível haver no vetor de avaliações duas ou mais avaliações que utilizem a mesma medida de avaliação. Este método deve retornar *true* caso a adição de uma avaliação seja bem sucedida. Caso a adição de uma avaliação não seja possível, deve imprimir uma mensagem na tela informando da impossibilidade e retornar o valor *false*.
- getAvaliacoes: deve retornar o vetor de avaliações obtidas neste resultado.
- getAvaliacaoPorMedida: deve receber como parâmetro o nome da medida de avaliação que se deseja obter o valor. Caso a medida não seja encontrada no vetor de avaliações, este método deve retornar *null*. Caso a medida seja encontrada, seu objeto correspondente do tipo *ValorAvaliacao* deve ser retornado.

Classe Experimento

Esta é a classe que armazena todos os dados dos experimentos e que permite ao pesquisador a obtenção de algumas informações sobre o experimento. São atributos desta classe:

- dia: variável inteira que armazena o dia de realização do experimento.
- mes: variável inteira que armazena o mês de realização do experimento.

– ano: variável inteira que armazena o ano de realização do experimento.

– resultados: vetor de 10 posições do tipo Resultado que armazena os resultados obtidos por algoritmos em conjuntos de dados. Um experimento pode ter de 1 a 10 resultados. Assim, caso uma posição não seja preenchida, ela deve assumir valor *null*.

São métodos desta classe:

– Método construtor: deve receber como parâmetros o dia, o mês e o ano do experimento. Deve inicializar as variáveis de instância de maneira adequada.

– addResultado: método utilizado para adição de um resultado no experimento. Um resultado só pode ser adicionado se não exceder o máximo de 10 resultados. Além disso, não é possível haver repetição de resultados. Desta forma, seu vetor de resultados não deve conter duas vezes resultados do mesmo algoritmo, com os mesmos parâmetros, no mesmo conjunto de dados. Caso um resultado seja adicionado com sucesso, este método deve retornar o valor *true*. Caso o resultado não seja adicionado, deve-se imprimir uma mensagem informativa na tela e retornar o valor *false*.

– getValoresResultados: deve receber como parâmetro o nome da medida que deseja-se saber o valor nos diferentes resultados. Este método retorna um vetor de 10 posições, contendo o valor da medida de avaliação nos (até) 10 resultados. Caso a medida não tenha sido utilizada em um determinado resultado, deve-se preencher a posição correspondente no vetor de resposta com o valor curinga $Float.MAX\_VALUE$.

– getMediaResultados: deve receber como parâmetro o nome da medida que deseja-se saber a média ao longo dos diferentes resultados. A média deve ser calculada considerando o número de resultados válidos contidos no conjunto de resultados (ou seja, considerando o número de resultados que, de fato, utilizaram aquela medida de avaliação). Deve retornar um valor em ponto flutuante contendo a média. Caso a medida não tenha sido utilizada em nenhum resultado, deve retornar o valor curinga $Float.MAX\_VALUE$.

– getMediaResultadosPorAlgoritmo: similar ao método anterior, deseja-se saber a média de uma medida obtida por um algoritmo ao longo dos diferentes resultados. Deve receber como parâmetros o nome do algoritmo e o nome da medida a serem pesquisadas. A média deve ser calculada considerando o número de resultados válidos contidos no conjunto de resultados (ou seja, considerando o número de resultados para o algoritmo pesquisado que, de fato, utilizaram aquela medida de avaliação). Deve retornar um valor em ponto flutuante contendo a média. Caso a medida ou o algoritmo não tenham sido utilizados em nenhum resultado, deve retornar o valor curinga $Float.MAX\_VALUE$.

– getMediaResultadosPorDataset: também similar ao método anterior, deseja-se saber a média de uma medida obtida por qualquer algoritmo ao longo dos diferentes resultados que utilizaram um determinado conjunto de dados. Deve receber como parâmetros o nome do conjunto de dados e o nome da medida a serem pesquisadas. A média deve ser calculada considerando o número de resultados válidos contidos no conjunto de resultados (ou seja, considerando o número de resultados que, de fato, utilizaram aquela medida de avaliação naquele conjunto de dados). Deve retornar um valor em ponto flutuante contendo a média. Caso a medida ou o algoritmo não tenham sido utilizados em nenhum resultado, deve retornar o valor curinga $Float.MAX\_VALUE$.

– getMelhorResultado: deve receber como parâmetro do nome da medida a ser pesquisada e retornar o objeto do tipo Resultado que apresenta o maior valor para a medida pesquisada. Caso a medida não tenha sido utilizada, deve-se retornar o valor *null*.

- getMelhorResultadoPorAlgoritmo: deve receber como parâmetros o nome da medida e o nome do algoritmo a serem pesquisados. Deve retornar o objeto do tipo Resultado que apresenta o maior valor para a medida pesquisada apresentado pelo algoritmo especificado. Caso a medida e o algoritmo não tenham sido utilizados em um mesmo resultado, deve retornar o valor *null*.

- getMelhorResultadoPorDataset: similar ao anterior, deve receber como parâmetros o nome da medida e o nome do conjunto de dados a serem pesquisados. Deve retornar o objeto do tipo Resultado que apresenta o maior valor para a medida pesquisada apresentado por qualquer algoritmo no conjunto de dados especificado. Caso a medida e o conjunto de dados não tenham sido utilizados em um mesmo resultado, deve retornar o valor *null*.

- getPiorResultado: deve receber como parâmetro do nome da medida a ser pesquisada e retornar o objeto do tipo Resultado que apresenta o menor valor para a medida pesquisada. Caso a medida não tenha sido utilizada, deve-se retornar o valor *null*.

- getPiorResultadoPorAlgoritmo: deve receber como parâmetros o nome da medida e o nome do algoritmo a serem pesquisados. Deve retornar o objeto do tipo Resultado que apresenta o menor valor para a medida pesquisada apresentado pelo algoritmo especificado. Caso a medida e o algoritmo não tenham sido utilizados em um mesmo resultado, deve retornar o valor *null*.

- getPiorResultadoPorDataset: similar ao anterior, deve receber como parâmetros o nome da medida e o nome do conjunto de dados a serem pesquisados. Deve retornar o objeto do tipo Resultado que apresenta o menor valor para a medida pesquisada apresentado por qualquer algoritmo no conjunto de dados especificado. Caso a medida e o conjunto de dados não tenham sido utilizados em um mesmo resultado, deve retornar o valor *null*.

- getDatasetMelhorResultado: deve receber como parâmetro o nome da medida a ser considerada e retornar, dentre todos os resultados apresentados, o objeto do tipo Dataset que representa o conjunto de dados que obteve o maior valor da medida especificada. Caso a medida não tenha sido utilizada em nenhum resultado, deve retornar *null*.

- getDatasetMelhorResultadoAlgoritmo: deve receber como parâmetros os nomes da medida e do algoritmo a serem considerados. Este método deve retornar um objeto do tipo Dataset que representa o conjunto de dados no qual o algoritmo especificado apresentou o maior valor da medida especificada. Caso a medida e o algoritmo não tenham sido utilizados juntos em nenhum resultado, deve retornar *null*.

- getNomesAlgoritmosUtilizados: deve retornar um vetor contendo o nome dos algoritmos utilizados, sem repetição (isso é, se um mesmo algoritmo é apresentado em mais de um resultado, seu nome deve aparecer uma única vez neste vetor). Posições não preenchidas neste vetor devem conter o valor *null*.

- getAlgoritmoMelhorResultado: deve receber como parâmetro o nome da medida a ser considerada e retornar o objeto do tipo Algoritmo que, dentre todos os resultados, obteve o maior valor daquela medida. Caso a medida não tenha sido utilizada em nenhum resultado, deve retornar o valor *null*.

- getAlgoritmoMelhorResultadoDataset: deve receber como parâmetros o nome do conjunto de dados e da medida a serem considerados. Este método deve retornar o objeto do tipo Algoritmo que, para a medida e conjunto de dados especificados, apresenta o maior valor. Caso a medida e o conjunto de dados não tenham sido utilizados em um mesmo resultado, deve retornar *null*.

– imprimeResumoExperimento: este método deve imprimir um resumo completo do experimento. Deve mostrar, a data de realização dos experimentos e listar todos os detalhes dos resultados obtidos, sendo, para cada resultado: nome do algoritmo utilizado, com os respectivos parâmetros; nome do conjunto de dados utilizado, com seu número de exemplos, atributos e classes; nome e valor de cada medida de avaliação obtida pelo algoritmo no conjunto de dados.

2 Entrega

Este trabalho poderá ser realizado em grupos de no mínimo 3 e no máximo 4 integrantes. O grupo deve entregar, compactados em um único arquivo .zip ou .rar, via AVA: (i) os arquivos .java e .class das três classes aqui descritas; e (ii) os arquivos .java e .class de uma classe ExperimentoTeste, em que TODAS as funcionalidades das classes aqui descritas são testadas.

O prazo para entrega é às 23:55 do dia 14/06/2019. Serão aceitas apenas entregas realizadas dentro do prazo. Não serão aceitas entregas realizadas por outros meios que não a postagem do arquivo no AVA.

Serão critérios de correção do código:

1. Completude do modelo apresentado - todas as classes com todos os métodos e atributos (peso 4);

2. Corretude de funcionalidades - todos os métodos funcionando corretamente (peso 5);

3. Classe de teste com todos os métodos sendo testados (peso 1).

As classes serão testadas por um script automático gerado pelo professor. Assim, suas classes devem atender ESTRITAMENTE o diagrama apresentado. Métodos adicionais podem ser criados, mas os métodos especificados no diagrama não podem ser modificados. Não poderão ser criados atributos para as classes, além dos que aqui foram especificados. Além disso, será realizada análise de plágio entre os códigos entregues. QUALQUER indício de plágio será considerada uma transgressão grave e incindirá em nota zero para todos os componentes dos grupos envolvidos.

O desenvolvimento do trabalho será acompanhado pelo professor em pontos de controle, a serem realizados em nossas aulas de laboratório. Assim, o código do desenvolvimento deverá estar disponível em todas as nossas aulas, para apresentação ao professor. Nos pontos de controle, o professor poderá indagar qualquer integrante do grupo sobre o desenvolvimento, sendo o conjunto das respostas e participação no desenvolvimento do trabalho um componente da nota individual final do trabalho.

Qualquer dúvida, não hesite em consultar o professor.

Bom trabalho!