UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Departamento de Ciência da Computação

DCC207 -- Algoritmos 2
Prof. Renato Vimieiro

Trabalho Prático 1 -- Geometria computacional

Objetivos

Nesse trabalho serão abordados os aspectos práticos de diversos algoritmos de geometria computacional visto nas aulas teóricas. Especificamente, serão explorados aspectos de implementação de algoritmos de envoltória convexa e verificação de interseção de segmentos para o problema de classificação em aprendizado de máquina supervisionado.

O objetivo secundário é fixar o conteúdo. Entende-se que ao implementar a estrutura o aluno conseguirá compreender melhor os conceitos explorados. Dessa forma, o conteúdo teórico será melhor absorvido e fixado. Além disso, os alunos terão a oportunidade de ver conceitos não abordados na disciplina, no caso específico, o problema de classificação em aprendizado de máquina.

Tarefas

Os alunos deverão implementar um algoritmo para criar modelos de classificação em aprendizado supervisionado. O problema de aprendizado supervisionado consiste em criar um modelo que, baseado em dados de um conjunto de treinamento, é capaz de representar as características principais dos dados e posteriormente atribuir o valor de uma variável alvo a dados para os quais se desconhece tal valor. No caso do problema de classificação, essa tarefa consiste em 'aprender' as características das amostras com diferentes rótulos na base de dados para, depois, atribuir rótulos a amostras desconhecidas. Exemplificando, suponha que se deseja fazer a triagem automática de pacientes que chegam à emergência de um hospital. Com base nos sintomas apresentados pelos pacientes que visitaram a emergência anterior e na classificação atribuída pelos profissionais de saúde a esses pacientes (alta, média ou baixa prioridade, por exemplo), o objetivo do algoritmo de classificação é ajustar um modelo sobre esses dados para que, no futuro, novos pacientes sejam rotulados automaticamente simplesmente analisando os sintomas descritos. Naturalmente, tal problema é bastante complexo e, por se tratar de uma disciplina de algoritmos, não exploraremos a fundo todas as questões relacionadas ao problema. Dessa forma, o algoritmo que será desenvolvido será uma versão simplificada de outros métodos que utilizam processos similares para o objetivo exposto.

O algoritmo a ser criado deverá ajustar modelos lineares para os dados. Considerando que os pacientes podem ser representados como ponto num espaço n-dimensional, o objetivo é encontrar os hiperplanos que separam as classes (suas amostras) umas das outras. Note, contudo, que as classes podem não ser linearmente separáveis, já que amostras de diferentes classes podem ocupar uma mesma região no espaço considerado. Os algoritmos, nesse caso, precisam fazer ajustes para obter os hiperplanos que melhor separam as classes (mesmo esta divisão não sendo perfeita). Novamente, nosso objetivo não é implementar de forma fidedigna tais abordagens, assim, consideraremos a seguinte simplificação: caso os dados não sejam linearmente separáveis, o algoritmo deverá simplesmente reportar a impossibilidade de se criar um modelo linear para tal conjunto de dados. Caso os dados sejam linearmente separáveis, então o algoritmo deverá reportar tal modelo.

Assim, em princípio, deverá ser implementado um método que verifique a separabilidade linear dos dados. Nesse sentido, adotaremos outra simplificação no trabalho prático: serão consideradas apenas duas variáveis para descrever as amostras. Em outras palavras, nossas amostras serão pontos no plano cartesiano. Para verificar a separabilidade dos dados, o método deverá computar as envoltórias convexas de cada classe (será assumido que os dados possuem apenas dois rótulos para simplificar). Em seguida, deverá ser verificado se há ou não sobreposição dessas classes/envoltórias através do algoritmo de varredura linear para verificação de interseção de segmentos. Caso não haja interseção entre segmentos de envoltórias distintas, então os dados são linearmente separáveis.

Para gerar os modelos lineares, o algoritmo utilizará as envoltórias convexas computadas na etapa anterior. O algoritmo deve encontrar os pontos mais próximos entre as duas envoltórias. O modelo a ser reportado será a reta perpendicular ao segmento que une esses dois pontos e que passa sobre seu ponto médio. O modelo a ser reportado é a equação dessa reta.

O classificador deve operar da seguinte forma: o modelo atribui o mesmo rótulo a uma nova amostra que os pontos da envoltória que se localizam na mesma região. Isto é, se um ponto estiver abaixo/à esquerda da reta gerada pelo modelo, então ele receberá o mesmo rótulo que os pontos da envoltória dessa região.

A figura abaixo ilustra essa situação. Os pontos em vermelho representam amostras de uma classe, enquanto os pontos em azul representam os da outra classe. Como vemos, eles são linearmente separáveis. Dessa forma, localizamos os pontos mais próximos entre as duas envoltórias (os pontos ligados pelo segmento preto). Depois, calculamos a reta perpendicular ao ponto médio entre esses dois pontos (reta em verde). Para classificar uma nova amostra, usamos a equação da reta exibida no gráfico. Se o ponto estiver abaixo, ele recebe o rótulo vermelho, se ele estiver acima, então recebe o rótulo azul. Pontos que estiverem exatamente sobre a reta não recebem qualquer rótulo (possuem a mesma chance de serem vermelhos ou azuis).

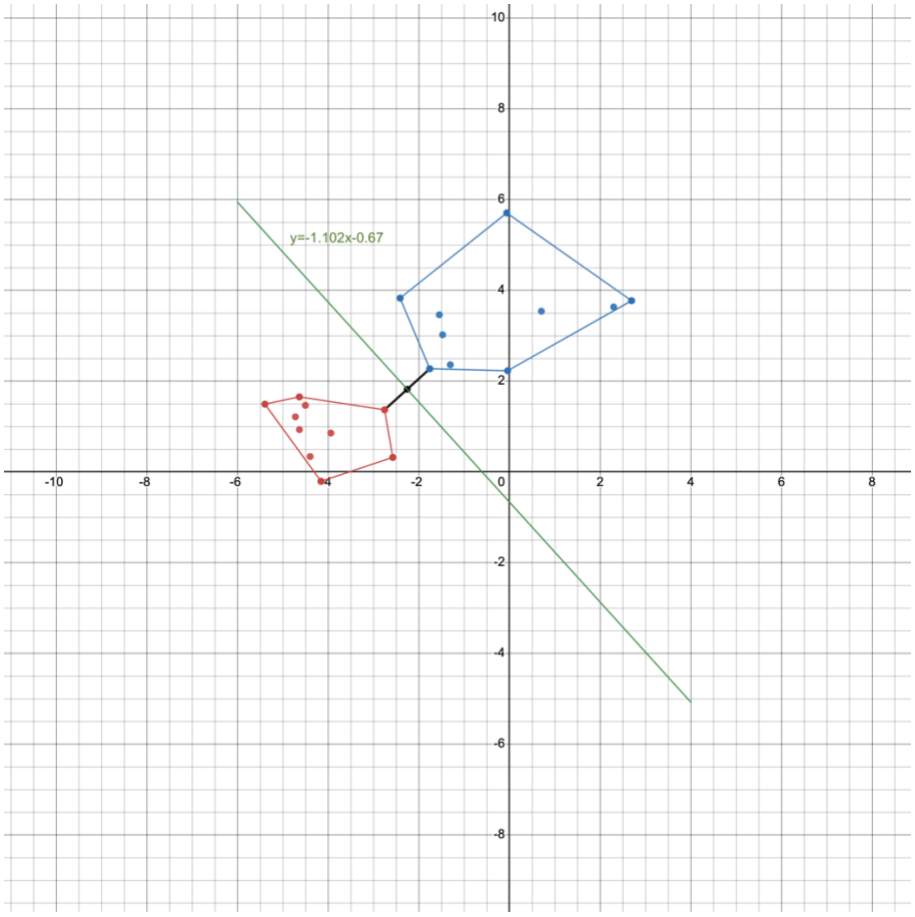

Figura 1. Exemplo de construção de um modelo classificador para um conjunto de dados sintético.

Após a conclusão da implementação dos algoritmos, os alunos deverão avaliar o desempenho do classificador na sua tarefa. Para isso, serão realizados diversos experimentos com conjuntos de dados reais (modificados para o propósito do trabalho). Esses conjuntos deverão ser divididos em dois: um subconjunto com 70% das amostras que será usado para o ajuste do modelo (treinamento), e um outro de 30% usado para avaliar o desempenho do modelo na tarefa de classificação (mensurar a taxa de acertos com dados desconhecidos). Deverão ser computadas as métricas: precisão, revocação e f1-escore. A computação dessas métricas poderá ser realizada através da biblioteca Scikit-Learn de Python, ou através de implementação própria com base nas suas definições (verificar entrada na Wikipedia).

Por fim, deverá ser escrito um relatório, podendo esse ser escrito diretamente no Jupyter Notebook em que o código tiver sido desenvolvido, descrevendo as etapas, escolhas de implementações, e resultados dos experimentos. O relatório deve conter um exemplo gráfico similar ao que acompanha essa descrição mostrando: as envoltórias convexas encontradas para cada classe, o segmento que une os pontos mais próximos das envoltórias, e a reta que define o modelo bem como sua equação. A escolha da base fica a critério do aluno. Os gráficos podem ser gerados através de implementação no próprio notebook, ou através de softwares externos e anexados ao documento. Esses, contudo, quando anexados, devem ter boa resolução para permitir que outros leitores consigam visualizá-los perfeitamente.

Os experimentos devem contemplar, no mínimo, dez bases de dados distintas extraídas de repositórios de dados públicos. Alguns links serão disponibilizados abaixo. Devem ser escolhidas somente bases de dados com atributos numéricos, podendo, inclusive, ser imagens (exemplos bases de dados de classificação de objetos ou dígitos manuscritos). Como trabalharemos em duas dimensões, os dados deverão ser pré-processados para transformá-los em bidimensionais. Para isso, podem ser usados algoritmos implementados na biblioteca Scikit-Learn. Exemplos de algoritmos que podem ser usados: t-SNE, PCA, SVD.

Em resumo, são tarefas do trabalho prático:

1. Implementar um algoritmo de envoltória convexa visto em sala de aula
2. Implementar o algoritmo de varredura linear para detecção de interseções em conjuntos de segmentos
3. Implementar o método para verificação de separabilidade linear conforme descrito acima
4. Implementar o método para construir o modelo, caso os dados sejam linearmente separáveis
5. Implementar o classificador que recebe um conjunto de amostras desconhecidas e atribui rótulos a elas
6. Implementar o método para computar as métricas de classificação para os experimentos
7. Realizar os experimentos conforme a descrição acima
8. Redigir o relatório especificado acima

O uso de bibliotecas adicionais deve ser discutido com o professor.

O trabalho poderá ser feito em **grupos de até três alunos**.

O que entregar?

Devem ser entregues todos os arquivos fonte usados na implementação. Caso o relatório não esteja junto com o código fonte, então esse deverá ser entregue à parte em formato pdf. O trabalho deverá ser entregue em um repositório do GitHub a ser tornado público somente após a entrega. O link para o repositório será entregue no Teams.

Política de Plágio

Os alunos podem, e devem, discutir soluções sempre que necessário. Dito isso, há uma diferença bem grande entre implementação de soluções similares e cópia integral de ideias. Trabalhos copiados na íntegra ou em partes de outros alunos e/ou da internet serão prontamente anulados. Caso haja dois trabalhos copiados por alunos/grupos diferentes, ambos serão anulados.

Datas

Entrega final Teams: 22/10/2023 às 23h59

Dada a complexidade do trabalho, seguiremos um esquema de entregas parciais em que os grupos não submeterão efetivamente seus trabalhos para avaliação, porém, espera-se que tenham terminado as tarefas nas respectivas datas a fim de organizar e viabilizar a entrega final. Segue a sugestão de cronograma:

1. Envoltória convexa: **03/10/2023**
2. Varredura linear: **05/10/2023** (pode ser desenvolvida em paralelo com a envoltória)
3. Verificação de separabilidade: 1**0/10/2023**
4. Construção do modelo e classificador: **15/10/2023**
5. Método para computação das métricas: **15/10/2023**
6. Conclusão dos experimentos: **17/10/2023**
7. Conclusão do relatório: **22/10/2023**

Notem que os experimentos podem requerer bastante tempo (horas) para serem executados, dependendo da implementação do grupo. Esse tempo deve ser considerado na organização do trabalho para evitar problemas com a entrega final.

Política de atraso

Haverá tolerância de 30min na entrega dos trabalhos. Submissões feitas depois do intervalo de tolerância serão penalizados.

- Atraso de 1 dia: 30%
- Atraso de 2 dias: 50%
- Atraso de 3+ dias: não aceito

Serão considerados atrasos de 1 dia aqueles feitos após as 0h30 do dia de entrega. A partir daí serão contados o número de dias passados da data de entrega.

Referências

- https://sci2s.ugr.es/keel/category.php?cat=clas#inicio
- https://scikit-learn.org/stable/datasets/real_world.html
- https://scikit-learn.org/stable/datasets/toy_dataset.html
- https://www.cs.toronto.edu/~kriz/cifar.html
- https://archive.ics.uci.edu
- https://pt.wikipedia.org/wiki/Precisão_e_revocação