Instituto Federal de Educação, Ciência e Tecnologia

Câmpus Campinas
ED1 – Estruturas de Dados I

Professores: José Américo (jose.americo@ifsp.edu.br)
Samuel Martins (samuel.martins@ifsp.edu.br)

Lab 08 – Machine Learning

Machine Learning é uma das áreas de maior interesse (e mais legal =]) ultimamente. Suas técnicas objetivam fazer com que a máquina aprenda a fazer previsões sobre dados, a partir de um conhecimento prévio. Em outras palavras, a partir de um conjunto de dados previamente conhecidos, a máquina aprende padrões para predizer comportamentos, ações, etc, em novos dados.

Neste laboratório, você aprenderá diversos conceitos da área e um algoritmo simples, porém poderoso, para a classificação de dados. *Todos os dados usados são fictícios.*

1) Descrição

A empresa "Se Liga nos Dados" foi recentemente contratada por uma escola para desenvolver o sistema "Será que Passa" que visa prever se os alunos atuais do ensino médio passarão ou não no vestibular. Para isso, a empresa analisou o histórico de aprovações dos alunos egressos da escola nos últimos anos, colhendo os seguintes dados de cada um: **nota média na escola, horas de estudos semanais**, **resultado do vestibular (aprovado ou não)**.

Cada aluno consiste de uma **amostra de dados *(sample)***, que possui um **conjunto de características *(feature vector)*** a serem analisadas. Em problemas de **classificação (aprendizado supervisionado)**, cada amostra possui um ou mais **rótulos/classes *(labels)***.

No exemplo acima, cada **amostra** (aluno) possui apenas duas características: **nota média, horas de estudos semanais**, e um único rótulo: seu **resultado do vestibular**, que indica se o aluno passou ou não no vestibular. Amostras conhecidas previamente denominam-se **amostras de treinamento.** O conjunto de amostras de treinamento constitui o **conjunto de treinamento**, que será usado para treinar a máquina a tomar a decisão sozinha.

Machine Learning possui técnicas de **classificação** que visam fazer com que a máquina aprenda, a partir de **amostras de treinamento** (conhecimento), a predizer o **rótulo** de *novas amostras*, denominadas **amostras de teste**. Tais técnicas aprendem a reconhecer padrões nos dados conhecidos, e atribuem um rótulo (classificação) nas novas amostras ainda não vistas.

Para o problema em questão, queremos que a máquina preveja sozinha se um dado aluno

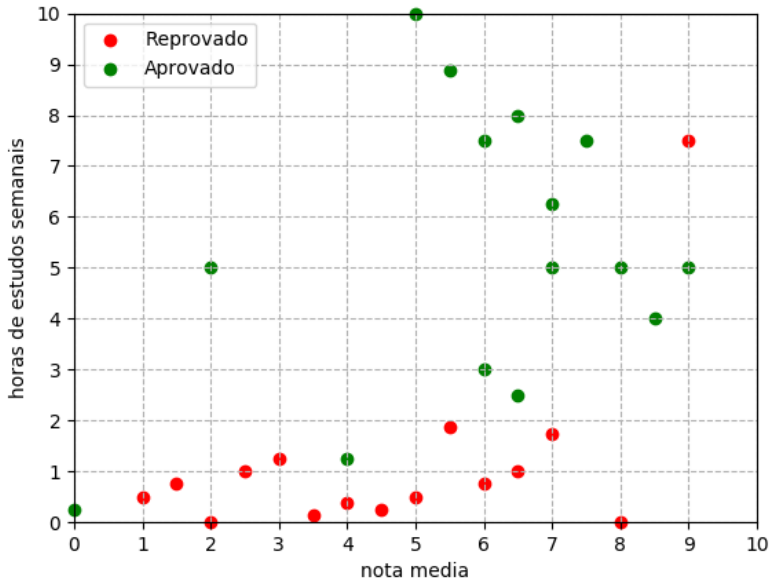
atual, que ainda não prestou vestibular, passará ou não no mesmo, sabendo apenas sua nota média na escola e as horas de estudos semanais que ele se dedica.

Suponha que a empresa colheu os seguintes dados de 30 alunos egressos:

Nota média, horas de estudos semanais, resultado do vestibular (1=aprovado, 0=não)

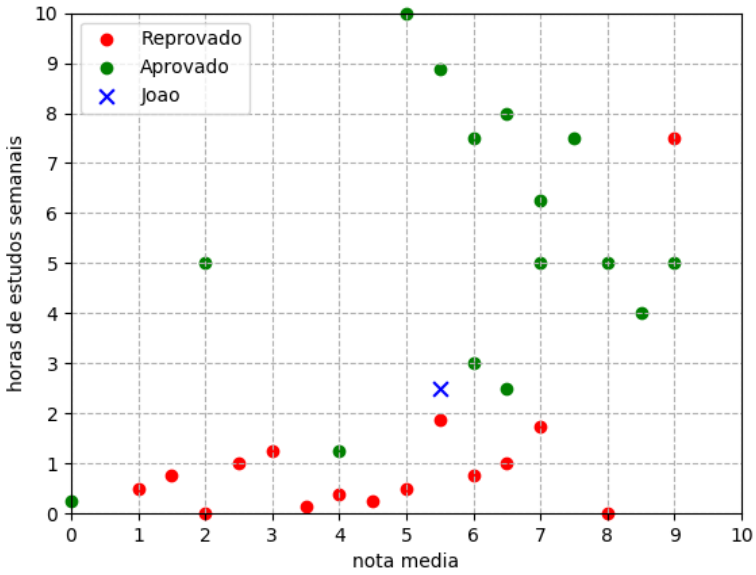
1.00,0.50,0	6.00,0.75,0	6.50,2.50,1
1.50,0.75,0	6.50,1.00,0	7.00,5.00,1
2.00,0.00,0	7.00,1.75,0	7.00,6.25,1
2.50,1.00,0	8.00,0.00,0	7.50,7.50,1
3.00,1.25,0	9.00,7.50,0	8.00,5.00,1
3.50,0.13,0	5.00,10.00,1	8.50,4.00,1
4.00,0.38,0	5.50,8.88,1	9.00,5.00,1
4.50,0.25,0	6.00,3.00,1	0.00,0.25,1
5.00,0.50,0	6.00,7.50,1	4.00,1.25,1
5.50,1.88,0	6.50,8.00,1	2.00,5.00,1

Podemos então plotar tais dados em um gráfico 2D, uma vez que observamos apenas 2 características de cada aluno. Se observássemos uma terceira característica (p.ex. a idade), seria um gráfico 3D. Assuma que o eixo horizontal corresponde à nota média do aluno na escola, e o eixo vertical às horas de estudos semanais.



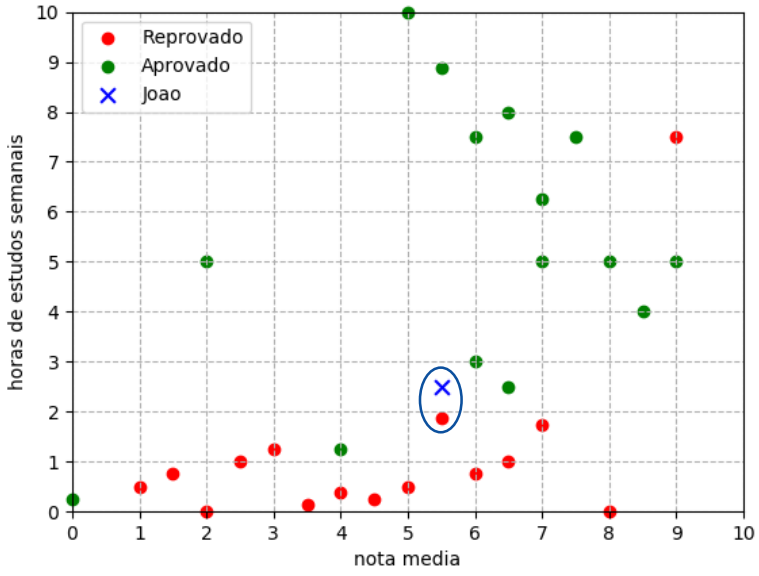
Suponha que desejamos saber se o aluno João passará no vestibular. Suas características (nota média, horas de estudos semanais) são, respectivamente: 5.5, 2.5

Ao plotar tais valores no mesmo gráfico, temos o seguinte:

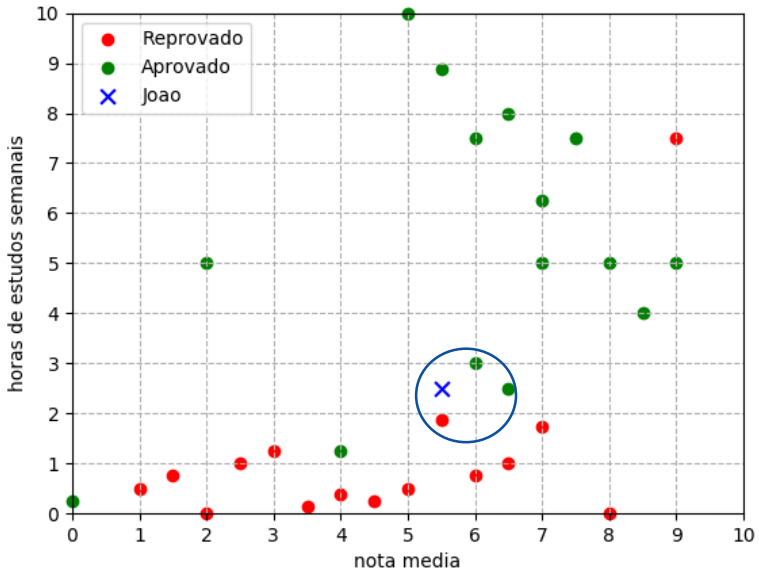


Para predizer se João irá passar ou não no vestibular, baseado nos dados previamente e conhecidos, podemos assumir que João terá **o rótulo do aluno com as características mais similares** a ele. Ao analisar o gráfico, selecionamos a amostra/ponto mais próximo a João, como indicado abaixo:

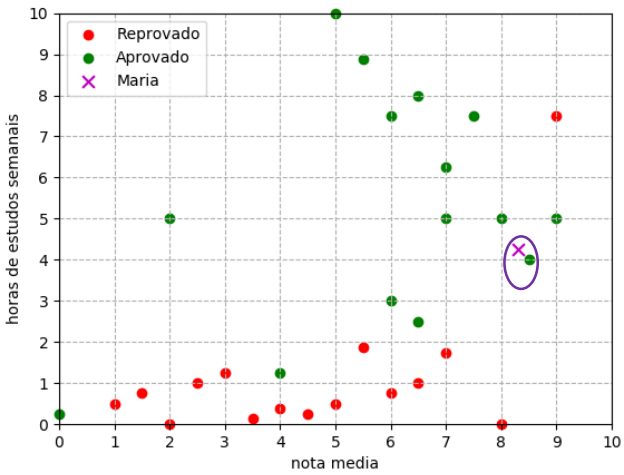
Como o aluno **mais similar** ao João **não passou no vestibular**, então o sistema previu que João **não passará no vestibular**.

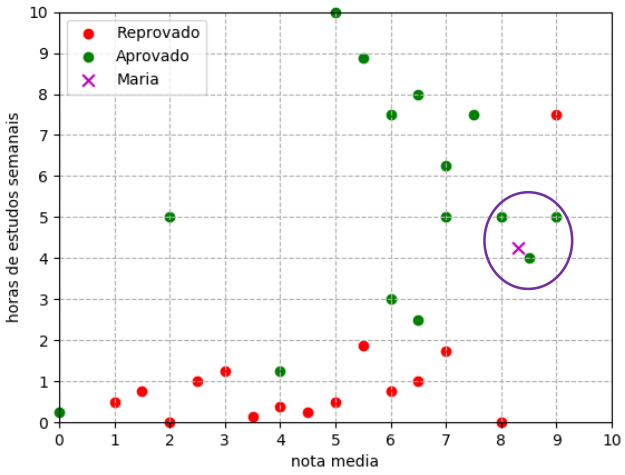


Entretanto, se ao invés de checar apenas o aluno mais similar, poderíamos checar os **3** alunos mais similares, atribuindo o rótulo que **mais frequente**, ou seja, que **mais vezes ocorreu**. Neste caso, o sistema previu que João **passará no vestibular**, como mostrado no gráfico abaixo.



Ao analisar a aluna Maria, cujas características são: 8.3 e 4.25, temos:





Ao considerar apenas o aluno mais similar e os 3 mais similares, o sistema previu que maria passará no vestibular.

A técnica de classificação apresentada é chamada **kNN (k-Nearest Neighbors).** Dado uma amostra de teste **q**, o **rótulo mais frequente** das **k amostras vizinhas mais próximas** (mais similares) de **q** será o rótulo de **q**. A distância entre as amostras pode ser calculada utilizando a **distância euclidiana entre dois pontos**, que, para o caso de apenas 2 características, resulta em:

$$d = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2},$$

em que $p = (p_1, p_2)$ é uma amostra de treinamento qualquer, e $q = (q_1, q_2)$ é a amostra de teste analisada. As coordenadas p_1, p_2 correspondem, respectivamente, à nota média e às horas de estudos do aluno p. Idem para a amostra q.

O número k de vizinhos é determinado pelo usuário. Para o caso de problemas de classificação com apenas 2 classes/rótulos, geralmente se escolhe um número ímpar, para evitar empates na atribuição do rótulo.

Não há um número k mágico para resolver todos os problemas. O melhor valor de k varia de problema para problema, sendo encontra por meio de experimentação e avaliação de vários possíveis valores.

Em resumo, o algoritmo kNN possui os seguintes passos.

Dado a amostra de teste **q** a ser classificada:

1) computa a distância euclidiana de **q** com todas as amostras de treinamento;
2) ordena as amostras de treinamento, em ordem crescente, baseado nas distâncias;
3) atribui o rótulo mais frequente das k amostras mais próximas a **q**;

Seu objetivo será desenvolver o sistema de classificação utilizando o algoritmo kNN.

2) Entrada

A primeira linha corresponderá a 3 valores: o número de amostras de treinamento **m**, o número de amostras de teste **n** a serem classificadas, e o valor de **k**.

As **m** linhas subsequentes contém as características e o rótulo de cada amostra do conjunto de treinamento.

As próximas **n** linhas consistem das características das amostras de teste a serem classificadas.

PS: Veja a dica de como plotar o gráfico com as amostras dos casos de teste na Seção 5;

3) Saída

Para cada amostra de teste, o programa deverá exibir a seguinte mensagem:

Aluno índice_amostra: (nota_media, horas_de_estudo) = resultado_previsto

ex:

Aluno 15: (5.50, 11.00) = Reprovado

Aluno 16: (8.50, 17.00) = Aprovado

Tanto as notas médias quanto às horas de estudos deverão ser impressas com precisão de 2 casas decimais.

4) Exemplos

Entrada	**Saída**
10 2 3 1.00 0.50 0 1.50 0.75 0 2.00 0.00 0 2.50 1.00 0 3.00 1.25 0 5.00 10.00 1 5.50 8.88 1 6.00 3.75 1 6.00 7.50 1 6.50 8.00 1 2.0 1.0 6.0 8.5	Aluno 0: (2.00, 1.00) = Reprovado Aluno 1: (6.00, 8.50) = Aprovado

5) Dicas

- Utilize o script python ***plot_test_case.py***, disponibilizado na página de submissão deste lab, para plotar um gráfico com as amostras de treinamento e de teste de um dado caso de teste:
  - python *plot_test_case.py 01.in out.png 1*
    - Plota o gráfico do caso de teste 01, salvando-o na imagem out.png.
    - 1 significa que você quer exibir o índice das amostras de treinamento e teste.
    - Para não exibi-los, utilize 0
    - É preciso a instalação dos pacotes python: **numpy** e **matplotlib**

- Para **compilar** seu código no terminal:
  - gcc lab.c -o lab
- **-o** significa output. Ele é responsável por gerar o binário do seu programa para execução. É OBRIGATÓRIO que o arquivo tenha a função **main;**
- Logo, o que você está dizendo é: *compile o código **lab.c** com o compilador **gcc**, gerando o executável (saída) **lab**;*
- Para **executar** seu programa:
  - ./lab
- Você pode baixar os arquivos de casos de teste do run.codes e executá-los manualmente:
  - ./lab < 01.in
- A diretiva < redireciona o conteúdo do arquivo 01.in para o terminal, cujas entradas/dados serão lidos pelo scanf;
- Você pode ainda redirecionar a saída impressa no terminal para um arquivo:
  - ./lab < 01.in > 01.res

- Por fim, você poder comparar sua reposta com o gabarito (resultado do caso de teste), fazendo
  - diff 01.res 01.out
  - onde 01.out é a saída esperada para a entrada 01.in

6) Observações Gerais

- A nota é dada pelo **número de casos de teste acertados**;
- É obrigatório desalocar a lista corretamente. Caso contrário, pontos serão descontados.
- Códigos com **erros de compilação e execução**, tais como Segmentation Fault, **serão considerados errados**;
- Utilize ***return 0;*** na main de seu programa;
- Qualquer tentativa de fraude, plagio e afins, corresponderá em **nota ZERO** para os envolvidos;