AULA 08 - Exercício teórico Aprendizado de máquina

October 17, 2023

Aluno: Vitor Albuquerque de Paula

1 Aplique o KNN com K = 3 para classificar o 11o elemento da tabela. Considere a distância Euclidiana e Manhattan, compare e há diferença entre elas.

	A1	A2	Classe
1	0.5	1	2
2	2.9	1.9	2
3	1.2	3.1	2
4	0.8	4.7	2
5	2.7	5.4	2
6	8.1	4.7	1
7	8.3	6.6	1
8	6.3	6.7	1
9	8	9.1	1
10	5.4	8.4	1
11	5	7	?

Vamos aplicar o KNN com K=3 para classificar o 11º elemento da tabela. Primeiro, calculamos as distâncias Euclidiana e Manhattan entre o 11º elemento (5, 7) e todos os outros elementos da tabela.

Distâncias Euclidianas:

1. d(1, 11) = sqrt((0.5-5)^2 + (1-7)^2) = sqrt(4.5^2 + (-6)^2) = sqrt(20.25 + 36) = sqrt(56.25) 7.5
2. d(2, 11) = sqrt((2.9-5)^2 + (1.9-7)^2) = sqrt((-2.1)^2 + (-5.1)^2) = sqrt(4.41 + 26.01) = sqrt(30.42) 5.5
3. d(3, 11) = sqrt((1.2-5)^2 + (3.1-7)^2) = sqrt((-3.8)^2 + (-3.9)^2) = sqrt(14.44 + 15.21) = sqrt(29.65) 5.44
4. d(4, 11) = sqrt((0.8-5)^2 + (4.7-7)^2) = sqrt((-4.2)^2 + (-2.3)^2) = sqrt(17.64 + 5.29) = sqrt(22.93) 4.79
5. d(5, 11) = sqrt((2.7-5)^2 + (5.4-7)^2) = sqrt((-2.3)^2 + (-1.6)^2) = sqrt(5.29 + 2.56) = sqrt(7.85) 2.8
6. d(6, 11) = sqrt((8.1-5)^2 + (4.7-7)^2) = sqrt(3.1^2 + (-2.3)^2) = sqrt(9.61 + 5.29) = sqrt(14.9) 3.86

7. $d(7, 11) = \mathrm{sqrt}((8.3-5)^2 + (6.6-7)^2) = \mathrm{sqrt}(3.3^2 + (-0.4)^2) = \mathrm{sqrt}(10.89 + 0.16) = \mathrm{sqrt}(11.05)$ 3.32
8. $d(8, 11) = \mathrm{sqrt}((6.3-5)^2 + (6.7-7)^2) = \mathrm{sqrt}(1.3^2 + (-0.3)^2) = \mathrm{sqrt}(1.69 + 0.09) = \mathrm{sqrt}(1.78)$ 1.33
9. $d(9, 11) = \mathrm{sqrt}((8-5)^2 + (9.1-7)^2) = \mathrm{sqrt}(3^2 + 2.1^2) = \mathrm{sqrt}(9 + 4.41) = \mathrm{sqrt}(13.41)$ 3.66
10. $d(10, 11) = \mathrm{sqrt}((5.4-5)^2 + (8.4-7)^2) = \mathrm{sqrt}(0.4^2 + 1.4^2) = \mathrm{sqrt}(0.16 + 1.96) = \mathrm{sqrt}(2.12)$ 1.46

Distâncias de Manhattan (distância L1):

1. $d(1, 11) = |0.5-5| + |1-7| = 4.5 + 6 = 10.5$
2. $d(2, 11) = |2.9-5| + |1.9-7| = 2.1 + 5.1 = 7.2$
3. $d(3, 11) = |1.2-5| + |3.1-7| = 3.8 + 3.9 = 7.7$
4. $d(4, 11) = |0.8-5| + |4.7-7| = 4.2 + 2.3 = 6.5$
5. $d(5, 11) = |2.7-5| + |5.4-7| = 2.3 + 1.6 = 3.9$
6. $d(6, 11) = |8.1-5| + |4.7-7| = 3.1 + 2.3 = 5.4$
7. $d(7, 11) = |8.3-5| + |6.6-7| = 3.3 + 0.4 = 3.7$
8. $d(8, 11) = |6.3-5| + |6.7-7| = 1.3 + 0.3 = 1.6$
9. $d(9, 11) = |8-5| + |9.1-7| = 3 + 2.1 = 5.1$
10. $d(10, 11) = |5.4-5| + |8.4-7| = 0.4 + 1.4 = 1.8$

Agora vamos pegar os 3 vizinhos mais próximos para cada métrica de distância.

Métrica Euclidiana:

- Elemento 8 (distância 1.33)
- Elemento 10 (distância 1.46)
- Elemento 5 (distância 2.8)

Métrica de Manhattan:

- Elemento 8 (distância 1.6)
- Elemento 10 (distância 1.8)
- Elemento 5 (distância 3.9)

Para ambas as métricas, os 3 vizinhos mais próximos são os mesmos e têm as seguintes classes: {1, 1, 2}. Como a classe 1 aparece duas vezes e a classe 2 aparece apenas uma vez, a classe do elemento 11 é 1.

Então, aplicando o KNN com K=3, a classe do elemento 11 é 1, para ambas as distâncias Euclidiana e Manhattan. Não há diferença entre os resultados obtidos com as duas métricas neste caso.

2 Aplique o algoritmo de Bayes no problema a seguir:

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	no	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
glia monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	no	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	no	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	???

Vamos aplicar o algoritmo de Bayes para classificar o animal desconhecido com as características: dá à luz (yes), não voa (no), vive na água (yes) e não tem pernas (no).

Primeiro, contamos a frequência de cada classe e suas características no conjunto de dados.

Mamíferos (M) : 7 Não mamíferos (NM) : 13

Característica	M (yes)	M (no)	NM (yes)	NM (no)
Give Birth	6	1	1	12
Can Fly	1	6	3	10
Live in Water	2	5	8	6
Have Legs	6	1	10	3

Agora, vamos calcular as probabilidades das características dadas a classe: P(Característica | Classe).

Característica	P(Característica M)	P(Característica NM)
Give Birth	6/7 0.86	1/13 0.08
Can Fly	1/7 0.14	3/13 0.23
Live in Water	2/7 0.29	8/13 0.62
Have Legs	6/7 0.86	10/13 0.77

Agora podemos calcular as probabilidades para cada classe: P(M) e P(NM).

P(M) = 7/20 = 0.35 P(NM) = 13/20 = 0.65

Agora vamos aplicar o teorema de Bayes:

P(M Características) proporcional a P(Give Birth M) * P(Can Fly M) * P(Live in Water M) * P(Have Legs M) * P(M)

P(M Características) proporcional a 0.86 * 0.14 * 0.29 * 0.14 * 0.35   0.0011

P(NM Características) proporcional a P(Give Birth NM) * P(Can Fly NM) * P(Live in Water NM) * P(Have Legs NM) * P(NM)

P(NM Características) proporcional a 0.08 * 0.23 * 0.62 * 0.77 * 0.65   0.0076

Podemos normalizar esses valores pela soma das duas probabilidades para obter a probabilidade final.

P(M Características) = 0.0011 / (0.0011 + 0.0076)   0.126

P(NM Características) = 0.0076 / (0.0011 + 0.0076)   0.874

Como P(NM Características) é maior do que P(M Características), o algoritmo de Bayes classifica o animal desconhecido como "não mamífero".

3 Execute árvores de decisão:

3.1 a AND b

Uma árvore de decisão para a operação AND pode ser representada da seguinte maneira:

[ ]:
```
      [a]
     /   \
   [b]    [0]
  /   \
[1]  [0]
```

Nesta árvore, primeiro verificamos o valor de "a". Se "a" for falso (0), a resposta é "0". Se "a" for verdadeiro (1), passamos para o nó seguinte e verificamos o valor de "b". Se "b" for verdadeiro (1), a resposta é "1"; caso contrário, a resposta é "0".

3.2 a XOR b

Uma árvore de decisão para a operação XOR pode ser representada da seguinte maneira:

[ ]:
```
       [a]
      /   \
    [b]    [b]
   /   \ /   \
 [0]  [1]  [1]  [0]
```

Nesta árvore, primeiro verificamos o valor de "a". Se "a" for falso (0), passamos para o nó à esquerda e verificamos o valor de "b". Nesse caso, a saída é igual a "b". Se "a" for verdadeiro (1), passamos para o nó à direita e verificamos o valor de "b". Nesse caso, a saída é o inverso de "b" (se "b" for verdadeiro, a saída é falsa, e se "b" for falso, a saída é verdadeira).

3.3 (a AND b) OR (b AND c)

Uma árvore de decisão para a operação (a AND b) OR (b AND c) pode ser representada da seguinte maneira:

[ ]:
```
        [b]
       /   \
     [a]    [c]
    /   \ /   \
 [1] [1] [1] [0]
```

Nesta árvore, primeiro verificamos o valor de "b". Se "b" for falso (0), a resposta é "0". Se "b" for verdadeiro (1), passamos para os nós à esquerda e à direita. No nó à esquerda, verificamos se "a" é verdadeiro, e no nó à direita, verificamos se "c" é verdadeiro. Se pelo menos um deles for verdadeiro, a resposta é "1"; caso contrário, a resposta é "0".

4 Calcular a medida de entropia para os dados abaixo:

C1	C2	E=?
0	6	
1	5	
2	4	
3	3	

A entropia pode ser calculada pela seguinte fórmula:

E(S) = - p1 * log2(p1) - p2 * log2(p2)

Onde p1 e p2 são as probabilidades das duas classes (C1 e C2, neste caso).

Para cada conjunto de dados:

4.1 Conjunto 1:

C1: 0 C2: 6 Total: 6

p(C1) = 0/6 = 0 p(C2) = 6/6 = 1

E(S) = -0 * log2(0) - 1 * log2(1) Dado que log2(1) = 0 e o logaritmo de 0 é indefinido, a entropia se torna 0.

E(S) = 0

4.2 Conjunto 2:

C1: 1 C2: 5 Total: 6

p(C1) = 1/6 p(C2) = 5/6

E(S) = -1/6 * log2(1/6) - 5/6 * log2(5/6) E(S) 0.65

4.3 Conjunto 3:

C1: 2 C2: 4 Total: 6

p(C1) = 2/6 = 1/3 p(C2) = 4/6 = 2/3

E(S) = -1/3 * log2(1/3) - 2/3 * log2(2/3) E(S) 0.92

4.4 Conjunto 4:

C1: 3 C2: 3 Total: 6

p(C1) = 3/6 = 1/2 p(C2) = 3/6 = 1/2

E(S) = -1/2 * log2(1/2) - 1/2 * log2(1/2) E(S) = 1

4.5 Resumo dos resultados:

1. E = 0
2. E 0.65
3. E 0.92
4. E = 1

5 Pesquise as principais diferenças entre os algoritmos de árvores de decisão:

- **Hunt**
- **ID3**
- **C4.5**
- **J4.8**
- **C5.0**
- **CART**
- **Random-Forest**

Aqui estão as principais diferenças entre os algoritmos de árvores de decisão especificados:

5.1 Hunt:

- Constrói uma árvore de decisão de forma recursiva, partindo o conjunto de dados de treinamento em subconjuntos sucessivamente mais puros[1].
- Desenvolvido na década de 1960 para modelar o aprendizado humano em Psicologia, serve como base para muitos algoritmos populares de árvores de decisão, incluindo o ID3[2].
- É a base de muitos algoritmos de indução de árvores de decisão existentes, incluindo ID3, C4.5 e CART[3].

5.2 ID3 (Iterative Dichotomiser 3):

- Um dos primeiros algoritmos de árvores de decisão, desenvolvido por Ross Quinlan.
- Utiliza Entropia e Ganho de Informação para seleção de atributos em cada nó[4].

5.3 C4.5:

- Extensão do algoritmo ID3 criada por Ross Quinlan.
- Utiliza a razão de ganho para a seleção de atributos, além de lidar com valores de atributos contínuos e dados ausentes.
- Implementa métodos de poda para evitar o sobreajuste.

5.4 J4.8:

- Implementação do algoritmo C4.5 na linguagem Java.
- Permite a classificação por meio de árvores de decisão ou regras geradas a partir delas, construindo árvores de decisão baseadas em um conjunto de dados de treinamento da mesma forma que o ID3[5].
- Implementa uma versão mais recente e ligeiramente melhorada chamada C4.5 revisão 8[6].

5.5 C5.0:

- Versão comercial e mais rápida do algoritmo C4.5, também desenvolvida por Ross Quinlan.
- Oferece melhorias em termos de velocidade, memória e usabilidade em relação ao C4.5[4].

5.6 CART (Classification and Regression Trees):

- Podendo ser utilizado tanto para classificação quanto para regressão.
- Utiliza o índice Gini ou a redução na variância para a seleção de atributos, diferentemente do ID3 e C4.5 que usam medidas baseadas em entropia[4].

5.7 Random Forest:

- Método de aprendizado em conjunto para classificação, regressão e outras tarefas que opera construindo uma multidão de árvores de decisão durante o tempo de treinamento.
- Combina a saída de múltiplas árvores de decisão para alcançar um único resultado, melhorando a precisão e a estabilidade das previsões[8].

5.8 Links das Fontes:

- Bookdown[1]
- IBM (Hunt)[2]
- AI from scratch (Hunt)[3]
- PUC-Rio[4]
- Medium (J4.8)[5]
- ScienceDirect Topics (J4.8)[6]
- Built In (Random Forest)[8]