

Prof. Dr. Agnès Voisard
Nicolas Lehmann

Datenbanksysteme, SoSe 18

Übung 014

TutorIn: Toni Draßdo
Tutorium 04

Eduard Beiline, Mark Niehues, Antoen Oehler

20. Mai 2018

Task 1: ER-Modellierung

Task 2: Relationales Modell

Task 3: Reverse Engineering

Task 4: Data Mining

1 - K-Means

Center for this Cluster: [6.66666667 9.66666667 1.66666667]

Contains:

[3 10 1]

[10 10 3]

[7 9 1]

Center for this Cluster: [7.33333333 3.66666667 1.66666667]

Contains:

[9 2 3]

[9 4 1]

[4 5 1]

Center for this Cluster: [5. 4.5 6.5]

Contains:

[6 6 8]

[4 3 5]

Abbildung 1: Ergebnis des K-Means Algorithmus

Listing 1: K-Means Implementierung

```

1  #!/usr/bin/env python3
2  # coding: utf-8
3  import numpy as np
4  import sys

6  DATA = np.array([
7      [3,9,9,10,6,7,4,4],
8      [10,2,4,10,6,9,5,3],
9      [1,3,1,3,8,1,1,5]
10 ]).T

13 def expectation(data, centers):
14     """
15     Assigns datapoints to centers
16     """
17     clusters = [[] for _ in range(centers.shape[0])]

19     for point in data:
20         # Calculate distances from centers
21         d = np.linalg.norm(centers - point, axis=1)

23         # Find the index according to the lowest distance
24         id_min = np.argmin(d)

26         # Assigning point to minimum distance
27         clusters[id_min].append(point)

29     for clt in clusters:
30         if len(clt) == 0:
31             print("Error: Empty cluster occurred, please re-run the program.")
32             sys.exit(1)
33     return clusters

36 def minimization(clusters):
37     """
38     Computes new cluster means
39     """
40     centers = [np.mean(cls, axis=0) for cls in clusters]

42     return np.vstack(centers)

45 def k_means(data, k, sigma):
46     # Initialize with k random points
47     centers = data[np.random.randint(data.shape[0], size=k)]

49     dist = sigma + 1
50     while dist > sigma:
51         # Assign data points to centers
52         clusters = expectation(data, centers)

54         # Calculate new centers
55         new_centers = minimization(clusters)

57         # Calc maximum center movement
58         dist = max(np.linalg.norm(centers - new_centers, axis=1))
59         centers = new_centers

61     return centers, clusters

64 def print_clusters(centers, clusters):
65     for center, clst in zip(centers, clusters):
66         print("Center for this Cluster: {}".format(center))
67         print("Contains:")

```

```

68     for _ in clst:
69         print(_)
70         print()

73 centers, clusters = k_means(DATA, k=3, sigma=3/4)
74 print_clusters(centers, clusters)

```

2 - Naive Bayes

1 - Wahrscheinlichkeit einer Grippe bei laufender Nase

Naive Bayes Formula:

$$P(C|x) = \frac{P(C) P(x|C)}{P(x)} \quad (1)$$

Aus Formel 1 folgt für die Wahrscheinlichkeit an einer Grippe zu leiden, bei laufender Nase:

$$P(Grippe|Nase) = \frac{P(Gripp) P(Nase|Grippe)}{P(Nase)} \quad (2)$$

wobei:

$$\begin{aligned}
 P(Nase) &= 4/8 = 1/2 \\
 P(Grippe) &= 1/2 \\
 P(Nase|Grippe) &= 3/4
 \end{aligned}$$

Durch einsetzen in Formel 2 erhält man:

$$P(Grippe|Nase) = 3/4$$

2 - Grippe, wenn X

Um die Frage zu beantworten, ob jemand eher Grippe oder keine Grippe besitzt wird, um die Rechnung zu Vereinfachen der Quotient aus $P(Grippe|x)$ und $P(\neg Grippe|x)$ gebildet, dadurch kürzt sich die aufwändig zu berechnende Evidenz $P(x)$ heraus:

$$Q = \frac{P(Grippe|x)}{P(\neg Grippe|x)} = \frac{P(x|Grippe) P(Grippe)}{P(x|\neg Grippe) P(\neg Grippe)} \quad (3)$$

wobei:

$$\begin{aligned}
x &= \{Schttelfrost, schwacheKopfschmerzen, Fieber\} \\
P(Schttelfrost|Grippe) &= 3/4 \\
P(Schttelfrost|\neg Grippe) &= 1/2 \\
P(schwacheKopfschmerzen|Grippe) &= 1/4 \\
P(schwacheKopfschmerzen|\neg Grippe) &= 1/4 \\
P(Fieber|Grippe) &= 1/2 \\
P(Fieber|\neg Grippe) &= 1/2 \\
P(Grippe) &= P(\neg Grippe) = 1/2
\end{aligned}$$

Unter Annahme der (hinreichenden) Unabhängigkeit der Variablen, gilt $P(x_1, x_2|C) = P(x_1|C) P(x_2|C)$.

Daraus ergibt sich schließlich:

$$Q = \frac{\frac{3}{4} \frac{1}{4} \frac{1}{2}}{\frac{1}{2} \frac{1}{4} \frac{1}{2}} = 3/2$$

Aus der $Q > 1$ folgt, dass der Patient wahrscheinlicher Grippe hat als keine.

3 - Apriori

Listing 2: Ergebnis des Apriori Algorithmus unter Zuhilfenahme der Implementierung aus [1].

```

1 item: ('F',) , 0.500
2 item: ('A',) , 0.500
3 item: ('C',) , 0.500
4 item: ('D', 'F') , 0.500
5 item: ('A', 'E') , 0.500
6 item: ('C', 'B') , 0.500
7 item: ('C', 'D') , 0.500
8 item: ('A', 'B') , 0.500
9 item: ('E', 'F') , 0.500
10 item: ('C', 'B', 'D') , 0.500
11 item: ('E', 'D', 'F') , 0.500
12 item: ('A', 'B', 'E') , 0.500
13 item: ('B', 'E', 'D') , 0.500
14 item: ('B', 'D') , 0.667
15 item: ('B', 'E') , 0.667
16 item: ('E', 'D') , 0.667
17 item: ('D',) , 0.833
18 item: ('B',) , 0.833
19 item: ('E',) , 0.833

21 ----- RULES:
22 Rule: ('B', 'D') ==> ('C',) , 0.750
23 Rule: ('E', 'D') ==> ('F',) , 0.750
24 Rule: ('B', 'E') ==> ('A',) , 0.750
25 Rule: ('B', 'E') ==> ('D',) , 0.750
26 Rule: ('B', 'D') ==> ('E',) , 0.750
27 Rule: ('E', 'D') ==> ('B',) , 0.750
28 Rule: ('B',) ==> ('D',) , 0.800
29 Rule: ('D',) ==> ('B',) , 0.800
30 Rule: ('B',) ==> ('E',) , 0.800
31 Rule: ('E',) ==> ('B',) , 0.800
32 Rule: ('E',) ==> ('D',) , 0.800
33 Rule: ('D',) ==> ('E',) , 0.800
34 Rule: ('F',) ==> ('D',) , 1.000
35 Rule: ('A',) ==> ('E',) , 1.000
36 Rule: ('C',) ==> ('B',) , 1.000

```

```
37 Rule: ('C',) ==> ('D',) , 1.000
38 Rule: ('A',) ==> ('B',) , 1.000
39 Rule: ('F',) ==> ('E',) , 1.000
40 Rule: ('C',) ==> ('B', 'D') , 1.000
41 Rule: ('C', 'B') ==> ('D',) , 1.000
42 Rule: ('C', 'D') ==> ('B',) , 1.000
43 Rule: ('F',) ==> ('E', 'D') , 1.000
44 Rule: ('E', 'F') ==> ('D',) , 1.000
45 Rule: ('D', 'F') ==> ('E',) , 1.000
46 Rule: ('A',) ==> ('B', 'E') , 1.000
47 Rule: ('A', 'B') ==> ('E',) , 1.000
48 Rule: ('A', 'E') ==> ('B',) , 1.000
```

4 - Lineare Regression

Leider ist bei uns erst eine Korrektur eingetragen, deswegen können wir keine lineare Regression durchführen (unterbestimmtes System).

Literatur

- [1] asaini. *Python Implementation of Apriori Algorithm*. <https://github.com/asaini/Apriori.git>. 2017.