



KECERDASAN
BUATAN

Cardiovascular Prediction with Decision Tree Classification

Group Assignment Kelompok 14



ANGGOTA KELOMPOK



01

Adinda Luthfiah Sya'bani
20/463587/TK/51579

02

Anisya Mahira Salienka
20/463593/TK/51585

03

Auletta Khansa Pradiviasari
20/456359/TK/50489

04

Wardatul Radhiyyah
20/456381/TK/50511

INTRODUCTION

Penyakit kardiovaskuler adalah salah satu penyakit yang angkanya selalu meningkat tiap tahunnya. Bahkan, penyakit ini sempat menempati peringkat tertinggi sebagai penyebab kematian di Indonesia terutama pada usia produktif. Berdasarkan data Rikesdas 2018, penduduk pada area perkotaan memiliki potensi lebih banyak menderita penyakit kardiovaskuler khususnya penyakit jantung dengan prevalensi 1.6%. Hal ini disebabkan oleh perubahan gaya hidup yang kurang sehat seperti pola makan yang tidak seimbang. Dalam masa pandemi seperti sekarang, orang yang memiliki komorbid penyakit kardiovaskuler akan lebih rentan terpapar covid-19 yang akan semakin memperburuk kesehatan atau bahkan hingga menyebabkan kematian pada penderita.

Untuk itu, diperlukan deteksi dini terhadap gejala - gejala yang mengarah kepada penyakit kardiovaskuler khususnya pada jantung koroner. Dengan menyusun program untuk memprediksi gejala dari penyakit jantung koroner dengan metode decision tree diharapkan mampu membantu mendeteksi penyakit tersebut sedini mungkin.



PROBLEM (Dataset Detail)

1	sbp	tobacco	ldl	adiposity	typea	obesity	alcohol	age	chd
2	160	12.00	5.73	23.11		49 25.30	97.20	52	1
3	144	0.01	4.41	28.61		55 28.87	2.06	63	1
4	118	0.08	3.48	32.28		52 29.14	3.81	46	0
5	170	7.50	6.41	38.03		51 31.99	24.26	58	1
6	134	13.60	3.50	27.78		60 25.99	57.34	49	1
7	132	6.20	6.47	36.21		62 30.77	14.14	45	0
8	142	4.05	3.38	16.20		59 20.81	2.62	38	0
9	114	4.08	4.59	14.60		62 23.11	6.72	58	1
10	114	0.00	3.83	19.40		49 24.86	2.49	29	0
11	132	0.00	5.80	30.96		69 30.11	0.00	53	1
12	206	6.00	2.95	32.27		72 26.81	56.06	60	1
13	134	14.10	4.44	22.39		65 23.09	0.00	40	1
14	118	0.00	1.88	10.05		59 21.57	0.00	17	0
15	132	0.00	1.87	17.21		49 23.63	0.97	15	0
16	112	9.65	2.29	17.20		54 23.53	0.68	53	0
17	117	1.53	2.44	28.95		35 25.89	30.03	46	0
18	120	7.50	15.33	22.00		60 25.31	34.49	49	0
19	146	10.50	8.29	35.36		78 32.73	13.89	53	1
20	158	2.60	7.46	34.07		61 29.30	53.28	62	1

Dataset ini kami dapatkan dari kaggle.com dengan "judul Cardiovascular Disease". pada sumber ini terdapat 463 data, tetapi kelompok kami hanya memakai 100 data.

- sbp : systolic blood pressure (tekanan darah sistolik)
- tobacco : tingkat konsumsi tembakau (kg)
- ldl : tingkat kolesterol LDL
- adiposity : kelebihan lemak pada tubuh
- age : umur
- typea : type a behaviour
- obesity : tingkat obesitas
- alcohol : tingkat konsumsi alkohol
- chd (coronary heart disease) : hasil prediksi/response penyakit jantung koroner, 1: iya 0: tidak



OBJECTIVE

Tujuan analisis dataset dan penggunaan metode decision tree

Tujuan dari analisis dataset ini adalah untuk melakukan prediksi dini terhadap gejala-gejala jantung koroner berdasarkan dataset yang sudah disediakan dengan jumlah kurang lebih ada 100 sample. Jika memang seseorang mengidap penyakit tersebut berdasarkan program yang dibuat maka diharapkan orang tersebut melakukan pemeriksaan kesehatan lebih lanjut melalui fasilitas kesehatan terdekat dan segera berkonsultasi dengan dokter atau tenaga medis lainnya.

Penggunaan metode decision tree sendiri dilakukan untuk memprediksi apakah seseorang mengidap jantung koroner melalui berbagai gejala-gejala umum. Selain itu, decision tree juga memudahkan kita untuk memetakan gejala-gejala dari jantung koroner mulai dari gejala umum hingga spesifik sehingga hasil yang diperoleh pun lebih akurat.

METHOD

Decision Tree

Decision Tree merupakan model prediksi dengan menggunakan struktur pohon atau struktur berhirarki. Metode ini digunakan sebagai alat pendukung untuk memodelkan hasil, utilitas, dan kemungkinan konsekuensi dengan menyajikan algoritma dan pernyataan kontrol bersyarat. Strukturnya sendiri memiliki tiga tingkatan yaitu root node, node, dan leaf dengan root node sebagai titik awal dari decision tree lalu kemudian diikuti node-node (atribut) dan leaf yang memuat keputusan akhir.

Implementasi

Pada implementasinya dari dataset yang ada dengan menggunakan decision tree method akan menghasilkan decision tree dan kemudian dengan sample data yang ada akan mengeluarkan output predikisi berdasarkan decision tree

ALUR PROGRAM

```
] #Membentuk decision tree berdasarkan dataset
myDecisionTree = createDecisionTree(df, df, attributes, label, parent)

#Fungsi Main menggunakan trained model
sampleData = {'sbp':168,'tobaco':4.58,'ldl':6.68,'adiposity':28.47,'typea':43,'obesity':24.25,'alcohol':24.38,'age':56}
testData = pd.Series(sampleData)
prediction = predictionDisease(testData, myDecisionTree)

if(prediction == 1.0):
    print("Hasil prediksi : Kamu menderita penyakit cardiovaskular")
else:
    print("Hasil prediksi : Kamu tidak menderita penyakit cardiovaskular")

pprintTree (myDecisionTree)
```

- Membuat/input dataset
- Membuat decision tree dari nilai dataset dan attributes dataset
- Memasukkan sample data
- Membuat prediksi dari sample data berdasarkan decision tree
- Mengeluarkan prediksi

PROGRAM

Fungsi Membuat Decision Tree

```
❶ #Membuat decision tree
def createDecisionTree(dataset, df, attributes, label, parent):

    generalData = np.unique(df[label], return_counts = True)
    uniqueData = np.unique(dataset[label])

    if len(uniqueData) <= 1:
        return uniqueData[0]
    elif len(dataset) == 0:
        return uniqueData[np.argmax(generalData[1])]
    elif len(attributes) == 0:
        return parent
    else:
        parent = uniqueData[np.argmax(generalData[1])]

        item_values = [calcInfoGain(dataset, attribute, label) for attribute in attributes]

        maxAttributeIndex = np.argmax(item_values)
        maxAttribute = attributes[maxAttributeIndex]
        myDecisionTree = {maxAttribute:{}}

        attributes = [i for i in attributes if i != maxAttribute]

        for value in np.unique(dataset[maxAttribute]):
            minData = dataset.where(dataset[maxAttribute] == value).dropna()

            minTree = createDecisionTree(minData, df, attributes, label, parent)

            myDecisionTree[maxAttribute][value] = minTree

    return myDecisionTree
```

Fungsi Menghitung Nilai Entropi

```
#Menghitung nilai entropy
def calcEntropy(dfLabel):
    classes, classCount = np.unique(dfLabel, return_counts = True)

    entropy = np.sum([(-classCount[i] / np.sum(classCount)) * np.log2(classCount[i] / np.sum(classCount))
                     for i in range(len(classes))])

    return entropy
```

$$Entropy = \sum_{i=1}^C -p_i * \log_2(p_i)$$

Fungsi Menghitung Nilai Information Gain

```
#Menghitung nilai information gain
def calcInfoGain(dataset, attribute, label):
    datasetEntropy = calcEntropy(dataset[label])
    values, attributeCount = np.unique(dataset[attribute], return_counts = True)

    maxAttributeEntropy = np.sum([(attributeCount[i] / np.sum(attributeCount)) * calcEntropy(dataset.where(dataset[attribute] == values[i]).dropna()[label])
                                    for i in range(len(values))])

    InfoGain = datasetEntropy - maxAttributeEntropy

    return InfoGain
```

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

Fungsi Membuat Prediksi Berdasarkan Decision Tree

```
#Melakukan prediksi terhadap data yang ada
def predictionDisease(testData, myDecisionTree):

    for nodes in myDecisionTree.keys():
        value = testData[nodes]
        myDecisionTree = myDecisionTree[nodes][value]

        prediction = 0
        if type(myDecisionTree) is dict:
            prediction = predictionDisease(testData, myDecisionTree)
        else:
            prediction = myDecisionTree
            break;

    return prediction
```

Decision Tree

- Menghitung nilai entropi
- Menghitung information gain
- Menentukan root/node selanjutnya dengan nilai information gain yang terbesar

RESULT

Input : Dataset cardiovascular

```
[ ] #Input file dataset yang akan digunakan (format .csv)
from google.colab import files
data_to_load = files.upload()

[ ] Tidak ada file yang dipilih Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Cardiovascular.csv to Cardiovascular.csv

[ ] #Input library yang dibutuhkan
from math import log
import pandas as pd
import numpy as np
import csv
import pprint
import json

[ ] filename = "Cardiovascular.csv"

[ ] #Dataset yang akan digunakan
df = pd.read_csv('Cardiovascular.csv')
df.head()

      sbp  tobacco  ldl  adiposity  typea  obesity  alcohol  age  chd
0    160     12.00   5.73     23.11    49    25.30    97.20    52     1
1    144      0.01   4.41     28.61    55    28.87    2.06    63     1
2    118      0.08   3.48     32.28    52    29.14    3.81    46     0
3    170      7.50   6.41     38.03    51    31.99    24.26    58     1
4    134     13.80   3.50     27.78    60    25.99    57.34    49     1

[ ] #Inisialisasi
attributes = df.columns[:-1]
label = 'chd'
parent = None
```

- Membuat dataset dengan membaca file dataset .csv
- Menyimpan nilai dataset serta atribut yang ada pada dataset file .csv pada variabel

Output : Hasil Prediksi Berdasarkan Sample Data

```
Hasil prediksi : Kamu menderita penyakit cardiovaskular

ldl:
| 1.07: 0.0
| 1.55: 1.0
| 1.74: 1.0
| 1.8: 0.0
| 1.82: 0.0
| 1.87: 0.0
| 1.88: 0.0
| 1.94: 0.0
| 2.19: 0.0
| 2.29: 0.0
| 2.4: 0.0
| 2.43: 0.0
| 2.44: 0.0
| 2.66: 0.0
| 2.72: 1.0
| 2.82: 0.0
| 2.83: 0.0
| 2.85: 0.0
| 2.95: 1.0
| 2.96: 1.0
| 2.99: 0.0
| 3.24: 0.0
| 3.3: 0.0
| 3.31: 0.0
| 3.37:
|   sbp:
|   | 122.0: 1.0
|   | 130.0: 0.0
|
| 3.38: 0.0
| 3.48: 0.0
| 3.5: 1.0
```

- Memunculkan hasil decision tree
- Memunculkan hasil prediksi berdasarkan input sample data

≡

Terima kasih!