Laporan Tugas Pemrograman 03
Learning

Disusun Oleh :
1301200428 - Muhammad Rafiqi Masrur
1301200457 - Rasyid Riyaldi

IF-44-01

Program Studi S1 Informatika
Fakultas Informatika
Telkom University
2022

I. Rumusan Masalah

1. Deskripsi Tugas

Diberikan file traintest.xlsx yang terdiri dari dua sheet: train dan test, yang berisi dataset untuk problem klasifikasi biner (binary classification). Setiap record atau baris data dalam dataset tersebut secara umum terdiri dari nomor baris data (*id*), fitur input ($x1$ sampai $x3$), dan output kelas (y). Fitur input terdiri dari nilai-nilai integer dalam range tertentu untuk setiap fitur. Sedangkan output kelas bernilai biner (0 atau 1).

Sheet train berisi 296 baris data, lengkap dengan target output kelas (y). Gunakan sheet ini untuk tahap pemodelan atau pelatihan (training) model sesuai metode yang digunakan. Adapun sheet test berisi 10 baris data, dengan output kelas (y) yang disembunyikan. Gunakan sheet ini untuk tahap pengujian (testing) model yang sudah dilatih. Nantinya output program untuk data uji ini akan dicocokkan dengan target atau kelas sesungguhnya.

Berikut adalah metode yang dapat digunakan :

- Decision Tree (ID 3)
- KNN
- Naive Bayes

Metode yang kami pilih untuk menyelesaikan tugas ini adalah metode Naive Bayes.

Berikut adalah proses yang harus diimplementasikan ke dalam program :

- Membaca data latih/uji
- Pelatihan atau training model
- Menyimpan model hasil training
- Pengujian atau testing model
- Evaluasi model
- Menyimpan output ke file

2. Output Program

Program secara umum memiliki dua tahap: pelatihan (training) dan pengujian (testing). Pada tahap training, akan dihasilkan output berupa model sesuai metode yang digunakan. Sedangkan pada tahap testing, dihasilkan output berupa kelas (0 atau 1); Lebih jauh lagi, jika ada lebih dari satu record/baris sebagai input data untuk tahap testing, maka program dapat mengeluarkan list output kelas yang bersesuaian dengan setiap baris data testing tersebut.

II. Kajian Pustaka

1. Pengertian AI

Kecerdasan buatan atau Artificial Intelligence (AI) adalah simulasi dari kecerdasan yang dimiliki oleh manusia yang dimodelkan di dalam mesin dan diprogram agar bisa berpikir seperti halnya manusia. Sedangkan menurut McLeod dan Schell, kecerdasan buatan adalah aktivitas penyediaan mesin seperti komputer dengan kemampuan untuk menampilkan perilaku yang dianggap sama cerdasnya dengan jika kemampuan tersebut ditampilkan oleh manusia.

2. Pengertian Metode Naive Bayes

Naïve Bayes Classifier merupakan sebuah metode klasifikasi yang berakar pada teorema Bayes. Metode pengklasifikasian dg menggunakan metode probabilitas dan statistik yg dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya sehingga dikenal sebagai Teorema Bayes. Ciri utama dr Naïve Bayes Classifier ini adalah asumsi yg sangat kuat (naïf) akan independensi dari masing-masing kondisi/kejadian.

$$P(H|X) = \frac{P(X|H)\,P(H)}{P(X)}$$

Di mana X adalah bukti, H adalah hipotesis, P(H|X) adalah probabilitas posterior H dengan syarat X, P(X|H) adalah probabilitas posterior X dengan syarat H, P(H) adalah probabilitas prior hipotesis H, dan P(X) adalah probabilitas prior bukti X.

Dalam *machine learning*, X adalah sebuah tuple atau objek data, H adalah hipotesis bahwa tuple X berada di kelas C. Pada masalah klasifikasi, P(H|X) adalah probabilitas bahwa hipotesis H benar untuk tuple X. Dengan kata lain, P(H|X) adalah probabilitas bahwa tuple X berada di kelas C. Sementara itu, P(H) adalah probabilitas prior bahwa hipotesis H benar untuk setiap tuple, tidak peduli nilai-nilai atributnya. Sedangkan P(X) adalah probabilitas prior dari tuple X.

III. Pembahasan

Cara Kerja Naive Bayes

Lima langkah Naive Bayes Classifier :

1. Misalkan D adalah himpunan data latih (*training set*) yang berisi sejumlah tuple beserta label kelasnya. Setiap tuple berdimensi n yang dinyatakan sebagai $X = (x_1, x_2, \ldots, x_n)$ yang didapat dari n atribut $A_1, A_2, \ldots, A_n$

2. Misalkan terdapat m kelas, yaitu $C_1, C_2, \ldots, C_m$. Untuk sebuah tuple masukan X, *Naïve Bayes classifier* memprediksi bahwa tuple X termasuk ke dalam kelas C_i jika dan hanya jika $P(C_i|X) > P(C_j|X)$ untuk. Dengan teorema Bayes, $P(C_i|X)$ diestimasi menggunakan rumus,
$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

3. Mengingat $P(X)$ bernilai sama untuk semua kelas, maka hanya $P(X|C_i)$ kali $P(C_i)$ yang perlu dimaksimalkan. Jika probabilitas *prior* untuk setiap kelas tidak diketahui, maka probabilitas setiap kelas biasanya diasumsikan sama, yaitu $P(C_1) = P(C_2)$ dan seterusnya hingga $P(C_m)$. Dengan demikian, Naive Bayes hanya memaksimalkan $P(X|C_i)$.

4. Jika berhadapan dengan himpunan data yang memiliki sangat banyak atribut, kompleksitas perhitungan dapat direduksi, dengan asumsi naif tentang independensi bersyarat kelas, yaitu: "nilai-nilai atribut saling independen". Jadi, Naïve Bayes memaksimalkan $P(X|C_i)$, untuk semua atribut independen tersebut.
$$P(C_i|X) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

   Untuk atribut yang bernilai kategorial, $P(X_k|C_i)$ didefinisikan sebagai jumlah tuple di kelas C_i dalam himpunan data D, yang memiliki nilai x_k pada atribut A_k, dibagi dengan jumlah semua tuple di kelas C_i. Sementara itu, untuk atribut yang bernilai kontinu, $P(X_k|C_i)$ diestimasi dengan fungsi kepadatan peluang Gaussian.
$$P(x_k|C_i) = \frac{1}{\sigma_{i,k}\sqrt{2\pi}} e^{-\frac{(x_k - \mu_{i,k})^2}{2\sigma_{i,k}^2}}$$

5. Untuk memprediksi label kelas dari tuple X, probabilitas $P(X|C_i)$ kali $P(C_i)$ untuk setiap kelas C_i harus dihitung. Selanjutnya, probabilitas tersebut hanya perlu dimaksimalkan. Secara matematis, *tuple* X diberi label kelas C_i jika dan hanya jika,
$$P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \text{ untuk } 1 \le j \le m, j \ne i$$

IV. Source Code

1. Import data dari file excel

```
from openpyxl import load_workbook

## Load Workbook
## Change load_workbook("File Location")
book = load_workbook("./traintest.xlsx")
trainSheet = book['train']
testSheet = book['test']
Phi = 3.14259
EULER = 2.71828

class data():
    def __init__(self, id, x1, x2, x3, y):
        self.id = id
        self.x1 = x1
        self.x2 = x2
        self.x3 = x3
        self.y = y

## Get data for train
def get_DataTrain():
    dataList = []
    i = 2
    while trainSheet["A"+str(i)].value != None:
        dataList.append(data(trainSheet["A"+str(i)].value, trainSheet["B"+str(i)].value, trainSheet["C"+str(i)].value, trainSheet["D"+str(i)].value, trainSheet["E"+str(i)].value))
        i += 1
    return dataList

## Get data for test
def get_DataTest():
    dataList = []
    i = 2
    while testSheet["A"+str(i)].value != None:
        dataList.append(data(testSheet["A"+str(i)].value, testSheet["B"+str(i)].value, testSheet["C"+str(i)].value, testSheet["D"+str(i)].value, testSheet["E"+str(i)].value))
        i += 1
    return dataList
```

2. Menghitung nilai rata-rata dari setiap data

```
## Calculate average value of data
def get_average(dataList):
    i = 0
    ptTrue = 0
    ptFalse = 0
    avgTrue = data(-1,0,0,0,-1)
    avgFalse = data(-1,0,0,0,-1)
    while i != len(dataList):
        if dataList[i].y == 1:
            avgTrue.x1 += dataList[i].x1
            avgTrue.x2 += dataList[i].x2
            avgTrue.x3 += dataList[i].x3
            ptTrue += 1
        else:
            avgFalse.x1 += dataList[i].x1
            avgFalse.x2 += dataList[i].x2
            avgFalse.x3 += dataList[i].x3
            ptFalse += 1
        i += 1

    avgTrue.x1 = avgTrue.x1 / ptTrue
    avgTrue.x2 = avgTrue.x2 / ptTrue
    avgTrue.x3 = avgTrue.x3 / ptTrue
    avgFalse.x1 = avgFalse.x1 / ptFalse
    avgFalse.x2 = avgFalse.x2 / ptFalse
    avgFalse.x3 = avgFalse.x3 / ptFalse

    return avgTrue, avgFalse
```

3. Menghitung nilai simpangan baku (Standar Deviasi) dari setiap data

```
## Calculate standart deviation of data
def get_standDev(avgTrue, avgFalse, dataList):
    i = 0
    ptTrue = 0
    ptFalse = 0
    standDevTrue = data(-1,0,0,0,-1)
    standDevFalse = data(-1,0,0,0,-1)
    while i != len(dataList):
        if dataList[i].y == 1:
            standDevTrue.x1 = standDevTrue.x1 + ((dataList[i].x1 - avgTrue.x1) ** 2)
            standDevTrue.x2 = standDevTrue.x2 + ((dataList[i].x2 - avgTrue.x2) ** 2)
            standDevTrue.x3 = standDevTrue.x3 + ((dataList[i].x3 - avgTrue.x3) ** 2)
            ptTrue += 1
        else:
            standDevFalse.x1 = standDevFalse.x1 + ((dataList[i].x1 - avgFalse.x1) ** 2)
            standDevFalse.x2 = standDevFalse.x2 + ((dataList[i].x2 - avgFalse.x2) ** 2)
            standDevFalse.x3 = standDevFalse.x3 + ((dataList[i].x3 - avgFalse.x3) ** 2)
            ptFalse += 1
        i += 1

    standDevTrue.x1 = standDevTrue.x1 / ptTrue
    standDevTrue.x2 = standDevTrue.x2 / ptTrue
    standDevTrue.x3 = standDevTrue.x3 / ptTrue
    standDevTrue.x1 = standDevTrue.x1 ** 0.5
    standDevTrue.x2 = standDevTrue.x2 ** 0.5
    standDevTrue.x3 = standDevTrue.x3 ** 0.5
    standDevFalse.x1 = standDevFalse.x1 / ptFalse
    standDevFalse.x2 = standDevFalse.x2 / ptFalse
    standDevFalse.x3 = standDevFalse.x3 / ptFalse
    standDevFalse.x1 = standDevFalse.x1 ** 0.5
    standDevFalse.x2 = standDevFalse.x2 ** 0.5
    standDevFalse.x3 = standDevFalse.x3 ** 0.5

    return standDevTrue, standDevFalse
```

4. Proses Naive Bayes

```
## Naive Bayes method
def naiveBayes(dataList, testData, avgTrue, avgFalse, standDevTrue, standDevFalse):
    i = 0
    sumTrue = 0
    sumFalse = 0
    while i != len(dataList):
        if dataList[i].y == 1:
            sumTrue += 1
        else:
            sumFalse += 1
        i += 1
    i = 0
    while i != len(testData):
        valTrue = (sumTrue / len(dataList)) * ((EULER ** -((testData[i].x1 - avgTrue.x1) ** 2) / (2 * (standDevTrue.x1 ** 2)))) / (standDevTrue.x1 * ((2 * Phi) ** 0.5))) * (EULER ** -(testData[i].x2 - avgTrue.x2) **
        valFalse = (sumFalse / len(dataList)) * ((EULER ** -((testData[i].x1 - avgFalse.x1) ** 2) / (2 * (standDevFalse.x1 ** 2)))) / (standDevFalse.x1 * ((2 * Phi) ** 0.5))) * (EULER ** -(testData[i].x2 - avgFalse.x
        if valTrue > valFalse:
            testData[i].y = 1
        else:
            testData[i].y = 0
        i += 1
    return testData
```

```
.x2) ** 2) / (2 * (standDevTrue.x2 ** 2)) / (standDevTrue.x2 * ((2 * Phi) ** 0.5)) * (EULER ** -(testData[i].x3 - avgTrue.x3) ** 2) / (2 * (standDevTrue.x3 ** 2)) / (standDevTrue.x3 * ((2 * Phi) ** 0.5))
gFalse.x2) ** 2) / (2 * (standDevFalse.x2 ** 2)) / (standDevFalse.x2 * ((2 * Phi) ** 0.5)) * (EULER ** -(testData[i].x3 - avgFalse.x3) ** 2) / (2 * (standDevFalse.x3 ** 2)) / (standDevFalse.x3 * ((2 * Phi) ** 0.5))
```

5. Import output ke dalam file excel

```
## Import output to excel file
def outputToExcel(input):
    for i in range(len(input)):
        testSheet["E"+str(i+2)].value = input[i].y
    ## Change book.save("File Location")
    book.save("./traintest.xlsx")

## Training Session
print("### Training Session ###")
print("Press Any Key to start training")
input()
trainList = get_DataTrain()
avgTrue, avgFalse = get_average(trainList)
standDevTrue, standDevFalse = get_standDev(avgTrue, avgFalse, trainList)
print("### Train Complete ###")
print("")

## Testing Session
print("### Testing Session ###")
print("Press Any Key to start testing")
input()
testList = get_DataTest()
testList = naiveBayes(trainList, testList, avgTrue, avgFalse, standDevTrue, standDevFalse)
outputToExcel(testList)
print("### Test Complete ###")
print("### Data have been changed ###")
print("")
```

6. Output

```
### Training Session ###
Press Any Key to start training

### Train Complete ###

### Testing Session ###
Press Any Key to start testing

### Test Complete ###
### Data have been changed ###
```

id	x1	x2	x3	y
297	43	59	2	1
298	67	66	0	1
299	58	60	3	1
300	49	63	3	1
301	45	60	0	1
302	54	58	1	1
303	56	66	3	1
304	42	69	1	1
305	50	59	2	1
306	59	60	0	1

V. Testing Program

Test 1 Data Original

id	x1	x2	x3	y
297	43	59	2	1
298	67	66	0	1
299	58	60	3	1
300	49	63	3	1
301	45	60	0	1
302	54	58	1	1
303	56	66	3	1
304	42	69	1	1
305	50	59	2	1
306	59	60	0	1

Test 2 Data nilai x1 diubah

id	x1	x2	x3	y
297	67	59	2	1
298	43	66	0	1
299	49	60	3	1
300	58	63	3	1
301	54	60	0	1
302	45	58	1	1
303	42	66	3	1
304	56	69	1	1
305	59	59	2	1
306	50	60	0	1

Test 3 Data nilai x2 diubah

A	B	C	D	E
id	**x1**	**x2**	**x3**	**y**
297	43	66	2	1
298	67	59	0	1
299	58	63	3	1
300	49	60	3	1
301	45	58	0	1
302	54	60	1	1
303	56	69	3	1
304	42	66	1	1
305	50	60	2	1
306	59	59	0	1

Test 3 Data x3 diubah

id	x1	x2	x3	y
297	43	59	0	1
298	67	66	1	1
299	58	60	2	1
300	49	63	3	1
301	45	60	4	1
302	54	58	5	1
303	56	66	6	0
304	42	69	7	0
305	50	59	8	0
306	59	60	9	0

Test 4 Data diubah secara random

id	x1	x2	x3	y
297	68	55	0	1
298	65	53	5	1
299	55	58	2	1
300	45	60	6	0
301	66	43	3	1
302	66	60	3	1
303	42	55	9	0
304	45	64	8	0
305	57	67	4	1
306	55	55	2	1

VI. Kesimpulan

Dengan dilakukan test sebanyak 4 kali dimana test pertama menggunakan data asli dari file traintest.xlsx, lalu test kedua dimana nilai x1 diubah, test ketiga dimana nilai x2 diubah, test ketiga dimana nilai x3 diubah dan terakhir test keempat dimana nilai x1,x2 dan x3 diubah secara acak. Didapatkan bahwa dengan menggunakan algoritma Naive Bayes nilai y dari test sheet file traintest.xlsx akan bernilai 0 bila nilai dari x3 lebih dari 5 dan akan bernilai 1 bila nilai x3 kurang dari sama dengan 5 atau:

$$y \begin{cases} x3 \leq 5, 1 \\ x3 > 5, 0 \end{cases}$$

Daftar Pustaka

Dicoding. *Apa Itu Kecerdasan Buatan? Berikut Pengertian dan Contohnya*. Diakses pada 19 Juni 2022, dari https://www.dicoding.com/blog/kecerdasan-buatan-adalah/.

Haldi Widianto, Mochammad. *Algoritma Naive Bayes*. Diakses pada 19 Juni 2022, dari https://binus.ac.id/bandung/2019/12/algoritma-naive-bayes/.