Pokok Bahasan VIII
Naïve Bayes dengan Python

Kode Pokok Bahasan: TIK.RPL03.005.00.01

Deskripsi Pokok Bahasan:
Membahas bagaimana penerapan Algoritma Naïve Bayes pada dataset titanic.

No	Elemen Kompetensi	Indikator Kinerja	Jml Jam	Hal
1.	Menampilkan peluang dari kasus yang diberikan.	Mampu melakukan analisis terhadap peluang atas kejadian yang ditentukan.	1	12
2.	Melakukan perhitungan menggunakan R.	Mampu melakukan perhitungan peluang menggunakan R pada kondisi yang ditentukan	2	15
3.	Menggunakan fungsi naiveBayes dalam memprediksi data	Mengimplementasikan fungsi naive bayes pada prediksi data		

TUGAS PENDAHULUAN
Hal yang harus dilakukan dan acuan yang harus dibaca sebelum praktikum :
1. Menginstal R pada PC masing-masing praktikan.
2. Menginstal R Studio pada PC masing-masing praktikan.

DAFTAR PERTANYAAN
1. Apa itu algoritma Naïve Bayes?

Algoritma Naive Bayes adalah algoritma yang mempelajari probabilitas suatu objek dengan ciri-ciri tertentu yang termasuk dalam kelompok/kelas tertentu. Singkatnya, ini adalah pengklasifikasi probabilistik.

2. Apa kegunaan Naïve Bayes?

Prediksi multi-kelas: Algoritma klasifikasi Naive Bayes dapat digunakan untuk memprediksi probabilitas posterior dari beberapa kelas variabel target. Klasifikasi teks: Karena fitur prediksi multi-kelas, Naive Bayes algoritma klasifikasi sangat cocok untuk klasifikasi teks

3. Sebutkan tahapan dari proses algoritma Naïve Bayes!

Menghitung jumlah kelas/label. Menghitung jumlah kasus perkelas. Mengalikan semua hasil variable kelas. Membandingkan hasil perkelas.

TEORI SINGKAT

Algoritma Naive Bayes merupakan sebuah metoda klasifikasi menggunakan metode probabilitas dan statistik yg dikemukakan oleh ilmuwan Inggris Thomas Bayes. Algoritma Naive Bayes 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.

Naive Bayes Classifier bekerja sangat baik dibanding dengan model classifier lainnya. Hal ini dibuktikan pada jurnal Xhemali, Daniela, Chris J. Hinde, and Roger G. Stone. "Naive Bayes vs. decision trees vs. neural networks in the classification of training web pages." (2009), mengatakan bahwa "Naïve Bayes Classifier memiliki tingkat akurasi yg lebih baik dibanding model classifier lainnya".

Keuntungan penggunan adalah bahwa metoda ini hanya membutuhkan jumlah data pelatihan (training data) yang kecil untuk menentukan estimasi parameter yg diperlukan dalam proses pengklasifikasian. Karena yg diasumsikan sebagai variabel independent, maka hanya varians dari suatu variabel dalam sebuah kelas yang dibutuhkan untuk menentukan klasifikasi, bukan keseluruhan dari matriks kovarians.

LAB SETUP

Hal yang harus disiapkan dan dilakukan oleh praktikan untuk menjalankan praktikum modul ini.

1. Menginstall library yang dibutuhkan untuk mengerjakan modul.
2. Menjalankan Jupyter Notebook.

ELEMEN KOMPETENSI I

Deskripsi:
Menampilkan peluang dari kasus yang diberikan.

Kompetensi Dasar:
Mampu melakukan analisis terhadap peluang atas kejadian yang ditentukan.

Latihan

Penjelasan Singkat :
Pada latihan ini anda akan diminta untuk menampilkan summarize dari data menggunakan python.

Langkah-Langkah Praktikum:

1. Instal package berikut

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
```

2. Buat dataframe yang digunakan

```
df=pd.read_csv("train_clean.csv")
df_test=pd.read_csv("test_clean.csv")
```

3. Membuat sampel train dan test

```
train, test = train_test_split(df, test_size=0.2)
```

4. Jalankan perintah berikut

Hitung probabilitas selamat penumpang Titanic yang berjenis kelamin perempuan.

```
surviving_female = df[(df['Sex'] == "female") & (df['Survived'] == 1)]
P_yes=len(surviving_female)/len(train)
P_yes
```

Output :

```
In [1]: import numpy as np
        import pandas as pd
        from sklearn.model_selection import train_test_split

In [2]: df=pd.read_csv("D:/File Kuliah Semester 5/Penambangan Data/Prak-8/train_clean.csv")
        df_test=pd.read_csv("D:/File Kuliah Semester 5/Penambangan Data/Prak-8/test_clean.csv")

In [3]: train, test = train_test_split(df, test_size=0.2)

In [4]: surviving_female = df[(df['Sex'] == "female") & (df['Survived'] == 1)]
        P_yes=len(surviving_female)/len(train)
        P_yes

Out[4]: 0.32724719101123595
```

ELEMEN KOMPETENSI II

Deskripsi:
Melakukan perhitungan manual menggunakan R.

Kompetensi Dasar:
Mampu melakukan perhitungan peluang secara manual menggunakan R pada kondisi yang ditentukan

Latihan 1.2.1

Penjelasan Singkat :

Pada latihan ini anda akan diminta untuk melakukan perhitungan peluang secara manual menggunakan R.

Langkah-Langkah Praktikum:

Lakukan perhitungan kasus di atas dengan menggunakan R.

Output :

```
R 4.2.1 · ~/
> library(tidyverse)
> library(titanic)
> tdf <- titanic_train
> head(tdf)
  PassengerId Survived Pclass                                                 Name    Sex Age SibSp Parch
1           1        0      3                              Braund, Mr. Owen Harris   male  22     1     0
2           2        1      1 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
3           3        1      3                               Heikkinen, Miss. Laina female  26     0     0
4           4        1      1        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
5           5        0      3                             Allen, Mr. William Henry   male  35     0     0
6           6        0      3                                     Moran, Mr. James   male  NA     0     0
            Ticket     Fare Cabin Embarked
1        A/5 21171  7.2500              S
2         PC 17599 71.2833   C85        C
3 STON/O2. 3101282  7.9250              S
4           113803 53.1000  C123        S
5           373450  8.0500              S
6           330877  8.4583              Q
> tdf %>% summarize(prob = sum(Age >= 35 & Sex == "female", na.rm = TRUE)/sum(Age >= 35,na.rm=TRUE))
       prob
1 0.3446809
> |
```

ELEMEN KOMPETENSI III

Deskripsi:
Menggunakan fungsi naïve bayes dalam memprediksi data.

Kompetensi Dasar:
Mengimplementasikan fungsi naive bayes pada prediksi data

Latihan 1.3.1

Penjelasan Singkat :
Pada latihan ini anda akan diminta untuk mengimplementasikan naïve bayes pada kasus yang diberikan.

Langkah-Langkah Praktikum:

Gunakan titanic.csv yang berisi data 887 penumpang Titanic passengers. Kolom data menggambarkan survived (S), age (A), passenger-class (C), sex (G) and the fare paid (X).
Hitung peluang bersyarat (conditional probability) di bawah ini

$P(S=\text{true} \mid G=\text{female})$
$P(S=\text{true} \mid G=\text{male})$
$P(S=\text{true} \mid C=1)$
$P(S=\text{true} \mid C=2)$
$P(S=\text{true} \mid C=3)$
$P(S=\text{true} \mid G=\text{female}, C=1) =$

$P(S=\text{true} \mid G=\text{female}, C=2) =$
$P(S=\text{true} \mid G=\text{female}, C=3) =$
$P(S=\text{true} \mid G=\text{male}, C=1) =$
$P(S=\text{true} \mid G=\text{male}, C=2) =$
$P(S=\text{true} \mid G=\text{male}, C=3) =$

Gunakan Python dan R :

```
> #S=True/G=Female
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "female", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.6812865
> #S=True/G=Male
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "male", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.3187135
> #S=True/C=1
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Pclass == "1", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.3976608
> #S=True/C=2
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Pclass == "2", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
      prob
1 0.254386
> #S=True/C=3
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Pclass == "3", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.3479532
> #S=True/G=Female/C=1
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "female" & Pclass == "1", na.rm = TRUE)/sum(Survived == "1", na.rm = TRU
E))
       prob
1 0.2660819

> #S=True/G=Female/C=2
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "female" & Pclass == "2", na.rm = TRUE)/sum(Survived == "1", na.rm = TRU
E))
       prob
1 0.2046784
> #S=True/G=Female/C=3
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "female" & Pclass == "3", na.rm = TRUE)/sum(Survived == "1", na.rm = TRU
E))
       prob
1 0.2105263
> #S=True/G=male/C=1
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "male" & Pclass == "1", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.1315789
> #S=True/G=male/C=2
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "male" & Pclass == "2", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.0497076
> #S=True/G=male/C=3
> tdf %>%
+   summarize(prob = sum(Survived == "1" & Sex == "male" & Pclass == "3", na.rm = TRUE)/sum(Survived == "1", na.rm = TRUE))
       prob
1 0.1374269
>
```

Jalankan perintah R di bawah ini :

```
import pandas as pd
import numpy as np
titanic = pd.read_csv("train_clean.csv")
titanic.head()
```

Output :

```
In [5]: import pandas as pd
        import numpy as np
        titanic = pd.read_csv("D:/File Kuliah Semester 5/Penambangan Data/Prak-8/train_clean.csv")
        titanic.head()
```

Out[5]:

	Age	Cabin	Embarked	Fare	Name	Parch	Pclass	Sex	SibSp	Survived	Ticket
0	22.0	NaN	S	7.2500	Braund, Mr. Owen Harris	0	3	male	1	0	A/5 21171
1	38.0	C85	C	71.2833	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	1	female	1	1	PC 17599
2	26.0	NaN	S	7.9250	Heikkinen, Miss. Laina	0	3	female	0	1	STON/O2. 3101282
3	35.0	C123	S	53.1000	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	1	female	1	1	113803
4	35.0	NaN	S	8.0500	Allen, Mr. William Henry	0	3	male	0	0	373450

```
titanic.drop(['Name','SibSp','Parch','Ticket','Cabin','Embarked'],axis='columns',inplace=True)
titanic.head()
```

Output :

```
In [6]: titanic.drop(['Name','SibSp','Parch','Ticket','Cabin','Embarked'],axis='columns',inplace=True)
        titanic.head()
```

Out[6]:

	Age	Fare	Pclass	Sex	Survived
0	22.0	7.2500	3	male	0
1	38.0	71.2833	1	female	1
2	26.0	7.9250	3	female	1
3	35.0	53.1000	1	female	1
4	35.0	8.0500	3	male	0

```
# Variabel independen
x = titanic.drop(["Survived"], axis = 1)
x.head()
# Variabel dependen
y = titanic["Survived"]
y.head()
```

Output :

```
In [7]: # Variabel independen
        x = titanic.drop(["Survived"], axis = 1)
        x.head()
        # Variabel dependen
        y = titanic["Survived"]
        y.head()

Out[7]: 0    0
        1    1
        2    1
        3    1
        4    0
        Name: Survived, dtype: int64
```

```
x['Sex'].replace(['female','male'],[0,1],inplace=True)
x.head()
```

Output :

```
In [8]: x['Sex'].replace(['female','male'],[0,1],inplace=True)
        x.head()
```

Out[8]:

	Age	Fare	Pclass	Sex
0	22.0	7.2500	3	1
1	38.0	71.2833	1	0
2	26.0	7.9250	3	0
3	35.0	53.1000	1	0
4	35.0	8.0500	3	1

```
x.Age = x.Age.fillna(x.Age.mean())
x.head()
```

Output :

```
In [9]: x.Age = x.Age.fillna(x.Age.mean())
        x.head()
```

Out[9]:

	Age	Fare	Pclass	Sex
0	22.0	7.2500	3	1
1	38.0	71.2833	1	0
2	26.0	7.9250	3	0
3	35.0	53.1000	1	0
4	35.0	8.0500	3	1

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 123)

from sklearn.naive_bayes import GaussianNB
# Mengaktifkan/memanggil/membuat fungsi klasifikasi Naive Bayes
modelnb = GaussianNB()
# Memasukkan data training pada fungsi klasifikasi Naive Bayes
nbtrain = modelnb.fit(x_train, y_train)

y_pred = nbtrain.predict(x_test)
y_pred
```

Output :

```
In [10]: from sklearn.model_selection import train_test_split
         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 123)

         from sklearn.naive_bayes import GaussianNB
         # Mengaktifkan/memanggil/membuat fungsi klasifikasi Naive Bayes
         modelnb = GaussianNB()
         # Memasukkan data training pada fungsi klasifikasi Naive Bayes
         nbtrain = modelnb.fit(x_train, y_train)

         y_pred = nbtrain.predict(x_test)
         y_pred

Out[10]: array([1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
                0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1,
                0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0,
                0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0,
                0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1,
                0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
                0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0,
                1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0,
                0, 0, 0], dtype=int64)
```

np.array(y_test)

Output :

```
In [11]: np.array(y_test)

Out[11]: array([1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1,
                0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1,
                0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0,
                0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0,
                1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
                1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0,
                0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0,
                1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1,
                1, 0, 0], dtype=int64)
```

nbtrain.predict_proba(x_test)

Output :

```
In [12]: nbtrain.predict_proba(x_test)

Out[12]: array([[2.36324308e-01, 7.63675692e-01],
                [9.61620421e-01, 3.83795788e-02],
                [7.54065607e-01, 2.45934393e-01],
                [7.43127400e-01, 2.56872600e-01],
                [9.60584388e-01, 3.94156123e-02],
                [9.27823545e-01, 7.21764546e-02],
                [2.63372310e-01, 7.36627690e-01],
                [2.34057020e-01, 7.65942980e-01],
                [3.52692508e-01, 6.47307492e-01],
                [4.01731266e-01, 5.98268734e-01],
                [2.68524309e-01, 7.31475691e-01],
                [3.97377756e-01, 6.02622244e-01],
                [1.15912833e-02, 9.88408717e-01],
                [9.58220678e-01, 4.17793221e-02],
                [8.34959064e-02, 9.16504094e-01],
                [2.34055890e-01, 7.65944110e-01],
                [2.51153629e-04, 9.99748846e-01],
                [6.80452913e-01, 3.19547087e-01],
                [4.21379871e-01, 5.78620129e-01],

In [ ]:
```

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)
```

Output :

```
In [13]: from sklearn.metrics import confusion_matrix
         confusion_matrix(y_test, y_pred)

Out[13]: array([[96, 18],
                [18, 47]], dtype=int64)
```

Berikan penjelasan terhadap output di atas

CEK LIST

Elemen Kompetensi	No Latihan	Penyelesaian	
		Selesai	Tidak selesai
1	1.1.1	✓	
2	1.2.1	✓	
3	1.3.1	✓	

FORM UMPAN BALIK

Elemen Kompetensi	**Tingkat Kesulitan**	**Tingkat Ketertarikan**	**Waktu Penyelesaian dalam menit**
Memahami data pre-processing.	☐ Sangat Mudah ☐ Mudah ☐ ✓ Biasa ☐ Sulit ☐ Sangat Sulit	☐ Tidak Tertarik ☐ Cukup Tertarik ☐ Tertarik ☐ ✓ Sangat Tertarik	

Mengimplementasikan pre-processing data.	☐ Sangat Mudah ☐ Mudah ☐ ✓ Biasa ☐ Sulit ☐ Sangat Sulit	☐ Tidak Tertarik ☐ Cukup Tertarik ☐ Tertarik ☐ ✓ Sangat Tertarik	

Elemen Kompetensi	**Tingkat Kesulitan**	**Tingkat Ketertarikan**	**Waktu Penyelesaian dalam menit**
Menggunakan fungsi naiveBayes dalam memprediksi data	☐ Sangat Mudah ☐ Mudah ☐ ✓ Biasa ☐ Sulit ☐ Sangat Sulit	☐ Tidak Tertarik ☐ Cukup Tertarik ☐ Tertarik ☐ ✓ Sangat Tertarik	