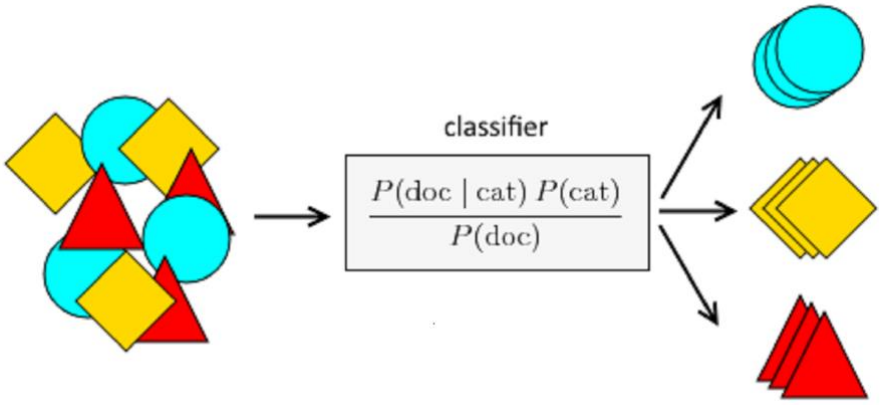
PERTEMUAN 4

Naïve Bayes

1. Konsep Naïve Bayes

Naïve Bayes Classifier merupakan salah satu model machine learning yang digunakan untuk membedakan suatu object berdasarkan fitur tertentu. Naïve Bayes Classifier adalah metode pengklasifikasian paling sederhana dari model pengklasifikasian yang ada dengan menggunakan konsep peluang, dimana diasumsikan bahwa setiap atribut contoh (data sampel) bersifat saling lepas satu sama lain berdasarkan atribut kelas. Munculnya ide metode klasifikasi Naïve Bayes bahwa metode klasifikasi ini diturunkan dari penerapan teorema Bayes dengan asumsi independence (saling bebas).



Teorema Bayes memprediksi probabilitas suatu peristiwa berdasarkan pengalaman di masa sebelumnya. Teorema Bayes ditemukan oleh seorang ahli statistik dan menteri Inggris bernama Thomas Bayes pada abad ke-18. Teorema bayes adalah sebagai berikut **Peluang kejadian A sebagai B ditentukan dari peluang B saat A, peluang A, dan peluang B.** Sehingga:

$$\boldsymbol{P(A|B) = \frac{P(A)P(B|A)}{P(B)}}$$

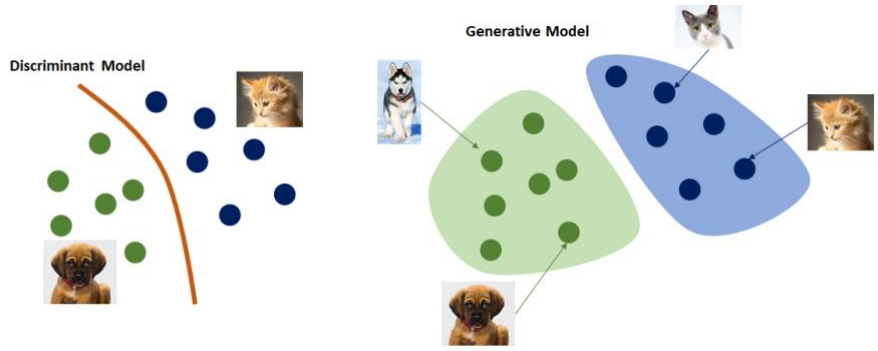
Dimana A dan B merupakan suatu kejadian/peristiwa, $\boldsymbol{P}$ ($\boldsymbol{A}$) adalah probabilitas mengamati peristiwa $\boldsymbol{A}$. $\boldsymbol{P}$ ($\boldsymbol{B}$) adalah probabilitas mengamati peristiwa $\boldsymbol{B}$. $\boldsymbol{P}$ ($\boldsymbol{A} \mid \boldsymbol{B}$) adalah probabilitas bersyarat untuk mengamati $\boldsymbol{A}$ mengingat $\boldsymbol{B}$ telah diamati. Dalam tugas klasifikasi, tujuannya adalah untuk memetakan fitur variabel penjelas (*explanatory variable*) ke variabel respon yang berupa variable diskrit.

2. Discriminative and Generative Model

Discriminative Model memisahkan kelas alih-alih memodelkan probabilitas bersyarat dan tidak membuat asumsi apa pun tentang titik data. Tetapi model ini tidak mampu memprediksi titik data baru. Oleh karena itu, tujuan akhir dari model diskriminatif adalah untuk memisahkan satu kelas dari yang lain.

Generative Model memodelkan pola atau distribusi titik-titik data yang mendasarinya. Model ini menggunakan konsep probabilitas bersama dan membuat contoh dimana fitur (x) atau input yang diberikan dan output atau label (y) yang diinginkan ada pada saat yang sama. Tidak seperti model diskriminatif, model ini juga mampu memprediksi titik data baru.

Contoh: Melakukan klasifikasi kucing dan anjing berdasarkan berat dan tinggi. Semisal diberikan 1000 data, maka:



Menggunakan Discriminative Model

- Hanya perlu menghitung P (y | x) untuk setiap titik data.
- Hanya perlu menghitung 2.000 probabilitas jika kumpulan data memiliki 1.000 titik data
- Contoh Discriminative models:
    - Logistic regression
    - SVM
    - Neural Networks

Menggunakan Generative Model:

- Harus dihitung probabilitas berikut ini untuk setiap titik data:
    - P (kucing, berat)
    - P (kucing, tinggi)

- P (anjing, berat)
- P (anjing, tinggi)

- JIKA terdapat 1.000 data untuk pelatihan terhadap model hal ini berarti bahwa setidaknya perlu menghitung 4.000 probabilitas.
- Contoh Generative models:
  - Naive Bayes
  - Gaussian mixture model
  - Hidden Markov Models (HMM)

3. Komputasi Naive Bayes

Formula naïve bayes

$$P(y|x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n|y)P(y)}{P(x_1, \dots, x_n)}$$

Dimana p(y) adalah peluang setiap kelas pada data training. P (x_1,…, x_n) merupakan konstan untuk semua input, sehingga bisa dihilangkan dan yang tersisa adalah probabilitas bersyarat $P(x_1, \dots, x_n|y)$ dan probabilitas peluang setiap kelas $P(y)$. Pada naïve bayes estimasi kelas menggunakan **maximum a posteriori estimation (MAP)** yaitu hipotesa yang diambil berdasarkan nilai probabilitas berdasarkan kondisi prior yang diketahui.

Sehingga didapatkan formula pada persamaan berikut ini:

$$P(y|x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n|y)P(y)}{P(x_1, \dots, x_n)}$$

Ketika $P(x_1, \dots, x_n)$ dihilangkan, maka persamaan yang dihasilkan:

$$P(y|x_1, \dots, x_n) \propto P(y)P(x_1|y)P(x_2|y) \dots . P(x_n|y)$$

Formula $P(x_1|y)P(x_2|y) \dots . P(x_n|y)$ dapat dituliskan dengan $\prod_{i=1}^{n} P(x_i|y)$, sehingga:

$$P(y|x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

Untuk melakukan prediksi label kelas digunakan MAP. MAP inilah yang digunakan di dalam machine learning sebagai metode untuk mendapatkan hipotesis untuk suatu keputusan. Sehingga didapatkan persamaan berikut ini:

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

4. Tipe-tipe Naïve Bayes

a. Multinomial Naive Bayes

Multinominal Naïve bayes banyak digunakan untuk masalah klasifikasi dokumen, karena multinominal dapat mengatasi permasalahan multiclass. Misal apakah suatu dokumen termasuk dalam kategori olahraga, politik, teknologi dll. Fitur / prediktor yang digunakan oleh classifier adalah frekuensi kata-kata yang sering muncul dalam dokumen.

b. Bernoulli Naive Bayes

Bernoullli Naïve Bayes mirip dengan multinomial Naïve Bayes tetapi fitur independennya adalah variabel Boolean. Parameter yang digunakan untuk memprediksi variabel kelas hanya mengambil nilai ya atau tidak, misalnya apakah terdapat kata 'x' dalam teks atau tidak.

c. Gaussian Naive Bayes

Gaussian naïve bayes digunakan ketika fitur independennya adalah nilai kontinu dan tidak diskrit, sehingga diasumsikan bahwa nilai-nilai ini diambil sampelnya dari distribusi Gaussian.

5. Contoh Perhitungan Manual Naïve Bayes

Jika diketahui data berikut ini sebagai kebiasaan olahraga seseorang dan ditabelkan dalam bentuk sebagai berikut:

Tabel 6. 1 Tabel olahraga

#	Cuaca	Kecepatan Angin	Berolahraga
1	Cerah	Pelan	Ya
2	Cerah	Pelan	Ya
3	Hujan	Pelan	Tidak
4	Cerah	Kencang	Ya
5	Hujan	Kencang	Tidak
6	Cerah	Pelan	Ya
7	Cerah	Kencang	?

Asumsi:

Y = berolahraga, X1 = cuaca,

X2 = temperatur, X3 = kecepatan angin

Berdasarkan Tabel didapatkan:

peluang setiap kelas pada data training P(Y=ya) dan P(Y=tidak)

P(Y=ya) = $\frac{4}{6}$ P(Y=tidak) = $\frac{2}{6}$

Jika terdapat data baru berupa cuaca cerah dan kecepatan angin kencang, Maka keputusan yang diambil apakah berolahraga atau tidak?

Tentukan terlebih dahulu peluang bersyarat saat cuaca cerah dan keputusan belorahraga "ya", cuaca cerah dan keputusan belorahraga "tidak", kecepatan angin kencang dan keputusan belorahraga "ya", kecepatan angin kencang dan keputusan belorahraga "tidak". Sehingga didapatkan :

P(X1=cerah | Y=ya) = 1, P(X1=cerah | Y=tidak) = 0

P(X3=kencang | Y=ya) = $\frac{1}{4}$, P(X3=kencang | Y=tidak) = $\frac{1}{2}$

MAP dari keadaan ini dapat dihitung dengan:

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

P(X1=cerah,X3=kencang | Y=ya)

= { P(X1=cerah | Y=ya).P(X3=kencang | Y=ya) } . P(Y=ya)

= { (1) . ($\frac{1}{4}$) } . ($\frac{4}{6}$)) = $\frac{1}{6}$

P(X1=cerah,X3=kencang | Y=tidak)

= { P(X1=cerah | Y=tidak).P(X3=kencang | Y=tidak) } . P(Y=tidak)

= { (0) . ($\frac{1}{2}$) } . ($\frac{2}{6}$) = 0

Keputusannya adalah Y=ya karena nilai untuk kelas Ya memiliki probabilitas yang lebih besar dibandingkan dengan kelas Y=tidak.

6. Gaussian Naive Bayes

Naive bayes classifier juga dapat menangani atribut bertipe kontinyu. Salah satu caranya adalah menggunakan distribusi Gaussian. Distribusi ini dikarakterisasi dengan 2

parameter yaitu mean (μ), dan variansi(σ). Untuk setiap kelas Yj , peluang kelas bersyarat untuk atribut Xi dinyatakan dengan persamaan distribusi Gaussian. Fungsi densitas mengekspresikan probabilitas relatif. Data dengan mean μ dan standar deviasi σ, fungsi densitas probabilitasnya adalah:

$$\varphi_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Dimana μ adalah rata-rata dari fitur x, σ adalah variance (standar deviasi) dari fitur x, dan $\varphi_{\mu,\sigma}(x)$ untuk menghitung Likelihood P(X | Y).

Contoh kasus: Misal terdapat data pada tabel 6.2 untuk menentukan apakah suatu objek masuk dalam kategori yang dipilih untuk perumahan atau tidak dengan menggunakan algoritma *Naive Bayes Classifier*. Untuk menetapkan suatu daerah akan dipilih sebagai lokasi untuk mendirikan perumahan, telah dihimpun 10 data. Data yang digunakan adalah harga tanah, jarak dari pusat kota, dan ada tidaknya angkutan umum. Variabel harga tanah dan jarak kota merupakan variable kontinyu, sedangkan ada tidaknya angkutan umum adalah variable diskrit.

Tabel 6. 2 Data Penentuan Perumahan

Aturan ke-	Harga tanah (C1)	Jarak dari pusat kota (C2)	Ada angkutan umum (C3)	Dipilih untuk perumahan (C4)
1	100	2	Tidak	Ya
2	200	1	Tidak	Ya
3	500	3	Tidak	Ya
4	600	20	Tidak	Tidak
5	550	8	Tidak	Tidak
6	250	25	Ada	Tidak
7	75	15	Ada	Tidak
8	80	10	Tidak	Ya
9	700	18	Ada	Tidak
10	180	8	Ada	Ya

Peluang setiap kelas pada data training P(C4=ya) dan P(C4=tidak)

P(C4=ya) = $\frac{5}{10}$ P(Y=tidak) = $\frac{5}{10}$

Jika terdapat data baru berupa **C1 = 300, C2 = 17, C3 = Tidak**, maka keputusan yang diambil apakah dipilih perumahan dan tidak?

- Menghitung nilai mean dan variance variabel C1

	Ya	Tidak
1	100	600
2	200	550
3	500	250
4	80	75
5	180	700
Mean (μ) Deviasi standar (σ)	212 168,8787	435 261,9637

- Menghitung nilai mean dan variance variabel C2

	Ya	Tidak
1	2	20
2	1	8
3	3	25
4	10	15
5	8	18
Mean (μ) Deviasi standar (σ)	4,8 3,9623	17,2 6,3008

- Densitas probabilitas untuk C1 dan C2

$$f(C1=300\,|\,ya)=\frac{1}{\sqrt{2\pi}(168{,}8787)}e^{\frac{-(300-212)^2}{2(168{,}8787)^2}}=0{,}0021.$$

$$f(C1=300\,|\,tidak)=\frac{1}{\sqrt{2\pi}(261.9637)}e^{\frac{-(300-435)^2}{2(261.9637)^2}}=0{,}0013.$$

$$f(C2=17\,|\,ya)=\frac{1}{\sqrt{2\pi}(3.9623)}e^{\frac{-(17-4{,}8)^2}{2(3.9623)^2}}=0{,}0009.$$

$$f(C2=17\,|\,tidak)=\frac{1}{\sqrt{2\pi}(6{,}3008)}e^{\frac{-(17-17{,}2)^2}{2(6{,}3008)^2}}=0{,}0633.$$

- Probabilitas kemunculan P(C3=tidak | Y=ya) = 4/5, P(C3=tidak | Y=tidak) = 2/5
- MAP dari keadaan ini dapat dihitung dengan:

$$y = argmax_y P(y)\prod_{i=1}^{n} P(x_i|y)$$

Probabilitas terhadap "ya"

- P (C1=300, C2=17, C3=tidak | Y=ya)
- {P (C1=300|Y=ya). P (C2=17 |Y=ya). P (C3=tidak |Y=ya)}. P(Y=ya)
- 0,0021 * 0,0009 * (4/5) * (1/2)
- 0,000000756

Probabilitas terhadap "tidak"

- P (C1=300, C2=17, C3=tidak | Y=tidak)
- {P (C1=300|Y=tidak). P (C2=17 |Y=tidak). P (C3=tidak |Y=tidak)}. P(Y=tidak)
- 0,0013 * 0,0633 * (2/5) * (1/2)
- 0,000016458.

Keputusannya adalah C4 = tidak, karena nilai untuk kelas tidak memiliki probabilitas yang lebih besar dibandingkan dengan kelas Y=Ya. Data tersebut dapat dilakukan normalisasi sebagai berikut:

$$\text{Probabilitas Ya} = \frac{0{,}000000756}{0{,}000000756 + 0{,}000016458} = 0{,}0439.$$

$$\text{Probabilitas Tidak} = \frac{0{,}000016458}{0{,}000000756 + 0{,}000016458} = 0{,}9561.$$

Klasifikasi : TIDAK

7. Komputasi Naive Bayes Bernoulli

- Diberikan data-data sebagai berikut, lakukan komputasi Naive Bayes dan ujikan hasil pelatihan pada hasil pembelajarannya.

Tabel 1. Data Pelatihan Naive Bayes.

	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

- **Langkah-langkah Komputasi Naive Bayes Bernoulli**

1. **Hitung Probabilitas Apriori Play dan Not Play.**

   a. $P(Play) = \frac{Jumlah\ Play}{Jumlah\ Total\ Play\ dan\ \neg Play} = \frac{9}{14} = 0{,}64$

   b. $P(\neg Play) = \frac{Jumlah\ \neg Play}{Jumlah\ Total\ Play\ dan\ \neg Play} = \frac{5}{14} = 0{,}36$

   Total Probabilitas Play dan Not Play adalah $P(Play) + P(\neg Play) = 0{,}64 + 0{,}36 = 1{,}00$.

2. **Hitung Probabilitas Apriori masing-masing fitur untuk *Play* sebagai berikut dengan jumlah data sebanyak 9 buah.**

Tabel 2. Komputasi Probabilitas masing-masing fitur untuk Label Play

No.	Nama Fitur	Nama Subfitur	Kemunculan	Probabilitas	
				Perbandingan	Angka
1.	Outlook	Rainy	2	2/9	0,2222
		Overcast	4	4/9	0,4444
		Sunny	3	3/9	0,3333
		Total	**9**	**9/9**	**0,999**
2.	Temperature	Hot	2	2/9	0,2222
		Mild	4	4/9	0,4444

		Cool	3	3/9	0,3333
		Total	**9**	**9/9**	**0,999**
3.	Humidity	High	3	3/9	0,3333
		Normal	6	6/9	0,6667
		Total	**9**	**9/9**	**1,0**
4.	Windy	True	3	3/9	0,3333
		False	6	6/9	0,6667
		Total	**9**	**9/9**	**1,0**

3. **Hitung Probabilitas Apriori masing-masing fitur untuk $\neg Play$ sebagai berikut dengan jumlah data sebanyak 5 buah.**

Tabel 3. Komputasi Probabilitas masing-masing fitur untuk Label Not Play.

No.	Nama Fitur	Nama Subfitur	Kemunculan	Probabilitas	
				Perbandingan	Angka
1.	Outlook	Rainy	3	3/5	0,6
		Overcast	0	0/5	0
		Sunny	2	2/5	0,4
		Total	**5**	**5/5**	**1,0**
2.	Temperature	Hot	2	2/5	0,4
		Mild	2	2/5	0,4
		Cool	1	1/5	0,2
		Total	**5**	**5/5**	**1,0**
3.	Humidity	High	4	4/5	0,8
		Normal	1	1/5	0,2
		Total	**5**	**5/5**	**1,0**
4.	Windy	True	3	3/5	0,6
		False	2	2/5	0,4
		Total	**5**	**5/5**	**1,0**

4. **Uji Coba pada Data Pelatihan**. Hitung prediksi Play dan Not Play bila diberikan kondisi cuaca: Outlook (Rainy), Temperatur (Mild), Humidity (High), dan Windy (False).

$$P\left(Play\middle|P\big((Outlook|Rainy),(Temp|Mild),(Humidity|High),(Windy|False)\big)\right)$$
$$=\frac{P\big((Outlook|Rainy),(Temp|Mild),(Humidity|High),(Windy|False)\big).P(Play)}{P\big((Outlook|Rainy),(Temp|Mild),(Humidity|High),(Windy|False)\big)}$$

Bila $x_1 = (Outlook|Rainy),\ x_2 = (Temp|Mild), x_3 = (Humidity|High),\ dan\ x_4 = (Windy|False)$, sedangkan $y = Play$ maka persamaan di atas dapat direpresentasikan sebagai berikut:

$$P(y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|y).P(y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto P(y).\prod_{i=1}^{4} P(x_i|y)$$
$$\propto (0{,}64).\big((0{,}2222).(0{,}4444).(0{,}3333).(0{,}6667)\big)$$

$$\propto (0{,}64).(0{,}02194)$$
$$\propto 0{,}01404$$

$$P(\neg y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|\neg y).P(\neg y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto P(\neg y).\prod_{i=1}^{4} P(x_i|\neg y)$$
$$\propto (0{,}36).\big((0{,}6).(0{,}4).(0{,}8).(0{,}4)\big)$$
$$\propto (0{,}36).(0{,}044118)$$
$$\propto 0{,}0276$$

Diperoleh hasil bahwa $P(y|x_1,x_2,x_3,x_4) < P(\neg y|x_1,x_2,x_3,x_4)$ disimpulkan $\neg\boldsymbol{Play}$. Pembandingan dengan data ke-8 (indeks 7) diperoleh hasil yang sama, yakni **No**.

5. **Uji Coba pada Data Uji**. Hitung prediksi Play dan Not Play bila diberikan kondisi cuaca: **Outlook (Overcast), Temperatur (Mild), Humidity (Normal), dan Windy (True)**.

$$P\Big(Play\Big|P\big((Outlook|Overcast),(Temp|Mild),(Humidity|Normal),(Windy|True)\big)\Big)$$
$$= \frac{P\big((Outlook|Overcast),(Temp|Mild),(Humidity|Normal),(Windy|True)\big).P(Play)}{P\big((Outlook|Overcast),(Temp|Mild),(Humidity|Normal),(Windy|True)\big)}$$

Bila $x_1 = (Outlook|Overcast),\ x_2 = (Temp|Mild), x_3 = (Humidity|Normal), dan\ x_4 = (Windy|True)$, sedangkan $y = Play$ maka persamaan di atas dapat direpresentasikan sebagai berikut:

$$P(y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|y).P(y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto P(y).\prod_{i=1}^{4} P(x_i|y)$$
$$\propto (0{,}64).\big((0{,}4444).(0{,}4444).(0{,}6667).(0{,}3333)\big)$$
$$\propto (0{,}64).(0{,}02194)$$
$$\propto 0{,}02808$$

$$P(\neg y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|\neg y).P(\neg y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto P(\neg y).\prod_{i=1}^{4} P(x_i|\neg y)$$
$$\propto (0{,}36).\big((0).(0{,}4).(0{,}2).(0{,}6)\big)$$
$$\propto (0{,}36).(0)$$
$$\propto 0$$

Diperoleh hasil bahwa $P(y|x_1,x_2,x_3,x_4) > P(\neg y|x_1,x_2,x_3,x_4)$ disimpulkan $\boldsymbol{Play}$. Namun hasil ini tampaknya kurang masuk akal karena apakah mungkin bermain golf pada kondisi berangin.

a. **Alternatif 1**. Untuk meyakinkan hasil dari prediksi Naive Bayes, maka dilakukan penyesuaian dengan cara menambahkan satu data pada **semua fitur** baik pada Play maupun Not Play sebagaimana ditunjukkan pada Tabel 4 dan Tabel 5.

Tabel 4. Komputasi Probabilitas masing-masing fitur untuk Label Play dengan penambahan satu data pada semua fitur

No.	Nama Fitur	Nama Subfitur	Kemunculan	Probabilitas	
				Perbandingan	Angka
1.	Outlook	Rainy	3	3/12	0,25
		Overcast	5	5/12	0,4166
		Sunny	4	4/12	0,3333
		Total	**12**	**12/12**	**0,9999**
2.	Temperature	Hot	3	3/12	0,25
		Mild	5	5/12	0,4166
		Cool	4	4/12	0,3333
		Total	**12**	**12/12**	**0,9999**
3.	Humidity	High	4	4/11	0,3636
		Normal	7	7/11	0,6363
		Total	**11**	**11/11**	**0,9999**
4.	Windy	True	4	4/11	0,3636
		False	7	7/11	0,6363
		Total	**11**	**11/11**	**0,9999**

Tabel 5. Komputasi Probabilitas masing-masing fitur untuk Label Not Play dengan penambahan satu data pada semua fitur

No.	Nama Fitur	Nama Subfitur	Kemunculan	Probabilitas	
				Perbandingan	Angka
1.	Outlook	Rainy	4	4/8	0,5
		Overcast	1	1/8	0,125
		Sunny	3	3/8	0,375
		Total	**8**	**8/8**	**1,0**
2.	Temperature	Hot	3	3/8	0,375
		Mild	3	3/8	0,375
		Cool	2	2/8	0,25
		Total	**8**	**8/8**	**1,0**
3.	Humidity	High	5	5/7	0,7142
		Normal	2	2/7	0,2857
		Total	**7**	**7/7**	**0,9999**
4.	Windy	True	4	4/7	0,5714
		False	3	3/7	0,4285
		Total	**7**	**7/7**	**0,9999**

Maka dengan $x_1 = (Outlook|Overcast),\ x_2 = (Temp|Mild), x_3 = (Humidity|Normal),\ dan\ x_4 = (Windy|True)$, sedangkan $y = Play$ diperoleh:

$$P(y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|y).P(y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto (0{,}64).\big((0{,}4166).(0{,}4166).(0{,}6363).(0{,}3636)\big)$$
$$\propto (0{,}64).(0{,}0402)$$
$$\propto 0{,}0257$$

$$P(\neg y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|\neg y).P(\neg y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto (0{,}36).\big((0{,}125).(0{,}375).(0{,}2857).(0{,}5714)\big)$$
$$\propto (0{,}36).(0{,}0075)$$
$$\propto 0{,}0027$$

Diperoleh hasil bahwa $P(y|x_1,x_2,x_3,x_4) > P(\neg y|x_1,x_2,x_3,x_4)$ disimpulkan ***Play***. Dari hasil konfirmasi dengan menambahkan satu data pada masing-masing fitur, diperoleh kesimpulan yang sama yakni ***Play.***

b. **Alternatif 2**. melakukan penyesuaian dengan cara menambahkan satu data **hanya** pada subfitur Outlook baik pada Play maupun Not Play sebagaimana ditunjukkan pada Tabel 6 dan Tabel 7.

Tabel 6. Komputasi Probabilitas masing-masing fitur untuk Label Play dengan penambahan satu data hanya pada fitur Outlook

No.	Nama Fitur	Nama Subfitur	Kemunculan	Probabilitas	
				Perbandingan	Angka
1.	Outlook	Rainy	3	3/12	0,25
		Overcast	5	5/12	0,4166
		Sunny	4	4/12	0,3333
		Total	**12**	**12/12**	**0,9999**
2.	Temperature	Hot	2	2/9	0,2222
		Mild	4	4/9	0,4444
		Cool	3	3/9	0,3333
		Total	**9**	**9/9**	**0,999**
3.	Humidity	High	3	3/9	0,3333
		Normal	6	6/9	0,6667
		Total	**9**	**9/9**	**1,0**
4.	Windy	True	3	3/9	0,3333
		False	6	6/9	0,6667
		Total	**9**	**9/9**	**1,0**

Tabel 7. Komputasi Probabilitas masing-masing fitur untuk Label Not Play dengan penambahan satu data hanya pada fitur Outlook

No.	Nama Fitur		Kemunculan	Probabilitas

		Nama Subfitur		Perbandingan	Angka
1.	Outlook	Rainy	4	4/8	0,5
		Overcast	1	1/8	0,125
		Sunny	3	3/8	0,375
		Total	**8**	**8/8**	**1,0**
2.	Temperature	Hot	2	2/5	0,4
		Mild	2	2/5	0,4
		Cool	1	1/5	0,2
		Total	**5**	**5/5**	**1,0**
3.	Humidity	High	4	4/5	0,8
		Normal	1	1/5	0,2
		Total	**5**	**5/5**	**1,0**
4.	Windy	True	3	3/5	0,6
		False	2	2/5	0,4
		Total	**5**	**5/5**	**1,0**

$$P(y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|y).P(y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto (0{,}64).\big((0{,}4166).(0{,}4444).(0{,}6667).(0{,}3333)\big)$$
$$\propto (0{,}64).(0{,}0411)$$
$$\propto 0{,}0263$$

$$P(\neg y|x_1,x_2,x_3,x_4) = \frac{P(x_1,x_2,x_3,x_4|\neg y).P(\neg y)}{P(x_1,x_2,x_3,x_4)}$$
$$\propto (0{,}36).\big((0{,}125).(0{,}4).(0{,}2).(0{,}6)\big)$$
$$\propto (0{,}36).(0{,}006)$$
$$\propto 0{,}0021$$

Diperoleh hasil bahwa $P(y|x_1,x_2,x_3,x_4) > P(\neg y|x_1,x_2,x_3,x_4)$ disimpulkan ***Play***. Dari hasil konfirmasi dengan menambahkan satu data hanya pada subfitur Outlook, diperoleh kesimpulan yang sama yakni ***Play.***

8. Implementasi menggunakan Python

Contoh data yang digunakan:

Tabel 8. Data untuk Prediksi Cuaca

No	Weather	Temp	Humidity	Windy	Play
1	Rainy	Hot	High	f	no
2	Rainy	Hot	High	t	no
3	Overcast	Hot	High	f	yes
4	Sunny	Mild	High	f	yes
5	Sunny	Cool	Normal	f	yes
6	Sunny	Cool	Normal	t	no
7	Overcast	Cool	Normal	t	yes
8	Rainy	Mild	High	f	no
9	Rainy	Cool	Normal	f	yes

10	Sunny	Mild	Normal	f	yes
11	Rainy	Mild	Normal	t	yes
12	Overcast	Mild	High	t	yes
13	Overcast	Hot	Normal	f	yes
14	Sunny	Mild	High	t	no

Pada contoh ini, data yang digunakan kumpulan data dummy dengan 5 kolom: Weather, Temp, Humidity, Windy, dan Play. Empat kolom pertama adalah fitur (Weather, Temp, Humidity, Windy) dan yang lainnya adalah label.

```
from sklearn import preprocessing
import numpy as np

#Generating the Gaussian Naive Bayes model
from sklearn.naive_bayes import GaussianNB

# Assign features and encoding labels
weather=['Rainy','Rainy','Overcast','Sunny','Sunny','Sunny','Overcast','Rainy',
         'Rainy','Sunny','Rainy','Overcast','Overcast','Sunny']
temp =['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild',
       'Mild','Mild','Hot','Mild']
humidity=['High','High','High','High','Normal','Normal','Normal','High',
          'Normal','Normal','Normal','High','Normal','High']
windy=['f','t','f','f','f','t','t','f','f','f','t','t','f','t']

LabelClass=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
```

Pertama, yang perlu dilakukan adalah merubah label string/text menjadi angka. Misalnya: Weather: 0 : Overcast, 2 : Sunny , 1 : Rainy proses ini disebut sebagai transformasi data kategorik. Scikit-learn telah menyediakan library LabelEncoder untuk melakukan tranformasi variable.

```
# Creating labelEncoder
le = preprocessing.LabelEncoder()
# Converting string labels into numbers.
weather_encoded=le.fit_transform(weather)
hum_encoded=le.fit_transform(humidity)
temp_encoded=le.fit_transform(temp)
wind_encode=le.fit_transform(windy)
label=le.fit_transform(LabelClass)
print(weather_encoded,temp_encoded,hum_encoded,wind_encode,label)
```

Selanjutnya gabungkan ke-4 fitur (Weather, Temp, Humidity, Windy) dalam satu variabel.

```
#Combining weather and humidity in a single tuple as features
features=list(zip(weather_encoded,temp_encoded,hum_encoded,wind_encode))
print (features)
```

Buat model menggunakan pengklasifikasi naive bayes dengan langkah-langkah berikut:

1. Buat naive bayes classifier
2. Latih data training dengan menggunakan method fit()
3. Lakukan prediksi pada data testing

```
#Create a Gaussian Classifier
model = GaussianNB()
model.fit(features,label) #Train the model using training set.

#data test : Sunny, Hot, Normal, False
X_tes=[[2,1,1,0]]

#Predict Output
# ''' For Weather : 0:Overcast, 2:Sunny , 1:Rainy ''' For Humidity : 0:High, 1:Normal
# For temp 0:Cool, 1:Hot, 2:Mild ''' For windy : 0 :f, 1: t
predicted= model.predict(X_tes)
print(" ")
print("today :")
print(predicted) # --> [1] that means yes, the player should bat first and [0] that means No, player should bowl first.

```

Output :

```
[1 1 0 2 2 2 0 1 1 2 1 0 0 2] [1 1 1 2 0 0 0 2 0 2 2 2 1 2] [0 0 0 0 1 1 1 0 1 1 1 0 1 0] [0
[(1, 1, 0, 0), (1, 1, 0, 1), (0, 1, 0, 0), (2, 2, 0, 0), (2, 0, 1, 0), (2, 0, 1, 1), (0, 0, 1

today :
[1]
```

Angka [1] menunjukkan bahwa untuk data test dengan nilai Outlook=sunny, Temp=hot, Humidity=normal, Windy=false menunjukkan bahwa pemain dapat 'bermain' play =yes.

9. Naïve Bayes Gaussian

Pada percobaan ketiga ini, kita akan menggunakan data riil untuk melakukan klasifikasi dengan Naive Bayes. Data yang digunakan adalah **Social_Network_Ads**. Data tersebut menggambarkan usia seseorang dengan pendapatkan yang akan menentukan apakah orang tersebut akan membeli sebuah barang atau tidak. Gaji merupakan data kontinu.

Tahap Persiapan

Pada tahap ini kita akan melakukan beberapa hal,

1. Load data ke dalam data frame
2. Memisahkan fitur dan label
3. Split data untuk training dan testing

```
import numpy as np
import pandas as pd

# Load data CSV
df = pd.read_csv('Social_Network_Ads.csv')

# Cek data
display(df.head())

# Memisahkan fitur dengan label
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

	Age	EstimatedSalary	Purchased
0	19	19000	0
1	35	20000	0
2	26	43000	0
3	27	57000	0
4	19	76000	0

```
# Split data training dan testing

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=30)
```

Training dan Evaluasi Model

```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

# Inisiasi obyek MultinomialNB
gnb = GaussianNB()

# Fit model
# Label y harus dalam bentu 1D atau (n_samples,)
gnb.fit(X_train, y_train)

# Prediksi dengan data training
y_train_pred = gnb.predict(X_train)

# Evaluasi akurasi training
acc_train = accuracy_score(y_train, y_train_pred)

# Prediksi test data
y_test_pred = gnb.predict(X_test)

# Evaluasi model dengan metric akurasi
acc_test = accuracy_score(y_test, y_test_pred)

# Print hasil evaluasi
print(f'Hasil akurasi data train: {acc_train}')
print(f'Hasil akurasi data test: {acc_test}')
```

```
Hasil akurasi data train: 0.8964285714285715
Hasil akurasi data test: 0.8833333333333333
```

Apakah kita telah melakukan hal yang benar?

Perhatikan kembali fitur dari data. Terdapat **Age** dan **Salary**. Selain keduanya merupakan data kontinu, *Age* dan *Salary* memiliki skala yang berbeda. Hal ini "mungkin" dapat menyebabkan kurang akuratnya model dalam memprediksi sebuah kelas. Apakah ini benar?

Pembuktian

Kita akan melakukan percobaan lanjutan dengan menggunakan standarisasi untuk menjawab pertanyaan, apakah kita perlu melakukan hal tersebut pada model Naive Bayes khususnya tipe Gaussian.


```
from sklearn.preprocessing import StandardScaler

# Inisasi obyek StandardScaler
scaler = StandardScaler()

# Standarisasi pada fitur di X_train dan X_test
X_train_std = scaler.fit_transform(X_train)
X_test_std = scaler.transform(X_test)
```

Pertanyaan

Mengapa pada X_test kita tidak perlu melakukan proses fitting?

Jawaban

Proses fitting (fit) akan menyimpan perhitungan berdasarkan data yang dilatih. Pada kasus standarisasi ini, nilai yang disimpan adalah *mean* dan *standar deviasi* dari data **X_train**. Jika kita melakukan proses fitting kembali pada **X_test** maka nilai mean dan standar deviasi akan berdasarkan data X_test. Kita tidak ingin itu terjadi, karena model diharapkan mampu melakukan klasifikasi dengan baik pada data yang tidak diketahui (data test). Oleh karena itu, pada X_test hanya dilakukan proses **transform** agar pada saat pembuatan model, model akan menggunakan nilai mean dan standar deviasi yang sama dengan data training.

```
# Buat obyek GaussianNB lain
gnb_std = GaussianNB()

# Fit dengan data yang telah di standarisasi
gnb_std.fit(X_train_std, y_train)

# Prediksi dengan data training
y_train_std_pred = gnb_std.predict(X_train_std)

# Evaluasi akurasi training data
acc_train_std = accuracy_score(y_train, y_train_std_pred)

# Prediksi test data yang telah di standarisasi
y_test_std_pred = gnb_std.predict(X_test_std)

# Evaluasi akurasi testing data
acc_test_std = accuracy_score(y_test, y_test_std_pred)

# Print hasil evaluasi
print(f'Hasil akurasi data training terstandarisasi: {acc_train_std}')
print(f'Hasil akurasi data testing terstandarisasi: {acc_test_std}')
```

```
Hasil akurasi data training terstandarisasi: 0.9
Hasil akurasi data testing terstandarisasi: 0.8833333333333333
```

Kesimpulan

Jika diperhatikan, tidak terjadi perubahan yang signifikan antara model dari nilai asli dengan model dengan nilai yang telah di standarisasi, terlebih pada hasil dengan menggunakan data test. Hal ini dikarenakan, Naive Bayes bukan jenis algoritma klasifikasi yang mengandalkan jarak, namun probabilitas. Mean dan standar deviasi mungkin berubah, namun probabilitas akan menghasilkan nilai yang sama

10. Naïve Bayes dengan Data Multinomial

Pada percobaan keempat ini, kita akan menggunakan nilai multinomial untuk melakukan klasifikasi dengan Naive Bayes. Nilai multinomial adalah data yang nilainya didapatkan dari proses menghitung. Sehingga, pada konteks fitur, nilai multinomial fitur berdasarkan proses perhitungan (counting) probabilitas kemunculan fitur tersebut dalam sebuah data. Contoh klasik fitur multinomial adalah perhitungan jumlah kata pada klasifikasi teks.Pada percobaan ini, kasus klasifikasi teks diberikan untuk mempermudah pemahaman terhadap algoritma Naive Bayes tipe Multinomial.

Kita akan menggunakan data `spam.csv` yang berisi data teks sms dengan label **spam** dan **ham**. Spam adalah sms sampah, sedangkan ham adalah sebaliknya

Load Data

Pada tahap ini kita akan *loading* data ke dalam data frame dan melakukan inspeksi sederhana untuk memastikan apakah kita perlu proses pra pengolahan data sebelum melakukan ekstraksi fitur dan permodelan

```
import numpy as np
import pandas as pd

df = pd.read_csv('spam.csv', encoding='latin-1') # spesifiksi encoding diperlukan karena data tidak menggunakan UTF-8

df.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Terdapat 3 kolom yang tidak bermanfaat untuk proses selanjutnya, maka kita perlu membuang kolom tersebut. Selain itu, untuk memudahkan pembacaan data, kita juga akan mengubah nama kolom **v1** yang berupa label dan **v2** yang berupa teks sms

Pra Pengolahan Data

Beberapa hal yang akan dilakukan pada tahap ini yaitu,

1. Drop kolom yang tidak digunakan
2. Ubah nama kolom v1 (label) dan v2 (teks sms)
3. Inspeksi Data
4. Encode label
5. Memisahkan fitur dengan label

Drop Kolom

```
# Drop 3 kolom terakhir dengan fungsi iloc
df = df.drop(df.iloc[:,2:], axis=1)

# Cek data
df.head()
```

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Ubah Nama Kolom V1 dan V2

```
# Data untuk rename kolom v1 dan v2
new_cols = {
    'v1': 'Labels',
    'v2': 'SMS'
}

# Rename nama kolom v1 dan v2
df = df.rename(columns=new_cols)

# cek data
df.head()
```

	Labels	SMS
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Inspeksi Data

```
# Cek Jumlah Data Per Kelas
print(df['Labels'].value_counts())
print('\n')

# Cek Kelengkapan Data
print(df.info())
print('\n')

# Cek Statistik Deskriptif
print(df.describe())
```

```
ham     4825
spam     747
Name: Labels, dtype: int64


<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Labels  5572 non-null   object
 1   SMS     5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
None


       Labels                    SMS
count    5572                   5572
unique      2                   5169
top       ham  Sorry, I'll call later
freq     4825                     30
```

Encode Label

```
# Data untuk label
new_labels = {
    'spam': 1,
    'ham': 0
}

# Encode label
df['Labels'] = df['Labels'].map(new_labels)

# Cek data
df.head()
```

	Labels	SMS
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...

Memisahkan Fitur dengan Label

```
X = df['SMS'].values
y = df['Labels'].values
```

Ekstraksi fitur

Ekstraksi fitur untuk setiap SMS akan menggunakan konsep Bag of Words. Kita dapat menggunakan fungsi `CountVectorizer` dari scikit-learn. Akan tetapi untuk mencegah **leaking information** kita akan melakukan split data terlebih dahulu, baru melakukan transformasi terhadap data training dan testing.

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer

# Split data training dan testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=50)

# Inisiasi CountVectorizer
bow = CountVectorizer()

# Fitting dan transform X_train dengan CountVectorizer
X_train = bow.fit_transform(X_train)

# Transform X_test
# Mengapa hanya transform? Alasan yang sama dengan kasus pada percobaan ke-3
# Kita tidak menginginkan model mengetahui paramter yang digunakan oleh CountVectorizer untuk fitting data X_train
# Sehingga, data testing dapat tetap menjadi data yang asing bagi model nantinya
X_test = bow.transform(X_test)
```

Training dan Evaluasi Model

Kita akan menggunakan algoritma Multinomial Naive Bayes. Fungsi `MultinomialNB` dari scikit-learn dapat digunakan pada kasus ini.

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

# Inisiasi MultinomialNB
mnb = MultinomialNB()

# Fit model
mnb.fit(X_train, y_train)

# Prediksi dengan data training
y_pred_train = mnb.predict(X_train)

# Evaluasi akurasi data training
acc_train = accuracy_score(y_train, y_pred_train)

# Prediksi dengan data training
y_pred_test = mnb.predict(X_test)

# Evaluasi akurasi data training
acc_test = accuracy_score(y_test, y_pred_test)

# Print hasil evaluasi
print(f'Hasil akurasi data train: {acc_train}')
print(f'Hasil akurasi data test: {acc_test}')
```

```
Hasil akurasi data train: 0.9946152120260264
Hasil akurasi data test: 0.9775784753363229
```

11. TUGAS PRAKTIKUM

Tugas 1

Buatlah model klasifikasi Multinomial Naïve Baiyes dengan ketentuan:

1) Menggunakan data spam.csv
2) Fitur CountVectorizer dengan mengaktifkan stop_words
3) Evaluasi hasilnya!

Tugas 2

Buatlah model klasifikasi Multinomial Naïve Baiyes dengan ketentuan:

1) Menggunakan data spam.csv
2) Fitur TF_IDF dengan mengaktifkan stop_words
3) Evaluasi hasilnya dan bandingkan dengan hasil tugas 1
4) Berikan kesimpulan fitur mana yang terbaik pada kasus data spam.csv