Dari tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Harap dapat dikumpulkan sebelum batas waktu yang ditentukan!

<u>Tugas 07</u>

Nama	: Juminovario
NIM	: 202420018
Kelas	: MTI 23 Reg-A
MK	: Advanced Database

Pada tutorial ini akan membahas bagaimana cara penggolahan data dengan algoritma apriori dengan tools rapid miner.

Sebelum masuk ke tahap penggolahan data, yang harus disiapkan adalah

- Data yang akan di olah
- Tools rapid miner

Untuk pengolahan data dengan algoritma apriori langkah pertama yang harus dilakukan adalah penetuan atribut, atribut yang di gunakan atau yang akan di pilih harus ada hubungan dan kaitan satu sama lain. Pada tutorial kali ini data yang akan digunakan adalah data penjualan pada sebuah toko furniture dan elektornik dengan 346 record dan 16 atribut tapi yang akan digunakan hanya 3 atribut saja.

Setelah penentuan atribut dilakukan, langkah selanjutnya yaitu tahap preprocessing data, pada tahap ini akan dilakukan beberapa hal, yaitu cleaning data dan transformas data.

Pada tahap cleaning dilakukan pembersihan data transaksi penjualan furniture dan elektronik. Tahap cleaning ini dilakukan dengan cara membuang data yang kosong atau data yang tidak sempurna kemudian membuang field-field yang tidak dibutuhkan agar tidak memperlambat dalam proses asosiasi dana mempercepat mendapatkan hasil pola gabungan. Tahap selanjutnya adalah transformasi data, tahap ini akan dilakukan agar mempermudah saat memasukkan data ke dalam tools rapid miner dengan membuat inisial atau symbol symbol untuk data.

Seperti contoh berikut:

Tabel Inisial item pembelian data transaksi penjualan

No.	Items Pembelian						
1	MC1 (mesin cuci 1 Tempat)						
2	MC2 (mesin cuci 2 Tempat)						
3	SP (speaker aktif)						
4	RP (rak piring)						
5	PFK (palung fadhil kaca)						
6	LED24 (TV uk.24 dengan semua merk)						
7	LED 32 (TV uk.32 dengan semua merk)						
8	LED 43 (TV uk.43 dengan semua merk)						
9	PR (parabola)						
10	DG (digital)						
11	LP2 (lemari pakaian 2 pintu)						
12	LP3 (lemari pakaian 3 pintu)						
13	LH (lemari hias)						
14	LM (lemari mini)						
15	MB (meja belajar)						
16	KR (kursi)						

Setelah dilakukan transformasi data, tahap selanjutnya adalah melakukan tranformasi data penjualan kedalam bentuk tabular. Berikut merupakan hasil transformasi data ke dalam bentuk tabular dapat dilihat pada Tabel:

cam	MC1	MC2	RP	PFK	LED24	LED32		JM
C1	0	0	0	0	0	0		0
C2	0	0	0	0	0	0		0
C3	1	0	0	0	0	0		0
C4	0	1	0	0	0	0		0
C5	0	0	0	0	0	0		0
C6	0	0	0	0	0	0		0
С7	0	0	0	0	0	0		0
	••••		••••	••••	••••	••••		••••
C346	0	0	0	0	1	0	•••••	0
L								

Tabel Tabular atribut item pembelian data transaksi penjualan

Keterangan

- a) Customers merupakan nomor faktur setiap pembeli
- b) MC1, MC2, RP, PFK, LED24 Dll merupakan barang-barang atau produk yang dijual oleh PT.Citra Mustika Pandawa cabang Kerinci.

:

- c) 0 merupakan tanda bahwa barang tersebut tiak dibeli oleh pembeli
- d) 1 merupakan tanda bahwa barang tersebut dibeli oleh pembeli

Setelah melalui tahap preprocessing dan transformasi data, selanjutnya adalah tahap asosiasi dengan menggunakan algoritma apriori untuk menentukan pola pembelian pelanggan. Hasil ini diukur dengan menggunakan nilai Support dan Confidence. Percobaan perhitungan ini menggunakan tools Rapid Miner dengan 346 record data. Dalam tahap imlementasi dengan rapidminer ini ada 3 operator yang digunakan dalam tools, yaitu: Read Exel, Numeric to Binominal dan W-Apriori. Operator pertama adalah Read Exel, operator ini berfungsi untuk tempat data dan akan diinputkan data yang sudah diolah. Karena data yang diolah berupa exel maka operator yang digunakan adalah read Exel. Operator kedua adalah Numeric to Binominal, operator ini berfungsi untuk merubah data yang tadinya numeric menjadi nominal dua nilai pada operator read Exel karena semua atribut dari masukan wajib merupakan bilangan binominal yaitu nilai true/false. Operator ketiga adalah W-Apriori, operator ini berfungsi untuk perhitungan algoritma apriori. Desain dari ketiga operator ini dapat dilihat pada gambar berikut:



Setelah ketiga operator terhubung dan dijalankan maka akan keluar hasil untuk item yang sering muncul dan saling berhubungan, pada data penjualan elektronik dan furniture ini hasil

yang didapat nilai minimal support 4% dan minimal confidance 90%. Hasil yang diperoleh dari rapidminer ada 2 rule yaitu:

- 1. Jika membeli parabola maka akan membeli digital
- 2. Jika membeli LED32 dan parabola maka akan memebeli digital

Best rules found: 1. PR=true 19 ==> DG =true 19 conf:(1) 2. LED32=true PR=true 14 ==> DG =true 14 conf:(1)

Demikian tutorial pengolahan data dengan algoritma apriori dengan tools rapidminer yang sangat sederhana ini, mohon maaf apabila banyak terdapat kesalahan, terima kasih.

Tugas 7

Dari <u>tugas 06</u> sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Jawab :

Untuk pemrograman phyton nya saya menggunkan google colab karena laptop gak mendukung tuk install aplikasi python. Tiba-tiba latop error

Maaf ya Pak

Tutorialnya

No	Outlook	Temperature	Humidity	Windy	Play
1	Sunny	Hot	High	FALSE	No
2	Sunny	Hot	High	TRUE	No
3	Cloudy	Hot	High	FALSE	Yes
4	Rainy	Mild	High	FALSE	Yes
5	Rainy	Cool	Normal	FALSE	Yes
6	Rainy	Cool	Normal	TRUE	Yes
7	Cloudy	Cool	Normal	TRUE	Yes
8	Sunny	Mild	High	FALSE	No
9	Sunny	Cool	Normal	FALSE	Yes
10	Rainy	Mild	Normal	FALSE	Yes
11	Sunny	Mild	Normal	TRUE	Yes
12	Cloudy	Mild	High	TRUE	Yes
13	Cloudy	Hot	Normal	FALSE	Yes
14	Rainy	Mild	High	TRUE	No

Untuk dapat menggunakan Google Colab kita perlu menambahkan ekstensi baru ke Google Drive kita dengan cara klik tombol **New > More > Connect more apps** lalu tuliskan "colab" pada kolom search kemudian klik tombol **connect**.

Membuat Notebook

Jika Colab sudah terintegrasi dengan Drive kita maka kita siap untuk menggunakannya, pertama kita buat direktori baru terlebih dahulu dengan cara klik tombol **New** kemudian pilih **Folder** lalu berikan nama direktori tersebut (misal: Colab) kemudian buat file Notebook baru dengan cara klik kanan pada area kosong didalam direktori yang baru saja kita buat **Klik** kanan > More > Colaboratory.

÷	Upload files				
Ð	Upload folder				
=	Google Docs	>			
Ħ	Google Sheets	>			
	Google Slides	>			
	More	>	:=	Google Forms	>
_		_	•	Google Drawings	
			•	Google My Maps	
				Google Sites	
			co	Colaboratory	
			+	Connect more apps	

Maka tampilan awal dari Notebook kita adalah seperti berikut



Ada baiknya kita berikan nama notebook kita dengan cara double klik pada area Untitled0.ipnyb lalu tuliskan nama notebook seperti berikut



Setup GPU

Karena Colab menyediakan GPU gratis untuk penggunanya kita pun dapat memberikan pengaturan pada notebook untuk menggunakan layanan GPU gratis tersebut caranya klik menu **Edit > Notebook settings** kemudian ubah "Hardware accelerator" menjadi GPU dan kita juga dapat mengubah runtime versi Python pada notebook sedang aktif.

Notebook setting	IS	
Runtime type		
Python 3	Ψ	
Hardware accelerator		
GPU	~	

Menjalankan Kode Pertama

Proses setup sudah selesai dan mari kita mencoba beberapa operasi tipe data dasar, berikut potongan kodenya

```
from sklearn import tree
#datasets
#outlook = 0 is sunny, 1 is overcast, 2 is rainy
#temperature = 0 is hot, 1 is mild, 2 is cool
#humidity = 0 is high, 1 is normal
#isWindy? 0 false : 1 true
#outlook
                   #temperature
                                       #humidity #windy
x = [
    [0,
                       Ο,
                                           Ο,
                                                           0],
    [0,
                       Ο,
                                           Ο,
                                                           1],
```

```
[1,
                         Ο,
                                              Ο,
                                                               0],
    [2,
                         1,
                                              Ο,
                                                               01,
    [2,
                         2,
                                              1,
                                                               0],
    [2,
                                                               1],
                         2,
                                              1,
    [1,
                                              1,
                                                               1],
                         2,
    [0,
                         1,
                                              Ο,
                                                               01,
    [0,
                         2,
                                              1,
                                                               0],
    [2,
                         1,
                                              1,
                                                               0],
    [0,
                         1,
                                              1,
                                                               1],
    [1,
                         1,
                                              Ο,
                                                               1],
    [1,
                         Ο,
                                              1,
                                                               0],
    [2,
                         1,
                                                               1]
                                              Ο,
]
y = [0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(x, y)
def prog() :
    outlook = int(input("Outlook? (0 is sunny, 1 overcast, 2 rainy) : "
))
    humidity = int(input("Temperature? (0 is hot, 1 is mild, 2 is cool)
: "))
    temp = int(input("Humidty? (0 is high, 1 is normal) : "))
    isWindy = int(input("Is Windy? 0 false, 1 true : " ))
    if clf.predict([[outlook, humidity, temp, isWindy]]):
       print("\nYes,I Play golf")
    else:
        print("\nNo Play golf")
```

```
prog()
```

C	CO ^(A) UntitledO.ipynb ☆ File Edit Lihat Sisipkan Runtime Fitur Bantuan <u>Semua perubahan disimpan</u>									
≔	+ Ko	de + Teks								
Q	0	from sklearn	import tree							
<>		<pre>#datasets #outlook = 0 is sunny, 1 is overcast, 2 is rainy #temperature = 0 is hot, 1 is mild, 2 is cool #humidity = 0 is high, 1 is normal #isWindy2 0 false : 1 true</pre>								
		<pre>#outlook x = [</pre>	#temperature	#humidity	#windy					
			0.	0.	01.					
		10,	0.	0,	11.					
		[1,	0,	0,	0],					
		[2,	1,	ο,	0],					
		[2,	2,	1,	0],					
		[2,	2,	1,	1],					
		[1,	2,	1,	1],					
		[0,	1,	Ο,	0],					
		[0,	2,	1,	0],					
		[2,	1,	1,	0],					
		[0,	1,	1,	1],					
		[1,	1,	Ο,	1],					
		[1,	Ο,	1,	0],					
		[2,	1,	Ο,	1]					
]								
		y = [0, 0, 1]	, 1, 1, 0, 1, 0, 1,1,1,1,	1,0]						
		clf = tree.De	ecisionTreeClassifier()							
		clf = clf.fit(x, v)								

Untuk menjalankannya klik runtime / jalankan semuanya. Maka akan muncul seperti ini hasilnya

🝐 Untitled0.ipynb 🛛 😭 CO File Edit Lihat Sisipkan Runtime Fitur Bantuan Semua perubahan disimpan + Kode + Teks ≣ erse: 0 print("\nMales golf") Q prog() $\langle \rangle$ → Outlook? (0 is sunny, 1 overcast, 2 rainy) : 0 Temperature? (0 is hot, 1 is mild, 2 is cool) : 0 Humidty? (0 is high, 1 is normal) : 0 Is Windy? O false, 1 true : 1 Males golf

Selesai.

Nama : Oman Arrohman Nim : 202420042 Kelas : MTI Reguler A MK : Advanced Database

Tugas 7

Dari <u>Tugas 06</u> sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

		<u> </u>		<u> </u>
Hari	Cuaca	Temperatur	Kecepatan Angin	Berolahraga
Hari ke 1	cerah	normal	pelan	ya
Hari ke 2	hujan	tinggi	pelan	tidak
Hari ke 3	cerah	normal	kencang	ya
Hari ke 4	cerah	normal	pelan	ya
Hari ke 5	hujan	tinggi	kencang	tidak
Hari ke 6	hujan	tinggi	pelan	ya
Hari ke 7	cerah	normal	kencang	tidak

Dataset Berolahraga

Tutorial atau tata cara Perhitungan dan Penerapan Algoritma C 4.5 Pada Phyton Dengan Jupyter Notebook

- 1. Pastikan Sudah menginstal Phyton dan Jupyter Notebook
- 2. Siapkan Dataset dengan format CSV
- 3. Pastikan Library pendukung seperti Scikit-Learn, Pydotplus, dan Graphviz, sedangkan untuk pandas dan IPhyton.display sudah tersedia sendiri pada saat menginstal phyton.

Link dibawah untuk download :

Scikit-learn > <u>https://scikit-learn.org/stable/install.html</u> Graphviz >

https://www2.graphviz.org/Packages/stable/windows/10/cmake/Release/x64/ Pydotplus > https://pypi.org/project/pydotplus/

- Untuk graphviz silahkan atur path dan masukan ke system environtment variables sesuai dengan directory penginstalan pada komputer, misal : C:\Program Files\Graphviz 2.44.1\bin;
- Setelah semua selesai lalu masuk ke halaman Jupyter Notebook > pilih File > New > Phyton 3

6. Masukan semua variabel library yang telah di instal seperti gambar berikut :



 Lalu setelah selesai lalu masukan atribut sesuai dataset yang telah dibaut seperti gambar dibawah dan panggil sesuai nama dataset seperti gambar dibawah dengan format csv atau txt.



8. Maka data akan muncul seperti pada gambar dibawah ini.

File	Edit	Viev	v Ins	ert Ce	ell Kernel	Widgets	Help	
+	≫ 4	2	•	↓	Run 🔳 C	Code	▼	
	Out[4]:		Hari	i Cuaca	Temperatur	Kecenatan Angin	Berolahraga	a
		_	Tiai	Cuaca	lemperatur	Reception_Angin	Derolamaga	-
		0	Hari ke 1	cerah	normal	pelan	уа	а
		1	Hari ke 2	2 hujan	tinggi	pelan	tidak	k
		2	Hari ke 3	cerah	normal	kencang	уа	a
		3	Hari ke 4	cerah	normal	pelan	уа	a
		4	Hari ke 5	5 hujan	tinggi	kencang	tidak	k

9. Lalu cara yang selanjutnya yaitu mengubah data menjadi data binner dengan cara pada gambar dibawah kecuali atribut 'berolahraga' karena merupakan atribut target.

<pre>In [6]: olahraga = pd.get_dummies(ds[['Hari', 'Cuaca', 'Temperatur', 'Kecepatan_Angin'] olahraga.head()</pre>]]
--	----

10. Dan maka dataset yang kita masukan sudah berubah menjadi data binner yang hanya berupa angka 1 dan angka 0 seperti gambar dibawah :

File	Edit	View	Insert	Cell	Kernel	Widgets	Help			Trusted	Python 3 C
-	× 2		• •	Run	C	▶ Code					
	Out[6]:	H	Hari_Hari ke 1	Hari_Hari ke 2	Hari_Hari ke 3	Hari_Hari ke 4	Hari_Hari ke 5	Hari_Hari ke 6	Hari_Hari ke 7	Cuaca_cerah	Cuaca_I
		0	1	0	0	0	0	0	0	1	
		1	0	1	0	0	0	0	0	0	
		2	0	0	1	0	0	0	0	1	
		3	0	0	0	1	0	0	0	1	
		4	0	0	0	0	1	0	0	0	

11. Pada Cara terakhir ini yaitu untuk menampilkan decision tree :



12. Maka Tampilan Decision Tree akan muncul seperti gambar dibawah ini :



Nama : Puspita Dewi Setyadi Nim : 202420011 Kelas : MTI 23 Reguler A

Langkah 1. Menyiapkan data pelatihan

Seret data Titanic Training dari Sampel repositori ke dalam proses Anda.

Repository ×	Process ×	
Add Data ≡ ▼	Process	100% 🔑 🔑 📴 🛃 💷 🔛
Ripley-Set (v1) Sonar (v1) Titanio (v1) Titanic Training (v1) Titanic Onrabeled (v1) Transactions (v1) Weighting (v1) e processes G Templates	Process Retrieve Titanic Training Dinp C out Decision Tree	Naive Bayes Rule Induction

Kita telah menyiapkan data Pelatihan Titanic untuk model pelatihan: dalam artian tidak ada nilai yang hilang serta labelnya telah didefinisikan. Harap diingat bahwa label adalah atribut yang ingin kita prediksi , dalam kasus ini : selamat(survived). kita memerlukan data pelatihan dengan label yang dikenal sebagai masukan untuk metode pembelajaran mesin semacam ini. Inilah sebabnya mengapa kita menyebut metode pembelajaran yang diawasi (*supervised learning*)

Langkah 2. Membangun tiga model yang berbeda.

Process Operators X 100% 🔎 🔑 📮 🏹 📝 Process deci × Modeling (8) Process -Predictive (8) Retrieve Titanic Trai... Decision Tree 🕶 📒 Trees (8) C out tra mod) inp Decision Tree exa Random Forest Naive Gradient Boosted Trees Bayes ID3 Rule Decision Stump Induction Decision Tree (Multiway Decision Tree (Weight-E) Random Tree

1. Seret pada operator **Decision Tree** dan hubungkan ke port "**out**" dari **Retrieve Titanic Training**.

2. Seret di operator **Naive Bayes** dan hubungkan port input sampel set nya dengan output "exa" dari **Decision Tree**.



3. Seret ke operator **Rule Induction** dan hubungkan contohnya dengan set port input dengan output "exa" dari **Naive Bayes**.



4. Hubungkan port "mod" dari operator pemodelan ke port hasil "res" di kanan, lalu jalankan prosesnya.

5. Periksa tiga model yang berbeda.

Hasil prediksi dari Rule Model



Hasil prediksi dari Naive Bayes



Hasil prediksi dari Decision Tree (Pohon Keputusan)



Pohon keputusan dengan jelas menunjukkan bahwa ukuran keluarga lebih menentukan daripada kelas penumpang untuk wanita. Pola perilaku ini tidak bisa dideteksi untuk pria. Secara umum, pria memiliki kemungkinan lebih rendah untuk bertahan hidup ("dahulukan wanita dan anak "). Cara termudah untuk melihat hal ini adalah pada visualisasi Chart model Naive Bayes. Meski biasanya Naive Bayes bukan tipe model yang paling akurat, secara umum aturan yang ditetapkan adalah format yang mudah dibaca, di mana bisa berguna saat kita ingin menafsirkan model.

Tugas 07

Advanced Database

Dari tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya. Harap dapat dikumpulkan sebelum batas waktu yang ditentukan!

No	Tester	Tang Crimping	RJ45	Kabel UTP
1	1	1	1	1
2	0	1	1	1
3	1	0	0	1
4	1	0	1	0
5	1	1	0	1
6	0	1	1	1
7	1	1	1	0

Data Perangkat Praktik Jaringan

Berdasarkan data perangkat disamping untuk praktik jaringan maka langsung saja dilakukan proses *Association Rule* dengan menggunakan Bahasa pemograman *Python*, dimana dalam proses tersebut saya menggunakan platform yang disediakan google yaitu *Google Collab Research*.

Step 1. Install plugin Mlxtend

Proses ini dilakukan untuk load library yang dibutuhkan dalam proses Association Rule.

Association Rule.ipynb ☆ File Edit View Insert Runtime Tools Help <u>All changes saved</u>	Comment	📇 Share	\$	R
+ Code + Text	V RAM Disk	- 🖍 E		
[] ! pip install mlxtend Requirement already satisfied: mlxtend in /usr/local/lib/python3.6/dist-packages (0.14.0)				
Requirement already satisfied: scipy>=0.17 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.4.1) Requirement already satisfied: matplotlib=1.5.1 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (3.2.2) Requirement already satisfied: pandas>=0.17.1 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.22.2.post1) Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.22.2.post1) Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.22.2.post1) Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.8.5) Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from mlxtend) (50.3.2) Requirement already satisfied: pythoradetutib=2.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib=-1.5.1->mlxt Requirement already satisfied: kuis010er>=0.10 in /usr/local/lib/python3.6/dist-packages (from matplotlib=-1.5.1->mlxtend) Requirement already satisfied: kuis010er>=0.10 in /usr/local/lib/python3.6/dist-packages (from pandas>=0.17.1->mlxtend) (2018. Requirement already satisfied: jobhlb>=0.11 in /usr/local/lib/python3.6/dist-packages (from scikit-learn>=0.18->mlxtend) (2018. Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from scikit-learn>=0.18->mlxtend) (0 Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from scikit-learn>=0.18->mlxtend) (0 Req	olotlib>=1.5.1-> end) (2.8.1) 10.0) (1.3.1) 9) 17.0) 5.1->mlxtend) (mlxtend) (2 1.15.0)	.4.7)	
[] ! pip install xird				
Requirement already satisfied: xlrd in /usr/local/lib/python3.6/dist-packages (1.1.0)	¹			

Proses Instalasi Plugin Mlxtend

Step 2. Import Library & Datasets

Proses selanjut nya yaitu mengimport library Transaction Encoder, Apriori, Association Rule dan

datasets yang akan digunakan.

0	<pre>import pandas as pd from mlxtend.preprocessing import TransactionEncoder from mlxtend.frequent_patterns import apriori from mlxtend.frequent_patterns import association_rules</pre>	^ ↓	/ ©	╡\$\$	₽ :	
	datasets = [[['Tester','Tang Crimping','RJ45', 'Kabel UTP'], ['Tang Crimping', 'RJ45', 'Kabel UTP'], ['Tester', 'RJ45', 'Kabel UTP'],					
	['Tester', 'RJ45'], ['Tester', 'Tang Crimping','Kabel UTP'], ['Tang Crimping', 'RJ45', 'Kabel UTP'], ['Tester', 'Tang Crimping', 'RJ45']]					

Proses Import Library & Datasets

https://colab.research.google.com/drive/1HDctIu8DCxPVsydjEH28Sd_pkmdRP52H?usp=sharing

Step.3 Transaction Encoder

Proses ini digunakan untuk menemukan semua itemset yang memenuhi minimum support.

te te_ df fre	= Transact _array = te = pd.Dataf equent_iter	tionEncoder() e.fit(datasets).transform Frame(te_array, columns=t msets = apriori(df, min_s
fre	equent_iter	nsets
	support	itemsets
	0.714286	(Kabel UTP)
	0.857143	(RJ45)
2	0.714286	(Tang Crimping)
3	0.714286	(Tester)
4	0.571429	(Kabel UTP, RJ45)
5	0.571429	(Kabel UTP, Tang Crimping)
6	0.571429	(Tang Crimping, RJ45)
7	0.571429	(Tester, RJ45)



Step.4 Rule Generation & Selection Criteria

Proses ini digunakan untuk mengekstrak semua aturan yang memiliki high-confidence dari itemsets

yang ditemukan dari langkah sebelumnya.

[6]]	ass	ociation_rules(frequent_items	ets, metric="confid	ence", min_th	hreshold=(9.6)					
Đ		antecedents	consequents	antecedent support	t consequent	support	support	confidence	lif	t levera	ge convicti	ion
	0	(Kabel UTP)	(RJ45)	0.714286	6	0.857143	0.571429	0.800000	0.93333	3 -0.0408	16 0.7142	286
	1	(RJ45)	(Kabel UTP)	0.857143	3	0.714286	0.571429	0.666667	0.93333	3 -0.0408	16 0.8571	143
	2	(Kabel UTP)	(Tang Crimping)	0.714286	5	0.714286	0.571429	0.800000	1.12000	0 0.0612	24 1.4285	571
	3	(Tang Crimping)	(Kabel UTP)	0.714286	6	0.714286	0.571429	0.800000	1.12000	0 0.0612	24 1.4285	571
	4	(Tang Crimping)	(RJ45)	0.714286	6	0.857143	0.571429	0.800000	0.93333	3 -0.0408	16 0.7142	286
	5	(RJ45)	(Tang Crimping)	0.857143	3	0.714286	0.571429	0.666667	0.93333	3 -0.0408	16 0.8571	143
	6	(Tester)	(RJ45)	0.71428	6	0.857143	0.571429	0.800000	0.93333	3 -0.0408	16 0.7142	286
	7	(RJ45)	(Tester)	0.857143	3	0.714286	0.571429	0.666667	0.93333	3 -0.0408	16 0.8571	143
[41]	rul rul	es = associatio es	on_rules(frequer	nt_itemsets, metric	="lift", min_	_threshold	i=1)					
		antecedents	consequents	antecedent support	consequent	support	support	confidence	lift l	everage	conviction	
	0	(Kabel UTP)	(Tang Crimping)	0.71428	6	0.714286	0.571429	0.8	1.12 (0.061224	1.428571	
	1	(Tang Crimping)	(Kabel UTP)	0.714286	;	0.714286	0.571429	0.8	1.12 (0.061224	1.428571	
[38]	rul rul	.es["antecedent_ .es	len"] = rules['	'antecedents"].appl	y(lambda x:]	len(x))						
		antecedents	consequents	antecedent support	t consequent	support	support	confidence	lift l	everage	conviction	antecedent_len
	0	(Kabel UTP)	(Tang Crimping)	0.71428	6	0.714286	0.571429	0.8	1.12 (0.061224	1.428571	
	1	(Tang Crimping)	(Kabel UTP)	0.714286	3	0.714286	0.571429	0.8	1.12 (0.061224	1.428571	1

Proses Rule generation \$ Selection Criteria

TUGAS 07

APRIORI - PHYTON



Dibuat Oleh Robby Prabowo 202420001

Dosen Pengampu

TRI BASUKI KURNIAWAN, S.Kom, M.Eng, Ph.D

Program Pasca Sarjana Universitas Binadarma Palembang 2020/2021

Step 1: Import the libraries



Catatan penulis : pada Collabs modul apyori harus diinstall dulu menggunakan perintah

```
!pip install apyori
```

Step 2: Load the dataset

#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('<u>/content/drive</u>')
store_data = pd.read_csv('drive/My Drive/Colab Notebooks/data-sheet-tugas-7-ok.csv', header=None)

Step 3: Have a glance at the records



Output

Q	0	sto	ore_data	Э												
	C→	Dri	ve alre	eady m	ounted at ,	/content	/drive;	to att	empt to	forcibly	remount,	call	drive.mo	unt("/c	ontent/o	lrive",
<>			0	1	2	3	4	5	6							
-		0	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP							
		1	PENA	ROTI	MENTEGA	NaN	NaN	NaN	NaN							
		2	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN							
		3	NaN	NaN	NaN	TELUR	BUNCIS	SUSU	NaN							
		4	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN							
		5	NaN	ROTI	MENTEGA	TELUR	NaN	SUSU	KECAP							

Step 4: Look at the shape

● (← →	■ / △ > C	ː ː ͡ O ː ː ː ː ː ː ː ː ː ː ː ː ː ː ː ː	G (G ()) / / / G	€ ☆ \$
C	File	tgs-7.ipynb ☆ Edit View Insert Runtime Tools Help	Comment	🕻 Share 🏾 🇱
:=	+ Coo	le + Text	✓ RAM Disk ■	🖍 Editing
	0	store_data		
Q		#look shape store_data.shape		
<>		Drive already mounted at /content/drive; to attempt to forcibly remount, call dr	ive.mount("/conten	nt/drive", for
		(6, 7)		

Step 5: Convert Pandas DataFrame into a list of lists

● (■ ← →	I / ▲ : O (] t; X ▲ (] Z (] G : ∞ : ▲ (] ▷ 1 □ 1 G i G i G i 0 1 G i Z 1 ₽ / ∞ 1 G i G i ⊙ : G C ■ colab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrolITo=CxEbfdtVR_hn	ارد (• ا • ا • ا • • • • • • • • • • • • •
CO	▲ tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	🔲 Comment 🛛 👫 Share 🏟
=	+ Code + Text	✓ RAM Disk Cliting
Q <>	<pre>#converting pandas dataframe records = [] for i in range (0,6): records.append([str(store_data.values[i,j]) for j in range(0,7)])</pre>	
	Drive already mounted at /content/drive; to attempt to forcibly remount, call dr	ive.mount("/content/drive", fo

Step 6: Build the Apriori model



Step 7: Print out the number of rules

) 0 () ← →	■ / 🍐 : Î 🖸 < Î t × Î 🍐 < Î 🖉 : Î Ġ : Î 🐵 : Î 🖾 : Î 📾 : Î ঊ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î Ġ : Î ⊕ : Î ⊕ : Î Ġ : Î Ġ : Î Ġ : Î ⊕ : Î ⊕ : Î ↔ : Î ⊕ : Î ↔ : Î ⊕ : Î ↔ : Î ⊕ : Î ↔ : Î ⊕ : Î ↔ : Î ⊕ : Î ↔ : I ⊕ : Î ↔ : I ⊕ : I ↔ : I ↔ : I ↔ : I ↔ : I ⊕ : I ↔ :	G (+
CC	→ tgs-7.ipynb ☆ ■ Comment	🚓 Share 🌣 🐠
≣ Q \$	+ Code + Text RAM Disk RAM DIs	 Editing
	Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/cont 40	cent/drive", force_rem

Step 8: Have a glance at the rule

0 (◙ ↓ ً ▲ ⊧ Î Q └] ײַ× Ĩ ▲ ‹] ﷺ ↓ G ⊧ Ĩ ∞ ፣ Ĩ ▲ ‹] ∞ ፣ Ĩ ◘ ፣ Ĩ G ፣ Ĩ G ፣ Ĩ @ ፣ Ĩ G ‹] ℋ ፣ Ĩ ₽ ↓ Ĩ ₩ ፣ Ĩ G · Ĩ ⊗ ፣ Ĩ G	G i G i B i G i H	x
~	C Calab.research.google.com/drive/1_6JvzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	० 🛧 🛸 🚷	:
C	● tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	📮 Comment 🛛 🙁 Share 🌣 🚳	
≣	+ Code + Text	✓ RAM Disk ✓ Paint Editing	`
Q	<pre>#Lauching at the 1st rule print(association_results)</pre>		Γ
<>	Drive already mounted at /content/drive; to attempt to forcibly remount, call dr.	<pre>vive.mount("/content/drive", force_r vive.mount("/content/drive", force_r</pre>	em
	<pre>[kelalionkecord(luems=trozensel({ mENIEGA }), support=0.8333333333333334, ordered </pre>	a_statistics=[orderedstatistic(item	s_ ▶

Script Phyton Lengkap

```
#import
import numpy as np
import pandas as pd
from apyori import apriori
#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('/content/drive')
store data = pd.read csv('drive/My Drive/Colab Notebooks/data-sheet-
tugas-7-ok.csv', header=None)
#having glance of record
store data
#look shape
store data.shape
#converting pandas dataframe
records = []
for i in range (0, 6):
  records.append([str(store data.values[i,j]) for j in range(0,7) ])
#Build the apriori model
association rules=apriori(records,min support=0.20, min confidence=0.60
, min lift=1,min length=2)
association results=list(association rules)
#Print out the number of rules
print(len(association results))
#Lauching at the 1st rule
```

print(association_results)

Nama : Shabila Fitri Aulia Nim : 202420024 Kelas : MTI A 23 MK : Advanced Databased

Tugas 07

Dari tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python,

lalu buat tutorialnya.

Harap dapat dikumpulkan sebelum batas waktu yang ditentukan!

Jawaban :

Dari Tugas 6 didapatkan data sebagai berikut :

TID	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP
001	1	1	1	0	0	0	0
002	0	1	1	1	0	0	0
003	0	0	0	1	1	1	0
004	0	1	1	0	0	0	0
005	0	1	1	1	0	1	1

Dari data diatas akan saya coba untuk membuat association rule meggunakan bahasa pyhton dengan colab.research.google.com, adapun langkahnya sebagai berikut :

Langkah 1,

Download Libabry Association Rule



Langkah 2,

Masukkan data mengenai assosiation rule yang akan diproses (data terlampir diatas), Tentukan

minimum supportnya pada kali ini saya menggunakan 0.6

```
File Edit View Insert Runtime Tools Help All changes saved
```

```
+ Code + Text
```

```
[ ] ! pip install xlrd
```

Requirement already satisfied: xlrd in /usr/local/lib/python3.6/dist-packages (1.1.0)

Association Rules Generation from Frequent Itemsets

Source: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/

Langkah 3,

Tentukan nilai rules sesuai yang diinginkan :

```
rules[ (rules['antecedent_len'] >= 2) &
    (rules['confidence'] > 0.75) &
    (rules['lift'] > 1.2) ]
```

Langkah 4,

Kemudian RUN All dari seluruh config yang sudah ada, adapun hasilnya sebagai berikut :

C⇒		support	itemsets								
	0	0.8	(Mentega)								
	1	0.8	(Roti)								
	2	0.6	(Telur)								
	3	0.8 (F	Roti, Mentega)								
[]]	as	sociation_r	ules (frequer	nt_itemsets, metric	="confidence", min	_thresho	ld=0.7)				
		antecedent	s consequer	nts antecedent supp	port consequent su	pport su	upport conf	idence	e lift l	leverage co	nviction
	0	(Ro	ti) (Mente	ega)	0.8	0.8	0.8	1.0	0 1.25	0.16	inf
	1	(Menteg	a) (R	Roti)	0.8	0.8	0.8	1.(0 1.25	0.16	inf
	ass	ociation_rul	es(frequent_i	itemsets, metric="co	nfidence", min_thres	shold=0.7)				
									_		
		antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	
	0	antecedents (Roti)	(Mentega)	antecedent support 0.8	Consequent support 0.8	support 0.8	Confidence	11ft 1.25	leverage 0.16	conviction	
	0 1	(Roti) (Mentega)	(Mentega) (Roti)	antecedent support 0.8 0.8	0.8	0.8	1.0	1.25 1.25	0.16 0.16	conviction inf	
[]	0 1 rul rul	(Roti) (Mentega) es = associa es	(Mentega) (Roti) tion_rules (fr	antecedent support 0.8 0.8 requent_itemsets, me	consequent support 0.8 0.8 tric="lift", min_thr	0.8 0.8 reshold=1	.2)	1.25 1.25	0.16 0.16	inf	
[]	0 1 rul rul	antecedents (Roti) (Mentega) es = associa es antecedents	(Mentega) (Roti) tion_rules(fr	antecedent support 0.8 0.8 requent_itemsets, me antecedent support	consequent support 0.8 0.8 tric="lift", min_thi consequent support	0.8 0.8 reshold=1 support	confidence	11ft 1.25 1.25 1.17	leverage 0.16 0.16 leverage	conviction inf conviction	
[]	0 1 rul rul	antecedents (Roti) (Mentega) es = associa antecedents (Roti)	consequents (Mentega) (Roti) tion_rules (fr consequents (Mentega)	antecedent support 0.8 0.8 requent_itemsets, me antecedent support 0.8	consequent support 0.8 0.8 tric="lift", min_th: consequent support 0.8	support 0.8 0.8 reshold=1 support 0.8	confidence 1.0 1.0 .2) confidence 1.0	11ft 1.25 1.25 11ft 1.25	1everage 0.16 0.16 1everage 0.16	conviction inf conviction inf	
[]	0 1 rul rul 0 1	antecedents (Roti) (Mentega) es = associa es (Roti) (Mentega)	consequents (Mentega) (Roti) tion_rules (fr consequents (Mentega) (Roti)	antecedent support 0.8 0.8 requent_itemsets, me antecedent support 0.8 0.8	consequent support 0.8 0.8 tric="lift", min_thr consequent support 0.8 0.8	support 0.8 0.8 reshold=1 support 0.8 0.8	confidence 1.0 1.0 1.0 1.0 1.0 1.0 1.0	11ft 1.25 1.25 11ft 1.25 1.25	leverage 0.16 0.16 leverage 0.16 0.16	conviction inf conviction inf	
[]]	0 1 rul rul 0 1 rul rul	<pre>antecedents (Roti) (Mentega) es = associa antecedents (Roti) (Mentega) es ["antecede es</pre>	<pre>consequents (Mentega) (Roti) consequents (Mentega) (Roti) (Roti) cnt_len"] = ru </pre>	antecedent support 0.8 0.8 requent_itemsets, me antecedent support 0.8 0.8 0.8 0.8 0.8 0.8 0.8	<pre>consequent support</pre>	support 0.8 0.8 reshold=1 support 0.8 0.8 0.8 (x))	confidence 1.0 1.0 1.0 1.0 1.0 1.0 1.0	11ft 1.25 1.25 1.1ft 1.25 1.25	1everage 0.16 0.16 1everage 0.16 0.16	conviction inf conviction inf	
	0 1 rul rul 1 rul rul	antecedents (Roti) (Mentega) es = associa es = associa es (Roti) (Mentega) es ["antecedents antecedents	<pre>consequents (Mentega) (Roti) .tion_rules(fr consequents (Mentega) (Roti) .nt_len"] = ru consequents</pre>	antecedent support 0.8 0.8 requent_itemsets, me antecedent support 0.8 0.8 ules["antecedents"]. antecedent support	consequent support 0.8 0.8 tric="lift", min_thi consequent support 0.8 0.8 0.8 0.8 0.8 0.8	support 0.8 0.8 reshold=1 support 0.8 0.8 (x)) support	confidence 2) confidence 1.0 1.0 1.0 2.0 1.0 1.0 1.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2	111ft 1.25 1.25 1.25 1.25 1.25	leverage 0.16 0.16 leverage 0.16 0.16 0.16	conviction inf conviction inf inf conviction	antecedent_le
	0 1 rul 0 1 rul rul 0	antecedents (Roti) (Mentega) es = associa es = associa antecedents (Roti) (Mentega) es ["antecedents antecedents (Roti)	<pre>consequents (Mentega) (Roti) .tion_rules(fr consequents (Mentega) (Roti) .nt_len"] = ru consequents (Mentega)</pre>	antecedent support 0.8 0.8 requent_itemsets, me antecedent support 0.8 0.8 ules["antecedents"]. antecedent support 0.8	<pre>consequent support</pre>	support 0.8 0.8 reshold=1 support 0.8 0.8 (x)) support 0.8	confidence 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0	111ft 1.25 1.25 1.1 1.25 1.25 1.25 1.25 1.25	leverage 0.16 0.16 1everage 0.16 0.16 1everage 0.16	conviction inf conviction inf inf conviction	antecedent_le

Didaptkan kesimpulan bahwa Assosiation rule sebagaimana diatas untuk hasilny.

Nama : Siti Ratu Delima Nim :202420025 Kelas :MTI 23

TID	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP
001	1	1	1	0	0	0	0
002	0	1	1	1	0	0	0
003	0	0	0	1	1	1	0
004	0	1	1	0	0	0	0
005	0	1	1	1	0	1	1

Dari data diatas akan saya coba untuk membuat association rule meggunakan bahasa pyhton dengan colab.research.google.com, adapun langkahnya sebagai berikut :

Langkah 1,

Download Libabry Association Rule

[] ! pip insta	11 mlxtend
Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement	<pre>already satisfied: mlxtend in /usr/local/lib/python3.6/dist-packages (0.14.0) already satisfied: pandas>=0.17.1 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.1.4) already satisfied: stuptools in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.post)) already satisfied: stuptool.31 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.post)) already satisfied: mumpy>=1.0.4 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.) already satisfied: python=0.17 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (2.1.5) already satisfied: python=0.17 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (2.1.5) already satisfied: python=0.17 in /usr/local/lib/python3.6/dist-packages (from padas>=0.17.1>mlxtend) (2.8.1) already satisfied: python=0.11 in /usr/local/lib/python3.6/dist-packages (from mathend) (0.17.0) already satisfied: python=0.11 in /usr/local/lib/python3.6/dist-packages (from mathend) (0.1.3.1) already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1>-mlxtend) (1.3.1) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1>-mlxtend) (1.3.0) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from mplotlib>=1.5.1>-mlxtend) (1.15.0) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1>-mlxtend) (1.15.0) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1>-mlxtend) (1.15.0) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from mplotlib>=1.5.1>-mlxtend) (1.15.0) already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from m</pre>

Langkah 2,

Masukkan data mengenai assosiation rule yang akan diproses (data terlampir diatas), Tentukan

minimum supportnya pada kali ini saya menggunakan 0.6

```
[ ] ! pip install xlrd
Requirement already satisfied: xlrd in /usr/local/lib/python3.6/dist-packages (1.1.0)
```

Association Rules Generation from Frequent Itemsets

Source: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/

Langkah 3,

Tentukan nilai rules sesuai yang diinginkan :

```
rules[ (rules['antecedent_len'] >= 2) &
    (rules['confidence'] > 0.75) &
    (rules['lift'] > 1.2) ]
```

Langkah 4,

Kemudian RUN All dari seluruh config yang sudah ada, adapun hasilnya sebagai berikut :

C*		support	itemsets									
	0	0.8	(Mentega)									
	1	0.8	(Roti)									
	2	0.6	(Telur)									
	3	0.8 (Roti, Mentega)									
[]	as	sociation_	rules(freque	nt_itemsets, met	tric="confid	ence", min	_thresho	ld=0.7)				
		anteceden	ts conseque	nts antecedent	support con	sequent su	pport su	upport conf	idence	e lift]	leverage co	nviction
	0	(Ro	oti) (Mente	ega)	0.8		0.8	0.8	1.0) 1.25	0.16	inf
	1	(Menteg	ga) (F	Roti)	0.8		0.8	0.8	1.0) 1.25	0.16	inf
E 1	ass	ociation rul	les (frequent	itemsets, metric=	="confidence"	, min thres	shold=0.7)				
-		-				· _		·				
	-	antecedents	consequents	antecedent supp	ort conseque	nt support	support	confidence	lift	leverage	conviction	
	0	(Roti)	(Mentega)		0.8	0.8	0.8	1.0	1.25	0.16	inf	
	1	(Mentega)	(Roti)		0.8	0.8	0.8	1.0	1.25	0.16	inf	
[]	rul rul	es = associa es	ation_rules(f	requent_itemsets,	, metric="lif	t", min_thi	reshold=1	.2)				
		antecedents	consequents	antecedent supp	ort conseque	nt support	support	confidence	lift	leverage	conviction	
	0	(Roti)	(Mentega)		0.8	0.8	0.8	1.0	1.25	0.16	inf	
	1	(Mentega)	(Roti)		0.8	0.8	0.8	1.0	1.25	0.16	inf	
[]	rul rul	es["antecede es	ent_len"] = r	ules["antecedents	s"].apply(lam	bda x: len	(x))					
		antecedents	consequents	antecedent supp	ort conseque	nt support	support	confidence	lift	leverage	conviction	antecede
	0	(Roti)	(Mentega)		0.8	0.8	0.8	1.0	1.25	0.16	inf	
	1	(Mentega)	(Roti)		0.8	0.8	0.8	1.0	1.25	0.16	inf	

Didaptkan kesimpulan bahwa Assosiation rule sebagaimana diatas untuk hasilny.

TUTORIAL DECISION TREE MENGGUNAKAN PYTHON

Dataset yang akan kami gunakan aadalah dataset untuk menentukan status kelayakan penerima beasiswa yang terdiri dari 2 kelas yaitu kelas Ya Layak dan Tidak Layak menerima beasiswa serta terdiri dari 6 atribut/feature antara lain:

- 1. Jenis Tinggal (JT)
- 2. Alat Transportasi (AT)
- 3. Pekerjaan Ayah (PKA)
- 4. Penghasilan Ayah (PHA)
- 5. Pekerjaan Ibu (PKI)
- 6. Penghasilan Ibu (PHI)

Berikut merupakan dataset yang telah melalui proses preprocessing dan siap untuk diproses kedalam teknik klasifikasi. Dataset yang digunakan sebanyak 240 record yang akan dibagi menjadi data training dan data testing dengan menggunakan teknik hold-out 70%-30%, dimana masing-masing data training berjumlah sebesar 70% dari total data yaitu 168 record dan data testing berjumlah 72 record dengan persentase 30% dari keseluruhan data.

No	Nama	JK	JT	AT	PKA	PHA	PKI	PHI	Status
									Kelayakan
1	S-1	L	0,500	1,000	0,083	0,625	0,833	1,000	Tidak
2	S-2	L	0,250	0,143	0,667	0,375	0,833	1,000	Tidak
3	S-3	L	0,250	1,000	0,667	0,500	0,833	1,000	Tidak
4	S-4	L	0,250	0,286	0,500	0,250	0,833	1,000	Tidak
5	S-5	L	0,250	0,000	0,000	0,500	0,917	0,875	Tidak
6	S-6	Р	0,250	1,000	0,333	0,500	0,083	0,875	Tidak
7	S-7	L	0,250	0,286	0,667	0,250	0,833	1,000	Tidak
8	S-8	L	0,250	0,000	0,083	0,500	0,833	1,000	Tidak
9	S-9	L	0,250	0,286	0,917	0,375	0,833	1,000	Tidak
10	S-10	Р	0,250	1,000	0,083	0,625	0,667	0,625	Tidak
•••	•••	•••	•••	•••	•••	•••	•••	•••	
168	S-168	Р	0,250	0,571	0,083	0,625	0,667	0,625	Tidak

Data Training yang digunakan

Data Testing yang digunakan

No	Nama	JK	JT	AT	РКА	РНА	PKI	PHI	Status Kelavakan
1	S-169	L	0,250	0,143	0,917	0,500	0,500	1,000	Tidak
2	S-170	Р	0,250	1,000	0,667	0,750	0,667	0,625	Tidak
3	S-171	Р	0,250	1,000	0,917	0,500	0,833	1,000	Tidak
4	S-172	L	0,250	1,000	0,083	0,625	0,833	1,000	Tidak
5	S-173	L	0,250	1,000	1,000	0,625	1,000	0,500	Tidak
6	S-174	L	0,250	0,000	0,917	0,500	0,833	1,000	Tidak
7	S-175	L	0,250	1,000	0,333	0,500	0,833	1,000	Ya
8	S-176	L	0,250	1,000	0,917	0,500	0,833	1,000	Tidak
9	S-177	Р	0,250	1,000	0,083	0,875	0,833	1,000	Ya
10	S-178	L	0,250	1,000	0,917	0,500	0,833	1,000	Tidak
•••	•••	•••	•••	•••	•••	•••	•••	•••	•••
72	S-240	Р	0,250	1,000	0,917	0,625	0,833	1,000	Tidak

• Jika data training dan data testing telah tersedia, kemudian buka lembaran kerja pada jupyter notebook lalu import beberapa library yang diperlukan, sebagaimana dibawah ini.

```
In [1]: import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt
```

import pandas as pd import numpy as np from sklearn.preprocessing import LabelEncoder from sklearn.metrics import accuracy_score, f1_score from sklearn.tree import DecisionTreeClassifier from sklearn.tree import plot_tree

import matplotlib.pyplot as plt

3. Kemudian, import data yang akan digunakan sebagaimana dibawah ini

Pada output yang dihasilkan dapat dilihat bahwa terdapat 2 dataset dengan masingmasing berjumlah 168 record & 10 row yang merupakan data training serta data dengan 72 record dengan 10 row yang merupakan data testing.

Kemudian, tentukan label/kelas yang digunakan sebagaimana dibawah ini,

```
target_encoder = LabelEncoder()
df_ho_train['Target'] = target_encoder.fit_transform(df_ho_train['Status Kelayakan'])
df_ho_train['Target'] = target_encoder.fit_transform(df_ho_train['Status Kelayakan'])
df_ho_test['Target'] = target_encoder.transform(df_ho_test['Status Kelayakan'])
```

 Dari 10 row yang ada, tidak semua row/kolom dijadikan sebagai atribut sebagaimana penjelasan sebelumnya, agar hanya kolom atribut yang diproses maka kolom yang berfungsi sebagai label/kelas atau kolom Nomor, id, serta row pelengkap perlu didrop terlebih dahulu sebagaimana dibawah ini.

```
in [6] * be_train - df_be_train.dep(['Inter Enlephan', 'Do', 'Enar', 'Integet', 'A'), exteni)
* be_train = df_ho_train.drop(['Inter Enlephan', 'To', 'Enar', 'Integet', 'A'), exteni)
* ho_train = df_ho_train.drop(['Status Kelayakan', 'No', 'Nama', 'Target', 'J
K'], axis=1)
y_ho_train = df_ho_train['Target']
```

```
x_ho_test = df_ho_test.drop(['Status Kelayakan', 'No', 'Nama', 'Target', 'JK'
], axis=1)
y ho test = df ho test['Target']
```

 Kemudian, input model algoritma decision tree untuk memproses data sekaligus melihat akurasi yang dihasilkan dengan menggunakan algoritma tersebut

Dapat dilihat pada gambar diatas, bahwa Algoritma Decision Tree menggunakan Hold-out dengan persentase 70:30 menghasilkan akurasi sebesar 84.72%

 Untuk melihat visualisasi dari hasil algoritma decision tree dengan menggunakan plot tree dapat dilihat pada gambar dibawah ini,



clf = DecisionTreeClassifier(max_depth = 3) clf.fit(x_ho_train, y_ho_train) tree.plot_tree(clf)); fn=['AT','JT','PHA','PHI','PKA','PKI']
Proses diatas akan menghasilkan plot tree dari algoritma Decision Tree untuk kasus status kelayakan beasiswa sebagaimana gambar berikut.



Tugas 07

Dari tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Harap dapat dikumpulkan sebelum batas waktu yang ditentukan !

Nama	: Vero Faloris
Nim	: 202420032
Kelas	: MTI 23 Reguler A
Mk	: Advanced Database

LINK RUJUKAN :

https://www.academia.edu/7712860/Belajar Data Mining dengan R apidMiner

https://medium.com/@ksnugroho/menerapkan-model-machinelearning-pada-rapidminer-142259846e13

Cari beberapa tutorial yang membahas pengolahan data menggunakan metode prediksi pada tool rapidminer. Silahkan buat ringkasan tutorialnya kembali dengan menggunakan dataset yang kamu buat sendiri. Tuliskan dalam format ms word dan sertakan semua sumber rujukan tutorial yang anda gunakan.

Dalam tutorial ini, kita akan membuat tiga model klasifikasi yang berbeda untuk data Titanic kita: pohon keputusan (decision tree),seperangkat aturan (a set of rules), dan model Bayes. Kita akan menjelajahi model-model tersebut dan melihat apakah kita bisa mengetahui lebih banyak tentang peristiwa kecelakaan itu dan lebih memahami siapa yang memiliki kesempatan terbaik untuk bertahan hidup.

Langkah 1. Menyiapkan data pelatihan

Seret data Titanic Training dari Sampel repositori ke dalam proses Anda.

Repository ×		Process ×	
+ Add Data	= •	Process	100% 🔎 🔑 📮 逢 💣 🔃
Ripley-Set (v1) Sonar (v1) Titanic (v1) Titanic Training (v Titanic Omabeled Transactions (v1) Weighting (v1) e processes G Templates	(v1) (v1)	Process Retrieve Titanic Training Dinp out Decision Tree	Naive Bayes Rule Induction

Kita telah menyiapkan data Pelatihan Titanic untuk model pelatihan: dalam artian tidak ada nilai

yang hilang serta labelnya telah didefinisikan. Harap diingat bahwa label adalah atribut yang ingin kita prediksi , dalam kasus ini : selamat(survived). kita memerlukan data pelatihan dengan label yang dikenal sebagai masukan untuk metode pembelajaran mesin semacam ini. Inilah sebabnya mengapa kita menyebut metode pembelajaran yang diawasi (*supervised learning*)

Langkah 2. Membangun tiga model yang berbeda.

Process Operators X 100% 🔎 🔑 📮 🏹 📝 Process deci × Modeling (8) Process -Predictive (8) Retrieve Titanic Trai... Decision Tree 🕶 📒 Trees (8) C out tra mod) inp Decision Tree exa Random Forest Naive Gradient Boosted Trees Bayes ID3 Rule Decision Stump Induction Decision Tree (Multiway Decision Tree (Weight-E) Random Tree

1. Seret pada operator **Decision Tree** dan hubungkan ke port "**out**" dari **Retrieve Titanic Training**.

2. Seret di operator **Naive Bayes** dan hubungkan port input sampel set nya dengan output "exa" dari **Decision Tree**.



3. Seret ke operator **Rule Induction** dan hubungkan contohnya dengan set port input dengan output "exa" dari **Naive Bayes**.



4. Hubungkan port "mod" dari operator pemodelan ke port hasil "res" di kanan, lalu jalankan prosesnya.

5. Periksa tiga model yang berbeda.

Hasil prediksi dari Rule Model



Hasil prediksi dari Naive Bayes



Hasil prediksi dari Decision Tree (Pohon Keputusan)



Pohon keputusan dengan jelas menunjukkan bahwa ukuran keluarga lebih menentukan daripada kelas penumpang untuk wanita. Pola perilaku ini tidak bisa dideteksi untuk pria. Secara umum, pria memiliki kemungkinan lebih rendah untuk bertahan hidup ("dahulukan wanita dan anak "). Cara termudah untuk melihat hal ini adalah pada visualisasi Chart model **Naive Bayes**. Meski biasanya Naive Bayes bukan tipe model yang paling akurat, secara umum aturan yang ditetapkan adalah format yang mudah dibaca, di mana bisa berguna saat kita ingin menafsirkan model.

Nama : Wahyu Putra Adi Wibowo

Nim : 202420041

```
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
```

Pada kolom name diperlukan pada saat loading dataset tidak mempunyai header/ bisa menggunakan set **header** = None, pada kasus ini untuk data setyang berformat csv sudah memiliki header

diag.head()

	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	visual blurring	Itching	Irritability	delayed healing	partial paresis	muscle stiffness	Alopecia	Obesity	
0	40	1	0	1	0	1	0	0	0	1	0	1	0	1	1	1	Po
1	58	1	0	0	0	1	0	0	1	0	0	0	1	0	1	0	Po
2	41	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	Po
3	45	1	0	0	1	1	1	1	0	1	0	1	0	0	0	0	Po
4	60	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	Po

Catatan Output :

- Gender 1 = Male ,
- 0 = Female ,
- Other attribute 1 = Yes,
- 0 = No

Note : Terdapat dua cara dalam pemilihan attribute, yaitu bisa menggunakan feature_cols & label (comment)/index.

Variable feature_cols akn digunakan saat menggambar sebuah grafis

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training and 30% test
```

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()
clf_tree = DecisionTreeClassifier(criterion='Entropy', random_state=1)
```

Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

Accuracy : 0.9423076923076923

Mengunakan modul graphviz utk menampilkan hasil dari penganalisa data Output :



Catatan:

- Label = indikasi Diabetes
- Age : Umur
- Gender : Jenis Kelamin
- Polyuria : sering buang air kecil
- Polydispia : sering merasa haus
- Sudden Weight loss : Berat Badan turun tiba-tiba

- Weakness : Lemah
- Polyphagia : **banyak makan (sering lapar)**
- Genital Thrush : gatal di kemaluan
- Visual blurring : Penglihatan Buram
- Itching : Gatal
- Irritability : gampang marah
- Delayed Healing : Sulit sembuh (luka)
- Partial Paresis : lumpuh sebagian
- Muscle stiffness : otot kaku
- Alopecia : **kebotakan, rambut rontok**
- Obesity : Kegemukan

NAMA : WIDIA ASTUTI

NIM : 202420021

MATA KULIAH : ADVANCED DATABASE

TUGAS 7

Dari tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Jawab :

angkah-langkah

- 1. Install Python 3.8
- 2. Install Anaconda (www.anaconda.com)

	NDA NAVIGATOR			i Upgrade Now Sign in to An
🕇 Home	Applications on openCV	v Channels		
Environments	•	\$	\$	\$
🗳 Learning	Jupyter		×	\circ
Community	Notebook 6.1.4 Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.	PyCharm 2020.2.3 Full-feetured Python IDE by JetBrains. Supports code completion, linking, debugging, and domain-specific enhancements for web development and data science.	VS Code 1.51.1 Streamlined code editor with support for development operations like debugging, task running and version control.	CMD.exe Prompt 0.1.1 Run a cmd.exe terminal with your current environment from Navigator activated
	Launch	Launch	Launch	install 🗘
Documentation		lab	<u></u>	O
Developer Blog	Glueviz 1.00 Multidimensional data visualization across files. Explore relationships within and among related datasets.	JupyterLab 2.2.6 An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.	Orange 3 3.26.0 Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows	Powershell Prompt 0.0.1 Run a Powershell terminal with your current environment from Navigator activated

- Buat environment baru "OpenCV→pilih python 3.8
- Klik "openCV" → open library
- Buka google →pip.py→numpy→copy to clipboad→kembali ke environmet→paste→enter
- Buka google →pip.py→pandas→copy to clipboad→kembali ke environmet→paste→enter

C:\Windows\system32\cmd.exe	
Downloading tgdm-4.51.0-py2.py3-none-any.whl <70 kB>	
Collecting keyring>=15.1 Downloading keyring=21.5.0-py3-none-any.whl (32 kB) Collection wewste>>2.0	
Downloading requests -2.25.0-py2.py3-none-any.whl (61 kB)	
Collecting readme-renderer>=21.0 Downloading readme_renderer=28.0-py2.py3-none-any.whl (15 kB)	
Requirement already satisfied: colorama>=0.4.3 in c:\users\sony\appda python\python38\site-packages (from twine-)pyloco>=0.0.134-)matplot) Collecting numin32-ctumes*=0.1 0 *=0.1 1: sue nlafform == "win22"	ta\roaming\ (0.4.3)
Downloading pywin32_ctypes-0.2.0-py2.py3-none-any.whl (28 kB) Collecting idna(3,)=2.5	
Downloading idna-2.10-py2.py3-none-any.whl (58 kB)	
Downloading chardet-3.0.4-py2.py3-none-any.wh1 (133 kB)	
Collecting urllib3(1.27,>=1.21.1 Downloading urllib3-1.26.2-py2.py3-none-any.whl (136 kB)	
Requirement already satisfied: certifi>=2017.4.17 in c:\users\sony\.c pencv\lib\site-packages <from requests="">=2.20->twine->pyloco>=0.0.134-</from>	onda\envs\o >matplot> <
2020.6.20) Requirement already satisfied: Pygments>=2.5.1 in c:\users\sony\.cond cu\lih\site-packages (from readme-renderer>=21.0->tuine->puloco>=0.0.	a∖envs∖open 134->matplo
t) (2.7.2) Requirement already satisfied: bleach>=2.1.0 in c:\users\sony\.conda\	envs\opencv
\Lb\site-packages (from readme-renderer>=21.0->twine->pyloco>=0.0.13 (3.2.1) Collecting docutils>=0.13.1	4->matplot>
Downloading docutils-0.16-py2.py3-none-any.whl (548 kB)	
Requirement already satisfied: packaging in c:\users\sony\appdata\roa \python38\site-packages {from bleach>=2.1.0->readme-renderer>=21.0->t o\=0 0 134-\mathbf{20 4}	ming∖python wine->pyloc
B) Store aready satisfied: webencodings in c:\users\sony\.conda\e lib\site-packages (from bleach>=2.1.0->readme-renderer>=21.0->twine-> 134-bmarulot) (0.5 1)	nvs∖opencv∖ pyloco>=0.0
Building wheels for collected packages: typing, ushlex, SimpleWebSock Building wheel for typing (setup.py) done	etServer
Created wheel for typing: filename=typing-3,7,4,3-py3-none-any.whl sha256=358afe168638321caa31a4a77f71c8957d55c83b56a69c58d82d0e75c38f7e Stowed in dimetonu: c:\usevconubsymmidte\User1\usevconubsymbolsym	size=26312 b0 e\5d\01\308
3e091b57809dad979ea543def62d9d878950e3e74f0c930 Building wheel for ushlex (setup.py) done	6 /211 /01 /200
Created wheel for ushlex: filename=ushlex=0.99.1-py3-none-any.whl s a256=c10b5ae4e2f8983faa3d572a79c99ed2dc06203ef67f6ce33f8ffce131dd7a4	ize=4419 sh
actived in directory: C: (users)songvappdata(local)pi)(cache wheels)d actiddig618418c757b09c0b55dcb07bb3409a6780a127 Building wheel for SimpleWebSocketServer (setum.ny) done	1 /1 0 /01 /143
Created wheel for SimpleWebSocketServer: filename=SimpleWebSocketSe py3-none-any.whl size=9498 sha256=d8c78ae4ac619059ef0f98c7a6ed3a32621 DirebJoc0cftbccoc	ruer-0.1.1- 1511766370a
Stored in directory: c:\users\sony\appdata\local\pip\cache\wheels\0 920848bf7bbbf1e1f5328113363458e229efe2e2325522	2\28\f3\c97

- 3. Install jupyter
 - Klik launch→pilih documents→new→folder →rename"latihan"
 - Pilih "latihan" \rightarrow new \rightarrow python 3
 - Kembali ke" documents/latihan"→klik "untitled.ipynb"→shutdown
 - Pilih untitled.ipynb→rename→coba
 - Start (note:run → shift enter

	'notebooks/Documents/latihan/coba.ipynb	
💭 Jupyter 🧃	coba Last Checkpoint: 37 minutes ago (unsaved changes)	P Logout
File Edit V	fiew Insert Cell Kernel Help	Trusted Python 3
🖹 🕇 💥 🖄	The second s	
In [3]:	#library import numpy as np import pandas as np	

4. Kembali ke Jupyter

Ketik :

- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt
- from sklearn.linier_model import linearRegression
- from sklearn.model_selection import train_test_split
 → run
- dataku=pd.read_csv("data_kredit.csv")
- print (dataku)
 →run

)	Untitled Last Checkp	oint: an nour ago	(autosaved)	
ile Edit V	/iew Insert Cell	Kernel	lelp	Trusted 🖋 Python 3 C
+ % @	10 🛧 🔸 🕨 Ru	n 🔳 C 🇯	Code 🔻 🖾	
In [1]:	<pre>import numpy as np import pandas as p import matplotlib. from sklearn.linie from sklearn.model</pre>	d pyplot as pl r_model impo _selection in	t linearRegression port train_test_split	
In [9]:	dataku-pd.read_csv print (dataku)	("data_kredi	.csv")	
	JENIS USAHA LA	NCAR MACET	MAKSIMUM KREDIT	
	1 RESTORAN	20 5	250000000	
	2 JASA SEWA	10 2	10000000	
	3 PERTANIAN 4 PERIKANAN	100 50 70 30	25000000 50000000	
τυ []:				

Sumber :

https://www.youtube.com/watch?v=AnIU-QHyUXE

https://www.youtube.com/watch?v=HQaCF2F_MRo

https://www.youtube.com/watch?v=Xr7 zL0jyQY

TUGAS 07

APRIORI (MARKET BASCET ANALYSIS) - PHYTON



Dibuat Oleh

Aan Novrianto

Dosen Pengampu

TRI BASUKI KURNIAWAN, S.Kom, M.Eng, Ph.D

Program Pasca Sarjana

Universitas Binadarma Palembang

2020/2021

Step 1: Import the libraries



Catatan penulis : pada Collabs modul apyori harus diinstall dulu menggunakan perintah

```
!pip install apyori
```

Step 2: Load the dataset

```
#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('/content/drive')
store_data = pd.read_csv('drive/My Drive/Colab Notebooks/data-sheet-tugas-7-ok.csv', header=None)
```

Step 3: Have a glance at the records

<>	<pre>#having glance of record</pre>	
	store_data	
	#look shape	
	store_data.shape	

Output

Q	0	sto	re_data	9													
	C⇒	Dri	ve alre	eady mo	ounted at /	/content	/drive;	to att	empt to	forcibly	remount,	call	driv	e.moun	t("/co	ontent/	drive",
<>			0	1	2	3	4	5	6								
- I		0	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP								
		1	PENA	ROTI	MENTEGA	NaN	NaN	NaN	NaN								
		2	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN								
		3	NaN	NaN	NaN	TELUR	BUNCIS	SUSU	NaN								
		4	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN								
		5	NaN	ROTI	MENTEGA	TELUR	NaN	SUSU	KECAP								

Step 4: Look at the shape



Step 5: Convert Pandas DataFrame into a list of lists

0 (│ ॼ	G (G ()) I P I G () +
÷	C lacolab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	@ ☆ ;
C	O ▲ tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	🗐 Comment 🛛 👫 Share 🏟
≣	+ Code + Text	✓ RAM Disk ✓ Editing
Q	<pre>#converting pandas dataframe records = [] for i in range (0,6):</pre>	
<>	records.append([str(store_data.values[i,j]) for j in range(0,7)])	
	Drive already mounted at /content/drive; to attempt to forcibly remount, call d	rive.mount("/content/drive", fo
	4	

Step 6: Build the Apriori model

0	(SA /	៸៲៱៵៲៝៝៝៝៝៝៝៶៲៵៵៓៓៱៶៝៝៝៝៝៝៝៝៝៝៵៸៓៙៵៲៓៓៓៱៶៓៲៙៰៶៓៝ឨ៲៶៓៙៝៶៓៝៝៝៝៝៝៝៝៝៵៲៓៙៸៲៓៓៓៓៓៓៓៓៓៓៓៓៓៓៓៓៓៓៓	G (G () 🖗 () 🖉 () G (+	
\leftarrow	\rightarrow C	C a colab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn		⊕ ☆	🖈 🚷 E
C	0	▲ tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help <u>All changes saved</u>	🔲 Comment 🛛 🚢	Share 🏚	٩
≣	+	Code + Text records.append([str(store_data.vaiues[1,]]) tor] in range(0,/)])	✓ RAM Disk ▼	🖍 Editing	^
Q		<pre>#Build the apriori model association_rules=apriori(records,min_support=0.20, min_confidence=0.60, min_lif</pre>	t=1,min_length=2)		
<>		association_results=list(association_rules)			
		Drive already mounted at /content/drive; to attempt to forcibly remount, call draw $\ensuremath{\boldsymbol{\varepsilon}}$	ive.mount("/content	/drive", fo	rce_rem

Step 7: Print out the number of rules



Step 8: Have a glance at the rule

0 (ا ﷺ ہُ اُ کی آ کے دار ﷺ کا کہ اُ اُ ﷺ کا کہ اُ کے دار ہے اُ کے دار ﷺ کا کہ کا کہ کا کہ کا کہ کہ اُ	• (G () • (/ #) G () + • • • • • • • • •
\leftarrow	C Colab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	० 🖈 🛸 🐌 :
C	O Logs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	📮 Comment 🛛 🙁 Share 🌣 🔮
≣	+ Code + Text	✓ RAM Disk → Fediting ∧
Q	<pre>#Lauching at the 1st rule print(association_results)</pre>	
$\langle \rangle$	Drive already mounted at /content/drive; to attempt to forcibly remount, call de 40	rive.mount("/content/drive", force_rem
	<pre>[RelationRecord(items=frozenset({'MENTEGA'}), support=0.8333333333333333334, orderor </pre>	ed_statistics=[OrderedStatistic(items_

Script Phyton Lengkap

```
#import
import numpy as np
import pandas as pd
from apyori import apriori
#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('/content/drive')
store data = pd.read csv('drive/My Drive/Colab Notebooks/data-sheet-
tugas-7-ok.csv', header=None)
#having glance of record
store_data
#look shape
store_data.shape
#converting pandas dataframe
records = []
for i in range (0, 6):
  records.append([str(store data.values[i,j]) for j in range(0,7) ])
#Build the apriori model
association rules=apriori(records,min support=0.20, min confidence=0.60
, min lift=1,min length=2)
association_results=list(association_rules)
#Print out the number of rules
print(len(association results))
```

#Lauching at the 1st rule
print(association_results)

NAMA : AHMAD ALI MA'MUN NIM : 202420037

Tutorial Algoritma K-Nearest Neighbours Dengan Python

K-Nearest Neighbours (KNN)

KNN adalah algoritma pembelajaran mesin yang diawasi yang dapat digunakan untuk menyelesaikan masalah klasifikasi dan regresi. Prinsip KNN adalah nilai atau kelas suatu titik data yang ditentukan oleh titik data di sekitar nilai tersebut.

Untuk memahami algoritma klasifikasi KNN seringkali paling baik ditunjukkan melalui contoh. Tutorial ini akan mendemonstrasikan bagaimana Anda dapat menggunakan KNN dengan Python dengan masalah klasifikasi Anda sendiri. Notebook Jupyter yang sesuai dengan contoh ini dapat ditemukan di sini, jika Anda ingin mengikutinya.

Algoritme prediksi menghitung jarak dari titik x yang tidak diketahui, ke semua titik dalam data Anda. Titik-titik dalam data Anda kemudian diurutkan dengan menambah jarak dari x. Prediksi dilakukan dengan memprediksi label mayoritas dari titik terdekat 'K'.

Memilih K akan memengaruhi kelas tempat poin baru akan ditetapkan.

Dalam contoh di bawah ini, memilih nilai K 2 akan menetapkan titik yang tidak diketahui (lingkaran hitam) ke kelas 2. Namun, jika nilai K adalah 7, titik yang tidak diketahui akan ditetapkan ke kelas 1.





Membuat dataset palsu

Pertama, kita mengimpor pustaka yang kita butuhkan, dan kemudian membuat dataset palsu menggunakan fungsi makeblobs dari sklearn. Kita dapat mengirimkan jumlah sampel, fitur dalam kumpulan data kita, berapa banyak pusat atau kelas yang akan memasukkan data, dan terakhir deviasi standar dari cluster tersebut. Untuk konsistensi antara beberapa jalannya notebook Jupyter ini, saya telah menetapkan integer 101 ke parameter random_state.

Catatan, untuk memulai, kita akan memiliki standar deviasi cluster yang besar. Ini akan memperkenalkan varians ke dalam klasifikasi, yang dapat kita perbaiki nanti, dengan secara khusus memilih nilai K yang optimal. Ini dapat dicapai dengan menggunakan metode siku.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import make blobs
data = make_blobs(n_samples=300, n_features=5, centers=2, cluster_std=6.0, random_state=101)
data
(array([[ -0.95757537,
                        3.36332609, -15.54675979, -14.02967497,
           1.50545246],
                       -0.86726927, -19.42687054, -22.99153445,
        [-11.12008037,
          12.8409123 ],
        [ 5.02786886, -2.84037069, -5.9094317, -16.29765383,
           7.77075032],
        [ -8.02114181,
                         2.29827056, -13.80731349, -10.89022536,
           1.99399904],
        [ 10.87670302.
                         3.25562702, -6.25095388, -0.92884525,
           8.18286695],
        [ 7.86530195, -11.18764669, 6.36417619, -2.87676038,
           1.31626729]]),
 array([0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
        1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0,
        1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0,
        1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1,
        0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1,
```

Fungsi makeblobs dari sklearn mengembalikan tupel 2 elemen. Kita dapat membuat kerangka data fitur kita menggunakan pd.DataFrame, dan meneruskan indeks tupel pertama yang sesuai dengan data fitur. Elemen kedua dari tupel data sesuai dengan label fitur.

df_feat = pd.DataFrame(data[0], columns=['feature_' + str(i) for i in range(1, 6)])

df_feat.head(2)

feature_1 feature_2 feature_3 feature_4 feature_5

0	-0.957575	3.363326	-15.546760	-14.029675	1.505452
1	-11.120080	-0.867269	-19.426871	-22.991534	12.840912

y = data[1]

у

Sekarang kita dapat menskalakan data dengan mengimpor MinMaxScaler dari Sklearn.preprocessing. Tidak seperti algoritme pembelajaran mesin lainnya, kami menyesuaikan dan mengubah semua data pelatihan, sebelum melakukan pemisahan pengujian kereta kami.



Algoritme prediksi dan pengoptimalan

Untuk mengimplementasikan prediksi dalam kode, kita mulai dengan mengimpor

KNeighboursClassifier dari sklearn.neighbours. Kami kemudian membuat instance dari

KNeighboursClassifier, dengan meneruskan argumen 1 ke n_neighbours, dan menetapkan ini ke variabel knn. Nilai yang diteruskan ke n_neighbours mewakili nilai K.

Kami kemudian menyesuaikan dengan data pelatihan, sebelum membuat prediksi, dengan

memanggil metode prediksi pada objek KNeighboursClassifier kami.

Sekarang kita dapat menilai keakuratan prediksi menggunakan klasifikasi_report dan konfusi_matriks.

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=1)

knn.fit(X_train, y_train)

predictions = knn.predict(X_test)

predictions

array([0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1])

Metrik tersebut menunjukkan bahwa akurasinya sudah sangat baik. Ini mungkin karena fakta bahwa kami membuat kumpulan data dengan makeblobs dan secara khusus meminta 2 pusat. Namun, kami sengaja menempatkan nilai yang besar untuk standar deviasi cluster untuk memperkenalkan varians. Hal ini mengakibatkan kesalahan klasifikasi pada 4 poin dalam kumpulan data kami.

<pre>print(confusion_matrix(y_test, predictions))</pre>	
<pre>print(classification_report(y_test, predictions))</pre>	

[[42 1] [3 44]]				
	precision	recall	f1-score	support
0	0.93	0.98	0.95	43
1	0.98	0.94	0.96	47
accuracy			0.96	90
macro avg	0.96	0.96	0.96	90
weighted avg	0.96	0.96	0.96	90

Meningkatkan Akurasi Data

Kami dapat mencoba untuk meningkatkan akurasi hasil kami dengan memodifikasi jumlah tetangga. Ini dapat dicapai dengan menggunakan metode siku.

Pertama-tama kita melakukan iterasi melalui 40 nilai tetangga, membuat instance objek KNeighboursClassifier dengan jumlah tetangga tersebut. Kami kemudian dapat menyesuaikan data pelatihan dengan model KNN ini, mendapatkan prediksi, dan menambahkan nilai rata-rata antara prediksi, pred_i dan nilai yang benar, y_test. Jika pred_i dan y_test tidak cocok dalam array, nilai true dikembalikan yang memiliki nilai 1. Semakin tinggi angkanya, semakin tidak akurat klasifikasi tersebut.

Nilai yang lebih rendah untuk tingkat kesalahan akan sesuai dengan model yang berkinerja lebih baik.

Hasil ini dapat diplot menggunakan rentang nilai i pada sumbu x, versus tingkat kesalahan pada sumbu y.



Sekarang kita dapat memilih nilai K terendah yang akan menghasilkan, tingkat kesalahan terendah. Di sini, kita bisa memilih 5.



Sekarang kita dapat, menjalankan kembali penilaian akurasi dengan matriks kebingungan dan laporan klasifikasi sekali lagi, untuk melihat apakah kita mengklasifikasikan 4 poin yang tidak selaras dengan lebih akurat. Kami telah meningkatkan, dari 4 poin yang salah diklasifikasikan menjadi 2.

```
knn = KNeighborsClassifier(n neighbors=5)
knn.fit(X train, y train)
predictions = knn.predict(X test)
print(confusion matrix(y test, predictions))
print(classification report(y test, predictions))
[[42 1]]
 [ 1 46]]
              precision
                           recall f1-score
                                               support
                             0.98
           0
                   0.98
                                        0.98
                                                    43
           1
                   0.98
                             0.98
                                        0.98
                                                    47
                                                    90
                                        0.98
    accuracy
                   0.98
                                        0.98
                                                    90
   macro avg
                             0.98
weighted avg
                             0.98
                                        0.98
                   0.98
                                                    90
```

Pelatihan tentang titik data baru

Sekarang kita dapat membuat titik data menggunakan data asli. Pertama kita membuat dua dataframe; satu dengan fitur dan satu lagi dengan label, menggabungkannya menjadi satu kerangka data, dan memilih baris pertama, sebagai titik data untuk memprediksi labelnya. Kita harus ingat untuk menskalakan titik data karena model dilatih pada data yang diskalakan.

Prediksi menunjukkan bahwa data titik 1, akan memberikan label 0, yang sesuai dengan titik dataset asli, diverifikasi dengan memanggil df.head (1).

```
1 features = pd.DataFrame(data=data[0], columns=['feature_' + str(i) for i in range(1, 6)])
2 lables = pd.DataFrame(data[1], columns=['labels'])
3 dataset = pd.concat([features, lables], axis=1)
4 data_point_1 = scaler.transform(np.array(dataset.iloc[0][:-1]).reshape(-1, 5))
5 knn.predict(data_point_1)[0]
6
7 # Output
8 # 0
```

feature_1 feature_2 feature_3 feature_4 feature_5 labels

0 -0.957575 3.363326 -15.54676 -14.029675 1.505452 **0**

data_point_1 = scaler.transform(np.array(dataset.iloc[0][:-1]).reshape(-1, 5))

data_point_1

array([[0.3943128, 0.58849094, 0.1949979, 0.21287012, 0.54980842]])

knn.predict(data_point_1)[0]

0

TUGAS 7 ADVANCED DATABASE

Dari <u>tugas 06</u> sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Jawab :

Untuk dapat menggunakan Google Colab kita perlu menambahkan ekstensi baru ke Google Drive kita dengan cara klik tombol **New > More > Connect more apps** lalu tuliskan "colab" pada kolom search kemudian klik tombol **connect**.

• Membuat Notebook

Jika Colab sudah terintegrasi dengan Drive kita maka kita siap untuk menggunakannya, pertama kita buat direktori baru terlebih dahulu dengan cara klik tombol **New** kemudian pilih **Folder** lalu berikan nama direktori tersebut (misal: Colab) kemudian buat file Notebook baru dengan cara klik kanan pada area kosong didalam direktori yang baru saja kita buat **Klik kanan > More > Colaboratory**.





Maka tampilan awal dari Notebook kita adalah seperti berikut

Ada baiknya kita berikan nama notebook kita dengan cara double klik pada area **Untitled0.ipnyb** lalu tuliskan nama notebook seperti berikut.



• Setup GPU

Karena Colab menyediakan GPU gratis untuk penggunanya kita pun dapat memberikan pengaturan pada notebook untuk menggunakan layanan GPU gratis tersebut caranya klik menu **Edit > Notebook settings** kemudian ubah "Hardware accelerator" menjadi GPU dan kita juga dapat mengubah runtime versi Python pada notebook sedang aktif.

Notebook settings		
Runtime type		
Python 3	~	
Hardware accelerator		
GPU	~	
Omit code cell output v	when saving this i	notebook
	CANCEL	SAVE

• Menjalankan Kode Pertama

Proses setup sudah selesai dan mari kita mencoba beberapa operasi tipe data dasar, berikut potongan kodenya

from sklearn import tree

#datasets #outlook = 0 is sunny, 1 is overcast, 2 is rainy #temperature = 0 is hot, 1 is mild, 2 is cool #humidity = 0 is high, 1 is normal #isWindy? 0 false : 1 true #outlook #temperature #humidity #windy $\mathbf{x} = \begin{bmatrix} \\ \end{bmatrix}$ [0, 0, 0, 0], [0, 0. 0. 1], [1, 0. 0. 0], 0], [2, 1, 0. [2, 2, 0], 1, [2, 2, 1, 1], [1, 2, 1, 1], [0, 0], 1. 0. 0], [0, 2, 1, [2, 0], 1, 1, [0, 1], 1, 1. [1, 1, 0. 1], [1, 0, 1, 0], [2, 1, 0, 1] 1 y = [0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0]clf = tree.DecisionTreeClassifier() clf = clf.fit(x, y)def prog() : outlook = int(input("Outlook? (0 is sunny, 1 overcast, 2 rainy) : ")) humidity = int(input("Temperature? (0 is hot, 1 is mild, 2 is cool) : ")) temp = int(input("Humidty? (0 is high, 1 is normal) : ")) isWindy = int(input("Is Windy? 0 false, 1 true : ")) if clf.predict([[outlook, humidity, temp, isWindy]]): print("\nYes,I Play golf") else: print("\nNo Play golf")

prog()

CO La ContitledO.ipynb 🛱 File Edit Lihat Sisipkan Runtime Fitur Bantuan <u>Semua perubahan disimpan</u>										
≣	+	Kode	e + Teks							
Q	(D	from sklearn i	mport tree						
¢	<pre>#datasets #outlook = 0 is sunny, 1 is overcast, 2 is rainy #temperature = 0 is hot, 1 is mild, 2 is cool #humidity = 0 is high, 1 is normal #isWindy? 0 false : 1 true</pre>									
			#outlook x = [#temperature	#humidity	#windy				
			۲0.	0.	0.	01.				
			[0,	0.	0,	11,				
			[1,	0.	0,	01,				
			[2,	1,	0,	0],				
			[2,	2,	1,	0],				
			[2,	2,	1,	1],				
			[1,	2,	1,	1],				
			[0,	1,	Ο,	0],				
			[0,	2,	1,	0],				
			[2,	1,	1,	0],				
			[0,	1,	1,	1],				
			[1,	1,	Ο,	1],				
			[1,	0,	1,	0],				
			[2,	1,	Ο,	1]				
			1							
			y = [0, 0, 1,	1, 1, 0, 1, 0, 1,1,1,1,	,1,0]					
			clf = tree.Dec	isionTreeClassifier()						
			clf = clf.fit	(x. v)						

Untuk menjalankannya klik runtime / jalankan semuanya. Maka akan muncul seperti ini hasilnya

C)	A File	UntitledO.ipynb 🛱 Edit Lihat Sisipkan Runtime Fitur Bantuan <u>Semua perubahan disimpan</u>
⊨		+ Kod	e + Teks
Q		0	print("\nMales golf")
<>			prog()
		₽	Outlook? (0 is sunny, 1 overcast, 2 rainy) : 0 Temperature? (0 is hot, 1 is mild, 2 is cool) : 0 Humidty? (0 is high, 1 is normal) : 0 Is Windy? 0 false, 1 true : 1 Males golf

Nama : Andry Meylani NIM : 202420009 TUGAS 07

Dari Tugas 6 didapatkan data sebagai berikut :

TID	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP
001	1	1	1	0	0	0	0
002	0	1	1	1	0	0	0
003	0	0	0	1	1	1	0
004	0	1	1	0	0	0	0
005	0	1	1	1	0	1	1

Dari data diatasakan saya coba untuk membuatas sociation rule meggunakan bahasa pyhton dengan colab.research.google.com, berikut langkah-langkahnya :

Langkah 1

Download Libabry Association Rule

[] ! pip insta	all mixtend
Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement Requirement	<pre>t already satisfied: mlxtend in /usr/local/lib/python3.6/dist-packages (0.14.0) t already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.1.4) t already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.post) t already satisfied: mumpy=1.10.4 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.post) t already satisfied: mumpy=1.10.4 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (0.2.2.post) t already satisfied: mumpy=1.10.4 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.8.5) t already satisfied: muppy=1.10.4 in /usr/local/lib/python3.6/dist-packages (from mlxtend) (1.4.1) t already satisfied: pytx=2017.2 in /usr/local/lib/python3.6/dist-packages (from mathemd) (1.4.1) t already satisfied: pytx=2017.2 in /usr/local/lib/python3.6/dist-packages (from mathema) (2.010.0) t already satisfied: pytx=2017.2 in /usr/local/lib/python3.6/dist-packages (from scikt-learnx=0.18-mlxtend) (2.8.1) t already satisfied: pitx=2.0.4 i=2.1.6, -2.0.1 in /usr/local/lib/python3.6/dist-packages (from mathemation).1.5.1-mlxtend) (2.8.1) t already satisfied: pitx=1.0.1 in /usr/local/lib/python3.6/dist-packages (from mathemation).1.5.1-mlxtend) (2.8.1) t already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (1.3.1) t already satisfied: kiwisolver>=0.10 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (1.3.1) t already satisfied: six>=1.0 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (0.10.0) t already satisfied: six>=1.0 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (1.3.1) t already satisfied: six>=1.0 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (0.10.0) t already satisfied: six>=1.0 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1-mlxtend) (0.1.0.0) t already satisfied: six>=1.0 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=1.5.1</pre>

Langkah 2,

Masukkan data mengenai assosiation rule yang akan diproses (data terlampir diatas), Tentukan minimum supportnya pada kali ini saya menggunakan 0.6

Langkah 3,

Tentukan nilai rules sesuai yang diinginkan :

```
rules[ (rules['antecedent_len'] >= 2) &
    (rules['confidence'] > 0.75) &
    (rules['lift'] > 1.2) ]
```

Langkah 4

Kemudian RUN All dari seluruh config yang sudah ada, adapun hasilnya sebagai berikut :

C•	SI	pport	itemsets
	0	0.8	(Mentega)
	1	0.8	(Roti)
	2	0.6	(Telur)
	3	0.8	(Roti, Mentega)

	antecedents	s consequent	ts antecedent supp	port consequent s	apport s	upport con	fidence	e lift	leverage	conviction
0	(Roti) (Menteg	a)	0.8	0.8	0.8	1.	0 1.25	0.16	inf
1	(Mentega	i) (Ro	ti)	0.8	0.8	0.8	1.	0 1.25	0.16	inf
asso	ociation_rul	es(frequent_i	temsets, metric="co	onfidence", min_thre	shold=0.1	')				
z	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	1
0	(Roti)	(Mentega)	0.8	0.8	0.8	1.0	1.25	0.16	in	f
1	(Mentega)	(Roti)	0.8	0.0	0.8	1.0	1.25	0.16	in	f
rule rule	es = associa es	tion_rules(f	requent_itemsets, me	tric="lift", min_t	ireshold=1	2)				
rule rule	es = associa es antecedents	tion_rules(fr	equent_itemsets, me antecedent support	consequent support	nreshold=1 support	2) confidence	lift	leverage	conviction	1
rule rule	es = associa es antecedents (Roti)	consequents (Mentega)	equent_itemsets, me antecedent support 0.8	consequent support	support	confidence	lift 1.25	leverage 0.16	convictio	1 f
rule rule 0 1	es = associa antecedents (Roti) (Mentega)	consequents (Mentega) (Roti)	requent_itemsets, me antecedent support 0.8 0.8	consequent support 0.8 0.1	support 0.8 0.8	confidence 1.0	lift 1.25 1.25	leverage 0.16 0.16	conviction in in	a f f
rule rule 0 1 rule rule	es = associa antecedents (Roti) (Mentega) es["antecede es	consequents (Mentega) (Roti) nt_len"] = ru	equent_itemsets, me antecedent support 0.8 0.8 iles["antecedents"].	etric="lift", min_th consequent support 0.0 0.0 apply(lambda x: ler	<pre>support s 0.8 s(x))</pre>	confidence 1.0 1.0	lift 1.25 1.25	leverage 0.16 0.16	conviction in in	a f f
rule rule 0 1 rule rule	es = associa antecedents (Roti) (Mentega) es ["antecede es antecedents	<pre>tion_rules(f) consequents (Mentega) (Roti) nt_len"] = ru consequents</pre>	antecedent support 0.8 0.8 ules["antecedents"]. antecedent support	consequent support 0.4 apply(lambda x: ler consequent support	support 0.8 0.8 0.8 0.8 0.8 0.8 0.8	confidence	11ft 1.25 1.25 1.1ft	leverage 0.16 0.16	conviction in conviction	f f a antecedent_
rule rule 0 1 rule 2 0	es = associa antecedents (Roti) (Mentega) es ["antecede es antecedents (Roti)	<pre>consequents (Mentega) (Roti) nt_len"] = ru consequents (Mentega) </pre>	equent_itemsets, me antecedent support 0.8 0.8 ales["antecedents"]. antecedent support 0.8	consequent support 0.4 apply(lambda x: ler consequent support 0.4	<pre>ireshold=: i support i 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8</pre>	confidence 1.0 1.0	lift 1.25 1.25 lift 1.25	leverage 0.16 0.16 leverage 0.16	conviction in conviction	a f f f f

TUGAS 07

APRIORI (MARKET BASCET ANALYSIS) - PHYTON



Dibuat Oleh

Ari Hardiyantoro Susanto

Dosen Pengampu

TRI BASUKI KURNIAWAN, S.Kom, M.Eng, Ph.D

Program Pasca Sarjana

Universitas Binadarma Palembang

2020/2021

Step 1: Import the libraries



Pada Collabs modul apriori harus diinstall dulu menggunakan perintah

!pip install apyori

Step 2: Load the dataset



Step 3: Have a glance at the records



Output

Q	0	sto	ore_data	9											
	C→	Dri	ve alre	eady m	ounted at ,	/content	/drive;	to att	empt to	forcibly	remount,	call d	rive.mount	("/content	/drive",
<>			0	1	2	3	4	5	6						
-		0	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP						
		1	PENA	ROTI	MENTEGA	NaN	NaN	NaN	NaN						
		2	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN						
		3	NaN	NaN	NaN	TELUR	BUNCIS	SUSU	NaN						
		4	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN						
		5	NaN	ROTI	MENTEGA	TELUR	NaN	SUSU	KECAP						

Step 4: Look at the shape

) 0 ((-	■ / ▲ : Î O (] t × ▲ (] Z (] G (] ∞ (] ▲ (] ∞ (] ■ (] G (] ④ (] ④ (] ④ (] → (] ■ (] ∞ (] ⊕ (] ⊕ (] ∞ (] ⊕ (G () 1 3 1 G (+	☆ ≯
CC	▲ tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	📮 Comment 🛛 📇 Share	\$
=	+ Code + Text	V RAM Disk V E	diting
	store_data		
Q ()	<pre>#look shape store_data.shape</pre>		
-	Drive already mounted at /content/drive; to attempt to forcibly remount, call driv (6, 7)	/e.mount("/content/drive'	', for

Step 5: Convert Pandas Data Frame into a list of lists

0(
\leftarrow	C Colab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	@ ☆
C	• tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	🗐 Comment 斗 Share 🏟
≣	+ Code + Text	✓ RAM Disk Editing
Q <>	<pre>#converting pandas dataframe records = [] for i in range (0,6): records.append([str(store data.values[i,j]) for j in range(0,7)])</pre>	
	Drive already mounted at /content/drive; to attempt to forcibly remount, call drive	ive.mount("/content/drive", fo

Step 6: Build the Apriori model


Step 7: Print out the number of rules

0 (▏▆▗▎▙▖▌Q▖ૺૼૡ×ૻૣ૾&ۦૺૼઙۦૺૼૼૼૼૼૼૼૼૼૼઙ૽ૻૼૼૼૼૼૼૼૼૼૼૼૼ૾૾ૼૼૼૼૼ૾૾૾૽ૼૻ૽ૼૼૼૼૼૼૼૼૼૼ	+
\leftarrow	→ C a colab.research.google.com/drive/1_6JvzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	९ 🖈 🛊 🔮 :
C	O ▲ tgs-7.ipynb ☆	hare 🏚 🐠
≣ Q	+ Code + Text	Editing
<>	<pre>#Print out the number of rules print(len(association_results))</pre>	-
	Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/d 40	rive", force_rem
		,

Step 8: Have a glance at the rule

D (اَ اللهِ اللهِ عَلَى اللهِ ع	G (G ()) (9 1) G () +
\leftarrow	→ C	Q 🖈 뵭 🕲 :
C	O de tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	텩 Comment 🙁 Share 🌣 🔮
≣	+ Code + Text	✓ RAM Disk ✓ Editing ∧
Q	<pre>#Lauching at the 1st rule print(association_results)</pre>	
<>	Drive already mounted at /content/drive; to attempt to forcibly remount, call de 40	rive.mount("/content/drive", force_rem
	<pre>[RelationRecord(items=frozenset({'MENTEGA'}), support=0.8333333333333333333333333333333333333</pre>	<pre>red_statistics=[OrderedStatistic(items_</pre>

Script Phyton Lengkap

#import import numpy as np import pandas as pd from apyori import apriori

#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('/content/drive')
store_data = pd.read_csv('drive/My Drive/Colab Notebooks/data-sheet-tugas-7ok.csv', header=None)

#having glance of record
store_data

#look shape
store_data.shape

#converting pandas dataframe
records = []
for i in range (0,6):
 records.append([str(store_data.values[i,j]) for j in range(0,7)])

#Build the apriori model

association_rules=apriori(records,min_support=0.20, min_confidence=0.60, min_lift=1,min_leng
th=2)
association_results=list(association_rules)

#Print out the number of rules
print(len(association_results))

#Lauching at the 1st rule
print(association_results)

TUTORIAL PENGOLAHAN DATA ALGORITMA DECISION TREE C4.5 DENGAN MENGGUNAKAN PYTHON

MODEL PREDIKSI ELEKTABILITAS CALON LEGISLATIF MENGGUNAKAN DATA PEMILU SEBAGAI DATA TRAINING

A. Decision Tree

(Andriani, 2012) Pada decision tree terdapat 3 jenis node, yaitu: a. Root Node, merupakan node paling atas, pada node ini tidak ada input dan bisa tidak mempunyai output atau mempunyai output lebih dari satu. b. Internal Node, merupakan node percabangan, pada node ini hanya terdapat satu input dan mempunyai output minimal dua. Leaf Node atau Terminal N

B. Algoritma C4.5

Algoritma C4.5 dan pohon keputusan merupakan dua model yang tidak dapat terpisahkan, karena untuk membangun sebuah pohon keputusan dibutuhkan algoritma C4.5.

Menurut (Nasari, 2014) Secara umum alur proses algoritma C4.5 untuk membangun pohon keputusan dalam data mining adalah:

- 1. Pilih atribut sebagai simpul akar.
- 2. Buat cabang untuk tiap-tiap nilai.
- 3. Bagi kasus dalam cabang.
- 4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Pemilihan atribut sebagai simpul, baik akar (root) atau simpul internal didasarkan pada nilai Gain tertinggi dari atribut-atribut yang ada.

Untuk Menghitung nilai entropy digunakan rumus:

$$Entropy(S) = \sum_{i=1}^{n} -pt \log_2 pt$$

Keterangan : S : himpunan kasus. n : jumlah partisi S Pi : proporsi Si terhadap S

Kemudian hitung nilai informationgain menggunakan rumus:

$$Gain(S,A) = entropy(S) - \sum_{i=1}^{n} \frac{|Si|}{S} * entropy(Si)$$

NIM : 202420008 NAMA : BERI PERIMA

> Keterangan: S : himpunan kasus A : fitur n : jumlah partisi atribut A |S1| : proporsi Si terhadap S |S2| : jumlah kasus dalam S

Pada penelitian ini Algoritma C4.5 digunakan untuk perhitungan nilai information gain pada setiap paramater penentu keputusan pemilihan batik. Untuk melakukan perhitungan nilai entropy dan information gain tertinggi dari setiap parameter, sistem akan menghitung berapa banyak jumlah "Ya" dan "Tidak" dari setiap parameter.

Berikut ini merupakan contoh kasus Elektebilitas Calon Legislatif untuk memudahkan penjelasan mengenai algoritma C4.5.

Dataset yang digunakan menggunakan dataset "datapemilukpu.csv"

Tujuan dari tutorial ini adalah untuk memprediksi elektabilitas calon legislative dengan parameter "Ya" atau "Tidak". Dataset berisi data sekitar 425 dengan 11 atribut,

Tahap awal, melakukan pemanggilan dataset student-por.csv dan menseting modul pandas, sklearn Decision Tree dan memilihi data training maupun data testing, seperti kode dibawah ini

```
> N= M4
# Load libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
```

```
▶ ▶≣ Mi
```

Tampilan dataset dalam bentuk table

	NAMA PARTAI POLITIK	NAMA CALON LEGESLATIF	JENIS KELAMIN	KECAMATAN	NO.URUT PARPOL	SUARA SAH PARTAI	JUML.PEROLEHAN KURSI	DAERAH PEMILIHAN	NO.URUT CALEG	SUARA SAH CALEG	TERPILIH ATAU TIDAK
Ø	HATI NURANI RAKYAT	TOTO SUKISNO,BSc	L	LEBAKSIU	1	18578	1	1	1	594	TIDAK
1	HATI NURANI RAKYAT	EDI PURYANTO,SH	L	SLAWI	1	18578	1	1	2	943	TIDAK
2	HATI NURANI RAKYAT	ELI RETNOWATI,SH	Р	SLAWI	1	18578	1	1	3	1730	TIDAK
3	HATI NURANI RAKYAT	SAHYUDIN	L	DUKUHWARU	1	18578	1	1	4	2508	YA
4	HATI NURANI RAKYAT	H.FAJAR SIGIT KUSUMAJAYA,SH	L	SLAWI	1	18578	1	2	1	923	TIDAK

Menyesuaikan attribute yang dibutuhkan

⊳	>≓ MI	
	<pre>diag = diag.drop(["NAMA PARTAI POLITIK", "NAMA CALON LEGESLATIF", "JENIS KELAMIN", "KECAMATAN",</pre>	
	diag.head()	

	SUARA SAH PARTAI	DAERAH PEMILIHAN	SUARA SAH CALEG	TERPILIH ATAU TIDAK
Ø	18578	1	594	TIDAK
1	18578	1	943	TIDAK
2	18578	1	1730	TIDAK
3	18578	1	2508	YA
4	18578	2	923	TIDAK

Pisahkan dataset feature column dan target variable

```
> F Mi
#split dataset in features and target variable
feature_cols = ['SUARA SAH CALEG','SUARA SAH PARTAI','DAERAH PEMILIHAN']
#X = diag[feature_cols] # Features
#y = diag.Label # Target variable
X=diag.iloc[:, 0:3].values
y=diag.iloc[:, 3].values
```

Pisahkan dataset data Training set dan test set



NIM : 202420008 NAMA : BERI PERIMA

Membuat objek klasifikasi



Accuracy: 0.859375

Hasil Accuracy = 0.859375

```
from sklearn.tree import export_graphviz
from six import StringIO
from IPython.display import Image
import pydotplus
clf = DecisionTreeClassifier(criterion = 'entropy', max_depth = 3, min_samples_split = 20,
class_weight = "balanced")
clftree = clf.fit(X_train,y_train)
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,
                feature_names = feature_cols,
                class_names=['Ya','Tidak'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('datapemilukpu.png')
Image(graph.create_png())
```

NIM : 202420008 NAMA : BERI PERIMA

Bentuk Pohon Keputusan



Nama : Bhijanta Wyasa WM NIM : 202420019 Kelas : MTI 23

Tugas 07

Dari <u>tugas 06</u> sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

Harap dapat dikumpulkan sebelum batas waktu yang ditentukan!

Jawaban:

Dari Tugas 6 didapatkan data sebagai berikut :

TID	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP
001	1	1	1	0	0	0	0
002	0	1	1	1	0	0	0
003	0	0	0	1	1	1	0
004	0	1	1	0	0	0	0
005	0	1	1	1	0	1	1

Dari data diatas akan saya coba untuk membuat association rule meggunakan bahasa pyhton dengan colab.research.google.com, adapun langkahnya sebagai berikut :

Langkah 1,

Download Libabry Association Rule

ile Edit View Insert Runtime Tools Help <u>All changes saved</u> Code + Text	Reconnect	- /	Editing	
Association Analysis				
] ! pip install mixtend Requirement already satisfied: mixtend in /usr/local/lib/python3.6/dist-packages (0.14.0)				
Requirement already satisfied: pandas>=0.17.1 in /usr/local/lib/python3.6/dist-packages (from mixtend) (1.1.4) Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (from mixtend) (50.3.2) Requirement already satisfied: scikit-learn>=0.16 in /usr/local/lib/python3.6/dist-packages (from mixtend) (0.22.2.post) Requirement already satisfied: numpy>=1.04 in /usr/local/lib/python3.6/dist-packages (from mixtend) (1.18.5) Requirement already satisfied: matplotlib>=1.5.1 in /usr/local/lib/python3.6/dist-packages (from mixtend) (3.2.2) Requirement already satisfied: scipy>=0.17 in /usr/local/lib/python3.6/dist-packages (from mixtend) (1.4.1)				
Requirement already satisfied: pytp=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas>=0.17.1->mlxtend) (2018.9) Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.6/dist-packages (from pandas>=0.17.1->mlxtend) Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-packages (from soikt-learn>=0.18->mlxtend) (0.17 Requirement already satisfied: pyparsing!=2.0.4, !=2.1.2, !=2.1.6, >=2.0.1 in /usr/local/lib/python3.6/dist-packages (from matplo10.1) Requirement already satisfied: bijsolvery=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplo10.1) Requirement already s	(2.8.1) .0) tlib>=1.5.1-	->mlxtend) (2.4	
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/pytho3.6/dist-packages (from matplotlib>=1.5.1->mlxtend) (0.10. Requirement already satisfied: six>=1.5 in /usr/local/lib/pytho3.6/dist-packages (from python-dateutil>=2.7.3->pandas>=0.17.1	0) ->mlxtend) ((1.15.0)		

Requirement already satisfied: xlrd in /usr/local/lib/python3.6/dist-packages (1.1.0)

Nama : Bhijanta Wyasa WM NIM : 202420019 Kelas : MTI 23

Langkah 2,

Masukkan data mengenai assosiation rule yang akan diproses (data terlampir diatas),

Tentukan minimum supportnya pada kali ini saya menggunakan 0.6

```
Tugas O6_Bhijanta Wyasa WM.ipynb 
File Edit View Insert Runtime Tools Help <u>All changes saved</u>
+ Code + Text
```

```
[ ] ! pip install xlrd
```

Requirement already satisfied: xlrd in /usr/local/lib/python3.6/dist-packages (1.1.0)

Association Rules Generation from Frequent Itemsets

Source: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/

Langkah 3,

Tentukan nilai rules sesuai yang diinginkan :

Nama : Bhijanta Wyasa WM NIM : 202420019 Kelas : MTI 23

Langkah 4,

Kemudian RUN All dari seluruh config yang sudah ada, adapun hasilnya sebagai berikut :

C⇒		support	itemsets									
	0	0.8	(Mentega)									
	1	0.8	(Roti)									
	2	0.6	(Telur)									
	3	0.8 (I	Roti, Mentega)									
[]	as	sociation_r	rules(freque	nt_itemsets, metric	c="confidence", min	n_thresho	old=0.7)					
		antecedent	ts conseque	nts antecedent sup	port consequent su	upport s	upport con	fidenc	e lift	leverage c	onviction	
	0	(Ro	oti) (Mente	ega)	0.8	0.8	0.8	1	.0 1.25	0.16	inf	
	1	(Menteg	ja) (F	Roti)	0.8	0.8	0.8	1	.0 1.25	0.16	inf	
	ass	ociation_rul	es(frequent_:	itemsets, metric="co	nfidence", min_threa	shold=0.7)					
		antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction		
	0	(Roti)	(Mentega)	0.8	0.8	0.8	1.0	1.25	0.16	inf		
	1	(Mentega)	(Roti)	0.8	0.8	0.8	1.0	1.25	0.16	inf		
[]	rul rul	es = associa es	tion_rules(f	requent_itemsets, me	tric="lift", min_th	reshold=1	.2)					
		antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction		
	0	(Roti)	(Mentega)	0.8	0.8	0.8	1.0	1.25	0.16	inf		
	1	(Mentega)	(Roti)	0.8	0.8	0.8	1.0	1.25	0.16	inf		
[]	rul rul	es["antecede es	nt_len"] = ru	ules["antecedents"].	apply(lambda x: len	(x))						
		antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	antecedent_	len
	0	(Roti)	(Mentega)	0.8	0.8	0.8	1.0	1.25	0.16	inf		1
	1	(Mentega)	(Roti)	0.8	0.8	0.8	1.0	1.25	0.16	inf		1

Didaptkan kesimpulan bahwa Assosiation rule sebagaimana diatas untuk hasilny.

Nama: Cornelia tri WahyuniNim: 202420044Kelas: MTI Reguler AMK: Advanced Database

Tugas 7

Dari <u>Tugas 06</u> sebelum ini, coba gunakan data yang sama, tapi gunakan pemprograman python, lalu buat tutorialnya.

А	В	С	D	E
Peserta	Nilai	Ujian Kopetensi	Wawancara	Diterima
P1	tinggi	bagus	baik	уа
P2	tinggi	cukup	baik	ya
P3	tinggi	kurang	buruk	tidak
P4	sedang	cukup	baik	ya
P5	sedang	bagus	baik	уа
P6	sedang	cukup	baik	уа
P7	sedang	kurang	buruk	tidak
P8	rendah	bagus	baik	уа
P9	rendah	cukup	buruk	tidak

Dataset PSB	Penerimaan	Siswa Baru)
Dutuset I SD	(i chermaan	Jiswa Daraj

Tutorial atau tata cara Perhitungan dan Penerapan Algoritma C 4.5 Pada Phyton Dengan Jupyter Notebook Pada Dataset Penerimaan Siswa Baru

- 1. Pastikan Sudah menginstal Phyton dan Jupyter Notebook
- 2. Siapkan Dataset dengan format CSV
- 3. Pastikan Library pendukung seperti Scikit-Learn, Pydotplus, dan Graphviz, sedangkan untuk pandas dan IPhyton.display sudah tersedia sendiri pada saat menginstal phyton.

Link dibawah untuk download :

Scikit-learn > <u>https://scikit-learn.org/stable/install.html</u> Graphviz >

https://www2.graphviz.org/Packages/stable/windows/10/cmake/Release/x64/ Pydotplus > https://pypi.org/project/pydotplus/

- Untuk graphviz silahkan atur path dan masukan ke system environtment variables sesuai dengan directory penginstalan pada komputer, misal : C:\Program Files\Graphviz 2.44.1\bin;
- Setelah semua selesai lalu masuk ke halaman Jupyter Notebook > pilih File > New > Phyton 3

6. Masukan semua variabel library yang telah di instal seperti gambar berikut :



 Lalu setelah selesai lalu masukan atribut sesuai dataset yang telah dibaut seperti gambar dibawah dan panggil sesuai nama dataset seperti gambar dibawah dengan format csv atau txt.



8. Maka data akan muncul seperti pada gambar dibawah ini.



9. Lalu cara yang selanjutnya yaitu mengubah data menjadi data binner dengan cara pada gambar dibawah kecuali atribut **'wawancara'** karena merupakan atribut target.

10. Dan maka dataset yang kita masukan sudah berubah menjadi data binner yang hanya berupa angka 1 dan angka 0 seperti gambar dibawah :

	D	D	D	D	D	D	D	D	D	MPL-1 d-h	Million and an a	Million and and	
	Peserta_P1	Peserta_PZ	Peserta_P3	Peserta_P4	Peserta_Po	Peserta_P6	Peserta_P7	Peserta_Po	Peserta_P9	Nilal_rendan	Nilal_sedang	Nilai_tinggi	ĸ
0	1	0	0	0	0	0	0	0	0	0	0	1	
1	0	1	0	0	0	0	0	0	0	0	0	1	
2	0	0	1	0	0	0	0	0	0	0	0	1	
3	0	0	0	1	0	0	0	0	0	0	1	0	
4	0	0	0	0	1	0	0	0	0	0	1	0	
4													

11. Pada Cara terakhir ini yaitu untuk menampilkan decision tree :



12. Maka Tampilan Decision Tree akan muncul seperti gambar dibawah ini :



Nama : Cynthia Anisa Agatha

NIM : 202420022

import pandas as pd #Import Library Pandas import numpy as np #Import Library Numpy from sklearn.preprocessing import Imputer #Import Library Preprocesing dan Imputer df = pd.read_csv("Data.csv") #Impot csv data print(df['Income'].head(10)) #Data Cleansing NaN imputer =Imputer(missing_values= 'NaN',strategy='drop',axis=0) # Function to split the dataset def splitdataset(balance_data): # Separating the target variable X = balance_data.values[:, 1:5]

Y = balance_data.values[:, 0]

Splitting the dataset into train and test

X_train, X_test, y_train, y_test = train_test_split(

X, Y, test_size = 0.3, random_state = 100)

return X, Y, X_train, X_test, y_train, y_test

Function to perform training with giniIndex. def train_using_gini(X_train, X_test, y_train):

Creating the classifier object

clf_gini = DecisionTreeClassifier(criterion = "gini",

random_state = 100,max_depth=3, min_samples_leaf=5)

```
# Performing training
clf_gini.fit(X_train, y_train)
return clf_gini
```

Function to perform training with entropy. def tarin_using_entropy(X_train, X_test, y_train):

```
# Decision tree with entropy
clf_entropy = DecisionTreeClassifier(
    criterion = "entropy", random_state = 100,
    max_depth = 3, min_samples_leaf = 5)
```

Performing training
clf_entropy.fit(X_train, y_train)
return clf_entropy

Function to make predictions
def prediction(X_test, clf_object):

```
# Predicton on test with giniIndex
y_pred = clf_object.predict(X_test)
print("Predicted values:")
print(y_pred)
return y_pred
```

```
# Function to calculate accuracy
def cal_accuracy(y_test, y_pred):
```

print("Confusion Matrix: ",

```
confusion_matrix(y_test, y_pred))
```

print ("Accuracy : ",

```
accuracy_score(y_test,y_pred)*100)
```

print("Report : ",
classification_report(y_test, y_pred))

Driver code

def main():

Building Phase data = importdata() X, Y, X_train, X_test, y_train, y_test = splitdataset(data) clf_gini = train_using_gini(X_train, X_test, y_train) clf_entropy = tarin_using_entropy(X_train, X_test, y_train)

Operational Phase
print("Results Using Gini Index:")

Prediction using gini
y_pred_gini = prediction(X_test, clf_gini)
cal_accuracy(y_test, y_pred_gini)

print("Results Using Entropy:")
Prediction using entropy
y_pred_entropy = prediction(X_test, clf_entropy)
cal_accuracy(y_test, y_pred_entropy)

Calling main function

if __name__=="__main__":

main()

Step 1: Import the libraries



Catatan penulis : pada Collabs modul apyori harus diinstall dulu menggunakan perintah

!pip install apyori

Step 2: Load the dataset

#loading data sheet
#import datasheet
from google.colab import drive
drive.mount('<u>/content/drive</u>')
store_data = pd.read_csv('drive/My Drive/Colab Notebooks/data-sheet-tugas-7-ok.csv', header=None)

Step 3: Have a glance at the records



Output

0	sto	re_data	Э													
C→	Dri	ve alre	eady mo	ounted at ,	/content	/drive;	to att	empt to	forcibly	remount,	call d	lrive	e.mount	:("/con	tent/dr	iv
		0	1	2	3	4	5	6								
	0	PENA	ROTI	MENTEGA	TELUR	BUNCIS	SUSU	KECAP								
	1	PENA	ROTI	MENTEGA	NaN	NaN	NaN	NaN								
	2	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN								
	3	NaN	NaN	NaN	TELUR	BUNCIS	SUSU	NaN								
	4	NaN	ROTI	MENTEGA	TELUR	NaN	NaN	NaN								
	5	NaN	ROTI	MENTEGA	TELUR	NaN	SUSU	KECAP								

Nama : Efrik Kartono Ahsa MK : Advanced Database NIM : 202420030

Step 4: Look at the shape

← → C colab.research.google.com/drive/1_6JvzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn 0 🕁 簈 🝐 tgs-7.ipynb 🛛 ☆ **CO** Comment 🚓 Share 🛛 🏛 File Edit View Insert Runtime Tools Help ✓ RAM Link ► Editing + Code + Text ≣ store_data Q #look shape store_data.shape <> Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", for (6,7) .

Step 5: Convert Pandas DataFrame into a list of lists

) 0 (+ +	■ / 🍐 : 🎧 (] t × 🍐 (] Z (] G :] @ : 🍐 (] D0 1 D1 1 G i G] (D) 1 G i] Z / B / ∞ 1 G i] G (] ⊗ : G C () Colab.research.google.com/drive/1_6/vzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	ાં દ ાં ક ાં દ ાં ન ન ન હ જ ા
CC	▲ tgs-7.ipynb ☆ File Edit View Insert Runtime Tools Help	🔲 Comment 🛛 👫 Share 🏟
≔	+ Code + Text	✓ RAM Disk ► Editing
Q <>	<pre>#converting pandas dataframe records = [] for i in range (0,6): records.append([str(store_data.values[i,j]) for j in range(0,7)])</pre>	
	Drive already mounted at /content/drive; to attempt to forcibly remount, call dri	.ve.mount("/content/drive", fo

Step 6: Build the Apriori model



Step 7: Print out the number of rules

) () (I III / ▲ : [O] te ×] ▲ [Z] & [Z] & [M] I I I] B] G] G] [M] G] Z] #] #] &] &] G] G] G] B] #] #] A] A] A] A] A] A] A] A	(+) □ □ × Q ☆ ★
C	O ▲ tgs-7.ipynb ☆ 📮 Comment 🗳	🕻 Share 🏟 🐠
iii ⊲	+ Code + Text #Print out the number of rules print([len(association results)])	Editing
÷	Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/conten 40	t/drive", force_rem

Step 8: Have a glance at the rule

0 (🔲 / 🛆 : Î 🗘 (Î 👷 X 🚺 (Î 🧝 X 🗍 G : Î 👁 : Î 🛆 (Î 🕫 1] 🖬 1] G : Î G 1] 🔞 1] G : Î 🖉 1] 🖉 2] 🐼 1] G : Î 🖉 : Î (G (
←	Colab.research.google.com/drive/1_6JvzqcSFmFtyoK0D1-xyK6LSX71Honf#scrollTo=CxEbfdtVR_hn	० 🖈 🔅 :
C) 🖕 tgs-7.ipynb 🛱 File Edit View Insert Runtime Tools Help	텩 Comment 🙁 Share 🏟 🔮
≣	+ Code + Text	✓ RAM Disk ✓ Editing ∧
Q	<pre>#Lauching at the 1st rule print(association_results)</pre>	
<>	Drive already mounted at /content/drive; to attempt to forcibly remount, call d 40	rive.mount("/content/drive", force_rem
	<pre>[RelationRecord(items=frozenset({'MENTEGA'}), support=0.83333333333333334, order </pre>	ed_statistics=[OrderedStatistic(items_ ,

Nama : Efrik Kartono Ahsa MK : Advanced Database NIM : 202420030

Script Phyton Lengkap

#import import numpy as np import pandas as pd from apyori import apriori #loading data sheet #import datasheet from google.colab import drive drive.mount('/content/drive') store data = pd.read csv('drive/My Drive/Colab Notebooks/data-sheettugas-7-ok.csv', header=None) #having glance of record store data #look shape store data.shape #converting pandas dataframe records = [] for i in range (0, 6): records.append([str(store data.values[i,j]) for j in range(0,7)]) #Build the apriori model association rules=apriori(records,min support=0.20, min confidence=0.60 , min lift=1,min length=2) association results=list(association rules) #Print out the number of rules print(len(association results))

#Lauching at the 1st rule
print(association results)



TUGAS 07 ADVANCED DATABASE

Dari Tugas 06 sebelum ini, coba gunakan data yang sama, tapi gunakan pemrograman python, lalu buat tutorialnya.

Langkah tutorial :

Langkah Awal Yang Disiapkan

A. Python

Package yang harus diinstal :

- Pandas
- Numpy
- Sklearn
- Seaborn
- matplotlib
- B. Jupyter notebook
- C. Dataset

Langkah Kerja

1. Install semua aplikasi Python dan jupyter notebook, lalu run jupyter notebook via command prompt.



ſ	C:\Windows\system32\cmd.exe - jupyter notebook	J
	Microsoft Windows [Version 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved.	
	C:\Users\HP>jupyter notebook [I 20:22:45.987 NotebookApp] Serving notebooks from local directory: C:\Users\HP	
	[] 20:22:45.989 NotebookApp] The Jupyter Notebook is running at: [] 20:22:45.991 NotebookApp] http://localhost:8888/?token=81b5e3f7be332c26eb7b8a 2aae9198dcba7c54127c360ed4	
	[I 20:22:45.992 NotebookApp] or http://127.0.0.1:88888/?token=81b5e3f7be332c26eb 7b8a2aae9198dcba7c54127c360ed4 [I 20:22:45 994 NotebookApp] Use Control=C to stop this server and shut down all	
	kernels (twice to skip confirmation). [C 20:22:46.228 NotebookApp]	

Gambar 1.1 Cara menjalankan jupyter notebook

Saya akan menggunakan "Prediksi-Kelulusan" sebagai dataset yang digunakan pada kali ini.
 Dataset terdiri dari 12 sampel dari Prediksi kelulusan apakah akan :

- Tepat waktu
- Terlambat

S1	S2	S3	S4	S5	Keterangan
3.71	3.89	3.89	3.89	3.99	Tepat Waktu
3.67	3.81	3.87	3.87	3.78	Tepat Waktu
3.66	3.78	3.87	3.78	3.82	Tepat Waktu
3.59	3.78	3.81	3.78	3.82	Tepat Waktu
3.54	3.77	3.81	3.77	3.66	Tepat Waktu
3.49	3.75	3.79	3.71	3.81	Tepat Waktu
1.78	1.99	2.22	2.31	2.12	Terlambat
1.67	1.99	2.35	2.55	2.11	Terlambat
1.44	1.56	2.23	2.55	2.18	Terlambat
1.78	1.99	2.12	2.11	2.22	Terlambat
1.78	1.89	2.11	2.43	2.32	Terlambat
1.76	1.84	1.99	2.32	2.43	Terlambat

Dataset yang digunakan(Prediksi-Kelulusan.csv)

Dua fitur diukur dari sampel meliputi nilai IP pada : Semester 1, Semester 2, Semester 3, Semester 4, dan Semester 5.



```
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
import matplotlib.pyplot as plt
import math
```

Gambar 1.2 Import package

from pandas.plotting import scatter_matrix
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier

Gambar 1.3 Sklearn module

data = pd.read_csv("Prediksi-Kelulusan.csv") data.head()										
S1	S2	\$ 3	S 4	\$ 5	Keterangan					
3.71	3.89	3.89	3.89	3.99	Tepat Waktu					
3.67	3.81	3.87	3.87	3.78	Tepat Waktu					
3.66	3.78	3.87	3.78	3.82	Tepat Waktu					
3.59	3.78	3.81	3.78	3.82	Tepat Waktu					
3.54	3.77	3.81	3.77	3.66	Tepat Waktu					
	ta = ta.he 3.71 3.66 3.59 3.54	ta = pd.re ta.head() S1 S2 3.71 3.89 3.67 3.81 3.66 3.78 3.59 3.78 3.54 3.77	ta = pd.read_c: ta.head() S1 S2 S3 3.71 3.89 3.89 3.67 3.81 3.87 3.66 3.78 3.87 3.59 3.78 3.81 3.54 3.77 3.81	sta = pd.read_csv("Ps1s2s3s43.713.893.893.893.673.813.873.873.663.783.873.783.593.783.813.783.543.773.813.77	sta pd.read_csv("Predikta.head() sta sta sta sta 3.71 3.89 3.89 3.89 3.99 3.67 3.81 3.87 3.87 3.78 3.66 3.78 3.87 3.78 3.82 3.59 3.78 3.81 3.78 3.82 3.54 3.77 3.81 3.77 3.66					

Gambar 1.4 Import dataset dan Output yang dihasilkan

3. Selanjutnya saya akan membuat suatu boxplot dengan syntax di bawah ini

```
#box and whisker plots
data.plot(kind='box', subplots=True, layout=(3,2), sharex=False, sharey=False)
plt.show()
```

Gambar 1.5 Syntax boxplot

Dan otputnya akan seperti dibawah ini :





Gambar 1.6 Output dari boxplot

4. Untuk menampilkan scatter plot dari semua variable gunakan perintah dibawah ini.



Gambar 4.1 Syntax Scatter Plot dan Output dari Scatter Plot

5. Sebelum masuk ke analisis KNN, import sklearn pada program, kemudian ketikkan script program seperti dibawah. Script dibagi menjadi 2 bagian data 80% untuk data training, dan 20% untuk data test, sehingga pada test_size=0.2



```
x = data.iloc[:, :-5].values
y = data.iloc[:, 5].values
```

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

6. Setelah data di-split menjadi 80% training set dan 20% test set, selanjutnya masuk ke analisis KNN, ketikan script seperti dibawah. Goals dari penggunaan metode analisis ini adalah menemukan faktor-k yang sesuai, dilihat dari accuracy yang dihasilkan. Semakin tinggi akurasi, maka factor-k tersebut yang akan dijadikan referensi untuk semua data set.

```
#Import knearest neighbors Classifier model
from sklearn.neighbors import KNeighborsClassifier
#Create KNN classifier
knn = KNeighborsClassifier(n_neighbors=8)
#Train the model using the training sets
knn.fit(x_train, y_train)
#Predict the response for test dataset
y_pred = knn.predict(x_test)
#Import scikit-Learn metrics module for accurancy
from sklearn import metrics
#Model Accuracy, how often is the classifier carrect?
print("Accuracy:", metrics.accuracy_score(y_test, y pred))
```

Accuracy: 0.333333333333333333

Gambar 6.1 menghitung akurasi dari k=8



```
#Import knearest neighbors Classifier model
from sklearn.neighbors import KNeighborsClassifier
#Create KNN classifier
knn = KNeighborsClassifier(n_neighbors=3)
#Train the model using the training sets
knn.fit(x_train, y_train)
#Predict the response for test dataset
y_pred = knn.predict(x_test)
#Import scikit-Learn metrics module for accurancy
from sklearn import metrics
#Model Accuracy, how often is the classifier carrect?
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

Accuracy: 1.0

Gambar 6.2 menghitung akurasi dari k=3

SELESAI

Pengolahan data dengan Python dan menggunakan dataset yang sama yg di gunakan di RapidMiner.

Penjelasan :

Dataset terdiri dari 520 records dan 17 attribute/Feature/column, attribute ke-17 di jadikan label/class yg akan di analisa menggunakan Teknik data mining.

Sebagian besar datanya adalah nominal yg mempunyai dua nilai yang di sebut binominal, contoh : male/female, Yes/No, Positive/Negative.

Mengingat data label/class berupa data nominal, maka pilihan modeling data miningnya adalah classification, dalam hal ini di pilih algoritma Decision Tree

```
# Load Libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
col_names = ['Age', 'Gender', 'Polyuria', 'Polydispia', 'Sudden Weight loss', 'Weakness',
'Beleviewich' Formation', 'Polyuria', 'Polydispia', 'Sudden Weight loss', 'Weakness',
```

```
'Polyphagia', 'Genital Thrush', 'Visual blurring', 'Itching', 'Irritability',

'Delayed Healing', 'Partial Paresis', 'Muscle stiffness', 'Alopecia', 'Obesity', 'Label']

# Load dataset
```

diag = pd.read_csv("diabetes_data1.csv")

Penyebutan kolom name diperlukan jika loading dataset tidak mempunyai heade atau menggunakan opsi set **header** = None, namun dalam case ini, data set csv sudah memiliki header

diag.head()

	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	visual blurring	Itching	Irritability	delayed healing	partial paresis	muscle stiffness	Alopecia	Obesity	
0	40	1	0	1	0	1	0	0	0	1	0	1	0	1	1	1	Po
1	58	1	0	0	0	1	0	0	1	0	0	0	1	0	1	0	Po:
2	41	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	Po
3	45	1	0	0	1	1	1	1	0	1	0	1	0	0	0	0	Po:
4	60	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	Po

Catatan Output : Gender 1 = Male , 0 = Female , Other attribute 1 = Yes, 0 = No

Note : Ada dua cara pemilihan attribute, menggunakan feature_cols dan label (di comment), atau mengunakan index.

Variable feature_cols akan di gunakan nanti saat akan menggambar graphic -nya

```
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training and 30% test
```

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()
clf_tree = DecisionTreeClassifier(criterion='Entropy', random_state=1)
```

```
# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)
```

```
#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

```
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Accuracy: 0.9423076923076923

Mengunakan module graphviz untuk menampilkan hasil analisanya(dibawah)

Output :



Catatan:

Age : Umur Gender : Jenis Kelamin Polyuria : sering buang air kecil Polydispia : sering merasa haus Sudden Weight loss : Berat Badan turun tiba-tiba Weakness : Lemah Polyphagia : banyak makan (sering lapar) Genital Thrush : gatal di kemaluan Visual blurring : Penglihatan Buram Itching : Gatal Irritability : gampang marah Delayed Healing : Sulit sembuh (luka) Partial Paresis : lumpuh sebagian Muscle stiffness : otot kaku Alopecia : kebotakan, rambut rontok **Obesity** : Kegemukan Label > indikasi Diabetes

Tutorial Algoritma K-Nearest Neighbours Dengan Python

K-Nearest Neighbours (KNN)

KNN adalah algoritma pembelajaran mesin yang diawasi yang dapat digunakan untuk menyelesaikan masalah klasifikasi dan regresi. Prinsip KNN adalah nilai atau kelas suatu titik data yang ditentukan oleh titik data di sekitar nilai tersebut.

Untuk memahami algoritma klasifikasi KNN seringkali paling baik ditunjukkan melalui contoh. Tutorial ini akan mendemonstrasikan bagaimana Anda dapat menggunakan KNN dengan Python dengan masalah klasifikasi Anda sendiri. Notebook Jupyter yang sesuai dengan contoh ini dapat ditemukan di sini, jika Anda ingin mengikutinya.

Algoritme prediksi menghitung jarak dari titik x yang tidak diketahui, ke semua titik dalam data Anda. Titik-titik dalam data Anda kemudian diurutkan dengan menambah jarak dari x. Prediksi dilakukan dengan memprediksi label mayoritas dari titik terdekat 'K'.

Memilih K akan memengaruhi kelas tempat poin baru akan ditetapkan.

Dalam contoh di bawah ini, memilih nilai K 2 akan menetapkan titik yang tidak diketahui (lingkaran hitam) ke kelas 2. Namun, jika nilai K adalah 7, titik yang tidak diketahui akan ditetapkan ke kelas 1.



Feature_2

Pertama, kita mengimpor pustaka yang kita butuhkan, dan kemudian membuat dataset palsu menggunakan fungsi makeblobs dari sklearn. Kita dapat mengirimkan jumlah sampel, fitur dalam kumpulan data kita, berapa banyak pusat atau kelas yang akan memasukkan data, dan terakhir deviasi standar dari cluster tersebut. Untuk konsistensi antara beberapa jalannya notebook Jupyter ini, saya telah menetapkan integer 101 ke parameter random_state.

Catatan, untuk memulai, kita akan memiliki standar deviasi cluster yang besar. Ini akan memperkenalkan varians ke dalam klasifikasi, yang dapat kita perbaiki nanti, dengan secara khusus memilih nilai K yang optimal. Ini dapat dicapai dengan menggunakan metode siku.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import make blobs
data = make blobs(n samples=300, n features=5, centers=2, cluster std=6.0, random state=101)
data
(array([[ -0.95757537,
                       3.36332609, -15.54675979, -14.02967497,
          1.50545246],
       [-11.12008037, -0.86726927, -19.42687054, -22.99153445,
         12.8409123 ],
       [ 5.02786886, -2.84037069, -5.9094317, -16.29765383,
          7.77075032],
                       2.29827056, -13.80731349, -10.89022536,
       [ -8.02114181,
          1.99399904],
       [ 10.87670302,
                       3.25562702, -6.25095388, -0.92884525,
          8.18286695],
       [ 7.86530195, -11.18764669, 6.36417619, -2.87676038,
          1.31626729]]),
 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0,
       1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0,
       1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1,
```

Fungsi makeblobs dari sklearn mengembalikan tupel 2 elemen. Kita dapat membuat kerangka data fitur kita menggunakan pd.DataFrame, dan meneruskan indeks tupel pertama yang sesuai dengan data fitur. Elemen kedua dari tupel data sesuai dengan label fitur.

df_feat = pd.DataFrame(data[0], columns=['feature_' + str(i) for i in range(1, 6)])

df_feat.head(2)

 feature_1
 feature_2
 feature_3
 feature_4
 feature_5

 -0.957575
 3.363326
 -15.546760
 -14.029675
 1.505452

1 -11.120080 -0.867269 -19.426871 -22.991534 12.840912

```
y = data[1]
```

у

Sekarang kita dapat menskalakan data dengan mengimpor MinMaxScaler dari Sklearn.preprocessing. Tidak seperti algoritme pembelajaran mesin lainnya, kami menyesuaikan dan mengubah semua data pelatihan, sebelum melakukan pemisahan pengujian kereta kami.

```
Untitled1 Last Checkpoint: 11/03/2020 (unsaved changes)
 File
       Edit
             View
                     Insert
                            Cell
                                   Kernel
                                           Widgets
                                                     Help
🕒 🕂 😹 伦 🖪 🕇 🔸
                            ▶ Run 🔳 C 🕨 Code
                                                             2002
                                                         \sim
     In [ ]: from sklearn.preprocessing import MinMaxScaler
     In [ ]: scaler = MinMaxScaler()
     In [ ]: X = scaler.fit_transform(df_feat)
     In [ ]: from sklearn.model_selection import train_test_split
     In [ ]: X train, X test, y train, y test = train_test_split(X, y, test_size=0.3, random_state=101)
```

Algoritme prediksi dan pengoptimalan

Untuk mengimplementasikan prediksi dalam kode, kita mulai dengan mengimpor KNeighboursClassifier dari sklearn.neighbours. Kami kemudian membuat instance dari KNeighboursClassifier, dengan meneruskan argumen 1 ke n_neighbours, dan menetapkan ini ke variabel knn. Nilai yang diteruskan ke n_neighbours mewakili nilai K.

Kami kemudian menyesuaikan dengan data pelatihan, sebelum membuat prediksi, dengan memanggil metode prediksi pada objek KNeighboursClassifier kami.

Sekarang kita dapat menilai keakuratan prediksi menggunakan klasifikasi_report dan konfusi matriks.

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier(n_neighbors=1)
```

```
knn.fit(X_train, y_train)
```

predictions = knn.predict(X_test)

```
predictions
```

array([0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1])

Metrik tersebut menunjukkan bahwa akurasinya sudah sangat baik. Ini mungkin karena fakta bahwa kami membuat kumpulan data dengan makeblobs dan secara khusus meminta 2 pusat. Namun, kami sengaja menempatkan nilai yang besar untuk standar deviasi cluster untuk memperkenalkan varians. Hal ini mengakibatkan kesalahan klasifikasi pada 4 poin dalam kumpulan data kami.

print(confusion matrix(y test, predictions)) print(classification_report(y_test, predictions)) [[42 1]] [3 44]] precision recall f1-score support 0 0.93 0.98 0.95 43 1 0.96 0.98 0.94 47 0.96 90 accuracy 0.96 0.96 0.96 90 macro avg weighted avg 0.96 0.96 0.96 90

Meningkatkan Akurasi Data

Kami dapat mencoba untuk meningkatkan akurasi hasil kami dengan memodifikasi jumlah tetangga. Ini dapat dicapai dengan menggunakan metode siku.

Pertama-tama kita melakukan iterasi melalui 40 nilai tetangga, membuat instance objek KNeighboursClassifier dengan jumlah tetangga tersebut. Kami kemudian dapat

menyesuaikan data pelatihan dengan model KNN ini, mendapatkan prediksi, dan menambahkan nilai rata-rata antara prediksi, pred_i dan nilai yang benar, y_test.

Jika pred_i dan y_test tidak cocok dalam array, nilai true dikembalikan yang memiliki nilai 1. Semakin tinggi angkanya, semakin tidak akurat klasifikasi tersebut.

Nilai yang lebih rendah untuk tingkat kesalahan akan sesuai dengan model yang berkinerja lebih baik.

Hasil ini dapat diplot menggunakan rentang nilai i pada sumbu x, versus tingkat kesalahan pada sumbu y.

```
1
     error_rate = []
 2
 3
     for i in range(1, 40):
         knn = KNeighborsClassifier(n_neighbors=i)
 4
         knn.fit(X_train, y_train)
 5
         pred_i = knn.predict(X_test)
 6
 7
 8
         error_rate.append(np.mean(pred_i != y_test))
 9
10
     plt.figure(figsize=(10, 6))
11
12
13
     plt.plot(range(1, 40), error_rate, color='blue', linestyle='--',
14
              markersize=10, markerfacecolor='red', marker='o')
15
     plt.title('K versus Error rate')
16
     plt.xlabel('K')
     plt.ylabel('Error rate')
17
```

Sekarang kita dapat memilih nilai K terendah yang akan menghasilkan, tingkat kesalahan terendah. Di sini, kita bisa memilih 5.



K versus Error rate

Sekarang kita dapat, menjalankan kembali penilaian akurasi dengan matriks kebingungan dan laporan klasifikasi sekali lagi, untuk melihat apakah kita mengklasifikasikan 4 poin yang tidak selaras dengan lebih akurat. Kami telah meningkatkan, dari 4 poin yang salah diklasifikasikan menjadi 2.

```
knn = KNeighborsClassifier(n neighbors=5)
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)
print(confusion matrix(y test, predictions))
print(classification_report(y_test, predictions))
[[42
      1]
 [ 1 46]]
              precision
                            recall
                                    f1-score
                                                support
           0
                    0.98
                              0.98
                                         0.98
                                                     43
           1
                    0.98
                              0.98
                                         0.98
                                                     47
                                         0.98
                                                     90
    accuracy
                    0.98
                              0.98
                                         0.98
                                                     90
   macro avg
weighted avg
                    0.98
                              0.98
                                         0.98
                                                     90
```

Pelatihan tentang titik data baru

Sekarang kita dapat membuat titik data menggunakan data asli. Pertama kita membuat dua dataframe; satu dengan fitur dan satu lagi dengan label, menggabungkannya menjadi satu kerangka data, dan memilih baris pertama, sebagai titik data untuk memprediksi labelnya. Kita harus ingat untuk menskalakan titik data karena model dilatih pada data yang diskalakan.

Prediksi menunjukkan bahwa data titik 1, akan memberikan label 0, yang sesuai dengan titik dataset asli, diverifikasi dengan memanggil df.head (1).

```
1 features = pd.DataFrame(data=data[0], columns=['feature_' + str(i) for i in range(1, 6)])
2 lables = pd.DataFrame(data[1], columns=['labels'])
3 dataset = pd.concat([features, lables], axis=1)
4 data_point_1 = scaler.transform(np.array(dataset.iloc[0][:-1]).reshape(-1, 5))
5 knn.predict(data_point_1)[0]
6
7 # Output
8 # 0
```
feature_1 feature_2 feature_3 feature_4 feature_5 labels

0 -0.957575 3.363326 -15.54676 -14.029675 1.505452 **0**

data_point_1 = scaler.transform(np.array(dataset.iloc[0][:-1]).reshape(-1, 5))

data_point_1

array([[0.3943128, 0.58849094, 0.1949979, 0.21287012, 0.54980842]])

knn.predict(data_point_1)[0]

0