

Foundation for Data Science

Presented by



Daftar Isi

A.	Pengantar Data Science	4
B.	Computational Thinking	8
C.	Statistical Thinking	78
D.	Hypothesis	87
E.	Digital Leadership	109

Prologue

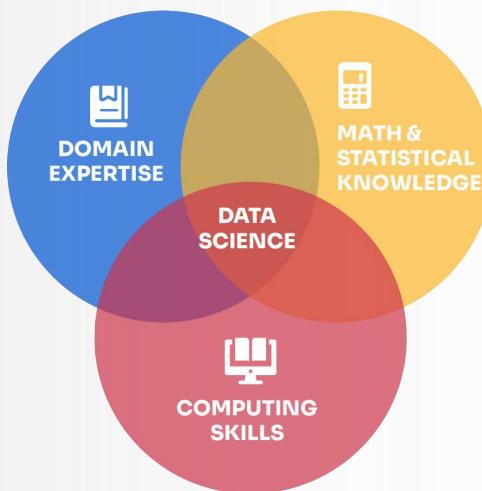
Selamat datang pada Modul Pembelajaran Data Science! Modul ini akan membahas tentang fundamental utama dalam Data Science yang sangat penting untuk dipahami sebelum memasuki materi yang lebih kompleks. Dalam modul ini, akan dibahas beberapa sub-topik, antara lain:

- I. Pengantar Data Science
- II. Computational Thinking
- III. Statistical Thinking
- IV. Digital Leadership

Setelah mempelajari seluruh materi pada modul pertama ini, diharapkan Pembaca dapat memahami informasi umum tentang ilmu data mulai dari teori dasar hingga pentingnya ilmu data dalam berbagai industri. Selain itu, Pembaca juga akan memiliki kemampuan berpikir komputasional dan memahami konsep Python, serta mampu untuk berpikir secara statistikal dan memahami fundamental statistik. Semoga modul ini dapat membantu Pembaca dalam mengembangkan pengetahuan dan keterampilan dalam bidang Data Science. Selamat belajar!

Pengantar Data Science

Pernahkah sebelumnya kalian mendengar istilah Data Science? Di-era digitalisasi saat ini Data Science menjadi perbincangan hangat, lantaran peranannya yang sangat penting dalam industri yang berpacu pada data sebagai *business driver* dan profesi semakin diminati saat ini.



Jika dilihat pada diagram diatas, terlihat bahwa Data Science merupakan irisan antara *computing skill*, *math & statistical knowledge*, dan *domain expertise*. Jelas bahwa Data Science merupakan ilmu yang lahir dari gabungan ilmu statistik dan ilmu komputer dan didukung pemahaman terkait keahlian dibidang tertentu (*business*). Tiga komposisi tersebut sangat berkaitan dan tidak akan pernah lepas, karena jika hanya mahir dalam komputasi dan statistik tanpa memiliki fundamental yang kuat terhadap *domain expertise*, maka model ataupun hasil analisisnya tidak akan tepat sesuai dengan kebutuhan bisnis, pun begitu untuk dua yang lainnya.

Seperti yang disebutkan diawal, Data Science kini menjadi perbincangan hangat karena peran pentingnya bagi organisasi atau perusahaan dalam menjalankan bisnisnya yang berbasis dengan data. Misalnya, sejumlah perusahaan kini sudah mulai menyadari bahwa data dalam jumlah besar memiliki nilai tersembunyi yang besar, sehingga mereka mencari cara agar nilai tersebut bisa diekstrak dan ditindaklanjuti untuk menghasilkan keputusan-keputusan bisnis yang strategis.

Pengantar Data Science

Data Science VS Data Mining

Dalam analogi sebelumnya disebutkan bahwa sebuah organisasi ingin mengekstrak nilai tersembunyi dalam jumlah yang besar, biasanya hal ini sebut sebagai Data Science dan Data Mining. Lantas apa perbedaan keduanya? Berikut beberapa perbedaan antara Data Science dan Data Mining:

1. Cakupan Bidang

Data Science merupakan bidang yang cukup luas, mulai dari proses menangkap data (ekstraksi data), menganalisis dan memperoleh wawasan (*insight*). Sedangkan, Data Mining adalah tentang bagaimana menemukan informasi yang berguna dalam kumpulan data, dan memanfaatkan informasi tersebut untuk menemukan pola tersembunyi.

2. Rumpun Ilmu

Data Science itu bidang multidisiplin yang terdiri dari statistik, ilmu sosial, visualisasi data, pemrosesan bahasa alami, data mining, dan lain-lain. Data Mining merupakan bagian dari Data Science.

3. Role (Peran)

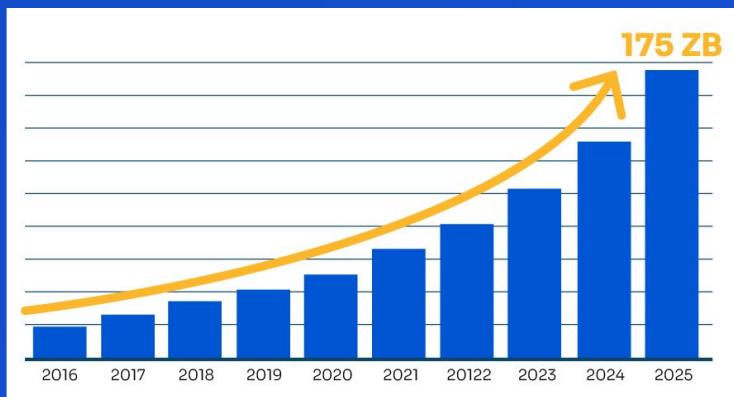
Seorang profesional Data Science dapat dianggap sebagai kombinasi AI Researcher, Machine Learning Engineer, atau Data Analyst dan Data Engineer. Sebaliknya, seorang profesional Data Mining tidak harus mampu melakukan semua peran itu.

4. Jenis Data

Data Science berurusan dengan setiap jenis data baik terstruktur, semi-struktur dan tidak terstruktur. Sedangkan, Data Mining biasanya hanya berkaitan dengan data terstruktur.

Pengantar Data Science

Data Science Dalam Industri



Gambar 1 Prediksi Pertumbuhan Big Data ([Link](#))

Dengan adanya pertumbuhan data yang semakin meningkat, hal ini membuat peran Data Science dan Data Mining menjadi sangat penting bagi organisasi di seluruh dunia jika ingin tetap kompetitif dan relevan. Baik Data Science maupun Data Mining berperan untuk mengidentifikasi peluang dan membuat

keputusan bisnis yang efektif sehingga organisasi dapat terus berkembang. Data Science kini mulai berkembang dalam berbagai macam aspek industri, diantaranya:

Marketing and Sales



Penggunaan Data Science dalam pemasaran dan penjualan adalah rekomendasi produk. Misal, saat kita memeriksa produk di E-commerce, sistem akan memberitahu kita bahwa ada produk lain yang mungkin saja kita sukai, kerja algoritma yang memberikan rekomendasi produk tersebut mengacu pada data pembelian produk kita sebelumnya, data pada *wishlist/preloved* item yang kita simpan, atau data dari pelanggan lain yang memiliki kemiripan data.

Pengantar Data Science

Data Science Dalam Industri

Finance

Pencegahan penipuan adalah salah satu tugas bank yang paling vital. Dengan menerapkan algoritma Data Science bank dapat mendeteksi penipuan dan menghindari kerugian secara real time. Karena sebagian besar aktivitas terkait transaksi perbankan dilakukan oleh nasabah melalui platform digital dan online, memastikan keamanan real-time dan akurasi transaksi menjadi bagian yang sangat penting.



Kesehatan



Salah satu contoh penerapan terbesar data science adalah pada sektor industri kesehatan. Bahkan, menurut laman Built In, Data Science pertama kali dikenalkan pada dunia lewat industri kesehatan pada tahun 2008. Pada tahun tersebut, Google menemukan bahwa mereka dapat memetakan wabah flu secara *real time* dengan melacak data lokasi pada pencarian terkait flu.

Itu tadi beberapa penerapan Data Science dalam industri, sekarang kita akan memasuki pembelajaran terkait dengan berpikir komputasional. Seperti yang disebutkan di awal, Data Science lahir dari gabungan ilmu statistik dan ilmu komputer. Pada bagian selanjutnya kita akan mempelajari lebih dalam mengenai *Computational Thinking* sebagai dasar utama yang diperlukan sebagai seorang profesional Data Science pada Ilmu Komputer.

Computational Thinking

Python Foundation - Intro to Python

Dalam membuat suatu model Machine Learning atau melakukan eksplorasi data untuk kebutuhan analisa, salah satu bahasa pemrograman yang cukup umum digunakan oleh profesional Data Science adalah Python.



Python merupakan salah satu bahasa pemrograman yang paling populer di dunia, terutama dalam pengembangan perangkat lunak dan data science. Selain karena Python hampir mirip dengan bahasa sehari-hari yang digunakan oleh manusia, berikut beberapa kelebihan Python lainnya:

1. Serbaguna, Python dapat digunakan untuk berbagai macam keperluan, seperti pengembangan perangkat lunak, data science (analisis data, visualisasi data dan pembuatan model machine learning), automasi tugas (pengolahan file, pengambilan data dari internet dan pengiriman email) dan lainnya.
2. *High end Programming* (pemrograman tingkat tinggi), hal ini membuat Python menjadi mudah dipelajari dan dipahami bahkan oleh pemula.
3. *Open Source*, Python adalah bahasa pemrograman sumber terbuka dan bebas. Ini memungkinkan para ilmuwan data untuk mengembangkan aplikasi dan alat mereka sendiri secara bebas dan tanpa biaya.
4. Bisa digunakan dengan berbagai cara seperti *Procedural way*, *Object-Oriented Programming* (OOP) atau *function way*.

Computational Thinking

Dasar-Dasar Python – Pengantar Python

Dalam proses pembelajaran di Startup Campus, penggunaan *interface* yang digunakan dibebaskan sesuai kenyamanan dan kemudahan. Namun, ada dua *environment* yang kami sarankan untuk digunakan, yaitu:



Google Colaboratory



jupyter

Jupyter Notebook

Dasar-Dasar Python – Variable

Variable pada Python dibuat dengan cara menugaskan nilai (*assigning value*) terhadap sebuah *element*. Jika variable yang telah dibuat kita butuhkan, maka kita dapat memanggil kembali variable tersebut. Berikut merupakan contoh untuk membuat variable pada Python:

```
● ● ●  
# Menentukan nama variable lalu mengisikan sebuah  
nilai  
variable = "Hello World!"  
print(variable)  
  
# Output: Hello World!
```

```
● ● ●  
# Contoh 2:  
nama = "Lingga"  
print("Perkenalkan nama Saya adalah ", nama)  
  
#Output: Perkenalkan nama Saya adalah Lingga
```

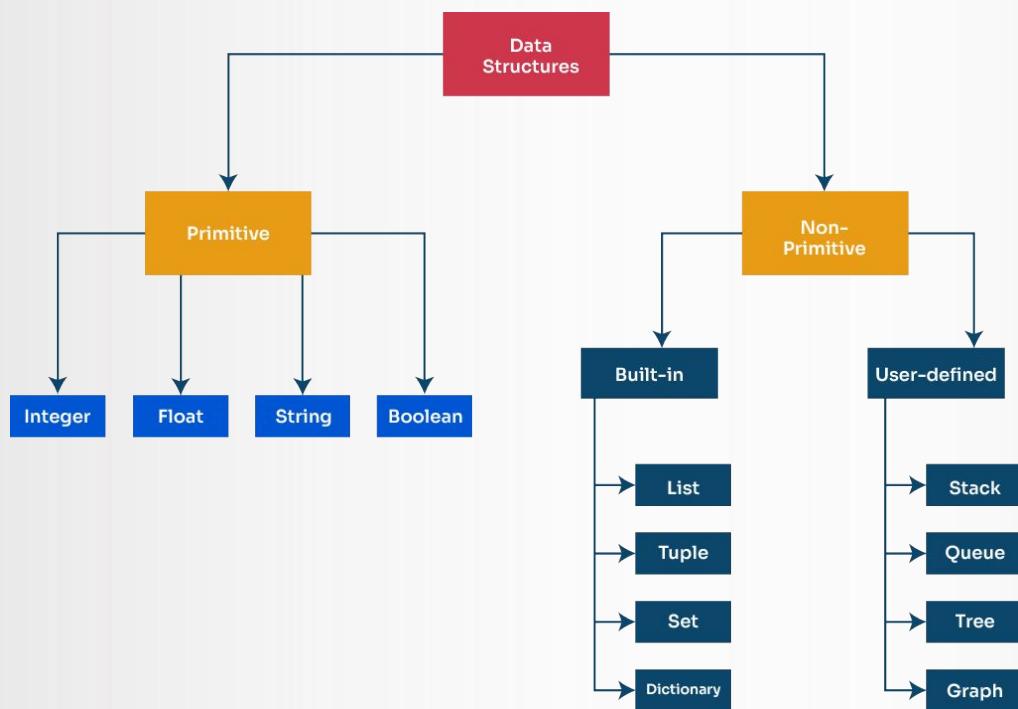
Computational Thinking

Dasar-Dasar Python - Variable

Terdapat beberapa hal yang perlu diperhatikan saat membuat variable, diantaranya:

1. Tidak dapat diawali oleh angka,
2. Hanya bisa terdiri dari karakter alphanumeric dan underscore (A-z, 0-9, dan _),
3. *Case sensitive*, artinya variable nama berbeda dengan Nama atau naMA,
4. Nama variable tidak boleh reserved keywords ([link](#))

Dasar-Dasar Python - Struktur Data

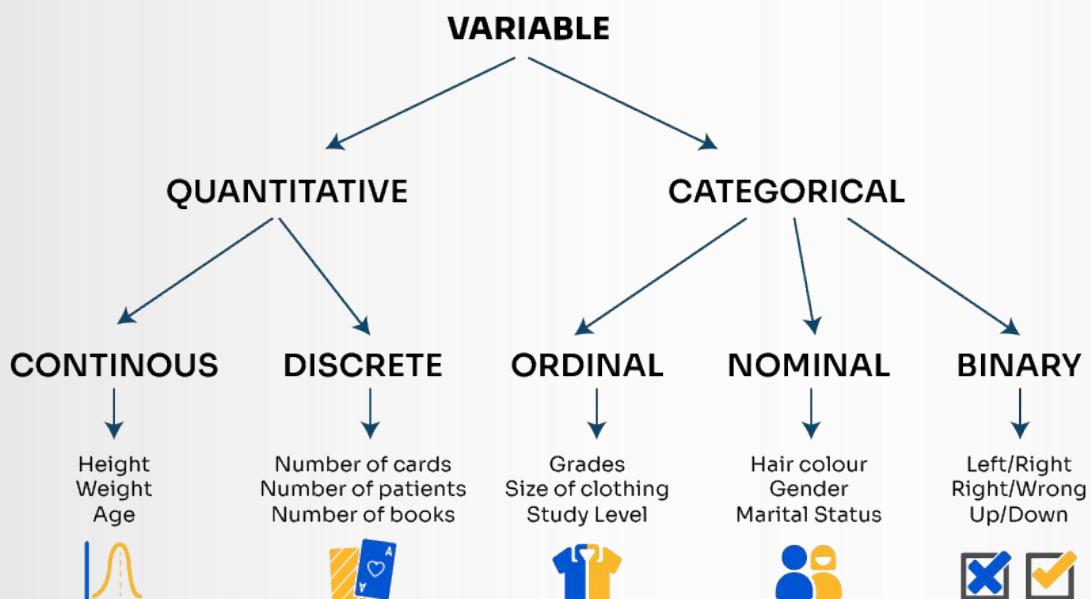


Secara umum, struktur data dapat diklasifikasikan menjadi dua jenis, yaitu primitif dan non-primitif. Struktur data primitif merupakan cara dasar untuk merepresentasikan data yang berisi nilai-nilai sederhana. Sedangkan, struktur data non-primitif merupakan cara yang lebih kompleks untuk merepresentasikan data yang berisi kumpulan nilai dalam berbagai format. Struktur data non-primitif selanjutnya dapat dikategorikan ke dalam struktur bawaan dan struktur yang ditentukan pengguna. Python menawarkan dukungan implisit untuk struktur bawaan yang mencangkup List, Tuple, Set dan Dictionary.

Computational Thinking

Dasar-Dasar Python – Tipe Variable

Variable pada Python diklasifikasikan menjadi kuantitatif (quantitative) dan kategorik (categorical). Variable Kualitatif merupakan variable yang menyatakan jumlah benda, seperti jumlah rokok dalam satu bungkus. Sedangkan variable kategorik merupakan variable yang mengungkapkan pengelompokan benda-benda, seperti jenis buah yang berbeda.



Variable Kuantitatif



Continuous (a.k.a Ratio), tipe variable ini digunakan untuk menggambarkan ukuran dan biasanya dibagi menjadi unit yang lebih kecil dari satu, misalnya 0.50 kg.



Discrete (a.k.a Interval), tipe variable ini digunakan untuk menggambarkan jumlah dan biasanya tidak dapat dibagi menjadi unit yang lebih kecil dari satu, misalnya 1 batang rokok.

Computational Thinking

Dasar-Dasar Python – Tipe Variable

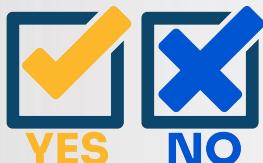
Variable Kategorik



Ordinal merupakan tipe variable yang mewakili data dengan urutan, misalnya peringkat.



Nominal merupakan tipe variable yang mewakili nama kelompok, seperti merek, nama spesies, dan jenis kelamin



Biner merupakan tipe variable yang mewakili data dengan hasil YA/TIDAK atau 1/0, misalnya KIRI atau KANAN.

Dasar-Dasar Python – Tipe Data

Dalam bahasa pemrograman Python, terdapat beberapa tipe data yang dapat digunakan untuk menyimpan nilai atau informasi. Setiap tipe data memiliki sifat dan karakteristiknya sendiri yang perlu dipahami agar dapat digunakan dengan benar dalam membuat program Python. Untuk mengetahui tipe data, karakteristik dan contohnya, mari simak tabel berikut:

Tipe Data		Contoh	Keterangan
Text	str (string)	“Hello world”	diapit oleh kutip (‘) / (“)
Numeric	int (integer)	1 atau 1680	bilangan bulat tanpa decimal
	float	1.6 atau 18.9	bilangan decimal
Boolean	bool	True atau False	Menyatakan benar atau salah
Sequence	list []	[1,2,3,4,5] atau ['hasan','bambang']	Menggunakan kurung siku
	tuple ()	(1,2,3,4,5)	Menggunakan kurung
	range	range (5)	
Mapping	dict	{'nama':'daffa'}, {'jumlah':5}	Menggunakan kurung kurawal
Set	set	{'hello', 3,4,6,6}	
	frozen set		

Computational Thinking

Dasar-Dasar Python – Tipe Data

Pada saat programmer bekerja dengan Python, mereka dapat memanfaatkan built-in data types yang telah disediakan oleh Python. Built-in data types adalah jenis tipe data yang sudah tersedia secara bawaan di Python, dan dapat langsung digunakan untuk mendeklarasikan dan menyimpan berbagai variabel dalam sebuah program.

Text Type: str

Numeric Types: int, float, complex

Mapping Type: dict

Sequence Types: list, tuple, range

Boolean Type: bool

Set Types: set frozenset

None Type:
NoneType

Binary Types: bytes, bytearray, memoryview

Dengan adanya built-in data types tersebut, programmer dapat dengan mudah mendeklarasikan variabel sesuai dengan kebutuhan program yang sedang dibuat. Misalnya, jika ingin menyimpan angka bulat, programmer dapat menggunakan tipe data integer. Sedangkan, jika ingin menyimpan kumpulan data yang dapat diubah, programmer dapat menggunakan tipe data list.

Dalam penggunaannya, programmer hanya perlu memanggil nama tipe data yang diinginkan ketika mendeklarasikan sebuah variabel. Python akan secara otomatis memperhitungkan jenis data tersebut dan menyediakan fitur-fitur khusus untuk memanipulasinya. Dengan begitu, para programmer dapat dengan mudah dan efisien mengembangkan program yang mereka buat menggunakan bahasa pemrograman Python.

Computational Thinking

Dasar-Dasar Python – Tipe Data

Dalam bahasa pemrograman Python, kita dapat menggunakan fungsi `type()` untuk mengetahui tipe atau jenis data yang terdapat pada suatu objek. Contohnya, jika kita ingin mengetahui tipe data dari sebuah variabel bernama `angka`, kita bisa menggunakan fungsi `type()` seperti berikut:

```
angka = 5
print(type(angka))
```

Output:

```
<class 'int'>
```

Dari hasil output tersebut, kita dapat mengetahui bahwa `angka` adalah sebuah objek bertipe data `int` (singkatan dari `integer` atau bilangan bulat). Dengan menggunakan fungsi `type()`, kita dapat memastikan bahwa kita sedang menggunakan tipe data yang tepat dalam program Python kita. Argumen pada fungsi `type()` tidak boleh kosong, karena akan menimbulkan error:



Computational Thinking

Dasar-Dasar Python – Tipe Data

Pada pembelajaran tipe operasi pada Python, salah satu hal yang perlu dipelajari adalah cara melakukan operasi matematika pada tipe data numerik seperti integer dan float. Untuk memudahkan pemahaman, berikut ini disajikan tabel yang berisi contoh operasi matematika pada tipe data numerik beserta hasilnya.

Python Operator	Apabila $x = 4$ dan $y = 2$	Hasil
Operator Aritmatika	Penjumlahan	$x+y$ 6
	Pengurangan	$x-y$ 2
	Perkalian	$x*y$ 8
	Pembagian	x/y 2
	Pangkat	$x**y$ 16
	Modulus (sisa)	$x//y$ 0
Operator Pembanding	Sama dengan	$x==y$ FALSE
	Tidak sama dengan	$x!=y$ TRUE
	Perbandingan (lebih besar/kecil.etc)	$x>y$ TRUE
		$x>=y$ TRUE
		$x<y$ FALSE
		$x<=y$ FALSE
Operator Penugasan	$x=x+2$	5
	$x = 5$ $y = 3$ <code>if</code> <code>if x>y: print ('benar')</code>	benar
Logical Operator	AND (Kedua syarat harus dipenuhi, maka TRUE)	$x = 5$ $y = 2$ $z = 1$ $x>y \text{ AND } x < z$ FALSE
	OR (salah satu syarat dipenuhi maka TRUE)	$OR = x < 4 \text{ or } y < 3 \#$ False, True <code>print (OR)</code> TRUE

Berdasarkan tabel yang diberikan, terdapat beberapa jenis operasi pada bahasa pemrograman Python seperti operasi aritmatika, operasi perbandingan, operasi penugasan, dan operasi logika. Setiap jenis operasi memberikan hasil perhitungan yang berbeda-beda tergantung dari input yang diberikan.

Computational Thinking

Dasar-Dasar Python – String

String pada Python merupakan tipe data yang terdiri dari kumpulan karakter. Pada saat Anda mempelajari bahasa pemrograman Python, Anda akan menemukan operasi `print("string")` yang sangat berguna untuk menampilkan teks atau nilai variabel pada konsol. Satu hal menarik tentang operasi ini adalah penggunaannya yang fleksibel, yaitu dengan menggunakan tanda kutip tunggal ('') atau tanda kutip ganda (""). Hal ini memungkinkan Anda untuk menampilkan teks yang mengandung tanda kutip tanpa harus melakukan escape character, seperti contoh berikut: `print("you're awesome")`. Anda dapat melihat bahwa Anda tidak perlu melakukan tindakan khusus untuk menampilkan tanda kutip tunggal di dalam kalimat yang dibungkus dengan tanda kutip ganda atau sebaliknya. Sebagai contoh lainnya, jika Anda ingin mencetak kalimat "She said, 'Hello!'", Anda bisa menuliskan operasi `print("She said, 'Hello!'")`.



```
# Contoh String 1
# \n digunakan untuk memisahkan string ke dalam dua
# baris
print('First line.\n Second Line')

# Output :
# First line
# Second line
```



```
# menghitung jumlah karakter dalam
# menggunakan fungsi len("string")

a = "ZulfikarLazuardiMaulana"
print(len(a))

#Output: 23
```

Dalam pemrograman, terkadang penggunaan `print()` menjadi sangat penting dalam melakukan debugging program, mengembangkan aplikasi, atau memberikan informasi yang berguna kepada pengguna. Dengan mengetahui cara yang tepat untuk menggunakan operasi `print("string")` pada Python, Anda akan dapat menampilkan pesan dengan lebih efektif dan efisien, sehingga dapat meningkatkan kualitas program yang Anda buat.

Computational Thinking

Dasar-Dasar Python – List

Ketika Anda mulai belajar bahasa pemrograman Python, Anda akan menemukan berbagai tipe data yang dapat digunakan untuk menyimpan data dalam program Anda. Salah satu tipe data yang paling umum dan sering digunakan adalah tipe data list. Dalam Python, list dapat digunakan untuk menyimpan berbagai jenis objek, seperti angka, string, atau bahkan objek lain seperti list atau dictionary. Tipe data list juga bersifat *mutable*, yang berarti bahwa kita dapat menambahkan nilai baru ke dalam list tersebut dengan menggunakan berbagai operasi Python yang tersedia.

```
● ● ●  
# Contoh List  
  
data = [1,2,3,4,5]  
print(str(data))  
  
#Output: [1,2,3,4,5]
```

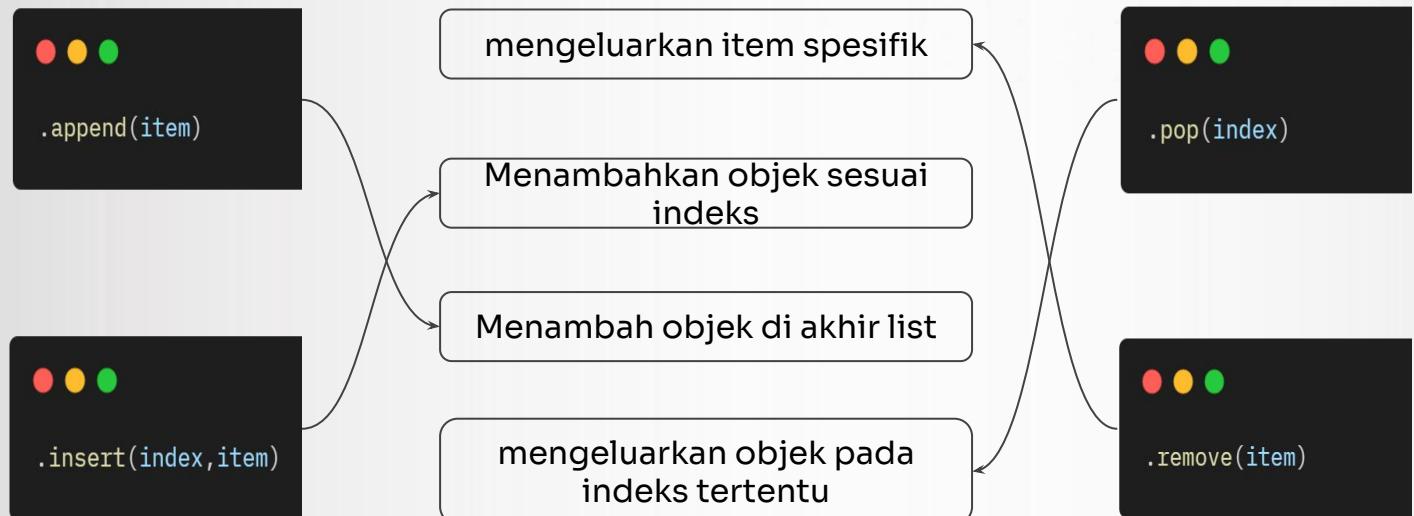
```
● ● ●  
# Untuk mengakses objek pada list dapat menggunakan metode  
indexing  
list = ["ini","contoh","list"]  
list[0]  
  
# Output: 'ini'
```

Untuk menuliskan list pada Python, Anda dapat menggunakan tanda kurung siku ([]), dengan menuliskan objek yang ingin Anda simpan di dalamnya, dipisahkan oleh koma (', ') jika terdapat lebih dari satu objek. Sebagai contoh, jika Anda ingin menyimpan beberapa nilai angka dalam sebuah list, Anda dapat menuliskan operasi `my_list = [1, 2, 3, 4, 5]`.

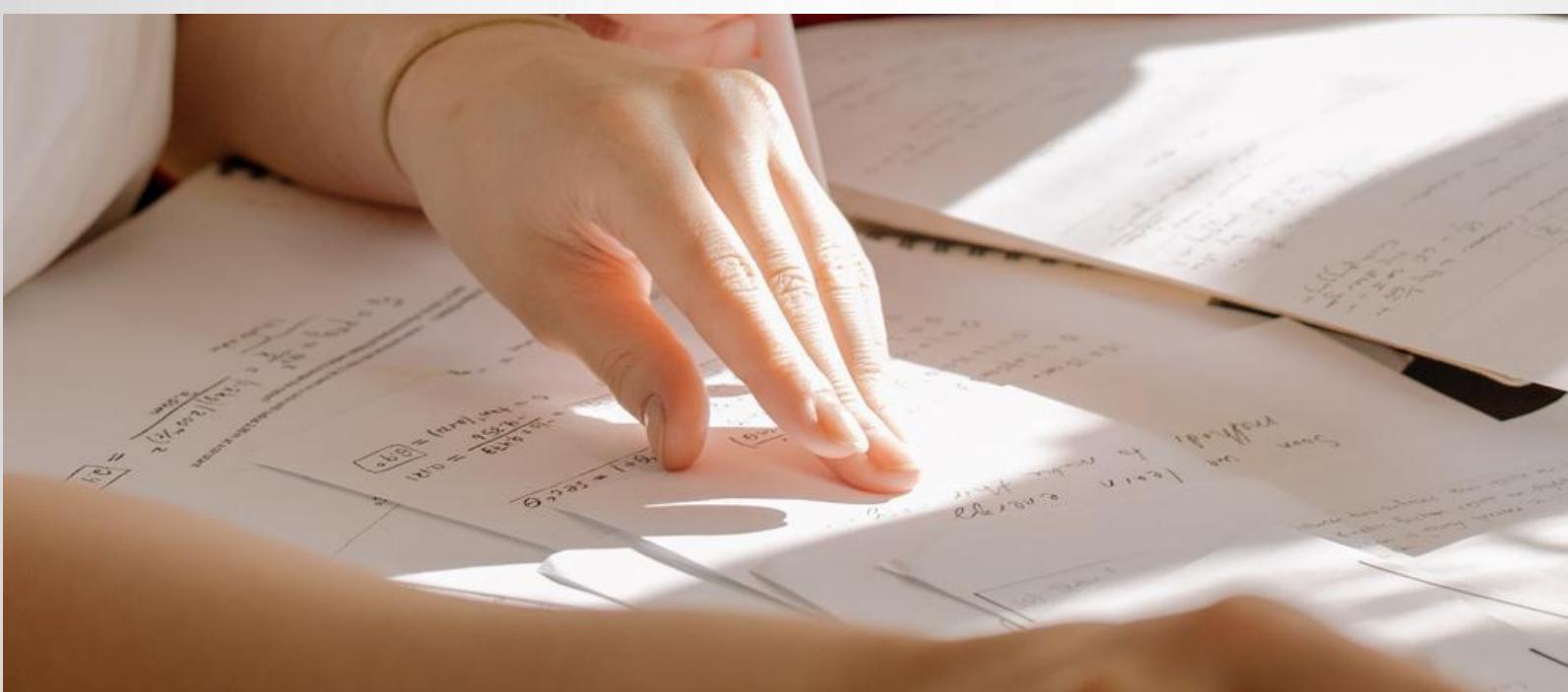
Dalam pemrograman Python, pemahaman tentang tipe data list sangat penting untuk membangun aplikasi dan program yang efektif dan efisien. Dengan memahami cara menuliskan dan menggunakan list dalam program Python, Anda akan dapat menyimpan dan mengelola data dengan lebih mudah dan terstruktur, sehingga dapat meningkatkan kualitas program Anda.

Computational Thinking

Dasar-Dasar Python – List

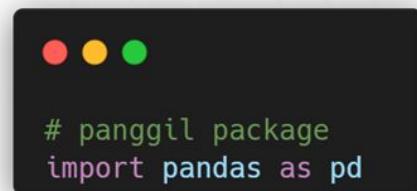


Dalam penggunaannya, tipe data list sangat fleksibel dan dapat diubah dengan berbagai cara. Anda dapat menambahkan atau menghapus objek dalam list dengan menggunakan operasi seperti `append()`, `insert()`, atau `remove()`. Sebagai contoh, jika Anda ingin menambahkan nilai baru ke dalam list yang sudah ada, Anda dapat menuliskan operasi `my_list.append(6)` untuk menambahkan nilai angka 6 ke dalam list.



Computational Thinking

Transforming Data: Python



Ketika berbicara mengenai pustaka perangkat lunak untuk manipulasi dan analisis data dalam bahasa pemrograman Python, maka nama yang tidak dapat dilewatkan adalah *pandas*. Pandas adalah salah satu pustaka perangkat lunak yang sangat populer dan digunakan secara luas oleh para ilmuwan data, analis data, dan programmer di seluruh dunia. Dengan pandas, para pengguna dapat memproses dan menganalisis data dengan lebih mudah, cepat, dan efektif. Pandas menawarkan struktur data yang cepat, fleksibel, dan ekspresif. Struktur data tersebut dirancang untuk membuat pekerjaan dengan data "relasional" atau "berlabel" menjadi mudah dan intuitif. Dengan berbagai kelebihan tersebut, tidak mengherankan jika pandas menjadi pilihan utama bagi para pengguna dalam melakukan manipulasi dan analisis data dalam bahasa pemrograman Python.

Selain itu, pandas juga menawarkan berbagai kelebihan yang sangat bermanfaat bagi para pengguna, antara lain:

1. Mudah digunakan: pandas menyediakan API yang intuitif dan mudah dipahami sehingga para pengguna dapat memproses data dengan lebih mudah dan cepat.
2. Fleksibilitas: pandas dapat digunakan untuk memproses berbagai jenis data, baik data numerik, waktu, maupun kategorikal.
3. Kinerja yang cepat: pandas dirancang untuk menangani data dengan jumlah yang sangat besar sehingga dapat memproses data dengan cepat dan efisien.
4. Integrasi dengan perangkat lunak lain: pandas dapat diintegrasikan dengan berbagai perangkat lunak seperti NumPy, SciPy, dan scikit-learn, sehingga para pengguna dapat memproses data dengan lebih luas dan efektif.

Computational Thinking

Numpy

Numpy merupakan salah satu library utama yang digunakan dalam pemrograman Python untuk analisis data dan ilmu data. Numpy berfungsi untuk melakukan operasi pada array dan matriks, sehingga memungkinkan untuk melakukan komputasi numerik secara efisien.

Berikut adalah beberapa konsep utama yang perlu dipahami dalam penggunaan Numpy dalam data science:

1. Array: Array adalah struktur data utama dalam Numpy, yang dapat digunakan untuk menyimpan data numerik dalam bentuk multidimensional. Array dapat berupa satu dimensi (vektor), dua dimensi (matriks), atau lebih.
2. Membuat Array: Untuk membuat array dalam Numpy, kita dapat menggunakan fungsi `np.array()`.

contoh:

```
● ● ●  
  
import numpy as np  
  
# membuat array satu dimensi  
a = np.array([1, 2, 3, 4, 5])  
  
# membuat array dua dimensi  
b = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

3. Operasi pada Array: Numpy menyediakan banyak fungsi untuk melakukan operasi pada array, seperti penjumlahan, pengurangan, perkalian, pembagian, dan lain sebagainya.

Computational Thinking

Contoh:

```
● ● ●  
import numpy as np  
  
a = np.array([1, 2, 3])  
b = np.array([4, 5, 6])  
  
# penjumlahan  
c = a + b  
  
# pengurangan  
d = a - b  
  
# perkalian  
e = a * b  
  
# pembagian  
f = a / b
```

4. Broadcasting: Broadcasting adalah fitur Numpy yang memungkinkan untuk melakukan operasi pada dua array dengan ukuran yang berbeda. Dalam broadcasting, Numpy akan melakukan penyesuaian otomatis pada ukuran array yang lebih kecil sehingga sesuai dengan ukuran array yang lebih besar.

```
● ● ●  
import numpy as np  
  
a = np.array([1, 2, 3])  
b = 2  
  
# broadcasting  
c = a * b
```

Computational Thinking

5. Fungsi Matematika: Numpy juga menyediakan banyak fungsi matematika yang sering digunakan dalam data science, seperti sin, cos, tan, exp, log, dan lain sebagainya.

Contoh:



```
import numpy as np

a = np.array([0, 1, 2, 3])

# fungsi sin
b = np.sin(a)

# fungsi cos
c = np.cos(a)

# fungsi exp
d = np.exp(a)
```

6. Indexing dan Slicing: Seperti pada struktur data Python lainnya, Numpy juga mendukung indexing dan slicing pada array.

Contoh:



```
import numpy as np

a = np.array([1, 2, 3, 4, 5])

# indexing
b = a[2]    # b = 3

# slicing
c = a[1:4]  # c = [2, 3, 4]
```

Computational Thinking

7. Fungsi Statistik: Numpy juga menyediakan banyak fungsi statistik yang sering digunakan dalam data science, seperti mean, median, mode, standar deviasi, dan lain sebagainya.

Contoh:

```
● ● ●  
import numpy as np  
  
a = np.array([1, 2, 3, 4, 5])  
  
#
```

Dalam kesimpulan, pemahaman tentang konsep dasar dari Numpy ini akan memberikan keuntungan besar dalam pengolahan data dan analisis numerik. Numpy membantu dalam meningkatkan efisiensi dan kecepatan pengolahan data, sehingga dapat memberikan hasil yang lebih baik dalam waktu yang lebih singkat.

Computational Thinking

Arrays and Matrix Handling

Arrays and Matrix handling adalah salah satu hal yang sangat penting dalam data science. Pemahaman tentang Arrays and Matrix handling akan membantu dalam memproses dan menganalisis data dengan lebih efisien. Pada dasarnya, Array adalah kumpulan nilai atau objek yang ditempatkan di dalam satu variabel dengan nama yang sama. Sedangkan Matrix adalah sebuah array dua dimensi yang terdiri dari baris dan kolom.

Arrays and Matrix handling sangat penting dalam data science karena memungkinkan pengolahan data yang lebih efisien, seperti operasi matematika, analisis statistik, dan visualisasi data.

Berikut ini adalah beberapa konsep dasar tentang Arrays and Matrix handling:

Deklarasi Array dan Matrix

- Untuk membuat Array dan Matrix dalam Python, kita dapat menggunakan sintaks berikut:

```
● ● ●  
# Deklarasi Array  
array_name = [value1, value2, value3, ...]  
  
# Deklarasi Matrix  
matrix_name = [[value1, value2, value3], [value4, value5, value6], [value7,  
value8, value9]]
```

- Akses Elemen Array dan Matrix

Untuk mengakses elemen dalam Array dan Matrix, kita dapat menggunakan indeks. Indeks dimulai dari nol, artinya indeks pertama adalah nol.

```
● ● ●  
# Akses elemen Array  
array_name[index]  
  
# Akses elemen Matrix  
matrix_name[row_index][column_index]
```

Computational Thinking

3. Mengubah Elemen Array dan Matrix

Kita dapat mengubah elemen dalam Array dan Matrix dengan menetapkan nilai baru pada indeks tertentu

contoh:

```
● ● ●

# Mengubah elemen Array
array_name[index] = new_value

# Mengubah elemen Matrix
matrix_name[row_index][column_index] = new_value
```

4. Operasi Matematika pada Array dan Matrix

Array dan Matrix dapat digunakan untuk melakukan berbagai operasi matematika seperti penjumlahan, pengurangan, perkalian, dan lain-lain. Operasi ini dapat dilakukan dengan mudah menggunakan fungsi-fungsi yang sudah disediakan oleh Python

Contoh:

```
● ● ●

# Penjumlahan Array
array1 + array2

# Penjumlahan Matrix
matrix1 + matrix2

# Perkalian Array
array1 * scalar

# Perkalian Matrix
np.dot(matrix1, matrix2)
```

Computational Thinking

5. Visualisasi Data dengan Array dan Matrix

Array dan Matrix dapat digunakan untuk membuat visualisasi data seperti heatmap dan scatterplot. Visualisasi ini dapat membantu kita memahami data dengan lebih baik.

Contoh:

```
● ● ●  
  
# Visualisasi Matrix  
import matplotlib.pyplot as plt  
plt.imshow(matrix, cmap='gray')  
plt.show()
```

Itulah beberapa konsep dasar tentang Arrays and Matrix handling dalam data science. Dengan memahami konsep ini, kita dapat lebih efisien dalam memproses dan menganalisis data.

Computational Thinking

Transformasi Data: Python

Dataframe adalah sebuah struktur data yang terdiri dari sebuah tabel dua dimensi yang berisi data, dimana setiap baris pada tabel ini mewakili sebuah observasi dan setiap kolom mewakili sebuah fitur atau variabel. Contohnya adalah file Excel atau CSV.

Untuk mengimpor data dari file Excel atau CSV ke lingkungan Python, kita dapat menggunakan perintah `pd.read_excel('file_name.xlsx')` untuk file Excel atau `pd.read_csv('file_name.csv')` untuk file CSV. Namun, jika file yang ingin diakses berada di direktori yang berbeda dengan file notebook Python, maka kita perlu menuliskan direktori secara lengkap.

Namun, perlu diingat bahwa kita harus memperhatikan tipe file dengan benar agar tidak terjadi kesalahan saat proses pengimporan. Setelah data berhasil diimpor, kita dapat memanggilnya kembali dengan menetapkan variabel untuk menyimpan data tersebut.

```
[ ] df = pd.read_csv('/content/Sample.csv', sep=',')
[ ] df
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States
2	3	CA-2016-152156	8/12/2016	8/16/2016	Second	DV-12045	Darrin Van	Corporate	United States

Pengimporan
data pada file CSV

Memanggil kembali variable dataframe

isi data pada dataframe

Computational Thinking

Transformasi Data: Python

Dalam pengolahan data menggunakan pandas/dataframe, terdapat beberapa operasi dasar yang sangat penting untuk dipahami. Salah satunya adalah informasi mengenai dataframe itu sendiri. Terdapat beberapa perintah dasar yang dapat digunakan untuk memperoleh informasi tersebut.

Info Mengenai Dataframe

- `df.shape` dapat digunakan untuk mengetahui dimensi dari dataframe yang ada (jumlah row dan column).
- `df.info()` dapat digunakan untuk mengetahui informasi dasar tentang dataframe seperti jumlah kolom, nama kolom, tipe kolom, dan null value.
- `df.describe()` dapat digunakan untuk mengetahui beberapa statistik numerikal dasar mengenai dataframe.
- `df.sort_values()` dapat digunakan untuk mengurutkan dataframe berdasarkan kolom dan nilai tertentu.
- `df['nama_kolom'].value_counts()` dapat digunakan untuk menjabarkan jumlah dari suatu value pada kolom tertentu.

Sampling

- `df.head(x)` mengambil x data teratas dari sebuah dataframe, dimana x merupakan jumlah baris yang ingin diambil. apabila x tidak dituliskan default nya 5
- `df.tail(x)` mengambil x data terbawah dari sebuah dataframe, dimana x merupakan jumlah baris yang ingin diambil. apabila x tidak dituliskan default nya 5
- `df.sample(x)` mengambil x baris data secara acak, dimana x merupakan jumlah baris yang ingin diambil

Computational Thinking

Transformasi Data: Python

Dalam pengolahan data menggunakan pandas/dataframe, terdapat beberapa teknik indexing dan filtering yang sangat penting untuk dipahami. Berikut merupakan beberapa teknik indexing dan filtering pada Python.

Indexing

Indexing merupakan teknik memanggil beberapa kolom/nilai dalam dataframe.

- `df.nama_kolom_1` → Memanggil satu kolom spesifik pada dataframe
- `df[['nama_kolom_1', 'nama_kolom_2', 'nama_kolom_3']]` → Memanggil kolom yang ditulis saja pada dataframe (filter)
- `df['nama_kolom_1'][x]` → akses value pada kolom 1 di row x

loc & iloc

loc adalah teknik indexing dan selection yang didasarkan pada label (nama kolom dan nama index). Perintah loc dapat digunakan untuk mengambil data berdasarkan label tertentu pada kolom dan index. iloc adalah teknik indexing dan selection yang didasarkan pada posisi index (numerik). Perintah iloc dapat digunakan untuk mengambil data berdasarkan posisi index pada kolom dan index. Berikut penerapan loc & iloc.

```

● ● ●

import pandas as pd

# Membuat dataframe
data = {'nama':['Andi', 'Budi', 'Caca', 'Dina', 'Eka'],
        'umur':[23, 25, 20, 27, 22],
        'kota':['Jakarta', 'Surabaya', 'Jakarta', 'Bandung',
        'Surabaya']}

df = pd.DataFrame(data)

# Menggunakan loc untuk indexing berdasarkan label
print(df.loc[0:2, ['nama', 'umur']])

# Menggunakan iloc untuk indexing berdasarkan posisi
print(df.iloc[0:2, 0:2])

```

Output: loc

	nama	umur
0	Andi	23
1	Budi	25
2	Caca	20

Output: iloc

	nama	umur
0	Andi	23
1	Budi	25

Computational Thinking

Transformasi Data: Python

Selecting/Filtering

Ketika bekerja dengan data dalam sebuah DataFrame, seringkali kita ingin menyeleksi atau memfilter data berdasarkan kriteria tertentu. Salah satu cara untuk melakukan seleksi/filtering adalah dengan menggunakan sintaks "`df[df['nama_kolom_1'] <operator> value]`". Sintaks ini memungkinkan kita untuk menyeleksi data berdasarkan kriteria tertentu. Sebagai contoh, jika kita ingin menampilkan semua kolom di DataFrame yang memiliki nilai "New York City" pada kolom "City", kita dapat menggunakan sintaks berikut:

```
● ● ●  
df['City'] == 'New York City'
```

Ini akan menampilkan semua kolom di DataFrame yang memiliki nilai "New York City" pada kolom "City".

jika kita ingin menampilkan kolom "City", "Category", dan "Customer Name" di DataFrame yang memiliki nilai "New York City" pada kolom "City" dan nilai "Technology" pada kolom "Category", kita dapat menggunakan sintaks berikut:

```
● ● ●  
df[['City', 'Category', 'Customer Name']][(df['City'] == 'New York City') &  
(df['Category'] == 'Technology')]
```

kita juga dapat menggunakan operator lain seperti "isnull()" untuk menampilkan row yang memiliki nilai null di sebuah kolom

```
● ● ●  
df[df['nama_kolom_1'].isnull()]
```

```
● ● ●  
df[df['nama_kolom'].str.contains(pola)]
```

Untuk menampilkan nilai dengan pola tertentu. Kita juga dapat menambahkan "^" atau "\$" pada pola untuk menentukan letak pola dalam nilai.

Computational Thinking

Transformasi Data: Python

Ketika bekerja dengan data dalam sebuah DataFrame, seringkali kita perlu mengubah tipe data kolom, menambah atau menghapus kolom, dan mengubah nama kolom agar lebih mudah dipahami.

Mengubah Tipe Data

Pertama, kita dapat mengubah tipe data pada kolom menggunakan metode ".astype()". Metode ini memungkinkan kita untuk merubah tipe data kolom menjadi float, integer, atau string sesuai kebutuhan. Misalnya, jika kita ingin merubah tipe data kolom menjadi integer, kita dapat menggunakan sintaks berikut:

Selain itu, kita juga dapat menggunakan metode ".str.lower()" atau ".str.upper()" untuk merubah semua karakter pada kolom menjadi huruf kecil atau huruf besar. Misalnya, jika kita ingin merubah semua karakter pada kolom "nama_kolom" menjadi huruf kecil, kita dapat menggunakan sintaks berikut:

Manipulasi Kolom

Selanjutnya, kita juga dapat menambahkan kolom baru pada DataFrame dengan mendefinisikan sebuah nama kolom baru dengan value dari kolom lainnya. Misalnya, jika kita ingin menambahkan kolom baru "kolom_baru" dengan value dari kolom "nama_kolom", kita dapat menggunakan sintaks berikut:

```
df['nama_kolom'].astype(int)
```

```
df['nama_kolom'].str.lower()
```

```
df['kolom_baru'] = df['nama_kolom']
```

Computational Thinking

Transformasi Data: Python

Manipulasi Kolom

Namun, jika kita ingin menghapus kolom pada DataFrame, kita dapat menggunakan metode ".drop()". Misalnya, jika kita ingin menghapus kolom "nama_kolom" pada DataFrame, kita dapat menggunakan sintaks berikut:



```
df = df.drop('nama_kolom', axis=1)
```

Selain itu, kita juga dapat menghapus row dengan value duplicate menggunakan metode ".duplicated()". Misalnya, jika kita ingin menghapus row yang memiliki nilai duplicate pada semua kolom, kita dapat menggunakan sintaks berikut:



```
df[df.duplicated(keep=False) == True]
```

Untuk mengubah nama kolom di suatu DataFrame, kita dapat menggunakan metode ".rename()". Misalnya, jika kita ingin mengubah nama kolom "nama_lama" menjadi "nama_baru", kita dapat menggunakan sintaks berikut:



```
df = df.rename(columns={'nama_lama': 'nama_baru'})
```

Namun, jika kita ingin mengubah beberapa nama kolom sekaligus, kita dapat menggunakan sintaks berikut:



```
df = df.rename(columns={'nama_lama_1': 'nama_baru_1',  
'nama_lama_2': 'nama_baru_2', 'nama_lama_3':  
'nama_baru_3', 'nama_lama_N': 'nama_baru_N'})
```

Computational Thinking

Transformasi Data: Python

Manipulasi Kolom

Terakhir, jika kita ingin mengubah semua nama kolom pada DataFrame, kita dapat menggunakan sintaks berikut:



```
df.columns = ['name_1', 'name_2', ..., 'name_N']
```

Semua metode ini dapat sangat berguna saat melakukan analisis data di Python dan memungkinkan kita untuk mengubah tipe data kolom, menambah atau menghapus kolom, serta mengubah nama kolom agar lebih mudah dipahami.

Grouping/Aggregation

Dalam analisis data, sering kali kita perlu mengelompokkan data berdasarkan kolom tertentu dan menghitung statistik tertentu pada kolom tersebut. Untuk melakukannya, kita dapat menggunakan method groupby() pada objek DataFrame pada library pandas. Jika kita ingin mengelompokkan data berdasarkan satu kolom dan menghitung satu statistik, kita dapat menggunakan syntax:

```
df.groupby('nama_kolom').agg({'kolom' : 'statistics'})
```

Jika kita ingin menghitung beberapa statistik pada satu kolom, kita dapat menggunakan syntax:

```
df.groupby('nama_kolom').agg({'kolom' : ['statistik_1', 'statistik_2', ...]})
```

Namun, jika kita ingin mengelompokkan data berdasarkan beberapa kolom dan menghitung beberapa statistik, kita dapat menggunakan syntax:

```
df.groupby('nama_kolom').agg({'nama_kolom_1' : ['statistik_1', 'statistik_2', ...], 'nama_kolom_2' : ['statistik_1', 'statistik_2', ...], ..., 'nama_kolom_n' : ['statistik_1', 'statistik_2', ...]})
```

Computational Thinking

Transformasi Data: Python

Grouping/Aggregation

Setelah mengelompokkan data, kita dapat mereset index dari DataFrame dengan syntax:

```
df = df.reset_index()
```

Selain itu, kita juga dapat membuat pivot table dengan menggunakan method pivot_table() pada objek DataFrame. Kita dapat membuat pivot table dengan syntax:

```
pd.pivot_table(df, index=[col_as_rows], columns=[col_as_columns],  
values=[col_as_values])
```

Menggabungkan Dataset

Ketika kita memiliki beberapa dataset, seringkali kita perlu menggabungkan dataset tersebut. Untuk menggabungkan dataset, kita dapat menggunakan method append() atau merge() pada objek DataFrame. Jika kita ingin menambahkan baris dari satu DataFrame ke DataFrame lainnya, kita dapat menggunakan syntax:

```
df.append(df_2)
```

Namun, jika kita ingin menggabungkan dua DataFrame berdasarkan kolom tertentu, kita dapat menggunakan syntax:

```
df_1.merge(df_2, left_on=kolom_1, right_on=kolom_2, how='inner')
```

Exporting Dataset to CSV/Excel (Flat Files)

Terakhir, jika kita ingin menyimpan DataFrame ke dalam format file CSV atau Excel, kita dapat menggunakan method to_csv() atau to_excel(). Syntax untuk menyimpan DataFrame ke dalam file CSV atau Excel adalah sebagai berikut:

```
df.to_csv('nama_file.csv', index=False)  
df.to_excel('nama_file.xlsx', index=False)
```

Computational Thinking

Visualization



Visualization atau visualisasi data adalah teknik dalam data science yang digunakan untuk mempresentasikan data secara grafis atau visual sehingga lebih mudah dipahami dan dianalisis. Dalam visualisasi data, kita menggunakan berbagai jenis grafik atau plot, seperti bar chart, line chart, scatter plot, heatmap, dan sebagainya.

Berikut ini adalah beberapa konsep penting dalam visualisasi data dalam data science:

1. Tujuan Visualisasi Data

Tujuan utama dari visualisasi data adalah untuk membuat data lebih mudah dipahami dan dianalisis oleh manusia. Dalam visualisasi data, kita dapat menampilkan data dalam bentuk grafik atau plot yang dapat memberikan informasi lebih jelas dan terstruktur. Hal ini dapat membantu kita dalam membuat keputusan yang lebih baik berdasarkan data yang ada.

2. Memilih Tipe Grafik yang Tepat

Pemilihan tipe grafik yang tepat sangat penting dalam visualisasi data. Tipe grafik yang dipilih harus sesuai dengan jenis data yang akan dipresentasikan dan tujuan visualisasi. Misalnya, untuk data kategorik, kita dapat menggunakan bar chart atau pie chart, sedangkan untuk data numerik, kita dapat menggunakan line chart atau scatter plot.

Computational Thinking

3. Memilih Warna yang Tepat

Pemilihan warna yang tepat juga sangat penting dalam visualisasi data. Warna yang dipilih harus mudah dibedakan dan memiliki kontras yang cukup sehingga grafik lebih mudah dipahami. Warna yang dipilih juga harus sesuai dengan jenis data yang ditampilkan dan tujuan visualisasi.

4. Menggunakan Animasi dan Interaktivitas

Animasi dan interaktivitas dapat membuat visualisasi data lebih menarik dan interaktif. Dalam visualisasi data yang kompleks, animasi dapat membantu pengguna dalam memahami data dengan lebih baik. Interaktivitas juga dapat memberikan pengalaman yang lebih interaktif dan memungkinkan pengguna untuk melakukan eksplorasi data yang lebih mendalam.

5. Memilih Tools atau Library yang Tepat

Dalam visualisasi data, kita dapat menggunakan berbagai tools atau library seperti Matplotlib, Seaborn, Plotly, dan lain-lain. Pemilihan tools atau library yang tepat sangat penting dalam memudahkan proses visualisasi data. Setiap tools atau library memiliki kelebihan dan kekurangan masing-masing, oleh karena itu, kita harus memilih yang paling sesuai dengan kebutuhan dan tujuan visualisasi.

Dalam kesimpulan, visualisasi data merupakan teknik penting dalam data science yang digunakan untuk mempresentasikan data secara grafis atau visual sehingga lebih mudah dipahami dan dianalisis. Dalam visualisasi data, kita harus memperhatikan beberapa konsep penting seperti tujuan visualisasi, pemilihan tipe grafik yang tepat, menampilkan data yang relevan, pemilihan warna yang tepat, penggunaan animasi dan interaktivitas, serta pemilihan tools atau library yang tepat.

Computational Thinking

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) adalah teknik dalam data science yang digunakan untuk mengeksplorasi dan menganalisis data sebelum dilakukan pemodelan atau analisis yang lebih lanjut. EDA bertujuan untuk memahami karakteristik data, mencari pola, mengidentifikasi outlier, dan menentukan variabel yang paling berpengaruh terhadap target.

Berikut ini adalah beberapa konsep penting dalam Exploratory Data Analysis (EDA) dalam data science:

1. Memahami Struktur Data

Sebelum melakukan analisis, kita harus memahami struktur data yang akan digunakan. Struktur data dapat berupa tabel, file CSV, atau data dalam format lainnya. Kita perlu memahami tipe data yang ada dalam data, jumlah baris dan kolom, serta properti lainnya seperti header, indeks, dan sebagainya.

2. Menentukan Variabel Utama

EDA bertujuan untuk menemukan variabel yang paling berpengaruh terhadap target. Oleh karena itu, kita perlu menentukan variabel utama yang akan dianalisis. Variabel utama ini dapat berupa variabel target (target variable) atau variabel yang mempengaruhi target (predictor variable).

3. Melakukan Deskripsi Data

Deskripsi data dilakukan untuk memahami karakteristik data yang digunakan. Deskripsi data dapat berupa ringkasan statistik seperti rata-rata, standar deviasi, median, dan lain-lain. Deskripsi data juga dapat dilakukan dengan membuat grafik atau plot untuk menampilkan pola yang ada dalam data.

Computational Thinking

4. Menganalisis Hubungan Antar Variabel

EDA juga dilakukan untuk menganalisis hubungan antar variabel dalam data. Analisis ini dapat dilakukan dengan membuat grafik atau plot yang menampilkan hubungan antar variabel. Misalnya, scatter plot untuk menampilkan hubungan antara dua variabel numerik atau heatmap untuk menampilkan korelasi antara dua variabel.

5. Identifikasi Outlier dan Missing Value

Outlier dan missing value dapat mempengaruhi hasil analisis. Oleh karena itu, kita perlu mengidentifikasi outlier dan missing value yang ada dalam data. Outlier dapat diidentifikasi dengan menggunakan boxplot atau histogram. Sedangkan missing value dapat diidentifikasi dengan melihat jumlah data yang hilang pada setiap variabel.

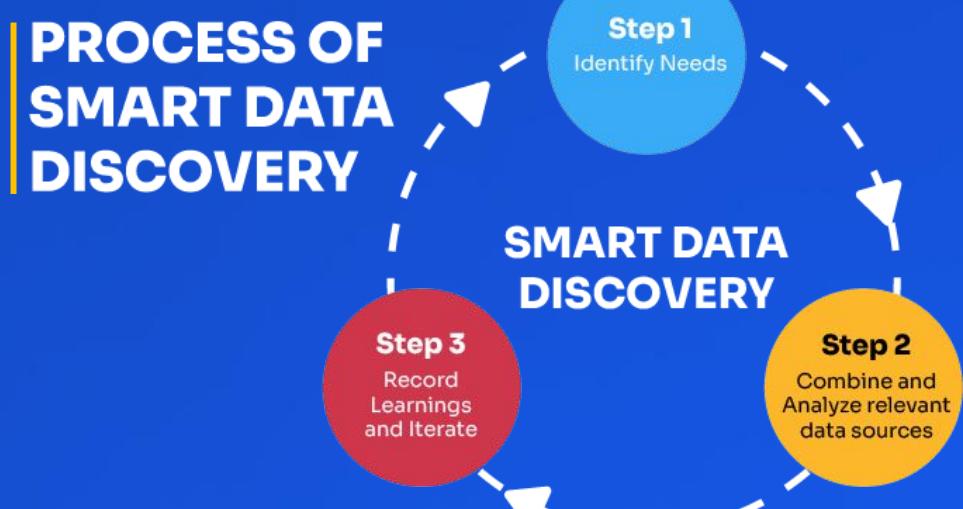
6. Menggunakan Statistik Inferensial

Statistik inferensial dapat digunakan dalam EDA untuk menguji hipotesis dan menentukan signifikansi dari hubungan antar variabel. Statistik inferensial juga dapat digunakan untuk memilih variabel yang paling berpengaruh terhadap target.

Sebagai kesimpulan, Exploratory Data Analysis (EDA) merupakan teknik penting dalam data science untuk mengeksplorasi dan menganalisis data sebelum dilakukan pemodelan atau analisis yang lebih lanjut. Dalam EDA, kita perlu memahami struktur data, menentukan variabel utama, melakukan deskripsi data, menganalisis hubungan antar variabel, mengidentifikasi outlier dan missing value, serta menggunakan statistik inferensial. Dengan melakukan EDA dengan baik, kita dapat memahami karakteristik data dengan lebih baik dan menghasilkan hasil analisis yang lebih akurat dan signifikan.

Computational Thinking

Data Discovery – Pengertian



Apa itu Data Discovery?

Data Discovery adalah istilah yang digunakan para analis untuk mengumpulkan data dari berbagai sumber dan mendeteksi pola-pola yang ada. Proses ini dilakukan tanpa perlu membangun model yang rumit. Banyak perusahaan menggunakan Data Discovery sebagai bagian dari perangkat lunak intelijen bisnis (BI) untuk melihat organisasi mereka secara visual melalui dasbor sederhana.

Pendekatan Dalam Data Discovery

Pendekatan dalam Data Discovery dapat membantu perusahaan dalam memahami dan menganalisis kumpulan data. Terdapat dua jenis pendekatan yang umum digunakan, yaitu Manual Discovery dan Smart Data Discovery.

1. Manual Discovery

Dalam Manual Discovery, para analis akan melihat jenis data yang ada dan di mana data tersebut disimpan. Data stewards bertanggung jawab untuk menangani aturan dan standar dokumen aset data. Selanjutnya, para Data Scientist akan membuat konsep peta agar dapat memahami semua data yang diperoleh suatu perusahaan.

Computational Thinking

Data Discovery – Pegertian

2. Smart Data Discovery

Dengan berkembangnya zaman, Smart Data Discovery menjadi lebih populer. Pendekatan ini menggunakan augmented intelligence dan teknologi machine learning untuk mempersiapkan, membuat konsep, mengintegrasikan, dan menyajikan data melalui visual, pola, dan wawasan yang tersembunyi. Pendekatan ini juga mendorong perusahaan untuk mempertimbangkan seluruh analisis dari kumpulan data, dan menggunakan mesin untuk menerima pertanyaan, melakukan pemrosesan dalam black box, dan menghasilkan jawaban yang masuk akal. Dengan demikian, proses dalam perusahaan semakin efektif dan efisien.

Data Discovery – Mengapa?

Mengapa Data Discovery Penting?

Dalam bisnis yang sukses, keahlian yang menjadi penilaian utama adalah ketangkasan. Namun, ketangkasan bisnis tidak dapat dicapai hanya dengan insting belaka. Dalam hal ini, Data Discovery memiliki peran penting sebagai bagian dari dasar ketangkasan bisnis. Melalui Data Discovery, perusahaan mampu memperoleh pandangan menyeluruh tentang operasional bisnisnya sehingga dapat lebih memahami dan menangani masalah yang terjadi.

Berkat kemajuan teknologi informasi, Data Discovery semakin populer dalam dunia bisnis. Banyak perusahaan yang kini memperlakukan data sebagai aset berharga mereka. Informasi yang dikumpulkan tentang pelanggan dan operasional perusahaan menjadi kunci dalam membedakan model bisnis mereka dari pesaing. Dengan Data Discovery, perusahaan dapat memanfaatkan informasi tersebut sebagai keunggulan kompetitif, baik dalam bentuk inovasi produk, meningkatkan pengalaman pengguna, maupun meningkatkan efisiensi operasional.

Computational Thinking

Data Discovery – Kategori

Kategori Data Discovery

Kategori Data Discovery memainkan peran penting dalam memudahkan perusahaan memahami data mereka. Sebagai proses yang menggabungkan analisis, pemodelan, dan keluaran visual, Data Discovery memberikan pemahaman yang lebih dalam tentang interaksi antara berbagai aliran data. Untuk memaksimalkan manfaat dari proses ini, perusahaan perlu memahami hubungan di antara data mereka dan menggunakan alat penemuan visual dan perangkat lunak intelijen bisnis (BI) untuk menggali tiga kategori Data Discovery berikut.

1. Persiapan Data

Data Discovery datang dalam berbagai bentuk, menggabungkan analisis, pemodelan dan keluaran visual. Untuk mendapatkan nilai maksimal dari proses tersebut, bisnis perlu memahami interaksi di antara berbagai aliran data mereka. Persiapan Data membantu perusahaan mempersiapkan data mereka dengan cara yang tepat sehingga dapat dianalisis dengan efektif.



2. Analisis Visual

Kedua, Analisis Visual memungkinkan perusahaan untuk memvisualisasikan data mereka dalam bentuk bagan, diagram aliran data, atau dashboard. Ini membantu mereka memahami hubungan di antara perspektif data mereka dengan cara yang mudah dicerna.

Computational Thinking

Data Discovery – Kategori

Misalnya, tim User Interface dapat dengan mudah mempelajari bagaimana pelanggan menggunakan produk mereka dan menyesuaikan pekerjaan mereka. Dan tim keuangan dapat memperoleh gambaran tentang biaya pendapatan untuk setiap departemen dalam bisnis dan menunjukkan area yang perlu ditingkatkan.



3. Analisis Visual Lanjutan

Ketiga, Analisis Visual Lanjutan adalah kategori Data Discovery yang lebih rumit, menggabungkan deskripsi dan visual untuk memberikan pemahaman yang lebih dalam tentang data perusahaan.

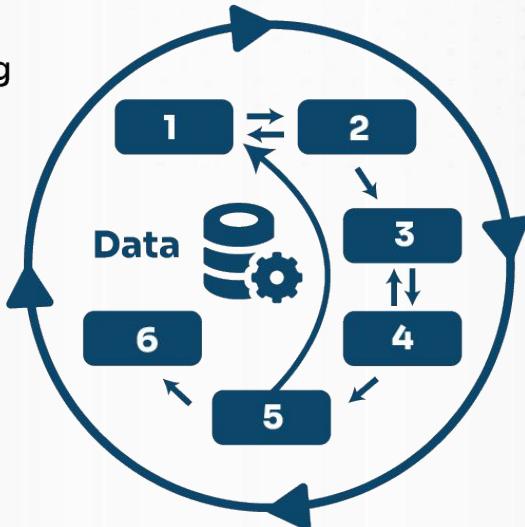


Dalam hal ini, keluaran analitik berfokus pada deskripsi kecil dari data itu sendiri, sementara analitik terpandu memungkinkan bisnis untuk melihat implikasi yang lebih besar dari upaya Data Discovery mereka, termasuk hubungan antara perspektif data science yang berbeda. Analisis lanjutan yang dipandu sangat berharga untuk bisnis yang menavigasi peralihan ke digital area, di mana integrasi data web dengan aliran data yang ada sangat penting untuk pengambilan keputusan strategis. Dengan menggunakan tiga kategori Data Discovery ini, perusahaan dapat memperoleh gambaran besar tentang datanya dalam satu format yang mudah dicerna dan membuat keputusan yang lebih baik dan lebih efektif.

Computational Thinking

Data Discovery – Process Data Discovery

- 1 Business Understanding
- 2 Data Understanding
- 3 Data Preparation
- 4 Modelling
- 5 Evaluating
- 6 Deployment



Data Discovery adalah sebuah proses penting dalam pengumpulan dan analisis data di sebuah perusahaan. Proses ini melibatkan beberapa langkah yang harus ditempuh untuk memastikan keberhasilan penggunaan data dalam meningkatkan bisnis. Selain itu, Data Discovery juga merupakan sebuah proses yang berulang, yang berarti perusahaan dapat terus mengumpulkan, menganalisis, dan menyempurnakan pendekatan Data Discovery mereka dari waktu ke waktu. Dalam hal ini, perusahaan dapat memanfaatkan hasil dan saran dari pemangku kepentingan bisnis untuk meningkatkan dan mengembangkan pendekatan mereka dalam menggunakan data. Sebagai hasil dari pengembangan ini, perusahaan akan semakin mampu untuk membuat keputusan bisnis yang lebih akurat dan efektif berdasarkan informasi yang diperoleh dari data. Berikut langkah untuk melakukan proses Data Discovery:

1. Identifikasi Kebutuhan



Langkah pertama adalah mengidentifikasi kebutuhan. Dalam langkah ini, perusahaan perlu mempertimbangkan jenis data yang akan berguna untuk diketahui dan tetap terbuka terhadap wawasan tak terduga. Sebagai contoh, distributor barang konsumen dapat memeriksa kembali data logistiknya untuk mengurangi limbah makanan selama pengiriman.

Computational Thinking

Data Discovery – Process Data Discovery

2. Gabungkan Data Dari Sumber Yang Relevan



Setelah identifikasi kebutuhan, langkah selanjutnya adalah menggabungkan data dari sumber yang relevan. Hal ini penting karena tidak ada aliran data tunggal yang mencakup semuanya. Proses ini disebut sebagai data crunching.

3. Bersihkan dan Siapkan Data



Langkah ketiga adalah membersihkan dan menyiapkan data. Langkah ini sangat penting untuk membantu perusahaan mengurangi "gangguan" dalam data mereka dan mendapatkan hasil yang lebih jelas dari analisis data mereka.

4. Analisis Data



Setelah data bersih dan siap untuk dianalisis, langkah keempat adalah melakukan analisis data. Dengan informasi yang digabungkan dari beberapa departemen, terintegrasi dengan data eksternal dan dibersihkan untuk analisis, perusahaan dapat memperoleh gambaran lengkap tentang operasi mereka dan memecahkan permasalahan operasional yang menghalangi efisiensi.

5. Rekam Pembelajaran dan Ulangi



Langkah terakhir adalah merekam pembelajaran dan mengulangi proses. Karena Data Discovery bukanlah proses satu kali, perusahaan perlu terus memperbaiki pendekatan mereka dengan memanfaatkan hasil dan saran dari pemangku kepentingan bisnis dan terus berkomitmen untuk perbaikan terus-menerus.

Computational Thinking

Python MySQL - Installing MySQL

Jika Anda ingin bekerja dengan MySQL menggunakan Python, ada beberapa prosedur yang harus diikuti, diantaranya:

1. Pertama-tama, Anda perlu menghubungkan ke database dengan menggunakan modul Python yang sesuai.
2. Kemudian, Anda perlu membuat objek untuk database Anda, yang akan memungkinkan Anda untuk mengakses data yang tersimpan di dalamnya.
3. Setelah itu, Anda dapat menjalankan query SQL untuk mengekstrak data yang dibutuhkan.
4. Ketika Anda selesai, jangan lupa untuk mengambil catatan dari hasilnya agar dapat digunakan nanti. Namun, jika Anda melakukan perubahan pada tabel, penting untuk memberi tahu Database agar dapat memperbarui tabel tersebut.

Dengan menggunakan bahasa pemrograman Python dan MySQL, Anda dapat dengan mudah mengelola dan mengakses data Anda dengan cepat dan efisien. Dengan demikian, penggunaan Python dan MySQL sangat penting bagi banyak aplikasi perangkat lunak dan dapat membantu dalam mengoptimalkan proses bisnis.

Installing MySQL

MySQL adalah salah satu database yang paling populer. Untuk mengikuti tutorial ini, kita perlu menginstal MySQL dari situs web resmi MySQL. Berikut adalah langkah-langkah untuk menginstal MySQL:

- Buka situs web resmi MySQL (<https://www.mysql.com/>) dan pilih "Downloads" di bagian atas halaman.
- Pilih versi MySQL yang sesuai dengan sistem operasi Anda.
- Ikuti petunjuk instalasi yang muncul di layar.

Setelah menginstal MySQL, kita perlu menginstal server MySQL untuk mengikuti tutorial ini.

Computational Thinking

Python MySQL - Installing MySQL

Berikut adalah langkah-langkah untuk menginstal server MySQL:

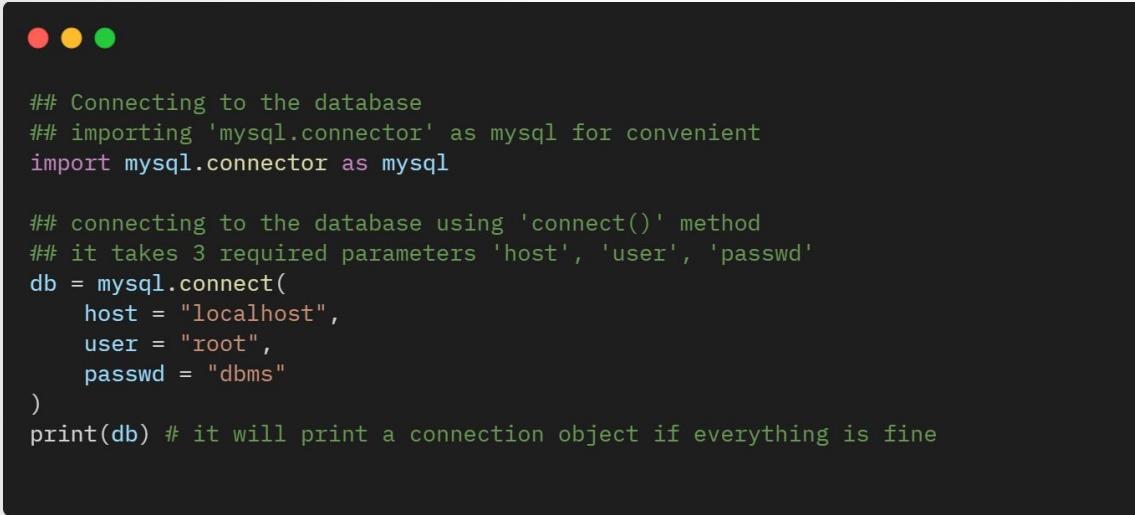
- Buka aplikasi MySQL yang sudah terinstal di komputer Anda.
- Buat koneksi ke server MySQL dengan menggunakan informasi login yang diberikan.
- Ikuti petunjuk instalasi yang muncul di layar.

Selanjutnya, kita perlu menginstal mysql.connector untuk Python. `mysql.connector` adalah library Python yang memungkinkan kita menghubungkan Python Script ke database MySQL



Python MySQL - Connecting and Creating

Sekarang, mari kita belajar bagaimana terhubung ke database dengan menggunakan username dan password MySQL. Jika Anda lupa nama pengguna atau kata sandi yang digunakan sebelumnya, jangan khawatir! Anda bisa membuat pengguna baru dengan kata sandi yang baru juga. Untuk membuat pengguna baru, Anda bisa melihat dokumentasi resmi MySQL yang tersedia. Setelah memiliki username dan password yang valid, langkah selanjutnya adalah menyambungkan ke database dengan menggunakan nama pengguna dan kata sandi tersebut.



The image shows a dark-themed terminal window. At the top, there are three small colored dots (red, yellow, green) as part of the window's decorative border. Below them, the Python code for connecting to a MySQL database is shown. It starts with importing the connector module, then creating a connection object using the connect() method with host, user, and passwd parameters set to "localhost", "root", and "dbms" respectively. Finally, it prints the connection object.

```
## Connecting to the database
## importing 'mysql.connector' as mysql for convenient
import mysql.connector as mysql

## connecting to the database using 'connect()' method
## it takes 3 required parameters 'host', 'user', 'passwd'
db = mysql.connect(
    host = "localhost",
    user = "root",
    passwd = "dbms"
)
print(db) # it will print a connection object if everything is fine
```

Computational Thinking

Python MySQL - Connecting and Creating

```
<mysql.connector.connection_cext.CMySQLConnection object at  
0x0000020C26A84C50>
```

Statement diatas merupakan output hasil fungsi `print(db)` untuk koneksi objek jika keseluruhan koneksi berjalan dengan baik

Python MySQL - Create Database

Setelah berhasil membuat koneksi database MySQL dengan Python, selanjutnya kita akan membuat database dengan nama “datacamp”. Untuk membuat database di MySQL, kita dapat menggunakan SQL Statement `CREATE DATABASE database_name`, seperti pada contoh dibawah ini:

```
● ● ●

import mysql.connector as mysql
db = mysql.connect(
    host = "localhost",
    user = "root",
    passwd = "dbms"
)

## creating an instance of 'cursor' class which is used to execute the 'SQL'
statements in 'Python'
cursor = db.cursor()

## creating a database called 'datacamp'
## 'execute()' method is used to compile a 'SQL' statement
## below statement is used to create the 'datacamp' database
cursor.execute("CREATE DATABASE datacamp")
```

Saat Anda mencoba membuat database baru di MySQL, Anda mungkin akan mendapatkan pesan kesalahan yang mengatakan bahwa database tersebut sudah ada. Untuk menghindari masalah tersebut, pastikan bahwa database yang ingin Anda buat belum ada di dalam MySQL. Untuk memeriksa apakah ada database yang sudah ada, Anda dapat menggunakan kode berikut di MySQL: `SHOW DATABASE`. Perintah ini akan menampilkan daftar semua database yang sudah ada di MySQL.

Computational Thinking

Python MySQL - Create Database

Berikut ini merupakan contoh yang dapat kita terapkan dalam *syntax* untuk memeriksa apakah ada database yang sudah ada:

```
● ● ●

import mysql.connector as mysql
db = mysql.connect(
    host = "localhost",
    user = "root",
    passwd = "dbms"
)
cursor = db.cursor()

## executing the statement using 'execute()' method
cursor.execute("SHOW DATABASES")

## 'fetchall()' method fetches all the rows from the last executed statement
databases = cursor.fetchall() ## it returns a list of all databases present

## printing the list of databases
print(databases)

## showing one by one database
for database in databases:
    print(database)
```

```
[('datacamp',), ('information_schema',), ('mysql',), ('performance_schema',),
('sakila',), ('sys',), ('world',)]
('datacamp',)
('information_schema',)
('mysql',)
('performance_schema',)
('sakila',)
('sys',)
('world',)
```

Output tersebut akan menampilkan nama-nama database yang telah terdaftar di dalam sistem basis data MySQL yang sedang Anda gunakan. Dengan mengetahui daftar database yang sudah ada, Anda dapat memastikan bahwa tidak terdapat database dengan nama yang sama dengan yang ingin Anda buat.

Computational Thinking

Python MySQL - Create Table

Saat kita ingin menyimpan informasi dalam sebuah sistem basis data, langkah pertama yang harus dilakukan adalah membuat sebuah tabel. Sebelum membuat tabel tersebut, kita harus memastikan bahwa database yang ingin kita gunakan sudah dipilih terlebih dahulu. Untuk memilih database yang akan digunakan, kita dapat menjalankan syntax berikut:

```
import mysql.connector as mysql
db = mysql.connect(
    host = "localhost",
    user = "root",
    passwd = "dbms",
    database = "datacamp"
)
```

Setelah terhubung ke database "datacamp", kita dapat mulai membuat tabel yang akan digunakan untuk menyimpan informasi dengan menggunakan *statement "CREATE TABLE"*. Dalam perintah ini, kita harus memberikan nama tabel yang ingin dibuat dan menentukan kolom-kolom yang akan digunakan untuk menyimpan informasi. Kolom tersebut harus diberi nama dan jenis data yang akan disimpan di dalamnya.

Contohnya, jika kita ingin membuat tabel yang akan digunakan untuk menyimpan data pelanggan, kita dapat menggunakan perintah berikut:

```
cursor = db.cursor()

## creating a table called 'users' in the 'datacamp' database
cursor.execute("""CREATE TABLE nama_tabel (
    id INT PRIMARY KEY,
    nama VARCHAR(50),
    alamat VARCHAR(100),
    nomor_telepon VARCHAR(20))""")
```

Computational Thinking

Python MySQL - Create Table

Dalam syntax tersebut, kita memberikan nama tabel "nama_tabel" dan menentukan empat kolom, yaitu "id", "nama", "alamat", dan "nomor_telepon". Kolom "id" diberi tipe data INT dan dinyatakan sebagai PRIMARY KEY, sedangkan kolom "nama", "alamat", dan "nomor_telepon" masing-masing diberi tipe data VARCHAR dan batasan panjang karakter yang dapat disimpan di dalamnya.

Setelah berhasil membuat tabel "user" di dalam database "datacamp", langkah selanjutnya adalah memastikan bahwa tabel tersebut sudah tersedia dan terdaftar di dalam database. Kita dapat melakukan hal ini dengan menggunakan pernyataan SQL "`SHOW TABLES`". Perintah ini akan menampilkan semua tabel yang ada di dalam database yang sedang aktif, sehingga kita dapat memastikan bahwa tabel yang telah kita buat sudah terdaftar di dalam database. Kita dapat menjalankan perintah berikut di MySQL untuk melihat daftar tabel yang ada di dalam database "datacamp":

```
● ● ●

cursor = db.cursor()
## getting all the tables which are present in 'datacamp' database

cursor.execute("SHOW TABLES")
tables = cursor.fetchall() ## it returns list of tables present in the database

## showing all the tables one by one
for table in tables:
    print(table)
```

```
('user',)
('orders',)
('products',)
```

Output tersebut menunjukkan bahwa terdapat tiga tabel yang ada di dalam database "datacamp", yaitu tabel "user", "orders", dan "products". Setiap tabel ditampilkan dalam bentuk tuple dengan satu elemen yang berisi nama tabelnya. Output ini akan dihasilkan jika ketiga tabel tersebut sudah terdaftar di dalam database yang dihubungkan dengan menggunakan syntax yang sama.

Computational Thinking

Python MySQL - Create Table

Primary Key



Dalam sebuah tabel, Primary Key adalah nilai yang unik dan berbeda untuk setiap baris data di dalam tabel. Primary Key ini bertujuan untuk memastikan bahwa setiap baris data dalam tabel dapat diidentifikasi secara unik dan dapat ditemukan dengan mudah. Misalnya, jika kita memiliki tabel "customer" yang berisi informasi tentang pelanggan sebuah toko online, Primary Key dapat diatur pada kolom "id_customer". Dengan membuat kolom ini sebagai Primary Key, setiap pelanggan yang terdaftar di dalam tabel akan memiliki nilai "id_customer" yang unik dan berbeda, sehingga setiap data pelanggan dapat diidentifikasi dan ditemukan dengan mudah.

Penggunaan Primary Key sangat penting dalam perancangan basis data karena memudahkan pengaksesan data, melakukan operasi pembaruan dan penghapusan data, serta meminimalkan terjadinya duplikasi data dalam sebuah tabel. Primary Key juga membantu memastikan integritas data, sehingga setiap baris data dalam tabel dapat diidentifikasi secara unik dan tidak tercampur dengan data lain di dalam tabel. Dengan demikian, penggunaan Primary Key sangat penting dalam membangun basis data yang efektif dan efisien.

Melihat Tabel

Untuk melihat tabel yang ada dalam sebuah database, dapat dilakukan dengan menjalankan kode tertentu di dalam bahasa pemrograman. Contohnya, jika kita menggunakan bahasa pemrograman Python dan telah terhubung ke database yang ingin dilihat tabel-tabelnya, maka dapat dilakukan dengan menjalankan syntax berikut:

```
● ● ●  
  
cursor = db.cursor()  
  
## 'DESC table_name' is used to get all columns information  
cursor.execute("DESC users")  
  
## it will print all the columns as 'tuples' in a list  
print(cursor.fetchall())
```

Computational Thinking

Python MySQL - Create Table

Melihat Tabel

```
[('id', 'int(11)', 'NO', 'PRI', None, 'auto_increment'), ('name', 'varchar(255)',  
'YES', "", None, ""), ('user_name', 'varchar(255)', 'YES', "", None, "")]
```

Output tersebut menunjukkan informasi kolom dari tabel users. Setiap baris merepresentasikan sebuah kolom, dengan informasi sebagai berikut:

- Nama kolom
- Tipe data kolom
- Apakah kolom dapat menerima nilai null atau tidak (YES/NO)
- Apakah kolom merupakan primary key (PRI) atau bukan (MUL jika kolom merupakan bagian dari indeks)
- Nilai default kolom (jika ada)
- Timestamp kapan kolom terakhir diupdate (jika kolom merupakan timestamp)

Menyisipkan Data

Seringkali dalam melakukan agregasi pada database kita membutuhkan data baru, sehingga diperlukan untuk menyisipkan satu atau lebih baris ke dalam tabel. Statement yang digunakan untuk menyisipkan data adalah `INSERT INTO table_name (column_name) VALUES (data)`. Berikut ini adalah cara bagaimana menyisipkan baris dalam tabel:

```
cursor = db.cursor()  
  
## defining the Query  
query = "INSERT INTO users (name, user_name) VALUES (%s, %s)"  
  
## storing values in a variable  
values = ("Hafeez", "hafeez")  
  
## executing the query with values  
cursor.execute(query, values)  
  
## to make final output we have to run the 'commit()' method of the database  
## object  
db.commit()  
  
print(cursor.rowcount, "record inserted")
```

Output

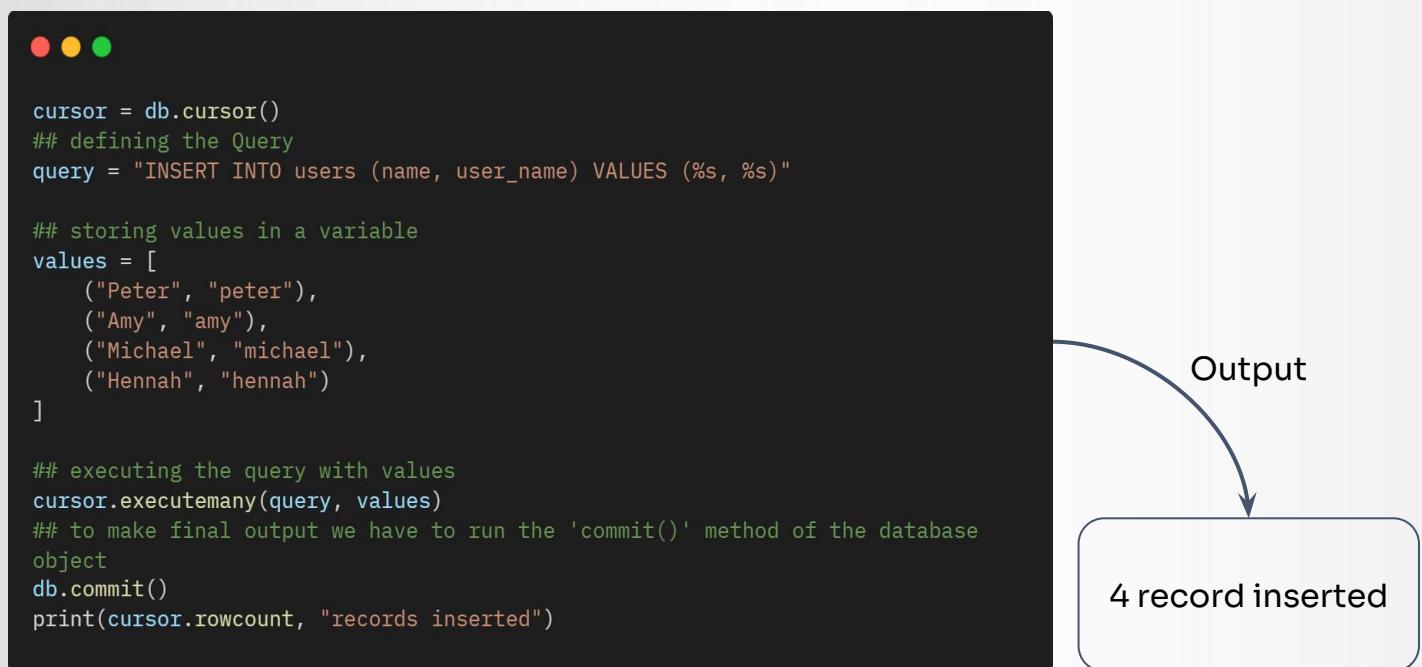
1 record inserted

Computational Thinking

Python MySQL - Create Table

Menyisipkan Data

Output ini menunjukkan bahwa satu baris data telah berhasil dimasukkan ke dalam tabel "users" menggunakan pernyataan INSERT INTO dengan nilai "Hafeez" untuk kolom "name" dan "hafeez" untuk kolom "user_name". Metode cursor.rowcount digunakan untuk mengambil jumlah baris yang terpengaruh oleh eksekusi query, yang dalam kasus ini adalah 1. Selanjutnya, metode db.commit() digunakan untuk mengkonfirmasi perubahan ke dalam database. Untuk menyisipkan beberapa baris ke dalam tabel syntax yang digunakan mirip seperti sebelumnya, namun yang membedakan adalah variable values yang akan memuat data pada table, seperti contoh dibawah ini:



```
● ● ●

cursor = db.cursor()
## defining the Query
query = "INSERT INTO users (name, user_name) VALUES (%s, %s)"

## storing values in a variable
values = [
    ("Peter", "peter"),
    ("Amy", "amy"),
    ("Michael", "michael"),
    ("Hannah", "hannah")
]

## executing the query with values
cursor.executemany(query, values)
## to make final output we have to run the 'commit()' method of the database
# object
db.commit()
print(cursor.rowcount, "records inserted")
```

Output

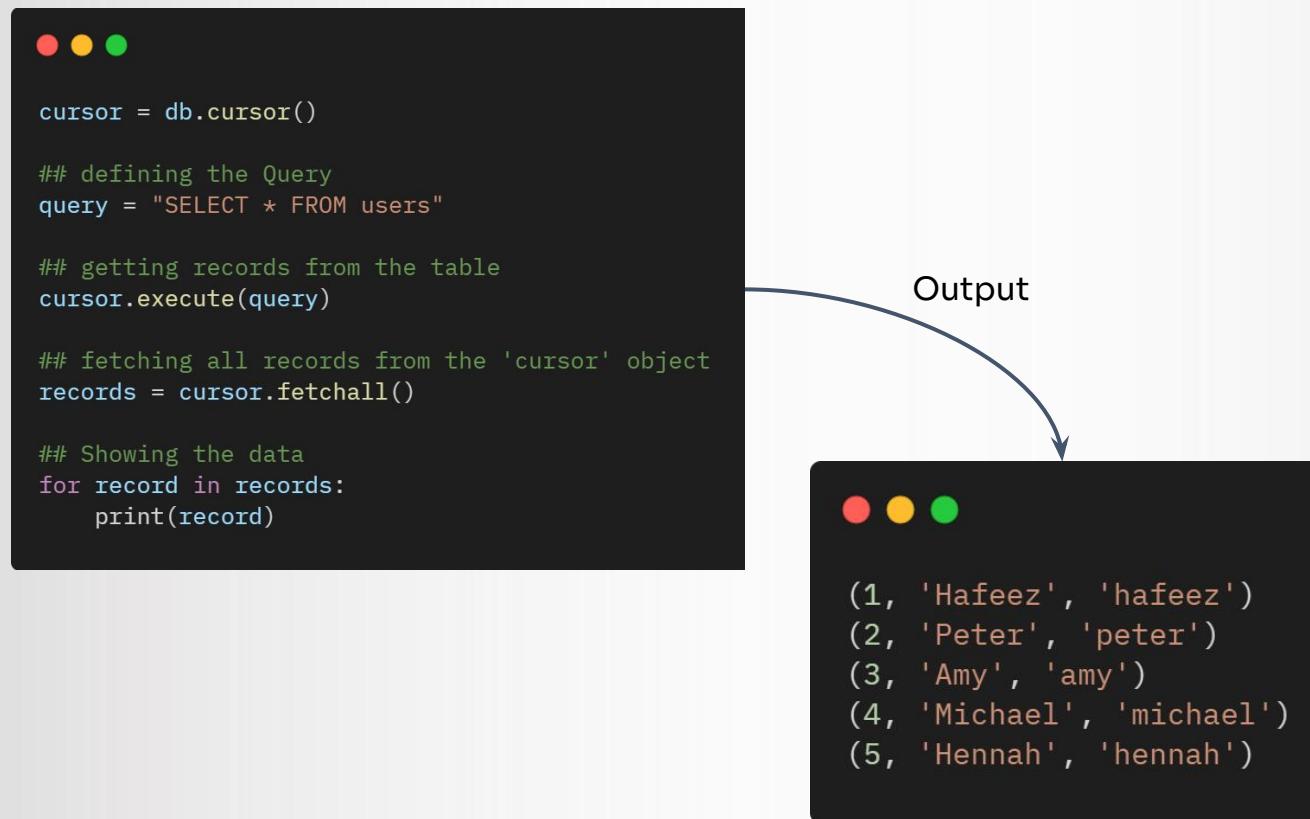
4 record inserted

Kode menyimpan data yang akan disisipkan ke dalam tabel dalam variabel values. Data tersebut ditempatkan dalam bentuk tupel atau daftar dari tupel, di mana setiap tupel berisi nilai untuk setiap kolom yang sesuai dalam tabel. Dengan demikian, kode tersebut akan menyisipkan beberapa baris data ke dalam tabel users pada database MySQL dan mencetak jumlah baris yang berhasil disisipkan ke layar.

Computational Thinking

Python MySQL - Select Data

Untuk mendapatkan data dari tabel, kita bisa menggunakan syntax `SELECT column_names FROM table_name`. Jika kita ingin mendapatkan semua record dari sebuah tabel, kita bisa menggunakan `*` sebagai pengganti nama kolom. Sebagai contoh, kita bisa mengambil semua data dari tabel `users` yang sudah kita masukkan sebelumnya.



```
● ● ●

cursor = db.cursor()

## defining the Query
query = "SELECT * FROM users"

## getting records from the table
cursor.execute(query)

## fetching all records from the 'cursor' object
records = cursor.fetchall()

## Showing the data
for record in records:
    print(record)
```

Output

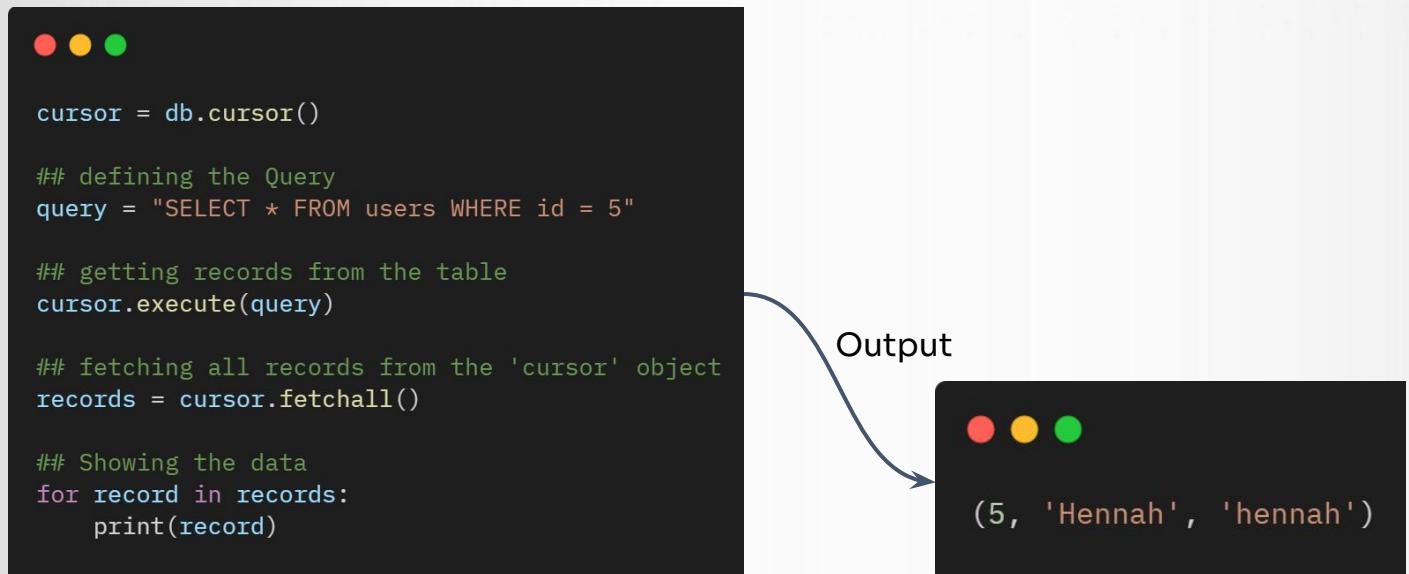
```
● ● ●
(1, 'Hafeez', 'hafeez')
(2, 'Peter', 'peter')
(3, 'Amy', 'amy')
(4, 'Michael', 'michael')
(5, 'Hennah', 'hannah')
```

Syntax ini melakukan iterasi pada setiap baris hasil query yang ditemukan (berupa tuple) menggunakan perulangan 'for', dan menampilkan data pada setiap baris dalam format yang dapat dibaca menggunakan perintah `'print(record)'`. Outputnya adalah data yang diambil dari tabel 'users' pada database 'datacamp', dimana setiap baris data ditampilkan sebagai tuple. Setiap tuple berisi nilai-nilai yang ada pada kolom-kolom dalam tabel 'users' (ID, nama depan, dan nama belakang).

Computational Thinking

Python MySQL - Where

WHERE digunakan dalam SQL untuk mengambil data dari tabel yang memenuhi kondisi tertentu. Contohnya, dalam kasus ini kita akan menggunakan WHERE untuk memilih record dengan id 5 dari tabel data.



The diagram illustrates the execution of a Python script to perform a MySQL query. On the left, a dark terminal window shows the script's code. On the right, a separate window labeled "Output" shows the resulting tuple of data.

```
cursor = db.cursor()

## defining the Query
query = "SELECT * FROM users WHERE id = 5"

## getting records from the table
cursor.execute(query)

## fetching all records from the 'cursor' object
records = cursor.fetchall()

## Showing the data
for record in records:
    print(record)
```

Output:

```
(5, 'Hennah', 'hannah')
```

`query = "SELECT * FROM users WHERE id = 5"`: Syntax ini mendefinisikan query SQL yang akan dieksekusi. Query ini akan mengambil semua kolom (*) dari tabel users di database yang memiliki nilai kolom id sama dengan 5. Output yang diberikan merupakan data yang ditemukan dalam kolom id, Hennah, dan hannah dari tabel users di database yang memiliki nilai kolom id sama dengan 5.

Python MySQL - Order By

Dalam modul pembelajaran ini, kita akan belajar tentang cara mengurutkan hasil query dalam database menggunakan perintah `ORDER BY`. `ORDER BY` adalah sebuah perintah SQL yang dapat digunakan untuk mengurutkan data dalam urutan menaik atau menurun berdasarkan kolom tertentu. Secara default, `ORDER BY` akan mengurutkan hasil dalam urutan menaik, yaitu dari nilai terkecil ke nilai terbesar (`Asc`). Namun, jika kita ingin mengurutkan hasil dalam urutan menurun, kita bisa menggunakan kata kunci `DESC` setelah kolom yang ingin diurutkan. Dengan demikian, kita memiliki fleksibilitas untuk mengatur urutan data sesuai kebutuhan kita.

Computational Thinking

Python MySQL - Order By

Berikut ini adalah contoh penggunaan ORDER BY untuk mengurutkan nilai terkecil hingga terbesar:

```
cursor = db.cursor()

## defining the Query
query = "SELECT * FROM users ORDER BY name"

## getting records from the table
cursor.execute(query)
## fetching all records from the 'cursor' object
records = cursor.fetchall()
## Showing the data
for record in records:
    print(record)
```

Output

```
(3, 'Amy', 'amy')
(1, 'Hafeez', 'hafeez')
(5, 'Hannah', 'hannah')
(4, 'Michael', 'michael')
(2, 'Peter', 'peter')
```

Dengan menggunakan syntax ini, kita akan mendapatkan data dari tabel "users" yang diurutkan berdasarkan nilai kolom "name" secara menaik (ascending), yaitu dari nilai terkecil ke nilai terbesar. Hasil data yang diambil akan diurutkan berdasarkan nilai pada kolom "name" dalam urutan alfabetis atau numerik, tergantung tipe data pada kolom tersebut dalam database.

Python MySQL - Delete

Dalam modul pembelajaran ini, kita akan membahas tentang penggunaan kata kunci **DELETE** dalam perintah SQL. **DELETE** digunakan untuk menghapus record atau data dari tabel dalam database. Untuk menghapus record tertentu, kita menggunakan pernyataan kondisi **WHERE** dalam perintah **DELETE FROM table_name**.

Jika kita tidak menentukan kondisi, maka semua catatan dalam tabel akan dihapus. Namun, kita harus berhati-hati dalam penggunaan perintah **DELETE** tanpa kondisi, karena ini dapat menghapus semua data dalam tabel tanpa tahu batasannya.

Computational Thinking

Python MySQL - Delete

Sebagai contoh, kita akan menghapus catatan dari tabel pengguna yang memiliki id 5.

Dengan menggunakan perintah SQL seperti berikut:

```
cursor = db.cursor()

## defining the Query
query = "DELETE FROM users WHERE id = 5"

## executing the query
cursor.execute(query)

## final step to tell the database
## that we have changed the table data
db.commit()
```

Maka data yang memiliki id 5 dalam tabel pengguna akan dihapus dari database. Penting untuk selalu memeriksa kondisi yang diberikan dalam perintah DELETE agar kita dapat menghapus record yang diinginkan dengan akurat dan menghindari menghapus data yang tidak perlu atau penting.

Python MySQL - Update

Dalam modul pembelajaran ini, kita akan belajar tentang penggunaan kata kunci **UPDATE** dalam perintah SQL untuk memperbarui data pada record atau baris tertentu dalam tabel. Perintah **UPDATE** digunakan untuk mengganti nilai kolom tertentu dengan nilai baru.

Dalam perintah **UPDATE**, kita harus menyebutkan nama tabel yang akan diperbarui, kemudian menggunakan kata kunci **SET** untuk menentukan kolom yang akan diubah, diikuti dengan nilai baru yang ingin diberikan pada kolom tersebut. Kita juga bisa menggunakan pernyataan kondisi **WHERE** untuk membatasi record mana yang akan diperbarui.

Computational Thinking

Python MySQL - Update

Sebagai contoh, kita akan mengganti nama pada record 1 dalam tabel dengan mengubah nama "Hafeez" menjadi "Kareem". Dengan menggunakan perintah SQL seperti berikut:

```
cursor = db.cursor()

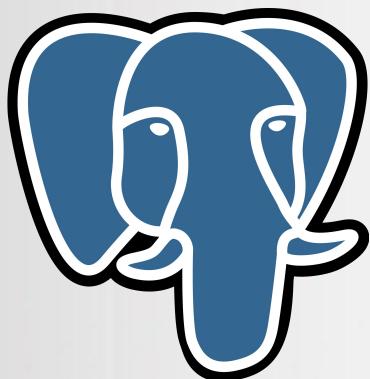
## defining the Query
query = "UPDATE users SET name = 'Kareem' WHERE id = 1"

## executing the query
cursor.execute(query)

## final step to tell the database that we have changed the table data
db.commit()
```

Maka nilai kolom "column_name" pada record dengan record_id 1 dalam tabel akan diperbarui menjadi "Kareem". Penting untuk memastikan kondisi yang diberikan dalam perintah UPDATE sesuai dengan record yang ingin diperbarui agar data diperbarui secara akurat dan sesuai dengan kebutuhan.

Python PostgreSQL - Connecting



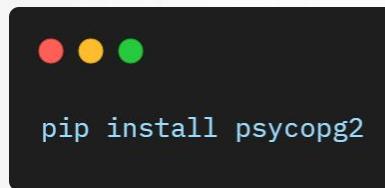
PostgreSQL, salah satu database relasional open-source yang paling populer, digunakan oleh perusahaan dari berbagai ukuran dan pengembang di seluruh dunia. Menurut DB-Engines, PostgreSQL menempati urutan keempat sebagai database paling populer di dunia, dengan tren yang terus meningkat. Tidak mengherankan, karena database PostgreSQL dapat ditemukan di balik banyak aplikasi web, seluler, dan bahkan perangkat lunak analitik.

Tidak hanya itu, PostgreSQL juga memiliki ekosistem yang sangat kaya dengan banyak alat dan ekstensi yang terintegrasi dengan baik dengan database inti. Dengan demikian, PostgreSQL menjadi pilihan yang tepat baik untuk kebutuhan database transaksional maupun analitik, serta menjadi solusi yang optimal untuk membangun database kustom sesuai kebutuhan kita.

Computational Thinking

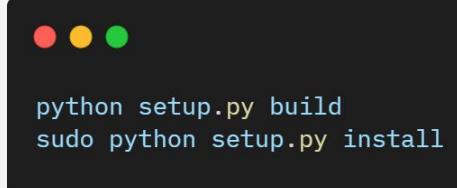
Python PostgreSQL – Connecting Connect To PostgreSQL Database Server

Untuk mempelajari lebih lanjut tentang psycopg2, salah satu package yang digunakan untuk menghubungkan Python dengan PostgreSQL, langkah pertama yang perlu dilakukan adalah mengunjungi website resmi psycopg2 di [Psycopg2](#). Setelah itu, langkah kedua adalah menggunakan baris perintah berikut dari terminal:



```
pip install psycopg2
```

Baris perintah "pip install psycopg2" adalah perintah yang digunakan untuk menginstal package psycopg2 menggunakan pip, yaitu alat manajemen package di Python. Jika telah mengunduh package ke komputer, kita dapat menggunakan setup.py sebagai berikut:



```
python setup.py build  
sudo python setup.py install
```

Baris perintah "python setup.py build" adalah perintah yang digunakan untuk membangun (build) package atau modul Python yang memiliki skrip setup.py. Skrip setup.py biasanya digunakan dalam package Python untuk mengatur proses instalasi, konfigurasi, dan distribusi package tersebut. Dengan menjalankan perintah ini, skrip setup.py akan dieksekusi, dan package atau modul Python akan dibangun atau dikompilasi sesuai dengan konfigurasi yang ditentukan dalam skrip tersebut. Sedangkan baris perintah "sudo python setup.py install" adalah perintah yang digunakan untuk menginstal package atau modul Python yang telah dibangun sebelumnya menggunakan perintah "python setup.py build". Perintah "sudo" digunakan untuk menjalankan perintah sebagai superuser atau administrator, yang mungkin diperlukan untuk mengakses folder atau direktori sistem yang memerlukan izin khusus. Setelah itu, perintah "python setup.py install" akan menginstal package atau modul Python yang telah dibangun ke dalam instalasi Python yang aktif pada sistem.

Computational Thinking

Python PostgreSQL – Create Database

Create a New Database

Untuk membuat database baru di PostgreSQL, langkah pertama adalah masuk ke server database menggunakan alat klien seperti pgAdmin atau psql. Setelah itu, langkah kedua adalah menggunakan pernyataan berikut untuk membuat database baru dengan nama "suppliers" di server database PostgreSQL.



```
CREATE DATABASE suppliers;
```

Connect to PostgreSQL Database Menggunakan psycopg2

Untuk terhubung ke database "suppliers" menggunakan modul psycopg2, kita dapat menggunakan fungsi connect() yang disediakan oleh modul tersebut. Fungsi connect() akan membuat sebuah sesi baru dengan database dan mengembalikan instance objek koneksi yang dapat digunakan untuk berinteraksi dengan database. Dengan objek koneksi ini, kita dapat membuat kursor baru yang dapat digunakan untuk mengeksekusi pernyataan SQL. Untuk memanggil fungsi connect(), kita perlu menyediakan parameter untuk koneksi ke database PostgreSQL sebagai string koneksi. Berikut adalah contoh cara memanggil fungsi connect():



```
conn = psycopg2.connect("dbname=suppliers user=postgres password=postgres")
```

Atau dapat menggunakan keyword arguments, seperti dibawah ini:



```
conn = psycopg2.connect(  
    host="localhost",  
    database="suppliers",  
    user="postgres",  
    password="Abcd1234")
```

Computational Thinking

Python PostgreSQL - Create Database

Dalam contoh di atas, kita harus mengganti "nama_database", "nama_pengguna", "kata_sandi", "alamat_host", dan "port_host" dengan nilai-nilai sesuai konfigurasi database PostgreSQL yang digunakan. Dengan demikian, kita dapat terhubung ke database "suppliers" dan mengoperasikan pernyataan SQL menggunakan objek koneksi 'conn'.

Python PostgreSQL - Create Table

Berikut adalah langkah-langkah untuk membuat tabel baru di database PostgreSQL:

1. Buat pernyataan CREATE TABLE yang akan mendefinisikan struktur tabel yang ingin dibuat.

```
CREATE TABLE nama_tabel (
    kolom1 tipe_data1,
    kolom2 tipe_data2,
    kolom3 tipe_data3
);
```

2. Sambungkan ke database PostgreSQL dengan menggunakan fungsi connect(). Fungsi connect() akan menghasilkan objek koneksi yang dapat digunakan untuk berkomunikasi dengan database.

```
import psycopg2

koneksi = psycopg2.connect(
    dbname="nama_database",
    user="nama_pengguna",
    password="kata_sandi",
    host="alamat_host",
    port="nomor_port"
)
```

3. Buat objek kursor dengan memanggil metode cursor() dari objek koneksi. Objek kursor digunakan untuk mengirim perintah SQL ke database.

Computational Thinking

Python PostgreSQL – Create Table

```
● ● ●  
kursor = koneksi.cursor()
```

4. Jalankan pernyataan CREATE TABLE dengan memanggil metode execute() dari objek kursor. Pernyataan CREATE TABLE akan dijalankan di database.

```
● ● ●  
kursor.execute("CREATE TABLE nama_tabel (kolom1 tipe_data1, kolom2 tipe_data2,  
kolom3 tipe_data3);")
```

5. Tutup komunikasi dengan server database PostgreSQL dengan memanggil metode close() dari objek kursor dan objek koneksi.

```
● ● ●  
kursor.close()  
koneksi.close()
```

Dengan langkah-langkah di atas, Anda dapat membuat tabel baru di database PostgreSQL dengan menggunakan pernyataan CREATE TABLE dan mengelola koneksi serta kursor untuk berkomunikasi dengan database.

Computational Thinking

Python PostgreSQL - Create Table

Berikut ini adalah implementasi untuk membuat table pada PostgreSQL dengan Python.

```
#!/usr/bin/python

import psycopg2
from config import config

def create_tables():
    """ create tables in the PostgreSQL database"""
    commands = (
        """
        CREATE TABLE vendors (
            vendor_id SERIAL PRIMARY KEY,
            vendor_name VARCHAR(255) NOT NULL
        )
        """,
        """
        CREATE TABLE parts (
            part_id SERIAL PRIMARY KEY,
            part_name VARCHAR(255) NOT NULL
        )
        """,
        """
        CREATE TABLE part_drawings (
            part_id INTEGER PRIMARY KEY,
            file_extension VARCHAR(5) NOT NULL,
            drawing_data BYTEA NOT NULL,
            FOREIGN KEY (part_id)
                REFERENCES parts (part_id)
            ON UPDATE CASCADE ON DELETE CASCADE
        )
        """,
        """
        CREATE TABLE vendor_parts (
            vendor_id INTEGER NOT NULL,
            part_id INTEGER NOT NULL,
            PRIMARY KEY (vendor_id , part_id),
            FOREIGN KEY (vendor_id)
                REFERENCES vendors (vendor_id)
            ON UPDATE CASCADE ON DELETE CASCADE,
            FOREIGN KEY (part_id)
                REFERENCES parts (part_id)
            ON UPDATE CASCADE ON DELETE CASCADE
        )
        """
    )
    conn = None
```

Computational Thinking

Python PostgreSQL - Create Table

```
try:
    # read the connection parameters
    params = config()
    # connect to the PostgreSQL server
    conn = psycopg2.connect(**params)
    cur = conn.cursor()
    # create table one by one
    for command in commands:
        cur.execute(command)
    # close communication with the PostgreSQL database server
    cur.close()
    # commit the changes
    conn.commit()
except (Exception, psycopg2.DatabaseError) as error:
    print(error)
finally:
    if conn is not None:
        conn.close()

if __name__ == '__main__':
    create_tables()
```

Berikut ini adalah penjelasan dari syntax diatas, agar lebih mudah untuk dipahami:

- Pada bagian pertama, library psycopg2 dan config diimpor ke dalam script Python. Library psycopg2 digunakan untuk berinteraksi dengan database PostgreSQL, sedangkan library config digunakan untuk membaca konfigurasi koneksi ke database dari file eksternal.
- Mendefinisikan Fungsi `create_tables()`: Fungsi ini digunakan untuk membuat tabel-tabel di database PostgreSQL. Tabel-tabel yang akan dibuat didefinisikan dalam pernyataan SQL yang terdapat pada variabel 'commands'. Setiap pernyataan SQL untuk membuat tabel disimpan dalam elemen terpisah dalam tuple.
- Fungsi `config()` dari library config digunakan untuk membaca konfigurasi koneksi ke database dari file eksternal. Konfigurasi ini biasanya berisi informasi seperti nama host, port, nama database, nama pengguna, dan kata sandi yang diperlukan untuk terhubung ke database PostgreSQL.

Computational Thinking

Python PostgreSQL - Create Table

- Membuka Koneksi ke Database: Setelah membaca konfigurasi koneksi, fungsi `psycopg2.connect()` digunakan untuk membuka koneksi ke database PostgreSQL menggunakan nilai-nilai konfigurasi yang telah dibaca sebelumnya. Objek koneksi disimpan dalam variabel '`conn`'.
- Membuka Kursor: Setelah terhubung ke database, objek kursor (`cursor`) digunakan untuk mengirim pernyataan SQL ke database. Objek kursor diperoleh dari objek koneksi dengan memanggil metode `cursor()` pada objek koneksi. Objek kursor disimpan dalam variabel '`cur`'.
- Mengeksekusi Pernyataan SQL: Melalui loop `for`, pernyataan SQL untuk membuat tabel-tabel dijalankan satu per satu menggunakan metode `execute()` pada objek kursor '`cur`'. Pernyataan SQL yang dijalankan diambil dari variabel '`commands`' yang berisi tuple dengan pernyataan SQL untuk setiap tabel.
- Menutup Kursor dan Mengcommit Perubahan: Setelah pernyataan SQL untuk membuat tabel-tabel telah dijalankan, kursor ditutup dengan memanggil metode `close()` pada objek kursor '`cur`'. Selanjutnya, perubahan yang telah dilakukan di-commit ke database dengan memanggil metode `commit()` pada objek koneksi '`conn`'.
- Menangani Error dan Menutup Koneksi: Jika terjadi error selama proses eksekusi pernyataan SQL atau koneksi ke database, error akan ditangani dengan menampilkan pesan error menggunakan pernyataan `print()`. Pada akhirnya, koneksi ke database ditutup dengan memanggil metode `close()` pada objek koneksi '`conn`'.
- Eksekusi Fungsi `create_tables()`: Pada bagian terakhir, fungsi `create_tables()` dijalankan dengan memanggilnya pada blok `if name == 'main':`, sehingga tabel-tabel akan dibuat di database PostgreSQL ketika script Python dieksekusi sebagai program mandiri.

Computational Thinking

Python PostgreSQL - Insert Data to Table

Berikut adalah langkah-langkah yang harus diikuti untuk menyisipkan baris baru ke dalam tabel PostgreSQL menggunakan Python:

```
#!/usr/bin/python

import psycopg2
from config import config

def insert_vendor(vendor_name):
    """ insert a new vendor into the vendors table """
    sql = """INSERT INTO vendors(vendor_name)
              VALUES(%s) RETURNING vendor_id;"""
    conn = None
    vendor_id = None
    try:
        # read database configuration
        params = config()
        # connect to the PostgreSQL database
        conn = psycopg2.connect(**params)
        # create a new cursor
        cur = conn.cursor()
        # execute the INSERT statement
        cur.execute(sql, (vendor_name,))
        # get the generated id back
        vendor_id = cur.fetchone()[0]
        # commit the changes to the database
        conn.commit()
        # close communication with the database
        cur.close()
    except (Exception, psycopg2.DatabaseError) as error:
        print(error)
    finally:
        if conn is not None:
            conn.close()

    return vendor_id
```

Computational Thinking

Machine Learning Myfeature engineering

Pengenalan Machine Learning

Machine Learning adalah suatu cabang ilmu komputer yang memungkinkan komputer untuk belajar dari data dan melakukan tugas-tugas tertentu secara otomatis. Tujuan utama dari Machine Learning adalah untuk membangun model atau sistem yang dapat melakukan tugas tertentu seperti klasifikasi, regresi, klastering, atau prediksi.

Feature Engineering adalah suatu proses di mana kita memilih, mengekstrak, atau mengubah fitur dari data mentah (raw data) yang akan digunakan sebagai input pada model Machine Learning. Fitur adalah karakteristik atau atribut yang mendefinisikan suatu objek atau entitas dalam dataset. Contohnya dalam data karyawan, fitur-fitur dapat berupa nama, usia, jenis kelamin, pendidikan, gaji, dan sebagainya.

Feature Engineering merupakan tahap yang sangat penting dalam Machine Learning, karena kualitas dan kuantitas fitur yang digunakan akan sangat mempengaruhi performa dan akurasi model. Pemilihan fitur yang buruk dapat menyebabkan model menjadi overfitting atau underfitting, yang dapat merugikan dalam pengambilan keputusan.

Terdapat beberapa teknik Feature Engineering yang dapat dilakukan dalam Machine Learning, antara lain:

1. Pemilihan Fitur (Feature Selection)

Teknik ini dilakukan dengan memilih fitur yang paling relevan dan memiliki pengaruh yang signifikan dalam model. Terdapat beberapa metode untuk pemilihan fitur, seperti Filter, Wrapper, dan Embedded. Metode Filter melakukan seleksi fitur berdasarkan korelasi dengan label, sedangkan metode Wrapper melakukan seleksi fitur dengan memilih subset fitur yang memberikan performa terbaik pada model. Metode Embedded melakukan seleksi fitur dengan menggabungkan proses pelatihan model dan pemilihan fitur.

Computational Thinking

2. Ekstraksi Fitur (Feature Extraction)

Teknik ini dilakukan dengan mengubah fitur yang ada menjadi fitur baru yang lebih informatif. Contoh teknik ekstraksi fitur adalah Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), dan Independent Component Analysis (ICA). PCA mengubah fitur menjadi fitur baru yang memaksimalkan varian data, sedangkan LDA mengubah fitur menjadi fitur baru yang memaksimalkan pemisahan antara kelas. ICA mengubah fitur menjadi fitur baru yang independen satu sama lain.

3. Transformasi Fitur (Feature Transformation)

Teknik ini dilakukan dengan melakukan transformasi pada nilai fitur, sehingga nilai tersebut dapat lebih representatif atau normalisasi. Contoh teknik transformasi fitur adalah Min-Max Scaling, Z-Score Scaling, dan Log Transformation. Min-Max Scaling mengubah rentang nilai fitur menjadi 0 hingga 1, sedangkan Z-Score Scaling mengubah nilai fitur menjadi z-score dengan rata-rata 0 dan standar deviasi 1. Log Transformation mengubah nilai fitur menjadi skala logaritmik.

4. Pembuatan Fitur Baru (Feature Creation)

Teknik ini dilakukan dengan membuat fitur baru yang tidak tersedia pada data mentah, tetapi dapat memberikan informasi tambahan yang relevan. Contoh pembuatan fitur baru adalah memecah tanggal menjadi bulan, tahun, dan hari, atau menggabungkan beberapa fitur menjadi satu fitur baru.

Kesimpulan

Feature Engineering adalah suatu tahap yang sangat penting dalam Machine Learning.

Computational Thinking

Machine Learning Postgrefeature engineering

Setelah proses Feature Engineering dilakukan, langkah selanjutnya adalah melakukan proses Postgrefeature Engineering untuk mengoptimalkan kualitas dan performa model. Postgrefeature Engineering adalah suatu proses di mana kita melakukan evaluasi, pengujian, dan perbaikan pada fitur yang telah diproses dalam tahap Feature Engineering.

Terdapat beberapa teknik Postgrefeature Engineering yang dapat dilakukan dalam Machine Learning, antara lain:

1. Evaluasi Model (Model Evaluation)

Teknik ini dilakukan dengan mengukur performa model pada data testing atau validasi. Terdapat beberapa metode untuk evaluasi model, seperti Akurasi, Precision, Recall, dan F1-Score. Akurasi mengukur seberapa tepat model dalam melakukan klasifikasi, sedangkan Precision mengukur seberapa akurat model dalam memprediksi kelas positif. Recall mengukur seberapa banyak data kelas positif yang terdeteksi oleh model, sedangkan F1-Score mengukur kebaikan model secara keseluruhan.

2. Pengoptimalan Model (Model Optimization)

Teknik ini dilakukan dengan melakukan tuning pada parameter model untuk meningkatkan performa dan akurasi model. Terdapat beberapa metode untuk pengoptimalan model, seperti Grid Search dan Randomized Search. Grid Search melakukan pencarian parameter pada suatu rentang nilai yang telah ditentukan, sedangkan Randomized Search melakukan pencarian parameter secara acak dalam suatu rentang nilai.

Computational Thinking

3. Pemilihan Model (Model Selection)

Teknik ini dilakukan dengan memilih model yang paling sesuai untuk suatu tugas Machine Learning. Terdapat beberapa model Machine Learning yang dapat dipilih, seperti Regresi Linier, Regresi Logistik, Naive Bayes, K-Nearest Neighbor, Decision Tree, dan Random Forest. Pemilihan model dilakukan berdasarkan karakteristik data, tugas Machine Learning yang diinginkan, serta performa dan akurasi model yang dihasilkan.

4. Ensemble Learning

Teknik ini dilakukan dengan menggabungkan beberapa model Machine Learning yang berbeda untuk meningkatkan performa dan akurasi model. Terdapat beberapa metode untuk ensemble learning, seperti Voting, Bagging, dan Boosting. Voting menggabungkan beberapa model dengan cara voting, sedangkan Bagging menggabungkan beberapa model dengan bootstrapping. Boosting menggabungkan beberapa model secara bertahap dengan memperbaiki kesalahan model sebelumnya.

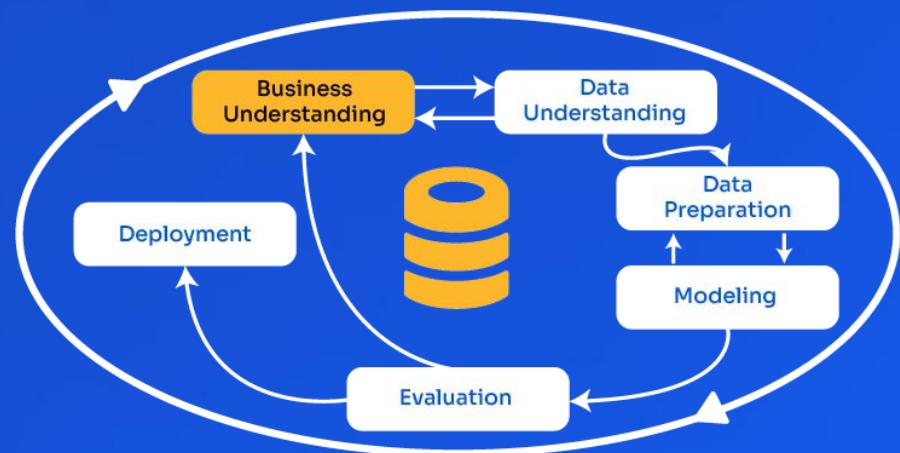
Kesimpulan

Postgrefeature Engineering adalah suatu tahap yang sangat penting dalam Machine Learning. Dengan melakukan evaluasi, pengoptimalan, pemilihan model, dan ensemble learning, performa dan akurasi model dapat ditingkatkan secara signifikan.

Statistical Thinking

Understanding Data and Business

Cross Industry Standard Process for Data Mining



Ketika berbicara tentang Data Science, banyak orang merasa sulit untuk memahami keseluruhan prosesnya. Mulai dari pengumpulan data, analisis, hingga presentasi hasilnya, semuanya terlihat rumit dan sulit dipahami. Namun, sebenarnya, proses data science dapat dipecah menjadi beberapa tahapan yang mudah dipahami:

Tahapan pertama: "Business understanding", di mana ahli data bekerja sama dengan pemangku kepentingan bisnis untuk memahami tujuan dan kebutuhan bisnis.

Tahap Kedua: "Data understanding", di mana ahli data mempelajari data yang tersedia untuk memahami kualitas dan karakteristiknya. Kemudian, dilanjutkan dengan tahap "data preparation", di mana ahli data membersihkan dan mempersiapkan data untuk analisis.

Tahapan Ketiga: ahli data mulai melakukan "modeling" atau pembuatan model analisis yang sesuai dengan tujuan bisnis.

Tahap Keempat: "Evaluation", di mana ahli data mengevaluasi model yang dibuat dan memastikan hasil analisisnya sesuai dengan tujuan bisnis.

Tahap Kelima: hasil analisis akan "dideploy" atau disajikan secara efektif kepada pemangku kepentingan bisnis.

Dengan memahami proses ini, maka akan lebih mudah memahami bagaimana data science bekerja dan memberikan manfaat bagi bisnis atau organisasi.

Statistical Thinking

Understanding Data and Bisnis

1. Business Understanding

- ❖ Fase *business understanding* berfokus pada pemahaman tujuan dan persyaratan proyek. Selain itu tugas lain dalam fase ini adalah aktivitas manajemen proyek dasar yang bersifat universal untuk sebagian besar proyek
- ❖ Tentukan tujuan bisnis dengan memahami apa yang diinginkan pelanggan secara menyeluruh dari perspektif bisnis (CRISP-DM Guide), dan tetapkan kriteria keberhasilan bisnis.
- ❖ Mengidentifikasi ketersediaan sumber daya yang dibutuhkan, persyaratan proyek yang harus dipenuhi, mengevaluasi risiko dan kemungkinan situasi darurat, serta melakukan analisis biaya dan manfaat proyek.
- ❖ Tentukan tujuan *Data Science*: Selain menentukan tujuan bisnis, perlu juga menentukan indikator keberhasilan proyek dari sudut pandang *Data Science*.
- ❖ Menghasilkan rencana proyek: Memilih teknologi dan alat yang sesuai serta merencanakan setiap tahapan proyek secara rinci sangat penting. Terkadang banyak orang yang tergesa-gesa melewati tahap *Business Understanding*, padahal tahap ini adalah fondasi dari seluruh tahapan *Data Science*.



Statistical Thinking

Understanding Data and Bisnis

2. Data Understanding



- ❖ Perolehlah data yang dibutuhkan dan apabila perlu, silakan masukkan ke dalam alat analisis yang Anda gunakan
- ❖ Penjelasan Data: Periksa dan dokumentasikan sifat-sifat permukaan data seperti format, jumlah catatan, dan identitas.
- ❖ *Explore* data: Carilah informasi yang lebih detail dari data. Lakukan pengajuan pertanyaan, gambarkan dalam bentuk visual, dan temukan kaitan antara data.
- ❖ Verifikasi kualitas data: Tingkat kebersihan data seperti apa? Catatlah semua masalah kualitas data yang teridentifikasi.

3. Data Preparation



- ❖ Pilih data: Tentukan set data yang akan digunakan dan catat alasan mengapa data tersebut dimasukkan atau dikecualikan.
- ❖ Bersihkan data: Seringkali ini memakan waktu yang cukup lama. Tugasnya adalah memperbaiki, menggabungkan, atau menghapus nilai yang salah dari data.
- ❖ *Construct Data*: Buat atribut baru yang dapat membantu analisis. Contohnya, buatlah indeks massa tubuh dari data tinggi dan berat badan seseorang.
- ❖ Gabungkan data: Buat satu set data baru dengan menggabungkan data dari beberapa sumber.
- ❖ Format Data: Ubah format data sesuai kebutuhan. Contohnya, konversi nilai string yang berisi angka menjadi nilai numerik agar dapat dilakukan operasi matematika.

Statistical Thinking

Understanding Data and Bisnis

4. Modeling



Langkah pertama dalam pemodelan data adalah mengurangi dimensi data. Tidak semua fitur atau nilai relevan untuk memprediksi model. Contoh pemodelan antara lain klasifikasi email sebagai "Inbox" atau "Spam" menggunakan regresi logistik, atau perkiraan nilai menggunakan regresi linier. Pemodelan juga dapat digunakan untuk mengelompokkan data dan memahami logika di balik suatu cluster.

Sebuah model bermanfaat jika dapat diakses oleh pengguna. Kompleksitas tahap ini bervariasi. Tahap terakhir terdiri dari empat tugas:

- ❖ *Plan Development:* Buat dan catat rencana untuk implementasi model.
- ❖ *Plan monitoring and maintenance:* Buat rencana yang menyeluruh untuk memantau dan menjaga model agar tidak mengalami masalah selama fase operasional atau pasca proyek.
- ❖ *Produce final report:* Tim proyek membuat ringkasan proyek yang mungkin meliputi presentasi akhir hasil dari Data Science.
- ❖ *Review project:* Lakukan retrospeksi proyek tentang apa yang berjalan dengan baik, apa yang bisa lebih baik, dan bagaimana meningkatkannya di masa depan.

Statistical Thinking

Understanding Data and Bisnis

5. Evaluation



Pada tahap Evaluation, dilakukan penilaian terhadap kualitas dan efektivitas model yang telah dibuat, dengan mempertimbangkan berbagai faktor seperti akurasi, keandalan, dan performa secara keseluruhan. Tahap ini penting untuk menentukan apakah model yang dibuat dapat diandalkan dan memenuhi kebutuhan bisnis atau tidak.

- ❖ Evaluasi Hasil: Dalam evaluasi model, perlu dipertimbangkan apakah model memenuhi kriteria keberhasilan bisnis. Oleh karena itu, perlu ditentukan kriteria apa yang harus dipenuhi oleh model agar dapat dianggap berhasil dalam konteks bisnis yang sedang dikerjakan. Setelah kriteria tersebut ditentukan, perlu diputuskan apakah model yang telah dibangun memenuhi kriteria tersebut atau tidak.
- ❖ Proses Peninjauan: Lakukan peninjauan terhadap pekerjaan yang telah dilakukan, pastikan tidak ada yang terlewat atau dilakukan dengan tidak benar. Setelah itu, buatlah ringkasan temuan dan lakukan perbaikan jika diperlukan.
- ❖ Langkah selanjutnya: Berdasarkan evaluasi hasil pekerjaan yang telah dilakukan, tentukan keputusan apakah akan melanjutkan penerapan, melakukan iterasi lanjutan, atau memulai proyek baru.

Anda mungkin perlu melanjutkan pekerjaan organisasi Anda setelah proyek selesai. Meskipun kerangka kerja proyek CRISP-DM tidak memberikan panduan untuk tindakan pasca-proyek, namun pemantauan dan penyetelan model yang berkala diperlukan jika Anda berencana untuk memproduksi model.

Statistical Thinking

Dasar-Dasar Statistik

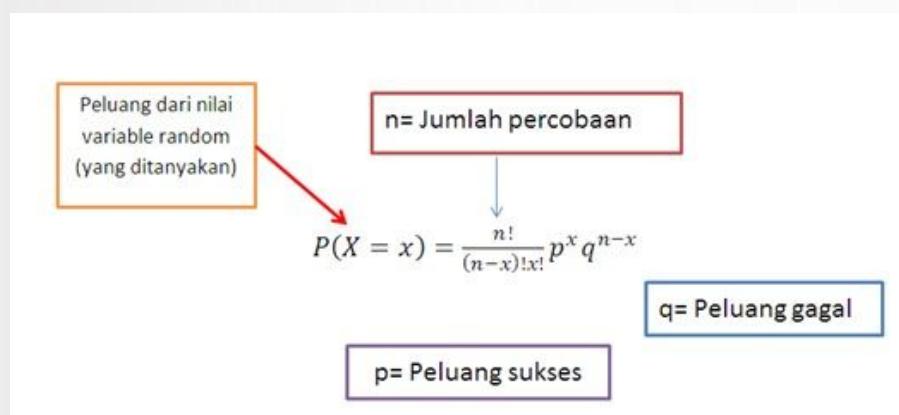
1. Binomial

Kata "BI" dalam kata BINOMIAL merujuk pada fakta bahwa hanya ada dua kemungkinan hasil yang muncul setiap kali percobaan atau kesempatan dilakukan. Namun, kapan sebaiknya kita menggunakan persamaan distribusi peluang Binomial untuk menghitung peluang suatu kejadian? Jawabannya adalah jika kejadian tersebut memenuhi syarat-syarat berikut:

1. Percobaan dilakukan sebanyak n kali.
2. Setiap kali percobaan memiliki dua kemungkinan hasil.
3. Kemungkinan hasil dari setiap percobaan adalah sama.
4. Hasil dari satu percobaan tidak mempengaruhi hasil percobaan yang lain (saling independen).

Jika kejadian memenuhi syarat-syarat tersebut, maka kita dapat menggunakan persamaan peluang dari distribusi binomial untuk menghitung peluang terjadinya kejadian tersebut.

Persamaan untuk menghitung peluang ini adalah:



Statistical Thinking

Dasar-Dasar Statistik



Contoh Soal

Kita memiliki sebuah koin yang mempunyai dua sisi, depan dan belakang. Kemudian kita melakukan pengundian sebanyak 10 kali. Pada setiap pengundian, hasilnya hanya bisa dua kemungkinan yaitu sisi depan atau sisi belakang. Untuk mencari peluang kemunculan sisi depan sebanyak dua kali dari sepuluh kali percobaan, kita dapat menggunakan persamaan distribusi peluang Binomial karena kejadian ini memenuhi syarat kejadian binomial. Dari sepuluh kali percobaan, berapa peluang sisi depan muncul sebanyak dua kali?

Diketahui:

- Jumlah percobaan = $n = 10$.
- Peluang sukses = peluang munculnya sisi depan dalam setiap percobaan = $p = 0.5$.
- Peluang gagal = peluang tidak munculnya sisi depan dalam setiap percobaan = $q = 1-p = 0.5$.

Ditanyakan:

Dari sepuluh kali percobaan, berapa peluang sisi depan muncul sebanyak dua kali? Atau $P(X = 2) \rightarrow$ Yang ditanyakan adalah peluang munculnya sisi depan maka kejadian yang dianggap sukses adalah jika sisi depan muncul ketika diundi.

Jawaban:

$$P(X = 2) = \frac{10!}{(10 - 2)! 2!} (0.5)^2 (1 - 0.5)^{10-2} = 0.0439$$

Referensi: [DISTRIBUSI PELUANG BINOMIAL](#)

Statistical Thinking

Dasar-Dasar Statistik

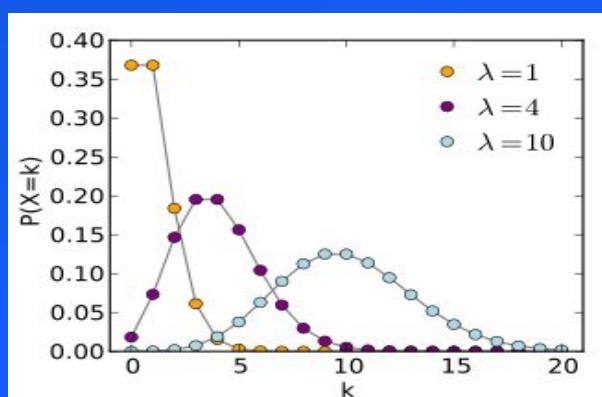
2. Poisson

Distribusi Poisson adalah suatu cara statistik yang digunakan untuk menghitung probabilitas terjadinya suatu peristiwa langka dalam suatu periode waktu atau daerah tertentu. Misalnya, kita dapat menggunakan distribusi Poisson untuk menghitung kemungkinan terjadinya kecelakaan lalu lintas dalam waktu satu hari atau kemungkinan terjadinya serangan jantung dalam suatu wilayah geografis dalam satu tahun.

Dalam distribusi Poisson, kejadianya dianggap langka, yang artinya kemungkinannya kecil, dan hasil pengamatannya berupa variabel diskret. Variabel-variabel tersebut biasanya saling independen dan tidak dipengaruhi oleh faktor-faktor lainnya. Selain itu, periode waktu atau daerah tertentu dapat berupa periode waktu atau daerah yang singkat atau panjang, tergantung pada tujuan pengamatan yang ingin dicapai. Contohnya, dapat menggunakan distribusi Poisson untuk menghitung kemungkinan terjadinya patah tulang dalam waktu sebulan atau kemungkinan terjadinya kebakaran dalam suatu gedung selama setahun.

Distribusi Poisson memiliki karakteristik sebagai berikut:

- Jumlah percobaan yang terjadi dalam suatu waktu atau daerah tertentu tidak tergantung pada jumlah percobaan di waktu atau daerah yang berbeda.
- Peluang terjadinya satu kejadian selama waktu atau daerah yang pendek sebanding dengan waktu atau daerah tersebut, dan tidak tergantung pada jumlah kejadian di luar waktu atau daerah tersebut.
- Peluang terjadinya lebih dari satu kejadian dalam waktu atau daerah yang pendek diabaikan.



Referensi: [Distribusi Poisson dan Hipergeometrik](#)

Statistical Thinking

Dasar-Dasar Statistik

3. Variabel Random Poisson



Variabel random Poisson adalah variabel acak yang digunakan dalam distribusi Poisson. Variabel ini digunakan untuk mengukur jumlah kejadian yang terjadi dalam suatu waktu atau daerah tertentu. Contohnya, dapat digunakan untuk menghitung jumlah mobil yang melewati jalan tol dalam waktu tertentu atau jumlah kasus flu yang terjadi dalam satu minggu di suatu wilayah.

Penggunaan variabel random Poisson membantu menghitung probabilitas kejadian pada waktu atau daerah tertentu, contohnya, ketika rata-rata jumlah mobil yang melintasi jalan tol dalam satu jam adalah 50.

Dengan variabel random Poisson probabilitas terjadinya lebih dari 60 mobil atau kurang dari 40 mobil dalam satu jam dapat dihitung. Namun, perlu dicatat bahwa variabel random Poisson hanya cocok untuk kejadian langka yang tidak berkorelasi dengan kejadian lain. Jika kejadian tersebut terkorelasi atau dipengaruhi oleh faktor-faktor lain, metode statistik yang berbeda harus digunakan.

Statistical Thinking

Dasar-Dasar Statistik

Sifat-sifat dari Proses Poisson antara lain:

- Tidak memiliki memori atau ingatan, artinya jumlah outcome dalam satu interval waktu atau daerah tidak bergantung pada jumlah outcome pada waktu atau daerah yang lain.
- Probabilitas terjadinya satu outcome dalam interval waktu atau daerah yang sangat pendek sebanding dengan lama waktu interval waktu atau luas daerahnya, dan tidak bergantung pada kejadian atau outcome di luar interval ini.
- Probabilitas terjadinya lebih dari satu outcome dalam interval waktu yang sangat pendek sangat kecil atau bisa diabaikan.

$$p(x; \lambda t) = \frac{e^{-\lambda t} (\lambda t)^x}{x!} \quad \Rightarrow \quad p(x; \mu) = \frac{e^{-\mu} (\mu)^x}{x!}$$


Variabel random Poisson X menunjukkan banyaknya outcome selama suatu percobaan, dengan rata-rata banyak outcome yang dinyatakan sebagai λt , di mana t adalah lama interval dan λ adalah laju terjadinya outcome. Distribusi probabilitas dari variabel random Poisson X yang menunjukkan banyaknya outcome dalam interval waktu atau daerah tertentu dengan laju terjadinya outcome λ , dapat diberikan (tanpa perlu diturunkan).

Statistical Thinking

Dasar-Dasar Statistik

4. Hipergeometrik

Distribusi hipergeometrik digunakan ketika Anda ingin mengambil sampel acak dari populasi tanpa mengembalikan sampel yang diambil sebelumnya. Untuk menggunakan distribusi ini, Anda perlu mengetahui informasi tentang susunan populasi agar dapat menentukan kembali probabilitas keberhasilan dalam setiap percobaan, karena probabilitas dapat berubah. Karakteristik distribusi hipergeometrik meliputi: distribusi diskrit dimana setiap hasil terdiri dari keberhasilan atau kegagalan, pengambilan sampel dilakukan tanpa pengembalian, populasi yang terbatas dan diketahui, serta jumlah keberhasilan dalam populasi (k) diketahui.



Distribusi Hipergeometrik mirip dengan distribusi binomial dikarenakan keduanya digunakan untuk menghitung probabilitas sejumlah tertentu percobaan yang masuk dalam kategori tertentu. Namun, ada beberapa perbedaan antara keduanya, salah satunya adalah dalam distribusi binomial, percobaan harus independen satu sama lain. Selain itu, sampling dalam binomial dilakukan dengan mengembalikan hasil percobaan sebelumnya, sedangkan dalam distribusi hipergeometrik, tidak perlu mengembalikan hasil percobaan yang sudah keluar.

Statistical Thinking

Dasar-Dasar Statistik

Distribusi Hipergeometrik adalah sebuah model matematis yang digunakan untuk menghitung probabilitas banyaknya hasil yang berhasil atau sukses dalam sebuah sampel acak tanpa pengembalian sebanyak n , yang diambil dari sebuah populasi sebanyak N . Populasi ini terdiri dari dua kelas, yaitu kelas sukses yang berjumlah k dan kelas gagal yang berjumlah $N-k$. Variabel acak X digunakan untuk menyatakan banyaknya hasil yang berhasil dalam sampel sebanyak n . Sehingga, distribusi hipergeometrik dapat memberikan estimasi probabilitas jumlah keberhasilan dalam sampel acak yang diambil dari populasi dengan jumlah sukses dan gagal yang diketahui.

$$h(x; N, n, k) = \frac{\binom{k}{x} \binom{N-k}{n-x}}{\binom{N}{n}}$$

Suku pembagi (denominator) dalam rumus distribusi hipergeometrik menghitung jumlah kombinasi yang terjadi ketika kita mengambil n objek dari N objek yang ada. Faktor pertama dari suku terbagi (numerator) dalam rumus tersebut menghitung jumlah kombinasi dari objek-objek yang termasuk dalam kategori "sukses", dengan asumsi bahwa kita mengambil sebanyak x buah setiap kali.

Statistical Thinking

Dasar-Dasar Statistik

Contoh:

Produsen mobil menjual suku cadang dalam paket berisi 10 buah. Paket dianggap “dapat diterima” jika tidak lebih dari 1 suku cadang cacat dalam setiap paket. Untuk memeriksa kualitas, beberapa paket dipilih secara acak dan diambil sampel 3 suku cadang dari setiap paket yang dipilih. Jika dari sampel tersebut tidak ada suku cadang cacat, maka paket dianggap baik. Jika kita berasumsi bahwa ada 2 paket yang tidak dapat diterima (masing-masing mengandung lebih dari 1 suku cadang cacat), maka kita akan mencoba memeriksa 3 suku cadang dari setiap paket. Namun, pada kenyataannya, ketika sampel 3 suku cadang diambil dari masing-masing paket, tidak ada satu pun suku cadang yang cacat. Sehingga, kita keliru dalam mengambil kesimpulan bahwa kedua paket tersebut tidak dapat diterima. Bagaimana probabilitas kesalahan ini terjadi?

Jawab:

Kita ingin mengetahui probabilitas teknik sampling kita dapat menemukan lot yang tidak bisa diterima karena 2 dari 10 suku cadang di dalamnya cacat. Untuk menghitungnya, kita menggunakan variabel X yang menyatakan banyaknya suku cadang cacat yang terambil dari sampel sebanyak 3 buah. Jumlah yg cacat di paket $k=2$, yg terambil tidak ada, $X=0$. Isi satu paket $N=10$, jadi yg baik $N-k=10-2=8$. Dari paket diambil $n=3$ sampel.

Banyaknya kombinasi bahwa dari $k=2$ cacat di paket tidak terambil sama sekali ($x=0$) adalah $C_2^0 = 2!/(0!2!) = 1$. Dan kombinasi dari 8 yg cacat diambil 3 buah ada sebanyak $C_8^3 = 8!/(3!5!) = 8 \times 7 \times 6 / 6 = 56$. Sedangkan kalau dari 10 diambil 3 buah item, banyak kombinasi item yg mungkin adalah $C_{10}^3 = 10!/(7!3!) = 10 \times 9 \times 8 / 6 = 120$. Jadi probabilitas bahwa yg terambil mengandung 3 buah item dan tak satupun cacat adalah:

$$h(x=0; N=10, n=3, k=2) = \frac{\binom{2}{0} \binom{8}{3}}{\binom{10}{3}} = \frac{1 * 56}{120} = 0.467 = 47\%$$

Statistical Thinking

Dasar-Dasar Statistik

5. Bayes

Statistika Bayes adalah sebuah teori dalam bidang statistika yang didasarkan pada interpretasi Bayes tentang probabilitas. Menurut interpretasi Bayes, probabilitas dapat diartikan sebagai tingkat kepercayaan pada suatu peristiwa. Tingkat kepercayaan tersebut dapat bergantung pada pengetahuan sebelumnya tentang peristiwa tersebut, seperti hasil percobaan sebelumnya, atau pada keyakinan pribadi tentang peristiwa tersebut.

Hal ini berbeda dengan sejumlah interpretasi probabilitas lainnya, seperti interpretasi frekuensi yang melihat probabilitas sebagai batas frekuensi relatif dari suatu peristiwa setelah melakukan percobaan dalam jumlah yang besar. Dengan menggunakan Statistika Bayes, kita dapat menghitung probabilitas suatu peristiwa dengan mempertimbangkan informasi sebelumnya dan memperbarui probabilitas tersebut ketika ada informasi baru. Ini memungkinkan kita untuk membuat keputusan yang lebih baik dan membuat prediksi yang lebih akurat dalam berbagai bidang, termasuk ilmu pengetahuan, bisnis, dan teknologi.

- Metode statistika Bayes menggunakan teorema Bayes untuk menghitung dan memperbarui probabilitas setelah mendapatkan data baru.
- Teorema Bayes menyediakan cara untuk menghitung probabilitas bersyarat pada suatu peristiwa berdasarkan informasi sebelumnya tentang peristiwa tersebut, atau kondisi yang terkait dengan peristiwa tersebut.
- Contohnya, dalam inferensi Bayes, teorema Bayes digunakan untuk memperkirakan parameter distribusi probabilitas atau model statistik.
- Karena statistika Bayes memperlakukan probabilitas sebagai tingkat kepercayaan, teorema Bayes dapat menetapkan distribusi probabilitas secara langsung untuk mengkuantifikasi keyakinan pada suatu parameter atau serangkaian parameter.

Dengan menggunakan metode statistika Bayes, kita dapat memperbarui probabilitas suatu peristiwa setelah mendapatkan informasi baru, sehingga dapat membantu kita membuat keputusan yang lebih baik dan prediksi yang lebih akurat dalam berbagai bidang, seperti ilmu pengetahuan, bisnis, dan teknologi.

Regenerate response

Referensi: [Bayes' Theorem: What It Is, the Formula, and Examples](#)

Statistical Thinking

Dasar-Dasar Statistik

Dengan menggunakan teorema Bayes, kita dapat mencari peluang terjadinya A, dengan syarat B telah terjadi. Di sini, B adalah bukti dan A adalah hipotesis. Asumsi yang dibuat di sini adalah bahwa prediktor/fiturnya independen. Artinya kehadiran satu fitur tertentu tidak mempengaruhi yang lain.

Contoh Numerik Teorema Bayes :

Sebagai contoh numerik, bayangkan ada tes narkoba yang akurat 98%, artinya 98% dari waktu, itu menunjukkan hasil positif yang benar untuk seseorang yang menggunakan narkoba, dan 98% dari waktu, itu menunjukkan hasil yang benar-benar negatif. untuk bukan pengguna obat.

Selanjutnya, asumsikan 0,5% orang menggunakan obat tersebut. Jika seseorang yang dipilih secara acak dites positif menggunakan narkoba, perhitungan berikut dapat dilakukan untuk menentukan probabilitas orang tersebut benar-benar pengguna narkoba

Dalam contoh numerik ini, diasumsikan bahwa 0,5% orang menggunakan obat tersebut dan tes narkoba memiliki akurasi 98%. Jika seseorang dipilih secara acak dan dites positif menggunakan narkoba, kita dapat menggunakan Teorema Bayes untuk menghitung probabilitas bahwa orang tersebut benar-benar menggunakan obat tersebut.

$$(0,98 \times 0,005) / [(0,98 \times 0,005) + ((1 - 0,98) \times (1 - 0,005))] = 0,0049 / (0,0049 + 0,0199) = 19,76\%.$$

Hasil perhitungan menunjukkan bahwa probabilitasnya sekitar 19,76%. Dengan kata lain, meskipun seseorang dites positif, masih ada kemungkinan sekitar 80% bahwa orang tersebut sebenarnya tidak menggunakan obat tersebut.

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

Statistical Thinking

Dasar-Dasar Statistik

6. Inferential Statistics



Statistik inferensial memungkinkan perbandingan perbedaan antara kelompok perlakuan. Pengukuran dari sampel subjek dalam eksperimen digunakan untuk membandingkan kelompok perlakuan dan membuat generalisasi tentang populasi subjek yang lebih besar. Terdapat berbagai jenis statistik inferensial yang sesuai untuk desain penelitian dan karakteristik sampel tertentu. Oleh karena itu, peneliti disarankan untuk membaca banyak teks tentang desain eksperimental dan statistik untuk menemukan uji statistik yang tepat untuk eksperimen mereka.

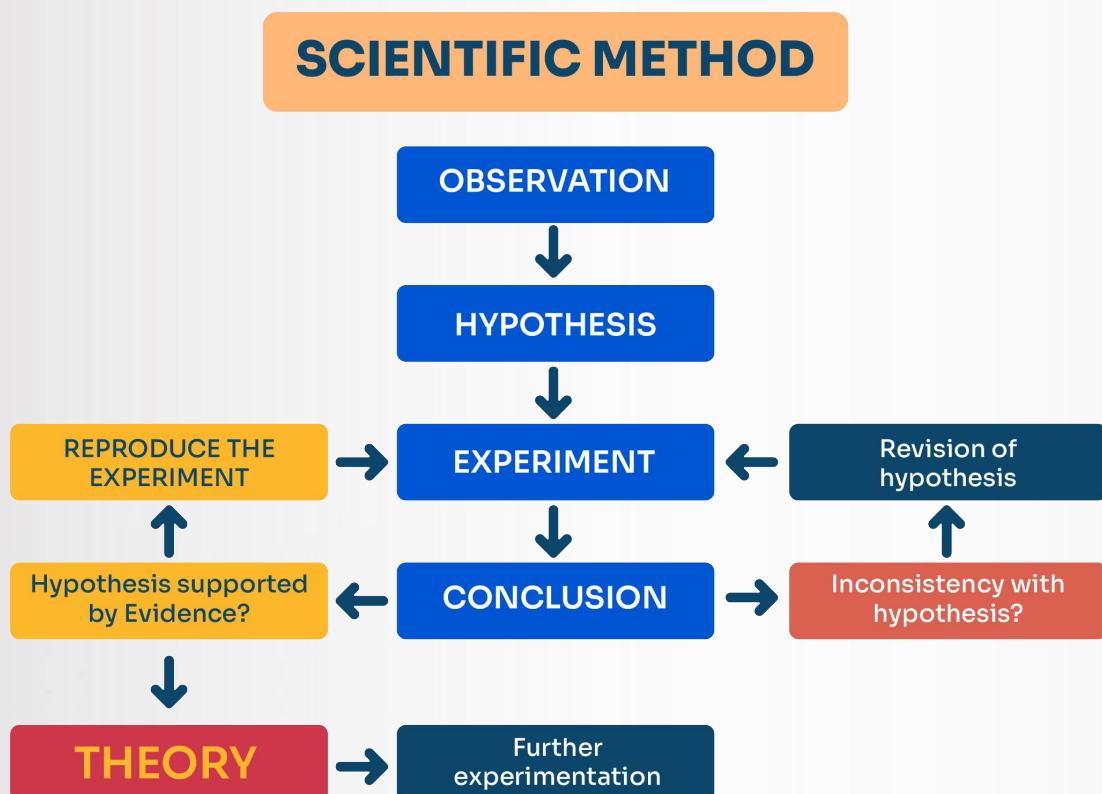
Statistik inferensial didasarkan pada perhitungan nilai uji statistik yang menggunakan rumus tertentu, derajat kebebasan, ukuran sampel, dan kriteria penolakan untuk menentukan perbedaan antara kelompok perlakuan. Ukuran sampel yang semakin besar akan meningkatkan kemungkinan adanya perbedaan antara kelompok perlakuan dan memperkuat statistik. Asumsi dasar yang penting dalam statistik inferensial adalah setiap replikasi dalam suatu kondisi dianggap independen, artinya nilai dalam kondisi tidak berhubungan dengan nilai lain dalam sampel.

Statistik inferensial digunakan utamanya untuk melakukan estimasi pada populasi yang besar dan mengambil kesimpulan dari data berdasarkan pengujian hipotesis. Penggunaan data sampel dalam statistik inferensial lebih ekonomis dan tidak membosankan dibandingkan dengan pengumpulan data dari seluruh populasi. Dengan demikian, hal ini memungkinkan untuk membuat asumsi yang rasional tentang karakteristik populasi yang lebih besar berdasarkan sampel. Penting untuk memastikan bahwa metode pengambilan sampel dilakukan secara acak dan tidak bias agar kesimpulan dan inferensi statistik dapat diandalkan.

Hypothesis

Pernyataan sementara/praduga peneliti terhadap masalah yang dihadapi dalam **penelitian**. Namun, hipotesis belum bisa dianggap suatu kebenaran apabila belum melalui proses testing. Oleh karena itu, dalam hypothesis testing, sebuah hipotesis dapat ditolak atau gagal ditolak (bukan berarti diterima karena masih belum terbukti kebenarannya).

Pada dasarnya, hypothesis merupakan prosedur yang didasarkan pada bukti sampel yang dipakai untuk **menentukan apakah hipotesis merupakan suatu pernyataan yang wajar** dan oleh karenanya tidak ditolak, **atau hipotesis tersebut tidak wajar** dan area itu ditolak.



Bagaimana prosedur pengujian Hipotesis?

Secara umum, pengujian dilakukan terhadap suatu populasi dimana analisisnya dilakukan melalui pengambilan sampel dari populasi tersebut.

Contoh:

Untuk menguji klaim bahwa rata-rata nilai mahasiswa Universitas Startup Campus sebesar 85, pengujian dilakukan pada sampel sebanyak 100 orang.



Membuat klaim awal mengenai rata-rata nilai populasi adalah 84 ($H_0: \mu = 84$)

Pengujian klaim dilakukan berdasarkan sampel pada 100 orang

Sebelum masuk kita masuk ke **bagaimana cara melakukan pengujian hipotesis**, ada beberapa hal yang harus kita ketahui, yaitu:

Hipotesis Nol (H_0)

- **Memprediksi adanya persamaan** antara satu kondisi dengan kondisi lain
- Selalu **ternotasi “=”, “≤”, atau “≥”**
- Ada kemungkinan dianggap ditolak

Hipotesis Alternatif (H_1)

- Menggambarkan apa yang akan Anda simpulkan jika menolak H_0
- **Memprediksi adanya perbedaan** antara satu kondisi dengan kondisi lain
- **Tidak pernah bernotasi “=”, “≤”, atau “≥”**



Berikut adalah langkah-langkah dalam melakukan pengujian hipotesis:

1. Merumuskan Hipotesis

Pengujian studi kasus dua arah dinotasikan sebagai berikut:

$$H_0: \mu = \mu_0$$

$$H_1: \mu \neq \mu_1$$

Misalkan kita menduga nilai rata-rata mahasiswa Universitas Startup Campus sama dengan 84, maka dapat dinotasikan:

$$H_0: \mu = 84$$

$$H_1: \mu \neq 84$$

Pengujian studi kasus satu arah dinotasikan sebagai berikut:

$$H_0: \mu \geq \mu_0$$

$$H_1: \mu < \mu_1$$

atau

$$H_0: \mu \leq \mu_0$$

$$H_1: \mu > \mu_1$$

Misalkan kita menduga nilai rata-rata mahasiswa Universitas Startup Campus lebih dari sama dengan 84, maka dapat dinotasikan:

$$H_0: \mu \geq 84$$

$$H_1: \mu < 84$$

Bagaimana cara membedakan suatu studi kasus disebut satu atau dua arah?

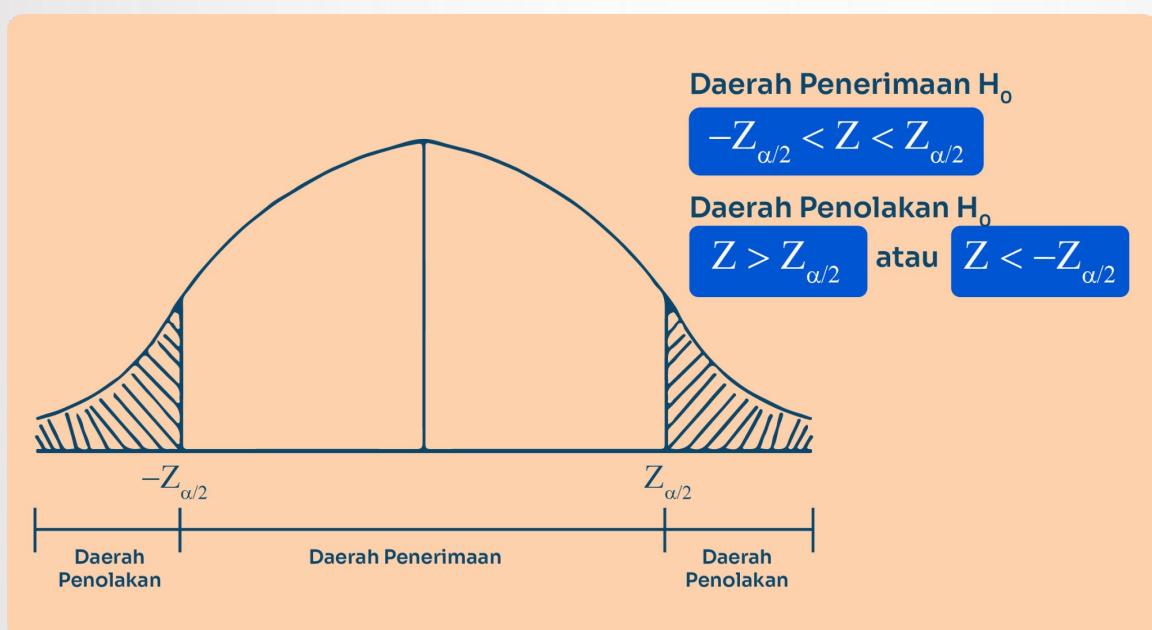
Pengujian Dua Arah

Biasanya **tidak dinyatakan apakah “lebih dari” atau “kurang dari”**. Perhatikan contoh berikut:

- Kita menduga nilai rata-rata mahasiswa Universitas Startup Campus sama dengan 84
- Rata-rata biaya menginap sebuah kamar di sebuah hotel di Washington DC adalah sekitar \$168 per malam
- Dengan adanya pengadaan alat produksi baru, Direktur PT. Startup Campus me-klaim bahwa rata-rata buku yang diproduksi adalah 1500 eksemplar

Terlihat dari beberapa contoh tersebut bahwa pengujian dua arah dilakukan apabila kita akan menguji suatu kondisi yang menyatakan “sama dengan”.

Gambar di bawah merupakan visualisasi pengujian dua arah



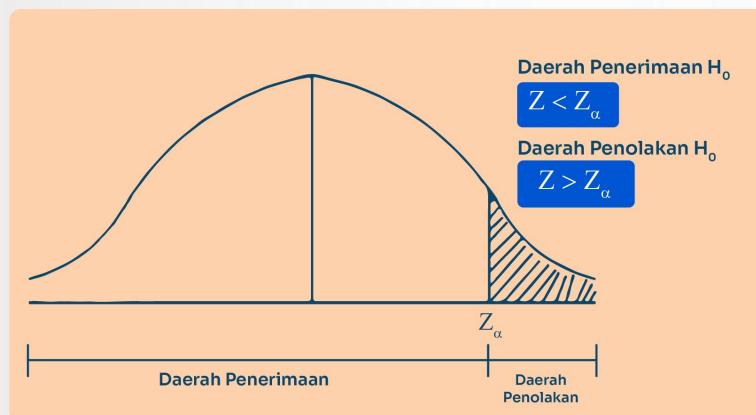
Pengujian Satu Arah

Berbeda halnya dengan pengujian dua arah, biasanya **dinyatakan apakah “lebih dari” atau “kurang dari”**. Perhatikan contoh berikut:

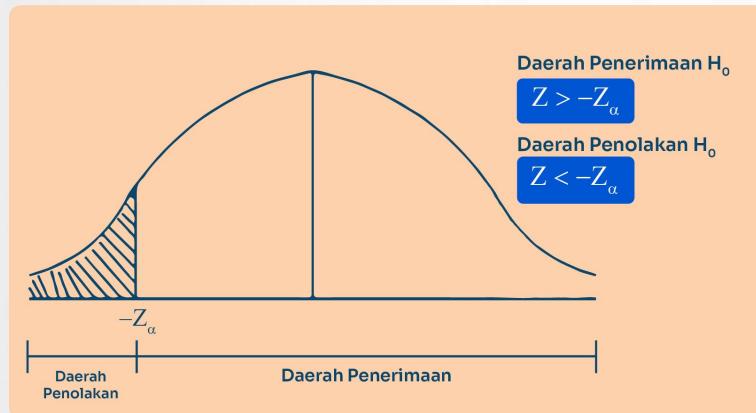
- Kita menduga nilai rata-rata mahasiswa Universitas Startup Campus lebih dari sama dengan 84
- Rata-rata biaya menginap sebuah kamar di sebuah hotel di Washington DC minimal sebesar \$168 per malam
- Dengan adanya pengadaan alat produksi baru, Direktur PT. Startup Campus me-klaim bahwa rata-rata buku yang diproduksi bisa lebih dari 1500 eksemplar

Terlihat dari beberapa contoh tersebut bahwa pengujian satu arah dilakukan apabila kita akan menguji suatu kondisi yang menyatakan suatu yang lebih dari atau kurang dari atau minimal.

Gambar di bawah merupakan visualisasi pengujian satu arah untuk kurang dari



Gambar di bawah merupakan visualisasi pengujian satu arah untuk lebih dari



[Image Ref](#)

2. Menentukan Significance Level

- Dalam merumuskan hipotesis, **kita harus menentukan taraf signifikansi** (significance level) yang dinotasikan sebagai α
- Sebagai **simbol berapa persen tingkat kesalahan** dalam mengajukan hipotesis atau klaim
 - Semakin kecil significance level \rightarrow semakin besar tingkat kepercayaan pengambilan keputusan
 - Umumnya significance level yang ditetapkan untuk studi kasus terkait kesehatan adalah 0,01 dan 0,05 untuk non kesehatan

3. Menguji Secara Statistik

Selanjutnya, kita bisa menguji hipotesis secara statistik dimana yang cukup sering digunakan adalah pengujian rata-rata. Perhatikan rumus berikut untuk menguji hipotesis:

Apabila standar deviasi populasi (dinotasikan σ) diketahui atau ukuran sampel lebih dari 30, maka kita bisa menggunakan z-test dengan persamaan berikut:

$$Z = (\bar{x} - \mu) / (\sigma / \sqrt{n})$$

Apabila standar deviasi populasi (dinotasikan σ) tidak diketahui atau ukuran sampel kurang dari 30, maka kita menggunakan t-test dengan persamaan berikut. Dimana S merupakan standar deviasi pada sampel.

$$t = (\bar{x} - \mu) / (S / \sqrt{n})$$

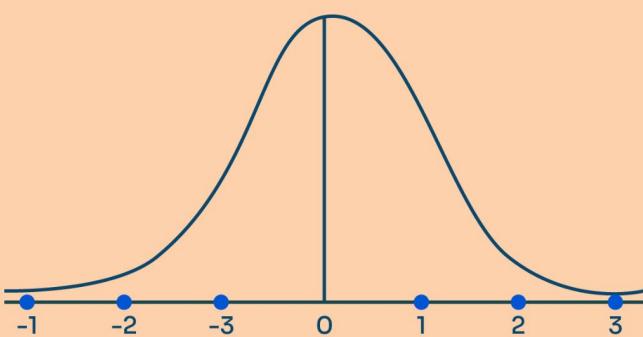
Additional Information

Kalau kita sedikit me-review modul sebelumnya, sekilas rumus Z-test mirip dengan Z-Score dengan persamaan berikut. Apakah keduanya sama?

$$Z = (\bar{x} - \mu) / \sigma$$

Dari segi penggunaannya, **Z-test biasa dipakai kalau kita ingin menguji hipotesis terhadap suatu populasi data yang besar**(jumlah datanya lebih dari 30) serta standar deviasinya telah diketahui. Sementara itu, **Z-Score merupakan pengukuran statistika yang digunakan untuk mengetahui bagaimana hubungan atau jarak terhadap nilai rata-rata** suatu kelompok data. Hubungan yang dimaksud adalah berapa jauh nilai standar deviasi terhadap nilai rata-rata. Oleh karena itu, kita membutuhkan nilai standar deviasi. Perhatikan gambar di bawah, apabila Z-Score sama dengan 0, maka mengindikasikan nilai titik data tersebut identik dengan rata-rata. Sebaliknya, semakin jauh Z-Score dari 0 maka semakin tidak identik dengan nilai rata-rata.

Z-scores & the standard normal distribution



Z-scores distribution

- if the Z score is negative, then the score falls **below** the mean
- If the Z score is 0, then the score falls **at** the mean
- If the Z score is positive, then the score falls **above** the mean

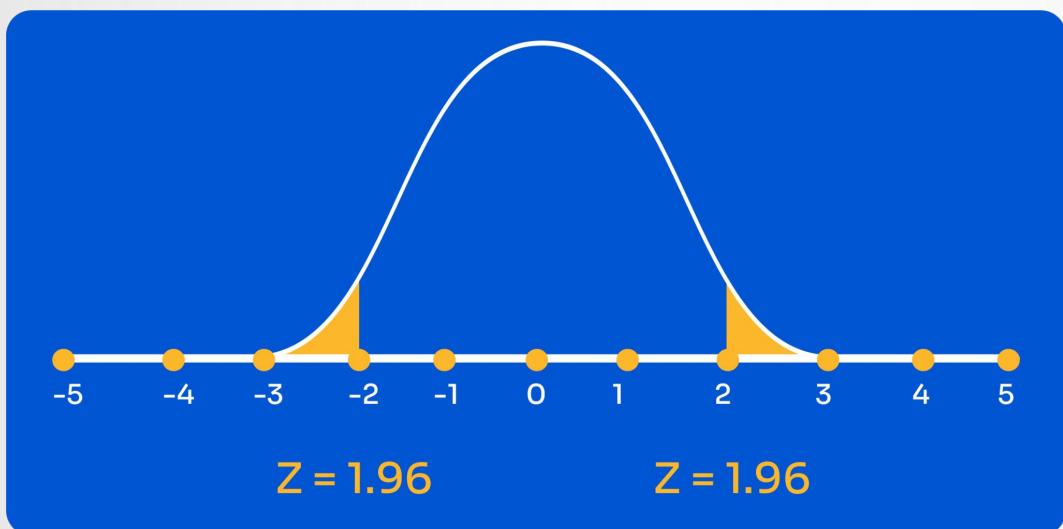
Misalkan nilai critical value kita pada pengujian dua arah adalah 0,025, maka nilai Z kita adalah 1,96. Cara melihatnya:

1. Cari nilai yang sama dengan critical value kita, misalkan 0,025
2. Berdasarkan gambar berikut, nilai 0,025 berada di baris -1,96 dan kolom 0,06 sehingga nilai $Z_{\alpha/2}$ adalah 1,96

Tabel Z Distribusi Normal

z	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
-3.5	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
-3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
-3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
-3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
-3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
-3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
-2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
-2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
-2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
-2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
-2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
-2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
-2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
-2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
-2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
-2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
-1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
-1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294

Gambar di bawah merupakan visualisasi daerah penerimaan dengan critical value 1,96. H_0 diterima selama masih berada dalam rentang -1,96 hingga 1,96.



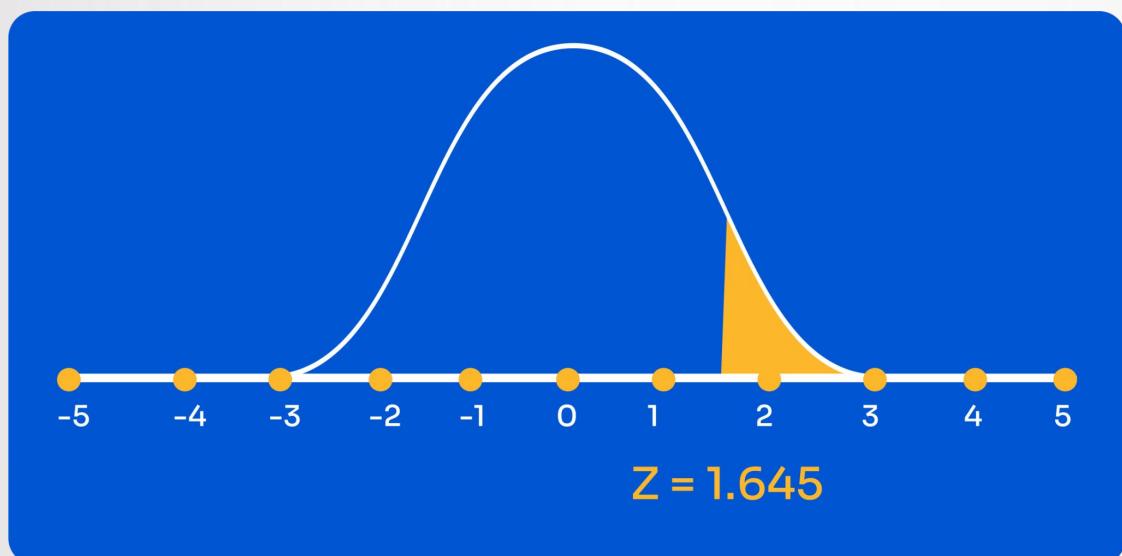
Misalkan nilai critical value kita pada pengujian satu arah adalah 0,025, maka nilai Z kita adalah 1,96. Cara melihatnya:

1. Cari nilai yang sama dengan critical value kita, misalkan 0,025
2. Berdasarkan gambar berikut, nilai 0,025 berada di baris -1,96 dan kolom 0,06 sehingga nilai Z_α adalah 1,96

Tabel Z Distribusi Normal

z	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
-3.5	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002
-3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
-3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
-3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
-3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
-3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
-2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
-2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
-2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
-2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
-2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
-2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
-2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
-2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
-2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
-2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
-1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
-1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294

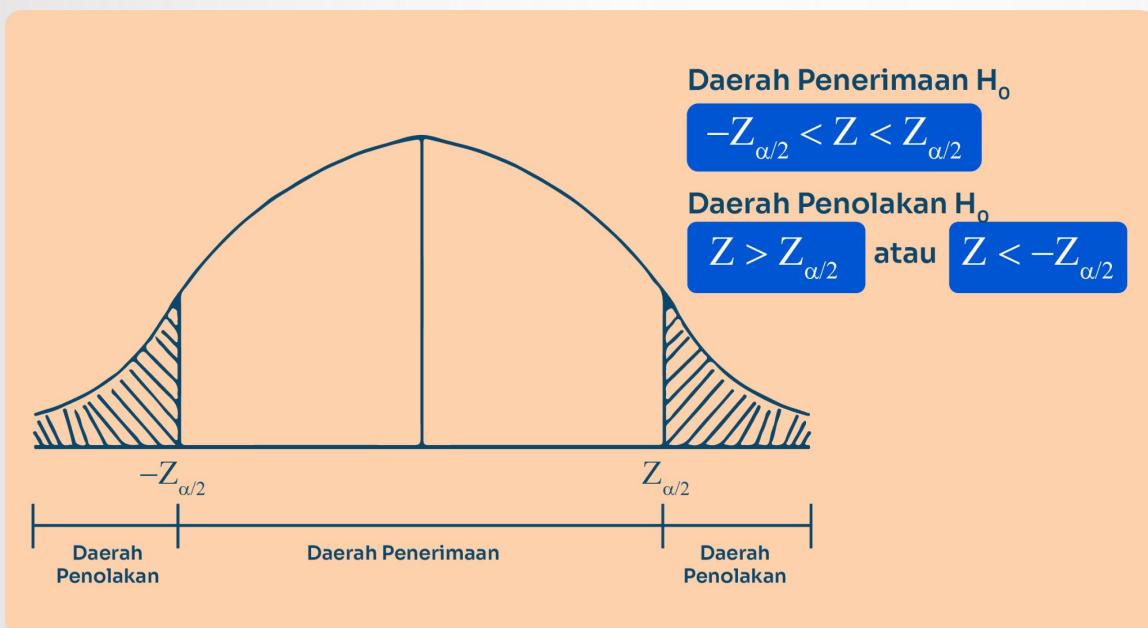
Gambar di bawah merupakan visualisasi daerah penerimaan dengan critical value 1,96. H_0 diterima selama masih berada dalam rentang di bawah 1,96.



4. Pengambilan Keputusan

(Pengujian ketika ukuran sampel lebih dari 30)

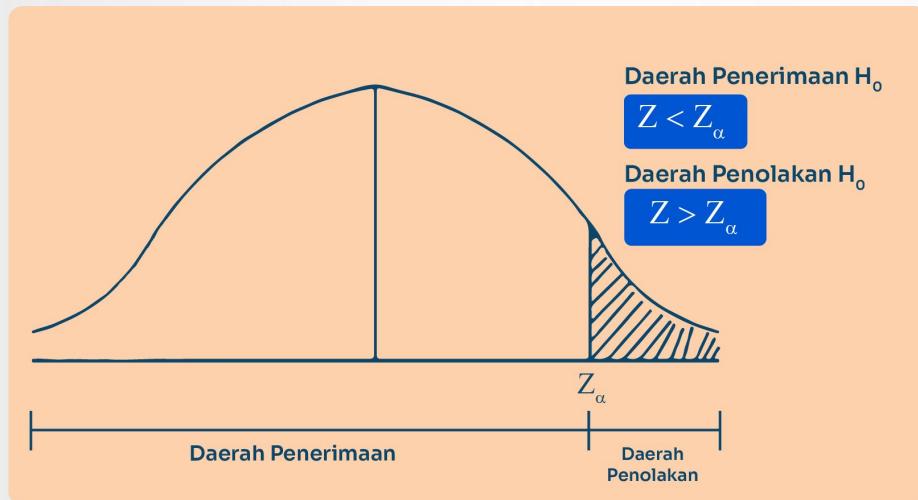
Pengambilan keputusan dapat dilakukan melalui pembuatan visualisasi daerah penerimaan dan penolakan berdasarkan significance level.



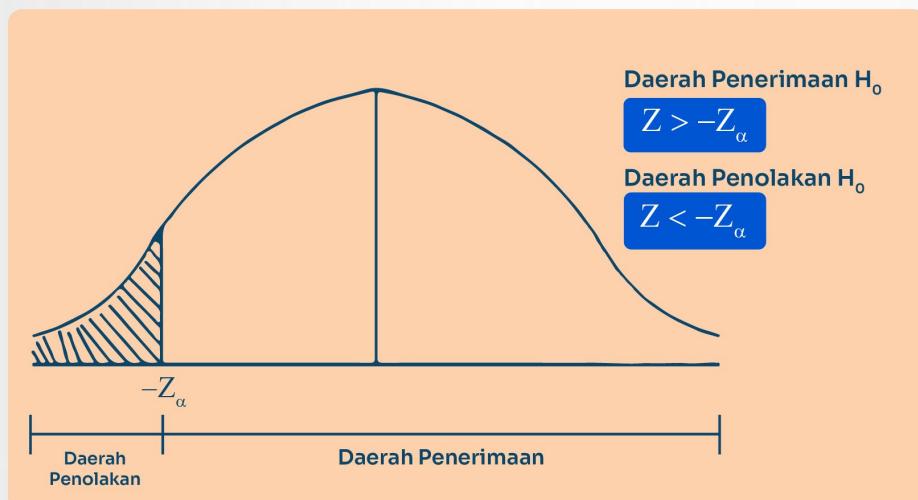
Pada pengujian dua arah, keputusan dapat berupa klaim tidak ditolak atau klaim yang diajukan sesuai dengan kondisi yang ada jika H_0 diterima. Berdasarkan gambar di atas, H_0 diterima selama nilainya masih berada di antara $-Z_{\alpha/2}$ dan $Z_{\alpha/2}$ dimana nilai $Z_{\alpha/2}$ atau biasa disebut critical value didapat dari tabel distribusi Z dengan membagi dua nilai α . Berikut merupakan bagaimana mengetahui critical value:

- Mengetahui significance level (α)
- Mengetahui nilai critical value dengan membagi nilai α dengan dua
- Berdasarkan nilai critical value, kita mencari nilai Z berdasarkan tabel distribusi Z yang bisa diakses [disini](#)

Pada pengujian satu arah, khususnya untuk suatu studi kasus kurang dari atau kurang sama dengan, keputusan dapat berupa klaim tidak ditolak jika H_0 tidak lebih dari Z_α seperti terlihat di gambar di bawah.



Gambar di bawah merupakan visualisasi daerah penerimaan dan penolakan pada pengujian satu arah dengan studi kasus lebih dari atau lebih sama dengan, keputusan dapat berupa klaim tidak ditolak jika H_0 tidak kurang dari Z_α .



Sama halnya dengan pengujian dua arah, berikut merupakan bagaimana mengetahui critical value pada pengujian satu arah:

- Mengetahui significance level (α)
- Mengetahui nilai critical value. Namun, yang menjadi catatan adalah kita tidak perlu membagi nilai α dengan dua karena kita hanya menguji di satu sisi saja. Berbeda halnya dengan pengujian dua arah yang mengharuskan menguji pada masing-masing sisi
- Berdasarkan nilai critical value, kita mencari nilai Z berdasarkan tabel distribusi Z yang bisa diakses [disini](#)



4. Pengambilan Keputusan (Pengujian ketika ukuran sampel kurang dari 30)

Sama halnya dengan pengujian pada sampel sebesar 30 atau lebih, pengambilan keputusan dapat dilakukan melalui pembuatan visualisasi daerah penerimaan dan penolakan berdasarkan significance level. Namun, yang sedikit membedakan adalah bagaimana dalam menentukan critical value. Hal ini dikarenakan kita harus mempertimbangkan degree of freedom(df) yang diketahui dari nilai sampel dikurangi 1. Selain itu, tabel yang harus kita lihat juga berbeda karena yang digunakan adalah tabel distribusi t sehingga pengujian ini biasa disebut t-test.

Berikut merupakan bagaimana mengetahui critical value:

- Mengetahui significance level (α)
- Mengetahui nilai critical value
 - Untuk pengujian dua arah, kita perlu membagi nilai α dengan dua
 - Untuk pengujian satu arah, kita tidak perlu membagi nilai α dengan dua
- Mengetahui nilai df yang diperoleh dari:
$$df = \text{nilai sampel} - 1$$
- Berdasarkan nilai α dan df, kita mencari berdasarkan tabel distribusi t yang bisa diakses [disini](#)

Misalkan nilai α kita adalah 0,025 dan ukuran sampelnya adalah 5, maka critical value kita adalah 2,78. Cara melihatnya:

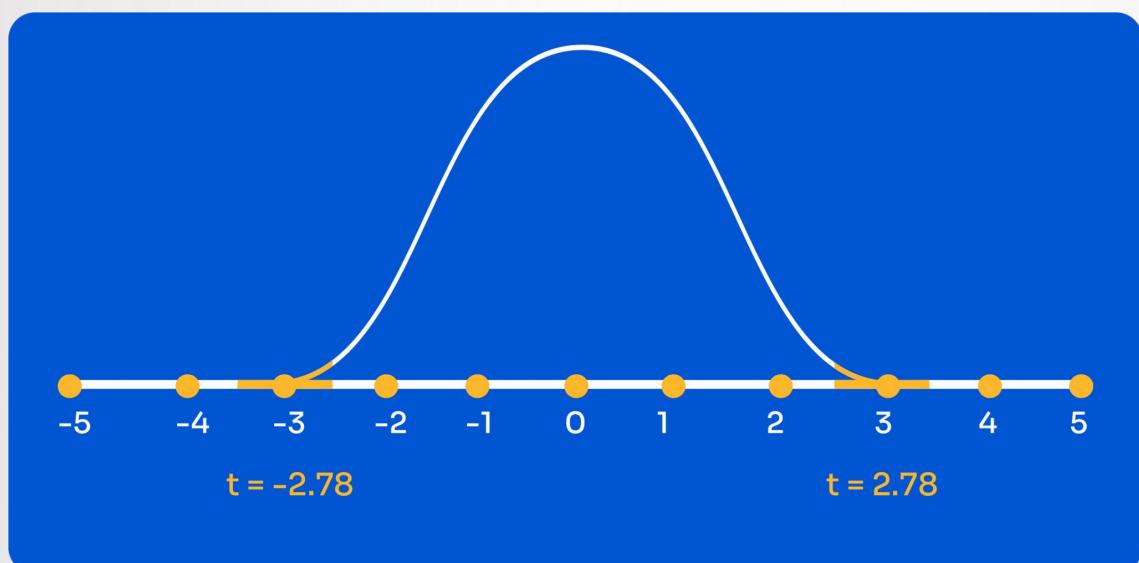
1. Perhatikan kolom yang memiliki nilai α sebesar 0,025
2. Perhatikan baris yang memiliki nilai 4, didapat dari persamaan:

$$\begin{aligned} df &= \text{nilai sampel} - 1 \\ &= 5 - 1 \\ &= 4 \end{aligned}$$

1. Berdasarkan gambar berikut, nilai critical value yang berada di kolom 0,025 dan baris 4 adalah 2,78 (hasil pembulatan)

α	0.1	0.05	0.025	0.01	0.005	0.0025	0.001
df							
1	3.077684	6.313752	12.706205	31.820516	63.656741	127.321336	318.308839
2	1.885618	2.919986	4.302653	6.964557	9.924843	14.089047	22.327125
3	1.637744	2.353363	3.182446	4.540703	5.840909	7.453319	10.214532
4	1.533206	2.131847	2.776445	3.746947	4.604095	5.597568	7.173182
5	1.475884	2.015048	2.570582	3.364930	4.032143	4.773341	5.893430
6	1.429756	1.942100	2.116912	2.112660	2.707420	3.216027	5.207626

Gambar di bawah merupakan visualisasi daerah penerimaan dengan critical value 2,78. H_0 diterima selama masih berada dalam rentang $-2,78$ hingga $2,78$.



Yuk kita latihan sedikit!

Rata-rata biaya menginap sebuah kamar di sebuah hotel di Washington DC adalah sekitar \$168 per malam. Sebuah sampel dilakukan terhadap 25 hotel menghasilkan rata-rata hitung sekitar \$172,5 dengan simpangan baku \$15,4. Ujilah hipotesis tersebut! ($\alpha = 0,05$)

Jawab:

1. Perumusan Hipotesis

Seperti yang sudah kita bahas sebelumnya, pertama kita rumuskan terlebih dahulu perumusan hipotesisnya. Berhubung rata-rata biaya menginap sebuah hotel di Washington DC adalah sekitar \$168 pe malam, maka perumusannya adalah seperti berikut:

$$H_0: \mu = 168$$

$$H_1: \mu \neq 168$$

2. Menentukan Significance Level

Sebelum menguji statistik, kita harus mengetahui berapa significance level yang ditentukan. Tercantum di soal bahwa significance level kali ini adalah 0,05 yang direpresentasikan sebagai α .

3. Menguji Secara Statistik

Kita akan menguji menggunakan pengujian menggunakan distribusi t karena sampel yang dimiliki hanya 25. Oleh karena itu, perhitungannya adalah sebagai berikut:

$$\begin{aligned}
 t &= (\bar{x} - \mu) / (S / \sqrt{n}) \\
 &= (172,5 - 168) / (15,4 / \sqrt{25}) \\
 &= 1,14
 \end{aligned}$$

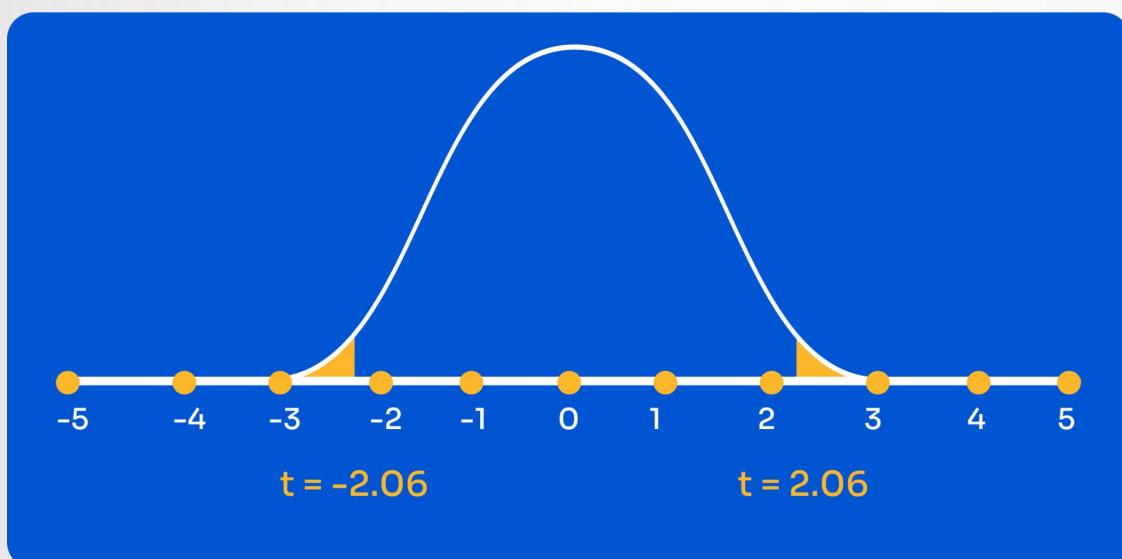
4. Pengambilan Keputusan

Seperti pembahasan sebelumnya, yang harus kita lakukan adalah:

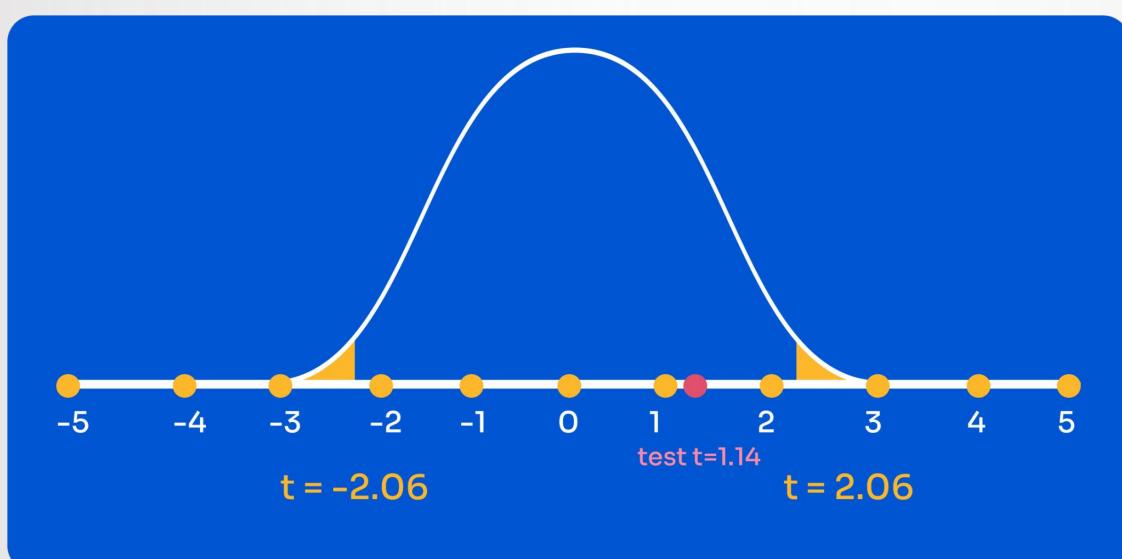
- Significance level (α) sebesar 0,05. Berhubung ini adalah pengujian dua arah, maka kita harus membagi dua nilai α sehingga menjadi 0,025
- $df = \text{nilai sampel} - 1$
 $= 25 - 1$
 $= 24$
- Berdasarkan $\alpha/2 = 0,025$ dan nilai $df = 24$, terlihat pada tabel distribusi t nilai critical value nya adalah 2,06 (pembulatan)

α	0.1	0.05	0.025	0.01	0.005	0.0025	0.001
df							
1	3.077684	6.313752	12.706205	31.820516	63.656741	127.321336	318.308839
2	1.885618	2.919986	4.302653	6.964557	9.924843	14.089047	22.327125
3	1.637744	2.353363	3.182446	4.540703	5.840909	7.453319	10.214532
4	1.533206	2.131847	2.776445	3.746947	4.604095	5.597568	7.173182
5	1.475884	2.015048	2.570582	3.364930	4.032143	4.773341	5.893430
6	1.439756	1.943180	2.446912	3.142668	3.707428	4.316827	5.207626
7	1.414924	1.894579	2.364624	2.997952	3.499483	4.029337	4.785290
8	1.396815	1.859548	2.306004	2.896459	3.355387	3.832519	4.500791
9	1.383029	1.833113	2.262157	2.821438	3.249836	3.689662	4.296806
10	1.372184	1.812461	2.228139	2.763769	3.169273	3.581406	4.143700
11	1.363430	1.795885	2.200985	2.718079	3.105807	3.496614	4.024701
12	1.356217	1.782288	2.178813	2.680998	3.054540	3.428444	3.929633
13	1.350171	1.770933	2.160369	2.650309	3.012276	3.372468	3.851982
14	1.345030	1.761310	2.144787	2.624494	2.976843	3.325696	3.787390
15	1.340606	1.753050	2.131450	2.602480	2.946713	3.286039	3.732834
16	1.336757	1.745884	2.119905	2.583487	2.920782	3.251993	3.686155
17	1.333379	1.739607	2.109816	2.566934	2.898231	3.222450	3.645767
18	1.330391	1.734064	2.100922	2.552380	2.878440	3.196574	3.610485
19	1.327728	1.729133	2.093024	2.539483	2.860935	3.173725	3.579400
20	1.325341	1.724718	2.085963	2.527977	2.845340	3.153401	3.551808
21	1.323188	1.720743	2.079614	2.517648	2.831360	3.135206	3.527154
22	1.321237	1.717144	2.073873	2.508325	2.818756	3.118824	3.504992
23	1.319460	1.713872	2.068658	2.499867	2.807336	3.103997	3.484964
24	1.317836	1.710882	2.063899	2.492159	2.796940	3.090514	3.466777
25	1.316245	1.708141	2.058520	2.485107	2.787406	3.079100	3.450100

Selanjutnya kita bisa menggambarkan daerah penerimaan dan penolakan berdasarkan nilai t dan critical value. Perhatikan gambar di bawah, berdasarkan identifikasi critical value dan studi kasus, daerah penerimaan H_0 adalah $-2,06 \leq t \leq 2,06$ dimana bagian yang berarsir warna biru adalah daerah penolakan.



Perhatikan gambar di bawah, diketahui nilai t kita adalah 1,14 dimana masih berada di daerah penerimaan. Hal ini mengindikasikan bahwa H_0 tidak ditolak dan keputusan atau kesimpulan yang bisa diambil adalah klaim yang diajukan bisa dibilang sesuai dengan kondisi yang ada.



Pembahasan uji hipotesis yang baru saja kita bahas biasa digunakan pada satu sampel atau kelompok data. **Lalu pertanyaannya, bagaimana kalau kita ingin menguji pada lebih dari satu kelompok sampel data?** Misalkan kita ingin membandingkan perbedaan antara data penjualan dan pendapatan atau kita ingin mengetahui bagaimana pengaruh jumlah klik terhadap pembelian pada suatu aplikasi website.

Salah satu teknik uji hipotesis yang dapat digunakan adalah A/B Testing. Teknik tersebut biasa digunakan untuk menguji seberapa signifikan perbedaan antar dua kelompok data. Teknik ini menggunakan t-test dan Mann-Whitney U Test. Bagaimana caranya?



1. Menguji Normality Assumption

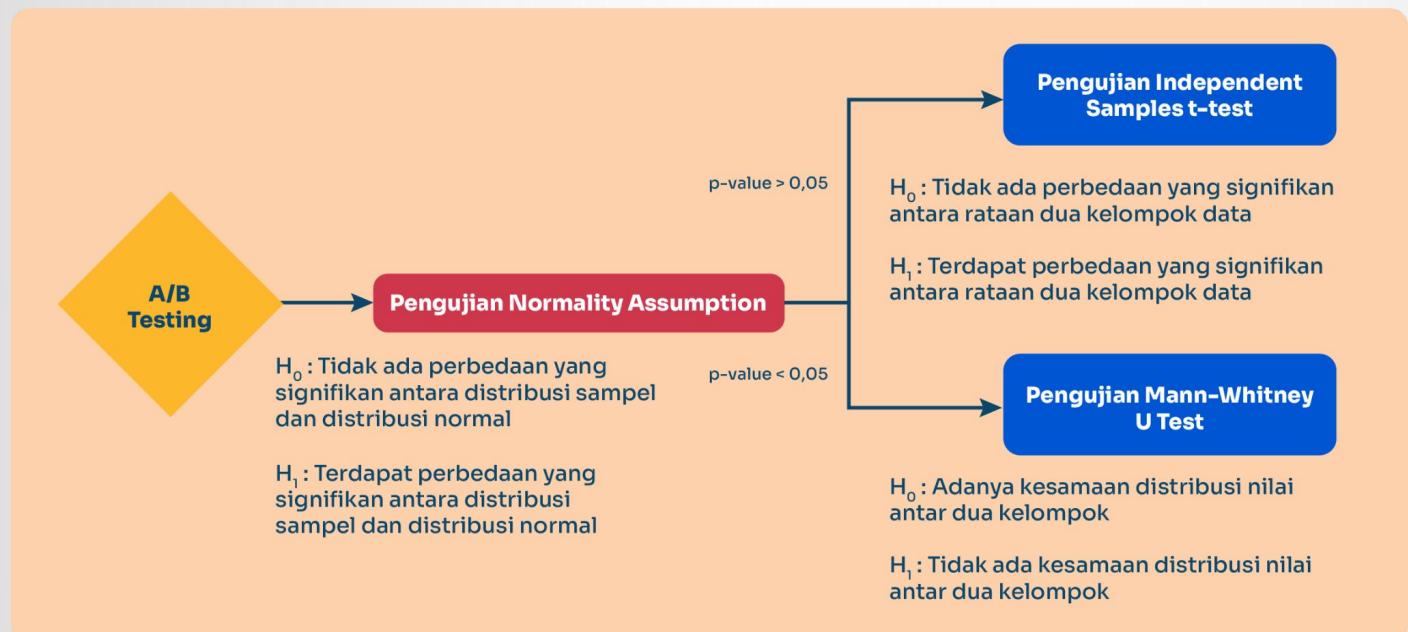
Hal yang pertama dilakukan adalah menguji apakah data yang dimiliki berdistribusi normal atau tidak. Pengujian yang menggunakan teknik Shapiro Wilks test ini akan menentukan kita akan menggunakan teknik Independent Samples t-test atau Mann-Whitney U Test pada pengujian selanjutnya. Bagaimana caranya?

Sama halnya dengan pembahasan sebelumnya, pertama kita menguji suatu hipotesis sebagai berikut:

H_0 : Tidak ada perbedaan yang signifikan antara distribusi sampel dan distribusi normal

H_1 : Terdapat perbedaan yang signifikan antara distribusi sampel dan distribusi normal

- Apabila nilai p-value lebih besar daripada 0,05, maka H_0 tidak dapat ditolak sehingga teknik pengujian yang digunakan selanjutnya adalah Independent Samples t-test
- Apabila nilai p-value lebih kecil daripada 0,05, maka H_0 dapat ditolak sehingga teknik pengujian yang digunakan selanjutnya adalah Mann-Whitney U test



2. Menguji Homogeneity Assumption

(Dilakukan ketika H_0 tidak ditolak pada langkah pertama)

Langkah selanjutnya adalah melakukan pengujian Homogeneity Assumption menggunakan Lavene's Test. Kali ini kita akan menguji apakah antar dua kelompok data memiliki perbedaan signifikan sehingga perumusan hipotesisnya adalah:

H_0 : Tidak ada perbedaan yang signifikan antara rataan dua kelompok data

H_1 : Terdapat perbedaan yang signifikan antara rataan dua kelompok data

Sama halnya dengan langkah pertama, kita juga memperhatikan p-value sebagai berikut:

- Apabila nilai p-value lebih besar daripada 0,05, maka H_0 tidak dapat ditolak sehingga terindikasi dua kelompok data tidak terdapat perbedaan
- Apabila nilai p-value lebih kecil daripada 0,05, maka H_0 dapat ditolak sehingga terindikasi dua kelompok data terdapat perbedaan

3. Menguji Homogeneity

Pengujian ini menggunakan Independent Samples t-test yang berfungsi menampilkan informasi apakah terdapat perbedaan yang signifikan antara dua kelompok data

H_0 : Tidak ada perbedaan nilai varians antara dua kelompok data

H_1 : Terdapat perbedaan nilai varians antara dua kelompok data

- Apabila nilai p-value lebih besar daripada 0,05, maka H_0 tidak dapat ditolak sehingga tidak terindikasi adanya perbedaan dua varians kelompok data
- Apabila nilai p-value lebih kecil daripada 0,05, maka H_0 dapat ditolak sehingga terindikasi adanya perbedaan dua varians kelompok data

2. Menguji Homogeneity Assumption

(Dilakukan ketika H_0 ditolak pada langkah pertama)

Apabila kita menolak H_0 pada langkah pertama, maka pengujian Homogeneity Assumption dilakukan menggunakan Mann-Whitney U Test. Maka perumusan hipotesis adalah sebagai berikut:

H_0 : Adanya kesamaan distribusi nilai antar dua kelompok

H_1 : Tidak ada kesamaan distribusi nilai antar dua kelompok

Sama halnya dengan langkah pertama, kita juga memperhatikan p-value sebagai berikut:

- Apabila nilai p-value lebih besar daripada 0,05, maka H_0 tidak dapat ditolak sehingga terindikasi tidak ada perbedaan distribusi nilai antar dua kelompok data
- Apabila nilai p-value lebih kecil daripada 0,05, maka H_0 dapat ditolak sehingga terindikasi adanya perbedaan distribusi nilai antar dua kelompok data

Digital Leadership

Digital leadership atau e-Leadership adalah bentuk kepemimpinan yang berkembang akibat adanya perubahan lingkungan menjadi berbasis teknologi dan elektronik. Kepemimpinan digital dibutuhkan dalam proses transformasi digital yang sedang berlangsung di berbagai sektor, karena pemimpin digital dapat mengawasi dan menggerakkan perubahan dan penggunaan teknologi dengan cepat dan efektif di dalam organisasi. Kehadiran pemimpin digital di dalam organisasi dapat mendorong percepatan transformasi digital. Pemimpin digital juga dapat memanfaatkan aset digital yang dimiliki oleh pegawainya untuk mencapai tujuan organisasi. Selain itu, pemimpin digital juga dapat memanfaatkan teknologi digital yang terhubung dengan proses bisnis masing-masing instansi pemerintah dalam melakukan transformasi layanan dan meningkatkan efektivitas dan efisiensi layanan publik. Terdapat empat karakteristik yang membedakan kepemimpinan biasa dengan e-Leadership, antara lain:



1. Digital leader harus mampu berkomunikasi secara efektif melalui media sosial.



2. Digital leader harus memiliki kemampuan untuk bekerja tanpa batasan waktu, ruang, dan budaya.



3. Digital leader harus mampu memantau dan mengelola pekerjaan secara virtual.



4. Digital leader harus beradaptasi dengan perubahan lingkungan teknologi.

EXERCISE

Berikut adalah *exercise* yang dapat digunakan untuk melatih pemahaman kamu:

1. Jelaskan tentang *skill - skill* yang diperlukan oleh *data scientist* untuk menyelesaikan pekerjaannya!
2. Berikan contoh-contoh penerapan *data science* pada bidang:
 - *Finance*
 - *Marketing*
 - *Operation*
 - *Sales*
3. Terdapat [Data Harga Rumah di Boston](#), tugas kamu:
 - Berikan analisis CRISP-DM sederhana yang bisa kamu dapatkan dari data tersebut!
 - Jelaskan *feature - feature* data yang ada, dan berikan hipotesis *feature* apa yang berpengaruh terhadap harga rumah!
 - Apa *outcome* yang dapat diharapkan dari data tersebut?



THANK YOU