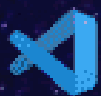


# DETEKSI SPAM SMS NAIVE BAYES

Rolly Maulana Awangga  
Zaky Muhammad Yusuf



$$\mu = \frac{\sum_{i=1}^n X_i}{n}$$



# DETEKSI SPAM SMS NAÏVE BAYES

**Rolly Maulana Awanga**  
**Zaky Muhammad Yusuf**



# DETEKSI SPAM SMS NAÏVE BAYES

***Penulis :***

Rolly Maulana Awangga  
Zaky Muhamamd Yusuf

**ISBN :**

***Editor :***

Rolly Maulana Awangga

***Penyunting :***

Rolly Maulana Awangga

***Desain sampul dan Tata letak :***

Rolly Maulana Awangga  
Zaky Muhammad Yusuf

***Penerbit :***

Penerbit Buku Pedia

***Redaksi :***

Athena Residence Blok. E No. 1, Desa Ciwaruga,  
Kec. Parongpong, Kab. Bandung Barat 40559  
Tel. 628-775-2000-300  
Email : penerbit@bukupedia.co.id

***Distributor :***

Informatics Research Center  
Jl. Sariasih No. 54  
Bandung 40151  
Email : irc@poltekpos.ac.id

Cetakan Pertama, 2023

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan  
dengan cara apa pun tanpa ijin tertulis dari penerbit

# KATA PENGANTAR

Buku Langkah mudah membuat deteksi spam ini disusun untuk memenuhi kebutuhan data dan information tentang python khususnya Data Mining di bagian tutorial, Sebelum tahun 1990-an, data mining awalnya dikenal sebagai subproses besar yang disebut KDD (Knowledge Discovery in Databases). Library baru seperti Pandas dan Sckitlearn dikembangkan untuk meminimalisir keakuratan dan memudahkan untuk diterapkan pada masyarakat luas.

Hingga kini, Python Data Mining menjadi bahasa skrip sisi klien yang banyak digunakan. Menurut data statistik, di tahun 2016 sebanyak 11.27% aplikasi menggunakan Python untuk menarik para pengunjung, termasuk situs-situs ternama seperti Google dan Twitter. Sehingga ini buku ini memenuhi pembaca untuk membuat program secara cepat simple dan sederhana, Berikut link github ini berisi tentang pembuatan system identifikasi spam sms menggunakan naïve bayes secara singkat padat dan jelas.

Bagi kami sebagai penyusun merasa bahwa masih banyak kekurangan dalam penyusunan buku ini karena keterbatasan pengetahuan dan pengalaman kami.

Untuk itu, kami sangat mengharapkan kritik dan saran yang membangun dari pembaca demi kesempurnaan buku ini.

Untuk memudahkan pembaca dalam mempelajari langkah-langkah pada buku ini, kami telah menyediakan kode yang bisa diunduh. Kode untuk buku ini di GitHub dengan alamat <https://github.com/bukped/Deteksi-Spam-SMS-Naive-Bayes>

Bandung, 8 Januari 2023

Penulis

# DAFTAR ISI

KATA PENGANTAR .....	i
BAB I PENGENALAN PYTHON.....	1
<b>A. Pendahuluan</b> .....	1
<b>B. Tujuan dan Capaian</b> .....	1
<b>C. Uraian materi</b> .....	2
1.1 Instalasi Python.....	3
1.2 Python Interpreter .....	5
<b>D. Latihan</b> .....	5
1.3 VARIABLE.....	7
1.4 TUPLE DAN LIST .....	8
1.5 DICTIONARY .....	9
1.6 MY AGE .....	10
1.7 CONDITIONAL .....	11
1.7.1 IF CONDITIONAL.....	11
1.7.2 IF ELSE CONDITIONAL .....	12
1.7.3 IF ELIF CONDITIONAL .....	13
1.7.4 READ WRITE FILE .....	14
1.8 OBJECT ORIENTED PROGRAMMING .....	15
1.9 DATABASE.....	16
1.10 HTTP REQUEST .....	17
1.11 PYTEST.....	18
BAB II DATA MINING .....	27
<b>A. Pendahuluan</b> .....	27
<b>B. Tujuan dan capaian</b> .....	27
<b>C. Uraian Materi</b> .....	28

2.1	Eksekusi Bersyarat.....	28
<b>D.</b>	<b>Latihan.....</b>	<b>29</b>
2.2	Library .....	29
2.3	Apa Itu Pandas? .....	30
2.4	Instalasi .....	30
2.5	DATA STRUCTURES.....	33
2.5.1	SERIES .....	33
2.5.2	DATAFRAME .....	35
2.6	INDEXING AND SELECTING .....	38
2.6.1	MULTI-INDEX.....	38
2.6.2	GETTING SINGLE VALUES .....	39
2.7	GROUP OPERATIONS.....	40
2.7.1	GROUP BY .....	41
2.7.2	AGG.....	41
2.7.3	FILTER .....	42
2.8	ROW-COLUMN TRANSFORMATIONS .....	42
2.8.1	UNSTACK .....	43
2.8.2	STACK .....	44
2.8.3	MELT AND PIVOT.....	45
2.8.4	DUMMY VARIABLES .....	47
2.9	COMBINING DATAFRAMES.....	47
2.9.1	MERGE .....	48
2.9.2	CONCATENATION .....	50
2.10	MISC FUNCTION.....	51
2.10.1	SAMPLE .....	51
2.10.2	ISIN .....	52
2.10.3	DROP_DUPLICATES .....	52

2.10.4	CUT.....	52
BAB III VISUALISASI DATA .....		54
A.	Pendahuluan .....	54
B.	Tujuan dan Capaian.....	54
3.1	VISUALIZING NUMERICAL CATEGORICAL DATA .....	55
3.1.1	VISUALIZING STATISTICAL RELATIONSHIPS .....	56
3.1.2	CATEGORICAL PLOT .....	58
3.1.3	VISUALIZING DISTRIBUTION.....	62
BAB IV STUDI KASUS .....		66
A.	Pendahuluan .....	66
B.	Tujuan dan Capaian.....	66
C.	Uraian Penelitian.....	66
D.	Latihan .....	67
4.1	PREDIKSI KALIMAT PESAN MENGGUNAKAN NAÏVE BAYES.....	67
4.2	Rangkuman .....	72
E.	DAFTAR PUSTAKA .....	73
TENTANG PENULIS .....		75

# BAB I

## Pengenalan Python

---

### A. Pendahuluan

Pengertian Python (bahasa pemrograman) merupakan bahasa pemrograman tinggi yang bisa melakukan eksekusi sejumlah instruksi multi guna secara langsung (interpretatif) dengan metode Object Oriented Programming dan juga menggunakan semantik dinamis untuk memberikan tingkat keterbacaan syntax. Sebagai bahasa pemrograman tinggi, python dapat dipelajari dengan mudah karena telah dilengkapi dengan manajemen memori otomatis (Ilham, 2020).

Mengapa saat ini disebut masa tepat untuk belajar Python? Karena Python dianggap memiliki kelebihan untuk melakukan pembuatan aplikasi-aplikasi kekinian yang mengandung kata kunci big data, data mining, deep learning, data science, hingga machine learning. Dengan kata lain, Python adalah Bahasa pemrograman simpel untuk pembuatan aplikasi berbasis kecerdasan buatan (artificial intelligence). Python secara umum berbentuk pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Istilah lainnya, Bahasa pemrograman multi-paradigma.

### B. Tujuan dan Capaian

Tujuan utama dari mempelajari Python adalah untuk memahami cara kerjanya dan membuat aplikasi atau solusi teknologi. Python dikenal sebagai bahasa yang mudah dipelajari dan digunakan, serta memiliki banyak library dan fitur yang dapat membantu mempermudah pembuatan aplikasi di berbagai bidang, seperti data science, machine learning, web development, dan lain-lain. Dengan mempelajari Python, kita dapat meningkatkan kemampuan dan keterampilan kita di bidang teknologi, yang dapat membuka peluang karir yang lebih baik. Pencapaian yang diharapkan setelah belajar bahasa pemrograman Python adalah:

- Mampu membuat program dan solusi sederhana atau rumit sesuai kebutuhan.
- Mampu memahami dan menggunakan library dan fitur Python untuk mempermudah pembuatan aplikasi.



- Mampu memecahkan masalah dan membuat solusi dengan logika dan algoritma yang baik.
- Memiliki pengetahuan dasar yang solid dan siap untuk terus belajar dan mengembangkan kemampuan dalam bidang teknologi.
- Dengan demikian, belajar bahasa pemrograman Python dapat membantu membuka peluang karier yang lebih baik dalam bidang teknologi.

### C. Uraian materi

Python merupakan bahasa pemrograman komputer, sama halnya dengan bahasa C, C++, Pascal, Java dan lain-lain. Python disusun oleh Guido Van Rossum pada tahun 1989 di Centrum Wiskunde & Informatica (CWI) Amsterdam Belanda. Publikasi pertama tahun 1991 dengan label versi 0.9.0, yang diikuti dengan versi 1.0 pada tahun 1994. Setiap bahasa pemrograman memiliki kosakata dan aturan-aturan yang berbeda. Sebagai alternative dari sekian banyak Bahasa pemrograman, Python banyak digunakan untuk melakukan administrasi system operasi dan jaringan computer, pengembangan aplikasi desktop maupun web, pengolahan data, dan pembuatan program antar muka ke perangkat keras dan mikrokontroler. Keunggulan Python (Raharjo, 2019):

- Memiliki konsep desain yang bagus dan sederhana. Kode Python mudah dibaca, digunakan ulang dan dirawat.
- Mendukung pemrograman berorientasi objek dan pemrograman fungsional
- Untuk memperoleh hasil yang sama, kode Python lebih sedikit dibandingkan dengan kode yang ditulis dalam bahasa lain sehingga dapat menghemat waktu para programmer. Contoh berikut memperlihatkan perbandingan penulisan kode program dalam Python, C dan C++.
- Program yang ditulis dalam Python dapat dijalankan di hampir semua system operasi, termasuk untuk perangkat mobile.
- Bersifat gratis dan open source.

<pre># Code Python Print("Halo ini Zaky") ←=====→ #Code-C-include &lt;stdio.h&gt; int main(){     print ("hello world!");     return 0;</pre>
---

```

}
←=====→
/* Code C++ */
#include <iostream>
int main(){
    std::cout<<"hello world!";
    return 0;
}

```

## 1.1 Instalasi Python

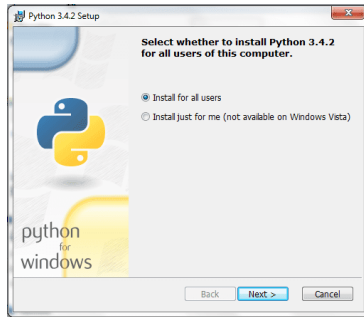
Langkah pertama untuk memulai pemrograman Python adalah dengan mengunduh terlebih dahulu aplikasi Python pada situs resmi <https://www.python.org/downloads/>. Python yang digunakan pada modul ini adalah versi 3.10.8. Setelah selesai diunduh, install aplikasi Python pada computer anda, dengan cara double klik pada file python-3.8.5 di Windows Explorer seperti terlihat pada Gambar dibawah ini. Selanjutnya ikuti instruksi yang ada pada proses instalasi sampai selesai. Setelah selesai maka aplikasi siap digunakan. Pada website python.org dapat ditemukan versi terbaru sesuai dengan OS laptop/PC yang dimiliki.



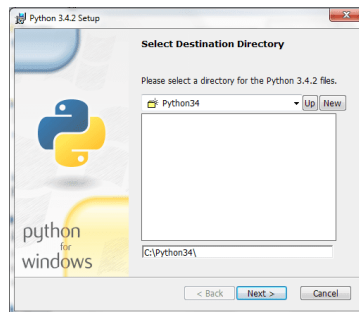
File yang telah diunduh kemudian dibuka untuk melanjutkan ke tahap selanjutnya.



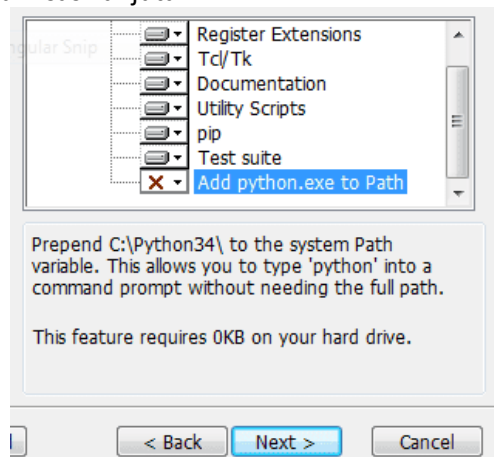
Installation Type: Install for all users agar bisa dipakai untuk semua user di komputernya.



Lokasi instalasi, disini saya menggunakan lokasi installasi default ya pada proses peng installasian ini.



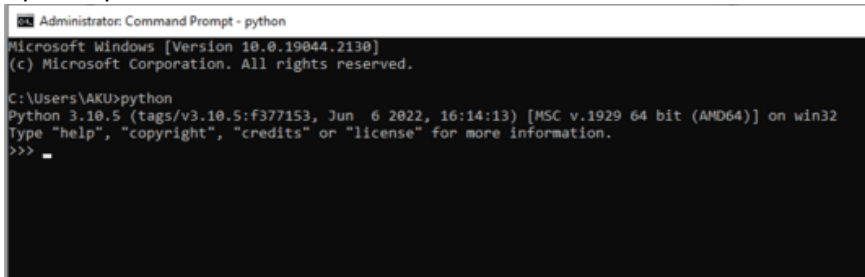
Opsi instalasi Kostumisasi lanjutan.



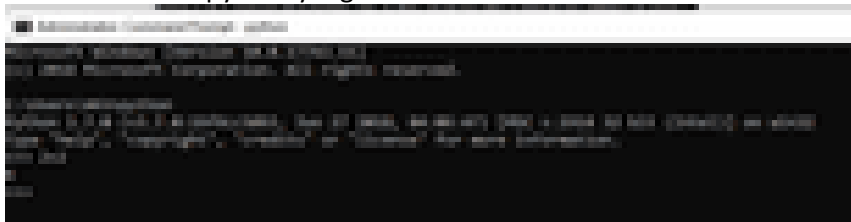
Instalasi selesai.

## 1.2 Python Interpreter

Python Interpreter untuk lingkungan Windows menggunakan program IDLE. Saat dijalankan Python Interpreter akan memunculkan window Python Shell dengan tanda prompt `>>>`. Pada bagian inilah kita dapat menuliskan perintah-perintah ke dalam interpreter. Gambar 2.3 memperlihatkan tampilan Python Interpreter pada Windows.



Untuk memastikan instalasi python berhasil, silahkan buka terminal / command prompt. Kemudian masukan perintah “python -V”. Jika python berhasil diinstal, maka akan ada versi python yang diinstal.



`>>> 2+2`

merupakan perintah untuk menghitung operasi pengurangan tersebut. Sehingga setelah kita menekan enter, maka baris berikutnya akan muncul hasil perhitungan yang dilakukan oleh mesin. Jadi perintah ditulis pada prompt `>>>`, hasilnya akan muncul pada baris tanpa prompt `>>>`. Jika perintah terlalu panjang tidak dapat ditulis dalam satu baris, yaitu perintah untuk menghitung `2+2`, ditulis dalam dua baris. Sedangkan baris ketiga merupakan hasil perhitungan yang dilakukan computer.

## D. Latihan

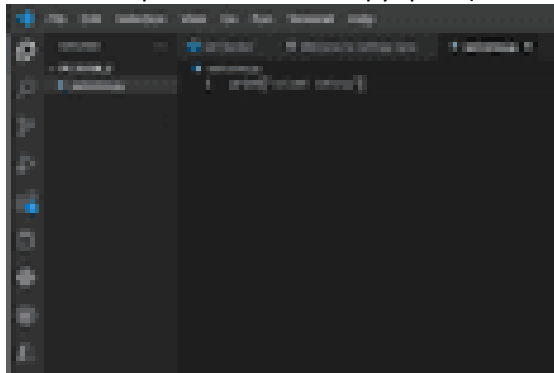
Visual Studio Code adalah salah satu editor teks paling populer dalam beberapa tahun terakhir. Text Editor open-source besutan Microsoft ini berhasil menarik perhatian developer dari berbagai bidang, mulai dari web development, desktop, android, dan lain sebagainya.

Hal ini tidak mengherankan mengingat fitur-fitur VSCode sangat membantu

proses coding, ditambah dukungan dari marketplace dengan banyak ekstensi yang siap memudahkan hidup kita sebagai developer.

Python memiliki sebuah fungsi bernama `print()`. Fungsi `print` akan menuliskan sebuah pesan di console. Biasanya fungsi `print` digunakan untuk kebutuhan debugging atau testing.

1. Pertama buatlah folder bernama `python_tutorial`
2. Buka folder tersebut pada Visual Studio Code (VSCode)
3. Buatlah file bernama `welcome.py` dengan klik kanan, pilih new file.
4. Masukkan kode berikut pada file `welcome.py`: `print("selamat datang")`



Fungsi `print` di Python adalah fungsi yang digunakan untuk memunculkan keluaran yang ingin kita `print` di konsol. Fungsi `print` terlihat sangat sederhana tetapi sebenarnya `print` adalah fungsi yang paling banyak digunakan dalam sintaks Python.

5. Jika sudah, coba jalankan dengan perintah `"python welcome.py"` pada terminal.



6. Jika berhasil, maka anda dapat melihat pesan `"welcome to python"` pada terminal anda.



Jika kode sudah di save lalu anda menjalankan kode tersebut namun pada hasilnya gagal mungkin saja terdapat kendala pada environment python nya pada saat instalasi.

### 1.3 VARIABLE

Pada bahasa pemrograman, variabel merupakan tempat yang digunakan untuk menampung data di memori yang mempunyai nilai yang dapat berubah-ubah selama proses program. Sesuai dengan namanya, isi dari variabel bisa berubah dari waktu ke waktu sesuai kebutuhan.

1. Buat file bernama variables.py
2. Untuk mendeklarasikan variabel pada python cukup dengan seperti berikut:

```
amount=30
```

Kode diatas berarti variabel amount memiliki nilai 30.

3. Tambahkan kode diatas dengan kode berikut

```
amount=30
data("{} is high".format(amount))
print(data)
```

Variabel data merupakan formatting string. Sehingga nantinya isi dari kurung kurawal ({} ) merupakan variabel amount. Fungsi format() berfungsi untuk melakukan pengaturan format string yang akan dicetak atau ditampilkan ke monitor.

4. Jalankan file variables.py menggunakan perintah “python variables.py” pada Terminal



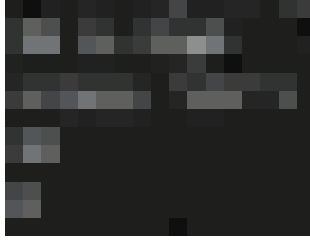
5. Untuk mengetahui tipe data dari variabel data, gunakan fungsi type() seperti berikut:

```
print(type(data))
```

6. Untuk mengetahui panjang dari variabel data, gunakan fungsi len() seperti berikut:

```
print(len(data))
```

7. Untuk mengambil data spesifik dari variabel data, dapat dilakukan seperti berikut:
8. Jalankan lagi variables.py pada terminal, maka hasilnya akan seperti berikut:



30 is high merupakan nilai dari variabel data <class 'str'> merupakan tipe data dari variabel data, yaitu string. Sederhananya, string merupakan teks. Nilai pada tipe data string berada diantara tanda petik. 10 merupakan panjang dari variabel data 0 merupakan karakter yang memiliki indeks ke 1 pada variabel data

#### 1.4 TUPLE DAN LIST

Tuple adalah urutan objek Python yang tidak dapat diubah. Tuple adalah urutan, seperti daftar. Perbedaan utama antara tupel dan daftar adalah bahwa tupel tidak dapat diubah, tidak seperti Daftar Python. Tuple menggunakan tanda kurung, sedangkan Daftar Python menggunakan tanda kurung siku.

Membuat tuple semudah memasukkan nilai yang dipisahkan koma. Secara opsional, Anda juga dapat memasukkan nilai yang dipisahkan koma ini di antara tanda kurung. Sebagai contoh:

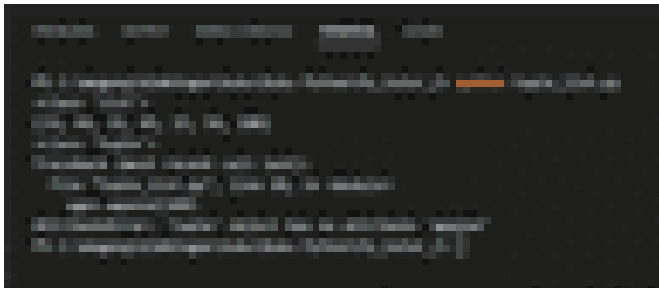
1. Buat file bernama tuple\_and\_list.py
2. Masukkan kode berikut:

```
#List
ages=[44,33,45,33,54]
print(type(ages))
ages.append(100)
ages.insert(0,33)
print(ages)
#Tuples
ages=(44,33,45,33,54)
print(type(ages))
ages.append(100)
ages.insert(0,33)
print(ages)
```

Fungsi type() berfungsi untuk mengetahui jenis atau tipe data dari suatu objek yang menjadi argumennya. Fungsi append ini berguna untuk menambahkan nilai array pada urutan terakhir. Fungsi untuk menyisipkan data baru di tengah array

list adalah fungsi insert() Ketika nilai dari variabel panjang ditampilkan dengan fungsi print().

3. Jalankan kode dengan perintah `py tuple_and_list.py` pada terminal



Pada list, kita telah berhasil menambahkan nilai 100 ke indeks terakhir dengan menggunakan fungsi `append()`. Juga berhasil mengubah indeks ke 0 menjadi bernilai 33. Sedangkan pada tuple, hal tersebut gagal dilakukan dan justru terjadi error seperti pada gambar diatas. Hal ini terjadi karena list yang bersifat mutable yang berarti dapat diubah elemennya, sedangkan tuple bersifat immutable yang berarti tidak dapat diubah elemennya

## 1.5 DICTIONARY

Pada python, dictionary dapat dibuat dengan menempatkan urutan elemen di dalam kurung kurawal {}, dipisahkan dengan 'koma'. Dictionary menyimpan pasangan nilai, satu menjadi key dan pasangannya menjadi value. Value dalam dictionary dapat berupa tipe data apa pun dan dapat digandakan, sedangkan key tidak dapat diulang dan harus tidak dapat diubah.

1. Buat file bernama `dictionary.py`
2. Masukkan kode berikut:

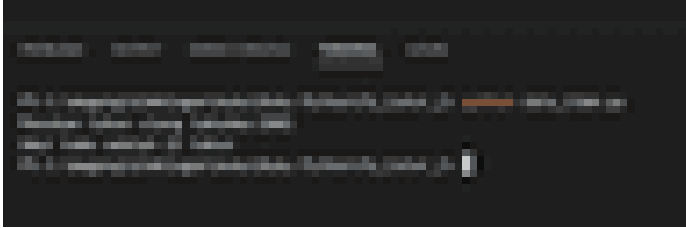
```
student={
    'name':"hussein alrubaye",
    'age':27,
    'salary':232.5
}
student['name']="Hussein Ahmed"
student['dept']="software engineer"
8
print(student,type(student))
```

3. Jalankan `dictionary.py` pada terminal





### 3. Jalankan my\_age.py pada terminal



Pada variabel `year_birth`, nilainya merupakan masukan dari user. Pada variabel `year_now`, nilainya merupakan hasil return dari fungsi `now()` pada `datetime` kemudian hanya diambil tahunnya saja. Untuk `my_age` merupakan hasil pengurangan dari variabel `year_now` dikurangi `year_birth`.

## 1.7 CONDITIONAL

Pernyataan bersyarat adalah alur yang mengontrol kode program Python berdasarkan pengujian pernyataan bersyarat. Seperti bahasa pemrograman lainnya, pernyataan bersyarat adalah salah satu aliran kontrol dalam bahasa pemrograman Python. Pernyataan bersyarat dapat berisi fungsi, operator matematika, dan operator logika. Sintaks yang digunakan untuk membuat control flow dalam bentuk pernyataan kondisional di Python adalah `if`, `elif`, dll. Jika Anda sudah familiar dengan materi logika matematika, Anda pasti pernah mendengar logika implikasi yang berbunyi `œif.....then....`. Begitu juga dengan bahasa pemrograman Python, sintaks `else if` atau `elif` merupakan cabang yang berasal dari jalur percabangan. Layaknya logika berpikir manusia dalam memikirkan sesuatu, sebuah program juga memiliki kesamaan. Perulangan atau percabangan dapat membuat program berfikir untuk melakukan sesuatu sesuai dengan kondisi yang diberikan.

### 1.7.1 IF CONDITIONAL

Python hanya mengetahui satu fungsi cabang (kondisi). Tidak ada sakelar atau kasing di python, hanya jika berfungsi. Pengambilan keputusan (jika kondisi) digunakan untuk mengantisipasi kondisi yang terjadi selama berjalannya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi tersebut. Jika tidak sesuai, berarti kondisi tersebut merupakan kondisi lain atau setelah kondisi umum. Kurang lebih seperti itu perumpamaan. Jika itu berarti ada keputusan yang harus diambil dari dua pilihan.

1. Buat file bernama `conditional_if.py`
2. Masukan kode berikut:

```
age=input("Masukan umur anda:")
if(int(age)>18):
```

```
print("welcome")
```

Statement pada bagian if diisi dengan variabel age lebih dari 18. Maka, jika user menginputkan umur dibawah 18, python tidak akan melanjutkan programnya. Sedangkan jika user menginput umur diatas 18, maka python akan melanjutkan program, dimana pada kode diatas python akan mencetak welcome.

3. Jalankan conditional\_if.py di terminal



Disini kita bisa lihat bahwa Ketika program di run terdapat hasil yaitu masukan umur kita yang akan kita masukan lalu setelah kita masukan maka akan memunculkan kalimat welcome

### 1.7.2 IF ELSE CONDITIONAL

Operasi pertama adalah operasi if else. Pada prinsipnya kondisi if else ini sebenarnya merupakan tambahan atau modifikasi tambahan dari kondisi if yang sudah ada. Blok program if akan tetap dijalankan ketika kondisi terpenuhi. Biasanya ditandai dengan kondisi True. Namun, tidak hanya jika tetapi ada tambahan lainnya. Selain kondisi tersebut, kode program yang dijalankan akan salah pada bagian else.

1. Buat file bernama conditional\_if\_else.py
2. Masukan kode berikut:

```
age=input("Masukan umur anda:")
if(int(age)>18):
    print("welcome")
else:
    print("Not Welcome")
```

Statement pada bagian if diisi dengan variabel age lebih dari 18. Maka jika user menginput umur diatas 18, maka python akan melanjutkan program, dimana pada kode diatas python akan mencetak welcome. Dan jika user menginputkan selain kurang dari 18, maka python akan melanjutkan program ke bagian else.

3. Jalankan conditional\_if\_else.py di terminal visual studio code.u



Disini kita bisa lihat bahwa Ketika program di run terdapat hasil yaitu masukan umur kita yang akan kita masukan lalu setelah kita masukan maka akan memunculkan kalimat welcome

### 1.7.3 IF ELIF CONDITIONAL

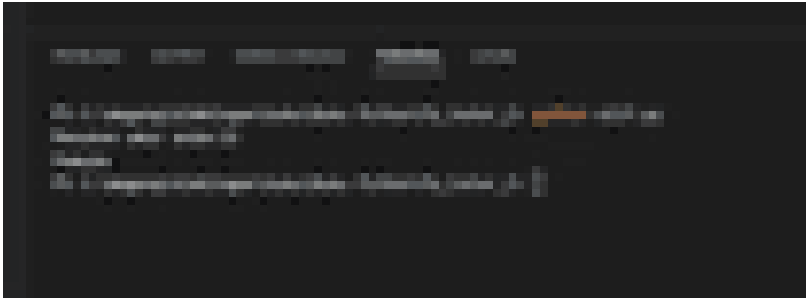
Elif berarti ketika kondisi lain tidak terpenuhi saat program sedang berjalan. Sedangkan else berlaku sebaliknya ketika tidak ada satu syarat pun yang terpenuhi. Selain percabangan, struktur ini disebut juga control flow, decision, condition structure, if structure, dll. Branching akan dapat membuat program berpikir dan menentukan tindakan sesuai dengan logika/kondisi yang kita berikan.

1. Buat file bernama `conditional_if_elif.py`
2. Masukan kode berikut:

```
age=input("Masukan umur anda:")
if(int(age)>=8 and int(age)<=10):
    print("Balita")
elif(int(age)>=11 and int(age)<=15):
    print("Anak anak")
elif(int(age)>=16 and int(age)<=18):
    print("Remaja")
elif(int(age)>=19 and int(age)<=30):
    print("Pemuda")
else:
    print("Tua")
```

Pada dasarnya, if elif conditional ini hampir sama dengan if else conditional. Akan tetapi, pada if elif conditional python akan melakukan pengecekan terhadap elif terlebih dahulu sebelum akhirnya dikembalikan pada else.

3. Jalankan `conditional_if_elif.py` di terminal visual studio code.



Disini kita bisa lihat bahwa Ketika program di run terdapat hasil yaitu masukan umur kita yang akan kita masukan lalu setelah kita masukan maka akan memunculkan kalimat pemuda dikarenakan logic padaumur saya kurang dari 30 tergolong pemuda

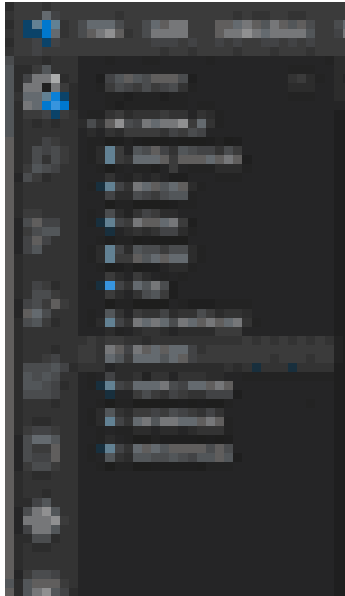
#### 1.7.4 READ WRITE FILE

1. Buat file bernama read\_write\_file.py
2. Masukan kode berikut:

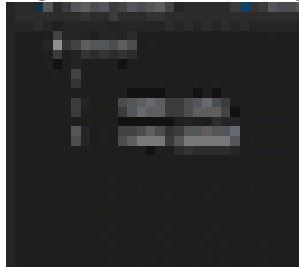
```
out=open("test.txt","a")
out.write(" \nname:zaky")
out.write(" \nname:yusuf")
out.close()
```

Kode diatas bertujuan untuk membuka test.txt, kemudian menambahkan isi pada test.txt sesuai pada fungsi write().

3. Ketika read\_write\_file.py dijalankan pada terminal, maka akan ada file baru bernama test.txt



4. Berikut adalah isi dari test.txt dari hasil percobaan pada kode di atas untuk point nomor 2 ketika program ini di run atau dijalankan.



Setelah kita melakukan pengkodean pada baris yang telah di lakukan pada gambar ataupun teks script sebelumnya maka ini lah hasilnya dari apa yang kita buat jika masih ada error tidak muncul harap cek lagi pada spasi nya.

## 1.8 OBJECT ORIENTED PROGRAMMING

Object Oriented Programming adalah paradigma pemrograman yang berpaku pada konsep kelas dan objek. Object Oriented Programming digunakan untuk menyusun program perangkat lunak menjadi potongan-potongan blueprint kode yang sederhana dan dapat digunakan kembali (kelas) yang digunakan untuk membuat objek individual.

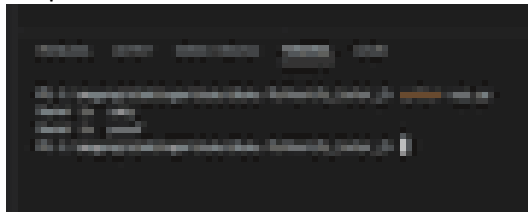
1. Buat file bernama oop.py
2. Masukkan kode berikut:

```
class Car:
```

```
def get_owner(self):
    print("Owner is ",self._Name)
def set_owner(self,Name):
    self._Name=Name
```

Kode diatas merupakan sebuah class bernama Car. Class Car memiliki instance function yaitu get\_owner dan set\_owner.

3. Tambahkan kode berikut: Variable my\_car dan rmd\_car merupakan sebuah object yang menggunakan class Car.
4. Jalankan oop.py dengan cara python oop.py pada terminal VSC maka hasilnya akan seperti ini.



```
python oop.py
Owner is  my
Owner is  rmd
Owner is  my
Owner is  rmd
```

Pada tahapan ini setelah kita lakukan run pada code diatas maka akan menampilkan owner nya disini kita masukan nama depan dan nama belakang sesuai pada run pertama dan kedua yang muncul.

## 1.9 DATABASE

Python Menggunakan Database SQLite3 - Pada postingan saya kali ini saya akan membahas sqlite3 di python, sebelum membahas cara menggunakan sqlite3 di python, kita harus mengetahui terlebih dahulu apa itu sqlite, adalah mesin database yang tidak terikat dengan server yang berdiri sendiri artinya berdiri sendiri tanpa ada pengaturan seperti kebanyakan database namun memiliki fungsi yang sama dengan database pada umumnya, perintah dasar seperti sql biasa tetap bisa dilakukan di sqlite. Contoh program sebagai berikut:

1. Buat file bernama database.py
2. Import library sqlite3 terlebih dahulu agar kita bisa terkoneksi dengan databasenya.

```
import sqlite3
```

3. Tambahkan kode berikut:

```
db=sqlite3.connect("information.db")
db.row_factory=sqlite3.Row
db.execute("create table if not exists admin(name text,age int)")
db.execute("insert into admin (name,age) values ( ?, ? )",("Hussein",26))
db.execute("insert into admin (name,age) values ( ?, ? )",("Jena",1))
db.commit()
```

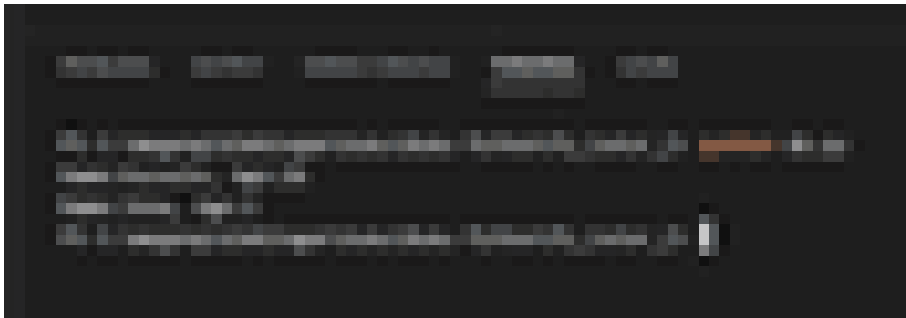
Kode diatas merupakan kode untuk membuat tabel admin pada database dengan field name dan age. Setelah membuat tabel, kemudian ditambahkan dua buah data.

4. Tambahkan kode berikut:

```
cursor=db.execute("select * from admin")
for row in cursor:
    print("Name:{}, Age:{}".format(row["name"],row["age"]))
```

Kode diatas merupakan kode untuk mengambil data pada tabel admin kemudian dilakukan iterasi menggunakan for dan dicetak ke console.

5. Jalankan database.py pada terminal, maka hasilnya akan menjadi seperti ini.



## 1.10 HTTP REQUEST

Permintaan adalah modul Python yang dapat Anda gunakan untuk mengirim berbagai permintaan HTTP. Permintaan adalah pustaka yang mudah digunakan dengan banyak fitur mulai dari meneruskan parameter di URL hingga mengirim tajuk khusus dan verifikasi SSL. Dalam tutorial ini, Anda akan belajar cara menggunakan pustaka ini untuk mengirim permintaan HTTP sederhana dengan Python. Contoh pada program sebagai berikut:

1. Buat file bernama http\_request.py
2. Import library yang dibutuhkan:

```
from urllib.request import urlopen
import json
```

3. Import library yang dibutuhkan:

```
url = "https://jsonplaceholder.typicode.com/todos/1"
with urlopen(url) as response:
    body = response.read()

todo_item = json.loads(body)
print(todo_item)
```



Kode diatas akan melakukan http request dengan method GET pada variabel url. Kemudian responsnya akan ditambahkan ke variabel body dan di parsing dengan module json.

4. Jalankan `http_request.py` pada terminal di VSC dan hasilnya menjadi seperti ini.

```
'userId': 1, 'id': 1, 'title': 'Sedectus aut autem', 'completed': False}
```

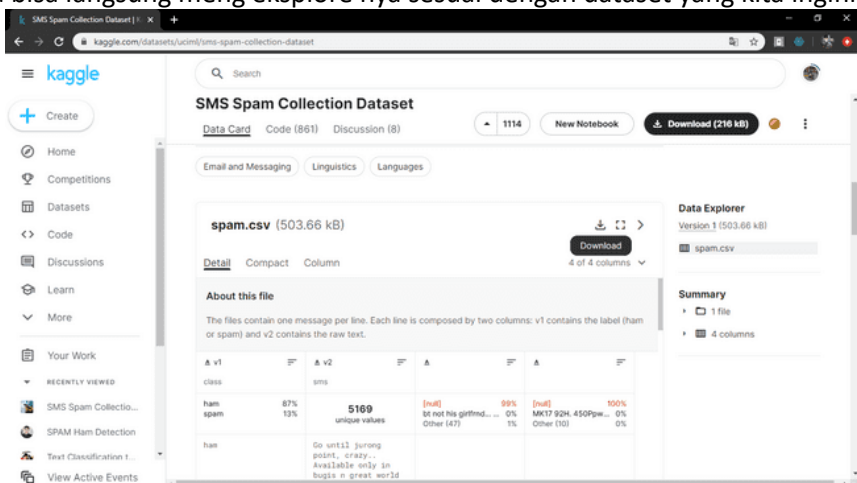
### 1.11 PYTEST

Pytest adalah pustaka yang sangat populer digunakan untuk pengujian unit dan fungsi untuk Bahasa Pemrograman Python.

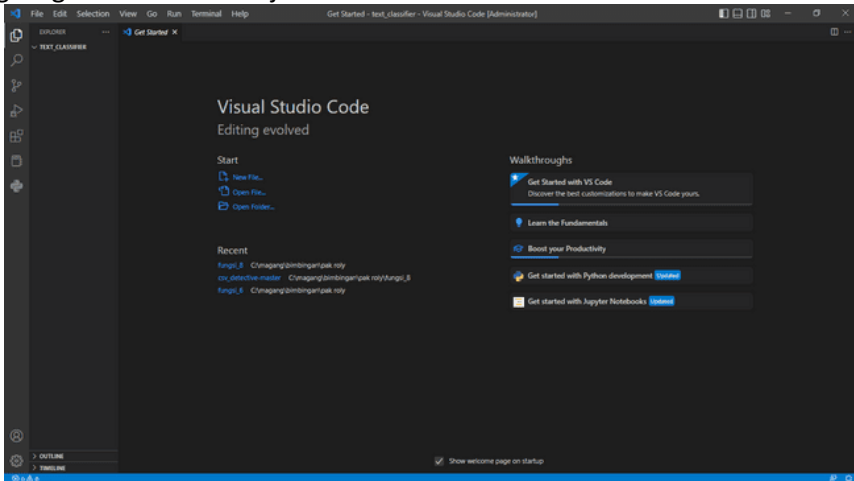
Memahami otomatisasi pengujian akan sangat membantu dalam membangun program bebas bug, dan ini merupakan keterampilan tambahan yang penting saat Anda melamar pekerjaan.



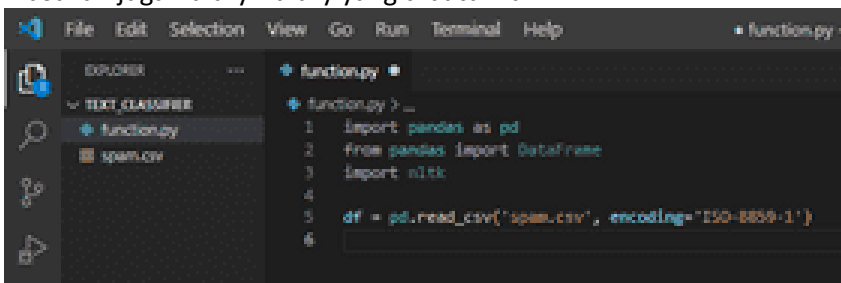
Pertama kita mencari datasetnya di google terlebih dahulu tentukan sesuai dengan apa yang kita butuhkan. Lalu setelah itu kita buka website nya kaggle dan kita bisa langsung meng eksplore nya sesuai dengan dataset yang kita inginkan.



Setelah ketemu dan kita lihat sesuai sama apa yang kita butuhkan maka langsung kita download saja. Setelah di download lalu kita buka Tools editor kita.



Sekarang kita bikin folder baru dan kita drag and drop di vsc maka otomatis akan langsung terbuka. Lalu kita buat file python nya dengan nama function.py menjadi seperti ini, setelah itu kita juga importkan dataset nya ke dalam folder kita. Lalu dari dataset yang kita download dari google tadi tidak lupa untuk kita masukan pada tools yang kita gunakan saat ini yaitu vsc(visual studio code). Lalu kita masukan juga library-library yang dibutuhkan.



Pandas merupakan library open source dengan Python yang sering digunakan untuk mengolah data yang meliputi pembersihan data, manipulasi data, hingga melakukan analisis data. Saat melakukan analisis, kami tidak dapat menggunakan data mentah.

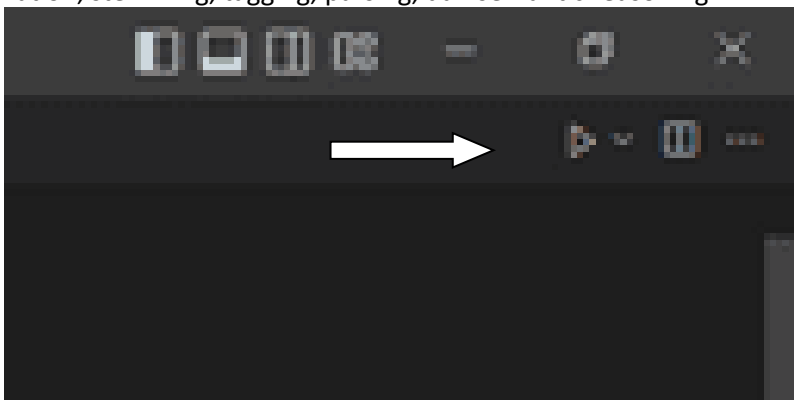
Data mentah harus diolah terlebih dahulu hingga layak untuk dianalisis. Tahap ini juga dikenal sebagai sengketa data. Perselisihan adalah proses dimana data dikelola dan dibentuk menjadi lebih terorganisir.

```

4
5 df = pd.read_csv('spam.csv', encoding='ISO-8859-1')
6
7 if __name__ == '__main__':
8     print(df)
9

```

NLTK adalah singkatan dari Natural Language Tool Kit, yang merupakan pustaka yang digunakan untuk membantu kita bekerja dengan teks. Library ini memudahkan kita untuk mengolah teks seperti melakukan classification, tokenization, stemming, tagging, parsing, dan semantic reasoning.



Disini program akan kita langsung jalan kan saja untuk melihat hasilnya

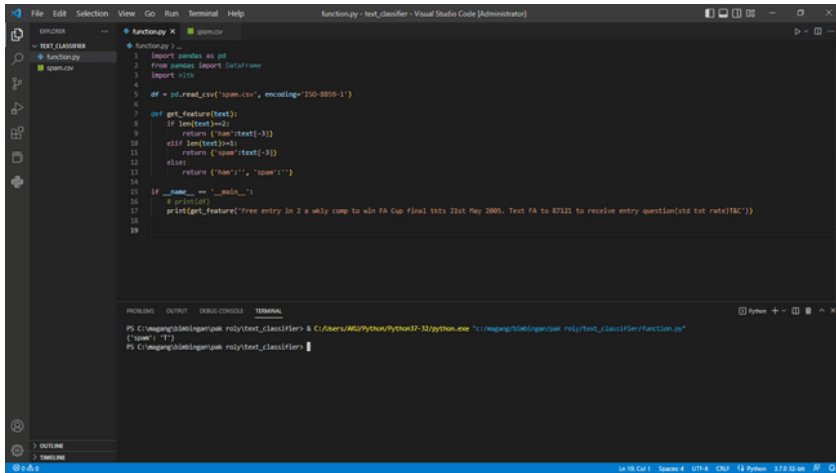
```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
v1 v2 Unnamed: 2 Unnamed: 3 Unnamed: 4
0 ham Go until Jurong point, crazy.. Available only ... NaN NaN NaN
1 ham Oh lar... Doking wif u oni... NaN NaN NaN
2 spam Free entry in 2 a wkly comp to win FA Cup fina... NaN NaN NaN
3 ham U dun say so early hor... U c already then say... NaN NaN NaN
4 ham Nah I don't think he goes to usf, he lives aro... NaN NaN NaN
... .. NaN NaN NaN
5567 spam This is the 2nd time we have tried 2 contact u... NaN NaN NaN
5568 ham Will i b going to esplanade fr home? NaN NaN NaN
5569 ham Pity, * was in mood for that. So...any other s... NaN NaN NaN
5570 ham The guy did some bitching but I acted like i'd... NaN NaN NaN
5571 ham Rofl. Its true to its name NaN NaN NaN

[5572 rows x 5 columns]
PS C:\magang\biabingan\pak roly\text_classifier>

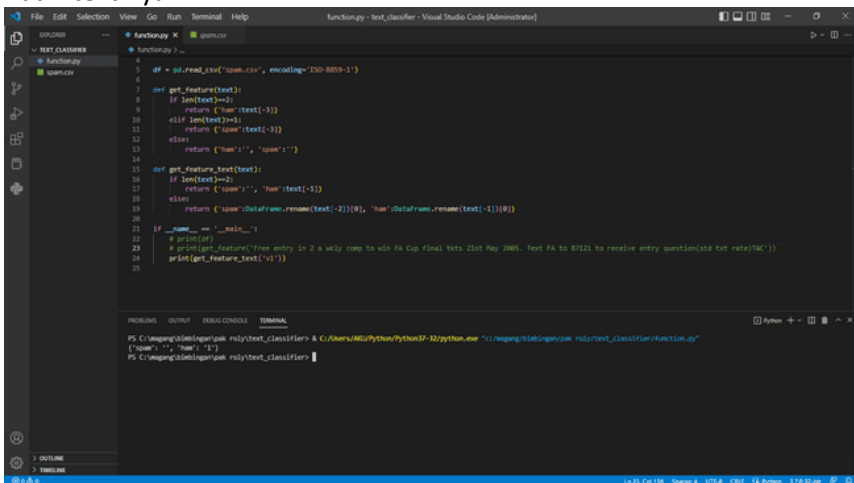
```

Dan ini hasil output dari dataset nya setelah kita print akan menjadi seperti ini disini saya memilih dataset dengan tipe v1 untuk tipe pada kalimatnya dan v2 untuk kalimat nya itu sendiri.



```
1 import pandas as pd
2 from pandas import DataFrame
3 import sys
4
5 df = pd.read_csv('spam.csv', encoding='ISO-8859-1')
6
7 def get_feature(text):
8     if len(text)>=1:
9         return 'ham' if text[0] == 'h' else 'spam'
10     elif len(text)>=1:
11         return 'spam' if text[0] == 's' else 'ham'
12     else:
13         return 'ham', 'spam'
14
15 if __name__ == '__main__':
16     # print(df)
17     print(get_feature('Free entry in 2 x wily comp to win FA Cup final this 21st May 2005. Text FA to 87121 to receive entry question(snd txt rate)13C'))
18
19
```

Sekarang kita buat kan functionnya, function ini untuk melakukan logic untuk melihat mana kalimat yang ber tipe spam dan juga mana kalimat yang bertipe ham dari text nya.



```
4
5 df = pd.read_csv('spam.csv', encoding='ISO-8859-1')
6
7 def get_feature(text):
8     if len(text)>=1:
9         return 'ham' if text[0] == 'h' else 'spam'
10     elif len(text)>=1:
11         return 'spam' if text[0] == 's' else 'ham'
12     else:
13         return 'ham', 'spam'
14
15 def get_feature_text(text):
16     if len(text)>=1:
17         return 'spam' if text[0] == 's' else 'ham'
18     else:
19         return 'spam' if df[df['text'] == text]['ham'].iloc[0] else 'ham'
20
21 if __name__ == '__main__':
22     # print(df)
23     # print(get_feature('Free entry in 2 x wily comp to win FA Cup final this 21st May 2005. Text FA to 87121 to receive entry question(snd txt rate)13C'))
24     print(get_feature_text('Free entry in 2 x wily comp to win FA Cup final this 21st May 2005. Text FA to 87121 to receive entry question(snd txt rate)13C'))
25
```

Sekarang kita buat kan functionnya, function ini untuk melakukan logic untuk melihat mana kalimat yang ber tipe spam dan juga mana kalimat yang bertipe ham dari text nya pada baris pertama di dataset.

```

23
24 def get_data(df, get_feature=get_feature):
25     features = []
26     for i, row in df.iterrows():
27         text = row["v1"]; type = row["v2"]
28         if isinstance(text, str):
29             if ' ' in text:
30                 text = text.replace(" ", "")
31             if '(' not in text:
32                 features.append((get_feature(text), type.strip('() ')))
33             else:
34                 text = text.partition('(')[0]
35                 features.append((get_feature(text), type.strip('() ')))
36     return features
37

```

pada baris fungsi ini untuk Fungsi isinstance() mengembalikan True jika objek yang ditentukan adalah tipe yang diambil dari df kemudian mengganti satu karakter dari string yang diberikan lalu di tambahkan elemen ke akhir spasi dan menghapus semua karakter awal (spasi di awal) dan karakter tambahan (spasi di akhir).

```

37
38 def get_train_test(features, ratio=0.9):
39
40     N = len(features)
41     T = int(N * ratio)
42     train = features[:T]
43     test = features[T:]
44     return train, test
45

```

Fungsi ini untuk membagi set data untuk melatih, ratio, menguji set data. Dataset perlu dikocok sebelum menggunakan fungsi.

```

46
47 def test_classifier(df, f=get_feature):
48     data = get_data(df, f)
49     train, test = get_train_test(data)
50     classifier = nltk.NaiveBayesClassifier.train(train)
51     acc = nltk.classify.accuracy(classifier, test)
52     return classifier, acc
53

```

Disini kita hanya memanggil dataset dan function get\_featire lalu kita lanjutkan dan gunakan. Kami menerapkan NAIVE BAYES CLASSIFIER ke dataset kami. Kami

pertama-tama membuat set data pelatihan untuk model kami, lalu kami membuat set data pengujian untuk mengujinya.

```

53 def show_type_of_text(text, texts=False, show_acc=False):
54     f = get_feature_text if texts else get_feature
55     classifier, acc = text_classifier(df, f)
56     if show_acc:
57         print(f'Accuracy: {acc:.4}')
58     clf = classifier.classify(f(text))
59     print(f'{text}: {clf}')
60     classifier.show_most_informative_features(10)
61
62

```

Dari hasil pemodelan dan klasifikasi yang kita buat disini ketika kita run maka akan keluar hasilnya untuk tipe spam. Kata kunci Python False mirip dengan kata kunci True, tetapi dengan nilai Boolean yang berlawanan dari false. Namun, jika kita menggunakan def show\_type\_of\_text(text, texts=False, show\_acc=False), tipe text dan isi item(kalimat) yang ditampilkan.

```

63
64 if __name__ == '__main__':
65     # print(df)
66     # print(get_feature("Free ent
67     # print(get_feature_text('v1
68     show_type_of_text("spam")

```

Sekarang kita akan mencoba untuk menjalankan pada tombol run.

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Python + -
PS C:\magang\blablabang\pak roly\text_classifier> & C:\Users\NBI\Python\Python37-32\python.exe "C:\magang\blablabang\pak roly\text_classifier\funcio
spam: Please call our customer service representative on FREEPHONE 0800 145 4742 between 9am-11pm as you have NOW a guaranteed £1000 cash or £500
0 prize!
Most Informative Features
spam = 'h'          Sorry, : II wal = 1.3 : 1.0
spam = 'p'          Please : se tri = 1.2 : 1.0
PS C:\magang\blablabang\pak roly\text_classifier>

```

Dan ini adalah hasilnya ketika kita jalankan, maka akan terlihat tipe kalimat nya berupa spam dan isi kalimat nya juga berupa kalimat spam

```

62 def give_type(type1='spam', type2='ham'):
63     data = get_data(df, get_feature)
64     classifier = nltk.NaiveBayesClassifier.train(data)
65     following = classifier.prob_classify({'spam':type2, 'ham':type1})
66     x = following.generate()
67     print(f'{type2}: {type1}{x}')
68
69

```

Pada baris ini diberikan penamaan type1 untuk spam dan type2 untuk ham pada parameter kemudian data yang di ambil dari function get\_data df, dan get\_feature lalu data akan di olah menggunakan library dari naive bayes lalu akan di labeling spam dan ham dan hasil nya akan di generate lalu di print.

```
69
70 if __name__ == '__main__':
71     # print(df)
72     # print(get_feature('Free entry in 2 a wkly comp to w
73     # print(get_feature_text('v1'))
74     # show_type_of_text("spam")
75     give_type(type1='Go until jurong point, crazy..')
```

Setelah itu coba kita masukan kalimat ham pada type1 yang berada di dataset disini saya memasukan kalimat depan nya saja.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\vagus\blabbing\pak roly\text_classifier> & C:\Users\ABU\Python\Python37-32\python.exe "c:\vagus\blabbing\pak roly\text_classifier\function.py"
ham: Go until jurong point, crazy..Of course I don't tease me ... You know I slepy must see ! "grins" ... Do keep me posted my prey ... "loving smile" "devou
ring kiss"
PS C:\vagus\blabbing\pak roly\text_classifier> []
```

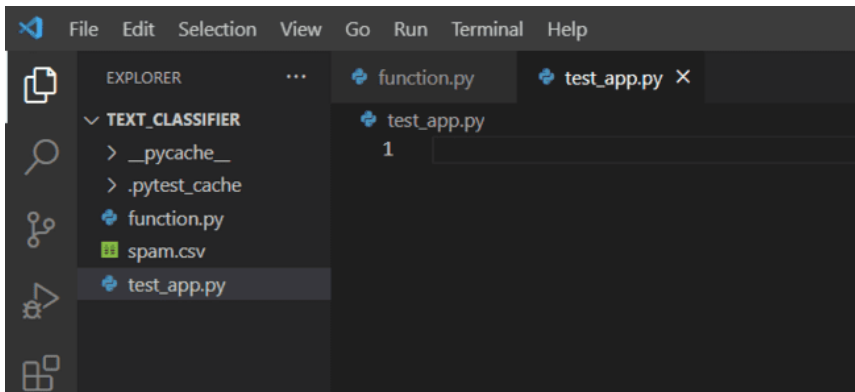
Nah ini hasilnya ketika di jalankan maka kalimat yang kita masukan di bagian depan saja maka akan keluar full pada baris kalimatnya.

```
76
77 if __name__ == '__main__':
78     print('-wait a minute-')
79     show_type_of_text("spam")
80     print('Take a name: (give text and first word)')
81     give_type(type1='Go until jurong point, crazy..')
```

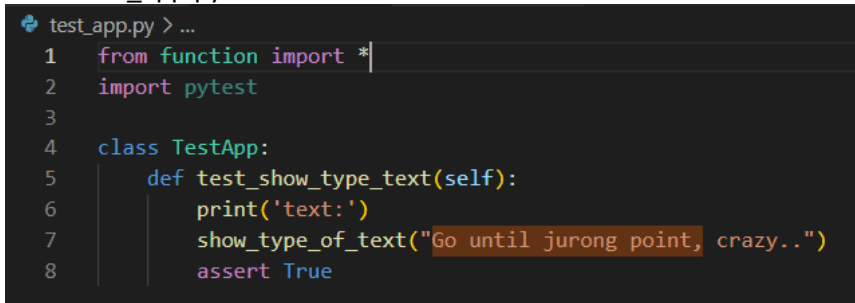
Sekarang pada bagian eksekusi ini kita ganti menjadi seperti ini untuk melihat hasil output secara keseluruhan dari program yang kita buat

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\vagus\blabbing\pak roly\text_classifier> & C:\Users\ABU\Python\Python37-32\python.exe "c:\vagus\blabbing\pak roly\text_classifier\function.py"
-wait a minute-
spam Please call our customer service representative on FREEDOME 8000 145 4742 between 9am-5pm as you have WON a guaranteed $1600 cash or $5000 prize!
Most Informative Features
spam = 'h'          Sorry, I'll wait -      1.3 : 1.0
spam = 'p'          Please, we tri -    1.2 : 1.0
Take a name: (give text and first word)
ham: Go until jurong point, crazy..This pen thing is beyond a joke. Went a Biro do? Don't do a masters as can't do this ever again!
PS C:\vagus\blabbing\pak roly\text_classifier> []
```

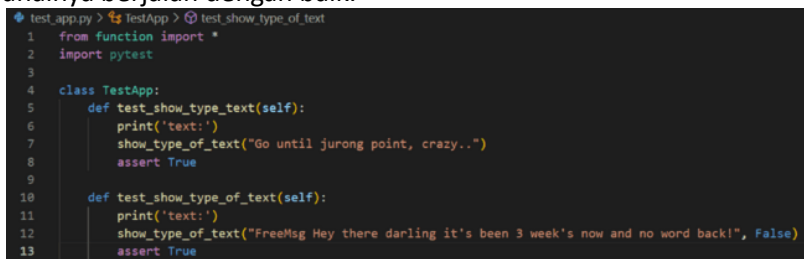
Dan ini adalah hasil output secara keseluruhan dari program yang kita buat ketika kita run, Sekarang kita akan membuat file test nya dengan menambahkan file .py baru



Kita namakan dengan test\_app.py disini kita akan membuat file testnya itu sendiri dari hasil kode yang kita buat. Pertama kita importkan terlebih dahulu dari file function dan juga import pytest agar command perintah kita menjadi "pytest -v test\_app.py".



Lalu disini kita buat class TestApp, lalu kita buat function sesuai dengan apa yang kita ingin test disini saya akan mengetest function dari show\_type\_text() dengan memasukan kalimat yang ada di dataset. Lalu kita run dengan pytest -v test\_app.py lalu enter, dan terdapat hasil dari kode yang dibuat bahwahnya berjalan dengan baik.



Lalu di function ini juga sama cuman bedanya untuk membaca tipe pada text nya. Lalu ini kita run dan ini adalah hasilnya maka berjalan dengan baik juga.



```

test_app.py > TestApp > test_type
1 from function import *
2 import pytest
3
4 class TestApp:
5     def test_show_type_text(self):
6         print('text:')
7         show_type_of_text("Go until jurong point, crazy..")
8         assert True
9
10    def test_show_type_of_text(self):
11        print('text:')
12        show_type_of_text("FreeMsg Hey there darling it's been 3 week's now and no word back!", False)
13        assert True
14
15    def test_type(self):
16        print('text: (give text and first word)')
17        give_type(type1='Yeah he got in at 2 and was v apologetic. ')
18        assert True

```

Disini saya akan mengetest give\_type dengan memasukan type1 dari label spam dengan kalimat yang bertipe kan ham.

```

cachedir: .pytest_cache
rootdir: C:\magang\bimbingan\pak roly\text_classifier
collected 3 items

test_app.py::TestApp::test_show_type_text PASSED [ 33%]
test_app.py::TestApp::test_show_type_of_text PASSED [ 66%]
test_app.py::TestApp::test_type PASSED [100%]

===== 3 passed in 23.09s =====
PS C:\magang\bimbingan\pak roly\text_classifier>

```

Ketika kita sudah membuat kode pada program tersebut dan kita jalankan maka akan keluar hasilnya disini saya membuat 3 testing yaitu test\_show\_type\_text, test\_show\_type\_of\_text, dan test\_type sudah bisa dilihat bahwalnya semua nya jalan passed.

# BAB II

## DATA MINING

---

### A. Pendahuluan

Penambangan data adalah cabang ilmu komputer yang cukup memadai banyak digunakan dan dipelajari oleh para ahli ilmu komputer dan programmer. Penambangan data adalah konsep yang rusak untuk menemukan pengetahuan atau informasi berharga yang tersembunyi di dalamnya dalam database. Penambangan data adalah proses semi-otomatis menerapkan matematika, teknik statistik, pembelajaran mesin, dan kecerdasan buatan untuk menguraikan dan mengidentifikasi informasi pengetahuan konten potensial dan bermanfaat yang terkandung dalam database besar.

Teknik data mining secara garis besar dapat dibagi menjadi dua kelompok: verifikasi dan penemuan. Metode verifikasi umumnya meliputi teknik statistik seperti fit dan analisis varians. Metode Discovery sendiri dapat memiliki dua model yaitu model prediktif dan model deskriptif. Teknik prediktif membuat prediksi pada data dengan menggunakan hasil yang diketahui dari data yang berbeda. Model ini dapat digunakan dengan data historis lainnya. Sedangkan Teknik Deskriptif bertujuan untuk mengidentifikasi pola atau hubungan antara data dan menyediakan cara untuk mengeksplorasi karakteristik data tersebut diselidiki.

Data mining sendiri memiliki beberapa teknik untuk menemukan pola atau informasi tersembunyi, salah satunya banyak digunakan dipenelitian sebagian besar teknik cluster. teknik cluster sendiri adalah teknik yang tidak menggunakan parameter atau disebut juga non parametrik dan diterapkan pada kasus nyata. Untuk Untuk mengimplementasikan teknik ini, kita membutuhkan algoritma kerja, data mining juga memiliki beberapa algoritma tetapi yang paling sederhana dan sering digunakan adalah algoritma k-means, algoritma ini sendiri bertujuan untuk mengelompokkan objek ke dalam cluster atau kelompok yang sudah ada bertekad.

### B. Tujuan dan capaian

Tujuan yang ingin dicapai setelah pembelajaran ini adalah.

1. Diberikan suatu program yang berkaitan dengan Data Mining, mahasiswa dapat menentukan output program tersebut secara otomatis dengan benar.
2. Diberikan suatu masalah, mahasiswa dapat menghasilkan program interaktif dalam Python Data Mining untuk menyelesaikan masalah tersebut dengan melibatkan struktur program dan algoritma secara terstruktur.
3. Diberikan suatu masalah, mahasiswa dapat menghasilkan program penyelesaian masalah pada Big Data dari yang didapat, diproses dan hingga menghasilkan output yang dapat berguna dalam menyelesaikan masalah.

### **C. Uraian Materi**

Sejauh ini, program yang ditulis mengikuti skema sederhana seperti berikut:

- Kumpulkan masukan dari pengguna.
- Lakukan satu atau lebih perhitungan.
- Menampilkan hasil di layar.

Pada kode program yang disusun seperti ini, pernyataan-pernyataan akan dieksekusi secara berurutan, satu demi satu, tanpa bercabang ke arah lain. Struktur program yang seperti ini disebut struktur sekuensial. Banyak algoritma yang memerlukan program untuk menjalankan beberapa pernyataan hanya dalam keadaan tertentu. Hal ini disebut dengan struktur desisi/ keputusan.

#### **2.1 Eksekusi Bersyarat**

Eksekusi bersyarat merupakan bentuk struktur keputusan yang paling sederhana, dimana tindakan atau serangkaian tindakan tertentu hanya dilakukan ketika kondisi tertentu dipenuhi. Sedangkan jika kondisinya tidak dipenuhi, maka tindakan tidak dilakukan. Bagan pada Gambar 4.1 menunjukkan gambaran logika dari eksekusi bersyarat. Simbol belah ketupat mewakili pertanyaan ya/ tidak atau kondisi benar/ salah. Jika jawaban untuk pertanyaannya adalah ya (atau jika kondisinya benar), aliran program mengikuti satu jalur yang mengarah ke tindakan yang dilakukan. Jika jawaban untuk pertanyaannya adalah tidak (atau kondisinya salah), aliran program mengikuti jalur lain yang mengabaikan tindakan.

```

age=input("Masukan umur anda:")
if(int(age)>=8 and int(age)<=10):
    print("Balita")
elif(int(age)>=11 and int(age)<=15):
    print("Anak-anak")

```

Struktur if dimulai dengan kata kunci if sampai statement n. Perhatikan bahwa pernyataan di dalam tubuh struktur if ditulis secara indent (menjorok ke dalam). Tanda titik-dua (:) ditempatkan setelah kondisi if dan sebelum pernyataan pertama dalam tubuh if. Struktur if berakhir pada statement n. Jadi terdapat empat hal penting yang harus diperhatikan:

1. Kata if, merupakan key word Python selalu ditulis dengan huruf kecil.
2. Key word if diikuti oleh kondisi yang harus diuji, `age=input("...")`. Kondisi dapat juga ditulis dalam tanda kurung tertutup.
4. Terdapat tanda titik-dua setelah kondisi
5. Semua statement yang harus dilakukan jika kondisi dipenuhi ditulis secara indent (satu kali tab) setelah tanda titik-dua.

## D. Latihan

### 2.2 Library

Seperti disebutkan sebelumnya, Python adalah salah satu bahasa pemrograman yang paling banyak digunakan. Python menawarkan peningkatan produktivitas karena tidak ada langkah kompilasi, dan siklus edit-test-debug sangat cepat.

Selain itu, Python juga bersifat interaktif, portabel, dan berorientasi objek. Bahasa pemrograman open source ini dapat berjalan di berbagai sistem, termasuk Linux, macOS, dan Windows. Python dapat digunakan dalam banyak hal seperti visi komputer, visualisasi data, pembelajaran mesin 3D, robotika, dan banyak lagi.

Python juga didukung oleh banyak library yang merupakan kode program tambahan yang digunakan untuk kebutuhan tertentu.

Pustaka Python ini sendiri merupakan kumpulan modul terkait yang berisi kumpulan kode yang dapat digunakan berulang kali di berbagai program. Pustaka ini membuat Pemrograman Python lebih sederhana dan lebih nyaman bagi pemrogram. Ini karena Anda tidak perlu menulis kode yang sama berulang kali untuk program yang berbeda. Pustaka Python memainkan peran yang sangat vital dalam bidang Pembelajaran Mesin, Ilmu Data, Visualisasi Data, dll

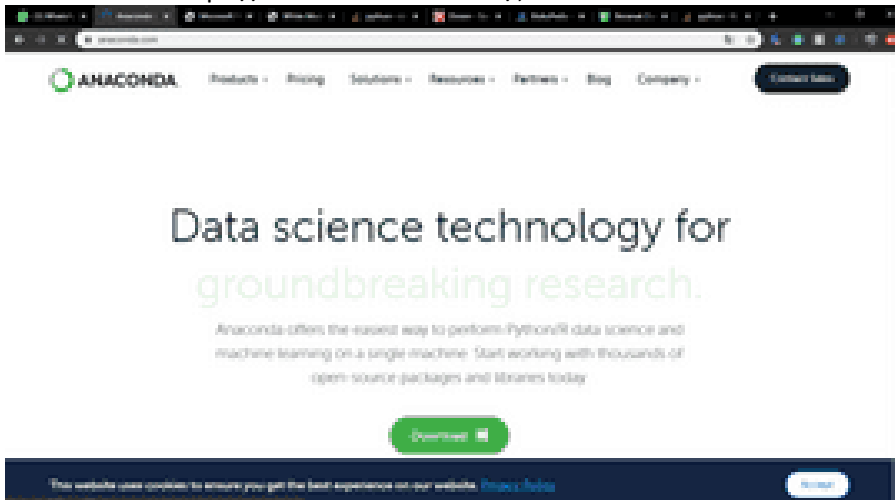
### 2.3 Apa Itu Pandas?

Pandas adalah open-source library yang dibuat terutama untuk bekerja dengan data relasional atau berlabel dengan mudah dan intuitif. Pandas menyediakan berbagai struktur data dan operasi untuk memanipulasi data numerik dan runtun waktu. Perpustakaan ini dibangun di atas perpustakaan NumPy. Pandas cepat dan memiliki kinerja & produktivitas tinggi bagi pengguna. Pandas bertujuan untuk menjadi blok bangunan dasar tingkat tinggi untuk melakukan analisis data dunia nyata yang praktis dengan Python. Selain itu, ia memiliki tujuan yang lebih luas untuk menjadi alat analisis / manipulasi data open source yang paling kuat dan fleksibel yang tersedia dalam bahasa apa pun.

### 2.4 Instalasi

Sebelum belajar penggunaan library pandas, Kita harus memastikan telah menginstal Jupyter Notebook. Jika teman-teman belum menginstal Jupyter Notebook, maka kalian bisa mengikuti langkah berikut:

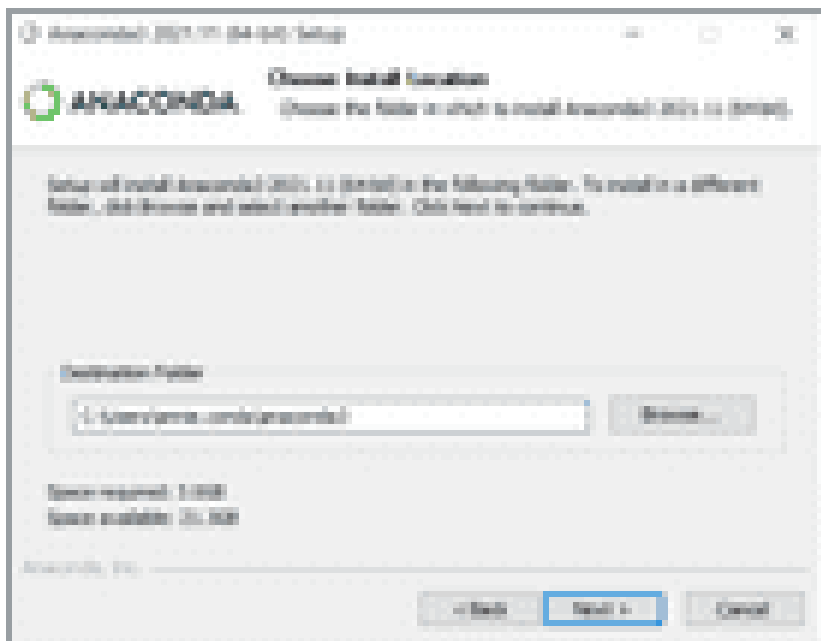
1. Akses situs <https://www.anaconda.com/>, kemudian klik download.



2. Akses situs <https://www.anaconda.com/>, kemudian klik download.



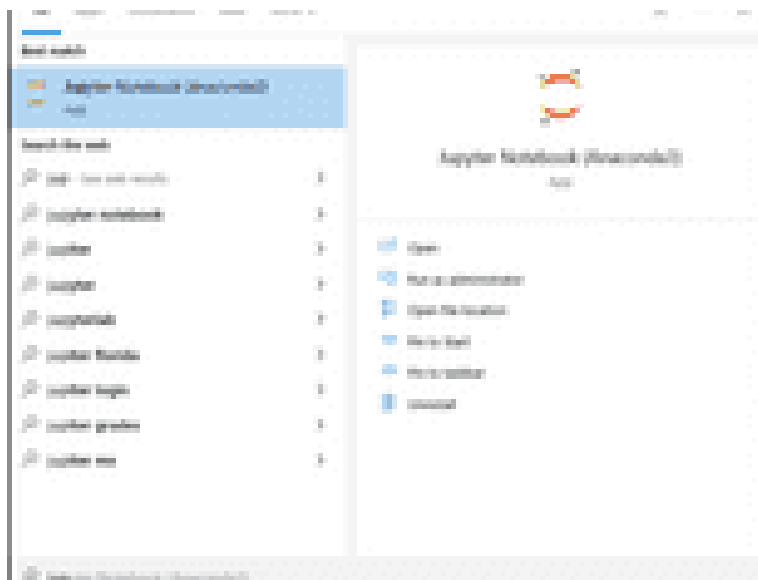
3. Ikuti hingga tahapan ini. Pilih directory sesuai keinginan anda. Disini saya menggunakan directory default anaconda nya.



4. Setelah berhasil melakukan instalasi Anaconda, maka akan ada dialog berikut.



5. Buka Start menu, cari Jupyter Notebook pada button window kita.



6. Berikut adalah tampilan awal dari Jupyter Notebook telah terinstall.

## 2.5 DATA STRUCTURES

Struktur Data adalah cara dasar dari setiap bahasa pemrograman di mana program dibangun dan mengatur data sehingga dapat diakses lebih efisien tergantung pada situasinya. Python membantu mempelajari dasar-dasar struktur data ini dengan cara yang lebih sederhana dibandingkan dengan bahasa pemrograman lainnya. Library Pandas mendukung dua buah struktur data yaitu series dan dataframe.

### 2.5.1 SERIES

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Pertama kita akan mempelajari tentang bagaimana cara membuat series.



```
In [1]: import pandas as pd
import numpy as np

s = pd.Series()
np.random.randn(5)
index = ['a', 'b', 'c', 'd', 'e']
name = "example"

s

Out[1]: a    0.401309
      b    1.469059
      c    0.281877
      d   -0.537009
      e    0.122892
      Name: example, dtype: float64
```

Data yang digunakan pada series diatas merupakan angka random sebanyak 5 buah. Untuk indexnya sendiri sebenarnya bisa kita ubah sesuai kebutuhan, akan tetapi pada contoh disini saya menggunakan huruf a hingga e. Yang terakhir merupakan nama dari series yaitu “example”.

3. Selanjutnya merupakan cara memperlakukan indeks pada series.



```
In [1]: s[0]
Out[1]: 0.000000000000000000

In [2]: s[1:4]
Out[2]:
a    0.000000
b    1.000000
c    0.000000
Name: example, dtype: float64
```

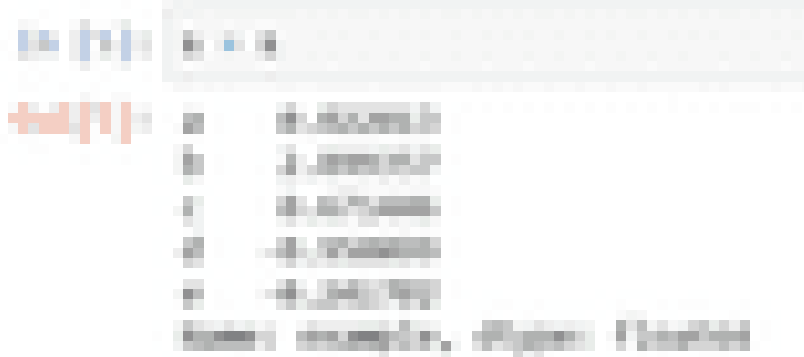
Pada contoh pertama, indeks pertama dari variabel `s` ditampilkan. Sedangkan untuk contoh kedua ialah indeks dari 0-3 dan data type nya itu float 64.

4. Berikutnya adalah cara untuk menampilkan atau menyembunyikan beberapa record pada series.

```
In [3]: s[[True, True, False, False, True]]
Out[3]:
a    0.000000
b    1.000000
a    0.000000
Name: example, dtype: float64
```

Terlihat bahwa ketika indeks yang dimaksud berisikan false, maka record tidak akan ditampilkan. Maka index yang di bikin pada baris kode diatas adalah nilai atau value yang bernilai benar, benar, salah, salah, benar.

5. Kalian juga bisa melakukan perhitungan cepat terhadap keseluruhan data dari series.



Untuk lebih lengkap mengenai series, kalian dapat mempelajarinya langsung menggunakan Jupyter source file yang sudah disediakan pada repositori GitHub buku ini.

## 2.5.2 DATAFRAME

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Pertama kita akan mempelajari tentang bagaimana cara membuat dataframe.



Pada dasarnya, dataframe merupakan sekumpulan series. Untuk pembuatannya sendiri, jika pada python bisa disamakan dengan dictionary of list. Pada tahapan ini kita membuat dataframe terlebih dahulu seperti ini

3. Pada kolom-kolomnya juga dapat dilakukan berbagai operasi seperti contoh Berikut

```
In [24]: df["nama"]

df["nama"] = df["nama"] + df["nama"]
df["nama"] = df["nama"]
df["nama"] = df["nama"] [1:]

df

Out[24]:
```

	nama	nama	nama	nama
0	1.0	2.0	NaN	NaN
1	2.0	4.0	NaN	NaN
2	3.0	6.0	NaN	NaN
3	4.0	8.0	NaN	NaN
4	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN

Lalu kita tambahkan 4 kolom dan 5 baris pada dataframe kita disini untuk kalimat yang bersifat NaN karena data yang kita buat belum kita masukan nilai value nya.

- Selanjutnya, untuk memperlakukan indeks juga bisa dilakukan dengan berbagai cara seperti gambar berikut

```
In [10]: df["Year"]
Out[10]:
a    1.0
b    2.0
c    3.0
d    4.0
e    NaN
f    NaN
Name: Year, dtype: float64
```

```
In [11]: df[["Year", "Name"]]
Out[11]:
   Year  Name
a     1.0   NaN
b     2.0   NaN
c     3.0   NaN
d     4.0   NaN
e     NaN   NaN
f     NaN   NaN
```

```
In [12]: df.loc["a", "Year"]
Out[12]: 1.0

In [13]: df.loc[["a", "a"], ["Year", "Name"]]
Out[13]:
   Year  Name
a     1.0   NaN
```

5. Kemudian ada beberapa fungsi yang akan berguna pada dataframe. Salah satunya ialah fungsi copy ini

	name	status	age	sex	height
0	A	1.0	2.0	Male	1.70
1	B	2.0	1.0	Male	1.75
2	A	1.0	1.0	Male	1.70
3	A	1.0	1.0	Male	1.70
4	F	1.0	1.0	Male	1.70
5	F	1.0	1.0	Male	1.70
6	F	1.0	1.0	Male	1.70

Untuk lebih lengkap mengenai dataframe, kalian dapat mempelajarinya langsung menggunakan Jupyter source file yang sudah disediakan pada repositori GitHub buku ini.

## 2.6 INDEXING AND SELECTING

### 2.6.1 MULTI-INDEX

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Perhatikan pada gambar dibawah ini, berikut adalah cara untuk melakukan multi-index

```

In [1]: import pandas as pd
import numpy as np

In [2]: df = pd.read_csv('data.csv')
df.head()

Out[2]:
   name  status  age  sex  height
0  A      1.0    2.0  Male   1.70
1  B      2.0    1.0  Male   1.75
2  A      1.0    1.0  Male   1.70
3  A      1.0    1.0  Male   1.70
4  F      1.0    1.0  Male   1.70
5  F      1.0    1.0  Male   1.70
6  F      1.0    1.0  Male   1.70

In [3]: df_multi = df.set_index(['name', 'status'])
df_multi

Out[3]:
name status age
A      1.0    2.0
      1.0    1.0
F      1.0    1.0
      1.0    1.0
      1.0    1.0
      1.0    1.0
      1.0    1.0

```

3. Tetapi, tentunya sulit untuk mendapatkan salah satu data dari dataframe tersebut. Contohnya kita ingin mengambil data dari Laki-laki bukan perokok.

Untuk menanggulangi hal tersebut, salah satu caranya ialah dengan melakukan reset index.

```
In [11]: data_gd = data_gd.reset_index()
data_gd
```

```
Out[11]:
```

	sex	smoker	tip
0	Male	Yes	3.000000
1	Male	No	3.134020
2	Female	Yes	1.010000
3	Female	No	1.010000

Pada tahapan ini terdapat sex, smoker dan tip pada output dataframe ini juga terdapat gender male dan female yaitu laki dan perempuan, lalu jumlah perokok dan yang tidak merokok.

4. Dengan dataframe seperti diatas, maka mudah bagi kita untuk mengambil data laki-laki bukan perokok.

```
In [12]: data_gd[data_gd["smoker"] == "No" & data_gd["sex"] == "Male"]
```

```
Out[12]:
```

	sex	smoker	tip
1	Male	No	3.134020

Untuk tahapan ini di ambil dari jumlah yang tidak merokok dan gender laki-laki terdapat disini 3.113402 gender laki-laki yang tidak merokok.

## 2.6.2 GETTING SINGLE VALUES

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Ini merupakan tips untuk menggunakan “at” ketika kita akan mendapatkan atau mengubah nilai pada sebuah data.

```
In [11]: figs.at[0, "total_bill"] = 100
figs.head(1)
```

	total_bill	tip	name	sex	smoker	day	time	waiter
0	16.99	1.01	Female		No	Sun	Dinner	2

kita dapat menelusuri setiap elemen dalam daftar dan memeriksa apakah elemen tersebut sudah ada di tips jika tidak ada di sana, maka kita dapat menambahkannya ke tips.

3. Mengapa demikian? Hal ini dikarenakan penggunaan “at” dapat mempercepat operasi yang dibuktikan pada gambar dibawah ini

```
In [12]: figs.at[0, "total_bill"] = 100
figs.at[0, "total_bill"]
```

```
In [13]: figs.at["total_bill", 0] = 100
figs.at["total_bill", 0]
```

## 2.7 GROUP OPERATIONS

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Pertama kita akan melakukan load terhadap dataset tips

```
In [14]: figs = pd.read_csv("figs")
figs.head(1)
```

	total_bill	tip	name	sex	smoker	day	time	waiter
0	16.99	1.01	Female		No	Sun	Dinner	2

grup operasi membutuhkan data untuk dipecah menjadi kelompok-kelompok untuk melakukan berbagai operasi pada kelompok-kelompok ini.

### 2.7.1 GROUP BY

1. Operasi pertama yang akan kita coba lakukan ialah group by.

```
In [13]: data_gbr = data.groupby(['sex', 'smoker'])
data_gbr
```

```
Out[13]: pandas.core.groupby.generic.DataFrameGroupBy object of DataFrame
```

Group By bekerja dengan menginputkan parameter berupa list dari nama kolom. Kemudian, group by akan mengembalikan kombinasi unik dari tiap tiap kolom tersebut. Sehingga nantinya dataset ini memiliki empat kelompok: perokok pria, perokok wanita, pria bukan perokok, wanita bukan perokok.

### 2.7.2 AGG

1. Selanjutnya ada operasi agg atau agregasi.

```
Out[14]:
```

			age		total_bill	
			mean	min	max	min
sex: smoker						
Male	Yes	2.000000	1.00	Yes	100	
	No	2.000000	1.00	No	50	
Female	Yes	2.000000	1.00	Yes	100	
	No	2.000000	1.00	No	50	

Operasi agregasi ini menggabungkan data dalam sebuah grup menjadi satu nilai. Contohnya seperti pada gambar, disini kita meminta mean dan min dari tiap-tiap grup.

2. Untuk mempermudah pemilihan, kita bisa melakukan reset\_index sehingga hasilnya seperti dibawah ini.

```
In [15]: data_agg = data_agg.reset_index()
```

```
Out[15]:
```

	sex	smoker	age	age	total_bill	total_bill
			mean	min	max	min
0	Male	Yes	2.000000	1.00	Yes	100
1	Male	No	2.000000	1.00	No	50
2	Female	Yes	2.000000	1.00	Yes	100
3	Female	No	2.000000	1.00	No	50



Method `reset_index()` berguna untuk mengembalikan ke default index, maka index lama akan dibuang dari dataframe. Secara default adalah `False`, jadi index lama akan dikembalikan sebagai kolom biasa pada dataframe.

### 2.7.3 FILTER

1. Pertama kita akan melakukan group by berdasarkan hari dan waktu pada dataset time.

```
In [17]: tips.groupby(['day', 'time'])
          tips
```

2. Selanjutnya kita akan mencari total dari kolom `size` pada masing masing grup. Lalu mediannya diambil dan dijadikan nilai untuk variabel `median_size`.

```
In [18]: median_size = tips.groupby(['day', 'time']).size().median()
```

Pada gambar dibawah, fungsi filter hanya akan menampilkan data yang nilai total kolom `size` kurang dari `median_size` saja. Fungsi `filter()` digunakan untuk menyaring elemen, item, atau anggota iterable dengan bantuan fungsi yang bertugas menguji setiap anggota iterable apakah bernilai `True` atau `False`. Hasil atau nilai yang dikembalikan berupa iterable baru dari anggota iterable lama yang sudah melewati proses pengujian (dari fungsi yang dibuat) dan bernilai `True`.

## 2.8 ROW-COLUMN TRANSFORMATIONS

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Pertama kita akan melakukan load terhadap dataset tips

```
In [1]: tips = sns.load_dataset('tips')
          tips.head()
```

```
Out[1]:
```

	total_bill	tip	smoker	smoker	day	time	size
0	16.99	1.01	Female	No	Mon	Dinner	2
1	10.34	1.66	Male	No	Mon	Dinner	3
2	21.01	3.50	Male	No	Mon	Dinner	3

Mungkin akan terbesit pertanyaan mengenai berapa rasio laki-laki dan perempuan pada hari yang berbeda dalam seminggu?

3. Mari kita coba menggunakan group by dan agg

```
In [11]: data_gd = data.groupby('day', 'sex').agg(['sum'])
data_gd
```

In [11]:

sex		
day	sex	
Thur	Male	75
	Female	75
Fri	Male	25
	Female	75
Sat	Male	100
	Female	75
Sun	Male	100
	Female	75

Sebenarnya dari sini sudah selesai, tetapi ketika anda akan mengolahnya lagi, maka akan lebih banyak operasi kolom untuk membandingkan pengunjung pria dan wanita.

2.8.1 UNSTACK

- 1. Mari kita coba menggunakan fungsi unstack terhadap dataframe diatas.

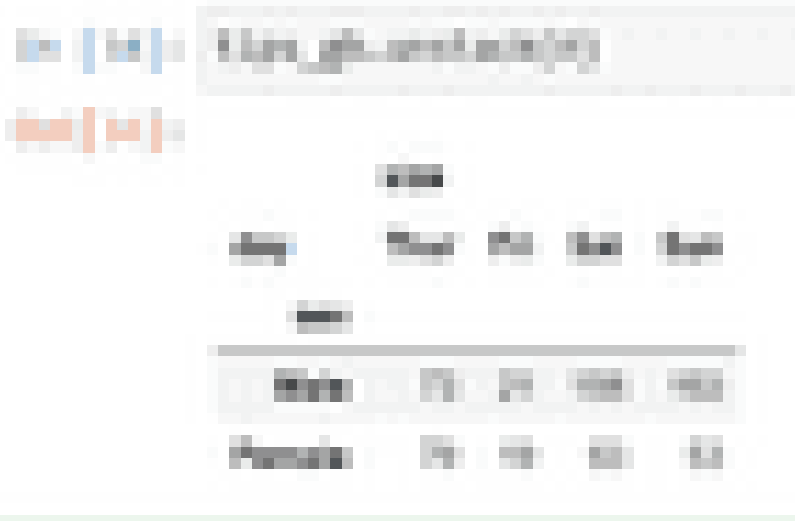
```
In [12]: data_un = data_gd.unstack()
data_un
```

In [12]:

sex		
day	Male	Female
Thur	75	75
Fri	25	75
Sat	100	75
Sun	100	75

Sederhananya, dengan menggunakan fungsi unstack kita memindahkan indeks ke kolom stacking dan unstacking sama dengan melt dan pivot secara berurutan, hanya saja tidak memasukkan index sebagai parameter di stack/unstack tapi harus set index terlebih dahulu, baru bisa melakukan stacking/unstacking dengan level yang bisa ditentukan sendiri.

2. Selain itu, anda juga bisa mengubah orientasi dari dataframennya seperti ini.



## 2.8.2 STACK

Tumpukan (stack) adalah struktur data yang menerapkan konsep LIPO (Last In First Out) artinya data terakhir yang ditambahkan ke stack akan berada pada posisi terakhir, sama seperti saat kita menumpuk buku, maka posisi buku terakhir atau itulah puncak yang akan kita ambil dulu.

1. Berikut ketika kita menggunakan fungsi stack pada kode yang dibuat.

```

In [10]: tigre_pus.stacked()

Out[10]:

```

	sex	age
Buar	Male	75
	Female	75
Pai	Male	25
	Female	75
Bui	Male	100
	Female	75
Bui	Male	100
	Female	75

2. Anda dapat menghapus salah satu indeks kolom dari dataset.

```

In [10]: tigre_pus.stacked()[0]

Out[10]:

```

	sex	Male	Female
Buar	sex	75	75
	age	25	75
Bui	sex	100	75
	age	100	75

Stack adalah struktur data yang dapat ditunjukkan oleh tempat penyisipan dan penghapusan elemen terjadi hanya pada satu tempat yang disebut puncak tumpukan. Cara dasar untuk mengakses data di stack adalah dengan metode Last In First Out (LIFO).

### 2.8.3 MELT AND PIVOT

Teknik peleburan melalui `.melt()` digunakan untuk mengembalikan kondisi data yang telah di pivot sebelum di pivot.

Untuk menerapkan metode `.pivot()` pada dataframe, dapat dilakukan pada dataframe yang memiliki single index atau multiindex index.

1. Buat dataframe terlebih dahulu seperti gambar dibawah.

```
In [10]: df_dataframe = pd.DataFrame({'level': ['John', 'Mary'],
                                     'sex': ['Male', 'Female'],
                                     'height': [1.7, 1.6],
                                     'weight': [150, 160]})

df_dataframe
```

```
Out[10]:
```

	level	sex	height	weight
0	John	Male	1.7	150
1	Mary	Female	1.6	160

2. Dengan menggunakan fungsi melt, maka akan ada kolom tambahan yaitu variable dan value.

```
In [11]: df_dataframe_melt = df_dataframe.melt(id_vars="level",
                                                variable_name="variable",
                                                value_name="value")

df_dataframe_melt
```

	level	sex	variable	value
0	John	Male	height	1.7
1	Mary	Female	height	1.6
2	John	Male	weight	150
3	Mary	Female	weight	160

3. Selanjutnya kita akan melakukan pivot pada dataframe tersebut dan kita jalankan.

```
In [12]: df_dataframe_melt.pivot(index="level",
                                   columns="variable",
                                   values="value")

df_dataframe_melt.pivot(index="level",
                           columns="variable",
                           values="value")
```

	level	sex	height_0	weight_0
0	John	Male	height	1.7
1	John	Male	weight	150
2	Mary	Female	height	1.6
3	Mary	Female	weight	160

Penggunaan custom aggregation lainnya pada dataframe yang telah digroupby dapat dilakukan dengan mempasskan sebuah dict yang berisi 'key' dict sebagai

nama kolomnya dan 'value' dict adalah fungsi untuk agregasi, baik user defined function atau yang telah tersedia.

## 2.8.4 DUMMY VARIABLES

Variabel dummy adalah variabel yang digunakan untuk mengkategorikan data kualitatif dan data kualitatif berskala nominal. Variabel dengan dua kategori tersebut disebut variabel dummy, namun skala pengukuran dalam model regresi yang dibentuk, data yang digunakan adalah campuran, misalnya: variabel bergantung pada skala nominal, sedangkan variabel bebas dapat menggunakan skala nominal dan rasio.

1. Pertama, mari kita tampilkan 5 data teratas pada dataframe.

```
In [14]: display(head(df))
```

```
Out[14]:
```

	nama_judul	tgl	sex	rentan	slay	stres	nilai
0	18-08-18	1-01	Female	No	Run	Comer	2
1	18-08-18	1-01	Male	No	Run	Comer	2
2	21-01-18	2-01	Male	No	Run	Comer	2
3	21-08-18	2-01	Male	No	Run	Comer	2
4	24-08-18	2-01	Female	No	Run	Comer	2

2. Ketika kita menggunakan fungsi `get_dummies` pada kolom `sex`, maka akan ada dua buah kolom tambahan yaitu `sex_male` dan `sex_female`. Baris dari kolom-kolom tersebut dapat digunakan untuk menjalankan berbagai operasi.

```
In [15]: df.get_dummies(df['sex'], columns=['sex'])
```

```
Out[15]:
```

	nama_judul	tgl	rentan	slay	stres	nilai	sex_Male	sex_Female
0	18-08-18	1-01	No	Run	Comer	2	0	1
1	18-08-18	1-01	No	Run	Comer	2	1	0
2	21-01-18	2-01	No	Run	Comer	2	1	0
3	21-08-18	2-01	No	Run	Comer	2	1	0
4	24-08-18	2-01	No	Run	Comer	2	0	1

## 2.9 COMBINING DATAFRAMES

Saat menggabungkan DataFrames hanya cukup menambahkannya satu sama lain, yaitu menumpuknya secara vertikal atau berdampingan. Cara lain untuk menggabungkan DataFrames adalah dengan menggunakan kolom di setiap kumpulan data yang berisi nilai umum (id unik umum). Menggabungkan DataFrame menggunakan bidang umum disebut "join". Kolom yang berisi nilai umum disebut "join key(s)". Bergabung dengan DataFrames dengan cara ini seringkali berguna ketika satu DataFrame adalah "tabel pencarian" yang berisi data tambahan yang ingin kita sertakan di yang lain. Untuk tahapan selanjutnya yaitu:

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Pertama kita akan melakukan load terhadap dataset tips

### 2.9.1 MERGE

Metode `.merge()` untuk menggabungkan Series/Dataframes memiliki bentuk yang mirip dengan sintaks join di SQL, menentukan tabel kiri dan kanan, kunci join dan cara bergabung (left, right, inner, full outer).

1. Pertama, mari kita pisahkan antara dataframe yang hanya memiliki kolom `total_bill` dan dataframe yang hanya memiliki kolom `tip`

```
In [14]: tips_bill = tips.groupby('sex', 'smoker')[['total_bill', 'tip']].reset()
tips_tip = tips.groupby('sex', 'smoker')[['total_bill', 'tip']].reset()

del tips_bill['tip']
del tips_tip['total_bill']

tips_bill

Out [14]:
```

		total_bill	
	sex	smoker	
Male	Yes	Not Sm	16.99
	No	Not Sm	10.34
Female	Yes	Not Sm	16.99
	No	Not Sm	17.01

```
tips_tip

Out [15]:
```

		tip	
	sex	smoker	
Male	Yes	Not Sm	2.01
	No	Not Sm	1.66
Female	Yes	Not Sm	2.01
	No	Not Sm	1.66

2. Untuk cara pertama, kita dapat melakukan merge dengan menggunakan index.

```
in [36]: pd.merge(Left, Right, left_on='index', right_on='index')
```

```
out [36]:
```

			index_Left	index_Right
Male	Male	1000000	1000000	
	Male	1000000		1000000
Female	Male	1000000		1000000
	Male	1000000	1000000	

3. Kita juga dapat menggabungkan sebagian kolom dan indeks pada data.

```
in [37]: pd.merge(Left, Right, left_on='index', right_on='index',
                  left_index=True, right_index=True,
                  left_keys=['index', 'index'],
                  right_keys=['index', 'index'])
```

```
out [37]:
```

			index_Left	index_Right
1	Male	Male	1000000	1000000
	Male	Male	1000000	1000000
2	Female	Male	1000000	1000000
	Female	Male	1000000	1000000



4. Kita juga dapat melakukan fungsi merge seperti pada SQL, menjadi seperti ini. Disini kita bisa melihat dari hasil method merge yang diterapkan pada baris kode berikut dan hasilnya adalah seperti ini.

```
In [17]: pd.merge(
         1: dfp, dfm, on='name', how='left',
         2: dfp, dfm, on='name', how='left',
         3: )
```

```
In [17]:
```

	name	sex	total_bill	tip
0	Male	Yes	16.99	1.01
1	Male	No	10.34	1.66
2	Female	Yes	16.99	1.01
3	Female	No	10.34	1.66

```
In [18]: pd.merge(
         1: dfp, dfm, on='name', how='left',
         2: dfp, dfm, on='name', how='left',
         3: )
```

```
In [18]:
```

	name	sex	total_bill	tip
0	Male	Yes	16.99	1.01
1	Male	No	10.34	1.66

2.9.2 CONCATENATION

Untuk concatenation sendiri, ibaratnya seperti menambahkan baris pada dataframe. Untuk lebih jelasnya perhatikan contoh dibawah ini.

```
In [7]: pd.concat([Tipe_Mall, Tipe_Rip, Tipe_Rip], sort=False)
```

```
Out[7]:
```

nama_jadi_rip			
nama_sumber			
Male	Yes	100-120"	100"
	No	100-120"	100"
Female	Yes	100-120"	100"
	No	100-120"	100"
Male	Yes	100"	100-120"
	No	100"	100-120"
Female	Yes	100"	100-120"
	No	100"	100-120"
Male	Yes	100"	100-120"
	No	100"	100-120"
Female	Yes	100"	100-120"
	No	100"	100-120"

## 2.10 MISC FUNCTION

Pandas sangatlah besar dan memiliki ratusan fungsi. Jadi kita tidak akan membahas semuanya di sini, tetapi berikut adalah beberapa fungsi yang akan berguna.

### 2.10.1 SAMPLE

```
In [88]: Tipe.sample(5)
```

```
Out[88]:
```

	nama_jadi	rip	nama_sumber	day	time	size
00	10-100	1-100"	Female	Yes	Yes Dinner	1
100	10-100	1-100"	Male	Yes	Yes Lunch	1
100	10-100	1-100"	Male	Yes	Yes Dinner	1
00	100-120	1-100"	Male	Yes	Yes Lunch	1
00	10-100	1-100"	Male	Yes	Yes Dinner	1

sample() adalah fungsi bawaan dari modul acak dengan Python yang mengembalikan daftar panjang tertentu dari item yang dipilih dari urutan yaitu

daftar, tuple, string, atau set. Digunakan untuk pengambilan sampel acak tanpa penggantian. Dengan fungsi ini, kita akan mendapatkan sampel acak pada sebuah dataframe.

### 2.10.2 ISIN

```
In [88]: 1.ign([1.ign.day.isin(["sat", "sun"])]).sample(5)
```

```
Out[88]:
```

	month_year	day	max	min	max	min	max	min
100	2017-07-01-01	Monday	Yes	Yes	Yes	Yes	Yes	Yes
100	10-06-01-01	Monday	Yes	Yes	Yes	Yes	Yes	Yes
100	10-06-01-01	Monday	Yes	Yes	Yes	Yes	Yes	Yes
100	10-06-01-01	Monday	Yes	Yes	Yes	Yes	Yes	Yes
100	10-06-01-01	Monday	Yes	Yes	Yes	Yes	Yes	Yes

isin() isin() digunakan untuk mengecek apakah ada elemen tertentu di dalam dataframe. Function ini mengembalikan nilai Boolean, True atau False, untuk tiap elemen pada dataframe. Fungsi ini, akan mengembalikan dataframe yang sesuai dengan kondisi isin.

### 2.10.3 DROP\_DUPLICATES

```
In [89]: 1.ign(["1.ign", "day"])[0].drop_duplicates(inplace=True)
```

```
Out[89]:
```

	max	min
100	Yes	Yes
100	Yes	Yes
100	Yes	Yes
100	Yes	Yes
100	Yes	Yes
100	Yes	Yes

Berbeda dengan method duplicated() pada modul sebelumnya, drop\_duplicates() berguna untuk membuang data duplikat pada DataFrame. Drop\_duplicates akan menghilangkan data yang duplikat pada sebuah Dataframe yang telah kita buat.

### 2.10.4 CUT

```

In [11]: pd.cut(hike['total_bill'], 5, labels=['low', 'mid', 'high'], include_lowest=True)

Out[11]:
0      low
1      low
2      mid
3      mid
4      mid
Name: total_bill, dtype: category
Categories (3, object): ['low' < 'mid' < 'high']

```

Fungsi pandas cut() digunakan untuk memisahkan elemen-elemen array ke dalam bin yang berbeda. Fungsi potong terutama digunakan untuk melakukan analisis statistik pada data skalar. Cut akan melabeli data sesuai dengan kategori yang sudah kita masukan pada parameter labels.

# BAB III

## VISUALISASI DATA

---

### A. Pendahuluan

Visualisasi data adalah salah satu senjata paling ampuh dalam ilmu data. Kata visualisasi mengacu pada teknik menggambarkan berbagai jenis grafik untuk mewakili data. Pada umumnya titik data divisualisasikan dalam bentuk scatter plot atau histogram dengan tujuan agar insight yang diperoleh dari hasil pengolahan data lebih sederhana. Selain itu, manusia cenderung lebih mudah mempelajari sesuatu dengan menggunakan bagan atau gambar, sehingga visualisasi data dianggap lebih efektif untuk menyampaikan informasi dan mudah diterima oleh otak manusia. Visualisasi data juga dapat membantu data scientist untuk mempresentasikan temuannya kepada pemangku kepentingan dan nantinya temuan tersebut akan digunakan sebagai data untuk mendukung pembuatan kebijakan.

Bagi seorang analis data besar, visualisasi data yang baik dan jelas adalah kunci untuk mengomunikasikan wawasan yang mereka temukan dengan lebih baik. Dengan volume data yang terus meningkat, mustahil bagi seorang ilmuwan data untuk menceritakan sebuah kisah tanpa visualisasi data. Satu alat yang digunakan oleh banyak profesional data untuk mengubah angka menjadi informasi yang berwawasan adalah Python. Bahasa pemrograman ini menawarkan berbagai paket visualisasi data yang dapat kita gunakan untuk membuat visualisasi yang menarik. Bagaimana caranya? Pada artikel kali ini, DQLab akan menjelaskan empat langkah yang dapat kita terapkan untuk membuat visualisasi data yang menarik.

### B. Tujuan dan Capaian

Tujuan yang ingin dicapai setelah pembelajaran ini adalah. Sebagai berikut:

1. Diberikan suatu program yang memuat fungsi, mahasiswa dapat menjelaskan, mempresentasikan, hingga memanfaatkan cara kerja program tersebut dengan benar.
2. Diberikan suatu masalah, mahasiswa dapat menghasilkan program untuk menyelesaikan masalah tersebut dengan melibatkan penggunaan Data Mining dalam Python dengan kreatif, dan dapat mempresentasikannya dengan bertanggung jawab.

3. Diberikan suatu fungsi yang didefinisikan secara rekursif, mahasiswa dapat menghasilkan program Data Mining dalam Python untuk mengimplementasikan dengan benar.

### **C. Uraian Materi**

Sejauh ini beberapa fitur Python Data Mining yang paling banyak digunakan adalah variabel, ekspresi, percabangan, perulangan, input dan cetak. Secara teori hanya dibutuhkan beberapa instruksi untuk menulis sebuah program. Akan tetapi instruksi saja tidak cukup. Ketika masalah semakin kompleks, mungkin sebuah program tidak dilaksanakan oleh satu orang, dibutuhkan tim pemrogram untuk membangun proyek besar. Dalam kehidupan sehari-hari, untuk memudahkan pengaturan, Proyek besar biasanya dipecah menjadi proyek yang lebih kecil. Setiap subproyek akan fokus pada pemecahan submasalah secara terpisah. Begitu juga dalam pemrograman. Program besar dapat dibagikan menjadi program-program kecil yang disebut rutinitas program (atau rutin). Rutinitas adalah bagian penting dari pemrograman. Sesuatu rutin adalah nama set pernyataan yang akan dieksekusi tugas tertentu. Fungsi dibuat untuk membuat program lebih terstruktur sehingga mudah dipahami dan mudah dikembangkan.

### **D. Latihan**

Ketika kita mempelajari ilmu data, tentu kita akan berurusan dengan banyak data. Tentunya kita akan kesulitan mendapatkan informasi jika melihat data atau tabel yang sangat panjang. Alih-alih melihat data baris demi baris, kami dapat menampilkan data secara grafis. Itulah yang disebut visualisasi data, menyajikan data kita dengan cara yang lebih menarik dan mudah dipahami, seperti tabel, grafik atau bentuk lainnya sesuai kebutuhan.

Kami akan membuat proyek sederhana, menganalisis data tentang popularitas bahasa pemrograman. Proyek ini bertujuan untuk menunjukkan betapa mudahnya mempelajari visualisasi data dengan Python Data Mining.

#### **3.1 VISUALIZING NUMERICAL CATEGORICAL DATA**

Categorical data adalah kumpulan informasi yang berbentuk kelompok/group dari gambar dibawah ini terdapat data: Name, Sex, Age, Height, Weight, Team, No, Games, Year, Season, City, Sport, Event, Medal.

1. Pertama, mari kita load dataset athlete\_events.csv dengan kita import library pandas dan di inisialisasi kan pd agar baris kode pada program lebih simple dan efisien.

```

14 [3]: report_pendiri_mn(pd)
      pd.read_csv('athlete_events.csv')

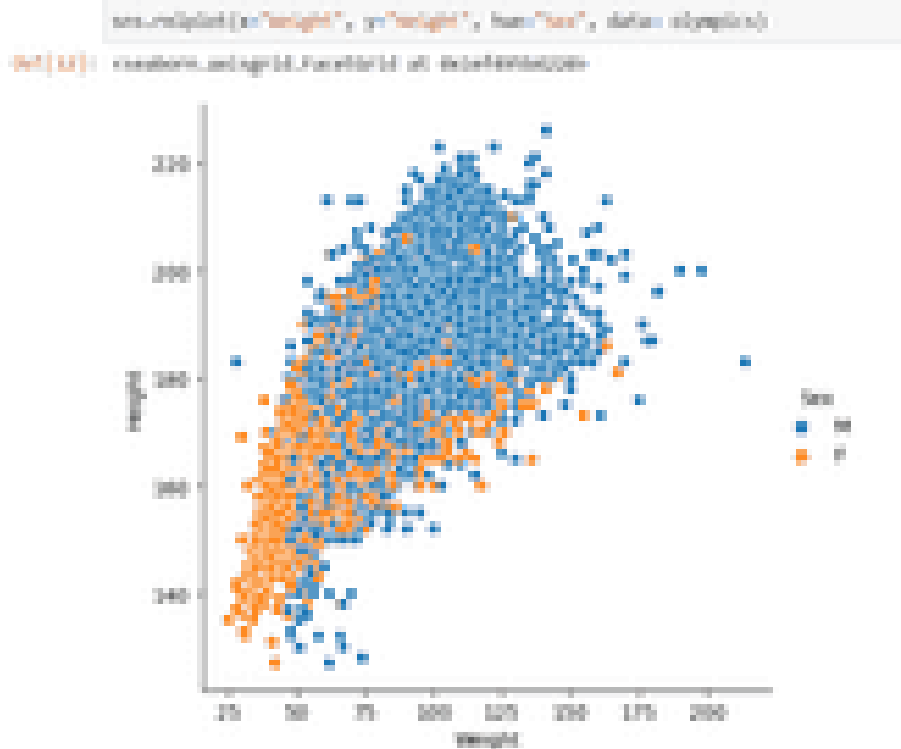
14 [3]: pd.read_csv('athlete_events.csv')
Out[3]:

```

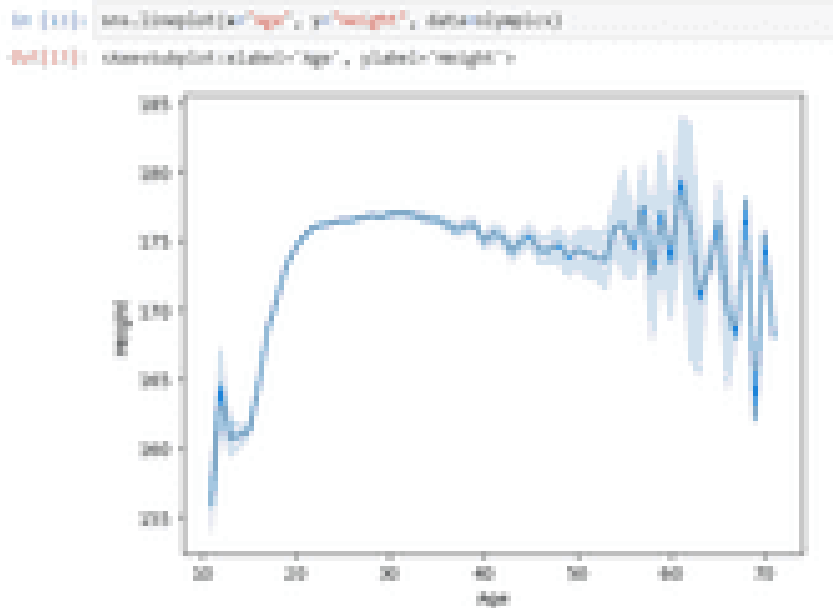
ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Result
1	Aleksandr	M	24.0	180.0	80.0	(URS)	(URS)	1980 Summer	1980	Summer	Barcelona	Handball	Handball Men's Handball	Gold
2	Aleksandr	M	23.0	175.0	80.0	(URS)	(URS)	1972 Summer	1972	Summer	London	Judo	Judo Men's Bantamweight	Gold
3	Bernard Hansen	M	24.0	NaN	NaN	(DEN)	(DEN)	1920 Summer	1920	Summer	Antwerpen	Football	Football Men's Football	Gold
4	Edgar Lindqvist	M	34.0	NaN	NaN	(FIN)	(FIN)	1920 Summer	1920	Summer	Paris	Top OA	Top OA Men's Top OA	Gold
5	Christina Zwarg	F	21.0	168.0	60.0	(GER)	(GER)	1992 Winter	1992	Winter	Calgary	Speed Skating	Speed Skating Women's 500m	Gold

### 3.1.1 VISUALIZING STATISTICAL RELATIONSHIPS

1. Pertama, kita akan mencoba relplot dengan x nya ialah kolom Weight dan y nya merupakan kolom Height. Untuk hue nya sendiri kita menggunakan kolom Sex. Relplot dapat memvisualisasikan hubungan statistic antara variabel kuantitatif.



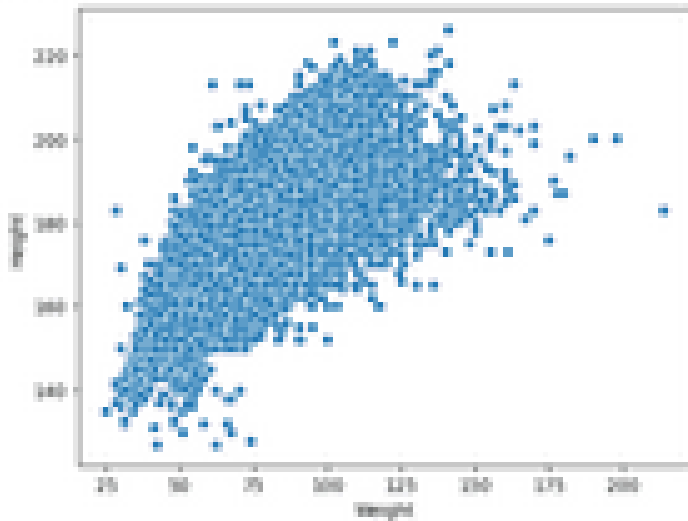
2. Selanjutnya ada lineplot yang menggunakan Age sebagai x dan Height sebagai y. Line plot adalah salah satu jenis visualisasi data yang banyak digunakan dan merupakan jenis plot dasar dalam visualisasi data. Jenis plot ini menampilkan informasi berupa rangkaian titik data yang terhubung dengan segmen garis lurus.



3. Yang terakhir adalah scatterplot yang dapat memvisualisasikan hubungan antara Weight dan Height seperti gambar dibawah.



```
In [34]: sns.scatterplot(x="weight", y="height", data=olympians)
```

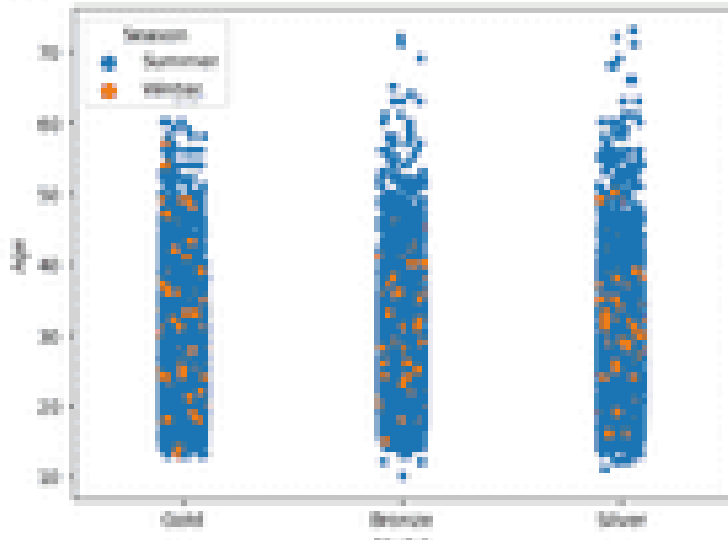


Scatter plot, juga dikenal sebagai diagram sebar, adalah diagram grafis yang dibangun dari dua sumbu X dan Y (variabel X dan variabel Y). Nilai dari pasangan variabel ini direpresentasikan sebagai titik. Oleh karena itu, Scatter plot sering disebut sebagai diagram sebar. Diagram ini dapat menunjukkan jenis hubungan.

### 3.1.2 CATEGORICAL PLOT

1. Hampir sama dengan scatterplot, hanya saja pada stripplot dibagi perkategori seperti pada gambar dibawah.

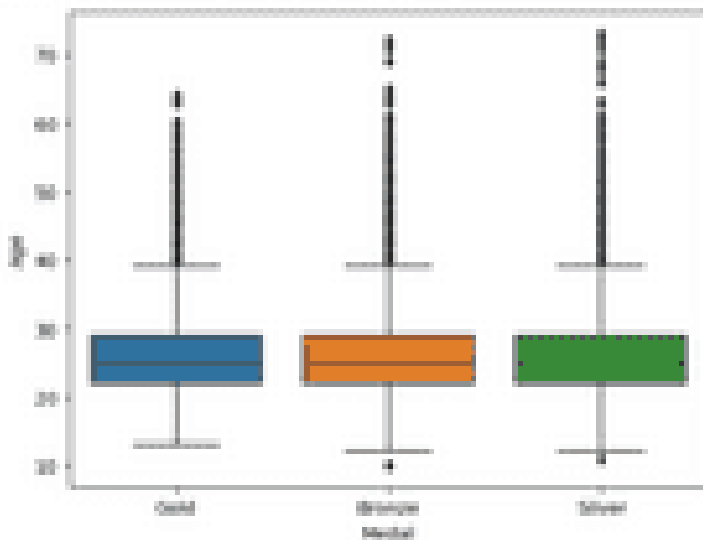
```
In [16]: sns.stripplot(x="Medal", y="Age", hue="Gender", data=olympics)
```



Rancangan Split-Blok adalah Strip-Plot atau Rancangan Petak-Berjalur. Rancangan ini sesuai untuk percobaan dua faktor dimana ketepatan pengaruh interaksi antar faktor lebih diutamakan dibandingkan dengan dua pengaruh lainnya.

2. Selanjutnya ada boxplot dengan Medal sebagai x dan Age sebagai y.

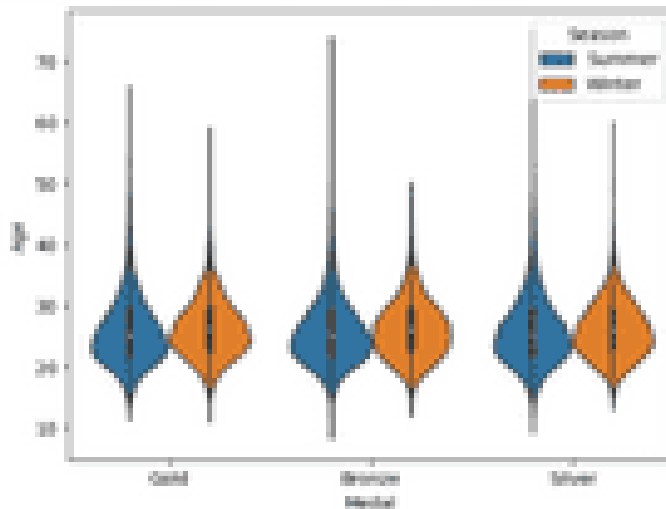
```
In [17]: sns.boxplot(x="Medal", y="Age", data=olympics)
```



Box plot adalah jenis visualisasi data yang secara statistik merepresentasikan distribusi data melalui lima dimensi utama, yaitu nilai minimum, kuartil 1, kuartil 2 (median), kuartil 3, dan nilai maksimum.

3. Lalu ada violinplot yang sudah dibagi perkategori dengan x adalah Medal dan y adalah Age.

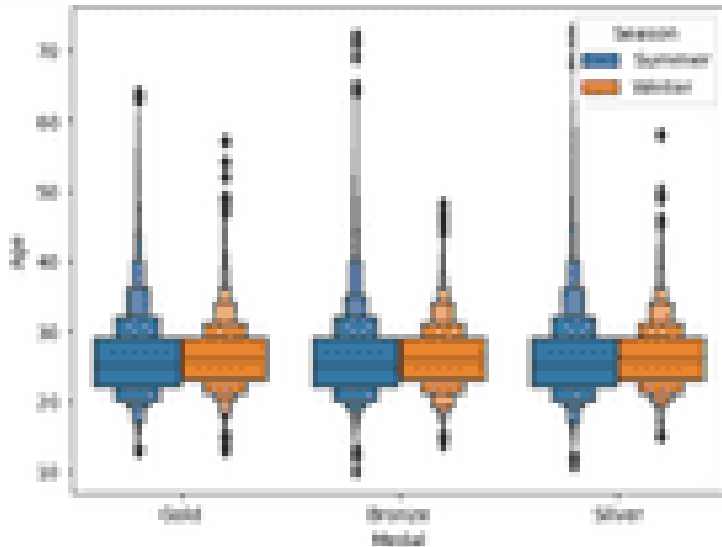
```
sns.violinplot(x="Medal", y="Age", hue="Season", data=olympics)
```



Violin Plot adalah kombinasi dari sebuah box plot dan sebuah kernel density plot (alur kepadatan titik). Detailnya, diawali dengan sebuah box plot. Lalu ada penambahan kepadatan titik rotasi plot ke setiap sisi box plot.

4. Selanjutnya ada boxenplot dengan Medal sebagai x dan Age sebagai y dan Season sebagai hue.

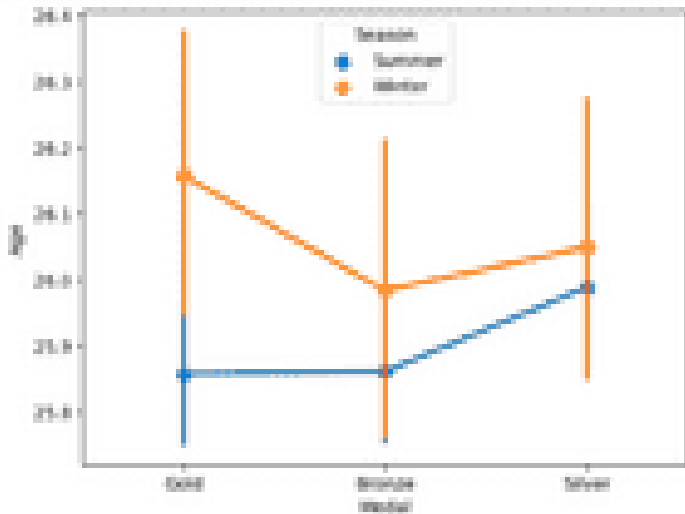
```
In [44]: sns.boxplot(x="Medal", y="Age", hue="Season", data=us_athlete_performance)
```



Gambar plot kotak yang disempurnakan untuk kumpulan data yang lebih besar. Gaya plot ini awalnya disebut plot "nilai-huruf" karena menunjukkan sejumlah besar kuantil yang didefinisikan sebagai "nilai-huruf". Ini mirip dengan plot kotak karena memplot representasi nonparametrik dari distribusi di mana semua fitur sesuai dengan pengamatan yang sebenarnya.

5. Dan yang terakhir adalah ada pointplot dengan Medal sebagai x dan Age sebagai y dan Season sebagai hue.

```
In [25]: sns.pointplot(x="Medal", y="age", hue="Gender", data=olympics)
```



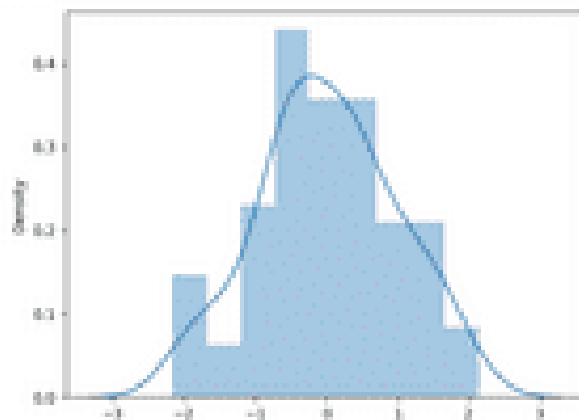
visualisasi data berbentuk box (kotak berbentuk bujur sangkar. Boxplot dapat digunakan untuk menunjukkan perbedaan antara populasi tanpa menggunakan asumsi distribusi statistik yang mendasarinya.

### 3.1.3 VISUALIZING DISTRIBUTION

1. Cara paling mudah untuk melihat sekilas distribusi univariat di seaborn adalah fungsi distplot.

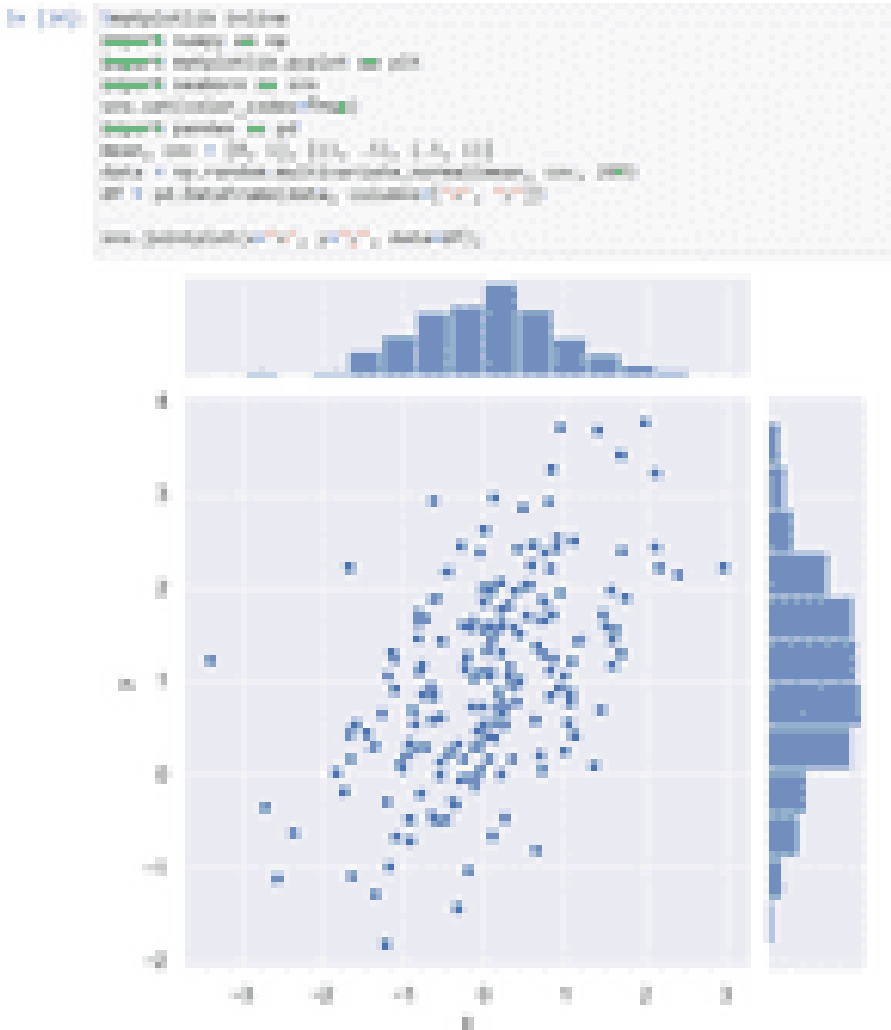
```
In [27]: x = np.random.randn(10000)
sns.distplot(x)

1/Warning:distplot is deprecated in 0.10.0. The package/seaborn/distplot.py:2000. For
it will be removed in a future version. Please adapt your code to use either
distplot() or 'histplot' (an axis-level function for histograms).
warnings.warn(msg, FutureWarning)
```



Fungsi normal acak Numpy menghasilkan sampel angka yang diambil dari distribusi normal, atau disebut distribusi Gaussian. Displot digunakan untuk membuat histogram. Misalnya kita ingin membuat distribusi tip yang diberikan pelanggan dengan dengan interval 100.

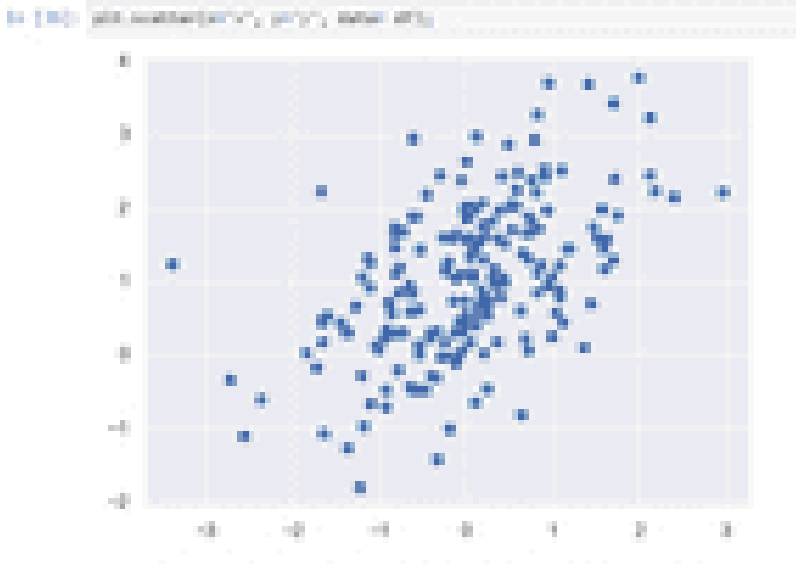
2. Kita juga dapat memvisualisasikan distribusi bivariat dari dua variabel n menggunakan seaborn. Cara termudah untuk melakukannya di seaborn adalah dengan menggunakan fungsi jointplot.



Jointplot digunakan untuk menganalisis kombinasi 2 data numerik untuk melihat korelasinya. Misalnya kita ingin melihat korelasi distribusi variabel total

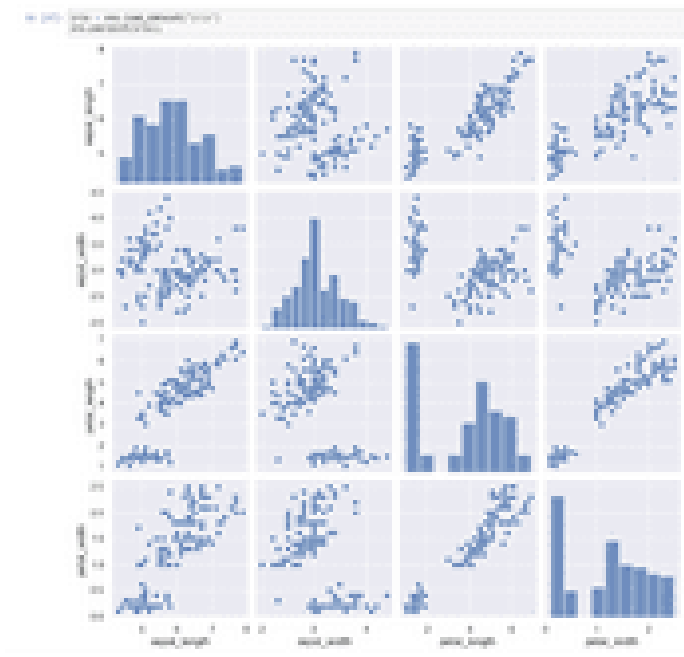
bil dan tip.. apakah semakin besar total bil pelanggan akan memberikan tip yang semakin besar. Kita juga harus tentukan sumbu X dan sumbu Y.

3. Untuk matplotlib kita bisa menggunakan fungsi scatter



Fungsi `scatter()` adalah untuk memplot satu titik pada setiap pengamatan. Fungsi ini membutuhkan dua array dengan panjang yang sama, satu untuk nilai sumbu x, dan satu untuk nilai pada sumbu y.

4. Untuk memplot beberapa distribusi bivariat berpasangan dalam kumpulan data, kita dapat menggunakan fungsi `pairplot`.



Plot berpasangan dibangun di atas dua gambar dasar, histogram dan plot pencar. Histogram pada diagonal memungkinkan kita untuk melihat distribusi satu variabel sementara plot pencar pada segitiga atas dan bawah menunjukkan hubungan (atau ketiadaan) antara kedua variabel. Misalnya, plot paling kiri di baris kedua menunjukkan plot pencar `life_exp` versus `tahun`.



# BAB IV

## STUDI KASUS

---

### A. Pendahuluan

Salah satu hal yang dapat dilakukan untuk mengatasi masalah pesan spam ini adalah dengan melakukan pemblokiran nomor pengirim pesan, tetapi hal ini juga tidak terlalu berefek dalam mengatasi masalah pesan spam. Cara lain yang bisa dilakukan adalah dengan melakukan klasifikasi teks pesan dengan repositori data pesan yang telah ada menggunakan teknik klasifikasi komputasi cerdas. Klasifikasi teks pesan dilakukan untuk membedakan pesan yang berisi spam dan pesan yang tidak berisi spam (ham).

Penelitian ini akan melakukan klasifikasi teks pesan dengan menerapkan algoritma Naïve Bayes karena naïve bayes merupakan salah satu algoritma yang efektif diterapkan untuk melakukan klasifikasi dengan jumlah data yang besar. Proses klasifikasi teks akan dilakukan dengan RapidMiner untuk mencari konfigurasi terbaik Naïve bayes dalam menghasilkan nilai akurasi yang tinggi. Dengan dilakukakannya klasifikasi teks pesan ini maka akan dapat menangani pesan yang terindikasi spam lebih awal sehingga penerima pesan akan membaca pesan yang benar-benar bermanfaat baginya.

### B. Tujuan dan Capaian

Tujuan yang ingin dicapai dalam pembelajaran ini adalah

1. Mahasiswa dapat memberikan contoh masalah terkait konsep prediksi dengan benar.
2. Diberikan suatu masalah, mahasiswa dapat menghasilkan program untuk memprediksi masalah tersebut melibatkan penggunaan Text Mining dalam Python dengan metode Naïve Bayes.
4. Diberikan suatu masalah, mahasiswa dapat menghasilkan program untuk menyelesaikan masalah dengan baik dan hasil prediksi yang layak sehingga dapat dilakukan publikasi.

### C. Uraian Penelitian

Text mining adalah proses mengeksplorasi dan menganalisis data teks tidak terstruktur dalam jumlah besar dengan bantuan perangkat lunak yang dapat mengidentifikasi konsep, pola, topik, kata kunci, dan atribut lain dalam data. Ini juga dikenal sebagai analitik teks, meskipun beberapa orang mengacaukan

kedua istilah tersebut; Dalam pandangan ini, analitik teks mengacu pada aplikasi yang menggunakan teknik penambangan teks untuk menentukan peringkat kumpulan data.

Berkat pengembangan platform data besar dan algoritme pembelajaran mendalam yang dapat menganalisis data tidak terstruktur dalam jumlah besar, penambangan teks menjadi lebih praktis bagi ilmuwan data dan pengguna lainnya.

```
In [18]: sma_bayes = ["hallohallo, transfer"]
pred = text_c[0].predict(sma_bayes)
print("hasil prediksi {}".format(pred))

hasil prediksi: 1
```

Mengenai gambar berikut ini adalah hasil dari text input yang telah di preproceasing, dan proses pengambilan datanya itu dengan berbagai cara yaitu crawling, Kaggle, dan juga log message dari iteung dan ini adalah hasil nya yaitu hasil prediksi 1 bersifat spam.

## D. Latihan

### 4.1 PREDIKSI KALIMAT PESAN MENGGUNAKAN NAÏVE BAYES

setelah mempelajari python, pandas, serta data visualisasi maka selanjutnya kita akan mencoba melakukan prediksi. Sesuai dengan judul buku ini, kita akan mulai mencoba melakukan prediksi dengan menggunakan naïve bayes

1. Untuk lebih mempermudah, download kode dari repositori GitHub buku ini. Lalu buka dengan Jupyter Notebook.
2. Buat file baru terlebih dahulu.
3. Import library yang dibutuhkan.

```
In [1]: import pandas as pd
from collections import Counter
import csv
```

Pandas adalah library open source pada Python yang sering digunakan untuk memproses data yang meliputi pembersihan data, manipulasi data, hingga melakukan analisis data. Ketika melakukan suatu analisis, kita tidak bisa menggunakan data mentah. Fungsi Counter() untuk mendapatkan frekuensi kata dalam buku (berapa kali kata muncul dalam buku). Format CSV (Comma Separated Values) adalah format impor dan ekspor data yang paling umum digunakan untuk data spreadsheet dan database.

4. Karena kita akan menggunakan dataset berupa file csv.

```
In [1]: import pandas as pd
from collections import Counter
import csv
# Import library yang di butuhkan

df = pd.read_csv('text-reviews.csv', engine='python') #membaca file csv nya
text = []
label = []
print(df)
```

5. Maka selanjutnya kita menggunakan fungsi read\_csv. Kemudian kolom Text dan label di parsing agar python bisa membaca kolom tersebut. Dan Hasilnya akan tercipta sebuah dataframe seperti pada gambar.

```
0      [REDACTED] beli paket Piasih mulai 100 di 10 Tahun...  0
1      2.5 00/00 hari hanya Rp 50 Ribu Special! best A...  0
2      2018-08-08 01:47:13 /Pig 100, nisa kanta Piasih ...  0
3      2018-08-08 01:47:47 /Pig 100, nisa kanta Piasih ...  0
4      4.000/00 hari hanya Rp 50 Ribu Special! best an...  0

...      ...      ...      ...
13400      https://www.klikshoppa.com/Order.html  0
13401      https://www.planner-planner.com/Order.php?prod...  0
13402      https://www.klikshoppa.com/Order.html?prod...  0
13403      https://www.klikshoppa.com/Order.html?prod...  0
13404      https://www.klikshoppa.com/Order.html?prod...  0
13405      https://www.klikshoppa.com/Order.html?prod...  0
13406      https://www.klikshoppa.com/Order.html?prod...  0
```

6. Setelah dokumen dibaca maka akan di append Fungsi append ini berguna untuk Menggabungkan baris teks ke akhir label (jumlah kolom harus sama)

```
In [2]: with open('text-reviews.csv', mode='a', encoding='utf-8') as f:
        reader = csv.reader(f)
        for row in reader:
            text.append(row[0])
            label.append(row[1])
```

Cara membaca file CSV, sama saja seperti cara membaca file teks biasa. Bedanya terletak pada cara parsing datanya. Pada teks biasa, kita bisa langsung ambil datanya. Sedangkan di file CSV, kita harus olah lagi dengan library csv.

7. Mengidentifikasi dan mengetahui seberapa panjang jumlah item pada objek

```
In [3]: print('jumlah data: {}'.format(len(text)))
        print('jumlah label: {}'.format(len(label)))

jumlah data: 13406
Counter({'label': 1, 'label': 0})
```

8. untuk mengidentifikasi pada kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna.

```
In [4]: # nltk stopwords

import nltk
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))
text ['-i'] = label['-i'].Format('labela') = [item for item in x if item not in stop_words]]
```

NLTK adalah singkatan dari Natural Language Tool Kit, yaitu sebuah library yang digunakan untuk membantu kita dalam bekerja dengan teks. Library ini memudahkan kita untuk memproses teks seperti melakukan classification, tokenization, stemming, tagging, parsing, dan semantic reasoning.

9. Proses stemming antara satu bahasa dengan bahasa yang lain tentu berbeda. Contohnya pada teks berbahasa inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia semua kata imbuhan baik itu sufiks dan prefiks juga dihilangkan.

```
In [11]: from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize, word_tokenize
nltk.download('punkt')

data="mantI ketemu di mana"
words = word_tokenize(data)
ps = PorterStemmer()
for w in words:
    rootword=ps.stem(w)
    print(rootword)

mantI
ketemu
di mana
I
```

```
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\Jani\AppData\Localing\...
[nltk_data] Package punkt is already up-to-date!
```

10. dari kalimat teks yang telah di stim maka akan data akan di latih dengan train test split itu, kita bagi dataset menjadi data training dan data test dengan proporsi data test sebesar 20% akan di pisah mana data hasil dari percobaan ini adalah 5737 data yang di train dan 1435 test

```
In [8]: import pandas as pd
stem_word = pd.read_csv('test-revella11.csv', delimiter='|')
stem_word = stem_word.sort_values(by='Teks')

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(stem_word['Teks'], stem_word['Label'], test_size=0.2, random_state=100)

print('Data training:')
print(X_train)
#
print('Data testing:')
print(X_test)

Data training:
00400
Data testing:
00000
```

11. Dari kalimat teks yang telah di stem maka akan data akan di latih dengan train test split Setelah itu, kita bagi dataset menjadi data training dan data test dengan proporsi data test sebesar 20% lalu akan di pisah mana data hasil dari percobaan ini adalah 5737 data yang di train dan 1435 test

```
In [9]: from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVecorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB

text_clf = Pipeline([('vect', CountVecorizer()), ('tfidf', TfidfTransformer()), ('clf', MultinomialNB())])

text_clf.fit(X_train, y_train)

Out[9]:
-- Pipeline --
- CountVecorizer
- TfidfTransformer
- MultinomialNB
```

12. Dari hasil eksperimen ini, dapat dilihat bahwa pembagian data berpengaruh terhadap akurasi yang diperoleh dari setiap model. Namun pengaruh pembagian data kecil karena perbedaan akurasi yang diperoleh pada setiap algoritma kecil. Hasil eksperimen menunjukkan bahwa dengan menggunakan pembagian data 80% data training dan 20% data testing dapat disimpulkan akurasi yang paling tinggi sebesar 66.11% diperoleh dengan menggunakan algoritma Naïve Bayes Classifier.

```
In [10]: import numpy as np
pred = text_clf.predict(X_test)
akurasi = np.mean(pred==y_test)
print("Akurasi: {}".format(akurasi))

Akurasi: 0.6611111111111111
```

13. Sekarang kita masukan kalimat yang serupa spam ke dalam kode program nyam maka terdapat hasil prediksi nya 1 yang dimana prediksi 1 itu berjenis spam

```
In [14]: sms_jamv = ["salam kenal"]
pred = text_clf.predict(sms_jamv)
print("hasil prediksi :", format(pred))

hasil prediksi: [1]
```

14. Sekarang kita masukan lagi kalimat yang berupa ham atau non spam ke dalam kode program nym maka terdapat hasil prediksi nya 0 yang dimana prediksi 0 itu berjenis ham atau non spam

```
In [15]: sms_jamv = ["jangan panggil sy"]
pred = text_clf.predict(sms_jamv)
print("hasil prediksi :", format(pred))

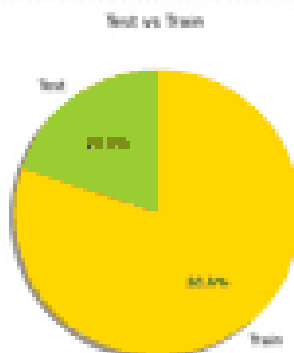
hasil prediksi: [0]
```

15. lalu Ketika kita masukan hasil dari pembagian training dan testing pada baris gambar ke 9 kita masukan di quantity nya dan terdapat hasilnya bahwa 80% data training dan 20% data testing.

```
In [4]: from sklearn.cross_validation import train_test_split
labels = ["spam", "ham"]
quantity = [80, 20]
colors = ["yellowgreen", "yellow"]

pdt.train_test_split(x, y, test_size=0.2)
pdt.plot(quantity, labels=labels, colors=colors, labels=["80.00%", "20.00%"], startangle=90)
pdt.show()

# dan ini lah perantara dari dua nya
```



Setelah dilakukan analisa, perancangan, implementasi, pelatihan algoritma, dan pengujian pada aplikasi klasifikasi sms dengan menggunakan metode Naïve Bayes, maka dapat disimpulkan bahwa aplikasi yang dibuat mampu untuk

mengklasifikasikan sms spam dan bukan spam dan hasil pengujian akurasi 66 % dari 10422 data latih dan 2606 data uji.

#### 4.2 Rangkuman

Dalam penelitian ini data yang diuji adalah SMS yang dimasukkan. Sedangkan yang sedang data pelatihan adalah SMS yang telah disimpan dalam basis data. Untuk mendapatkan hasil klasifikasi pengujian data, pertama-tama aplikasi akan mengambil semua pengujian SMS dari database yang berupa file CSV.

Tapi sebelum perhitungan frekuensi dilakukan, pertama setiap SMS yang ada akan melalui tahapan preprocessing dan transformasi, di mana di tahap preprocessing terjadi perubahan bentuk penulisan dengan huruf kecil (case folding) dan penghapusan karakter khusus (tokenizing), serta pada tahap transformasi SMS yang ada dibersihkan pertama dari kata-kata kurang penting (penghapusan kata berhenti) dan diubah menjadi bentuk kata dasar (stemming). Setelah analisis, desain, implementasi, pelatihan algoritma, dan pengujian pada aplikasi klasifikasi sms dengan menggunakan dengan metode Naïve Bayes, dapat disimpulkan bahwa aplikasi yang dibuat mampu mengklasifikasikan sms spam dan bukan spam dan hasil uji akurasi sebesar 66,11% dari 10422 data latih dan 2606 data uji.

## DAFTAR PUSTAKA

- Matthes, E. (2019). Python Crash Course: A Hands-on, Project-Based Introduction to Programming. 2nd ed. No Starch Press Inc. San Francisco.
- Abdulloh, Rohi. (2015) Web Programing is Easy. Jakarta: PT Elex Media Komputindo.
- Arhami, Muhammad. (2005) Konsep Dasar Sistem Pakar. Yogyakarta: Andi Offset.
- Arief M Rudianto. (2011) Pemrograman Web Dinamis menggunakan PHP dan MySQL. Yogyakarta: Andi Offset.
- Andre. (2013, oktober 01). Pengertian CSS, Apa yang dimaksud dengan CSS. Retrieved 06 12, 2020, from <https://www.duniaikom.com/tutorial-belajar-css-part-1-pengertian-css-apa-yang-dimaksud-dengan-css/>
- Barber, I. 2010. Bayesian Opinion Mining. [Online]. Tersedia di: <http://phpir.com/bayesian-opinion-mining> [diunduh: 15 Juni 2014].
- Feldman, R & Dagan, I. (1995) Knowledge discovery in textual databases (KDT). Dalam Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95), Montreal, Canada, August 20-21, AAAI Press, 112-117.
- Feldman, R & Sanger, J. (2007) The Text Mining Handbook-Advanced Approaches in Analyzing Unstructured Data, USA: New York.
- Go, Alec; Bhayani, Richa; Huang, Lei. (2009) Twitter Sentimen Classification using Distant Supervision.
- H, Azizul, "Berkenalan dengan scikit-learn (Part 1) – Preparations," hkaLabs, 2018. [Online]. Available: <https://hakim-azizul.com/berkenalan-dengan-scikitlearn/>.
- M. Sharma, "A Survey of Email Spam Filtering Methods," vol. 7, pp. 14–21, 2018.
- Hall, J, A,. (2004) Sistem Informasi Akuntansi. Jakarta: Penerbit Salemba Empat.
- Hearst, M. A. (1997) Text data mining: Issues, techniques, and the relationship to information access. Presentation notes for UW/MS workshop on data mining, July 1997.
- Hidayatullah, dan Kawistara. (2015) Pemrograman Web. Bandung: Informatika.
- Hoffer. (2002) dalam Laporan Kerja Praktik 2 Sistem Monitoring Pendistribusian Surat oleh Dwi Oktarina Sari. (2006).
- Ilham, M. (2020) Pengertian Python, Fungsi, Kelebihan dan Kekurangan. Retrieved from [materibelajar.co.id: https://materibelajar.co.id/pengertianpython/](https://materibelajar.co.id/pengertianpython/)
- Jhonsen. (2004) WEB DESIGNER untuk PEMULA. Jakarta : Elex Media Komputindo Kelompok Gramedia. Anggota IKAPI.



- Jogiyanto. (2008) *Metodologi Penelitian Sistem Informasi*. Yogyakarta: Andi Offset.
- Wahyuningsih, S., & Utari, D. R. (2018). Perbandingan Metode K-Nearest Neighbor, Naïve Bayes dan Decision Tree untuk Prediksi Kelayakan Pemberian Kredit. *Konferensi Nasional Sistem Informasi (KNSI) 2018*.
- Kusrini. (2008) *Aplikasi Sistem Pakar*. Yogyakarta: Andi Offset.
- Lamurias, Andre Francisco.M. 2014. *Identifying Interactions Between Chemical Entities in Text*. *Desertation Universidade De Lisboa*.
- M. K. Albert Verasius Dian Sano, S.T., "DEFINISI, KARAKTERISTIK, DAN MANFAAT DATA MINING -SERI DATA MINING FOR BUSINESS INTELLIGENCE (2)," BINUS UNIVERSITY, 2019. [Online]. Available: <https://binus.ac.id/malang/2019/01/definisi-karakteristikdan-manfaat-data-mining-seri-data-mining-for-businessintelligence-2/>.
- McLeon. (2008) *Sistem Informasi Manajemen*. Jakarta: Salemba Empat.
- Mudjahidin, & Putra, N. P. (2010) Rancang Bangun Sistem Informasi Monitoring Perkembangan Proyek Berbasis Web Studi Kasus Di Dinas Bina Marga dan Pemantusan. *Jurnal Teknik Industri Vol.11 No.1*, 75- 83.
- Romney, M., Steinbart, P, J,. (2005) *Accounting Information System 9<sup>th</sup> Edition*. Jakarta: Penerbit Salemba Empat.
- S. Kumar and N. Shah, False Information on Web and Social Media: A Survey. 1, 1 (April 2018), <https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>.
- Rizqiyani, V., Mulwinda, A., & Putri, R. D. M. (2017). Klasifikasi judul buku dengan algoritma naive bayes dan pencarian buku pada perpustakaan jurusan teknik elektro. *J. Tek. Elektro*, 9(2), 60-65.
- Triawati, Candra; Bijaksana, M.Arif; Indrawati, Nur; Saputro, Widyanto Adi. (2009) *Pemodelan Berbasis Konsep Untuk Kategorisasi Artikel Berita Berbahasa Indonesia*. Dalam *Seminar Nasional Aplikasi Teknologi Informasi 2009*
- Twitter, 04 Maret (2016), Twitter API, <https://dev.twitter.com/overview/documentation>
- V.L. Rubin, Y. Chen and N.J. Conroy, (2015) Deception detection for news: three types of fakes. *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*, American Society for Information Science, 83.
- V.L., Rubin. (2016) Deception detection and rumor debunking for social media, *The SAGE Handbook of Social Media Research Methods*: 342-364.
- William P. Wagner. (2017) *Trends in Expert System Development : A Longitudinal Content Analysis of Over Thirty Years of Expert System Case Studies*. *Expert System With Applications*, 85-96.

## TENTANG PENULIS



Zaky Muhammad Yusuf Lahir di Tangerang pada 2 Maret 2001. Ia merupakan mahasiswa aktif di Universitas Logistik dan Bisnis Internasional Bandung. Sebelumnya ia telah lulus dari SMK Diponegoro 1 Jakarta Timur.

Semasa Kuliah, Zaky sangat aktif dalam mengikuti kegiatan kelas akademik dan UKM di kampus. Diantaranya seperti MAPALA atau Mahasiswa Pecinta Alam, selain itu ia juga telah melaksanakan magang di

Direktorat Jenderal Bea dan Cukai dan divisi Informasi Kepabeanan dan Cukai.

Rolly Maulana Awangga adalah seorang profesional IT di bidang Machine



Learning, Artificial Intelligence, dan juga Data Science. Beliau adalah pengembang GIS “Agenda Presiden” dan “Sistem Database Dukungan Kebijakan Nasional” di Sekretariat Negara Republik Indonesia. Pada tahun 2018 Beliau mendukung implementasi SpaceMap(GIS Based Application) di LAPAN. Lahir di Indramayu tahun 1986. Menempuh pendidikan di Indramayu sejak TK hingga SMP. Setelah lulus SMPN 2, beliau melanjutkan ke SMA di SMAN 2 Cirebon. Aktif di organisasi intra sekolah (OSIS) dan asisten pelatih pencak silat Indonesia Merpati Putih. Tim Pelopor dan pembuat logo

dewan keamanan sekolah SMA Negeri 2 Cirebon. Setelah lulus SMA, mengejar program S1 Teknik Informatika di (Sekolah Tinggi Teknologi Telkom) STT Telkom pada tahun 2004. Banyak dari organisasi dan kepanitiaan, dari internal hingga eksternal kampus. Aktif dan beberapa agensi adalah: 1. Ketua Komunitas Riset Informatika 2. Ketua Kelompok Pemakai Linux STT Telkom 3. Bendahara Klub Linux Bandung 4. Advokasi Ubuntu Indonesia, Bagian Bandung 65 5. Asisten praktikum Sistem Operasi 6. Asisten Dosen Interaksi Manusia-Komputer 7. Anggota Aktif Blogger Kota Bandung 8. Relawan TIK Pelaksana Relawan TIK Jabar Semasa kuliah (2008-2010) bekerja di divisi marketing dan sales Telkom Divre III Jabar Banten di bagian Competitive Intelligence sebagai Community Content Developer. Setelah lulus sarjana pada tahun 2010, ia langsung mendapat

beasiswa dari Fakultas Informatika untuk melanjutkan S2 Teknik Informatika, Institut Teknologi Telkom (IT Telkom). Tesisnya berbicara tentang Analisis Jaringan Sosial pada Desain Ontologi. Selama mengikuti program magister, pernah menjadi dosen Jurusan Manajemen Informatika Politeknik Telkom selama satu semester sekaligus menjadi dosen profesional Fakultas Teknologi Informasi Institut Teknologi Telkom selama satu tahun bahkan bertindak sebagai Account Manager IT Project di bawah Vice. Rektor Kerjasama. Setelah lulus S2 pada tahun 2013, untuk memenuhi passionnya sebagai seorang profesional di bidang IT, mendirikan perusahaan dan menjadi Expert di beberapa IT Consultant di Bandung. Bersamaan dengan itu, komunitas Saung IT juga mendirikan organisasi nirlaba untuk pengembangan pendidikan IT di kota tersebut. Menjadi jurnalis di Pikiran Rakyat dan Relawan TIK Jawa Barat. Beliau juga mengembangkan program Open Source Project bersama Anton Rahardja (VoIP-ID) pada tahun 2015. Dosen Program Sarjana Terapan Teknik Informatika Politeknik Pos Indonesia sejak September 2015. Koordinator atau Komunitas Riset Informatika ([irc.poltekpos.ac.id](http://irc.poltekpos.ac.id)) dan Tim Hak Kekayaan Intelektual Poltekpos. Juga, Reviewer di Jurnal Elkomika (Akreditasi Kemendikbud) cetak ISSN 2338-8323, ISSN elektronik 2459-9638. Aktif dalam beberapa kegiatan International Reputable Conference sebagai Session Chair pada ICITISEE 2007 (Amikom Jogjakarta dan IEEE) dan Mecnit 2017 (Unprim dan IOP). Reviewer di ICAE 2018 (Polibatam dan IEEE), Transaksi IEEE, Jurnal Terakreditasi Nasional. Saat ini aktif sebagai Ph.D. mahasiswa Teknik Biomedik STEI ITB bersama Prof. Tati LR. Mengko. Decoding otak sebagai topik penelitian

Buku ini berisi tentang pembelajaran untuk melakukan prediksi pesan spam sms menggunakan naïve bayes. Buku ini akan diawali dengan pembelajaran Python dasar terlebih dahulu lalu diikuti dengan library pandas, data visualization, hingga ke studi kasus yaitu melakukan prediksi pesan spam sms menggunakan naïve bayes. Proses prediksi ini menggunakan bahasa pemrograman Python dan Jupyter Notebook sebagai Coding Environment.

