

# Data Preprocessing

Presented by



## Daftar Isi

A. Feature Engineering	03
B. Exploratory Data Analysis (EDA)	33
C. Categorical and Numerical Analysis	67
D. Automated Exploratory Data Analysis (EDA)	80
E. Reproducible Data Analysis	89
F. Data Storytelling and Communication	101

# Prologue

Sebelumnya kita sudah membahas fundamental Data Science mulai dari konsep dasar, computational thinking, statistical thinking, hingga digital leadership. Kini kita akan mencoba mengulik salah satu aktivitas sederhana namun memiliki peranan penting dalam Data Science, yaitu data preprocessing.

Aktivitas ini sebenarnya adalah hal mendasar proses siklus Data Science namun sering diabaikan loh! Padahal aktivitas ini bisa mendapatkan informasi berharga seperti pemahaman data melalui visualisasi, membantu menentukan proses analisis yang tepat terhadap suatu tipe data, hingga bisa meningkatkan performa model machine learning. Modul kali ini kita akan mempelajari secara lengkap proses penerapan data preprocessing seperti:

- I. Feature Engineering
- II. Exploratory Data Analysis (EDA)
- III. Categorical and Numerical Data Analysis
- IV. Automated Exploratory Data Analysis
- V. Reproducible Data Analysis
- VI. Data Storytelling and Communication

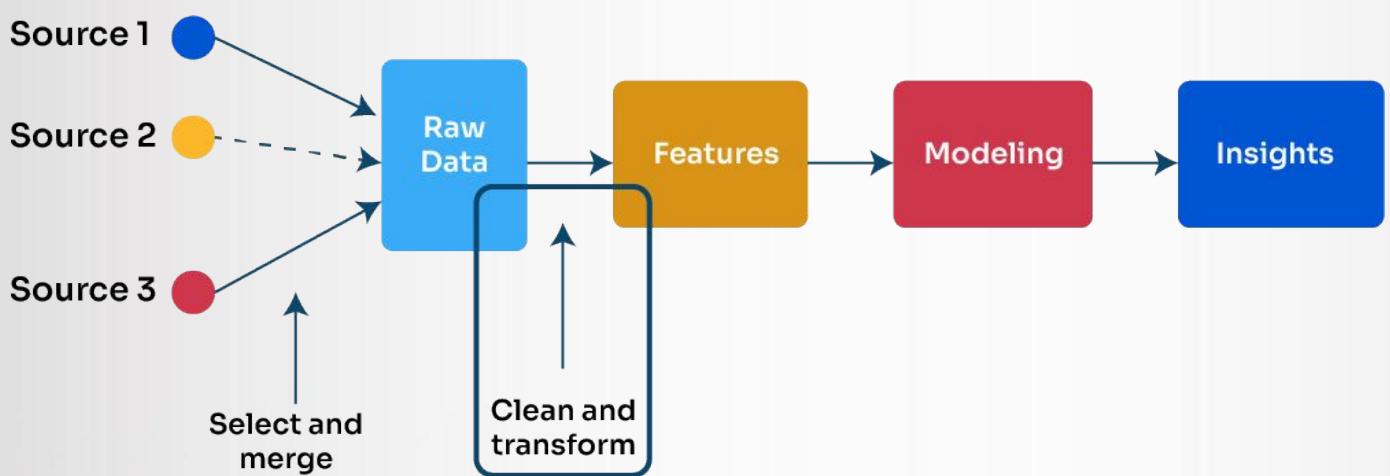
Harapannya setelah selesai mempelajari topik kedua ini kalian bisa memahami tahapan yang perlu dilakukan dalam proses Data Preprocessing dan tools yang digunakan.

# Apa itu Feature Engineering?

Feature engineering adalah topik mendasar dalam machine learning, namun sering diabaikan.

**Feature engineering adalah proses pemilihan, manipulasi, dan transformasi data mentah menjadi fitur yang dapat digunakan.** Proses ini diperlukan agar machine learning bekerja dengan baik pada tugas-tugas baru. Sederhananya, feature engineering adalah tindakan mengubah pengamatan yang bersifat mentah menjadi fitur yang diinginkan menggunakan pendekatan statistik atau machine learning.

Feature engineering juga **membantu merepresentasikan masalah mendasar ke model prediktif** dengan cara yang lebih baik, yang sebagai hasilnya, meningkatkan akurasi model untuk data yang tidak terlihat. Model prediktif berisi variabel prediktor dan variabel hasil, dan sementara proses feature engineering memilih variabel prediktor yang paling berguna untuk model tersebut.





Secara umum, semua algoritma machine learning mengambil data input untuk menghasilkan output. Data input tetap dalam bentuk tabel yang terdiri dari baris (instance atau pengamatan) dan kolom (variabel atau atribut) dimana atribut ini sering dikenal sebagai fitur.

Seperti yang sudah kita bahas sebelumnya, feature engineering dilakukan supaya model machine learning dapat bekerja dengan baik. Terdapat beberapa aktivitas yang dapat dilakukan seperti feature creation, transformation, feature extraction, dan feature selection. Kali ini, kita akan membahas satu per satu serta contoh implementasinya di python. Pertama, kita masukkan terlebih dahulu dataset dimana kita akan memakai dataset Titanic yang disediakan package Seaborn. Masukkan script ini untuk memasukkan dataset



```
# importing library
import seaborn as sns

df = sns.load_dataset("titanic")
```

## 1. Feature Creation

Proses ini biasanya **berupa pembuatan fitur baru yang diperoleh dari penjumlahan, pengurangan, atau perhitungan rasio** sehingga fitur baru ini memiliki fleksibilitas yang tinggi. Oleh karena itu, proses ini **sangat membutuhkan kreativitas** dan intervensi manusia.



Kali ini kita akan mencoba membuat fitur baru dengan melakukan pengelompokan fitur ‘age’ dan ‘fare’. Kita bisa menggunakan fungsi `.qcut()` yang disediakan oleh package Pandas untuk mengelompokkan data seperti script di bawah ini. Fungsi `.qcut()` akan mendiskritkan variabel ke dalam kelompok data dalam ukuran yang sama. Oleh karena itu, kita harus mendefinisikan barisan data yang ingin dikelompokkan dan berapa banyak kelompok yang ingin dibutuhkan.

```
● ● ●

# Importing the Library
import pandas as pd

# Bin the feature
df["age_bins"] = pd.qcut(df["age"], 16)
df["fare_Bins"] = pd.qcut(df["fare"], 10)
```

Berdasarkan script berikut, kita akan mengelompokkan fitur ‘age’ sebanyak 16 dan ‘fare’ sebanyak 10

```
# Importing the Library
import pandas as pd

# Bin the feature
df["age_bins"] = pd.qcut(df["age"], 16)
df["fare_Bins"] = pd.qcut(df["fare"], 10)
```

Berikut adalah hasil pengelompokan untuk fitur ‘age’. Kolom ‘age\_bins’ berisi informasi data tersebut tergolong ke dalam rentang nilai tertentu. Misalkan pada baris pertama, yaitu umur 22 tergolong ke dalam kelompok umur 20,125 hingga 22

age
22.0
38.0
26.0
35.0
35.0

age_bins
(20.125, 22.0]
(35.0, 38.0]
(24.0, 26.0]
(32.312, 35.0]
(32.312, 35.0]

Sebelum Pengelompokan

Sebelum Pengelompokan

## 2. Transformation

Melibatkan penyesuaian atau transformasi variabel prediktor untuk meningkatkan akurasi dan kinerja model.

Misalnya, memastikan bahwa modelnya fleksibel untuk menerima masukan dari berbagai data; itu memastikan bahwa semua variabel berada pada skala yang sama sehingga membuat model lebih mudah dipahami. Hal ini juga memastikan semua fitur berada dalam kisaran yang dapat diterima untuk menghindari kesalahan komputasi.

Salah satu transformasi yang bisa dilakukan adalah normalisasi data. Kita akan melakukannya menggunakan fungsi **MinMaxScaler()** yang disediakan package Sklearn. Fungsi tersebut akan mengubah rentang nilai suatu data menjadi antara 0 hingga 1.



Kita akan mencoba melakukan normalisasi salah satu fitur data Titanic, yaitu ‘fare’ seperti di gambar di samping.

fare
7.2500
71.2833
7.9250
53.1000
8.0500

Tuliskan script berikut untuk me-import fungsi MinMaxScaler() yang disediakan oleh Sklearn



```
# Importing the Library
from sklearn.preprocessing import MinMaxScaler
```

Definisikan variable scaler seperti gambar di bawah. Kita akan mendefinisikannya sebagai ‘scaler’

Selanjutnya, panggil fungsi **.fit\_transform()** dengan nilai parameter kolom atau fitur yang akan kita normalisasi, yaitu dengan menuliskan df[['fare']]. Apabila ingin menambahkan fitur lain seperti ‘age’, maka kita bisa menuliskannya dengan df[['age', 'fare']]



```
# Define the Scaler Variable
scaler = MinMaxScaler()
fare_standarized = scaler.fit_transform(df[['fare']])

# Replacing the fare column with the normalized data
df['fare'] = fare_standarized
df
```

Keluaran fungsi `.fit_transform()` adalah suatu baris bilangan yang berisi nilai hasil normalisasi

```
array([[0.01415106],  
       [0.13913574],  
       [0.01546857],  
       [0.1036443 ],  
       [0.01571255],  
       [0.0165095 ],  
       [0.10122886],  
       [0.04113566],  
       [0.02173075],  
       [0.05869429],  
       [0.03259623],  
       [0.05182215],
```

Berikut adalah perbandingan fitur ‘fare’ sebelum dan sesudah normalisasi data

	<b>fare</b>
	7.2500
	71.2833
	7.9250
	53.1000
	8.0500

**Sebelum Normalisasi**

	<b>fare</b>
	0.014151
	0.139136
	0.015469
	0.103644
	0.015713

**Sesudah Normalisasi**

Dokumentasi fungsi MinMaxScaler bisa dicek [disini](#) ya!

### 3. Feature Extraction

Tujuan utama langkah ini adalah **mengurangi dimensi data dalam bentuk mengurangi kolom atau atribut pada data** sehingga dapat dengan mudah digunakan dan dikelola untuk pemodelan data.

Salah satu contoh feature extraction meliputi analisis kluster, analitik teks, Principal Component Analysis (PCA), atau kombinasi fitur lainnya.

Salah satu feature extraction yang bisa dilakukan adalah ekstraksi fitur terkait apakah seorang penumpang tersebut bepergian sendirian atau bersama kerabatnya. Dataset Titanic memiliki fitur ‘sibsp’ dan ‘parch’ yang mengandung informasi jumlah saudara kandung atau pasangan dan jumlah orang tua atau anak tiap penumpang seperti di gambar berikut.



sibsp	parch
1	0
1	0
0	0
1	0
0	0

Kita akan membentuk fitur “pembantu” dimana berisi informasi jumlah keluarga atau kerabat penumpang tersebut. Sebagai contoh, kita menamakan fitur tersebut dengan ‘family\_size’ seperti berikut:



```
df['family_size'] = (df.sibsp + df.parch + 1)  
df
```



Keluaran  
program

family_size
2
2
1
2
1

Selanjutnya, fitur ‘family\_size’ akan kita gunakan sebagai pembentuk fitur baru, yaitu ‘is\_alone’. Apabila nilai dalam fitur ‘family\_size’ lebih dari 0, maka ‘is\_alone’ adalah 1. Sebaliknya, apabila nilainya sama dengan 0, maka nilainya adalah 0



```
df['is_alone'] = (df.family_size == 0).astype('int')
```



Keluaran  
program

is_alone
0
0
0
0
0

#### 4. Feature Selection

Feature Selection adalah **cara memilih subset fitur yang paling relevan** dari kumpulan fitur asli dengan menghapus fitur yang redundan, tidak relevan, atau berisik.

Jika memasukkan semua fitur yang redundan dan tidak relevan bisa berdampak negatif seperti penurunan kinerja model secara keseluruhan.

Oleh karena itu, sangat penting untuk mengidentifikasi dan memilih fitur yang paling sesuai.



Salah satu cara melakukan feature selection adalah memilih fitur yang tidak terindikasi data leakage.

Data leakage adalah fitur yang membuat model mengetahui karakteristik data kita seperti apa. Selain itu, dapat menyebabkan model prediksi yang dibangun menjadi sebuah model yang terlalu “bagus” dimana ini bukan indikasi yang baik. Mengapa? karena sama seperti kita mau ujian tapi sudah tau kunci jawabannya.

Data leakage adalah informasi dari luar dataset pelatihan digunakan untuk membuat model. Informasi tambahan ini dapat memungkinkan model untuk mempelajari atau mengetahui sesuatu yang sebelumnya tidak akan diketahui dan pada gilirannya membantalkan perkiraan kinerja model yang sedang dibangun.

Salah satu cara memeriksa fitur yang terindikasi data leakage adalah melakukan pemeriksaan terhadap tiap fitur yang bermakna sama. Yuk kita coba pada dataset Titanic!

survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

## Fitur ‘Survived’ dan ‘Alive’

Seperti yang kita tahu, dataset Titanic merupakan data yang berisi informasi apakah suatu penumpang selamat atau tidak dari kecelakaan naas tersebut. Namun dataset tersebut terdapat dua fitur yang memiliki informasi yang sama, yaitu ‘survived’ dan ‘alive’. Kedua fitur tersebut memiliki makna yang sama, yaitu apakah penumpang dinyatakan selamat atau tidak.

Penggunaan kedua fitur tersebut akan membuat model machine learning mengetahui “contekan” dalam membangun model prediksi suatu penumpang. Mengapa? karena model sudah memiliki informasi yang merepresentasikan apakah suatu penumpang dinyakan selamat atau tidak. Oleh karena itu, kita cukup menggunakan salah satu fitur seperti ‘survived’.

**survived alive**

0 no

1 yes

1 yes

1 yes

0 no

Silahkan gunakan script berikut untuk menghilangkan fitur ‘alive’



```
df = df.drop(columns = 'alive')  
df
```

## Fitur ‘pclass’ dan ‘class’

Selain itu, terdapat dua fitur lain yang memiliki informasi yang sama, yaitu ‘pclass’ dengan ‘class’ dan ‘embarked’ dengan ‘embark\_town’. Terlihat di dua gambar di bawah kalau keempat fitur ini memiliki informasi yang sejenis. Oleh karena itu, kita cukup menggunakan salah satu saja seperti ‘pclass’ dan ‘embark\_town’

pclass	class	embarked	embark_town
3	Third	S	Southampton
1	First	C	Cherbourg
3	Third	S	Southampton
1	First	S	Southampton
3	Third	S	Southampton

Silahkan gunakan script berikut untuk menghilangkan fitur ‘class’ dan ‘embarked’



```
df = df.drop(columns = 'class', 'embarked')
df
```

Dataset hasil proses feature selection adalah seperti berikut

survived	pclass	sex	age	sibsp	parch	fare	who	adult_male	deck	embark_town	alone
0	3	male	22.0	1	0	7.2500	man	True	NaN	Southampton	False
1	1	female	38.0	1	0	71.2833	woman	False	C	Cherbourg	False
1	3	female	26.0	0	0	7.9250	woman	False	NaN	Southampton	True
1	1	female	35.0	1	0	53.1000	woman	False	C	Southampton	False
0	3	male	35.0	0	0	8.0500	man	True	NaN	Southampton	True



Kinerja model machine learning bergantung pada bagaimana kita melakukan pra-pemrosesan data dan penanganan data. Jika kita membuat model tanpa pra-pemrosesan atau penanganan data, maka berpotensi tidak memberikan akurasi atau kinerja model yang baik.

Padahal, jika kita menerapkan feature engineering pada model yang sama, maka akurasi model tersebut akan meningkat. Jadi, sebisa mungkin kita tidak melewatkkan aktivitas ini sebelum membangun model machine learning ya!

# Pentingnya Feature Engineering

Di bawah ini adalah beberapa poin yang menjelaskan perlunya feature engineering:

## 1. Fleksibilitas

Dalam machine learning, kita selalu berusaha memilih model yang optimal untuk mendapatkan hasil yang baik. Namun, terkadang setelah memilih model yang salah, tetap saja kita bisa mendapatkan prediksi yang lebih baik, dan ini karena fitur yang lebih baik.

## 2. Model yang Lebih Sederhana

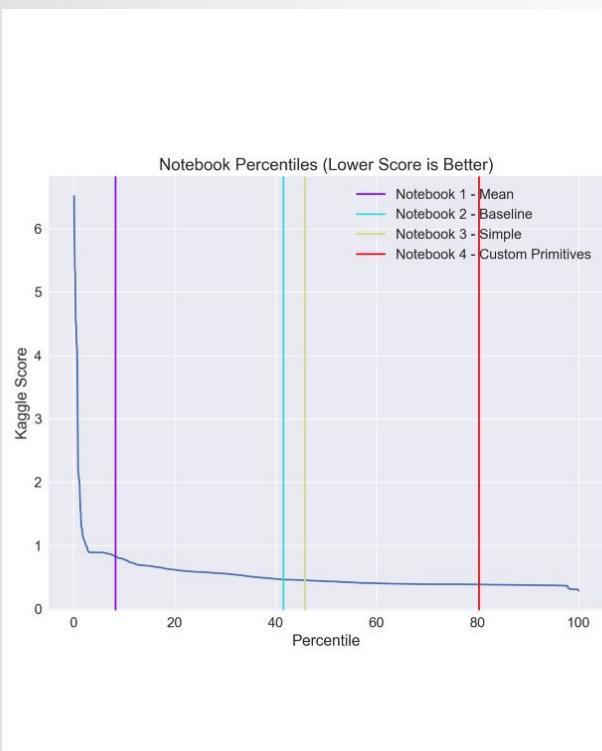
Jika kita memasukkan fitur yang direkayasa dengan baik ke model kita, bahkan setelah memilih parameter yang salah (Tidak terlalu optimal), kita dapat memperoleh hasil yang baik.

Setelah feature engineering, tidak perlu bersusah payah untuk memilih model yang tepat dengan parameter yang paling optimal. Jika kita memiliki fitur yang bagus, kita dapat merepresentasikan data lengkap dengan lebih baik dan menggunakannya untuk mengkarakterisasi masalah yang diberikan dengan sebaik-baiknya.

## 3. Hasil yang Lebih Baik

Seperti yang sudah dibahas, dalam machine learning, sebagai data yang akan kita berikan akan mendapatkan output yang sama. Jadi, untuk mendapatkan hasil yang lebih baik, kita harus menggunakan fitur yang lebih baik.

# Beberapa Tools Feature Engineer Terbaik



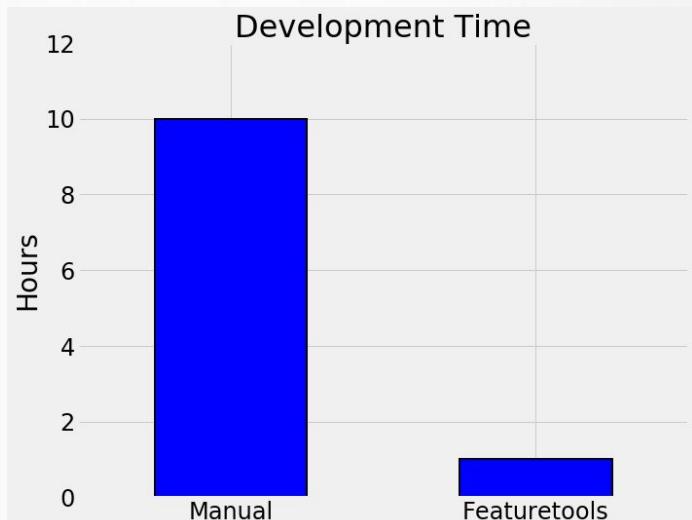
## 1. FeatureTools

Sebuah **framework feature engineering otomatis** seperti mengubah kumpulan data temporal dan relasional menjadi matriks fitur.

Featuretools terintegrasi dengan machine learning pipeline-building tools yang sudah kita miliki. Dalam sepersejadian waktu yang diperlukan untuk melakukannya secara manual, dapat memuat dalam pandas dataframes dan secara otomatis membuat fitur yang signifikan.

Tools ini memiliki beberapa keunggulan seperti:

- Lebih efisien dari segi waktu (bisa mencapai 10x)
- Performa prediktif yang lebih baik
- Fitur yang dapat ditafsirkan dengan signifikansi dunia nyata



Sebelum menggunakan FeatureTools di python, tuliskan script berikut untuk memasukkan package featuretools pada notebook python:

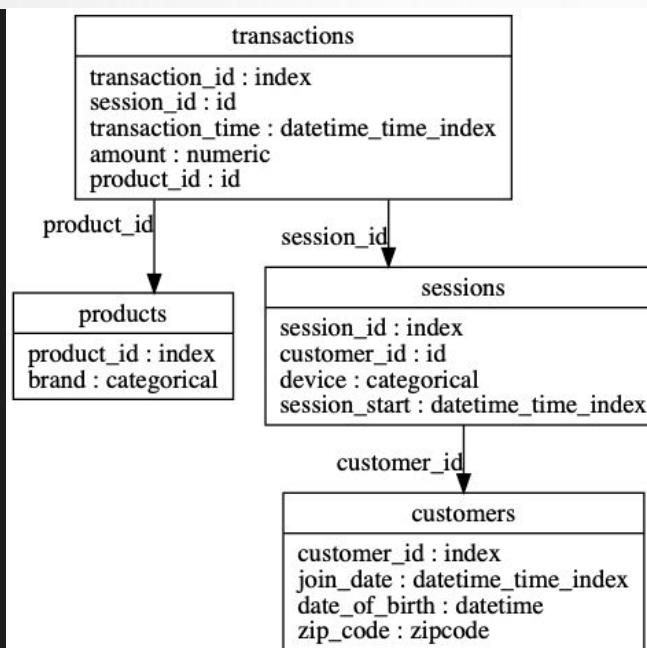
```
# Installing the library
!pip install featuretools

# Importing the Library
import featuretools as ft
```

Kali ini, kita akan menggunakan dataset yang disediakan oleh package featuretools untuk mendemonstrasikan tools ini. Disini kita memiliki tiga buah tabel, yaitu:

- Customer yang merepresentasikan profil dari customer
- Session yang merepresentasikan sesi atau aktivitas yang dilakukan oleh customer di aplikasi
- Transaction yang merepresentasikan transaksi yang dilakukan customer

Kita akan membuat fitur baru yang nantinya dapat digunakan untuk melakukan analisa atau membuat model machine learning.



Tuliskan script berikut untuk memasukkan data yang disediakan oleh featuretools. Kita akan mendefinisikan data yang disediakan oleh package dengan variabel ‘data’. Variabel tersebut merupakan sebuah dictionary yang berisi beberapa tabel seperti customers, sessions, transactions, dan product.

```
# Loading data
data = ft.demo.load_mock_customer()

# Store the data into a variable
cust_df = data["customers"]
session_df = data["sessions"]
transaction_df = data["transactions"]
```

Oleh karena itu, kita membutuhkan sebuah variabel baru yang dapat menyimpan tabel customers, sessions, atau transactions untuk memudahkan proses analisa. Dapat dilihat pada gambar di atas bahwa kita membuat variabel baru seperti:

- cust\_df untuk melihat data customers
- session\_df untuk melihat data session
- transaction\_df untuk melihat data transaction

Gambar di bawah merupakan sampel data customers yang diwakili oleh variabel ‘cust\_df’.

customer_id	zip_code	join_date	birthday
1	60091	2011-04-17 10:48:33	1994-07-18
2	13244	2012-04-15 23:31:04	1986-08-18
3	13244	2011-08-13 15:42:34	2003-11-21
4	60091	2011-04-08 20:08:14	2006-08-15
5	60091	2010-07-17 05:27:50	1984-07-28

## 1. Inisiasi DataFrame

Pertama, kita bisa membuat sebuah dictionary yang berisi tuple dimana mengandung informasi dataframe atau tabel yang akan kita gunakan. Silahkan tulis script di bawah ini. Apabila kita perhatikan, setiap key dan value dictionary adalah sebagai berikut:

- Key bernama “customers”

Menyimpan sebuah tuple yang berisi tabel customer yang diwakili variabel ‘cust\_df’ dan id customer

- Key bernama “sessions”

Menyimpan sebuah tuple yang berisi tabel sessions yang diwakili variabel ‘session\_df’ beserta id dan waktu dimulainya session

- Key bernama “transactions”

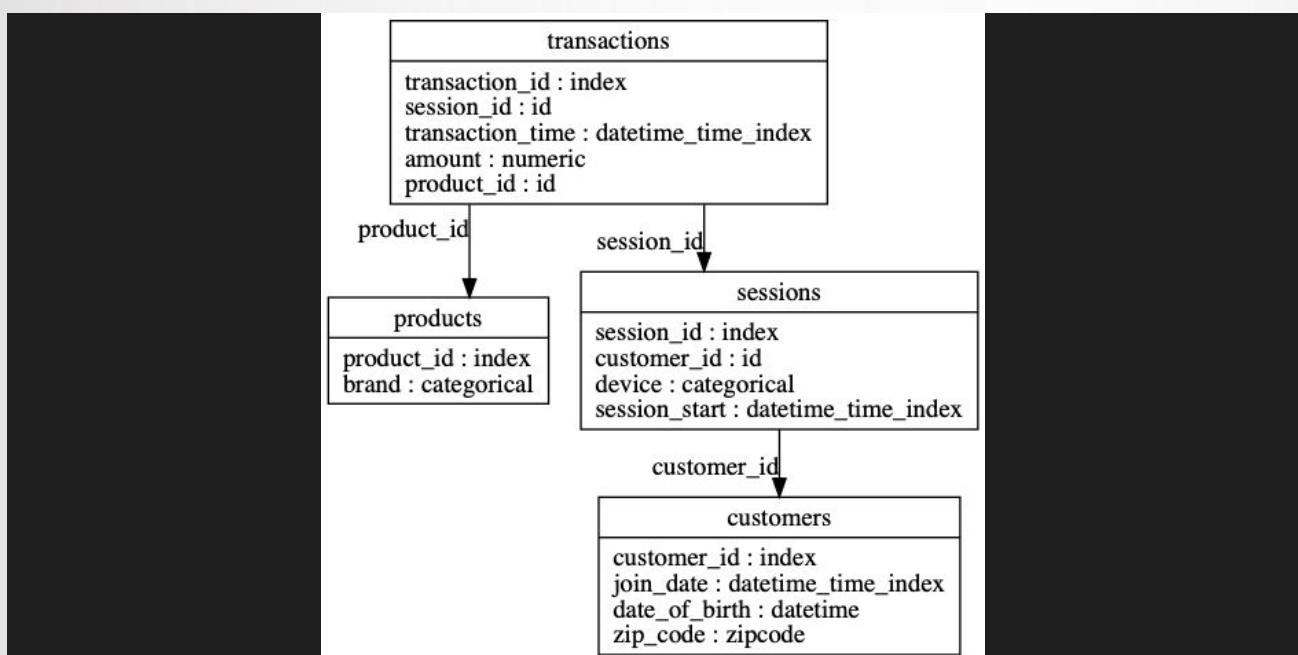
Menyimpan sebuah tuple yang berisi tabel transaksi yang diwakili variabel ‘transaction\_df’ dan waktu terjadinya transaksi



```
# Initiates the Dataframe
dataframes = {
    "customers": (cust_df, "customer_id"),
    "sessions": (session_df, "session_id", "session_start"),
    "transactions": (transaction_df, "transaction_id",
    "transaction_time"),
}
```

## 2. Inisiasi Relationship

Selanjutnya, kita membuat sebuah array yang merepresentasikan relationship atau hubungan antar tabel. Coba perhatikan gambar berikut, setiap transaksi memiliki catatan session tersendiri dimana terlihat tabel transaction memiliki session\_id dimana mewakili id sebuah session. Setiap session memiliki catatan mengenai id customer yang diwakili oleh customer\_id, tipe device, dan waktu terjadinya suatu session yang diwakili oleh session\_start. Terlihat juga setiap customer memiliki waktu mereka bergabung, tanggal lahir, dan kode pos.



Silahkan tulis script berikut untuk membuat relationship antar tabel

```

relationships = [
    ("sessions", "session_id", "transactions", "session_id"),
    ("customers", "customer_id", "sessions", "customer_id"),
]
  
```

### 3. Membuat Features

Sejatinya, konsep FeatureTools adalah “Deep Feature Synthesis”, yaitu menumpuk transformasi dan agregasi beberapa fitur untuk membuat sebuah fitur baru. Transformasi fitur merupakan teknik mengoperasikan satu atau dua fitur secara bersamaan melalui perhitungan selisih antara dua fitur. Sementara itu, agregasi merupakan teknik mengelompokkan beberapa fitur melalui melihat minimum, maksimum, atau nilai rata-rata.

Silahkan tulis script berikut untuk membuat fitur baru menggunakan FeatureTools. Kita akan menggunakan fungsi `.dfs()` yang disediakan oleh package FeatureTools untuk membuat fitur baru berdasarkan tabel dan relationship antar tabel.

```
● ● ●  
feature_matrix_customers, features_defs = ft.dfs(  
    dataframes=dataframes,  
    relationships=relationships,  
    target_dataframe_name="customers",  
)  
  
feature_matrix_customers
```

Terlihat kalau tabel customer hanya terdiri dari empat atribut seperti customer\_id, zip\_code, join\_date, dan birthday. Hasil implementasi FeatureTools merupakan atribut-atribut baru yang terdiri dari berbagai pengoperasian transformasi dan agregasi.

	<b>customer_id</b>	<b>zip_code</b>	<b>join_date</b>	<b>birthday</b>
	1	60091	2011-04-17 10:48:33	1994-07-18
	2	13244	2012-04-15 23:31:04	1986-08-18
	3	13244	2011-08-13 15:42:34	2003-11-21
	4	60091	2011-04-08 20:08:14	2006-08-15
	5	60091	2010-07-17 05:27:50	1984-07-28

Atribut lama

	<b>zip_code</b>	<b>COUNT(sessions)</b>	<b>MODE(sessions.device)</b>	<b>NUM_UNIQUE(sessions.device)</b>	<b>COUNT(transactions)</b>	<b>MAX(transactions.amount)</b>
	<b>customer_id</b>					
	1	60091	8	mobile	3	126
	2	13244	7	desktop	3	93
	3	13244	6	desktop	3	93
	4	60091	8	mobile	3	109
	5	60091	6	mobile	3	79

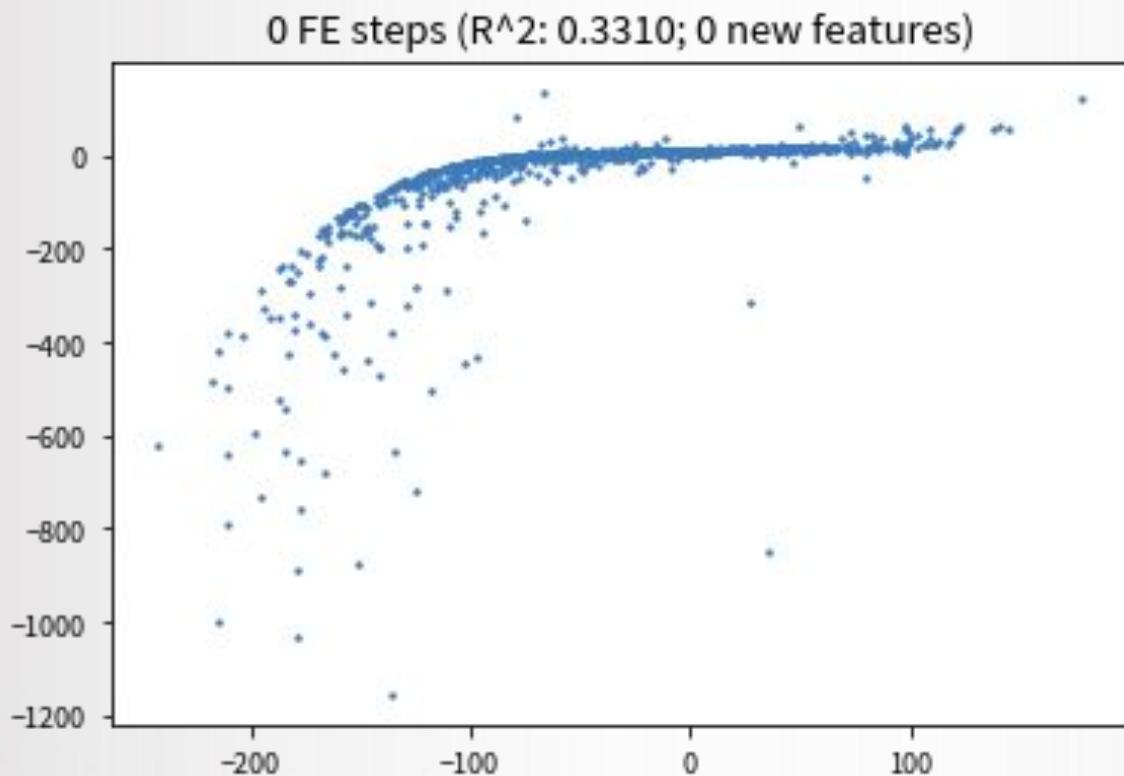
5 rows × 75 columns

Atribut baru hasil implementasi FeatureTools

## 2. AutoFeat

AutoFeat merupakan salah satu **package python yang menyediakan fitur rekayasa dan pemilihan fitur secara otomatis** melalui model regresi. Berbeda halnya dengan FeatureTools, package ini hasil implementasinya dalam sebuah tabel. Selain itu, package ini **memungkinkan kita memilih unit variabel input** untuk menghindari pembuatan fitur yang tidak masuk akal secara fisik.

**Package ini dikenal mudah digunakan dan diinterpretasikan** walaupun penggunanya tidak memiliki latar belakang ilmu statistik. Walaupun begitu, tools ini memiliki kekurangan pada beresiko kurang teliti dalam membaca semua fitur penting dan rawan tidak bisa mendeteksi missing value.



Ref:

[Guide To Automatic Feature Engineering Using AutoFeat \(analyticsindiamag.com\)](http://analyticsindiamag.com/guide-to-automatic-feature-engineering-using-autofeat/)

Kali ini, kita akan mencoba salah satu fungsi AutoFeat, yaitu pemilihan fitur otomatis. Coba tuliskan script berikut untuk memasukkan package AutoFeat pada notebook python:

```
# Installing the library
!pip install autofeat
from autofeat import FeatureSelector
```

Kali ini kita akan mencoba mengimplementasikan penggunaan AutoFeat pada dataset Titanic yang disediakan package seaborn. Tuliskan script berikut untuk menggunakan dataset:

```
# importing library
import seaborn as sns

df = sns.load_dataset("titanic")
```

Sebelum menerapkan AutoFeat, kita ubah dulu nilai data non-numerikal seperti ‘sex’, ‘embarked’, ‘embark\_town’ menjadi numerikal menggunakan teknik label encoding, yaitu sebuah teknik transformasi data non-numerik menjadi numerik. Tuliskan script berikut untuk menerapkan label encoding:



```
# Importing the Encoder Library
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

for column in df.drop(columns= "survived").columns:
    df[column] = label_encoder.fit_transform(df[column])
```

Gambar berikut merupakan hasil penerapan label encoding

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	2	1	28	1	0	18	2	2	1	1	7	2	0	0
1	1	0	0	51	1	0	207	0	0	2	0	2	0	1	0
2	1	2	0	34	0	0	41	2	2	2	0	7	2	1	1
3	1	0	0	47	1	0	189	2	0	2	0	2	2	1	0
4	0	2	1	47	0	0	43	2	2	1	1	7	2	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	1	1	35	0	0	85	2	1	1	1	7	2	0	1
887	1	0	0	24	0	0	153	2	0	2	0	1	2	1	1
888	0	2	0	88	1	2	131	2	2	2	0	7	2	0	0
889	1	0	1	34	0	0	153	0	0	1	1	2	0	1	1
890	0	2	1	42	0	0	30	1	2	1	1	7	1	0	1

891 rows × 15 columns

Kita memisahkan variabel atau atribut mana yang merupakan variabel independen dan mana yang dependen. Variabel yang berperan sebagai variabel dependen pada dataset Titanic adalah ‘survived’ sehingga kita akan menyebut variabel tersebut sebagai y. Berikut implementasi pemisahan variabel. Apabila kita perhatikan gambar di atas, kita tidak menggunakan variabel ‘alive’. Hal ini dikarenakan variabel tersebut mengandung informasi yang sama dengan ‘survived’. Selain itu, kita juga tidak menggunakan data missing value.



```
X = df.dropna().drop(columns = ["survived", "alive"])
y = df.dropna()[['survived']]
```

Kita bisa memanggil fungsi pemilihan fitur yang disediakan oleh AutoFeat, yaitu **FeatureSelector()**. Kita juga mengisi nilai parameter ‘verbose’ dengan 1 supaya dapat menampilkan proses keluaran fungsi tersebut. Selanjutnya, kita menggunakan fungsi tersebut untuk mengetahui fitur mana yang paling relevan dengan ‘survived’ dengan memanggil **.fit\_transform()**. Terlihat kalau kita memasukkan variabel independen yang diwakili oleh ‘x’ dan variabel dependen yang diwakili oleh ‘y’. Hasil keluaran fungsi tersebut akan disimpan dalam sebuah variabel python, ‘feature\_selected’.



```
autofeat_model = FeatureSelector(verbose=1)
feature_selected = autofeat_model.fit_transform(X, y)
```

Berikut adalah hasil keluaran program. Terlihat beberapa proses dalam menjalankan fungsi `.fit_transform()` dimana hasil akhirnya adalah 7 buah fitur yang paling relevan dengan ‘survived’.

```
[featsel] Scaling data...done.
[featsel] Feature selection run 1/5
[featsel] Feature selection run 2/5
[featsel] Feature selection run 3/5
[featsel] Feature selection run 4/5
[featsel] Feature selection run 5/5
[featsel] 8 features after 5 feature selection runs
[featsel] 7 features after correlation filtering
[featsel] 7 features after noise filtering
```

Kini coba jalankan variabel ‘feature\_selected’ yang sudah kita definisikan sebelumnya. Hasil keluarannya adalah fitur yang dipilih oleh AutoFeat seperti terlihat pada gambar di bawah.

	<code>class</code>	<code>sibsp</code>	<code>adult_male</code>	<code>deck</code>	<code>parch</code>	<code>embarked</code>	<code>fare</code>
0	2	1		1	7	0	2 18
1	0	1		0	2	0	0 207
2	2	0		0	7	0	2 41
3	0	1		0	2	0	2 189
4	2	0		1	7	0	2 43
...	...	...		...	...	...	...
<b>886</b>	1	0		1	7	0	2 85
<b>887</b>	0	0		0	1	0	2 153
<b>888</b>	2	1		0	7	2	2 131
<b>889</b>	0	0		1	2	0	0 153
<b>890</b>	2	0		1	7	0	1 30
891 rows × 7 columns							

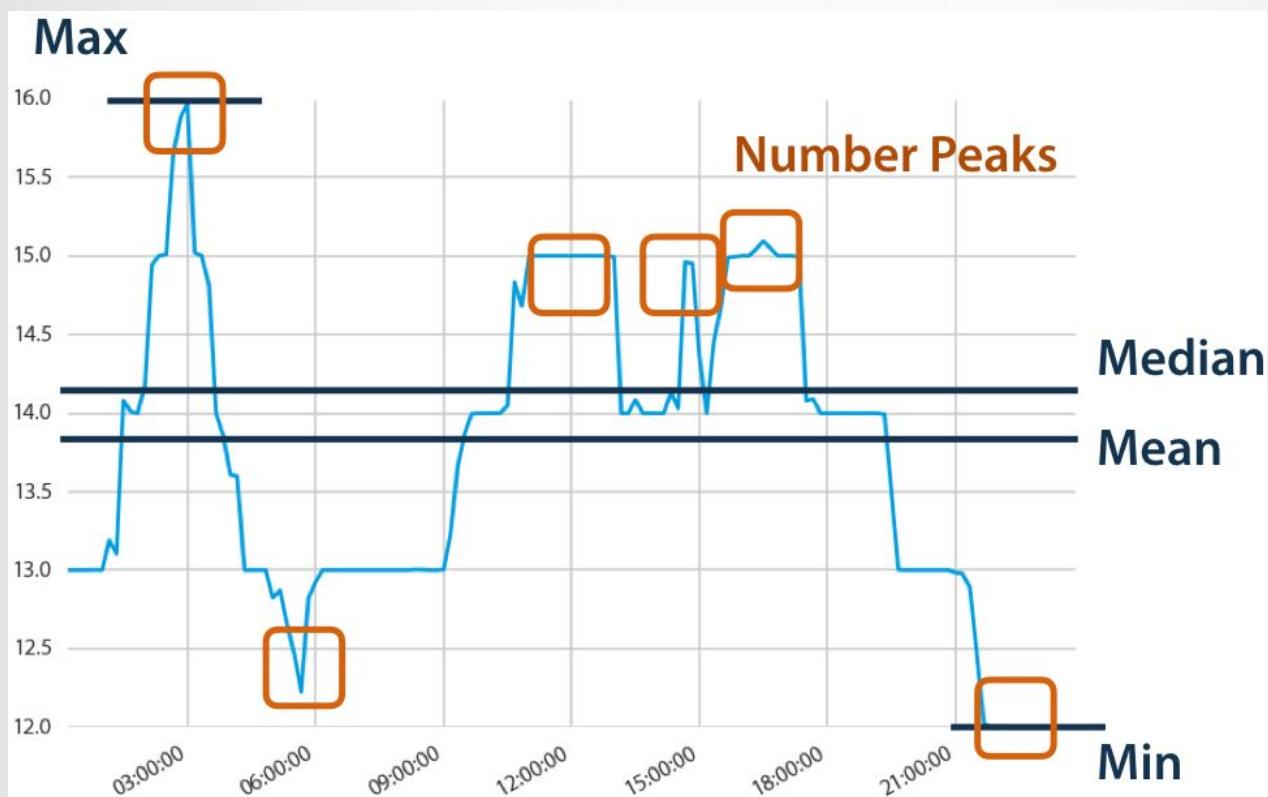
### 3. Tsfresh

Tsfresh merupakan salah satu package yang disediakan bahasa pemrograman Python.

Package ini **menghitung sejumlah besar karakteristik deret waktu atau fitur secara otomatis.**

Package tersebut juga mencakup metode untuk menilai kekuatan penjelas dan signifikansi dalam tugas regresi dan klasifikasi.

Package ini dikenal cukup cepat dalam mengimplementasikan feature engineering dan bisa mendeteksi fitur yang dianggap kurang relevan



Ref:

[Automatic Feature Extraction — milfinlab 1.5.0 documentation](#)

# Apa itu Exploratory Data Analysis?

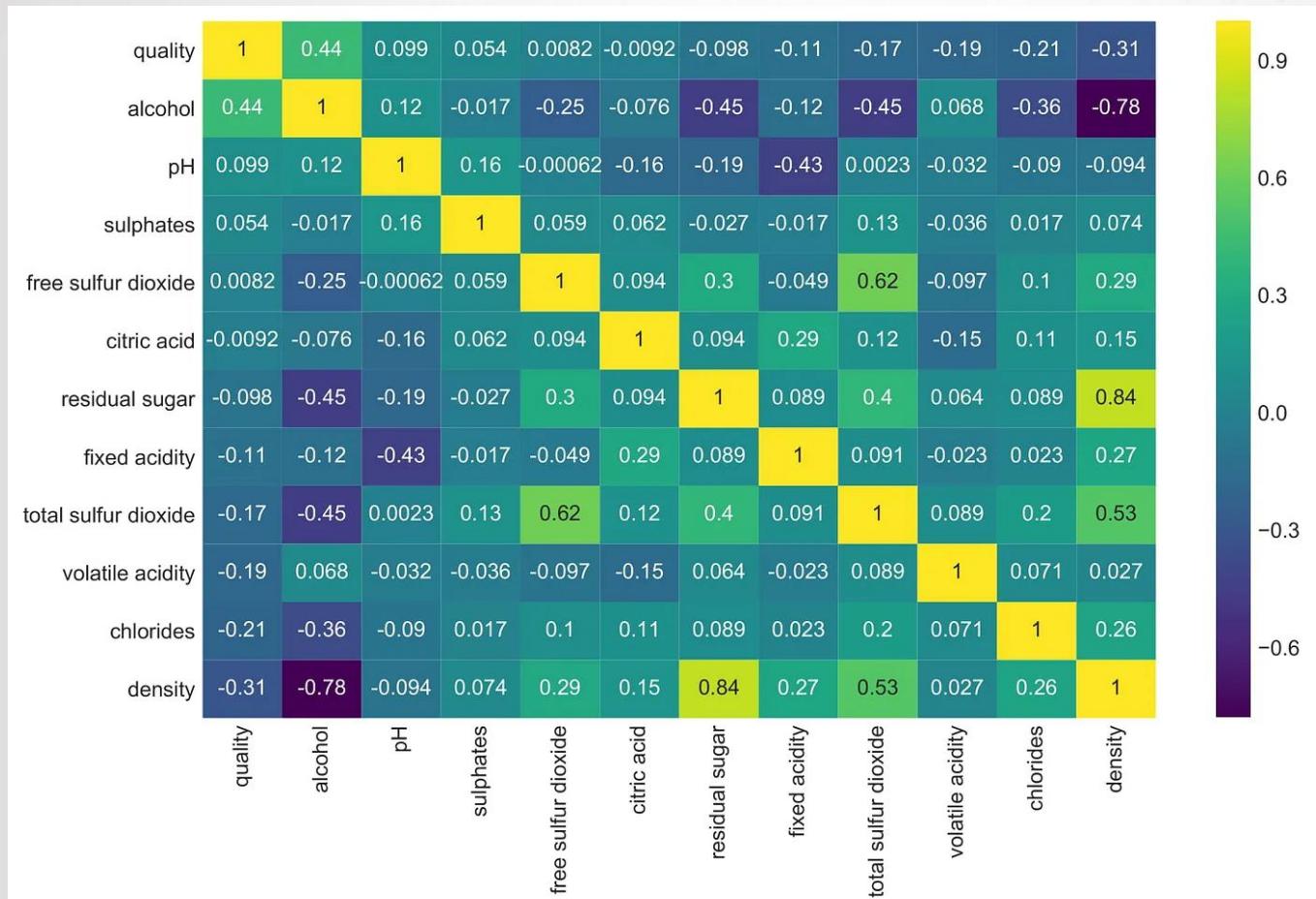
Exploratory Data Analysis (EDA) adalah sebuah **proses menganalisa dan menyelidiki sebuah dataset untuk melihat karakteristik utamanya**.

EDA **biasa digunakan untuk melihat insight yang mungkin tidak akan terlihat dalam proses modeling** dan akan memberikan pemahaman yang lebih baik dari dataset tersebut. Umumnya, EDA dilakukan melalui menampilkan grafik supaya memudahkan pemahaman informasi berharga dari suatu dataset seperti analisa atribut, analisa korelasi atau hubungan antar atribut, dan analisis trend pada data.

Jadi, sangat dianjurkan sekali kalau kita harus memiliki pemahaman sedetail mungkin di dataset yang kita miliki sebelum menggunakannya di proses selanjutnya seperti membangun model machine learning.



Coba perhatikan grafik berikut::



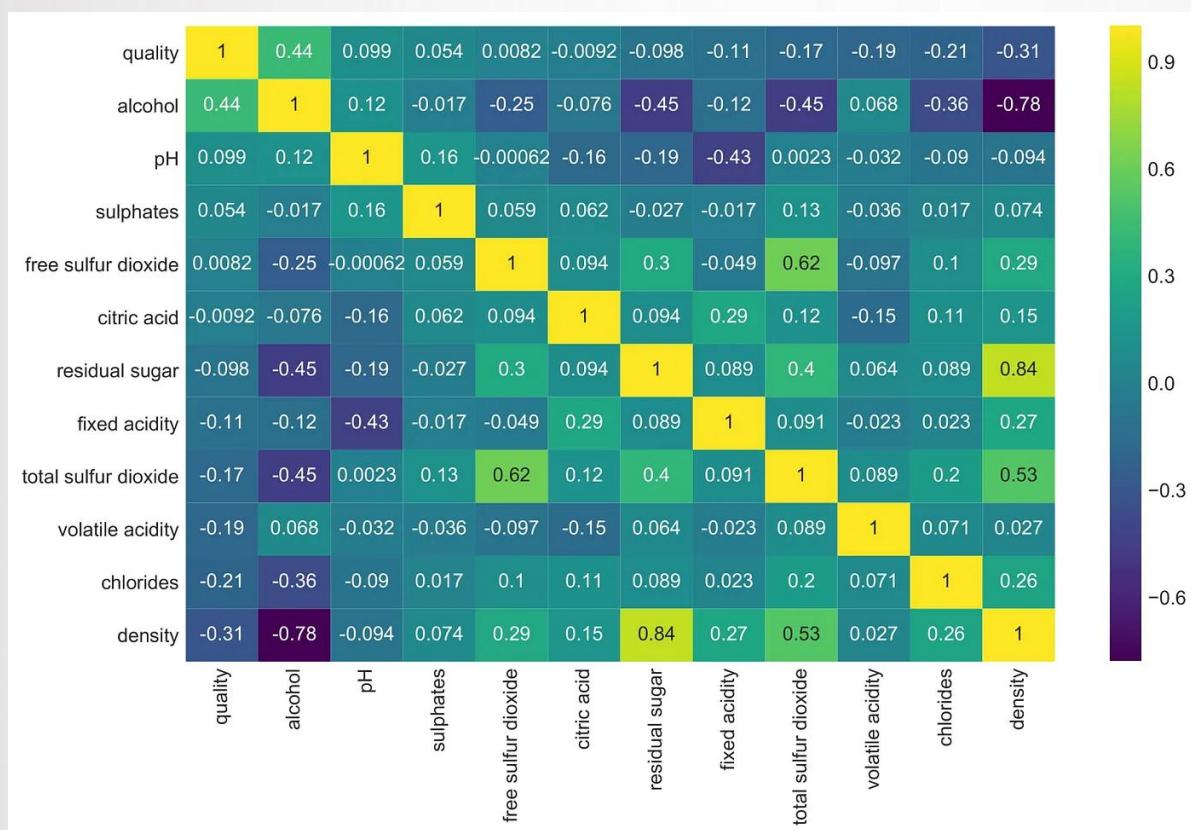
Grafik diatas merupakan matriks korelasi antar atribut di dataset kualitas wine. Terlihat kalau variabel "residual sugar" memiliki korelasi negatif terhadap "alkohol". Artinya, semakin besar kandungan alkohol, maka akan semakin kecil kandungan gulanya. Begitu sebaliknya, semakin banyak kandungan gula, maka semakin sedikit kandungan alkoholnya.

Berhubung korelasi mendekati nol tidak mengindikasikan adanya korelasi, maka bisa disimpulkan kalau "free sulphur dioxide" dan "citric acid" tidak memiliki korelasi terhadap "quality". Artinya, besaran kedua kandungan tersebut tidak memiliki dampak terhadap kualitas wine yang dihasilkan.

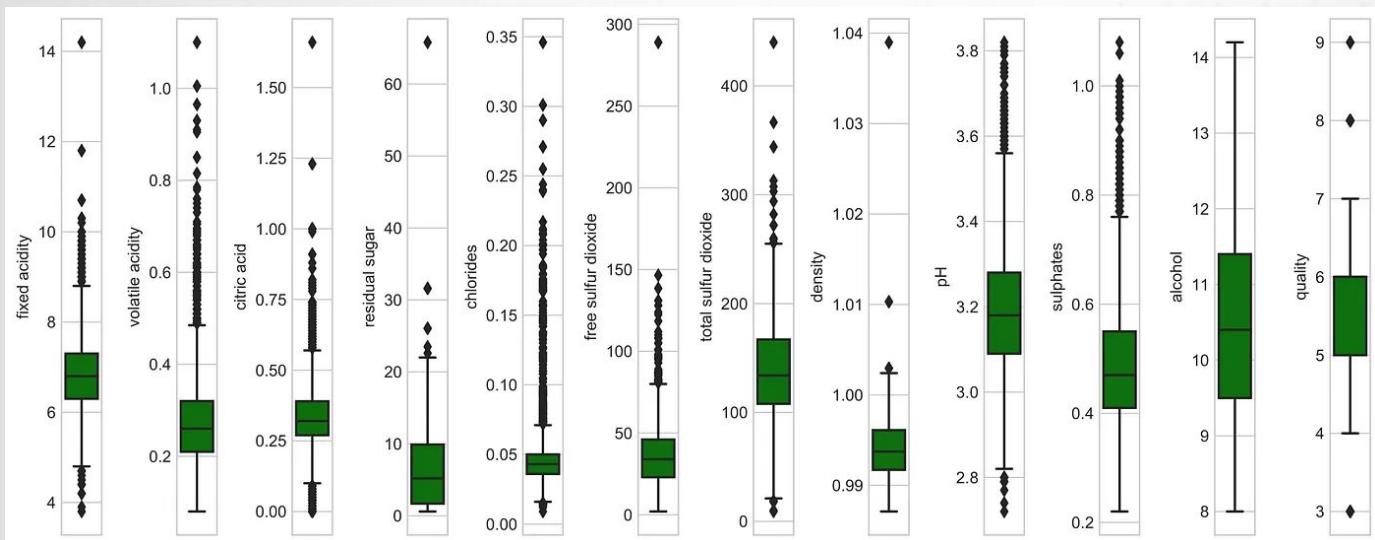
Setelah seorang data scientist memperoleh pemahaman terkait dataset yang dimiliki, dia bisa menentukan apa yang perlu dilakukan untuk memanipulasi sumber data yang dimiliki sehingga memudahkannya untuk menemukan pola, melihat anomali, menguji hipotesis, atau memeriksa asumsi yang mereka miliki.

**EDA merupakan bagian penting dalam framework data science**, khususnya dalam memahami data (data understanding) karena kita bisa melihat data sebelum mengajukan asumsi tentang data tersebut. Kita bisa melihat kesalahan yang mungkin ada dalam data tersebut dan juga menemukan hubungan-hubungan atau pola menarik antara variabel. Selain itu, kita juga bisa mendekripsi outliers atau anomali dalam data.

Kita ambil contoh di matriks korelasi seperti pembahasan kita sebelumnya. Kita bisa tahu bahwa atribut yang memiliki korelasi salah satunya adalah “density” terhadap “residual sugar” karena nilai korelasinya cukup besar dan mendekati 1. Langkah yang bisa diambil adalah menghilangkan atribut yang berkorelasi tersebut.



Selanjutnya kita coba melihat grafik berikut:



Grafik ini adalah boxplot. Grafik ini dapat memberikan kita informasi terkait rentang data, posisi kuartil, dan data outlier. Data outlier adalah data pengamatan yang terletak abnormal atau berbeda dari data yang lain yang biasanya ditunjukkan dengan adanya visual berbentuk diamond berwarna hitam. Bisa terlihat kalau hampir semua atribut yang kita miliki memiliki data outlier kecuali alcohol. Beberapa data scientist lebih suka untuk menghilangkan data outlier tersebut daripada membiarkannya ketika membangun model machine learning.

# Bagaimana Cara Melakukan Exploratory Data Analysis?

Sebenarnya sudah banyak metode dan tools yang dapat kita gunakan untuk melakukan EDA.

Gambar di bawah dapat menggambarkan bagaimana melakukan EDA sebagai aktivitas pemahaman data. Terlihat ada beberapa langkah yang dapat dilakukan, seperti pemahaman atribut atau kolom di data, analisa satu variabel (univariate analysis) atau lebih(bi-/multivariate analysis), hingga melihat adanya data yang hilang serta data pecilan(outlier).

Sebelum akhirnya melakukan feature engineering dan melihat insights yang ada dari data tersebut, yuk kita bahas satu per satu!



## 1. Load Dataset

Pertama-tama, kita harus memasukkan **dataset** yang dimiliki dan juga lupa memasukkan atau import package Python yang akan digunakan.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline

data=pd.read_csv('/content/train.csv') #cek directorynya ya
```

Terlihat dari gambar tersebut kita memasukkan beberapa packages seperti pandas. Packages tersebut biasa digunakan oleh Data Scientist untuk memasukkan data dalam bahasa pemrograman python.

Syntax **.read\_csv()** berfungsi memasukkan data yang kita miliki. Namun yang harus diingat, fungsi tersebut hanya bisa digunakan apabila file kita berformat .csv ya! Format file lain seperti excel, kita bisa menggunakan **.read\_excel()**. Sementara itu, kita bisa menggunakan **.read\_json()** kalau file kita berformat json. Lebih lengkapnya bisa dilihat [disini](#) ya!

Selain itu, kita juga bisa menggunakan dataset yang telah tersedia oleh package python. Biasanya, dataset yang disediakan sifatnya terbuka dan bisa kita gunakan untuk latihan menganalisis suatu dataset. Coba tuliskan script berikut.

Pada latihan kali ini, kita akan mencoba menggunakan dataset Titanic yang disediakan oleh package seaborn. Langkah-langkah menggunakan dataset Titanic cukup mudah, loh!

1. Memasukkan/me-import package seaborn

Tuliskan script “import seaborn as sns”. Artinya, kita akan menggunakan package seaborn dengan nama alias sebagai ‘sns’. Nama alias tersebut yang akan kita gunakan untuk proses selanjutnya

1. Memasukkan/me-import dataset

Memasukkan dataset dengan memanggil fungsi `.load_dataset("titanic")`. Berhubung kita ingin menggunakan dataset titanic, maka kita menuliskan “titanic” sebagai nilai parameter pada fungsi `.load_dataset()`. Selain itu, jangan lupa mendefinisikannya sebagai suatu variabel, seperti “df”.

```
# importing library
import seaborn as sns

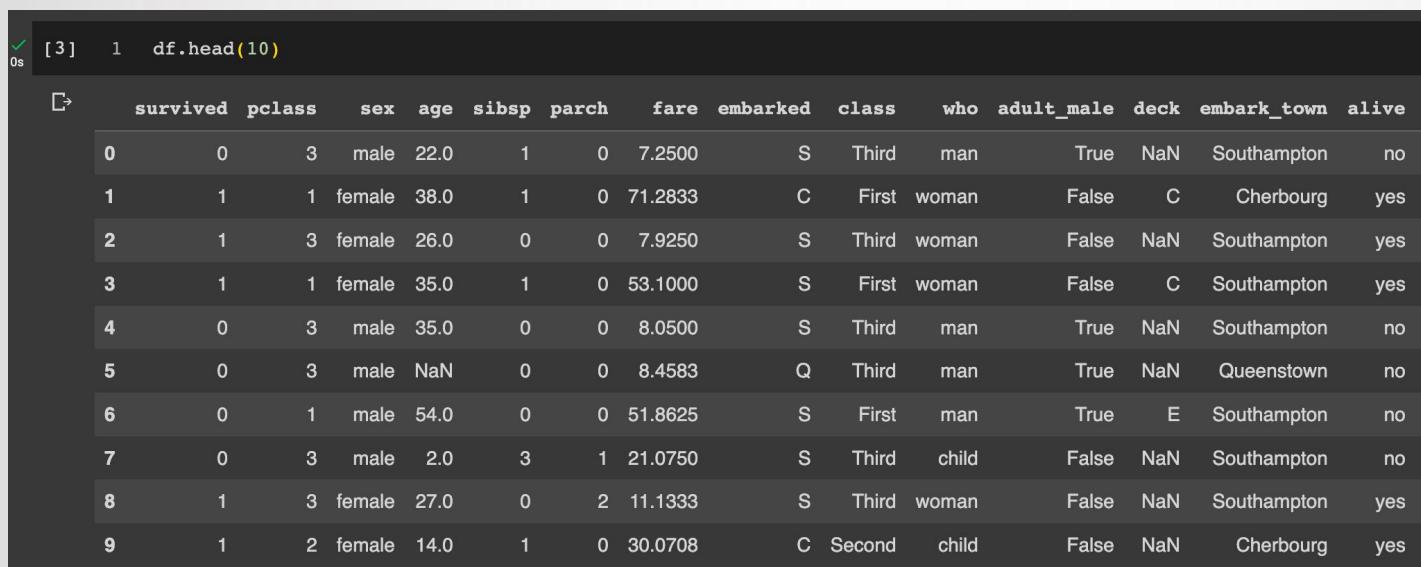
df = sns.load_dataset("titanic")
```

## 2. Distinguish Attributes

Selanjutnya kita bisa **mempelajari karakteristik attributes dasar/analisis deskriptif tiap variabel** menggunakan beberapa fungsi yang tersedia di bahasa pemrograman Python seperti berikut:

Melihat **gambaran beberapa baris awal data** yang dimiliki menggunakan syntax **.head()**.

Secara default, jumlah baris yang ditampilkan adalah sebanyak 5 baris pertama. Kita juga memasukkan angka tertentu apabila ingin menampilkan lebih dari 5 baris. Misalkan kita ingin menampilkan 10 baris pertama, kita bisa menuliskan **.head(10)**.



A screenshot of a Jupyter Notebook cell. The cell number is [3] and the code is 1 df.head(10). The output shows the first 10 rows of a DataFrame named df. The columns are survived, pclass, sex, age, sibsp, parch, fare, embarked, class, who, adult\_male, deck, embark\_town, and alive. The data includes various passenger details such as gender, age, class, and survival status.

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no
5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no
6	0	1	male	54.0	0	0	51.8625	S	First	man	True	E	Southampton	no
7	0	3	male	2.0	3	1	21.0750	S	Third	child	False	NaN	Southampton	no
8	1	3	female	27.0	0	2	11.1333	S	Third	woman	False	NaN	Southampton	yes
9	1	2	female	14.0	1	0	30.0708	C	Second	child	False	NaN	Cherbourg	yes

Selain itu, kita juga bisa menampilkan beberapa baris terakhir data dengan menggunakan syntax **.tail()**.

Sama seperti dengan fungsi `.head()`, secara default, jumlah baris yang ditampilkan adalah sebanyak 5 baris terakhir. Kita juga memasukkan angka tertentu apabila ingin menampilkan lebih dari 5 baris. Misalkan kita ingin menampilkan 10 baris terakhir, maka kita bisa menuliskan **.tail(10)**.



The screenshot shows a Jupyter Notebook cell with the code `[4] 1 df.tail()` and its output. The output displays the last 10 rows of the DataFrame `df`. The columns are: survived, pclass, sex, age, sibsp, parch, fare, embarked, class, who, adult\_male, deck, embark\_town, alive. The data includes various passenger details such as gender, age, class, and survival status.

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive
886	0	2	male	27.0	0	0	13.00	S	Second	man	True	NaN	Southampton	no
887	1	1	female	19.0	0	0	30.00	S	First	woman	False	B	Southampton	yes
888	0	3	female	Nan	1	2	23.45	S	Third	woman	False	NaN	Southampton	no
889	1	1	male	26.0	0	0	30.00	C	First	man	True	C	Cherbourg	yes
890	0	3	male	32.0	0	0	7.75	Q	Third	man	True	NaN	Queenstown	no

**Melihat informasi** seperti daftar kolom, nama kolom, jumlah baris, null data, dan tipe data menggunakan **.info()**.

```

[5] 1 df.info()

[1s] <class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   survived    891 non-null    int64  
 1   pclass      891 non-null    int64  
 2   sex         891 non-null    object  
 3   age         714 non-null    float64 
 4   sibsp       891 non-null    int64  
 5   parch       891 non-null    int64  
 6   fare        891 non-null    float64 
 7   embarked    889 non-null    object  
 8   class       891 non-null    category
 9   who         891 non-null    object  
 10  adult_male  891 non-null    bool   
 11  deck        203 non-null    category
 12  embark_town 889 non-null    object  
 13  alive       891 non-null    object  
 14  alone       891 non-null    bool  
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB

```

Berdasarkan gambar tersebut, kita bisa memperoleh beberapa informasi:

- Terdapat 14 buah atribut
- Daftar nama atribut dalam kolom “Column”
- Tipe data tiap atribut pada kolom “DType”, diantaranya:
  - survived, pclass, sibsp, dan parch bertipe data int64 atau bilangan bulat
  - age dan fare bertipe data float64 atau bilangan desimal
  - sex, embarked, who, embark\_town, dan alive bertipe data object atau biasanya berupa data yang memiliki karakter huruf (misalkan male untuk Sex)
  - alone bertipe bool atau boolean yang nilainya hanya terdiri dari true atau false
  - class dan deck bertipe category yang mengindikasikan data ordinal. Artinya, atribut tersebut memiliki urutan dari terendah hingga tertinggi
- Terdapat informasi “RangeIndex: 891 entries, 0 to 890”. Artinya, jumlah data kita adalah 891 baris. Di dalam kolom “Non-Null Count” kita bisa melihat atribut mana yang terindikasi tidak memiliki nilai (missing value) seperti atribut age dan deck karena terlihat jumlah baris non-null tidak mencapai 891

Melihat jumlah dari sekelompok data menggunakan `.count()` seperti syntax berikut. Misalkan kita ingin membandingkan jumlah perbandingan korban antara laki-laki dan perempuan.

Pertama, kita mengelompokkan berdasarkan atribut ‘sex’ untuk mendapatkan informasi jenis kelamin dan ‘alive’ untuk mendapatkan informasi status penumpang. Kita menggunakan `.groupby()` dengan parameter array atribut yang kita sebutkan sebelumnya. Apabila di kemudian hari kita perlu mengelompokkan data yang lain, kita cukup mengganti nilai array di dalam fungsi tersebut.

Selanjutnya, kita memanggil atribut alive dengan menuliskan `['alive']` yang diikuti pemanggilan fungsi `.count()`.

```
✓ [8] 1 df.groupby(['sex', 'alive'])['alive'].count()

[sex      alive
 female    no        81
              yes       233
 male     no        468
              yes       109
 Name: alive, dtype: int64]
```

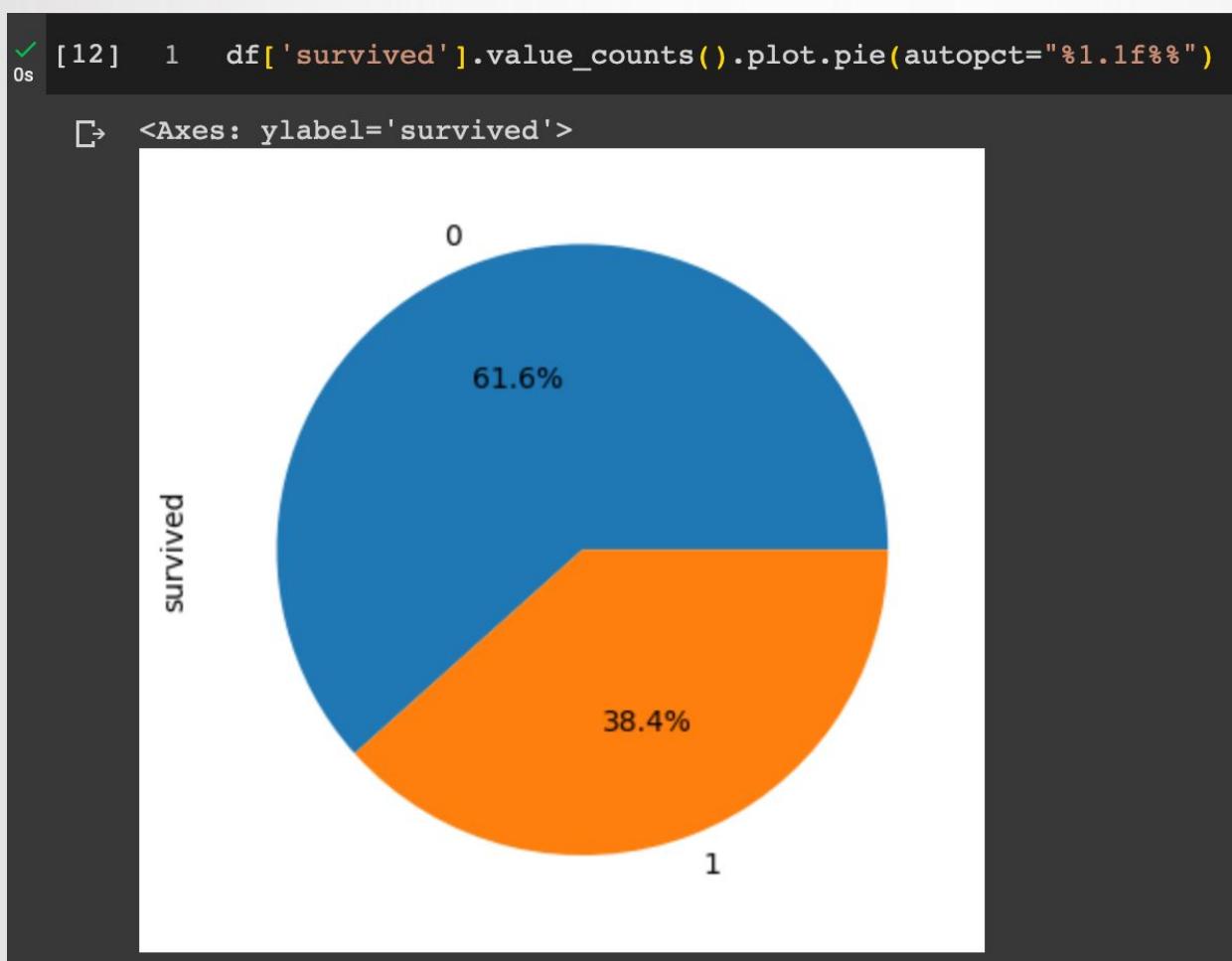
Berdasarkan gambar tersebut, kita jadi memperoleh informasi bahwa perbandingan jumlah korban berdasarkan jenis kelamin seperti jumlah penumpang yang tidak selamat didominasi oleh laki-laki sebesar 468 sementara jumlah penumpang perempuan yang tidak selamat “hanya” 81 orang.

Informasi yang kita coba sebelumnya memang berharga, namun alangkah baiknya kalau kita menampilkan dalam bentuk grafik supaya lebih intuitif. Oleh karena itu, yuk kita mencoba menampilkannya dalam grafik!

Kita bisa menampilkannya dalam sebuah grafik pie menggunakan fungsi `.value_counts()` seperti gambar berikut.

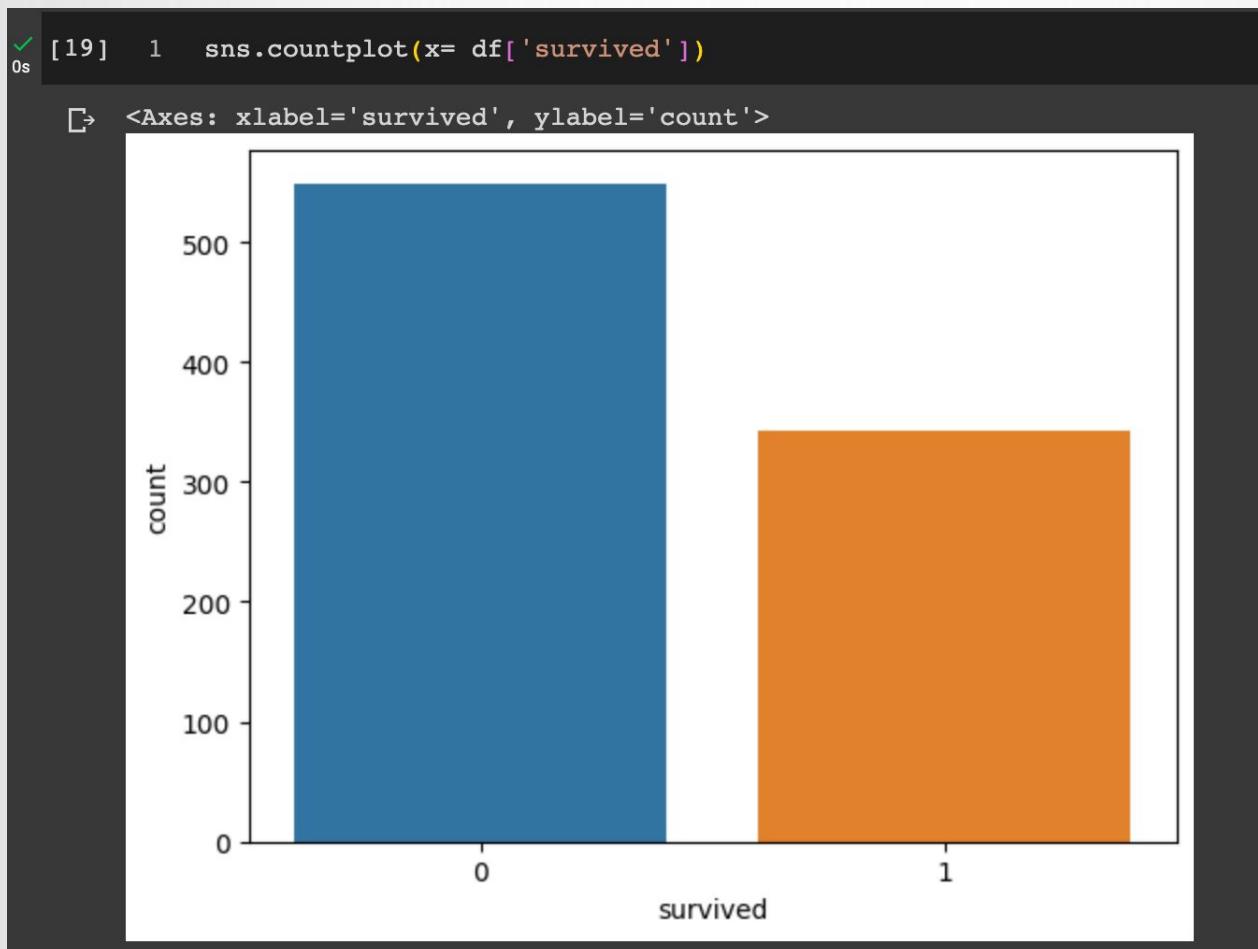
Pertama kita panggil atribut yang ingin divisualisasikan yang kemudian memanggil fungsi `.value_counts()`. Selanjutnya, kita menulis `.plot.pie(autopct='%.1lf%%')` untuk membuat grafik pie yang disediakan oleh package matplotlib.

Nilai autopct berfungsi menampilkan berapa bilangan di belakang koma. Apabila kita ingin menampilkan dua angka di belakang koma (misalkan menjadi 61.66), kita dapat menggantinya dengan `%1.2f%%`.



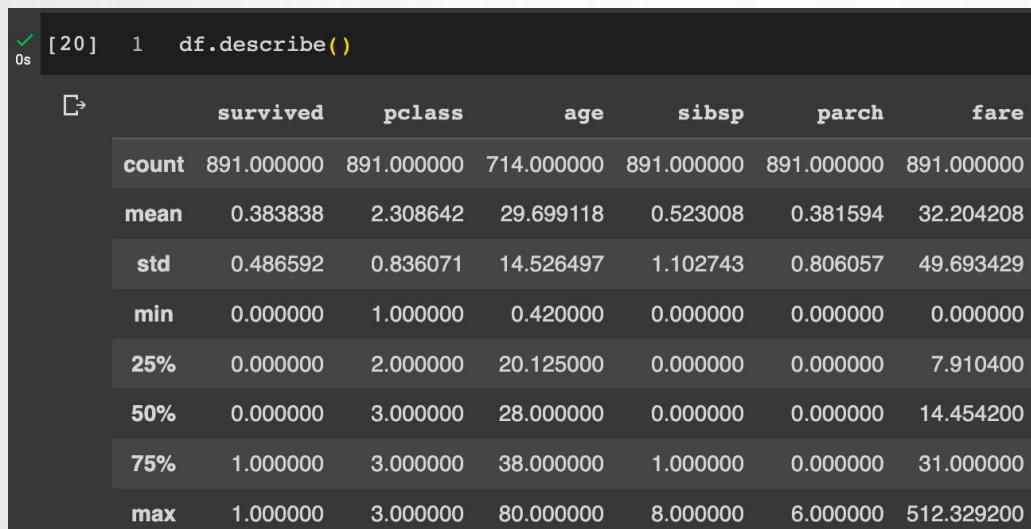
Kita juga bisa menampilkannya dalam sebuah grafik batang menggunakan fungsi **.countplot()** yang disediakan package seaborn seperti gambar berikut. Misalkan kita ingin menampilkan perbandingan jumlah penumpang hidup dengan yang tidak.

Implementasinya cukup mudah karena kita cukup memanggil fungsi **.countplot()** yang kemudian mengisi nilai parameter ‘x’. Kita mengisi nilai tersebut dengan `df['survived']` karena ingin menampilkan perbandingan jumlah yang selamat dan tidak yang direpresentasikan atribut ‘survived’.



Melihat **informasi statistik deskriptif** dari **kolom numerik** menggunakan **.describe()**.

Berdasarkan gambar berikut kita memperoleh informasi seperti jumlah data dan informasi statistik deskriptif seperti nilai rata-rata, standar deviasi, nilai terkecil-terbesar, dan nilai kuartil.



	survived	pclass	age	sibsp	parch	fare
<b>count</b>	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
<b>mean</b>	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
<b>std</b>	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
<b>min</b>	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
<b>25%</b>	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
<b>50%</b>	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
<b>75%</b>	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
<b>max</b>	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Kita juga bisa melihat **informasi statistik deskriptif** pada **kolom kategorikal** menggunakan **.describe(include=object)** seperti gambar berikut. Kita bisa memperoleh informasi seperti jumlah data, jumlah data unik, dan nilai yang memiliki frekuensi terbanyak.

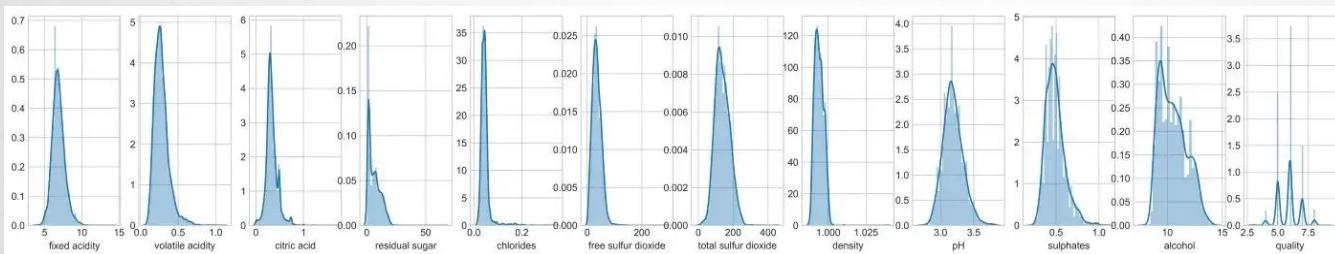
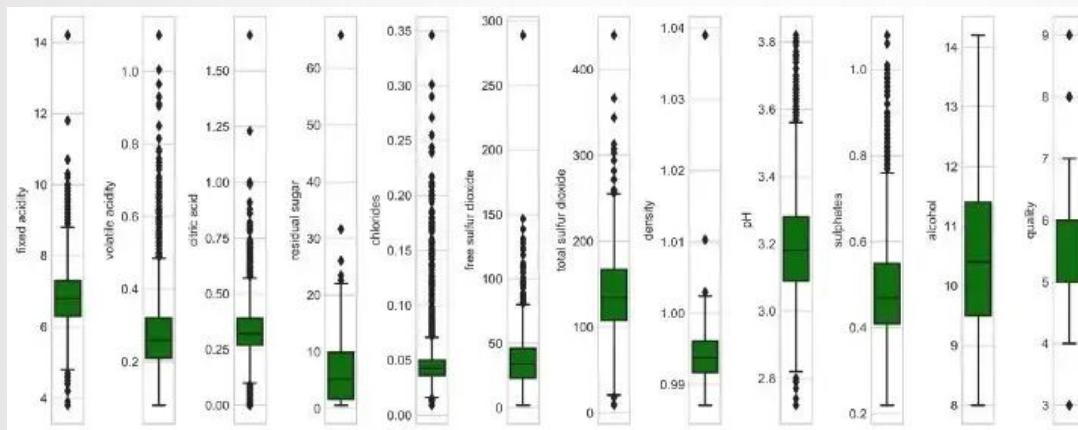
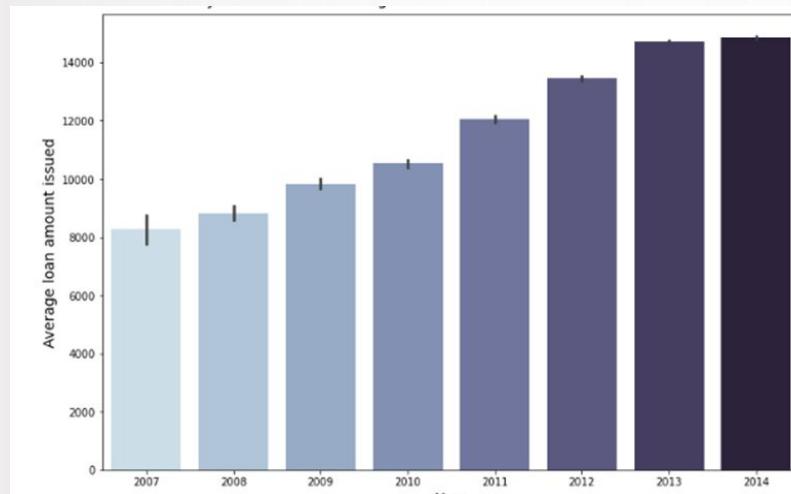


	sex	embarked	who	embark_town	alive
<b>count</b>	891	889	891	889	891
<b>unique</b>	2	3	3	3	2
<b>top</b>	male	S	man	Southampton	no
<b>freq</b>	577	644	537	644	549

### 3. Univariate Analysis

Analisis ini kita hanya **melihat variabel satu persatu** tanpa menghubungkan dengan variabel lainnya. Kita bisa melihat deskripsi dan juga pola-pola tiap variabel.

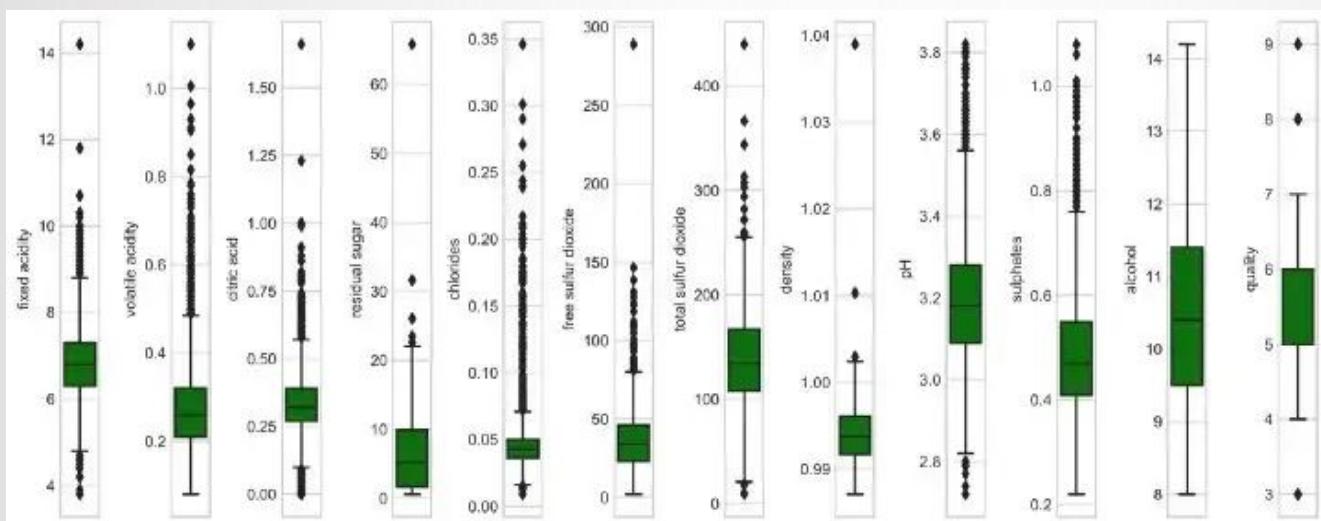
**Alangkah baiknya analisis ini dilakukan melalui data visualization**, yaitu membuat chart yang merepresentasikan suatu data. Beberapa chart yang bisa digunakan dalam proses univariate analysis menggunakan library yang ada dalam python antara lain, Bar chart, Boxplot, Distribution plot, dan lain-lain.



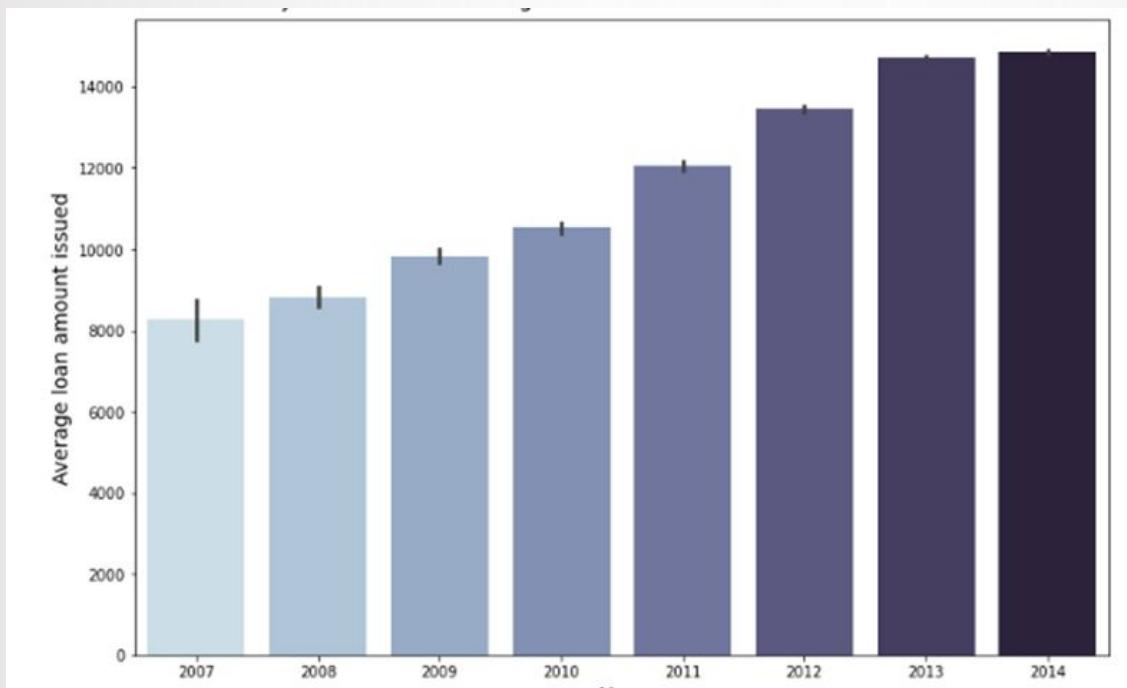
Seperti yang sudah kita bahas sebelumnya, grafik ini dinamakan boxplot. Grafik berfungsi memberikan informasi terkait rentang data, posisi kuartil, dan data outlier.

Kotak berwarna hijau merepresentasikan posisi kuartil data dimana sisi terbawah menggambarkan kuartil 1 dan sisi teratas menggambarkan kuartil 3. Posisi median digambarkan garis hitam di dalam kotak.

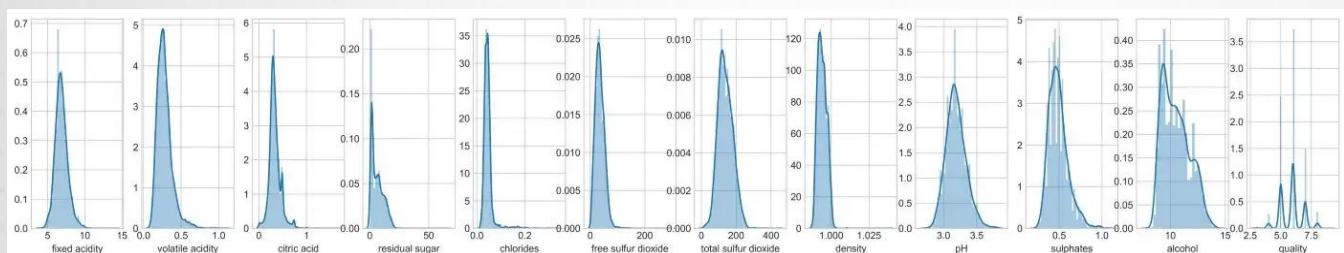
Data outlier biasanya digambarkan berupa diamond berwarna hitam dan terlihat berada di luar garis yang menghimpit kotak berwarna hijau. Semakin banyak gambar diamond, maka semakin banyak data yang diindikasi sebagai outlier. Bisa terlihat kalau hampir semua atribut yang kita miliki memiliki data outlier kecuali alcohol. Beberapa data scientist menghilangkan data outlier sebelum membangun model prediktif.



Grafik ini dinamakan bar chart atau diagram batang. Grafik ini biasa digunakan untuk menampilkan frekuensi atau jumlah data dalam satu atribut. Sebagai contoh, gambar berikut adalah perbandingan rata-rata jumlah pinjaman tiap tahunnya sehingga kita bisa memperoleh informasi bagaimana tren pertumbuhan rata-rata tiap tahunnya. Selain itu, kita bisa mengetahui bahwa tidak ada peningkatan yang signifikan antara 2014 dan 2013.



Sementara itu, grafik ini dinamakan *distribution plot*. Sesuai namanya, kita bisa melihat bagaimana distribusi data suatu atribut, apakah terdapat kecenderungan miring ke kiri atau kanan. Kemiringan ini biasa disebut *skewness*. Data yang ideal adalah kemiringan yang normal (tidak terlalu ke kiri dan tidak terlalu ke kanan). Berdasarkan grafik ini terlihat kalau atribut ‘pH’ berdistribusi normal karena kemiringannya pas di tengah.

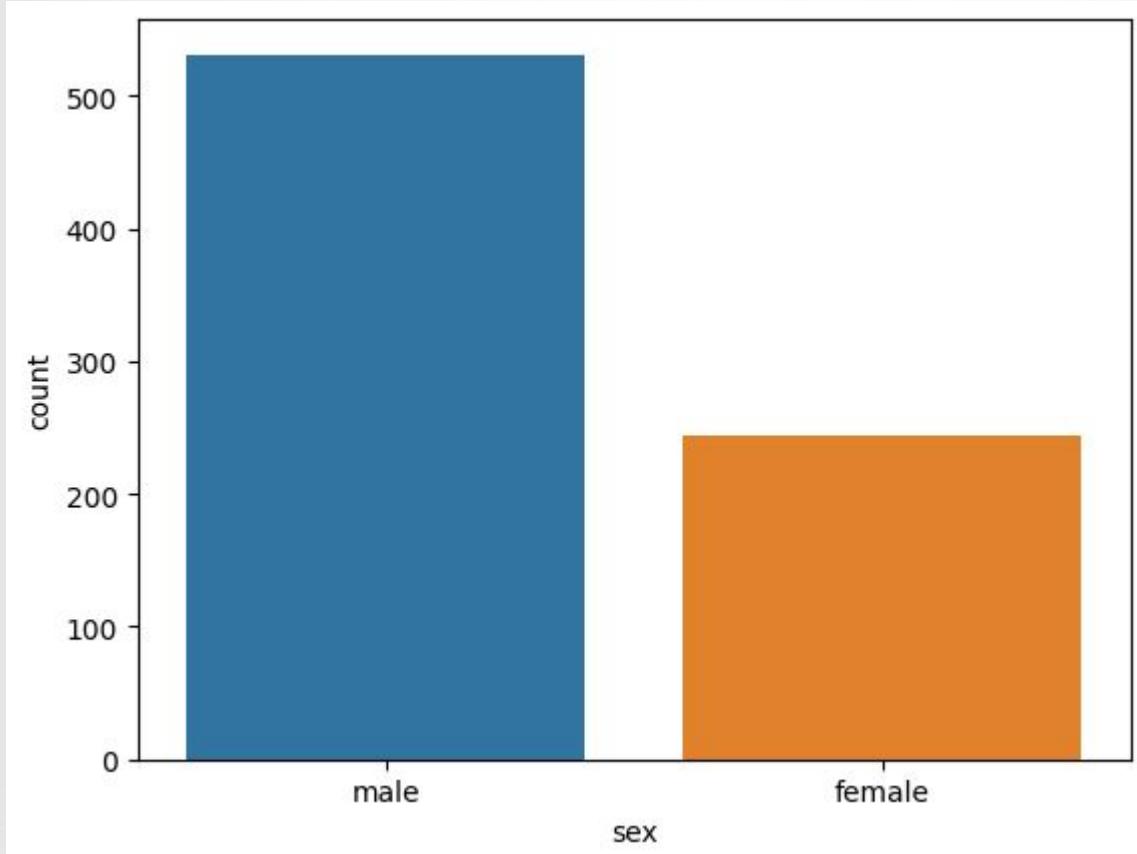


Kali ini, kita akan mencoba menampilkan bar chart yang membandingkan jumlah penumpang laki-laki dan perempuan. Silahkan tulis script berikut untuk menampilkan bar chart pada python. Kita dapat menggunakan fungsi `.countplot()` yang disediakan oleh package seaborn. Fungsi tersebut membutuhkan dua nilai parameter seperti ‘x’ yang kita isi atribut apa yang ingin ditampilkan dan ‘data’ yang diisi dengan tabel atau data.



```
sns.countplot(x='sex', data=df)
```

Berdasarkan gambar di bawah, terlihat bahwa penumpang didominasi oleh laki-laki



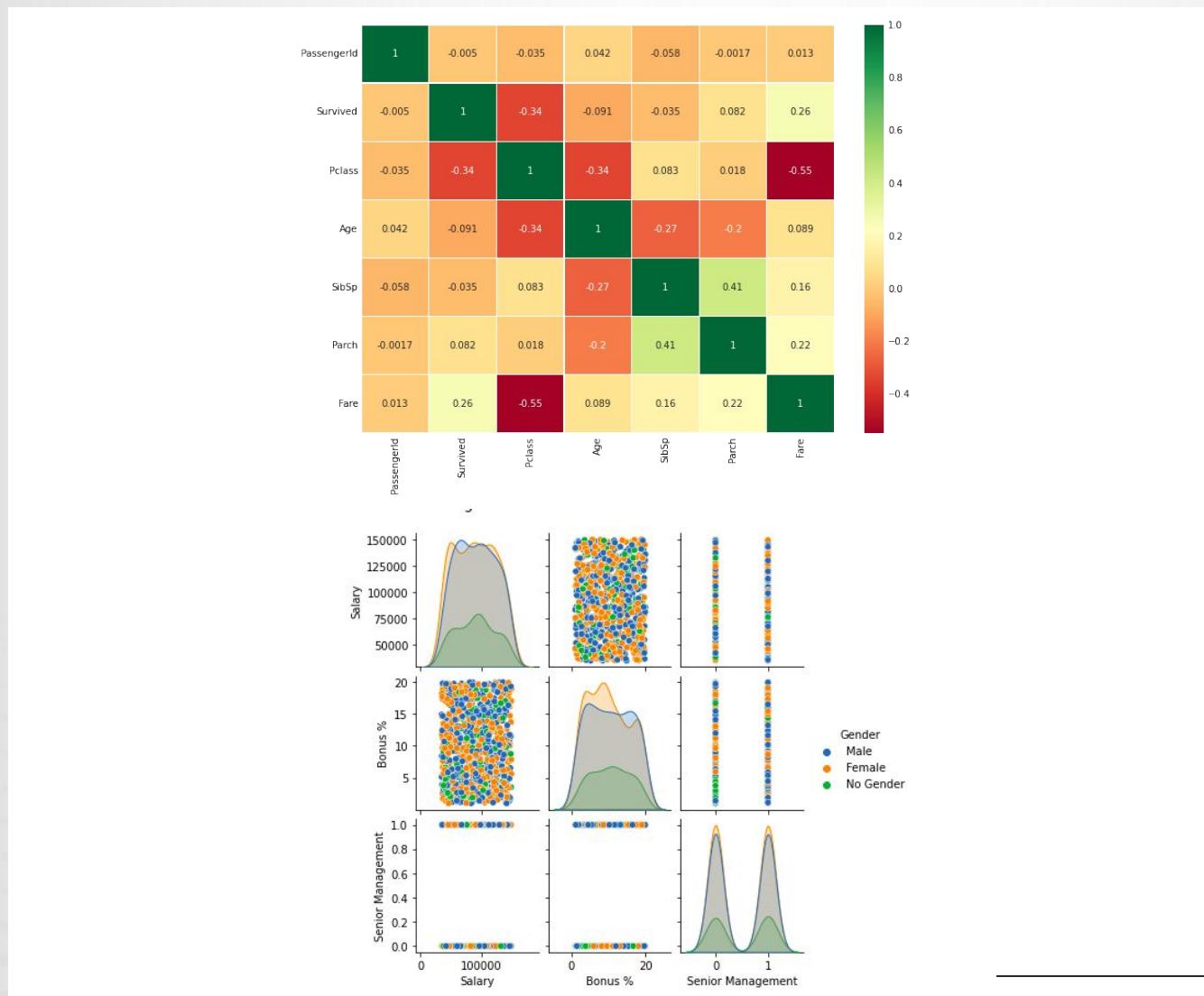
## 4. Bi-/Multivariate Analysis

Analisis ini kita akan **melihat dua atau lebih variabel** dan mencoba melihat pola yang ada antara hubungan data-data tersebut.

Kita bisa menggunakan beberapa visualisasi data untuk melihat hubungan yang ada antara data menggunakan chart seperti:

- correlation plot
- boxplot
- scatterplot
- pairplot
- etc

Di tahap ini, kita mencoba melihat hubungan antar variabel serta memahami bagaimana sebuah variabel dapat mempengaruhi variabel lain.



Grafik ini merupakan matriks korelasi. Matriks ini biasa digunakan untuk menggambarkan bagaimana hubungan antar atribut. Rentang nilai matriks ini adalah -1 hingga 1. Berikut adalah cara menginterpretasikannya:

- **Nilai korelasi semakin mendekati nol**

**Kedua atribut** dengan nilai korelasi tersebut diyakini **tidak saling berhubungan**

- **Nilai korelasi** lebih dari nol dan **mendekati 1**

**Kedua atribut** dengan nilai korelasi tersebut diyakini **berkorelasi positif**. Artinya, setiap penambahan nilai suatu atribut, maka akan diikuti atribut lainnya

- **Nilai korelasi** lebih dari nol dan **mendekati -1**

Kedua atribut dengan nilai korelasi tersebut diyakini **berkorelasi negatif**. Artinya, setiap penambahan nilai suatu atribut, maka atribut lain akan berperilaku sebaliknya seperti penurunan nilai



Sebenarnya python sudah menyediakan fungsi **.corr()** untuk menampilkan matriks korelasi dan hasilnya dapat dilihat di gambar di bawah. Namun, kita coba eksplorasi tampilannya supaya lebih intuitif yuk!

	<b>survived</b>	<b>pclass</b>	<b>age</b>	<b>sibsp</b>	<b>parch</b>	<b>fare</b>	<b>adult_male</b>	<b>alone</b>
<b>survived</b>	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307	-0.557080	-0.203367
<b>pclass</b>	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500	0.094035	0.135207
<b>age</b>	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067	0.280328	0.198270
<b>sibsp</b>	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651	-0.253586	-0.584471
<b>parch</b>	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225	-0.349943	-0.583398
<b>fare</b>	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000	-0.182024	-0.271832
<b>adult_male</b>	-0.557080	0.094035	0.280328	-0.253586	-0.349943	-0.182024	1.000000	0.404744
<b>alone</b>	-0.203367	0.135207	0.198270	-0.584471	-0.583398	-0.271832	0.404744	1.000000

Pertama, kita memasukkan package yang dibutuhkan seperti matplotlib dan numpy dengan menuliskan script berikut:



```
# Importing library
import matplotlib.pyplot as plt
import numpy as np
```

Tuliskan script berikut untuk menampilkan matriks korelasi dalam sebuah grafik.

- Pertama, kita bisa mengatur tampilan grafiknya seperti ukuran gambar dan judul. Perhatikan script berikut yang berfungsi mengatur berapa ukuran gambar dalam ukuran pixel. Kali ini kita akan mencoba membuat grafik dalam ukuran 14x8.

```
plt.figure(figsize=(14,8))
```

- Selanjutnya, kita mengatur judul grafik dengan menuliskan script di bawah:

```
title_obj = plt.title('Feature Correlation', size = 18)  
  
plt.setp(title_obj, color = 'black')
```

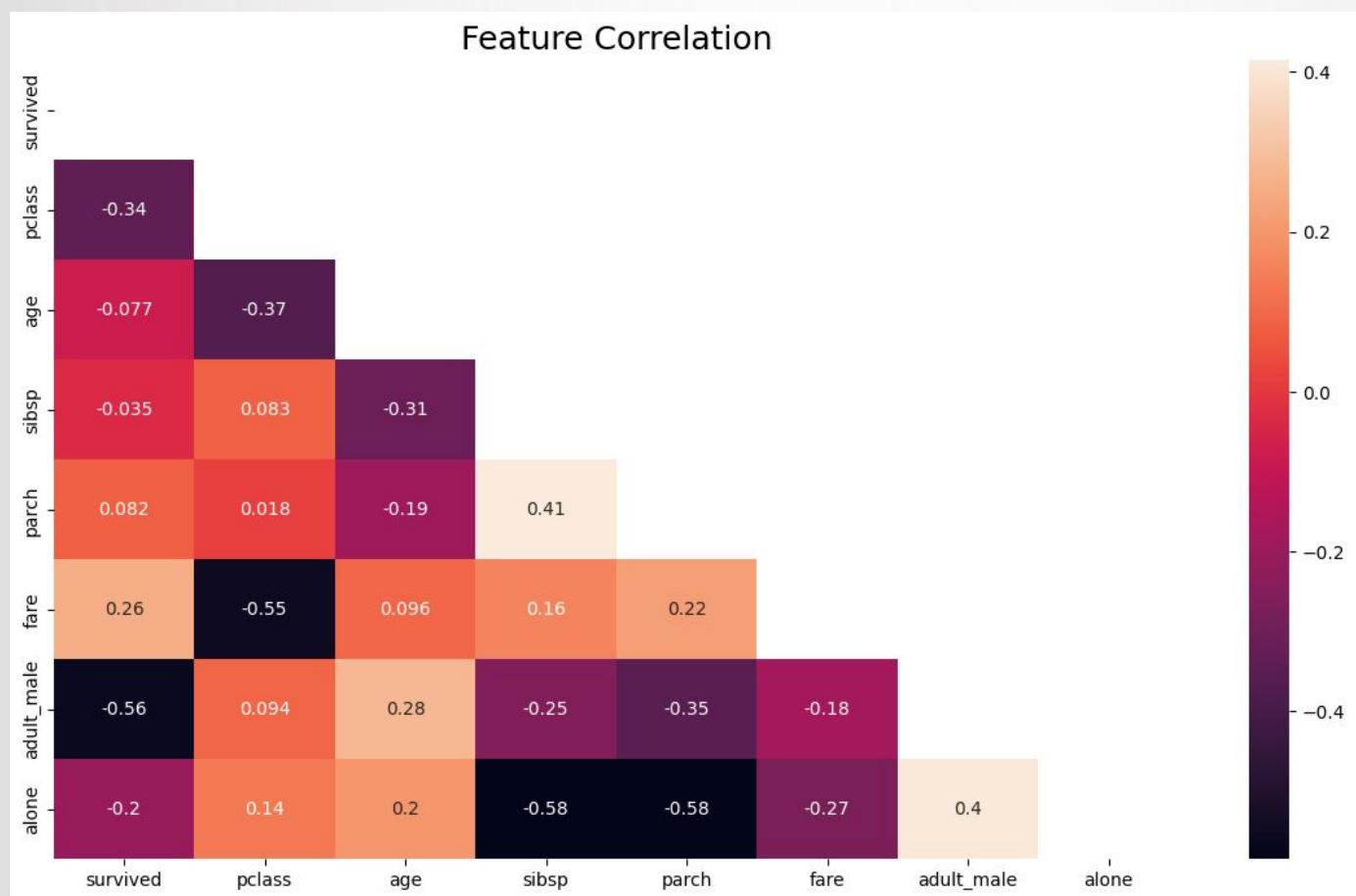
- Terakhir, kita menggunakan fungsi `.heatmap()` yang disediakan package seaborn. Di dalamnya, kita memasukkan:
  - ❖ `df.corr()` yang merupakan matriks korelasi seperti pembahasan kita sebelumnya
  - ❖ `mask` yang berfungsi menampilkan matriks menjadi setengah bagian saja
  - ❖ `annot = True`, yang berfungsi menampilkan teks di tiap kotak



```
# Define the graph size  
plt.figure(figsize = (14,8))  
# Define the graph title  
title_obj = plt.title('Feature Correlation', size = 18)  
plt.setp(title_obj, color='black')  
  
mask = np.triu(np.ones_like(df.corr()), dtype=np.bool)  
  
sns.heatmap(df.corr(), mask = mask, annot = True)  
  
plt.show()
```

Gambar berikut adalah grafik korelasi matriks dataset Titanic. Lebih intuitif daripada sebelumnya, bukan?

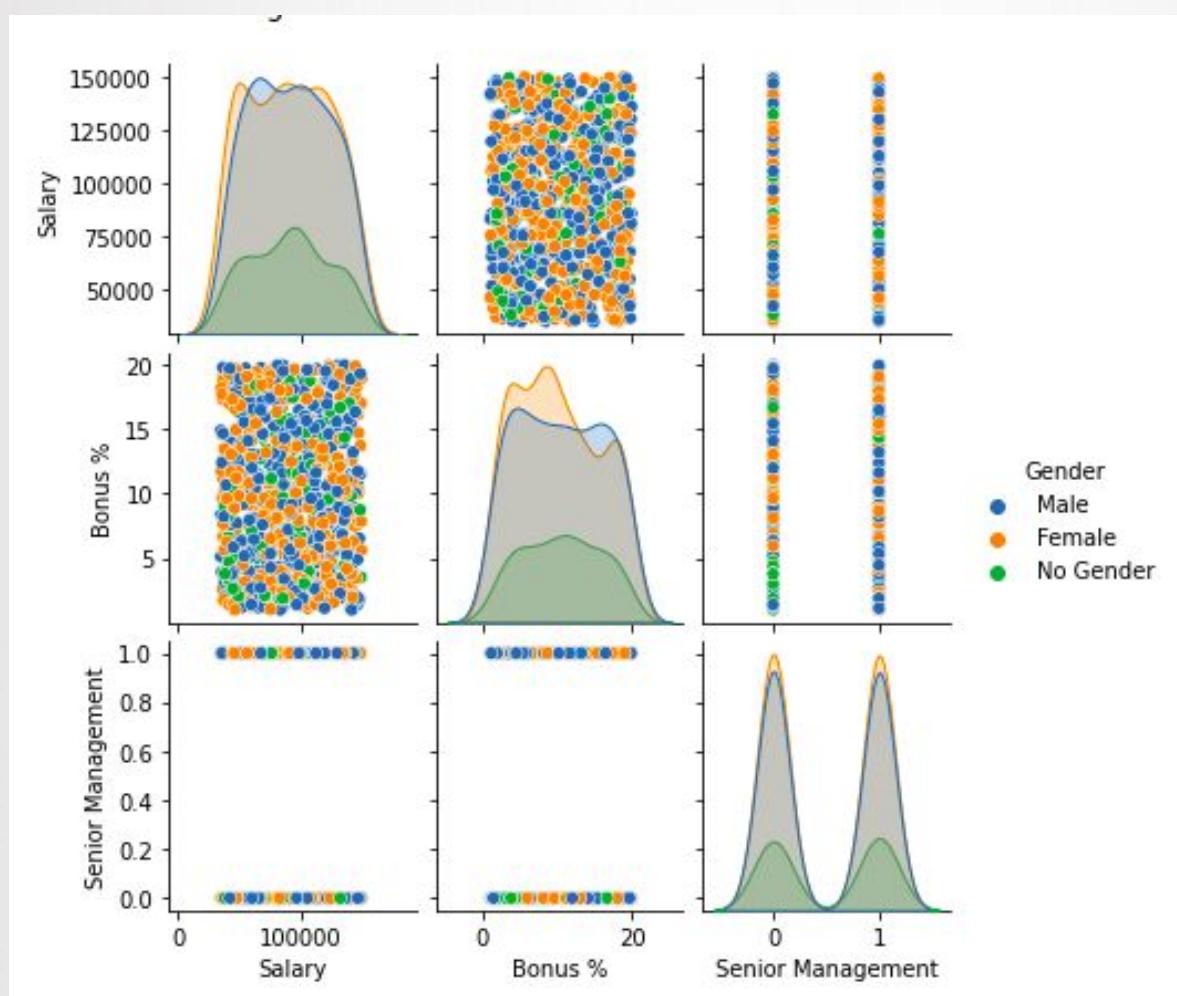
Berdasarkan matriks korelasi, diketahui atribut yang memiliki korelasi terkuat dengan ‘survived’ adalah ‘pclass’. Hal ini dilihat berdasarkan nilai korelasi atribut ‘pclass’ terhadap ‘survived’. Atribut ini berkorelasi kuat negatif dimana nilainya adalah -0,34. Selain itu, beberapa atribut seperti ‘parch’, ‘sibsp’, dan ‘age’ bisa dibilang tidak berkorelasi karena nilai korelasinya sangat mendekati nol.



Grafik ini biasa dikenal dengan pair plot. Grafik ini dapat mempersingkat waktu dalam melakukan analisis multivariate karena kita bisa menampilkan beberapa grafik sekaligus seperti scatter plot dan distribution plot. Selain itu, semua atribut dapat dianalisis dalam satu grafik, tidak harus setiap pasang atribut divisualisasikan.

Berdasarkan gambar berikut bisa terlihat bahwa bagaimana hubungan antara salary-bonus, salary-senior management, dan bonus-senior management tiap gender bentuk scatter plot.

Sebagai contoh, kita ingin melihat pengaruh bonus terhadap salary. Berdasarkan grafik, ada indikasi mayoritas yang menerima bonus semakin besar adalah pria yang digambarkan banyaknya bulat berwarna biru pada gambar. Selain itu, tidak ada indikasi kesenjangan dari sisi bonus dan salary yang diberikan karena grafik yang diperlihatkan pada scatter plot cukup simetris.

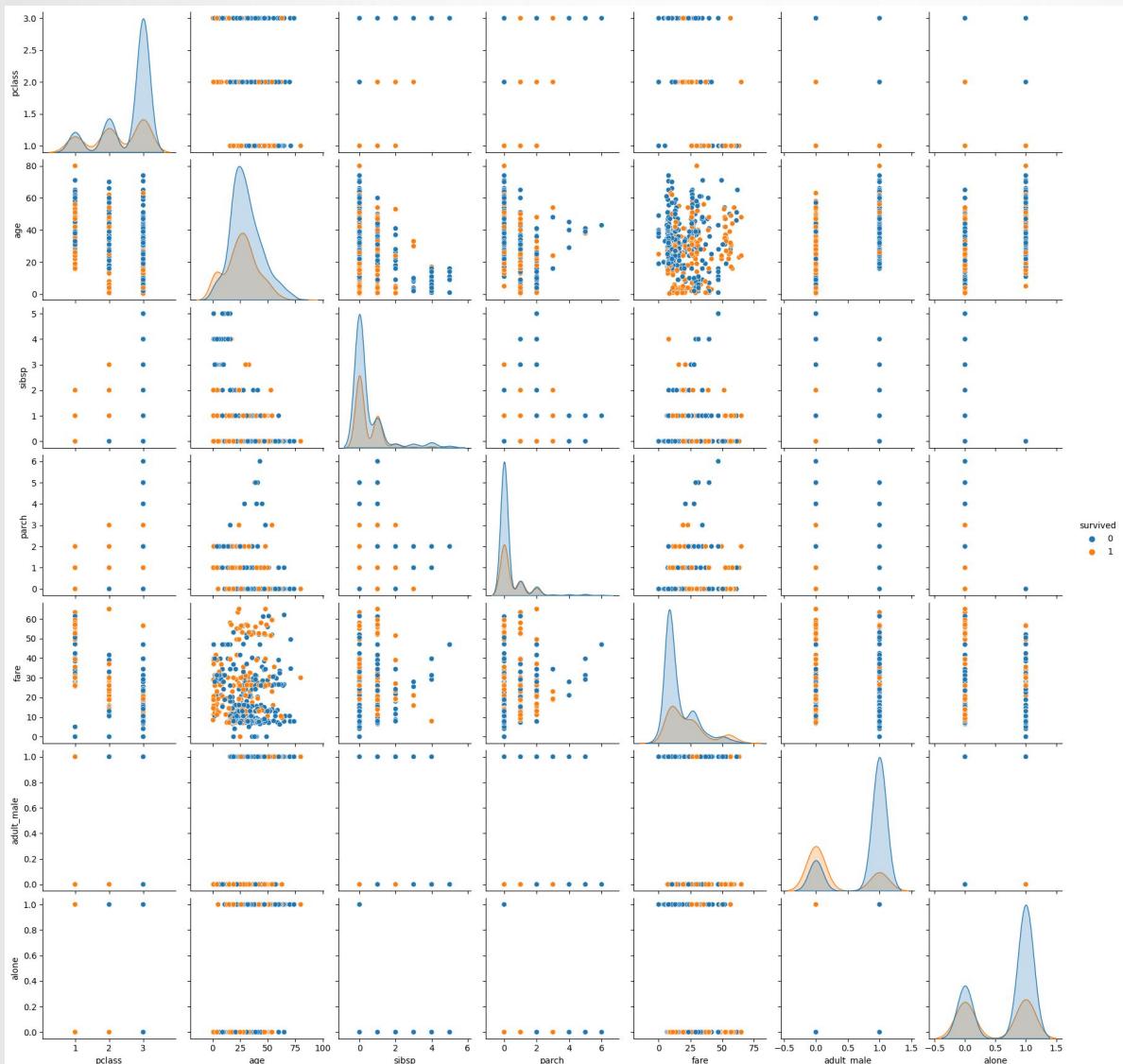


Cara menampilkan pair plot di python cukup mudah dimana kita cukup memanggil fungsi **.pairplot()** yang disediakan oleh package seaborn seperti berikut:



```
sns.pairplot(df, hue ='survived')
```

Terlihat bahwa fungsi tersebut membutuhkan dua parameter, yaitu dataset atau tabel apa yang akan divisualisasikan (didefinisikan sebagai variabel ‘df’) dan nilai ‘hue’ yang merepresentasikan atribut mana yang berperan sebagai target atau label kelas. Gambar di bawah adalah hasil keluaran fungsi **.pairplot()**.



## 5. Detect and Handle Missing Values

Kita bisa melihat data yang tidak memiliki nilai (missing value) yang ada dari tahapan statistik deskriptif menggunakan `.isnull()`.

**Langkah yang bisa kita lakukan setelah mengetahui adanya missing value** adalah **menghapus** atau **mengisi** dengan nilai lain. Perlakuan yang diambil dapat ditentukan dari berbagai pertimbangan seperti melihat jumlah data yang hilang, tipe data yang hilang, atau nilai dari kolom tersebut.

Sebagai contoh:

Misalnya null value yang hilang hanya merupakan sebagian kecil dari data, maka yang bisa dilakukan adalah menghapus baris tersebut karena hilangnya data tidak akan memberikan pengaruh besar.

Menghapus data adalah hal yang tidak dianjurkan apabila hasilnya justru menghilangkan kegunaan atau insight yang bisa diambil dari dataset. Sehingga, cara alternatif yang dapat ditempuh adalah mengisi data tersebut dengan rata-rata, modus, median, atau yang lain, sesuai dengan persebaran dan tipe data.



Kita dapat menuliskan script berikut untuk menampilkan berapa banyak missing value tiap atribut:



```
df.isnull().sum()
```

Diketahui bahwa atribut ‘deck’ merupakan atribut yang memiliki banyak missing value, yaitu 688 baris. Artinya, sekitar 77% data atribut tersebut tidak ada nilainya. Selain itu, atribut ‘age’ juga tergolong banyak yang missing value dimana hampir 20% datanya tidak ada.

Seperti yang sudah kita bahas sebelumnya, cara untuk mengatasi missing value adalah menghapus atau menggantinya dengan nilai yang lain. Yang harus diingat adalah tidak ada aturan pasti mengenai langkah yang harus diambil. Kita bisa menghilangkan semua baris data yang missing value atau menggunakan keduanya sekaligus.

Sebagai contoh, kita menghapus kolom ‘age’ dan ‘deck’ karena terlalu banyak missing value. Sementara itu, kita mengganti baris data yang hilang pada atribut ‘embarked’ dan ‘embark\_town’.

<b>survived</b>	0
<b>pclass</b>	0
<b>sex</b>	0
<b>age</b>	177
<b>sibsp</b>	0
<b>parch</b>	0
<b>fare</b>	0
<b>embarked</b>	2
<b>class</b>	0
<b>who</b>	0
<b>adult_male</b>	0
<b>deck</b>	688
<b>embark_town</b>	2
<b>alive</b>	0
<b>alone</b>	0
<b>dtype: int64</b>	

Tuliskan script berikut untuk menghilangkan ‘deck’ dan ‘age’ dari dataset:

```
● ● ●  
  
# Drop age and deck column  
df = df.drop(columns = ['age', 'deck'])  
df
```

Selanjutnya, kita mengganti baris data yang hilang pada atribut ‘embarked’ dan ‘embark\_town’. Kita dapat mengganti baris data yang kosong dengan nilai yang paling sering di tiap atribut. Pertama. coba tuliskan script berikut untuk mengetahui nilai yang paling sering muncul pada tiap atribut.

```
● ● ●  
  
most_embarked = df['embarked'].value_counts().idxmax()  
print("The value that appears most often of Embarked attributes:  
{}}".format(most_embarked))  
most_embark_town = df['embark_town'].value_counts().idxmax()  
print("The value that appears most often of Embark Town attributes:  
{}}\n".format(most_embark_town))
```

Diketahui bahwa nilai paling sering muncul pada ‘embarked’ adalah “S” dan nilai paling sering muncul pada ‘embark\_town’ adalah “Southampton”. Oleh karena itu, kita bisa mengganti nilai missing value dengan kedua nilai tersebut pada masing-masing atribut.

```
The value that appears most often of Embarked attributes: S  
The value that appears most often of Embark Town attributes: Southampton
```

Tuliskan script berikut untuk menggantikan missing value dengan nilai yang paling sering muncul pada masing-masing atribut:



```
df['embarked'] = df['embarked'].fillna(most_embarked)  
df['embark_town'] = df['embark_town'].fillna(most_embark_town)
```

Apabila kita cek lagi menggunakan script “df.isnull().sum()”, sudah tidak ada lagi atribut yang memiliki missing value

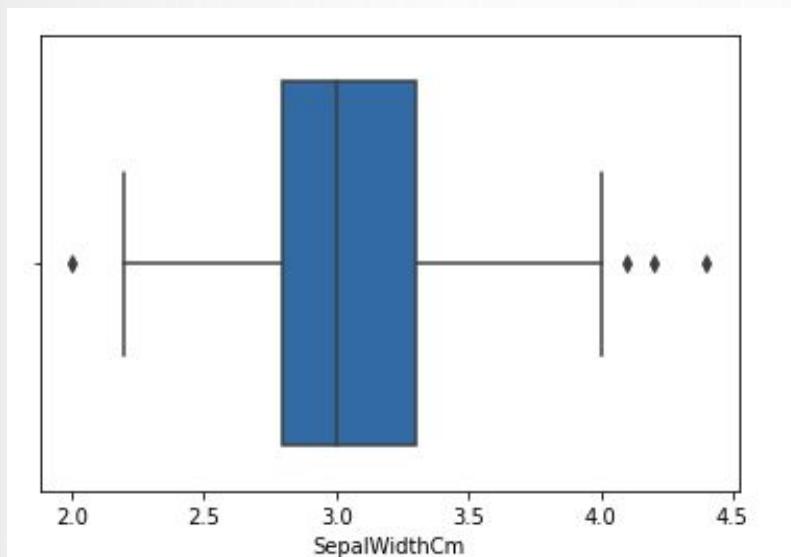
<b>survived</b>	0
<b>pclass</b>	0
<b>sex</b>	0
<b>sibsp</b>	0
<b>parch</b>	0
<b>fare</b>	0
<b>embarked</b>	0
<b>class</b>	0
<b>who</b>	0
<b>adult_male</b>	0
<b>embark_town</b>	0
<b>alive</b>	0
<b>alone</b>	0

## 6. Detect and Handle Outlier

Salah satu langkah untuk melihat data outlier adalah melalui visualisasi boxplot. Seperti yang kita tahu, data outlier adalah data yang menyimpang secara signifikan dari data lainnya. Hal ini dapat disebabkan oleh kesalahan pengukuran atau eksekusi. Analisis untuk deteksi outlier disebut sebagai outlier mining dimana banyak cara untuk mendeteksi outlier seperti menghilangkannya.

Visualisasi boxplot memiliki peranan penting dalam menangani data outlier. Kita dapat melihat berapa banyak data yang terindikasi outlier melalui visualisasi tersebut. Seperti pembahasan sebelumnya, data outlier biasanya digambarkan berupa diamond atau titik di luar garis yang menghimpit kotak seperti gambar di bawah. Semakin banyak gambar titik, maka semakin banyak yang terindikasi sebagai outlier.

Kita dapat melihat data yang terindikasi outlier menggunakan interkuartil. Data yang berada kurang dari kuartil 1 dan lebih dari kuartil 3 biasanya terindikasi outlier. Sebenarnya terdapat perbedaan pendapat mengenai mengatasi data outlier karena beberapa data scientist ada yang lebih suka menghilangkannya dan ada yang tidak. Hal ini dikarenakan adanya potensi dataset kita akan berkurang drastis dengan menghilangkan outlier. Oleh karena itu, sebelum kita memutuskan menghapus data outlier atau tidak, lebih baik menggunakan boxplot untuk memperoleh gambaran berapa banyak yang hilang apabila kita memutuskan untuk menghilangkan data outlier.

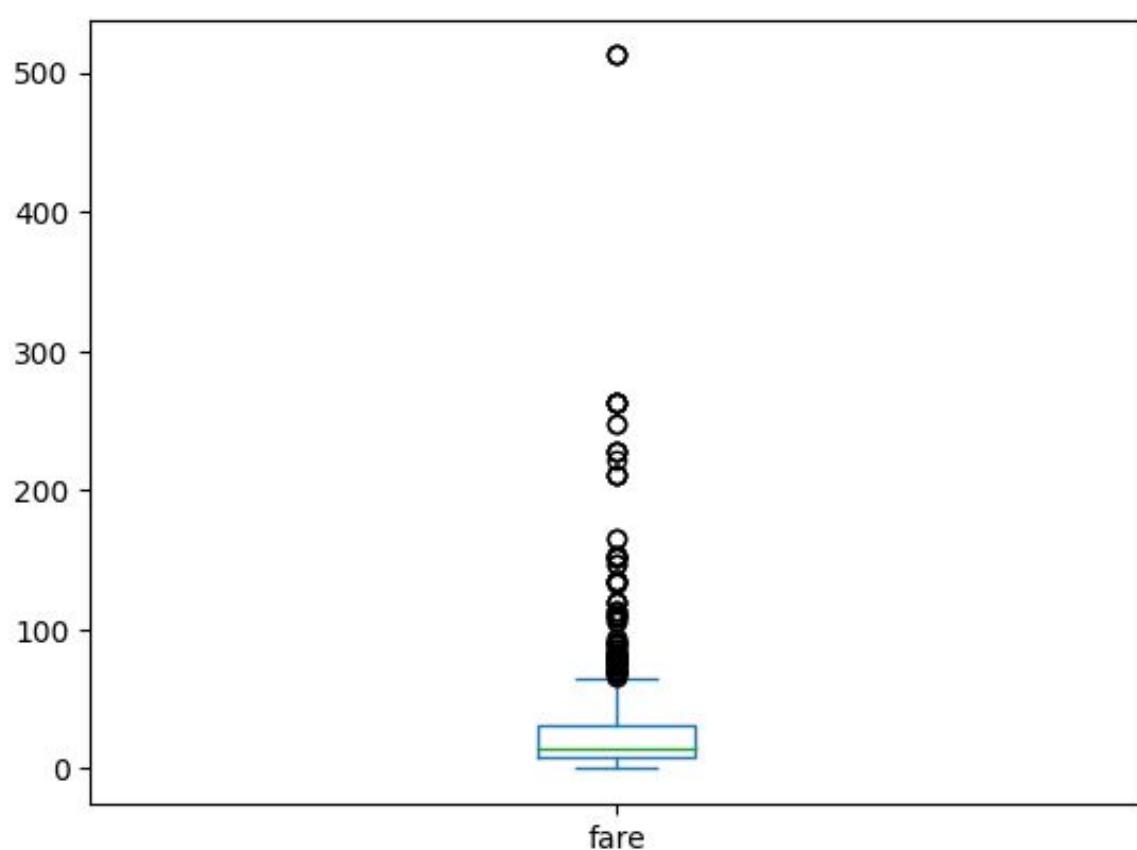


Kita dapat menuliskan script berikut untuk menampilkan apakah ada data terindikasi outlier dalam sebuah boxplot. Kali ini kita akan mencoba melihat apakah ada data yang terindikasi sebagai outlier pada atribut ‘fare’ yang merepresentasikan informasi terkait tarif yang dikeluarkan masing-masing penumpang.



```
df.fare.plot(kind = "box")
plt.show()
```

Hasilnya adalah seperti gambar di bawah. Terlihat bahwa ada banyak titik data yang berada di luar dua garis penghimpit kotak. Hal ini mengindikasikan bahwa dataset kita memiliki banyak data outlier. Apakah kita harus menghapusnya? Yuk kita coba periksa melalui analisis interkuartil.



Kita dapat menuliskan script berikut untuk menampilkan apakah ada data terindikasi outlier dalam sebuah boxplot. Kali ini kita akan mencoba melihat apakah ada data yang terindikasi sebagai outlier pada atribut ‘fare’ yang merepresentasikan informasi terkait tarif yang dikeluarkan masing-masing penumpang menggunakan python. Sebelum itu, coba tuliskan script berikut ya!

Pertama, kita memasukkan package numpy yang akan membantu kita menentukan nilai kuartil 1 dan kuartil 3 pada data. Kita memanfaatkan fungsi .percentile() dengan memasukkan array data dan array posisi kuartil yang diinginkan. Berhubung kita membutuhkan kuartil 1 dan 3, maka kita menuliskan [25,75]. Selanjutnya, kita membuat sebuah fungsi bernama outlier\_treatment() yang akan membantu menampilkan nilai yang berada pada posisi kuartil 1 dan 3.

```
● ● ●

# Importing the Library
import numpy as np

# Define the function to retrieve Q1 and Q3
def outlier_treatment(datacolumn):
    sorted(datacolumn)
    Q1,Q3 = np.percentile(datacolumn , [25,75])
    IQR = Q3 - Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    return lower_range,upper_range
```

Selanjutnya, tuliskan script berikut untuk melihat nilai kuartil 1 dan kuartil 3 beserta data mana saja yang memiliki nilai di bawah kuartil 1 dan di atas kuartil 3.

```
lowerbound,upperbound = outlier_treatment(df.fare)
print("Lowerbound (Q1): {} \nUpperbound (Q3): {} ".format(lowerbound, upperbound))
print("Outlier data in dataset: {} rows.\n".format(len(df[(df.fare < lowerbound) | (df.fare > upperbound)])))

df[(df.fare < lowerbound) | (df.fare > upperbound)]
```

Perhatikan gambar di bawah. Hasil keluaran program di atas adalah seperti berikut:

- Nilai kuartil 1 adalah -26,724 dan kuartil 3 adalah 65,6344. Bisa disimpulkan bahwa nilai yang berada di luar itu adalah data outlier
- Diketahui bahwa jumlah data yang terindikasi outlier sebanyak 116 baris. Hal ini bisa memberi kita gambaran berapa banyak baris yang hilang jika kita menghilangkan data outlier, yaitu sekitar 116 baris atau sekitar 13% dari data keseluruhan. Kita bisa menghilangkan data outlier jika dirasa jumlah baris tersebut tidak terlalu banyak

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
27	0	1	male	19.0	3	2	263.0000	S	First	man	True	C	Southampton	no	False
31	1	1	female	Nan	1	0	146.5208	C	First	woman	False	B	Cherbourg	yes	False
34	0	1	male	28.0	1	0	82.1708	C	First	man	True	Nan	Cherbourg	no	False
52	1	1	female	49.0	1	0	76.7292	C	First	woman	False	D	Cherbourg	yes	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
846	0	3	male	Nan	8	2	69.5500	S	Third	man	True	Nan	Southampton	no	False
849	1	1	female	Nan	1	0	89.1042	C	First	woman	False	C	Cherbourg	yes	False
856	1	1	female	45.0	1	1	164.8667	S	First	woman	False	Nan	Southampton	yes	False
863	0	3	female	Nan	8	2	69.5500	S	Third	woman	False	Nan	Southampton	no	False
879	1	1	female	56.0	0	1	83.1583	C	First	woman	False	C	Cherbourg	yes	False

Kita dapat menggunakan script berikut untuk menghilangkan data outlier



```
df.drop(df[(df.fare < lowerbound) | (df.fare > upperbound)].index ,  
inplace=True)
```

Gambar di bawah adalah dataset yang bisa dikatakan tidak ada data outlier. Terlihat bahwa jumlah data kita berkurang menjadi 775 baris data

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

775 rows × 15 columns

# Apa itu Categorical and Numerical Analysis?

Berdasarkan **sifatnya**, data dibedakan menjadi

## Kategorikal



Tipe data ini tidak memiliki nilai numerik dan **sifatnya lebih kepada kualitatif daripada kuantitatif**. Sesuai namanya, data tipe kategori **merupakan bagian dari beberapa kelompok**.



Misal, jika ada besar maka ada kecil, jika ada laki-laki maka ada perempuan, jika ada benar maka ada salah, jika ada ringan maka ada sedang kemudian berat.

## Numerikal

- 1
- 2
- 3

Tipe data yang hanya **dinyatakan dalam bentuk numerik dan tidak dinyatakan dengan penggunaan nama lain pada data**. Berdasarkan bentuknya, dapat dibedakan menjadi:

- **Variabel Kontinu**, dapat mengambil semua nilai apapun (dalam tingkatan apapun) dalam rentang waktu tertentu. (misal nilai apapun antara 3 dan 4)
- **Variabel Diskrit**, hanya bisa merepresentasikan data dalam bentuk bilangan bulat. (misalnya variabel yang mengambil nilai integer antara 0 dan 100)

Berdasarkan **tujuannya**, statistika dibedakan menjadi

### Deskriptif

**Mendeskripsikan data-data mentah** menggunakan statistik ringkasan, grafik, dan tabel.

Salah satu tools dasar yang digunakan adalah histogram/barchart untuk menggambarkan distribusi nilai untuk beberapa variabel dalam sampel subjek.

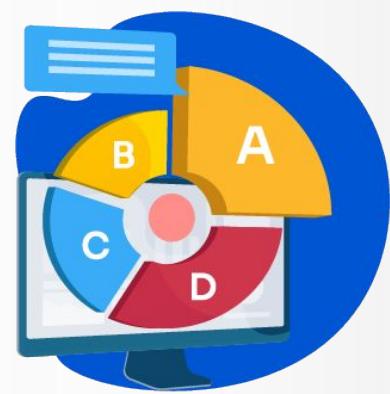
Secara garis besar, statistika deskriptif diawali dari pengambilan data (collecting data), menampilkan data dalam bentuk grafik atau tabel (present data), dan penarikan kesimpulan (characterize data).



Collecting Data



Present Data



Characterize Data

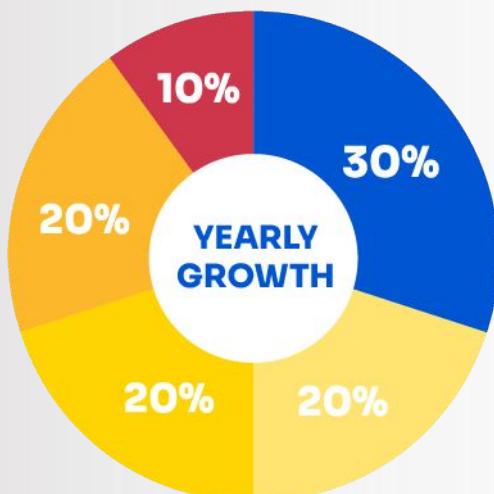
Berdasarkan **tujuannya**, statistika dibedakan menjadi

### Inferensial

**Menarik kesimpulan tentang populasi** yang lebih **berdasarkan pengambilan sampel** menggunakan sampel data yang kecil untuk kemudian berasal melalui uji hipotesis.

Analisis statistik **diawali pembuatan klaim atau hipotesa** terhadap suatu populasi (misalkan populasi mahasiswa Universitas X mayoritas berasal dari Surabaya).

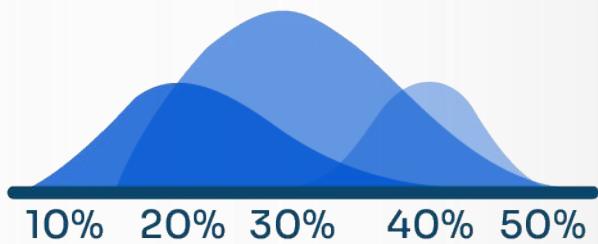
Selanjutnya, **dilakukan pengujian suatu klaim tersebut** tentang apakah benar mahasiswa Universitas X mayoritas berasal dari Surabaya.



**DESCRIPTIVE STATISTICS**  
described data

### INFERENTIAL STATISTICS

studies a sample  
of the same data



Di dalam statistika inferensial menyimpulkan tentang populasi dan sampel. Sebenarnya apa itu populasi dan sampel?

## Populasi

Sejumlah sesuatu yang kita amati, seperti manusia, peristiwa, hewan dll. Ini memiliki beberapa parameter seperti mean, median, modus, standar deviasi, dan lain-lain.



**Merupakan totalitas dari semua objek** yang hendak diteliti. (misal kita ingin mengamati seluruh masyarakat Bogor).

## Sampel

### Bagian acak dari populasi

Biasanya kita menggunakan sampel ketika populasinya cukup besar sehingga sulit untuk menganalisis seluruh rangkaian.

Contoh:

Berhubung pengamatan pada seluruh masyarakat Bogor terlalu luas, maka kita mencoba mengambil sampel pada 2 kecamatan saja.



# Ukuran Pemusatan Data

Merupakan nilai tunggal yang **mewakili** suatu **kumpulan data**.

Kita dapat **memahami karakteristik data melalui satu nilai**.

- **Mean**

Titik di mana distribusi berada dalam keseimbangan.

Titik tumpu atau titik keseimbangan dihitung sebagai mean atau mean aritmatika.

## Rumus Rataan Populasi

$$\mu = \frac{\sum_{i=1}^N xi}{N}$$

## Rumus Rataan Sampel

$$\bar{x} = \frac{\sum_{i=1}^n xi}{n}$$

$x_i$  → data ke i

N → banyaknya data populasi

n → banyaknya data sampel

- **Median**

**Nilai tengah** yang diukur dari nilai terendah hingga nilai tertinggi. Cara menemukan nilai median adalah sebagai berikut:

- Hitung jumlah data yang dimiliki
- Pastikan data telah urut dari nilai terkecil ke terbesar
- Tentukan posisi nilai median

Apabila jumlah data yang dimiliki ganjil, gunakanlah persamaan ini

$$me = \frac{X_{\frac{n+1}{2}}}{2}$$

Apabila jumlah data yang dimiliki genap, gunakanlah persamaan ini

$$me = \frac{X_{\frac{n}{2}} + X_{\frac{n}{2}+1}}{2}$$

## Yuk kita latihan sedikit!

Hitunglah nilai rataan dan median data di bawah ini

Kelurahan	Kasus
Airlangga	4610
Alun-Alun Contong	2327
Ampel	1306
Asem Rowo	4314
Babat Jerawat	4253
Babatan	8089
Balas Klumprik	2920
Balongsari	1878

Rata-Rata

$$= \frac{4610 + 2327 + 1306 + 4314 + 4253 + 8089 + 2920 + 1878}{8}$$

$$= \frac{34299}{8} = 4287,375$$

Jumlah data ( $n$ ) = 8

$$me = \frac{n}{2} = 4$$

$$me = \frac{X_{(4)} + X_{(5)}}{2}$$

$$= \frac{2920 + 4253}{2}$$

$$= \frac{7173}{2} = 3586,5$$

- **Modus**

**Nilai yang paling sering muncul** di dataset.

Jika terdapat beberapa nilai dengan frekuensi paling sering muncul sama, maka dataset bisa mempunyai lebih dari 1 modus.

Perhatikan tabel di bawah. frekuensi kemunculan nilainya adalah sebagai berikut:

- 35.000 → 1 x
- 40.000 → 1 x
- 45.000 → 3 x
- 55.000 → 1 x
- 60.000 → 2 x

Terlihat kalau 45.000 paling sering muncul yang mengindikasikan nilai tersebut adalah nilai modus.

Wilayah	Gaji Manager (in \$)
Arizona	60.000
California	45.000
Ohio	60.000
Texas	55.000
West Virginia	45.000
Illinois	45.000
Colorado	40.000
New Jersey	35.000

Terkadang kita tidak cukup dengan hanya melihat satu nilai. Oleh karena itu, ada kalanya kita harus mengetahui data lebih menyeluruh menggunakan dispersi. **Melalui dispersi**, kita bisa mengetahui **rentang data**, **persebaran data** kita bagaimana, atau bahkan bisa mengetahui seberapa **ragam data** kita.

- **Variasi**

Jumlah deviasi kuadrat (jarak) setiap skor dari rata-rata dibagi dengan jumlah skor dalam kumpulan data.

#### Rumus pada Populasi

$$\sigma^2 = \frac{\sum |X - \mu|^2}{N}$$

#### Rumus pada Sampel

$$s^2 = \frac{\sum (X - \bar{X})^2}{n-1}$$

X → nilai data

N → banyaknya data populasi

n → banyaknya data sampel

$\mu$  dan  $\bar{x}$  → nilai rata-rata

- **Standar Deviasi**

### Akar kuadrat dari variasi

Statistik yang berguna dan dapat diinterpretasikan karena mengambil akar kuadrat dari variasi.

#### Rumus pada Populasi

$$\sigma = \sqrt{\frac{\sum |X - \mu|^2}{N}}$$

#### Rumus pada Sampel

$$S = \sqrt{\frac{\sum (X - \bar{X})^2}{n-1}}$$

X → nilai data

N → banyaknya data populasi

n → banyaknya data sampel

$\mu$  dan  $\bar{x}$  → nilai rata-rata

## Yuk kita latihan sedikit!

Rektor Universitas B ingin mengetahui data jumlah mahasiswa dengan mengambil sampel terhadap 6 jurusan. Berikut banyaknya siswa yang mendaftar di tiap jurusan: 56, 75, 53, 49, 80, 24.

- Berapa rata-rata deviasinya?
- Hitunglah nilai variansnya!

$$s^2 = \frac{(56-56,17)^2 + (75-56,17)^2 + (53-56,17)^2 + (49-56,17)^2 + (80-56,17)^2 + (24-56,17)^2}{6-1}$$

$$= \frac{0,03 + 354,57 + 10,05 + 51,41 + 567,87 + 1034,91}{5} = \frac{2018,84}{5} = 403,77$$

$$s = \sqrt{403,77} = 20,09$$

Standar deviasi juga bisa kita libatkan untuk mengetahui bagaimana hubungan dengan nilai rata-rata suatu kelompok data yaitu dengan perhitungan Z-Score.

- **Z-Score**

Z-Score dapat mengidentifikasi seberapa jauh nilai standar deviasi terhadap nilai rata-rata. Oleh karena itu, kita membutuhkan nilai standar deviasi. Perhatikan gambar di bawah, apabila Z-Score sama dengan 0, maka mengindikasikan nilai titik data tersebut identik dengan rata-rata. Sebaliknya, semakin jauh Z-Score dari 0 maka semakin tidak identik dengan nilai rata-rata.

Apabila standar deviasi populasi (dinotasikan  $\sigma$ ) diketahui atau jumlah sampel data lebih dari 30, maka kita bisa menggunakan persamaan berikut:

$$Z = (X - \mu) / \sigma$$

Apabila standar deviasi populasi (dinotasikan  $\sigma$ ) tidak diketahui atau ukuran sampel kurang dari 30, maka kita menggunakan standar deviasi pada sampel yang direpresentasikan dengan  $S$  seperti di persamaan berikut:

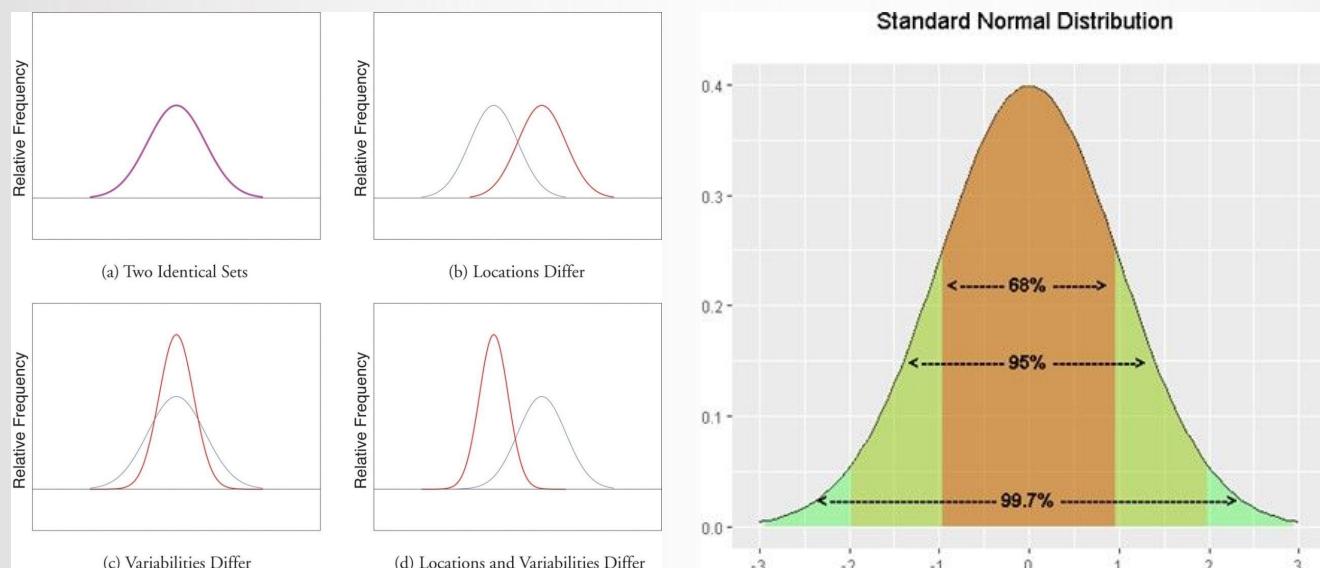
$$Z = (X - \mu) / S$$

Walaupun terdapat sedikit perbedaan, terlihat dari kedua persamaan di atas kita membutuhkan nilai  $X$  yang merepresentasikan nilai pengamatan. Selain itu, Z-Score mengikuti aturan distribusi normal dimana distribusi tersebut cukup sering digunakan.

# Apa kaitan antara Distribusi Normal dan Standar Deviasi?

**Standar deviasi adalah ukuran dari variabilitas yang sangat berguna ketika distribusi normal atau mendekati normal karena proporsi dari distribusi dalam nilai standar deviasi dari mean dapat dihitung. Untuk distribusi normal :**

- Sekitar 68% dari data berada dalam satu standar deviasi dari mean
- Sekitar 95% dari data berada dalam dua standar deviasi dari rata-rata
- Lebih dari 99% data berada dalam tiga simpangan baku rata-rata



# Apa itu Automated Exploratory Data Analysis?

Proses EDA juga bisa dilakukan secara otomatis sehingga kita tidak harus melakukannya secara manual. Kita bisa menggunakan beberapa *package* Python, seperti:

- dtale
- pandas profiling
- sweetviz
- autoviz



## Automated Exploratory Data Analysis



Terdapat beberapa tools yang dapat digunakan untuk menerapkan automated exploratory data analysis seperti pandas profiling, dtale, sweetviz, dan autoviz. Supaya bisa lebih paham penggunaan Automated Exploratory Data Analysis, yuk kita ulas satu per satu!

Pertama, kita bisa mencoba pada dataset Titanic yang disediakan oleh package Seaborn. Tuliskan syntax berikut untuk memasukkan dataset Titanic



```
# importing library
import seaborn as sns

df = sns.load_dataset("titanic")
```

- **Pandas Profiling**

Pandas Profiling adalah salah satu package python digunakan untuk EDA yang dapat menghasilkan laporan kerangka data Anda dalam berbagai format.

Sebelumnya kita sudah membahas penggunaan .describe(), tetapi syntax tersebut tidak memberikan laporan yang cukup komprehensif. Oleh karena itu kita bisa menggunakan pandas profiling. Berikut cara implementasi pandas profiling seperti terlihat di gambar berikut:

- Install package menggunakan syntax berikut yang kemudian kita masukkan package nya:

```
!pip install pandas-profiling
```

- Mendefinisikan variabel pandas profiling dengan memasukkan variable dataset (misal kita menamakan variabel dataset kita dengan “df”)
- Selanjutnya kita memanggil fungsi **.to\_file(output\_file=“report.html”)**. Artinya, kita akan me-export sebuah file berformat html yang akan berisi laporan hasil EDA oleh pandas profiling



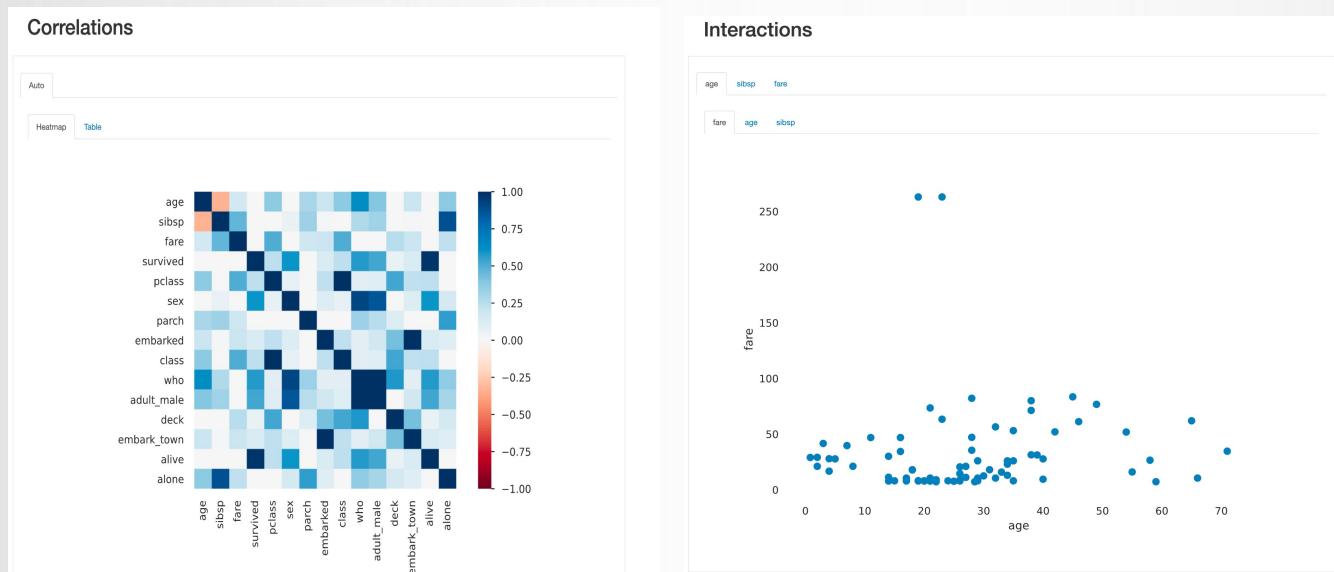
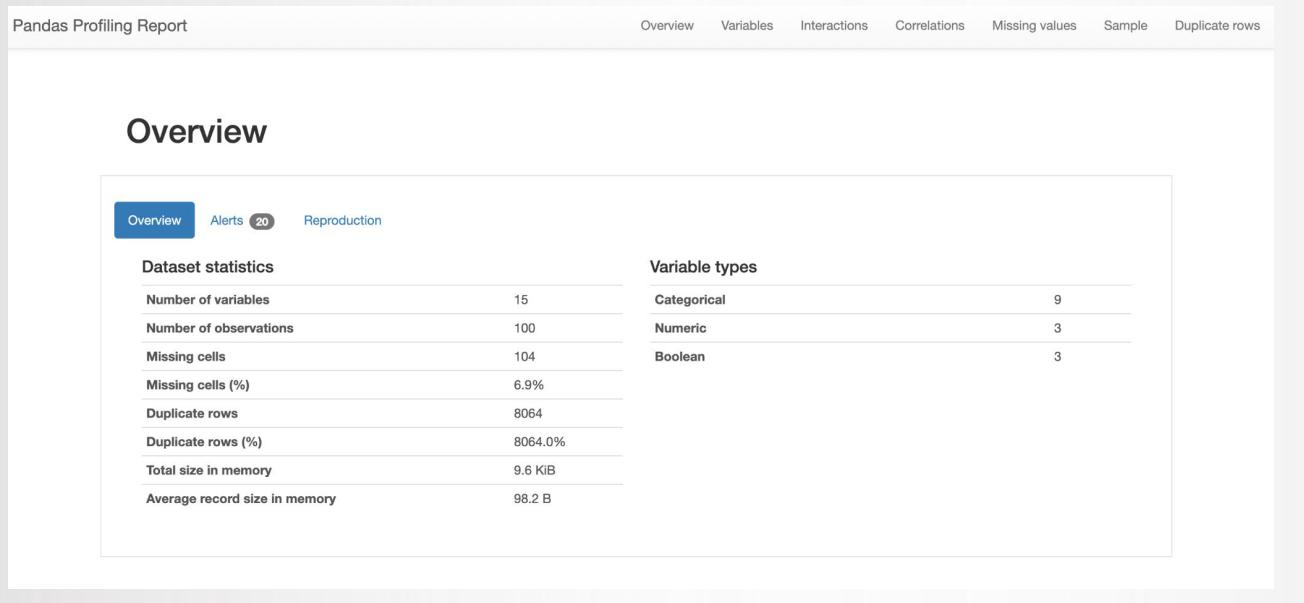
```
# installing package
!pip install pandas-profiling

#importing library
from pandas_profiling import ProfileReport

profile = ProfileReport(df)
# exporting your report as a html file
profile.to_file(output_file='report.html')
```

Hasil EDA oleh Pandas Profiling adalah seperti ini. Terlihat terdapat beberapa menu yang bisa diakses untuk mengakses hasil EDA secara cepat seperti melihat sampel data, missing values, dan data duplikat.

Kita bisa menyimpan hasil EDA dalam format .html sehingga kita bisa mengakses kapan pun. Namun yang harus menjadi catatan adalah package ini cukup membutuhkan banyak resource untuk mengimplementasikannya.



- **Dtale**

D-Tale mengkombinasikan backend dengan framework Flask dan front-end dengan framework React untuk menyediakan antarmuka yang mudah digunakan untuk melihat dan menganalisis struktur dataset. Package ini juga telah terintegrasi dengan notebook ipython dan terminal python/ipython.

Package ini sudah mendukung fungsi yang disediakan package Pandas seperti DataFrame, Series, MultiIndex, DatetimeIndex, dan RangeIndex. Selain itu, juga bisa digunakan berbagai visualisasi, termasuk peta panas, bagan, dan plot tiga dimensi. Gambar berikut adalah contoh penggunaan Dtale dalam bahasa pemrograman python.



```
# Installing the library
!pip install dtale

# Importing the library
import dtale

import dtale.app as dtale_app
dtale_app.USE_COLAB = True
dtale.show(df)
```

Gambar di bawah merupakan hasil implementasi dtale. Secara default, kita akan disajikan sebuah link yang akan mengarah ke tabel seperti pada gambar. Hal ini tentu memudahkan kita dalam melihat data secara keseluruhan karena seperti yang kita tahu, secara default package pandas hanya menampilkan beberapa baris data saja.

▶	15	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
891	0	0	3	male	22.00	1	0	7.25	S	Third	man	<input checked="" type="checkbox"/>	nan	Southampton	no	<input type="checkbox"/>
1	1	1	female	38.00	1	0	71.28	C	First	woman	<input type="checkbox"/>	C	Cherbourg	yes	<input type="checkbox"/>	
2	1	3	female	26.00	0	0	7.93	S	Third	woman	<input type="checkbox"/>	nan	Southampton	yes	<input checked="" type="checkbox"/>	
3	1	1	female	35.00	1	0	53.10	S	First	woman	<input type="checkbox"/>	C	Southampton	yes	<input type="checkbox"/>	
4	0	3	male	35.00	0	0	8.05	S	Third	man	<input checked="" type="checkbox"/>	nan	Southampton	no	<input checked="" type="checkbox"/>	
5	0	3	male	nan	0	0	8.46	Q	Third	man	<input checked="" type="checkbox"/>	nan	Queenstown	no	<input checked="" type="checkbox"/>	

Sama seperti pandas profiling, terdapat menu navigasi yang memudahkan menganalisis data seperti melihat tren time-series, data duplikat, dan missing value. Selain itu juga terdapat beberapa menu visualisasi seperti heatmap dan menunjukkan baris mana yang terdapat missing value dan outlier.

Secara sekilas, package ini lebih baik daripada pandas profiling dari segi resource memori.



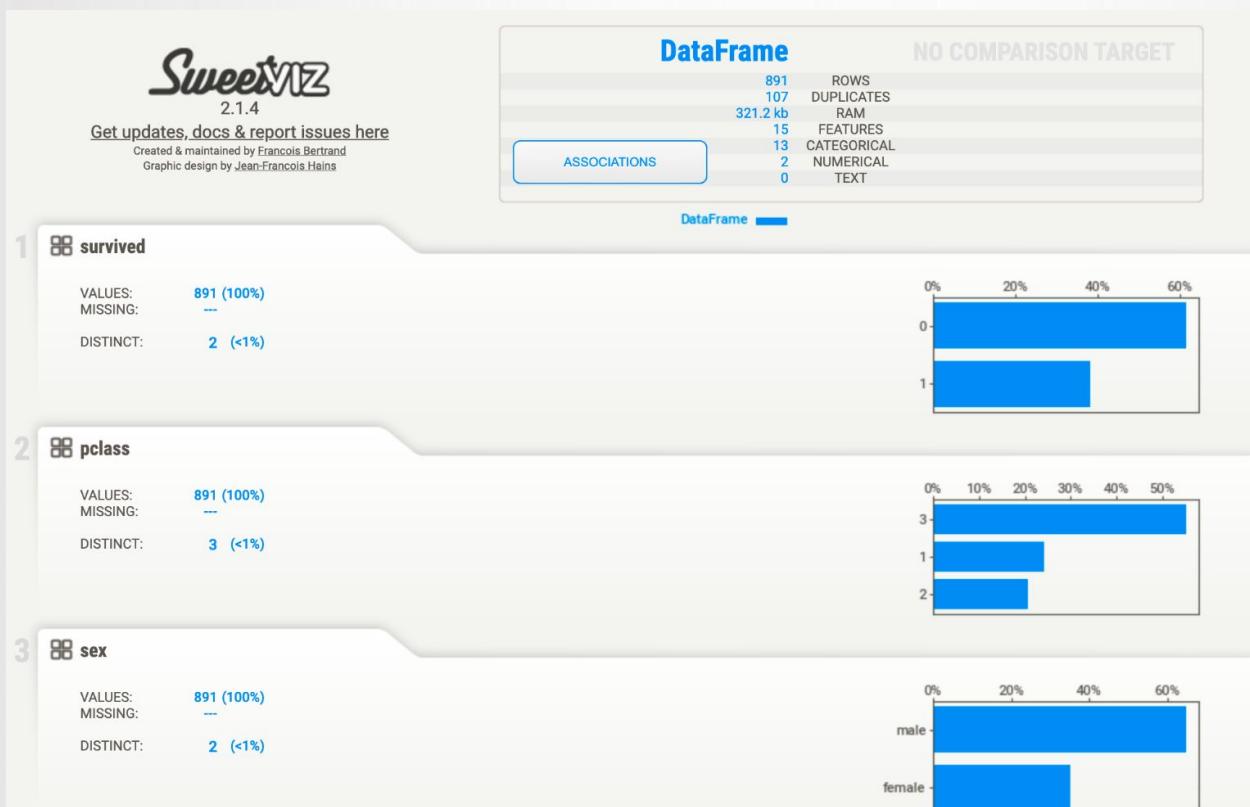
- **Sweetviz**

Sweetviz adalah package yang bisa menampilkan visualisasi untuk EDA hanya dengan beberapa baris kode seperti pada gambar berikut.

```
# Installing library
!pip install sweetviz

#importing sweetviz and visualizing our data
import sweetviz as sv
sweet_report = sv.analyze(df)
sweet_report.show_html('sweetviz_report.html')
```

Sama seperti pandas profiling, package ini akan me-export sebuah file berformat .html yang menampilkan bagaimana hubungan antar atribut.



- **Autoviz**

AutoViz secara otomatis memvisualisasikan dataset dengan satu baris kode. Selain itu, package ini juga mampu mengidentifikasi fitur yang paling penting dan memplot visualisasi yang menarik secara visual hanya berdasarkan fitur yang dipilih secara otomatis tersebut. AutoViz terkenal sangat cepat dalam menghasilkan visualisasi dalam hitungan detik. Berikut adalah contoh implementasi menggunakan python.



```
# Installing the library
pip install autoviz

# Importing the library
from autoviz.AutoViz_Class import AutoViz_Class

AV = AutoViz_Class()
#you have to specify the target variable
dft = AV.AutoViz(filename= "", dft = df, depVar = 'Survived')
```

Berbeda halnya dengan yang sudah kita bahas sebelumnya seperti pandas profiling, dtale, dan sweetviz, package ini tidak menghasilkan file tertentu seperti format html maupun sebuah link yang dapat kita akses. Keluaran program adalah sebuah rangkuman seperti informasi dimensi data, jumlah missing value, dan data unik. Selain cepat dalam prosesnya, package ini memiliki fitur rekomendasi langkah pemrosesan data tiap atribut dimana hal ini tidak tersedia di package lain.

	Nullpercent	NuniquePercent	dtype	N uniqueness	Nulls	Least num. of categories	Data cleaning improvement suggestions
deck	77.216611	0.785634	category	7	688	4	fill missing values
age	19.865320	9.876543	float64	88	177	0	fill missing values
embarked	0.224467	0.336700	object	3	2	77	fill missing values, fix mixed data types
embark_town	0.224467	0.336700	object	3	2	77	fill missing values, fix mixed data types
survived	0.000000	0.224467	int64	2	0	0	
pclass	0.000000	0.336700	int64	3	0	0	
sex	0.000000	0.224467	object	2	0	314	
sibsp	0.000000	0.785634	int64	7	0	0	
parch	0.000000	0.785634	int64	7	0	0	
fare	0.000000	27.833895	float64	248	0	0	right skewed distribution: cap or drop outliers
class	0.000000	0.336700	category	3	0	184	

# Apa itu Reproducible Scientific Analysis ?

Dalam beberapa dekade terakhir, banyak ilmuwan memikirkan bagaimana karya ilmiah di berbagai bidang dapat direproduksi (reproducibility).

Sebuah karya ilmiah dikatakan dapat direproduksi jika:

- **Seluruh kode dan data yang digunakan untuk menghasilkan angka dan gambar** dalam suatu penelitian tersedia
- Dapat **digunakan untuk mendapatkan hasil yang sama**

Tiga aspek dalam reproduksi karya ilmiah, antara lain:

## 1. Reproduksi Metode

Kemampuan untuk **mengimplementasikan prosedur eksperimen/komputasi** dengan semirip mungkin, dengan data dan tools yang sama, untuk mendapatkan hasil yang sama

## 1. Reproduksi Hasil

Kemampuan untuk **mendapatkan hasil yang sama** melalui riset independen menggunakan metode yang semirip mungkin dengan riset awal

## 1. Reproduksi Inferensial

Kemampuan untuk **menarik kesimpulan dari replikasi** atau analisis ulang suatu riset

Dalam lingkup Data Science, reproducible scientific analysis diidentifikasi sebagai pertimbangan yang penting dalam perkembangan prosesnya. Hal ini dilatarbelakangi oleh adanya potensi hasil yang salah dalam beberapa pendekatan yang diakibatkan oleh kurangnya pengetahuan mengenai proses/metode yang sedang dilakukan.

Analisis ilmiah dalam proses Data Science dapat membantu seorang Data Scientist untuk memahami suatu proses pengerjaan data yang sudah dilakukan sebelumnya. Analisis ilmiah dalam proses Data Science dapat didapatkan melalui open source coding environment seperti Python dan R Markdown menggunakan tools seperti Jupyter Notebook dan R Markdown.



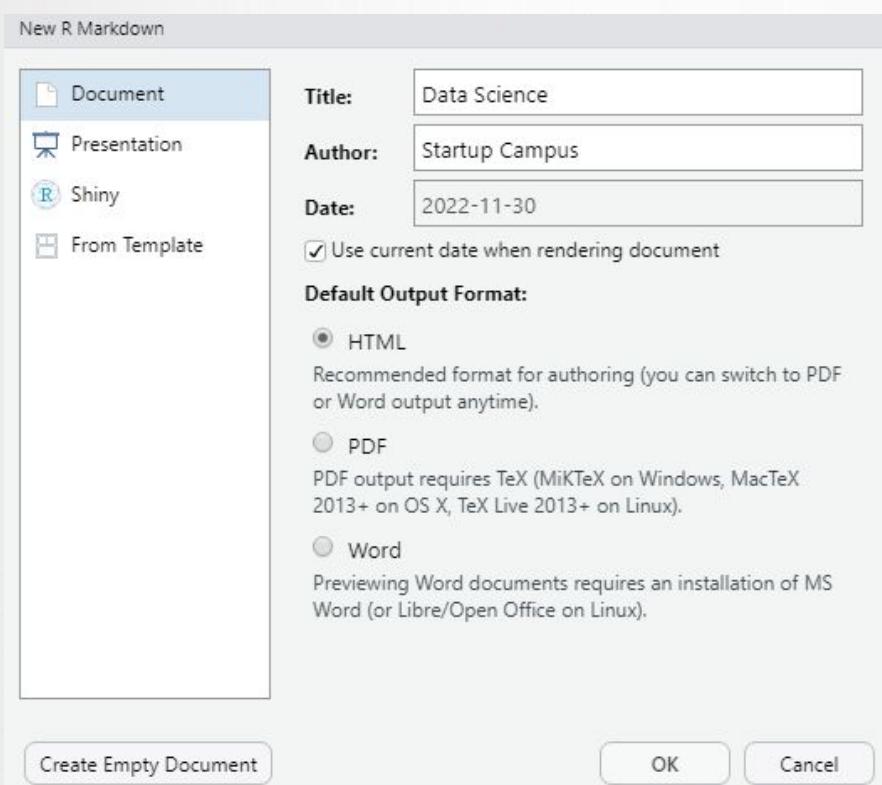
# Bagaimana Cara Menerapkan Scientific Analysis?

Salah satu tool yang dapat digunakan untuk memunculkan analisis ilmiah adalah **R Markdown**.

R Markdown adalah tool yang dapat digunakan untuk mengkombinasikan laporan dan analisis dalam satu dokumen. Tool ini **memungkinkan pengguna untuk menulis report bersamaan dengan teks naratif** untuk menjelaskan report tersebut.

Ada beberapa langkah yang dapat dilakukan:

1. Buka aplikasi R Studio
2. Pada menu File, klik R Markdown
3. Akan muncul dialog box seperti gambar di samping, isikan Title dan Author, pilih jenis file, lalu klik OK



4. Klik Save, lalu simpan file dalam format .rmd
5. Berikut adalah tampilan setelah dilakukan Save

```

1 --
2 title: "Data Science"
3 author: "Startup Campus"
4 date: ``r Sys.Date()``
5 output: html_document
6 ---
7
8 ````{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 -
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for
authoring HTML, PDF, and MS Word documents. For more details on using R Markdown
see <http://rmarkdown.rstudio.com>.
15
16 when you click the **Knit** button a document will be generated that includes both
content as well as the output of any embedded R code chunks within the document.
You can embed an R code chunk like this:
17
18 ````{r cars}
19 summary(cars)
20 -

```

6. Klik Knit pada bagian atas untuk melihat analisis ilmiah

## Data Science

Startup Campus

2022-11-30

### R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```

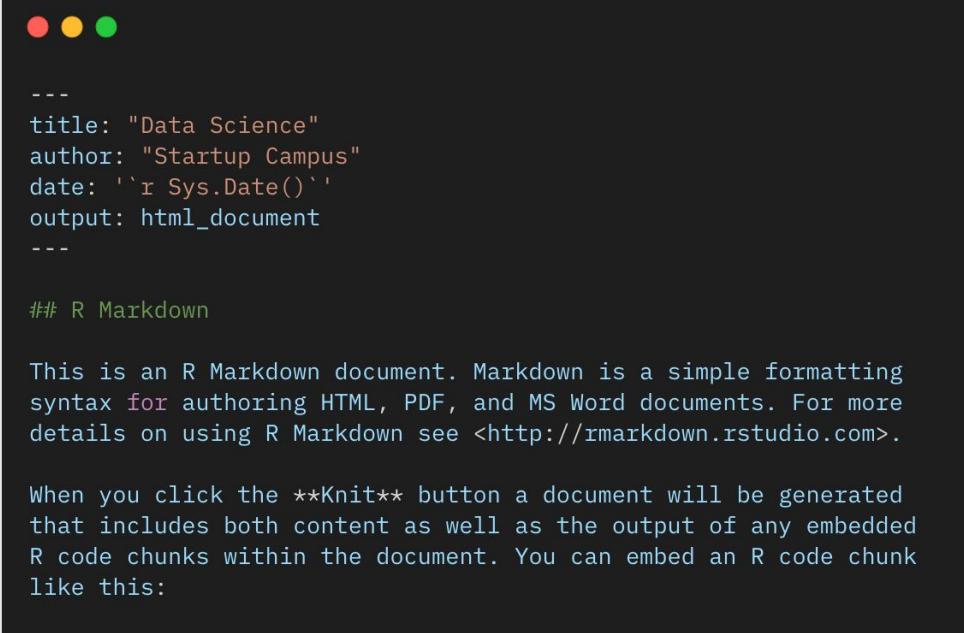
##      speed      dist
## Min.   : 4.0   Min.   : 2.00
## 1st Qu.:12.0   1st Qu.: 26.00
## Median :15.0   Median : 36.00
## Mean   :15.4   Mean   : 42.98
## 3rd Qu.:19.0   3rd Qu.: 56.00
## Max.   :25.0   Max.   :120.00

```

Secara umum, elemen R Markdown dibagi menjadi dua, antara lain:

## 1. Metadata

Bagian yang **memungkinkan pengguna untuk menuliskan teks naratif** mengenai kode yang sedang dijalankan. Pengguna dapat memberikan deskripsi/informasi/keterangan mengenai proses yang sedang berjalan dalam bagian ini.



```
---  
title: "Data Science"  
author: "Startup Campus"  
date: ``r Sys.Date()``  
output: html_document  
---  
  
## R Markdown  
  
This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
  
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
```

Tampilan metadata dalam format HTML adalah sebagai berikut:

## Data Science

Startup Campus

2022-11-30

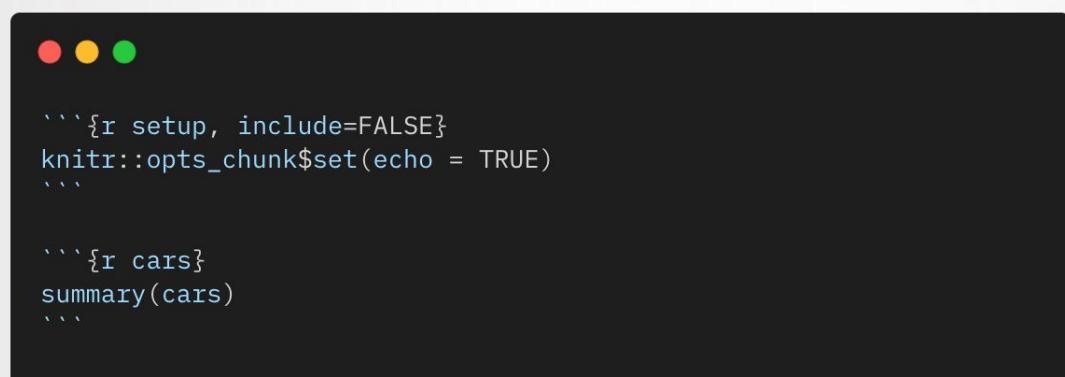
## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

## 2. Code Chunk

- a. Bagian yang **memungkinkan pengguna untuk menuliskan kode - kode yang dieksekusi** dalam proses yang sedang berjalan. Penulisan kode pada code chunk selalu diawali dan diakhiri dengan tanda petik `).



```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

```{r cars}
summary(cars)
```

```

Tampilan code chunk dalam format HTML adalah sebagai berikut:

```
summary(cars)

##      speed          dist
##  Min.   : 4.0   Min.   : 2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

File Rmarkdown dapat di-download [disini](#)

Silahkan akses link [berikut](#) lebih lanjut kalau membutuhkan referensi mempelajari lebih detail ya

Butuh demo versi video? Bisa cek di link [berikut](#) ya!

Selain menggunakan R, **analisis ilmiah juga dapat dilakukan menggunakan Python**. Penyusunan analisis ilmiah pada Python menggunakan LaTeX.

LaTeX adalah perangkat lunak untuk menyiapkan dokumen ilmiah yang biasa digunakan oleh researcher. Lateks kini sudah tersedia gratis di sebagian besar code editor online. Kalau ingin menggunakan python, kita bisa menggunakan package MathJax untuk me-render Lateks di dalam markdown/HTML. Untuk menggunakan LaTeX di notebook Jupyter, letakkan konten matematika yang ingin ditulis di dalam simbol '\$ ... \$' ganda '\$\$ ... \$\$'. Keluaran program file lateks disimpan dengan ekstensi (.tex). Selanjutnya, kita akan membahas contoh penggunaannya di python. Sebelum itu, jangan lupa install package-nya dulu ya! Dengan menulis “!pip install mathjax” seperti di gambar ini.

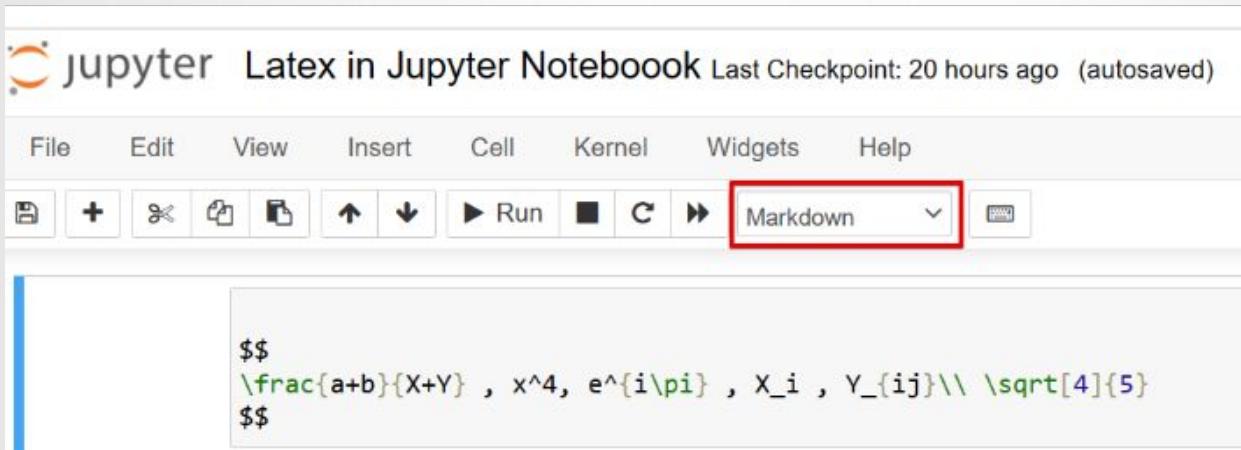


```
!pip install mathjax
```

## 1. Persamaan Matematika

Tuliskan syntax berikut untuk memunculkan persamaan matematika:

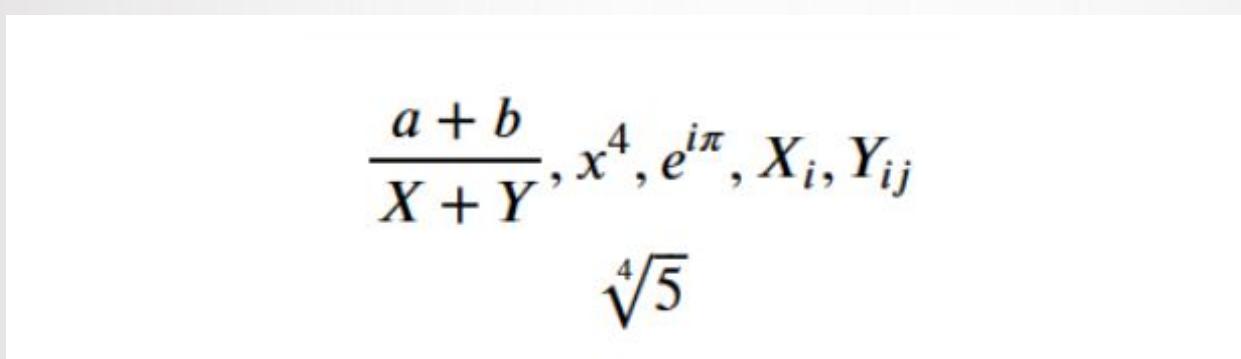
- Tulis “\” untuk spasi kecil
- Tulis “\frac{value1}{value2}” untuk memunculkan pecahan dimana value1 dan value2 adalah angka yang akan dibagi
- Tuliskan “^” dengan diikuti angka untuk menampilkan pangkat
- Tuliskan “\_{ }” untuk menampilkan indeks
- Tuliskan “\sqrt{n}{value}” untuk menampilkan akar dimana n adalah nominal akar sementara value adalah angka di dalam akar



The screenshot shows the Jupyter Notebook interface. The title bar reads "jupyter Latex in Jupyter Noteboook Last Checkpoint: 20 hours ago (autosaved)". The toolbar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A red box highlights the "Markdown" button in the toolbar. Below the toolbar is a cell containing the following LaTeX code:

```
$$\frac{a+b}{X+Y}, x^4, e^{i\pi}, X_i, Y_{ij}\sqrt[4]{5}
```

Output yang dihasilkan dari kode program di atas adalah sebagai berikut


$$\frac{a+b}{X+Y}, x^4, e^{i\pi}, X_i, Y_{ij}$$
$$\sqrt[4]{5}$$

## 2. Huruf Romawi dan Yunani

Kita bisa menampilkan beberapa simbol seperti sigma, lambda, gamma, dan theta seperti ini. Jangan lupa untuk menambahkan keyword `$$` di awal dan akhir kode. Untuk menampilkan simbol, tulis setelah keyword '`\`'.

The screenshot shows the Jupyter Notebook interface. The top navigation bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. Below the navigation bar is a toolbar with various icons for file operations like opening, saving, and running cells. A red box highlights the 'Markdown' button in the toolbar. The main workspace contains a code cell with the following LaTeX code:

```
$$\sigma, \Sigma \\ \delta, \Delta \\ \omega, \Omega \\ \gamma, \Gamma \\ \lambda, \Lambda \\ \theta, \Theta \\ $$
```

$\sigma, \Sigma$   
 $\delta, \Delta$   
 $\omega, \Omega$   
 $\gamma, \Gamma$   
 $\lambda, \Lambda$   
 $\theta, \Theta$

### 3. Set dan Logic

Kita bisa menampilkan simbol set dan logic dengan menulis syntax berikut.

Tulis notasi yang ingin ditampilkan setelah keyword '\'.

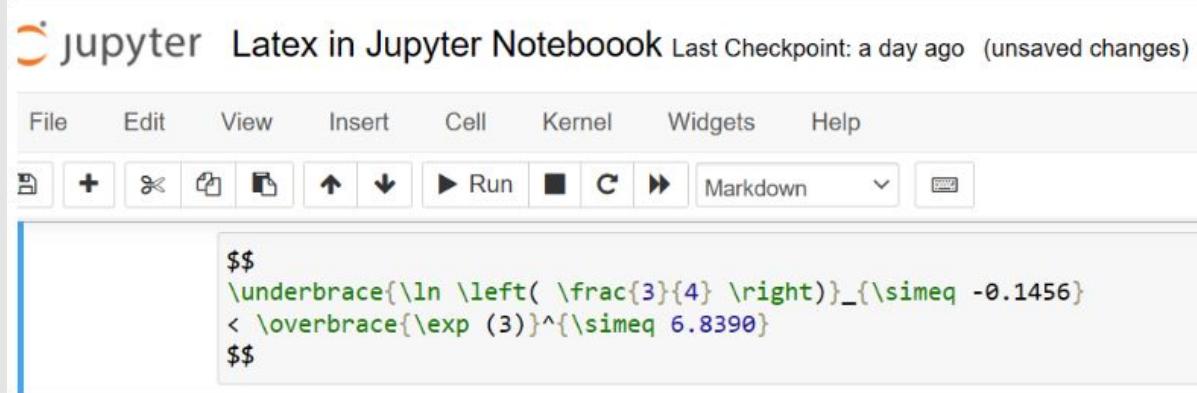
```
 $$\subset , \cap , \cup , \forall , \exists , \varnothing , \\emptyset$$
```

$\subset, \cap, \cup, \forall, \exists, \emptyset, \emptyset$

## 4. Tanda Kurung

Apabila membutuhkan penggunaan tanda kurung, bisa menuliskan ‘left(‘ untuk menampilkan tanda kurung sisi kiri dan ‘right)’ untuk tanda kurung sisi kanan.

$$\underbrace{\ln\left(\frac{3}{4}\right)}_{\approx -0.1456} < \overbrace{\exp(3)}^{\approx 6.8390}$$

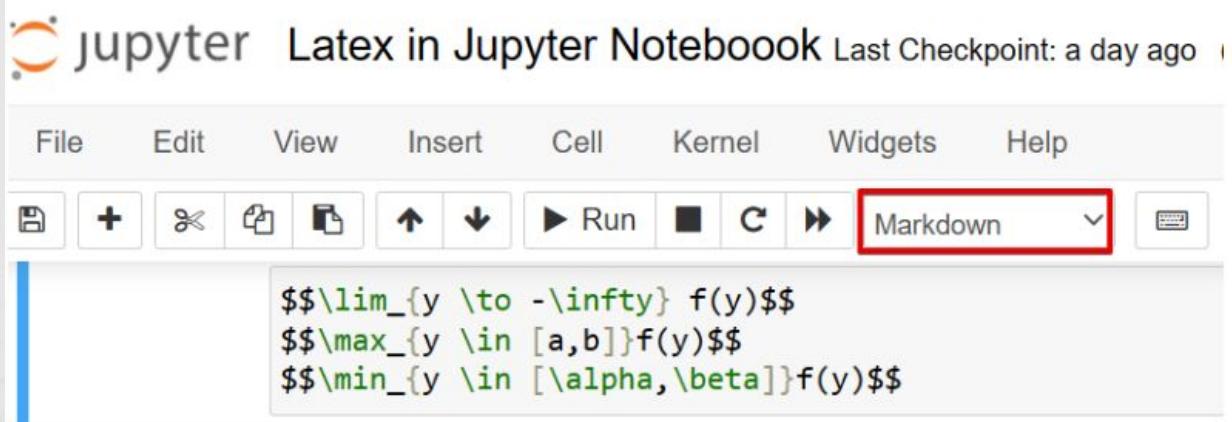


```
$$
\underbrace{\ln \left( \frac{3}{4} \right)}_{\approx -0.1456}
< \overbrace{\exp (3)}^{\approx 6.8390}
$$
```

## 5. Limit Fungsi

Kita bisa menambahkan kode ‘lim\_’ untuk menampilkan limit, ‘max\_’ untuk menampilkan notasi max, dan ‘min\_’ untuk menampilkan notasi min seperti di gambar berikut. Kode ‘infty’ digunakan untuk menampilkan notasi infinity.

$$\begin{aligned} & \lim_{y \rightarrow -\infty} f(y) \\ & \max_{y \in [a,b]} f(y) \\ & \min_{y \in [\alpha,\beta]} f(y) \end{aligned}$$



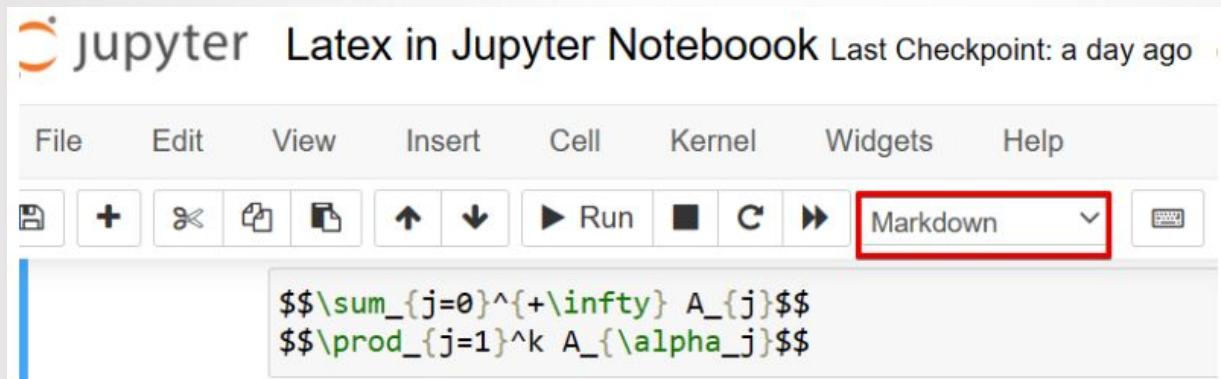
```
$$\lim_{y \rightarrow -\infty} f(y)$$
$$\max_{y \in [a,b]} f(y)$$
$$\min_{y \in [\alpha,\beta]} f(y)$$
```

## 6. Fungsi Sum dan Product

Simbol sigma beserta batasan nominalnya dapat mencoba seperti syntax berikut. Pertama, kita menuliskan ‘sum\_’ yang diikuti batas bawahnya (misalkan  $j=0$ ). Kemudian diikuti batas atasnya(misalkan batas atasnya adalah tak hingga, maka kita menulis keyword ‘ $+\infty$ ’). Selanjutnya, kita menuliskan persamaan sigma. Alur penulisan program juga berlaku untuk product dimana kita menulis keyword ‘prod\_’.

$$\sum_{j=0}^{+\infty} A_j$$

$$\prod_{j=1}^k A_{\alpha_j}$$



## 7. Matriks

Terkadang kita harus menuliskan sebuah matriks di dalam penggerjaan dan ingin menampilkannya dalam file notebook kita seperti gambar di samping? Tentu saja bisa!

Cara penulisannya sebagai berikut:

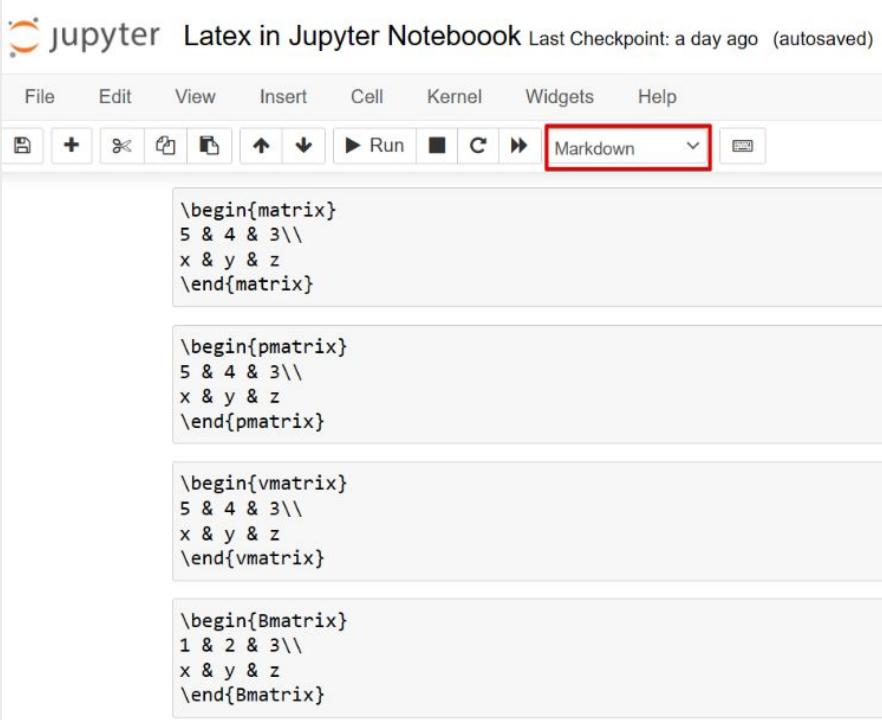
- Untuk memulai matriks, kita harus menuliskan ‘`\begin{matriks}`’
- Tuliskan daftar angka yang ingin dimasukkan ke dalam matriks
- Untuk menuliskan daftar angka secara horizontal, pisahkan dengan keyword ‘`&`’
- Untuk berpindah ke baris selanjutnya, tuliskan terlebih dahulu keyword ‘`\\\`’
- Untuk mengakhiri pembuatan matriks, kita harus menuliskan ‘`\end{matriks}`’

$$\begin{array}{ccc}
 5 & 4 & 3 \\
 x & y & z
 \end{array}$$

$$\left( \begin{array}{ccc}
 5 & 4 & 3 \\
 x & y & z
 \end{array} \right)$$

$$\begin{vmatrix}
 5 & 4 & 3 \\
 x & y & z
 \end{vmatrix}$$

$$\left\{ \begin{array}{ccc}
 1 & 2 & 3 \\
 x & y & z
 \end{array} \right\}$$



The screenshot shows the Jupyter Notebook interface with the title "jupyter Latex in Jupyter Noteboook" and "Last Checkpoint: a day ago (autosaved)". The toolbar has a "Markdown" button highlighted with a red box. Below the toolbar, there are four code snippets demonstrating different matrix types:

```

\begin{matrix}
5 & 4 & 3\\
x & y & z
\end{matrix}

```

```

\begin{pmatrix}
5 & 4 & 3\\
x & y & z
\end{pmatrix}

```

```

\begin{vmatrix}
5 & 4 & 3\\
x & y & z
\end{vmatrix}

```

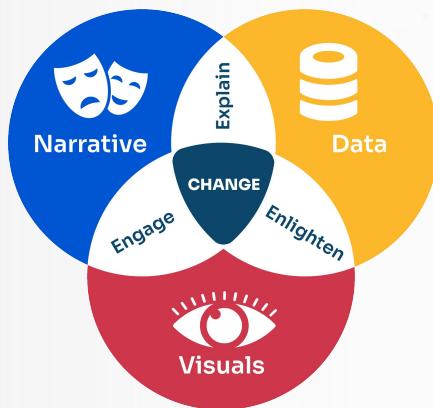
```

\begin{Bmatrix}
1 & 2 & 3\\
x & y & z
\end{Bmatrix}

```

Referensi ini bisa kalian baca lebih lanjut untuk eksplorasi penggunaan latex di python ya! Cek [disini](#)

# Data Storytelling and Communication



## Pengertian Data Storytelling

Data Storytelling melibatkan kemampuan untuk mengomunikasikan hasil data dengan cara yang naratif dan detail melalui penggunaan narasi dan visualisasi. Dalam praktiknya, Data Storytelling digunakan untuk menyajikan kumpulan data dengan konteks yang tepat dan memengaruhi strategi bisnis, sambil menjelaskan informasi kepada pengguna atau pelanggan.

Terdapat tiga komponen kunci dalam Data Storytelling yang perlu diperhatikan, diantaranya:

### Data



Pertama, data menjadi dasar utama dalam melakukan analisis yang akurat dan komprehensif. Dengan menganalisis data menggunakan berbagai metode seperti analisis deskriptif, diagnostik, prediktif, dan preskriptif, kita dapat memperoleh pemahaman yang komprehensif mengenai keseluruhan gambaran yang terkandung dalam data.

### Narasi



Selain itu, narasi juga memainkan peran penting dalam Data Storytelling. Baik melalui lisan maupun tulisan, narasi digunakan untuk mengkomunikasikan wawasan yang dihasilkan dari data, serta memberikan konteks yang melingkupinya. Melalui narasi ini, tujuan kita adalah menginspirasi audiens dengan rekomendasi tindakan yang dapat diambil berdasarkan hasil analisis data yang telah dilakukan.

# Data Storytelling and Communication

## Visualisasi



Selanjutnya, visualisasi juga menjadi elemen krusial dalam Data Storytelling. Representasi visual dari data dan narasi dapat membantu menyampaikan cerita dengan jelas dan mudah diingat oleh audiens. Visualisasi ini dapat berupa berbagai bentuk, seperti bagan, grafik, diagram, gambar, atau video, yang dipilih sesuai dengan konteks dan tujuan komunikasi.

Dalam rangka menguasai keterampilan Data Storytelling, penting bagi kita untuk memahami dan mengintegrasikan ketiga komponen tersebut secara efektif. Dengan demikian, kita dapat mengkomunikasikan hasil analisis data dengan cara yang menarik, mudah dipahami, dan berdampak pada pengambilan keputusan dalam suatu organisasi atau bisnis.

## Mengapa Data Storytelling Penting

Data storytelling menjadi penting karena data memiliki potensi besar untuk mengungkap pola dan peluang di dalam sebuah perusahaan. Ketika data dikumpulkan dan dibagikan, seorang Data Scientist dapat menganalisisnya untuk mendapatkan wawasan yang berguna.



Sebagai contoh, **seorang guru dapat menggunakan data nilai siswa untuk melihat bagaimana kinerja siswa di sekolah**. Dengan melihat data ini, guru dapat mengidentifikasi siswa yang memerlukan bantuan tambahan atau merancang strategi pembelajaran yang lebih efektif.

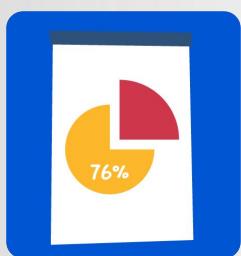
# Data Storytelling and Communication



Dokter juga dapat menggunakan data untuk membuat rekomendasi yang lebih baik bagi pasien. Dengan melihat data kemampuan darah pasien, dokter dapat memberikan saran gaya hidup yang tepat, mendiagnosis penyakit dengan lebih akurat, dan meresepkan pengobatan yang sesuai.



Selain itu, data juga memainkan peran penting dalam pengambilan keputusan bisnis. Perusahaan dapat melihat perubahan pendapatan dari data yang dikumpulkan untuk mengevaluasi sejauh mana strategi bisnis baru mereka berhasil. Data tersebut dapat membantu mereka mengidentifikasi tren pasar, memprediksi permintaan pelanggan, dan menyesuaikan strategi bisnis mereka secara lebih efektif.



Jika seorang Data Scientist memberitahu pernyataan sederhana "**76% adalah yang penting**", Tidak banyak orang tahu apakah itu berarti 76% adalah sebagian kecil dari hari seseorang yang dihabiskan untuk bekerja, berkumpul bersama keluarga, ataupun untuk bermain dengan teman.

Angka dan data perlu dimasukkan ke dalam konteks agar kita dapat memahaminya. Namun, hanya memiliki data saja tidak cukup. Penting bagi seorang Data Storyteller untuk memahami dan menggunakan data secara efektif. Data storytelling melibatkan kemampuan untuk mengubah data menjadi narasi yang dapat mempengaruhi cara orang memahami dan memandang suatu topik. Seorang Data Storyteller memiliki tanggung jawab untuk mengomunikasikan temuan dan wawasan dari data dengan cara yang jelas, menarik, dan mudah dipahami oleh audiensnya. Dengan menguasai keterampilan data storytelling, seorang Data Storyteller dapat memainkan peran penting dalam membantu organisasi dan individu mengambil keputusan yang lebih baik berdasarkan analisis data yang akurat dan relevan.

# Data Storytelling and Communication

## Apa yang membuat Data Storytelling menarik?

Mari kita melihat bagaimana Storytelling dapat membantu orang membuat keputusan yang lebih baik. Untuk itu, penting untuk memahami faktor-faktor yang membuat Data Storytelling menjadi efektif.

1. Pertama, **kontennya harus relevan dengan audiens yang dituju**. Hal ini berarti konten harus sesuai dengan tingkat pengetahuan audiens saat ini dan membantu mereka mencapai tujuan tertentu. Terlebih lagi, audiens dapat menjadi internal, seperti saat melakukan presentasi kepada pimpinan tentang perlunya berinvestasi dalam strategi atau taktik tertentu, atau eksternal, misalnya dalam kampanye untuk meyakinkan pelanggan agar membeli produk. Dalam hal ini, penting untuk memikirkan apa yang penting dan dibutuhkan oleh pelanggan tersebut.
2. Selanjutnya, **data yang disajikan dalam Data Storytelling harus baik**. Artinya, data harus berasal dari sumber yang memiliki reputasi baik dan dikumpulkan dengan metode yang benar sehingga mewakili apa yang diperlukan oleh pengguna. Data yang tersedia untuk publik dari entitas pemerintah, organisasi antar pemerintah, peneliti akademik, dan pemimpin analisis yang mapan tidak hanya mudah diakses, tetapi juga transparan dan dapat diverifikasi.
3. Selain itu, penting untuk memiliki **narasi yang jelas dalam Data Storytelling**. Dalam konteks ini, kita sudah terbiasa dengan struktur cerita tradisional yang terdiri dari awal, tengah, dan akhir. Dalam Data Storytelling, hal ini sering kali berarti memperkenalkan topik sebelum memahami data yang disajikan. Data Storytelling berbeda dari laporan yang langsung menyajikan fakta-fakta. Jika audiens tidak memiliki pengetahuan mendalam tentang subjek tersebut, penting untuk menggunakan bahasa yang sederhana agar tidak kehilangan mereka dengan jargon atau akronim yang membingungkan.

# Data Storytelling and Communication

## Langkah-langkah membuat Data Storytelling yang tepat

Sekarang, mari kita menjelajahi contoh sederhana yang akan memandu Anda melalui proses Data Storytelling yang luar biasa, yang akan memberi Anda gambaran tentang kehebatannya dari awal hingga akhir.



Mulailah dengan **mendengarkan audiens Anda**. Anda perlu **mengidentifikasi audiens** Anda dan berkomunikasi dengan mereka untuk memahami kepentingan, tujuan, pengetahuan saat ini, dan keputusan yang perlu mereka buat. Penelitian tambahan juga dapat dilakukan untuk mendapatkan pemahaman yang lebih dalam tentang audiens.



**Tentukan data yang penting.** Dengan memahami audiens Anda, Anda dapat menentukan **jenis data yang relevan** dan perlu disertakan dalam cerita. Data tersebut bisa berupa data kuantitatif seperti pendapatan, perubahan dari waktu ke waktu, atau jumlah yang terpengaruh, atau data kualitatif seperti proses, sistem, atau kutipan.

Line Chart



Misalnya, template **bagan garis sederhana** ini mudah diedit untuk menampilkan perubahan dari waktu ke waktu.

# Data Storytelling and Communication



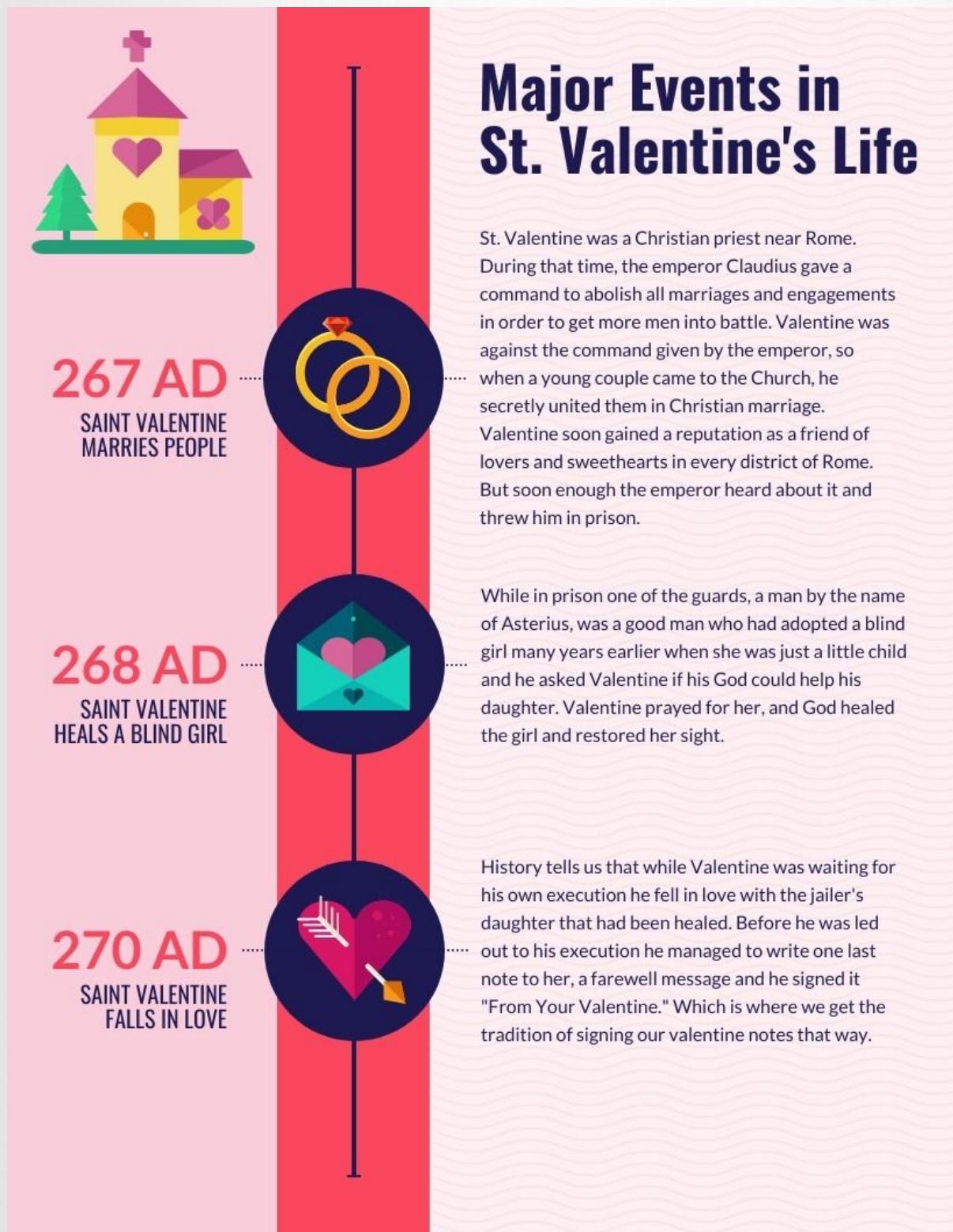
**Buat outline alur cerita.** Setelah Anda memiliki data yang diperlukan, buatlah rangkuman atau sketsa alur cerita yang akan diikuti. Anda dapat menjelajahi berbagai kemungkinan, misalnya menyoroti tiga poin penting atau mengeksplorasi tiga bagian cerita yang relevan. Dalam hal ini, Anda dapat mempertimbangkan alur cerita yang alami berdasarkan waktu atau peristiwa yang terjadi.



**Buat rancangan desain.** Dengan alur cerita yang telah ditentukan, pikirkanlah tata letak atau komposisi desain yang paling sesuai. Anda dapat membuat sketsa tangan atau menggunakan alat desain untuk menghasilkan beberapa tata letak dan komposisi yang berbeda. Ini membantu Anda memvisualisasikan ide-ide yang mungkin berhasil secara visual sebelum memulai pembuatan presentasi atau visualisasi data yang final. Contoh: kita akan melihat-lihat banyak template visualisasi data untuk mendapatkan beberapa ide tentang apa yang mungkin berhasil. kita tahu kita ingin mencari beberapa yang akan membantu kita menunjukkan perubahan dari waktu ke waktu, Seperti contoh pada halaman selanjutnya.

Dengan mengikuti langkah-langkah ini, Anda dapat membuat Data Storytelling yang efektif dan menarik, dengan fokus pada audiens yang dituju dan penggunaan data yang relevan untuk mengkomunikasikan pesan dengan jelas dan mempengaruhi pengambilan keputusan yang lebih baik.

# Data Storytelling and Communication



# Data Storytelling and Communication

## Perbedaan Data Storytelling dan Data Visualisasi

Dalam materi pembelajaran ini, kita akan mempelajari dunia data yang menawan dan memeriksa perbedaan utama antara penceritaan data dan visualisasi data. Kedua pendekatan memainkan peran penting dalam mengkomunikasikan insight (wawasan) yang diperoleh dari data secara efektif, tetapi keduanya berbeda dalam teknik dan tujuannya. Dengan memahami perbedaan ini, Anda akan memperoleh keterampilan untuk memanfaatkan kekuatan data dan membuat narasi menarik yang beresonansi dengan audiens Anda.

- **Data storytelling merupakan sebuah strategi** yang digunakan untuk mengkomunikasikan informasi dengan cara yang dapat dipahami oleh audiens. Di sisi lain, **data visualization berperan sebagai taktik** untuk mengemas penceritaan data. Dalam konteks ini, angka-angka yang dihasilkan oleh data akan digambarkan secara visual, seolah-olah kita sedang melukis gambar. Melalui visualisasi tersebut, cerita yang terkandung dalam data akan menjadi hidup dan memungkinkan audiens untuk memahami tujuan komunikasi serta membuat keputusan berdasarkan narasi yang disampaikan.
- **Data storytelling dapat dianggap sebagai jawaban atas pertanyaan "mengapa"** suatu fenomena terjadi, sementara **data visualization bertujuan untuk menunjukkan "apa"** yang terjadi secara konkret. Untuk memberikan gambaran yang lebih jelas, kedua aspek tersebut dapat dibandingkan dengan koin. Meskipun keduanya memiliki sisi yang berbeda, namun keduanya saling melengkapi. Dalam konteks sebab dan akibat, visualisasi data akan menyoroti peristiwa yang sedang terjadi secara visual, sementara penceritaan data akan merangkum berbagai faktor yang menjadi penyebab munculnya pola atau sebaran data tersebut.

# Data Storytelling and Communication

- **Data storytelling mengambil bentuk gambar** yang memvisualisasikan hasil analisis data, sementara **data visualization menggunakan berbagai model grafik**. Ketika menghadirkan hasil analisis data, penting untuk mempersiapkan grafik dengan teliti dan akurat. Namun, hanya memiliki grafik tersebut tidak cukup untuk memastikan bahwa audiens dapat memahaminya dengan cepat. Untuk itu, diperlukan narasi yang menjelaskan makna dari grafik yang ditampilkan. Oleh karena itu, perlu juga menyiapkan fakta dan informasi terkait grafik secara komprehensif untuk memastikan pesan yang ingin disampaikan melalui grafik dapat dipahami dengan baik oleh audiens.



## EXERCISE

Berikut adalah langkah - langkah untuk mengerjakan *exercise*:

1. Buka *link* Google Colab menggunakan Google Chrome.
2. Klik ‘File’.
3. Klik ‘Save a Copy in Drive’.
4. Kerjakan *exercise* dengan mengisi coding pada baris yang tertera dan menjawab Insight yang ditanyakan.
5. Untuk baris coding yang sudah terisi, anda disarankan untuk langsung melakukan run baris tersebut tanpa mengubah isinya.
6. Perhatikan instruksi lain yang tertera pada Google Colab.

Google Colab dapat diakses pada:

[Exercise Modul 3 DS](#)

Nb:

Pada exercise modul 3 ini, terdapat kunci jawaban exercise modul 2.

---



**THANK YOU**