BAB 3

Menelaah Data

TEORI PENDUKUNG

A. Pengantar

Tugas dari data scientist merupakan permasalahan yang sudah ada sejak lama, salah satunya membuat prediksi keluaran yang bersifat biner. Istilah umum yang kita pakai, apakah terjadi atau tidak terjadi. Misalnya, kita meramalkan menang atau tidak menang, lulus ujian atau tidak lulus ujian, diterima atau tidak diterima, dan lain sebagainya. Aplikasi bisnis yang umum adalah deteksi churn atau retensi pelanggan. Kasus penggunaan populer lainnya adalah, tingkat kematian atau analisis kelangsungan hidup. Peristiwa yang bersifat biner menciptakan dinamika yang menarik, karena kita tahu secara statistik, tebakan acak harus mencapai tingkat akurasi 50%, tanpa membuat satu pun algoritma atau menulis satu baris kode pun.

Gambar 1. Prediksi bagian atas koin saat dilempar, nilai probabilitas 50% benar, dadu?

Jika kita memiliki data lemparan koin sebanyak 10 ribu kali lemparan, ternyata koin bagian gambar angklung muncul sebanyak 6321 kali, maka pada 10 kali lemparan berikutnya, kita dapat prediksi angklung akan muncul 6 kali. Secara garis besar, ini yang kita lakukan dengan metode statistika sederhana. Pada kasus yang lebih kompleks adalah prediksi kemunculan dadu. Ada berapa kemungkinan angka yang muncul? Jika kita memprediksinya, berapa probabilitas kemungkinan prediksi kita benar.

B. Framework pada Data Science

Mendefinisikan Permasalahan

Jika ilmu data, big data, pembelajaran mesin (Machine Learning), analisis prediktif, business intelligence, atau kata kunci lainnya adalah solusinya, lalu apa masalahnya? Permasalahan harus diketahui dahulu sebelum mencari solusinya. Seringkali yang dilakukan adalah terburu-buru menggunakan teknologi, alat, atau algoritma tertentu yang canggih sebelum mengetahui permasalahan sebenarnya yang ingin dipecahkan. Pada CRISP-DM, tahapan ini dikenal dengan Business Understanding. Kasus pada dataset Titanic secara umum kita diminta memprediksi siapa saja yang selamat dan tidak selamat. Kasus lain apa yang dapat kita pecahkan dari data yang sama?. Prediksi jenis kelamin penumpang tersebut (jika nama tidak diketahui), Prediksi usia penumpang tersebut, prediksi dari pelabuhan mana dia berangkat, dan lain sebagainya.

Mengumpulkan Data

John Naisbitt menulis dalam bukunya Megatrends yang terbit pada tahun 1984, kita "drowning in data, yet starving for knowledge (tenggelam dalam data, namun berjuang untuk mendapatkan pengetahuan)." Jadi, kemungkinan besar, kumpulan data tersebut sudah ada di suatu tempat, dalam beberapa format. Bisa eksternal atau internal, terstruktur atau tidak terstruktur, statis atau stream (realtime), objektif atau subjektif, dst. Seperti kata pepatah, kita tidak perlu menciptakan kembali roda, kita hanya perlu tahu di mana menemukannya. Pada langkah berikutnya, kita akan memikirkan tentang mengubah "data kotor" menjadi "data bersih".

Menyiapkan Data untuk Diolah

Langkah ini sering disebut sebagai pengolahan data, proses yang diperlukan untuk mengubah data "liar" menjadi data yang "dapat dikendalikan". Pengolahan data meliputi penerapan arsitektur data untuk penyimpanan dan pemrosesan, pengembangan standar tata kelola data untuk kualitas dan kontrol, ekstraksi data (misalnya dengan menggunakan ETL dan web scraping), dan pembersihan data untuk mengidentifikasi data yang tidak normal, hilang, atau outlier.

Melakukan Analisa Exploratory

Langkah penting berikutnya adalah mekalukan ekplorasi data. Langkah ini dilakukan untuk menggunakan statistik deskriptif dan grafis guna mencari potensi masalah, pola, klasifikasi, korelasi, dan perbandingan dalam kumpulan data. Selain itu, kategorisasi data (yaitu

kualitatif vs kuantitatif) juga penting untuk memahami dan memilih uji hipotesis atau model data yang tepat. Pada Modul ini, tahapan yang kita lakukan berhenti disini.

Memodelkan Data

Seperti statistik deskriptif dan inferensial, pemodelan data dapat meringkas data atau memprediksi hasil di masa depan. Kumpulan data dan hasil yang diharapkan akan menentukan algoritme yang tersedia untuk digunakan. Penting juga untuk diingat, algoritme adalah alat dan bukan tongkat ajaib atau peluru ajaib. Kita harus tetap menjadi ahli yang tahu cara memilih alat yang tepat untuk pekerjaan tersebut. Analoginya seperti meminta seseorang memberi kita obeng Plus (+), dan mereka memberi Anda obeng Minus (-) atau palu. Hal yang sama berlaku dalam pemodelan data. Model yang salah dapat menyebabkan kinerja yang buruk dan prediksi yang salah.

Memvalidasi dan dan Implementasi Data Model

Setelah kita melatih model berdasarkan sebagian data, langkah berikutnya yang dilakukan adalah menguji model. Hal ii membantu memastikan kita tidak melakukan overfitting model atau membuatnya terlalu spesifik pada sebagian yang dipilih, sehingga tidak secara akurat melakukan overfitting pada sebagian lain dari kumpulan data yang sama. Pada langkah ini, ktia menentukan apakah model melakukan overfitting, generalisasi, atau underfit pada kumpulan data. Overfitting adalah proses mempelajari data dengan model yang terlalu sempurna, padahal ada kemungkinan data yang digunakan bukan data yang baik.

Optimasi dan Memilih Strategi

Tahapan ini adalah langkah perbaikan di mana kita mengulang kembali proses untuk membuatnya lebih baik, lebih kuat, dan lebih cepat dari sebelumnya. Sebagai data scientist, strategi kita haruslah mengoptimalkan hasil yang telah didapatkan, mencoba beberapa model baru, mempelajari waktu operasi, dan memilih model terbaik yang dapat digunakan pada proses deployment aplikasi.

PRAKTIKUM

Mendefinisikan permasalahan

Tahap 1: Mendefinisikan Permasalahan

Pada dataset Titanic, jelas sekali kita diminta untuk memprediksi apakah penumpang selamat atau tidak dari tenggelamnya kapal Titanic. Mau tidak mau, kita harus membuat algoritma untuk memprediksi kemungkinan selamatnya penumpang Titanic. Pada praktikum kali ini kita hanya akan sampai memvisualisasikan data saja.

Tahap 2: Mengumpulkan Data

Data yang dipakai adalah data dari Kaggle pada link berikut:
https://www.kaggle.com/c/titanic/data
pada library seaborn juga telah disediakan dataset Titanic, untuk praktikum ini kita akan coba menggunakan dataset tersebut.

```
import seaborn as sns
df = sns.load_dataset('titanic')
df.head()
```

	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
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	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

```
import seaborn as sns

# Load dataset Titanic dari Seaborn
df = sns.load_dataset('titanic')

# Tampilkan 5 data pertama
df.head()
```

	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
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	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

Tahap 3: Menyiapkan Data untuk Diolah

Kita coba melihat deskripsi dari data tersebut. Menggunakan describe(). Berapa usia rata-rata penumpang Titanic?

```
df.info()
df.describe()
```

```
<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
```

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


```
df.info()
df.describe()
```

```
<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
```

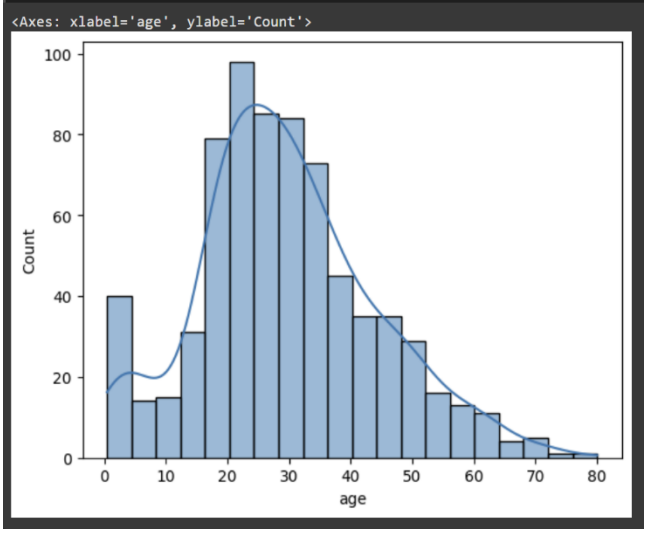
	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000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

Tahap 4: Melakukan Analisa Exploratory

Berikut contoh sederhana untuk memvisualisasikan data usia penumpang Titanic. Pada code ini juga kita terlihat menggunakan Analisa eksploratori sederhana seperti membuat kategorisasi pada usia penumpang.

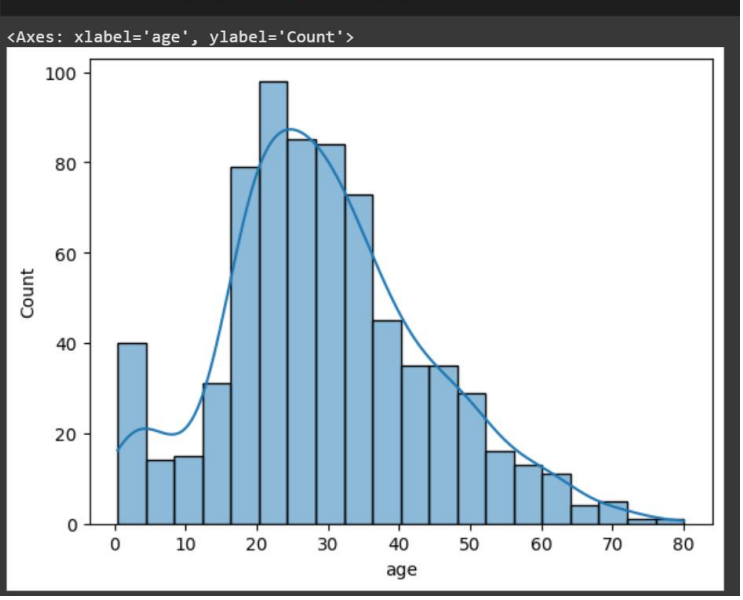
```
sns.histplot(df['age'].dropna(), kde=True)
```

```
<Axes: xlabel='age', ylabel='Count'>
```




```
# Visualisasi distribusi usia penumpang Titanic
sns.histplot(df['age'].dropna(), kde=True)
```

```
<Axes: xlabel='age', ylabel='Count'>
```



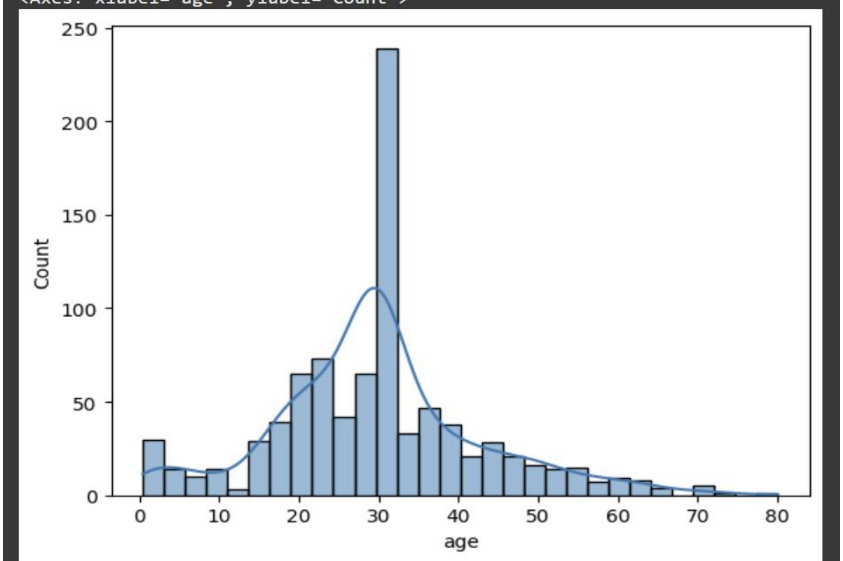
Pada atribut usia, kita dapat melihat ada beberapa data penumpang yang tidak ada data usianya. Karena usia merupakan nilai numerik dan terdistribusi normal, maka kita dapat isikan data kosong menggunakan Mean. Berikut contoh kodenya.

```
df['age'].fillna(df['age'].mean(), inplace=True)
```

```
<ipython-input-17-15e9ed7dc05b>:1: FutureWarning: A value is trying to be se
The behavior will change in pandas 3.0. This inplace method will never work

For example, when doing 'df[col].method(value, inplace=True)', try using 'df


  df['age'].fillna(df['age'].mean(), inplace=True)
```

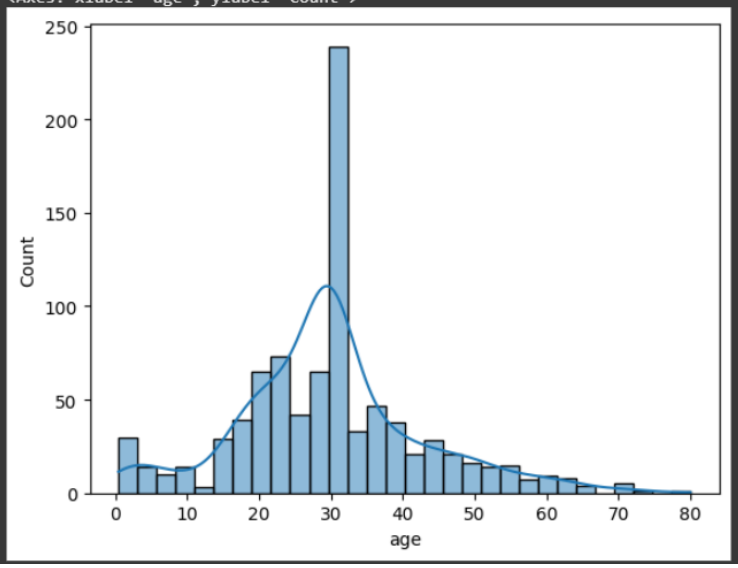
```
sns.histplot(df['age'].dropna(), kde=True)
```

```
<Axes: xlabel='age', ylabel='Count'>
```



```
df['age'].fillna(df['age'].mean(), inplace=True)

sns.histplot(df['age'].dropna(), kde=True)
```

```
<ipython-input-20-7fed588bd272>:9: FutureWarning: A value is trying to be set on a copy of a DataF
The behavior will change in pandas 3.0. This inplace method will never work because the intermedia

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value},

  df['age'].fillna(df['age'].mean(), inplace=True)
<Axes: xlabel='age', ylabel='Count'>
```

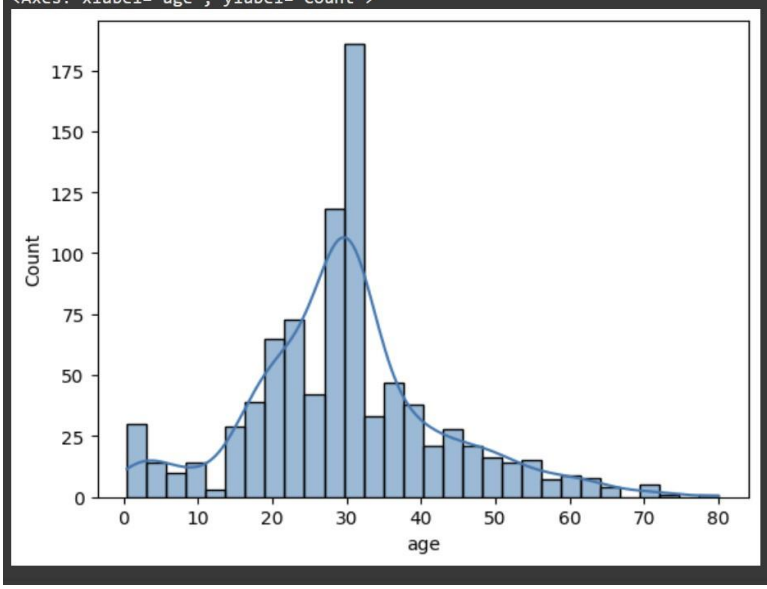


Ada cukup perbedaan distribusi data usia yang cukup signifikan. Kali ini kita coba untuk mengisikan dengan mean tetapi kita kelompokkan untuk tiap jenis kelamin.

```
df['age'] = df.groupby('sex')['age'].transform(lambda x: x.fillna(x.mean()))
```

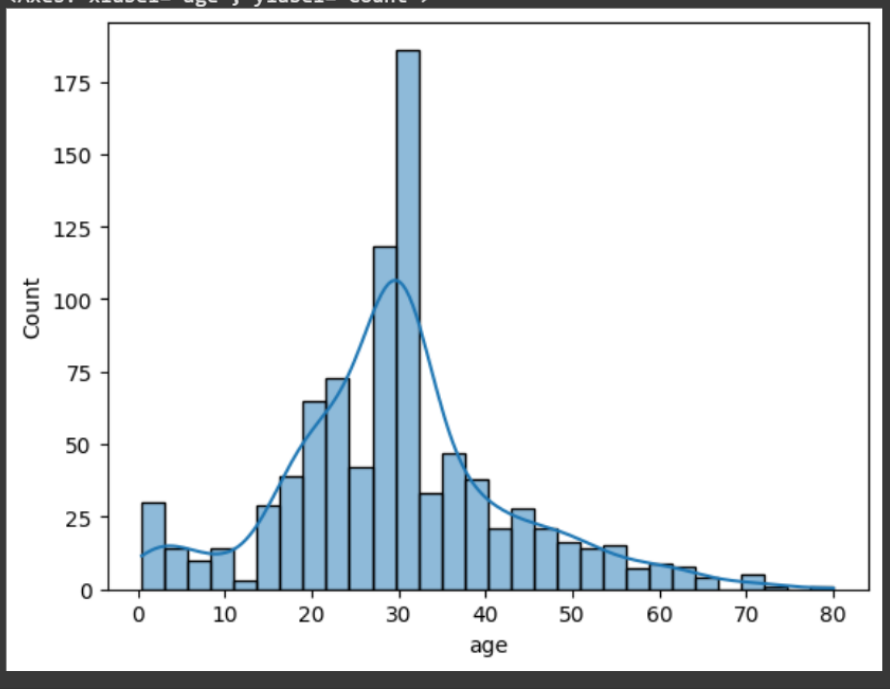
```
sns.histplot(df['age'].dropna(), kde=True)
```

```
<Axes: xlabel='age', ylabel='Count'>
```



```
# Visualisasi distribusi usia penumpang Titanic
df['age'] = df.groupby('sex')['age'].transform(lambda x: x.fillna(x.mean()))
sns.histplot(df['age'].dropna(), kde=True)
```

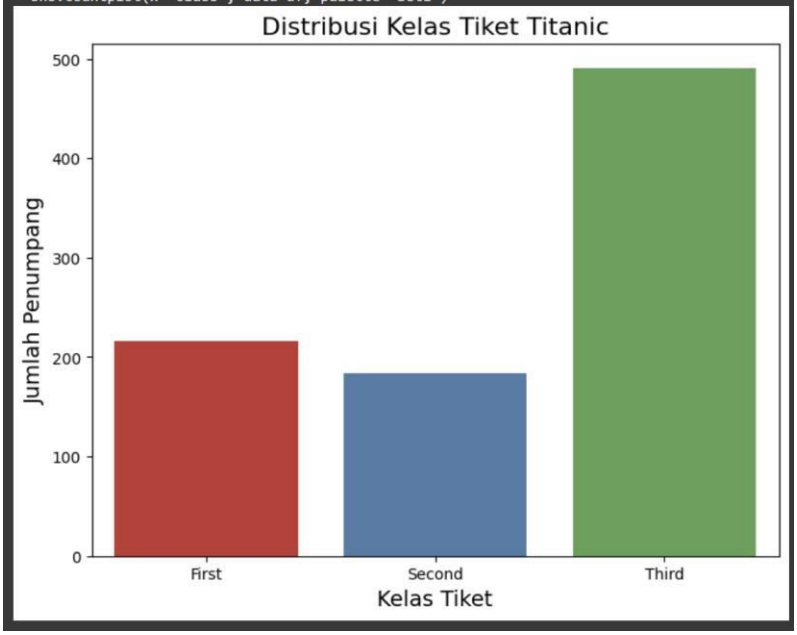
```
<Axes: xlabel='age', ylabel='Count'>
```



Berikutnya kita coba dapat melihat distribusi kelas penumpang dan menampilkannya dalam grafik bar.

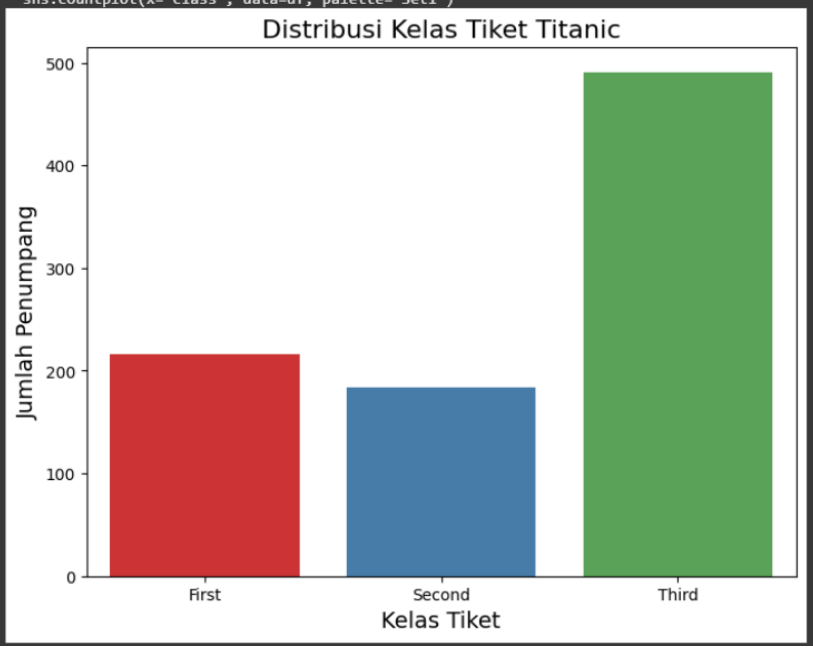
```
plt.figure(figsize=(8, 6))
sns.countplot(x='class', data=df, palette='Set1')
plt.title('Distribusi Kelas Tiket Titanic', fontsize=16)
plt.xlabel('Kelas Tiket', fontsize=14)
plt.ylabel('Jumlah Penumpang', fontsize=14)
plt.show()
```

```
<ipython-input-14-11370a7c013a>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assi

  sns.countplot(x='class', data=df, palette='Set1')
```



```python
# Visualisasi distribusi usia penumpang Titanic
plt.figure(figsize=(8, 6))
sns.countplot(x='class', data=df, palette='Set1')
plt.title('Distribusi Kelas Tiket Titanic', fontsize=16)
plt.xlabel('Kelas Tiket', fontsize=14)
plt.ylabel('Jumlah Penumpang', fontsize=14)
plt.show()
```

```
<ipython-input-13-b90ebdcfed18>:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x`

  sns.countplot(x='class', data=df, palette='Set1')
```



Kode berikut digunakan untuk menampilkan jumlah class dalam bentuk teks.

```python
# Hitung jumlah penumpang di setiap kelas
class_counts = df['class'].value_counts()

# Tampilkan hasil dalam bentuk teks
print("Jumlah Penumpang di Setiap Kelas Tiket:")
print(class_counts)
```

```
Jumlah Penumpang di Setiap Kelas Tiket:
class
Third     491
First     216
Second    184
Name: count, dtype: int64
```


```python
# Hitung jumlah penumpang di setiap kelas
class_counts = df['class'].value_counts()

# Tampilkan hasil dalam bentuk teks
print("Jumlah Penumpang di Setiap Kelas Tiket:")
print(class_counts)
```

```
Jumlah Penumpang di Setiap Kelas Tiket:
class
Third     491
First     216
Second    184
Name: count, dtype: int64
```


Tugas Praktikum

1. Jika kita mencoba melihat pada library seaborn, telah disediakan beberapa dataset yang dapat kita gunakan secara langsung. Perhatikan kode berikut:

```python
import seaborn as sns
print(sns.get_dataset_names())
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds', 'dot
```


```python
[50] import seaborn as sns

     print(sns.get_dataset_names())
```

```
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds', 'dots', 'dowjones', 'exercise',
```


2. Pilih 1 dataset yang ada pada seaborn, coba cari informasi data tersebut terkait apa, olah secara sederhana, isikan data kosong, dan tampilkan juga grafik-grafik visualisasi sederhananya.

```python
import seaborn as sns

# Load dataset taxis dari Seaborn
df = sns.load_dataset('taxis')

# Tampilkan 5 data pertama
print(df.head())
```

```
              pickup             dropoff  passengers  distance  fare   tip  \
0 2019-03-23 20:21:09 2019-03-23 20:27:24           1      1.60   7.0  2.15
1 2019-03-04 16:11:55 2019-03-04 16:19:00           1      0.79   5.0  0.00
2 2019-03-27 17:53:01 2019-03-27 18:00:25           1      1.37   7.5  2.36
3 2019-03-10 01:23:59 2019-03-10 01:49:51           1      7.70  27.0  6.15
4 2019-03-30 13:27:42 2019-03-30 13:37:14           3      2.16   9.0  1.10

   tolls  total   color      payment            pickup_zone  \
0    0.0  12.95  yellow  credit card        Lenox Hill West
1    0.0   9.30  yellow         cash  Upper West Side South
2    0.0  14.16  yellow  credit card          Alphabet City
3    0.0  36.95  yellow  credit card              Hudson Sq
4    0.0  13.40  yellow  credit card           Midtown East

           dropoff_zone pickup_borough dropoff_borough
0    UN/Turtle Bay South      Manhattan       Manhattan
1  Upper West Side South      Manhattan       Manhattan
2           West Village      Manhattan       Manhattan
3         Yorkville West      Manhattan       Manhattan
4         Yorkville West      Manhattan       Manhattan
```

```python
print(df.info())
print(df.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6433 entries, 0 to 6432
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   pickup           6433 non-null   datetime64[ns]
 1   dropoff          6433 non-null   datetime64[ns]
 2   passengers       6433 non-null   int64
 3   distance         6433 non-null   float64
 4   fare             6433 non-null   float64
 5   tip              6433 non-null   float64
 6   tolls            6433 non-null   float64
 7   total            6433 non-null   float64
 8   color            6433 non-null   object
 9   payment          6389 non-null   object
 10  pickup_zone      6407 non-null   object
 11  dropoff_zone     6388 non-null   object
 12  pickup_borough   6407 non-null   object
 13  dropoff_borough  6388 non-null   object
dtypes: datetime64[ns](2), float64(5), int64(1), object(6)
memory usage: 703.7+ KB
None
                             pickup                         dropoff  \
count                          6433                            6433
mean   2019-03-16 08:31:28.514223616  2019-03-16 08:45:49.491217408
min              2019-02-28 23:29:03           2019-02-28 23:32:35
25%              2019-03-08 15:50:34           2019-03-08 16:12:51
50%              2019-03-15 21:46:58           2019-03-15 22:06:44
75%              2019-03-23 17:41:38           2019-03-23 17:51:56
max              2019-03-31 23:43:45           2019-04-01 00:13:58
std                              NaN                             NaN

        passengers     distance         fare          tip        tolls  \
count  6433.000000  6433.000000  6433.000000  6433.00000  6433.000000
mean      1.539251     3.024617    13.091073     1.97922     0.325273
min       0.000000     0.000000     1.000000     0.00000     0.000000
25%       1.000000     0.980000     6.500000     0.00000     0.000000
50%       1.000000     1.640000     9.500000     1.70000     0.000000
75%       2.000000     3.210000    15.000000     2.80000     0.000000
max       6.000000    36.700000   150.000000    33.20000    24.020000
std       1.203768     3.827867    11.551804     2.44856     1.415267

             total
count  6433.000000
mean     18.517794
min       1.300000
25%      10.800000
50%      14.160000
75%      20.300000
max     174.820000
std      13.815570
```

Mengisikan data kosong

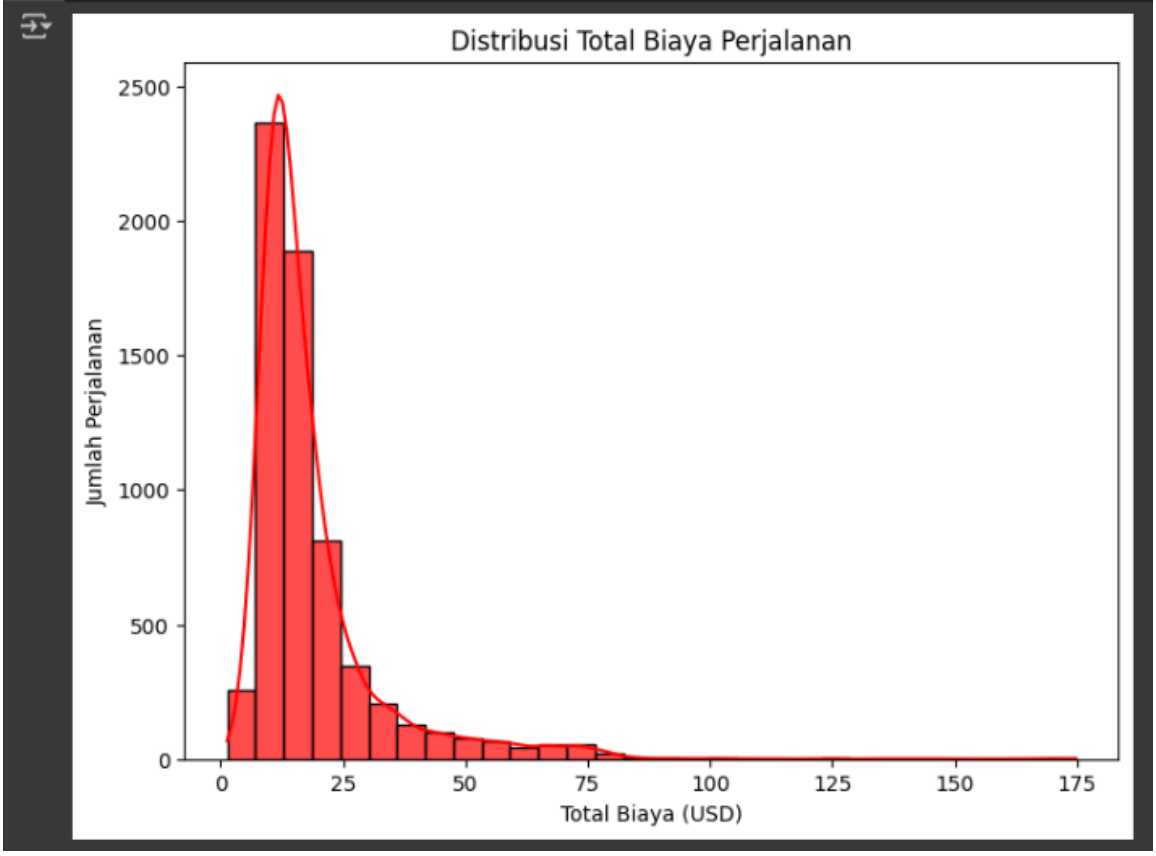
Untuk melakukan pengisisan data kosong, perlu pengecekan terlebih dahulu. Karena data set taxis tidak ada data kosong maka tidak memerlukan pengisian

```
# Mengecek apakah ada data kosong
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6433 entries, 0 to 6432
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype         
---  ------           --------------  -----         
 0   pickup           6433 non-null   datetime64[ns]
 1   dropoff          6433 non-null   datetime64[ns]
 2   passengers       6433 non-null   int64         
 3   distance         6433 non-null   float64       
 4   fare             6433 non-null   float64       
 5   tip              6433 non-null   float64       
 6   tolls            6433 non-null   float64       
 7   total            6433 non-null   float64       
 8   color            6433 non-null   object        
 9   payment          6389 non-null   object        
 10  pickup_zone      6407 non-null   object        
 11  dropoff_zone     6388 non-null   object        
 12  pickup_borough   6407 non-null   object        
 13  dropoff_borough  6388 non-null   object        
dtypes: datetime64[ns](2), float64(5), int64(1), object(6)
memory usage: 703.7+ KB
None
```

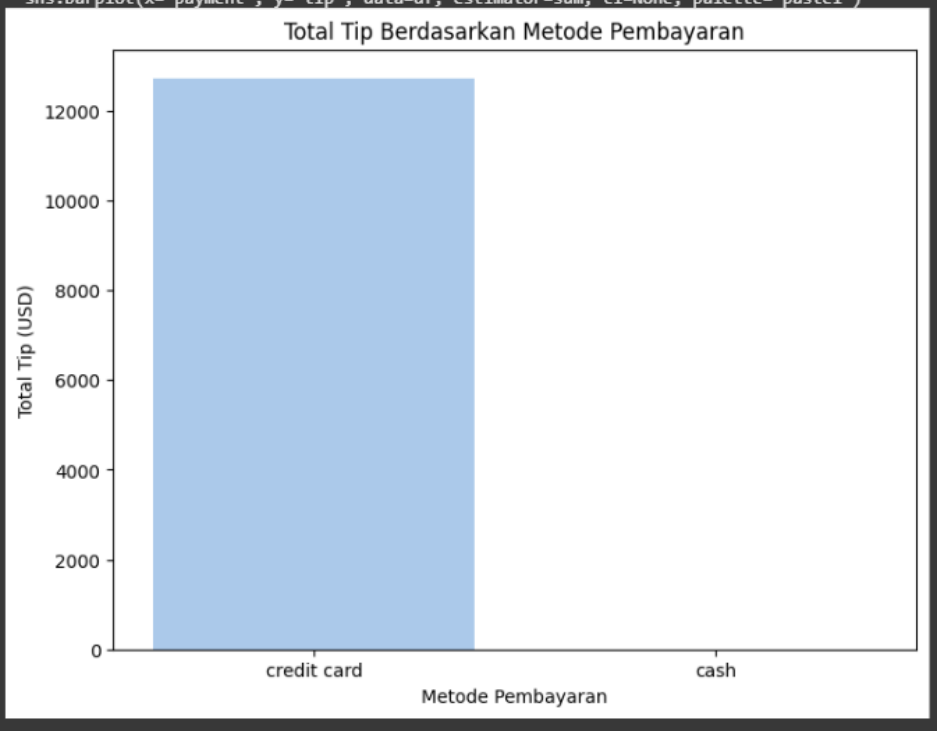
Menampilkan grafik-grafik visualisasi

```
plt.figure(figsize=(8,6))
sns.histplot(df['total'], kde=True, bins=30, color="red", alpha=0.7)
plt.xlabel("Total Biaya (USD)")
plt.ylabel("Jumlah Perjalanan")
plt.title("Distribusi Total Biaya Perjalanan")
plt.show()
```



```
plt.xlabel("Metode Pembayaran")
plt.ylabel("Total Tip (USD)")
plt.title("Total Tip Berdasarkan Metode Pembayaran")

plt.show()
```

```
<ipython-input-88-da545cbb3e1f>:8: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.barplot(x="payment", y="tip", data=df, estimator=sum, ci=None, palette="pastel")
<ipython-input-88-da545cbb3e1f>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x`

  sns.barplot(x="payment", y="tip", data=df, estimator=sum, ci=None, palette="pastel")
```

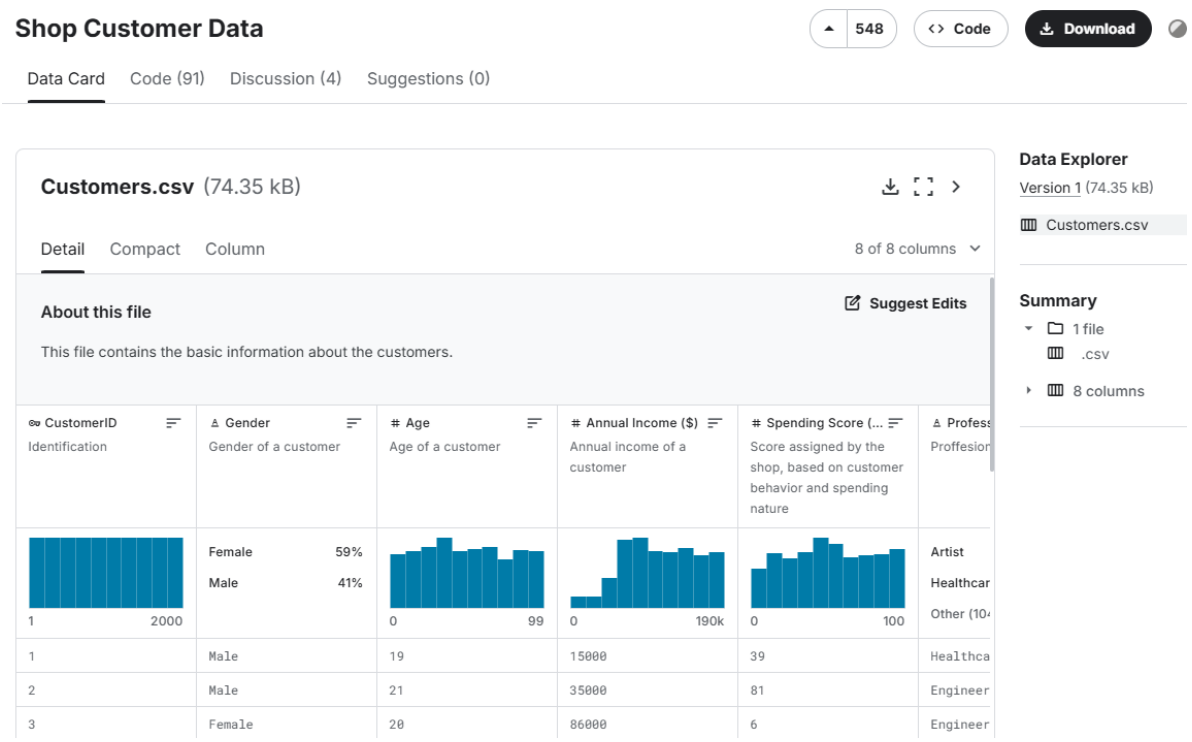



```
# Memfilter data hanya untuk perjalanan dengan tip == 0
not_tipped_taxis = taxis[taxis["tip"] == 0]

# Menghitung jumlah perjalanan yang tidak memberikan tip berdasarkan jumlah penumpang
passenger_not_tipped_counts = not_tipped_taxis["passengers"].value_counts().sort_index()

# Menampilkan hasil dalam bentuk teks
print("Jumlah perjalanan yang tidak memberikan tip berdasarkan jumlah penumpang:")
print(passenger_not_tipped_counts)
```

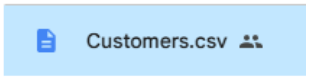
```
Jumlah perjalanan yang tidak memberikan tip berdasarkan jumlah penumpang:
passengers
0      22
1    1734
2     302
3      87
4      32
5      88
6      46
Name: count, dtype: int64
```

3. Gunakan dataset sekunder dari Kaggle atau UCI Machine Learning Repository. Pilih satu dataset dilink tersebut, olah dan visualisasikan sesuai dengan yang anda coba pada soal nomor 2.

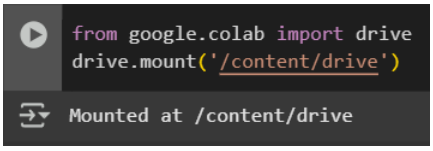
1. Saya memilih data set **“Shop Customer Data”**



2. Mengaupload ke google drive

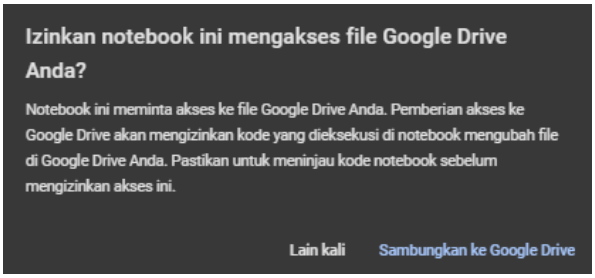


3. Menyambungkan Google Drive ke Google Colab

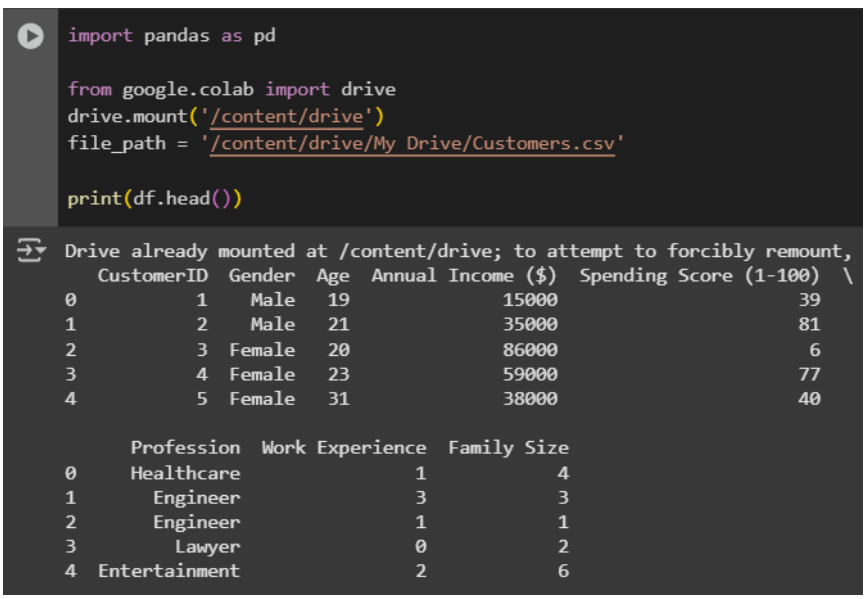


```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```



4. Menampikan 5 data pertama pada data set Customers.csv



```
import pandas as pd

from google.colab import drive
drive.mount('/content/drive')
file_path = '/content/drive/My Drive/Customers.csv'

print(df.head())
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount,
   CustomerID  Gender  Age  Annual Income ($)  Spending Score (1-100)  \
0           1    Male   19              15000                      39
1           2    Male   21              35000                      81
2           3  Female   20              86000                       6
3           4  Female   23              59000                      77
4           5  Female   31              38000                      40

      Profession  Work Experience  Family Size
0     Healthcare                1            4
1       Engineer                3            3
2       Engineer                1            1
3         Lawyer                0            2
4  Entertainment                2            6
```

5. Menampilkan deskripsi data

```
print(df.info())
print(df.describe())
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call d
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              2000 non-null   int64
 1   Gender                  2000 non-null   object
 2   Age                     2000 non-null   int64
 3   Annual Income ($)       2000 non-null   int64
 4   Spending Score (1-100)  2000 non-null   int64
 5   Profession              1965 non-null   object
 6   Work Experience         2000 non-null   int64
 7   Family Size             2000 non-null   int64
dtypes: int64(6), object(2)
memory usage: 125.1+ KB
None
         CustomerID          Age  Annual Income ($)  Spending Score (1-100)  \
count   2000.000000  2000.000000        2000.000000             2000.000000
mean    1000.500000    48.960000      110731.821500               50.962500
std      577.494589    28.429747       45739.536688               27.934661
min        1.000000     0.000000           0.000000                0.000000
25%      500.750000    25.000000       74572.000000               28.000000
50%     1000.500000    48.000000      110045.000000               50.000000
75%     1500.250000    73.000000      149092.750000               75.000000
max     2000.000000    99.000000      189974.000000              100.000000

       Work Experience  Family Size
count      2000.000000  2000.000000
mean          4.102500     3.768500
std           3.922204     1.970749
min           0.000000     1.000000
25%           1.000000     2.000000
50%           3.000000     4.000000
75%           7.000000     5.000000
max          17.000000     9.000000
```

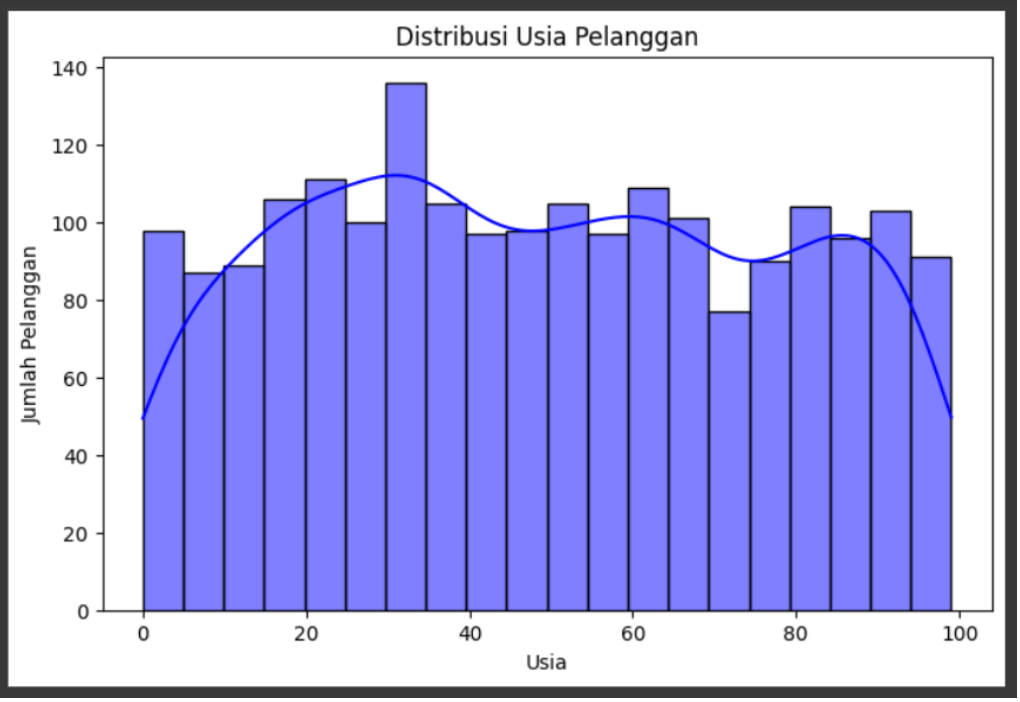
6. Mengisi data kosong

```
# Isi NaN pada kolom numerik dengan mean
df.fillna(df.select_dtypes(include=['number']).mean(), inplace=True)

# Isi NaN pada kolom non-numerik dengan mode (nilai terbanyak)
for col in df.select_dtypes(include=['object']).columns:
    df[col].fillna(df[col].mode()[0], inplace=True)

# Cek jumlah data kosong setelah pengisian
print("\nJumlah Data Kosong Setelah Pengisian:")
print(df.isnull().sum())
```

```
Jumlah Data Kosong Setelah Pengisian:
CustomerID              0
Gender                  0
Age                     0
Annual Income ($)       0
Spending Score (1-100)  0
Profession              0
Work Experience         0
Family Size             0
dtype: int64
<ipython-input-118-7574b1232b2a>:14: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perfor


  df[col].fillna(df[col].mode()[0], inplace=True)
```

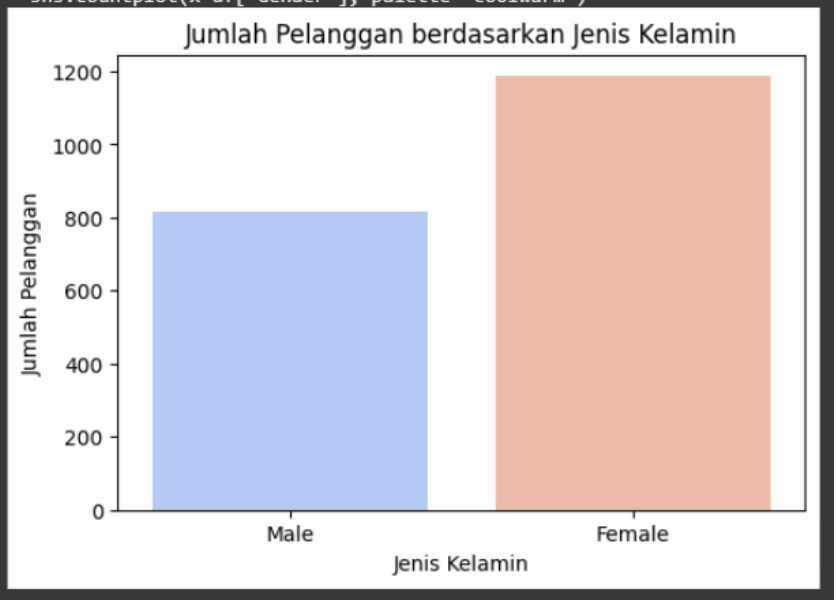
7. Analisis dalam bentuk visualisasi

```python
# Visualisasi distribusi usia pelanggan
plt.figure(figsize=(8, 5))
sns.histplot(df['Age'], bins=20, kde=True, color='blue')
plt.title('Distribusi Usia Pelanggan')
plt.xlabel('Usia')
plt.ylabel('Jumlah Pelanggan')
plt.show()
```



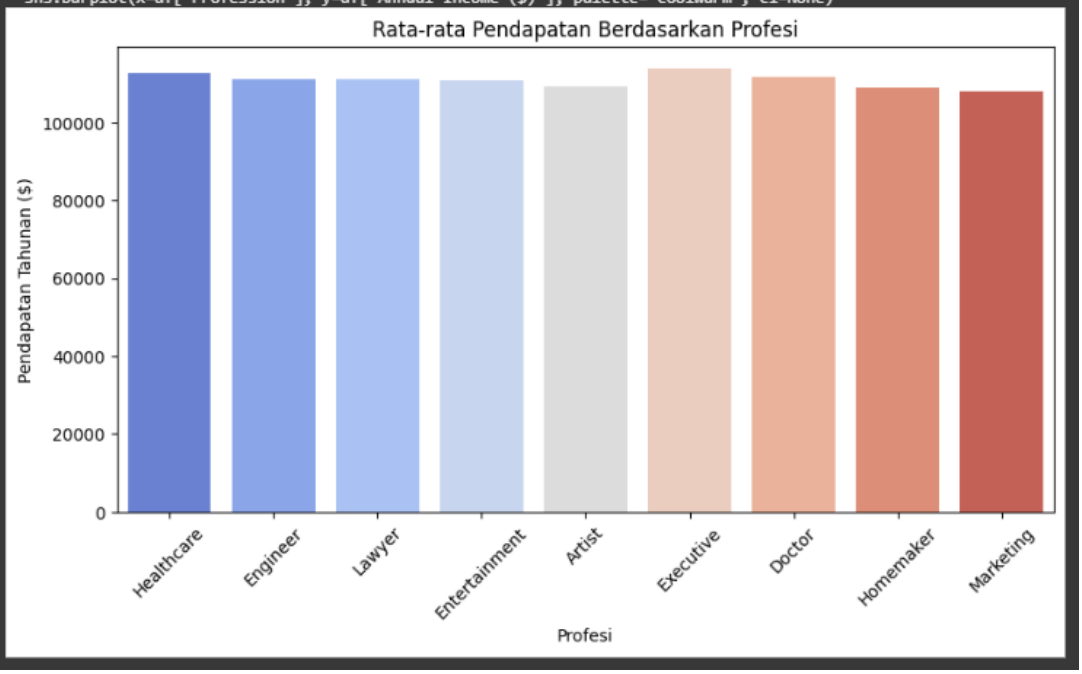

```python
# Visualisasi jumlah pelanggan berdasarkan jenis kelamin
plt.figure(figsize=(6, 4))
sns.countplot(x=df['Gender'], palette='coolwarm')
plt.title('Jumlah Pelanggan berdasarkan Jenis Kelamin')
plt.xlabel('Jenis Kelamin')
plt.ylabel('Jumlah Pelanggan')
plt.show()
```

```
<ipython-input-123-0acdd5ae9918>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remov

  sns.countplot(x=df['Gender'], palette='coolwarm')
```



```
plt.figure(figsize=(10, 5))
sns.barplot(x=df["Profession"], y=df["Annual Income ($)"], palette="coolwarm", ci=None)
plt.xticks(rotation=45)
plt.title("Rata-rata Pendapatan Berdasarkan Profesi")
plt.xlabel("Profesi")
plt.ylabel("Pendapatan Tahunan ($)")
plt.show()
```

```
<ipython-input-124-103660285d8f>:9: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.barplot(x=df["Profession"], y=df["Annual Income ($)"], palette="coolwarm", ci=None)
<ipython-input-124-103660285d8f>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to

  sns.barplot(x=df["Profession"], y=df["Annual Income ($)"], palette="coolwarm", ci=None)
```



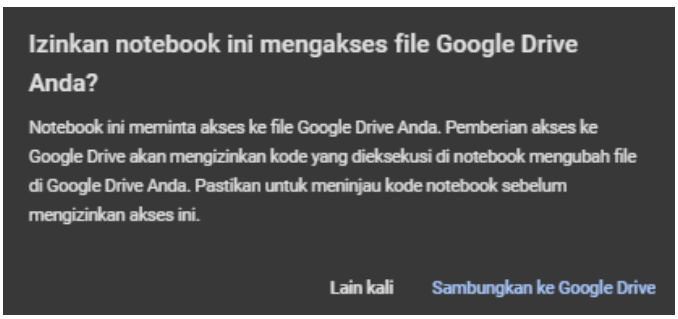
4. Gunakan dataset primer yang telah anda buat pada pertemuan 1, lakukan hal yang sama yang telah anda lakukan pada nomor 2 dan 3 diatas.

    1. Mengaupload ke google drive

    2. Menyambungkan Google Drive ke Google Colab

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```



3. Menampikan 5 data pertama pada data set Customers.csv

```
df = pd.read_excel(file_path)
print(df.head())
```

```
   Absensi             Nama Lengkap         Tempat Tanggal Lahir  Usia  \
0        1    Abhinaya Nuzuluzzuhdi      Malang, 31 Oktober 2004    20
1        2       Alvi Choirinnikmah     Blitar, 9 September 2004    20
2        3           Alya Ajeng Ayu     Malang, 18 November 2004    20
3        4  Ardhelia Putri Maharani      Malang, 11 Oktober 2004    20
4        5          Bagas Nusa Tama  Yogyakarta, 16 agustus 2005    19

  Jenis Kelamin                    Alamat Kota Tempat Tinggal  \
0     Laki-laki               Jl. Gajayana              Malang
1     Perempuan          Jln. Kembang Turi              Malang
2     Perempuan   Jl. Parkit Selatan no. 2              Malang
3     Perempuan  Serenia Garden Regency B9              Malang
4   Laki - Laki          Jl.Kembang kertas           Pontianak

  Jenis Kendaraan      Pengeluaran BBM   IPK               Hobi  \
0    Sepeda Motor           Rp. 90.000   NaN          Maen Game
1    Sepeda Motor                  NaN   NaN      Menonton film
2    Sepeda Motor  Rp. 25.000 - 30.000  3.74             Baking
3    Sepeda Motor           Rp. 30.000   NaN        Live Tiktok
4    Sepeda Motor           Rp. 20.000   NaN  Bermain alat musik

  Tinggi dan Berat Badan Data Lain
0             173cm 50kg       NaN
1                    NaN       NaN
2             155cm 49kg       NaN
3             164cm 46kg       NaN
4                    NaN       NaN
```

4. Menampilkan deskripsi data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28 entries, 0 to 27
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Absensi                 28 non-null     int64
 1   Nama Lengkap            28 non-null     object
 2   Tempat Tanggal Lahir    28 non-null     object
 3   Usia                    28 non-null     int64
 4   Jenis Kelamin           28 non-null     object
 5   Alamat                  28 non-null     object
 6   Kota Tempat Tinggal     28 non-null     object
 7   Jenis Kendaraan         28 non-null     object
 8   Pengeluaran BBM         23 non-null     object
 9   IPK                     6 non-null      float64
 10  Hobi                    18 non-null     object
 11  Tinggi dan Berat Badan  12 non-null     object
 12  Data Lain               1 non-null      object
dtypes: float64(1), int64(2), object(10)
memory usage: 3.0+ KB
```

	Absensi	Usia	IPK
count	28.000000	28.000000	6.000000
mean	14.500000	19.607143	3.731667
std	8.225975	0.566947	0.084479
min	1.000000	19.000000	3.660000
25%	7.750000	19.000000	3.680000
50%	14.500000	20.000000	3.710000
75%	21.250000	20.000000	3.740000
max	28.000000	21.000000	3.890000

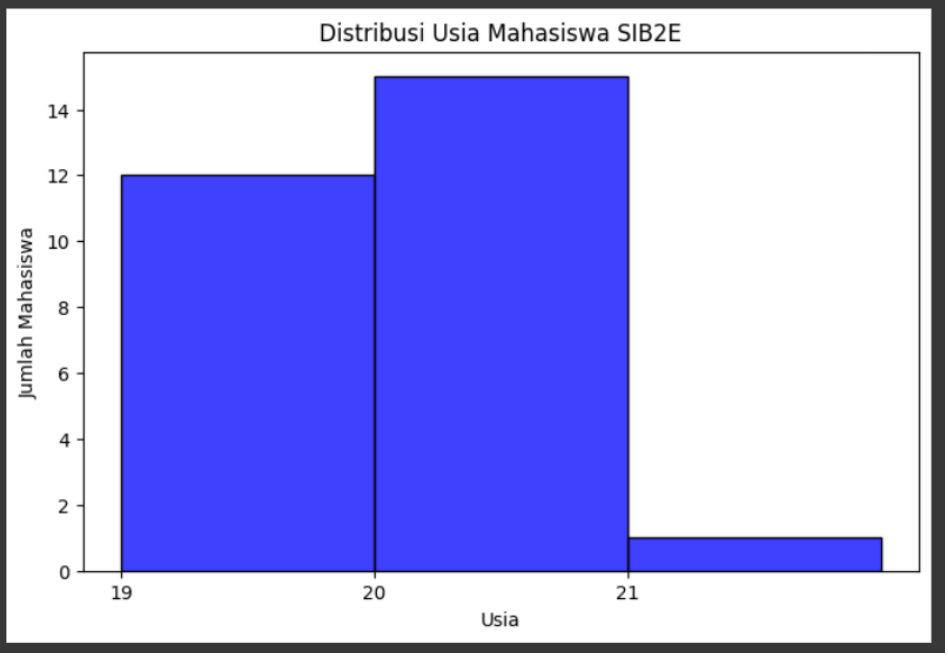
5. Mengisi data kosong

```python
import pandas as pd

# Assuming the file is in your Google Drive's My Drive folder
file_path = '/content/drive/My Drive/DataSIB2E.xlsx'

df = pd.read_excel(file_path)

# Menampilkan jumlah data kosong sebelum pengisian
print("Jumlah Data Kosong Sebelum Pengisian:")
print(df.isnull().sum())

# Mengisi data kosong
for col in df.columns:
    if df[col].dtype == "object":  # Jika tipe data string/categorical
        df[col].fillna(df[col].mode()[0], inplace=True)  # Isi dengan modus (nilai terbanyak)
    else:  # Jika tipe data numerik
        df[col].fillna(df[col].mean(), inplace=True)  # Isi dengan rata-rata

# Menampilkan jumlah data kosong setelah pengisian
print("\nJumlah Data Kosong Setelah Pengisian:")
print(df.isnull().sum())
```

```
Jumlah Data Kosong Sebelum Pengisian:
Absensi                   0
Nama Lengkap              0
Tempat Tanggal Lahir      0
Usia                      0
Jenis Kelamin             0
Alamat                    0
Kota Tempat Tinggal       0
Jenis Kendaraan           0
Pengeluaran BBM           5
IPK                      22
Hobi                     10
Tinggi dan Berat Badan   16
Data Lain                27
dtype: int64

Jumlah Data Kosong Setelah Pengisian:
Absensi                  0
Nama Lengkap             0
Tempat Tanggal Lahir     0
Usia                     0
Jenis Kelamin            0
Alamat                   0
Kota Tempat Tinggal      0
Jenis Kendaraan          0
Pengeluaran BBM          0
IPK                      0
Hobi                     0
Tinggi dan Berat Badan   0
Data Lain                0
dtype: int64
<ipython-input-159-bd836718c988>:19: Fut
The behavior will change in pandas 3.0.
```

6. Analisis dalam bentuk visualisasi

```
plt.figure(figsize=(8,5))
sns.histplot(df["Usia"].astype(int), bins=[19, 20, 21, 22], color='blue')
plt.xticks([19, 20, 21])
plt.title("Distribusi Usia Mahasiswa SIB2E")
plt.xlabel("Usia")
plt.ylabel("Jumlah Mahasiswa")
plt.show()
```




```
plt.figure(figsize=(6,5))
sns.countplot(x="Jenis Kelamin", data=df, palette="viridis")
plt.title("Jumlah Mahasiswa Berdasarkan Gender")
plt.xlabel("Gender")
plt.ylabel("Jumlah Mahasiswa")
plt.show()
```

```
<ipython-input-166-257aa2f6bd20>:21: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be re

  sns.countplot(x="Jenis Kelamin", data=df, palette="viridis")
```