PRAKTIKUM DATA WAREHOUSING DAN DATA MINING

MODUL 7

DATA PREPROCESSING

Disusun oleh:

Adinda Aulia Hapsari

L200220037

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS KOMUNIKASI DAN INFORMATIKA

UNIVERSITAS MUHAMMADIYAH SURAKARTA

TAHUN 2024

Setelah kegiatan selesai, lembar kerja ini dicetak (di-print) dan dikumpulkan ke asisten. NIM : L200220037 Nama : Adinda Aulia Hapsari Nama Asisten : Diva Halimah Tanggal Praktikum : 15 November 2024	(Diisi oleh Asisten) Nilai Praktek : Tanda Tangan :

KEGIATAN PRAKTIKUM

1. Buka Windows Explorer dan arahkan pada folder Praktikum Data Warehousing dan Data Mining. Buat sebuah folder sesuai dengan NIM mahasiswa di dalam folder Praktikum Data Warehousing dan Data Mining.
2. Unduh semua data dari dataset Titanic dari repository gitea Bab 07 dan disimpan pada folder NIM yang telah dibuat pada langkah 1.
3. Buka aplikasi Anaconda Navigator untuk menjalankan Jupyter Notebook. Atau jika menggunakan “command prompt”. Ketikkan perintah jupyter notebook kemudian tekan Enter.
4. Aplikasi Jupyter Notebook akan dijalankan pada sebuah browser yang terinstal di komputer.
5. Buat file kerja baru untuk menulis kode-kode program, klik tombol New -> Python 3.
6. Dengan mengklik nama file “Untitled”, ubahlah nama file dengan format “Latihan 7_1-NIM”, misalnya “Latihan_7_1-L2002201000”. File ini akan tersimpan di komputer dalam folder C:/Praktikum Data Warehousing dan Data Mining/NIM/Latihan 7_1-L2002201000.ipynb.
7. Pada sel pertama, kita melakukan import beberapa library yang dibutuhkan dalam data preprocessing, antara lain numpy, pandas, matplotlib, seaborn, dan sklearn. Eksekusi kode pada sel pertama dengan menekan tombol Run atau shift+Enter pada keyboard.

```
#Import Libraries
import numpy as np
import pandas as pd
import seaborn as sns
```

8. Selanjutnya, kita akan mengakses tiga dataset yang telah kita unduh sesuai pada Langkah 2 dengan menggunakan library pandas untuk disimpan dalam bentuk dataframe.

```
#Load data
train_data = pd.read_csv('train.csv')
test_data = pd.read_csv('test.csv')
gender_submission = pd.read_csv('gender_submission.csv')
```

9. Untuk menampilkan dataframe dari masing-masing dataset, gunakan method head(), misalnya train_data.head() yang secara default akan menampilkan 5 data pertama.

```
train_data.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

10. Informasi deskripsi dari masing-masing dataframe, seperti nilai rata-rata (mean), standard deviation (std), nilai minimum (min), nilai maksimum (max), dan lain-lainnya dapat dilihat dengan menambahkan method describe() pada dataframe.

```
train_data.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Data preprocessing merupakan salah satu tahap awal dalam data mining untuk melakukan pembersihan data dari berbagai kesalahan misalnya data yang kosong, menghapus atribut yang tidak diperlukan dalam data mining, maupun mengubah data yang ada menjadi data yang berbeda misalnya menggabungkan nilai dari 2 kolom menjadi 1, mengubah data teks menjadi data angka atau sebaliknya.

Number of Missing Values

11. Pada tahap ini, kita akan melihat jumlah data dari semua atribut/ kolom yang tidak memiliki nilai. Ketikkan perintah berikut ini pada sel jupyter kemudian dieksekusi.

```
column_names = train_data.columns
for column in column_names:
    print(column + ' - ' + str(train_data[column].isnull().sum()))
```

```
PassengerId - 0
Survived - 0
Pclass - 0
Name - 0
Sex - 0
Age - 177
SibSp - 0
Parch - 0
Ticket - 0
Fare - 0
Cabin - 687
Embarked - 2
```

Hasil ini menunjukkan bahwa ada beberapa atribut yang masih memiliki data yang kosong, misalnya kolom Age terdapat 177 data kosong, Cabin memiliki 687 data kosong, dan Embarked memiliki 2 data kosong.

12. Pada dataset Titanic, atribut Survived adalah atribut yang menjadi target dalam analisis, misalnya untuk keperluan prediksi keselamatan (survival) para penumpang. Untuk melihat data jumlah penumpang yang selamat dan tidak selamat dalam data pelatihan (train.csv), kita bisa menggunakan method value_counts() terhadap kolom dalam dataframe.

```
train_data['Survived'].value_counts()
```

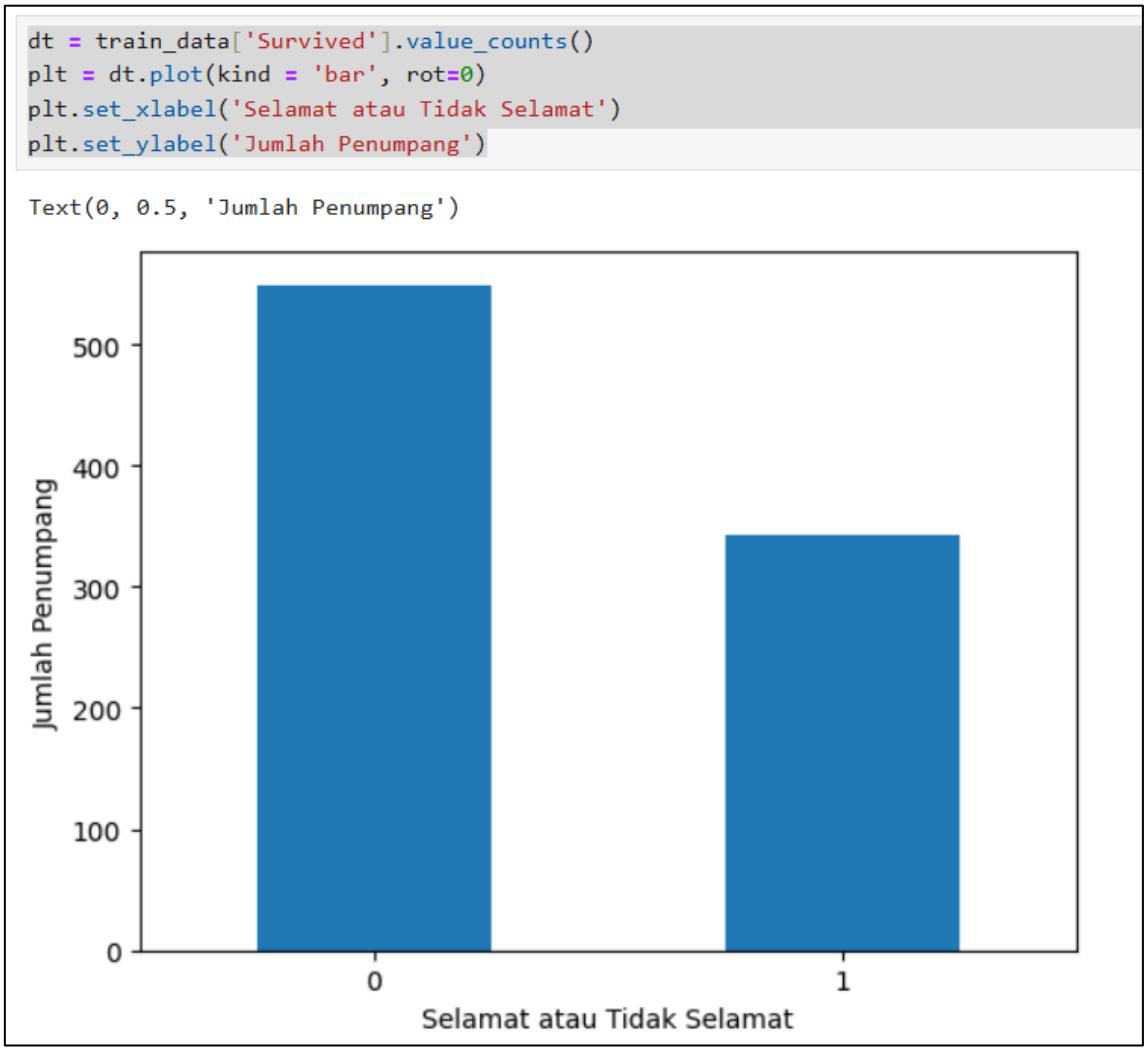
```
Survived
0    549
1    342
Name: count, dtype: int64
```

Dapat dilihat bahwa data yang bernilai 0 (survived) ada 549 orang, sedangkan yang bernilai 1 (not survived) ada 342 orang.

13. Untuk menampilkan jumlah orang yang selamat maupun tidak selamat dalam bentuk grafik batang, maka data dihitung berdasarkan kolom Survived. Kemudian untuk menampilkan grafik batang bisa menggunakan method plot(kind=’bar’).

```
dt = train_data['Survived'].value_counts()
plt = dt.plot(kind = 'bar', rot=0)
plt.set_xlabel('Selamat atau Tidak Selamat')
plt.set_ylabel('Jumlah Penumpang')
```

```
Text(0, 0.5, 'Jumlah Penumpang')
```

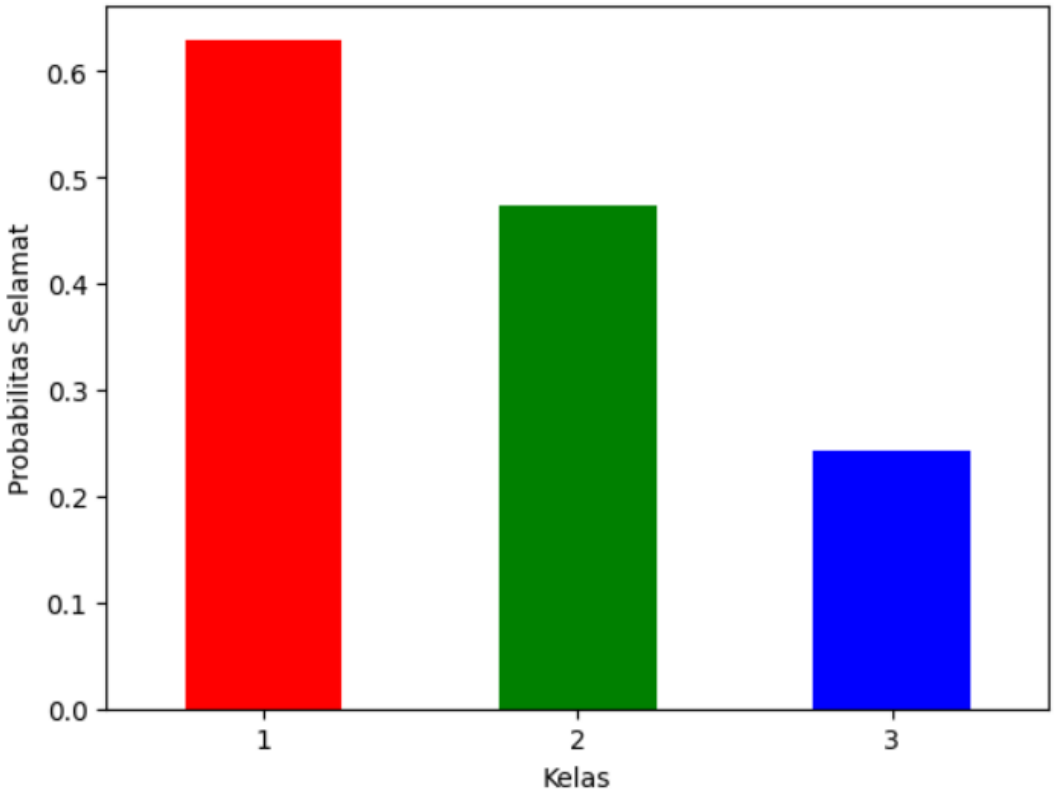


Dapat dilihat dari grafik bahwa penumpang yang tidak selamat (nilai=0) lebih banyak dibandingkan dengan penumpang yang selamat (nilai=1).

14. Tahap berikutnya adalah melihat tingkat kemungkinan keselamatan (Survived) berdasarkan kelas penumpang (Pclass).

```
dt = train_data[['Pclass', 'Survived']].groupby('Pclass').mean().Survived
plt = dt.plot(kind='bar', rot=0, color=['red', 'green', 'blue'])
plt.set_xlabel('Kelas')
plt.set_ylabel('Probabilitas Selamat')
```

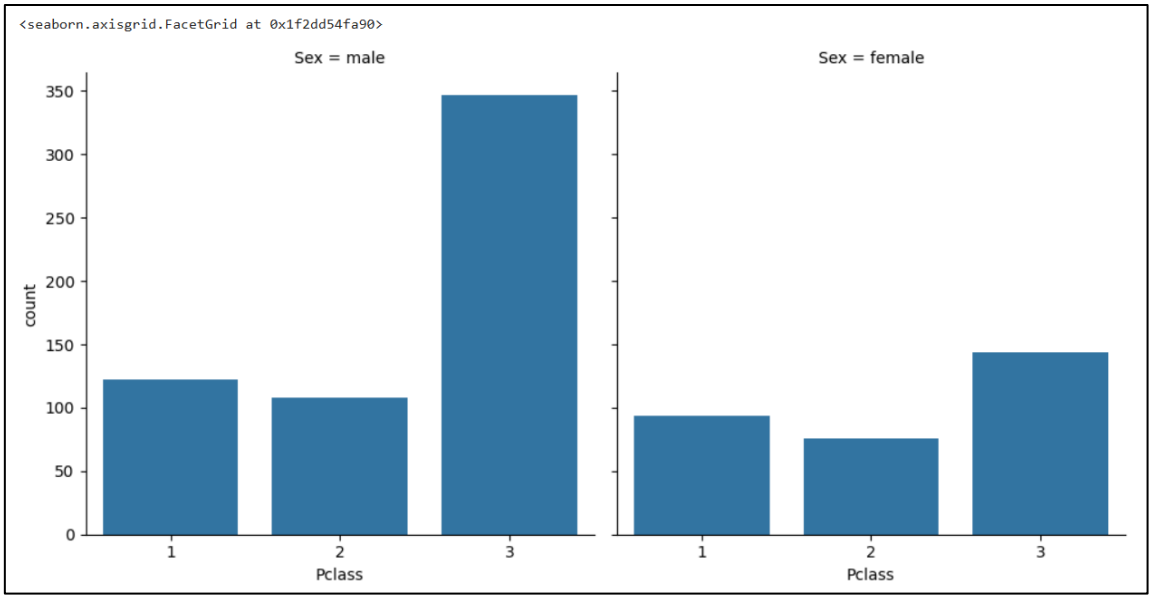
```
Text(0, 0.5, 'Probabilitas Selamat')
```



Dapat dilihat dari grafik bahwa penumpang yang berada di Kelas 1 memiliki probabilitas keselamatan paling tinggi dibandingkan dengan kelas yang lainnya.

15. Dataset Titanic juga bisa dilihat secara multidimensi dengan kode python, yaitu dengan menggunakan seaborn (sns) dan method catplot(). Misalnya untuk menampilkan kelas penumpang (Pclass) vs. jenis kelamin (Sex).

```
sns.catplot(x = 'Pclass', col = 'Sex', data = train_data, kind = 'count')
```



16. Dalam proses data mining, beberapa atribut terkadang tidak diperlukan karena dianggap tidak penting untuk dianalisis. Pada contoh dataset Titanic, kolom Id Penumpang (PassengerId) dan Nomor Tiket (Ticket) tidak penting untuk dianalisis sehingga bisa saja dihapus dari dataset. Begitu juga atribut yang memiliki data kosong banyak, juga bisa dihapus dari dataset karena justru akan mengganggu proses data mining jika tetap digunakan, misalnya kolom Cabin.
    Untuk menghapus 3 kolom tersebut, maka gunakan method drop() terhadap dataframe yang digunakan, kemudian kode dieksekusi.

```
train_data = train_data.drop(columns=['Ticket', 'PassengerId', 'Cabin'])
train_data.head()
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	7.2500	S
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	71.2833	C
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	7.9250	S
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	53.1000	S
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	8.0500	S

    Hasil eksekusi kode menunjukkan bahwa 3 kolom yang telah dihapus sudah hilang dari dataset.

17. Pada tahap data preprocessing, penambahan fitur (kolom) baru berdasarkan atribut yang telah ada terkadang diperlukan untuk analisis lebih lanjut. Dalam contoh ini, jumlah penumpang, jumlah saudara/pasangan, dan jumlah orang

tua/anak bisa digabung menjadi 1 kolom misalnya kolom FamiliySize (Jumlah keluarga) yang dihitung dari jumlah saudara/pasangan ditambah dengan jumlah orang tua/anak dan termasuk si pemilik tiket dengan formula SibSp + Parch + 1.

```
train_data['FamilySize'] = train_data['SibSp'] + train_data['Parch'] + 1
train_data.head()
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize
0	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	7.2500	S	2
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	71.2833	C	2
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	7.9250	S	1
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	53.1000	S	2
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	8.0500	S	1

Hasil penambahan fitur baru bisa dilihat dari kolom FamilySize yang berada di paling kanan.

18. Pengolahan data mining, beberapa algoritma hanya bisa dilakukan pada data yang bertipe angka (numeric). Sehingga jika ada data yang dimiliki bertipe teks padahal penting untuk digunakan dalam data mining, maka perlu diubah (transform) menjadi tipe angka. Pada contoh dataset Titanic, atribut jenis kelamin (Sex) dan lokasi keberangkatan (Embarked) akan diubah menjadi tipe angka dengan ketentuan berikut:
Sex: male=0; female=1
Embarked: C=0; Q=1; S=2

```
train_data['Sex'] = train_data['Sex'].map({'male':0, 'female':1})
train_data['Embarked'] = train_data['Embarked'].map({'C':0, 'Q':1, "S":2})
train_data.head()
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize
0	0	3	Braund, Mr. Owen Harris	0	22.0	1	0	7.2500	2.0	2
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	1	0	71.2833	0.0	2
2	1	3	Heikkinen, Miss. Laina	1	26.0	0	0	7.9250	2.0	1
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	35.0	1	0	53.1000	2.0	2
4	0	3	Allen, Mr. William Henry	0	35.0	0	0	8.0500	2.0	1

19. Berikutnya adalah melakukan preprocess terhadap nama penumpang. Di dataset Titanic, nama penumpang mengandung sebutan/gelar misalnya Mr, Miss, Mrs, Dr, Lady, dan lain-lain. Sebutan/gelar ini diperlukan dalam analisis data mining, namun nama penumpang tidak diperlukan. Sehingga

gelar penumpang akan diekstraksi terlebih dahulu kemudian dikategorikan sebelum proses lebih lanjut. Untuk mengekstraksi data sebutan/gelar, maka diperlukan sebuah Regular Expression (Regex), yaitu sebuah teks (string) yang mendefinisikan sebuah pola pencarian sehingga dapat membantu untuk melakukan matching (pencocokan), locate (pencarian), dan manipulasi teks. Regex yang diperlukan untuk mengekstraksi sebutan/gelar pada kolom Name adalah '([A-Za-z]+)\.' yang dimasukkan pada method extract().

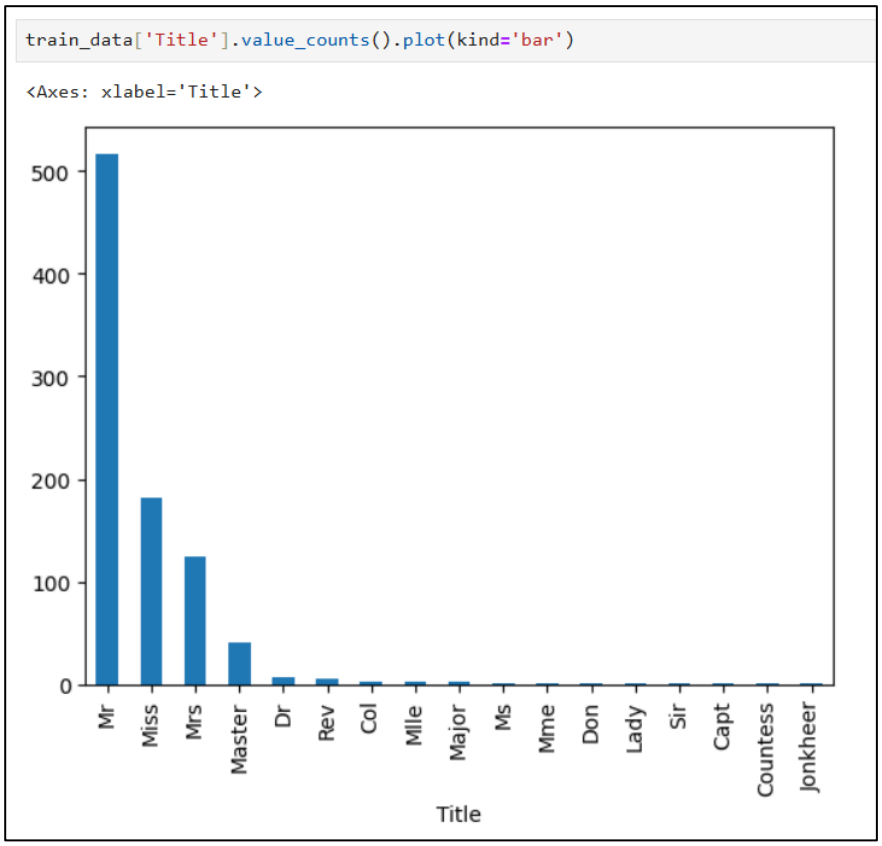
```
train_data['Title'] = train_data['Name'].str.extract('([A-Za-z]+)\.', expand=False)
train_data = train_data.drop(columns='Name')
train_data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize	Title
0	0	3	0	22.0	1	0	7.2500	2.0	2	Mr
1	1	1	1	38.0	1	0	71.2833	0.0	2	Mrs
2	1	3	1	26.0	0	0	7.9250	2.0	1	Miss
3	1	1	1	35.0	1	0	53.1000	2.0	2	Mrs
4	0	3	0	35.0	0	0	8.0500	2.0	1	Mr

Untuk melihat sebaran data penumpang berdasarkan kategori sebutan/ gelar (Title) penumpang, maka bisa dilihat dengan grafik batang.

```
train_data['Title'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Title'>
```



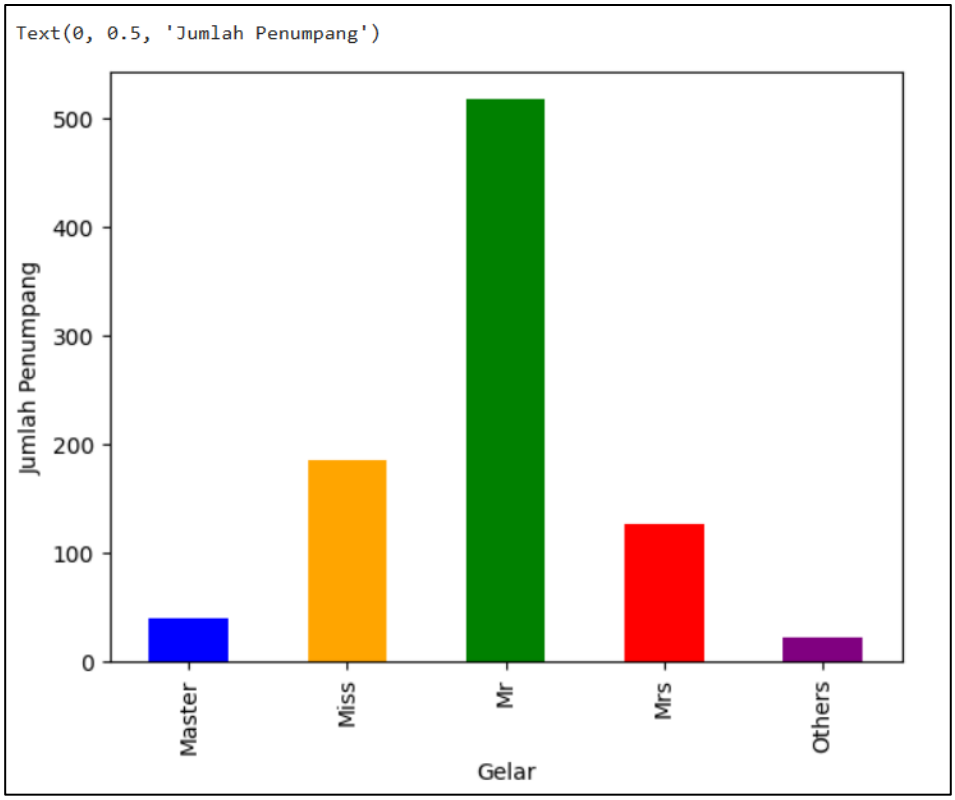
20. Berdasarkan grafik pada langkah 19, dapat dilihat bahwa terdapat banyak sebutan/gelar nama penumpang yang tidak biasa atau unik, seperti Rev, Major, Col, Mlle, dan lain-lainnya. Pada kasus ini, misalnya gelar nama hanya akan dikategorikan menjadi 5 jenis, yaitu Master, Mr, Mrs, Miss, dan Others. Sehingga diperlukan mengganti gelar gelar yang unik berikut ini dengan kategori yang telah ditentukan:
    a. Dr, Rev, Col, Major, Countess, Sir, Jonkheer, Lady, Capt, Don diganti dengan Others.
    b. Ms diganti dengan Miss.
    c. Mme diganti dengan Mrs.
    d. Mlle diganti dengan Miss.

    Sedangkan jika gelar nama penumpang sudah berupa Master, Mr, Mrs, dan Miss, maka tidak perlu diganti. Gunakan method replace() untuk mengganti gelar nama penumpang sesuai dengan kategori yang ditentukan.

```
train_data['Title'] = train_data['Title'].replace(
    ['Dr', 'Rev', 'Col', 'Major', 'Countess', 'Sir', 'Jonkheer', 'Lady', 'Capt', 'Don'], 'Others')
train_data['Title'] = train_data['Title'].replace('Ms', 'Miss')
train_data['Title'] = train_data['Title'].replace('Mme', 'Mrs')
train_data['Title'] = train_data['Title'].replace('Mlle', 'Miss')
```

21. Untuk mengetahui sebaran data penumpang berdasarkan sebutan/ gelarnya, maka bisa dilihat dengan grafik batang dari 5 kategori sebutan tersebut.

```
plt = train_data['Title'].value_counts().sort_index().plot(
    kind='bar', color=['blue', 'orange', 'green', 'red', 'purple'])
plt.set_xlabel('Gelar')
plt.set_ylabel('Jumlah Penumpang')
```



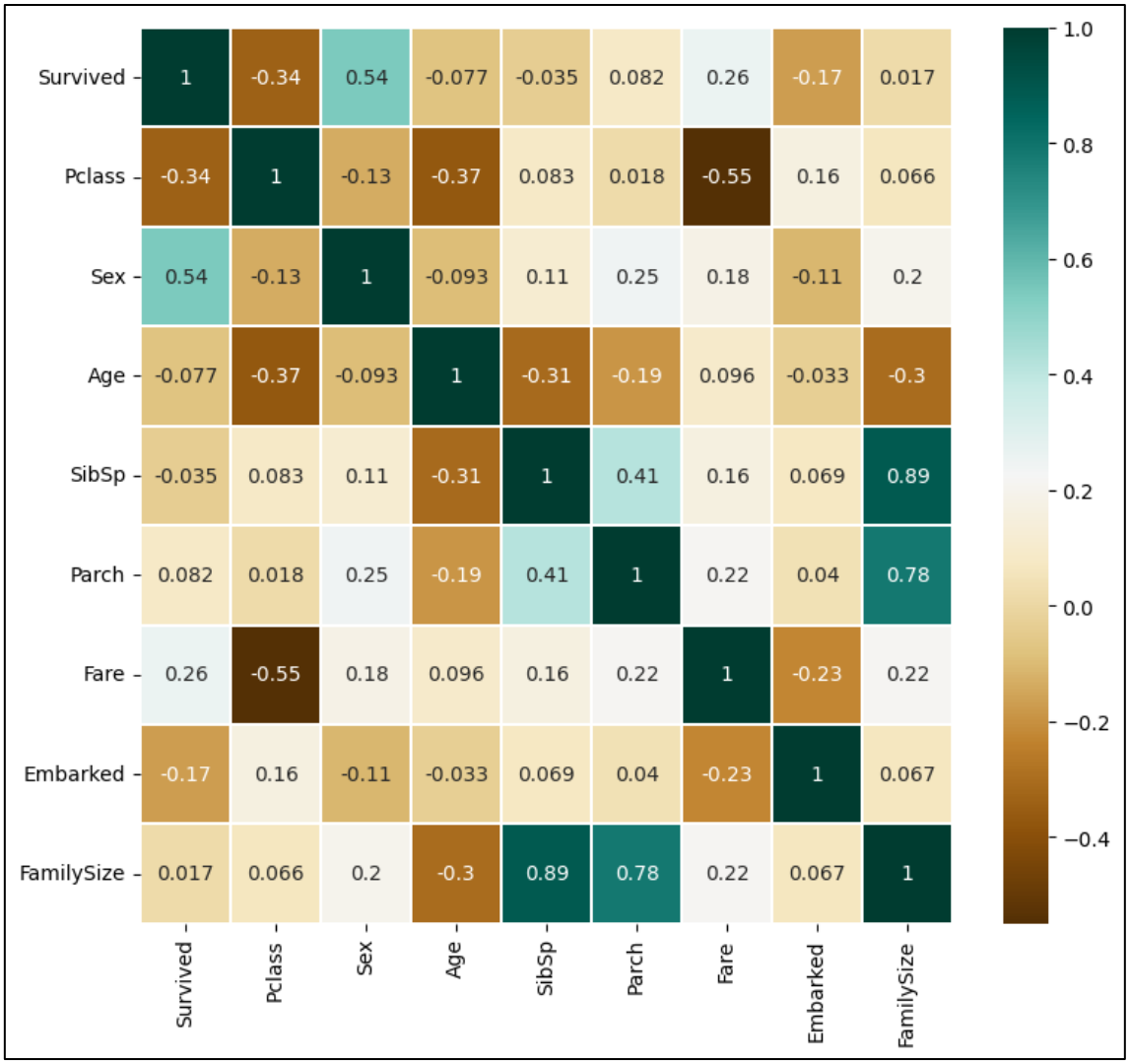
22. Sebelum melakukan analisis data lebih lanjut, para peneliti terkadang mencari korelasi antar atribut. Korelasi antar atribut bisa dicari dengan menerapkan method corr() pada dataframe yang dianalisis.

```
num = train_data.select_dtypes(include = [np.number])
corr_matrix = num.corr()
```

Kemudian gambarkan dengan grafik heatmap menggunakan library matplotlib.

```
import matplotlib.pyplot as plt
plt.figure(figsize = (9, 8))
sns.heatmap(data = corr_matrix, cmap = "BrBG", annot = True, linewidths = 0.2)
```

Sehingga akan ditampilkan grafik heatmap.



23. Berdasarkan hasil langkah 11, kita ketahui terdapat 3 atribut yang memiliki data kosong, yaitu Age, Cabin, dan Embarked. Namun atribut Cabin sudah dihilangkan dari dataset pada langkah 16, maka hanya tinggal 2 atribut yang memiliki data kosong. Untuk menangani data yang kosong, kita bisa melakukan beberapa pendekatan misalnya mengisi data kosong menggunakan nilai mayoritas, nilai rerata, nilai median, dan lain-lain tergantung datanya. Pada langkah ini, kita akan menangani data kosong pada atribut Embarked terlebih dahulu.

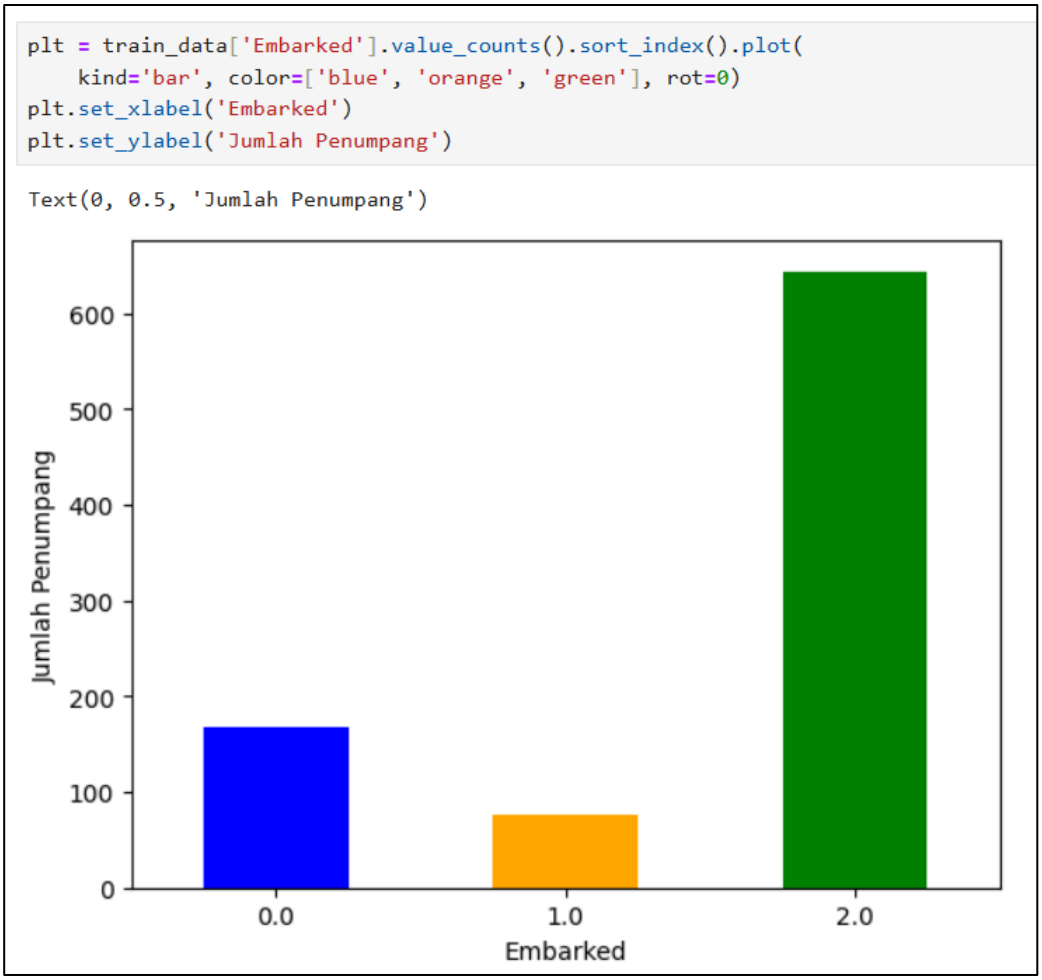
```
missing = train_data['Embarked'].isnull().sum()
print("Jumlah data kosong pada atribut Embarked: ",missing)
```

```
Jumlah data kosong pada atribut Embarked:  2
```

24. Berdasarkan langkah 23, atribut ini hanya memiliki 2 data kosong, sehingga kita bisa mengisikannya berdasarkan nilai mayoritas pada atribut tersebut.

```
plt = train_data['Embarked'].value_counts().sort_index().plot(
    kind='bar', color=['blue', 'orange', 'green'], rot=0)
plt.set_xlabel('Embarked')
plt.set_ylabel('Jumlah Penumpang')
```

```
Text(0, 0.5, 'Jumlah Penumpang')
```



25. Nilai mayoritas pada atribut Embarked adalah “S” (Southampton). yaitu data dengan nilai 2 (berdasarkan langkah 18), sehingga data yang kosong bisa kita isikan dengan nilai 2 menggunakan method fillna().

```
train_data['Embarked'] = train_data['Embarked'].fillna(2)
train_data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize	Title
0	0	3	0	22.0	1	0	7.2500	2.0	2	Mr
1	1	1	1	38.0	1	0	71.2833	0.0	2	Mrs
2	1	3	1	26.0	0	0	7.9250	2.0	1	Miss
3	1	1	1	35.0	1	0	53.1000	2.0	2	Mrs
4	0	3	0	35.0	0	0	8.0500	2.0	1	Mr

26. Sedangkan untuk menangani data kosong pada atribut Age, kita akan menggunakan nilai median yang diambilkan dari semua data yang memiliki nilai yang sama dari 3 atribut lainnya, yaitu SibSp, Parch, dan Pclass. Jadi,

mula-mula jika ada data yang bernilai kosong pada kolom Age, maka kita perlu melihat nilai atribut SibSp, Parch, dan Pclass pada data tersebut. Kemudian kita cari data lainnya dalam dataset yang memiliki nilai sama persis dalam atribut SibSp, Parch, dan Pclass. Pencarian ini bisa saja menghasilkan beberapa data lainnya yang memiliki nilai yang sama pada ketiga atribut tersebut. Data-data yang bernilai sama pada ketiga atribut tersebut, kemudian dihitung nilai mediannya yang kemudian nilai median ini digunakan untuk mengisi data kosong dalam atribut Age. Namun jika tidak ditemukan data yang sama dari ketiga atribut tersebut dalam dataset, maka nilai Age yang kosong akan diisi dengan nilai median atribut Age secara keseluruhan. Langkah ini diulang beberapa kali sebanyak jumlah data Age yang kosong dalam dataset.

27. Mula-mula kita identifikasi indeks data Age yang bernilai kosong.

```
NaN_indexes = train_data['Age'][train_data['Age'].isnull()].index
print(NaN_indexes)
```

```
Index([  5,  17,  19,  26,  28,  29,  31,  32,  36,  42,
       ...
       832, 837, 839, 846, 849, 859, 863, 868, 878, 888],
      dtype='int64', length=177)
```

28. Sesuai langkah 27, kita akan menggantikan data kosong pada kolom Age dengan nilai mediannya. Jika muncul informasi terkait dengan: SettingWithCopyWarning, maka abaikan saja.

```
for i in NaN_indexes:
    pred_age = train_data['Age'][((train_data.SibSp == train_data.iloc[i]["SibSp"]) &
                                  (train_data.Parch == train_data.iloc[i]["Parch"]) &
                                  (train_data.Pclass == train_data.iloc[i]["Pclass"])
                                 )].median()
    if np.isnan(pred_age):
        train_data['Age'].iloc[i] = train_data['Age'].median()
    else:
        train_data['Age'].iloc[i] = pred_age
```

```
C:\Users\Acer\AppData\Local\Temp\ipykernel_15860\588149898.py:9: FutureWarning: ChainedA
You are setting values through chained assignment. Currently this works in certain cases
aviour in pandas 3.0) this will never work to update the original DataFrame or Series, b
l behave as a copy.
```

29. Untuk melihat jumlah data yang kosong pada atribut Age setelah preprocess. Dapat dilihat bahwa sekarang semua atribut tidak memiliki data yang kosong (missing values).

```
train_data.isnull().sum()
```

```
Survived      0
Pclass        0
Sex           0
Age           0
SibSp         0
Parch         0
Fare          0
Embarked      0
FamilySize    0
Title         0
dtype: int64
```

30. Untuk melihat dataset versi terakhir setelah melakukan preprocessing, bisa dengan cara menuliskan nama dataframe-nya, yaitu train_data. Data train dalam versi final setelah melalui serangkaian preprocessing. Data final inilah yang kemudian siap untuk diolah dalam data mining.

```
train_data
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	FamilySize	Title
0	0	3	0	22.0	1	0	7.2500	2.0	2	Mr
1	1	1	1	38.0	1	0	71.2833	0.0	2	Mrs
2	1	3	1	26.0	0	0	7.9250	2.0	1	Miss
3	1	1	1	35.0	1	0	53.1000	2.0	2	Mrs
4	0	3	0	35.0	0	0	8.0500	2.0	1	Mr
...	...	...	...	...	...	...	...	...	...	...
886	0	2	0	27.0	0	0	13.0000	2.0	1	Others
887	1	1	1	19.0	0	0	30.0000	2.0	1	Miss
888	0	3	1	13.5	1	2	23.4500	2.0	4	Miss
889	1	1	0	26.0	0	0	30.0000	0.0	1	Mr
890	0	3	0	32.0	0	0	7.7500	1.0	1	Mr

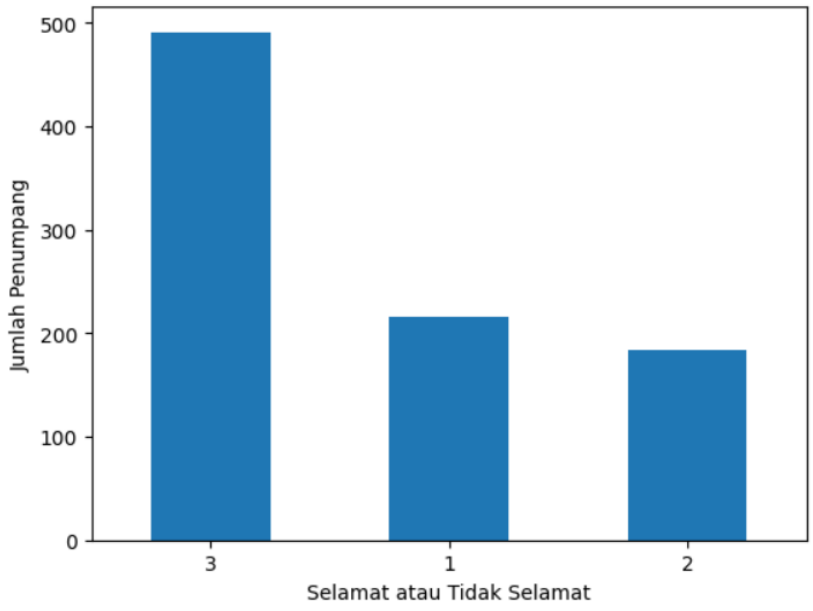
891 rows × 10 columns

TUGAS

Dengan menggunakan dataset train.csv, kerjakan tugas berikut ini:

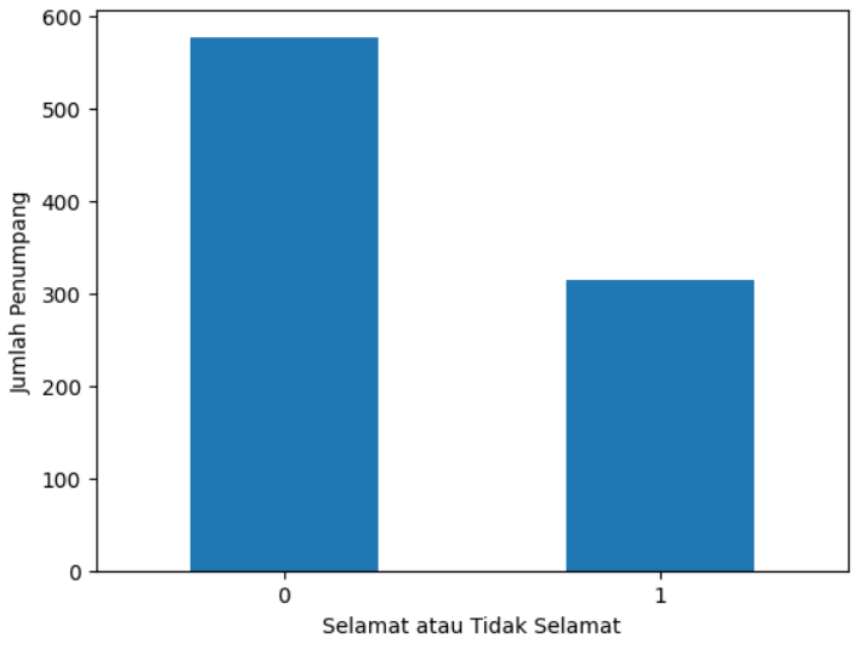
1. Lakukan kembali langkah 13 pada prosedur praktikum untuk melihat data atribut lainnya dengan grafik batang, misalnya Pclass, Sex, dan Embarked!
   a. Grafik batang Pclass

```
[43]: dt = train_data['Pclass'].value_counts()
      plt = dt.plot(kind = 'bar', rot=0)
      plt.set_xlabel('Selamat atau Tidak Selamat')
      plt.set_ylabel('Jumlah Penumpang')

[43]: Text(0, 0.5, 'Jumlah Penumpang')
```



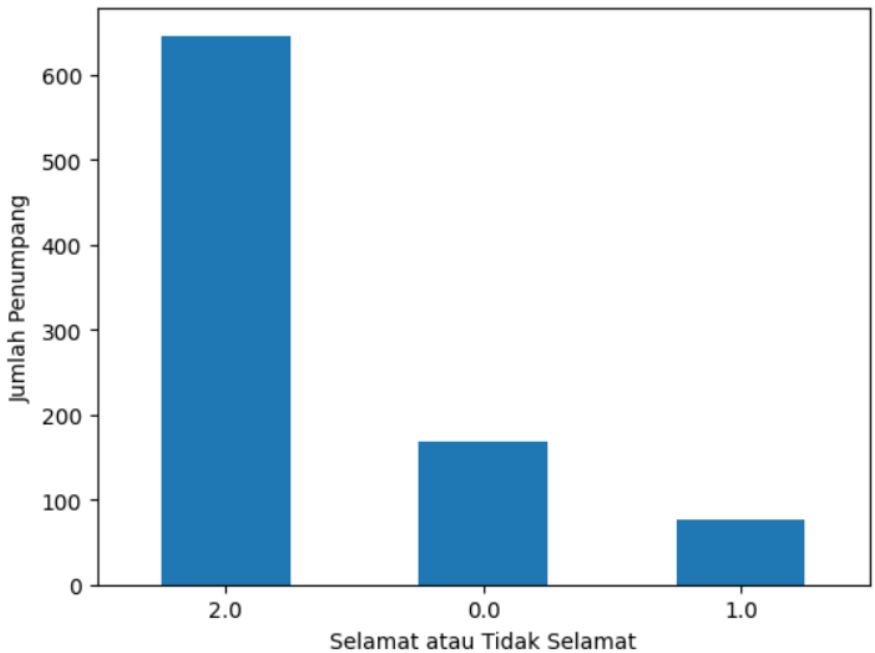
b. Grafik batang Sex

```
[44]: dt = train_data['Sex'].value_counts()
      plt = dt.plot(kind = 'bar', rot=0)
      plt.set_xlabel('Selamat atau Tidak Selamat')
      plt.set_ylabel('Jumlah Penumpang')

[44]: Text(0, 0.5, 'Jumlah Penumpang')
```



c. Grafik batang Embarked

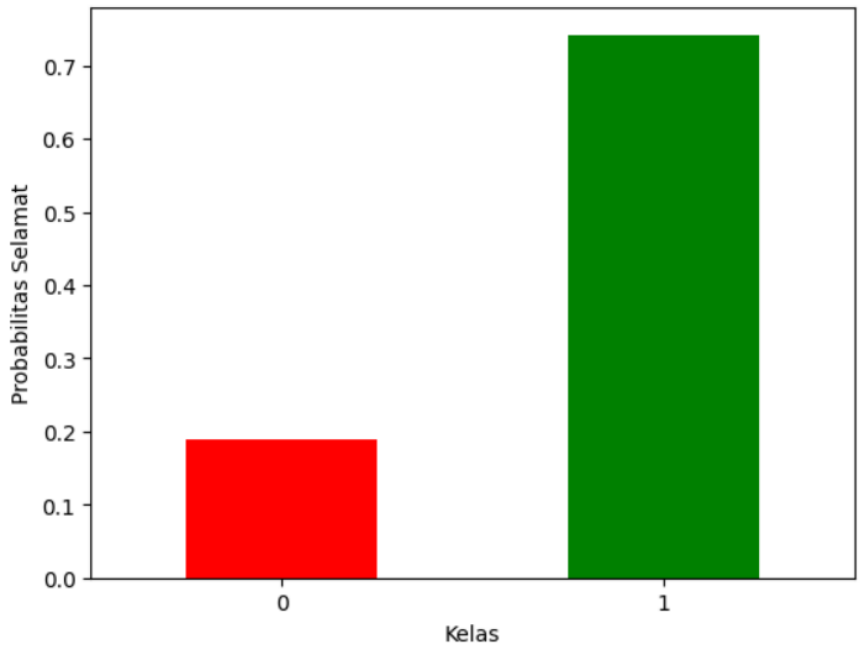
```
[45]: dt = train_data['Embarked'].value_counts()
      plt = dt.plot(kind = 'bar', rot=0)
      plt.set_xlabel('Selamat atau Tidak Selamat')
      plt.set_ylabel('Jumlah Penumpang')

[45]: Text(0, 0.5, 'Jumlah Penumpang')
```



2. Lakukan kembali langkah 14 pada prosedur praktikum untuk melihat probabilitas keselamatan (Survived) berdasarkan jenis kelamin (Sex), lokasi keberangkatan (Embarked), jumlah saudara/pasangan yang ikut (SibSp), dan jumlah orang tua/anak yang ikut (Parch)!
   a. Berdasarkan Jenis Kelamin (Sex)

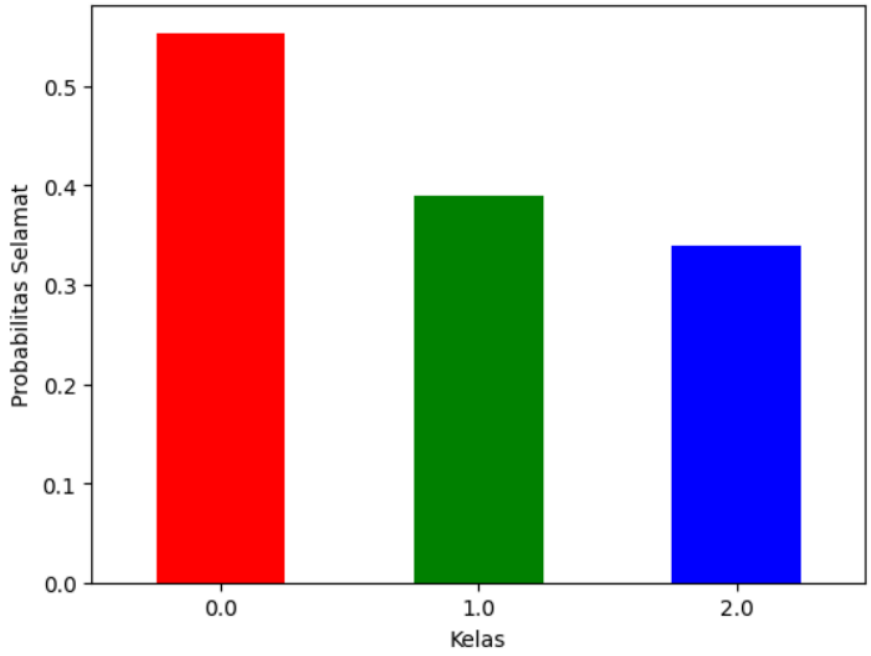
```
[52]: dt = train_data[['Sex', 'Survived']].groupby('Sex').mean().Survived
      plt = dt.plot(kind='bar', rot=0, color=['red', 'green', 'blue'])
      plt.set_xlabel('Kelas')
      plt.set_ylabel('Probabilitas Selamat')
```

```
[52]: Text(0, 0.5, 'Probabilitas Selamat')
```



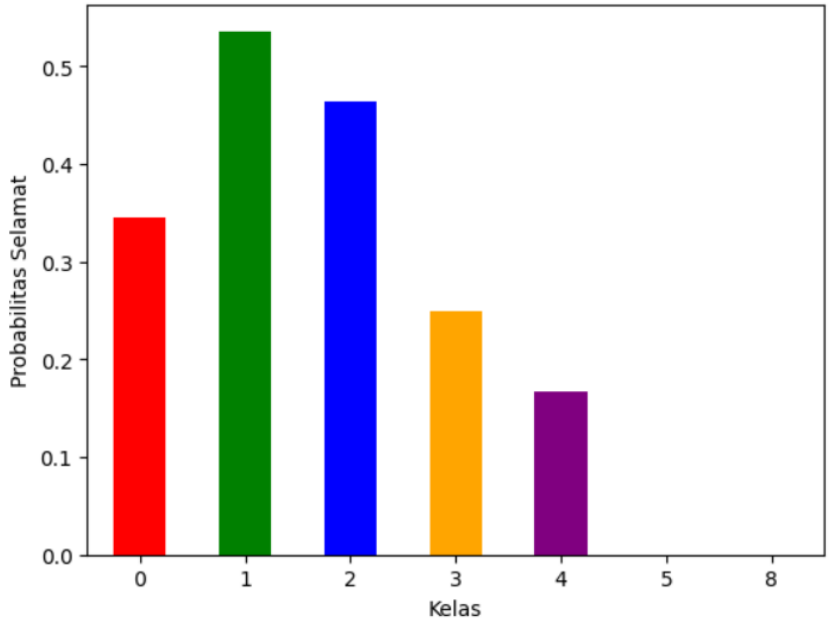
b. Berdasarkan Lokasi Keberangkatan (Embarked)

```
[53]: dt = train_data[['Embarked', 'Survived']].groupby('Embarked').mean().Survived
      plt = dt.plot(kind='bar', rot=0, color=['red', 'green', 'blue'])
      plt.set_xlabel('Kelas')
      plt.set_ylabel('Probabilitas Selamat')

[53]: Text(0, 0.5, 'Probabilitas Selamat')
```



c. Berdasarkan Jumlah Saudara/Pasangan yang Ikut (SibSp)

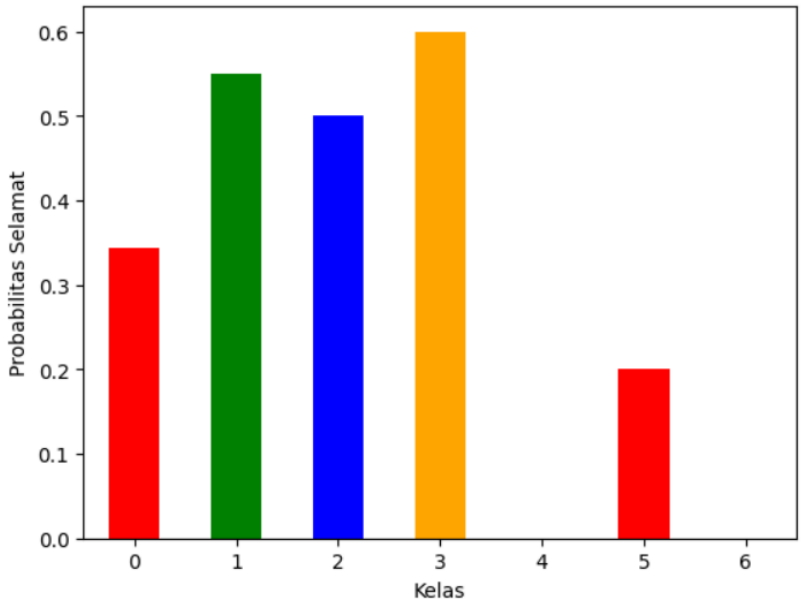
```
[55]: dt = train_data[['SibSp', 'Survived']].groupby('SibSp').mean().Survived
      plt = dt.plot(kind='bar', rot=0, color=['red', 'green', 'blue', 'orange', 'purple'])
      plt.set_xlabel('Kelas')
      plt.set_ylabel('Probabilitas Selamat')

[55]: Text(0, 0.5, 'Probabilitas Selamat')
```



d. Berdasarkan Jumlah Orang Tua/Anak yang Ikut (Parch)

```
[58]: dt = train_data[['Parch', 'Survived']].groupby('Parch').mean().Survived
      plt = dt.plot(kind='bar', rot=0, color=['red', 'green', 'blue', 'orange', 'purple'])
      plt.set_xlabel('Kelas')
      plt.set_ylabel('Probabilitas Selamat')
```

```
[58]: Text(0, 0.5, 'Probabilitas Selamat')
```

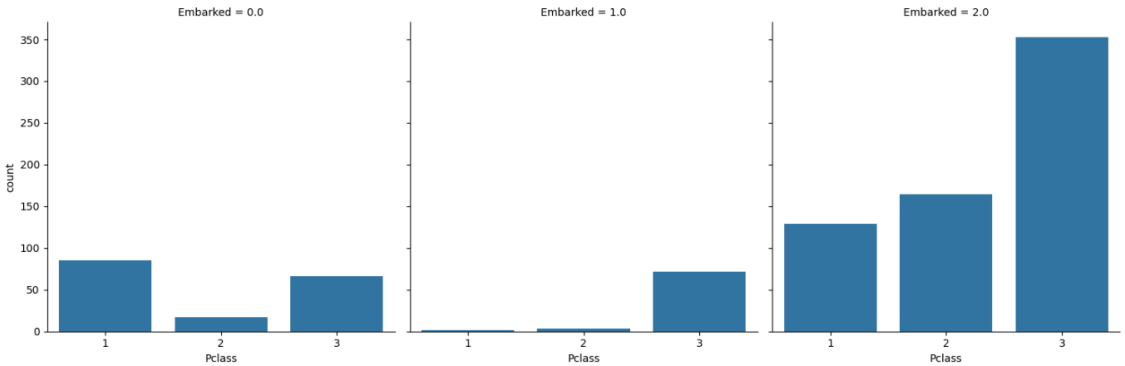


3. Ulangi kembali langkah 15 pada prosedur praktikum untuk melihat multidimensi terhadap atribut kelas penumpang (Pclass) vs lokasi keberangkatan (Embarked), dan jenis kelamin (Sex) vs lokasi keberangkatan (Embarked)!

a. Pclass vs Embarked

```
[59]: sns.catplot(x = 'Pclass', col = 'Embarked', data = train_data, kind = 'count')
```

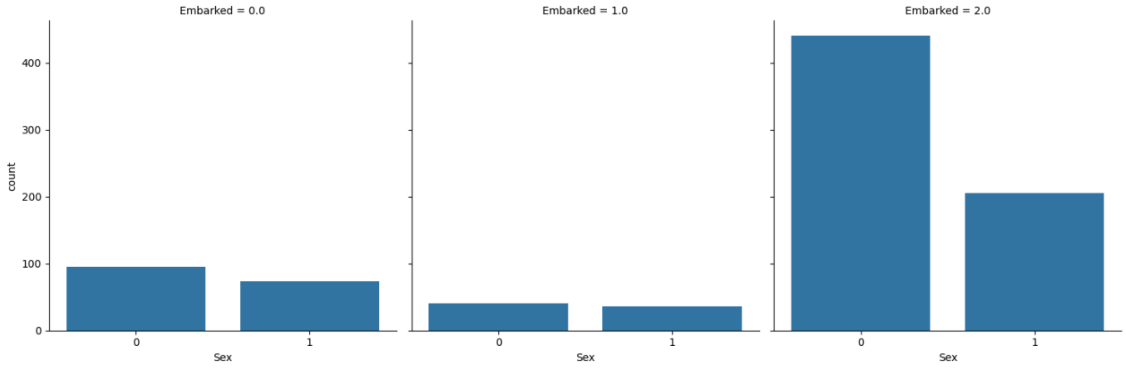
```
[59]: <seaborn.axisgrid.FacetGrid at 0x1f2e75e04c0>
```



b. Sex vs Embarked

```
[60]: sns.catplot(x = 'Sex', col = 'Embarked', data = train_data, kind = 'count')
```

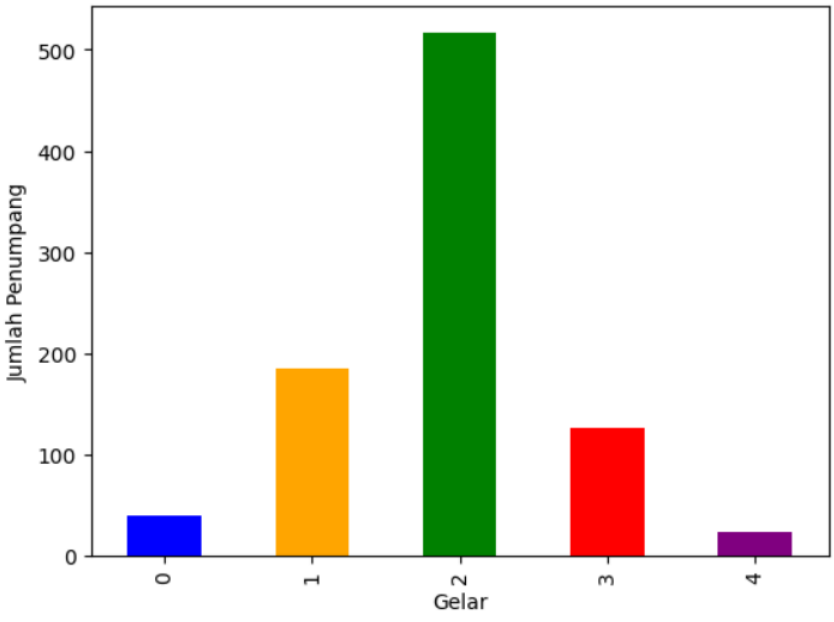
```
[60]: sns.catplot(x = 'Sex', col = 'Embarked', data = train_data, kind = 'count')

[60]: <seaborn.axisgrid.FacetGrid at 0x1f2e7962020>
```



4. Ubahlah data sebutan/gelar penumpang (Title) menjadi data angka dengan ketentuan sebagai berikut:
Master: 0, Miss: 1, Mr: 2, Mrs: 3, Others: 4

```
[62]: train_data['Title'] = train_data['Title'].replace('Master', '0')
      train_data['Title'] = train_data['Title'].replace('Miss', '1')
      train_data['Title'] = train_data['Title'].replace('Mr', '2')
      train_data['Title'] = train_data['Title'].replace('Mrs', '3')
      train_data['Title'] = train_data['Title'].replace('Others', '4')
```

```
[63]: plt = train_data['Title'].value_counts().sort_index().plot(
          kind='bar', color=['blue', 'orange', 'green', 'red', 'purple'])
      plt.set_xlabel('Gelar')
      plt.set_ylabel('Jumlah Penumpang')

[63]: Text(0, 0.5, 'Jumlah Penumpang')
```



5. Carilah nilai korelasi antar atribut termasuk atribut Title setelah diubah menjadi data angka dengan menggunakan heatmap!

```
[64]: corr_matrix = train_data.corr()

[68]: import matplotlib.pyplot as plt

      plt.figure(figsize=(9,8))
      sns.heatmap(data = corr_matrix, cmap='BrBG', annot=True, linewidths=0.2)
```