

# Modul Praktikum Data Mining



## **Tim Penyusun:**

Dr. Rakhmat Arianto, S.ST., M.Kom

Ir. Rudy Ariyanto, ST., M.Cs

Prof. Dr. Eng. Rosa Andrie Asmara, ST., MT

**Jurusan Teknologi Informasi**

**Sistem Informasi Bisnis**

**Politeknik Negeri Malang**

**Maret 2025**

## DAFTAR ISI

|                                                 |    |
|-------------------------------------------------|----|
| DAFTAR ISI .....                                | 2  |
| JOBSHEET 5 Perbaikan dan Visualisasi Data ..... | 3  |
| Pendahuluan .....                               | 3  |
| Tujuan Praktikum.....                           | 3  |
| Peralatan yang dibutuhkan .....                 | 3  |
| Praktikum .....                                 | 3  |
| Framework pada DataMining .....                 | 3  |
| Tahap 1: Mendefinisikan Permasalahan .....      | 5  |
| Tahap 2: Mengumpulkan Data .....                | 5  |
| Tahap 3: Menyiapkan Data untuk Diolah .....     | 5  |
| Tahap 4: melakukan Analisa Exploratory .....    | 11 |

## JOB SHEET 6

### Perbaikan dan Visualisasi Data

#### Pendahuluan

Modul ini menjelaskan proses perbaikan data dengan implementasi pada dataset Titanic.

#### Tujuan Praktikum

Setelah menyelesaikan praktikum ini, mahasiswa mampu:

- Memahami tentang perbaikan data.
- Membuat visualisasi data dengan baik.

#### Peralatan yang dibutuhkan

Beberapa peralatan yang dibutuhkan dalam menyelesaikan praktikum ini adalah:

- Aplikasi Microsoft Excel
- Google Colab
- Google Drive
- Koneksi Internet
- Browser Web

#### Praktikum

Framework pada Data Mining.

##### 1. 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.

##### 2. 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".

### 3. 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.

### 4. Melakukan Analisa Exploratory

Langkah penting berikutnya adalah melakukan eksplorasi 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.

### 5. 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.

### 6. Memvalidasi dan Implementasi Data Model

Setelah kita melatih model berdasarkan sebagian data, langkah berikutnya yang dilakukan adalah menguji model. Hal ini 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, kita 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.

## 7. 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

### 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 mengembangkan algoritma untuk memprediksi kemungkinan selamatnya penumpang Titanic.

**Ringkasan Dataset:** Tenggelamnya RMS Titanic adalah salah satu kecelakaan kapal paling terkenal dalam sejarah. Pada tanggal 15 April 1912, selama pelayaran perdananya, Titanic tenggelam setelah bertabrakan dengan gunung es, menewaskan 1502 dari 2224 penumpang dan awak kapal. Tragedi sensasional ini mengejutkan masyarakat internasional dan menjadi pedoman untuk penyusunan peraturan keselamatan kapal yang lebih baik.

Salah satu alasan mengapa kecelakaan kapal tersebut mengakibatkan hilangnya banyak nyawa adalah karena tidak cukupnya sekoci penyelamat bagi penumpang dan awak kapal. Meskipun ada sedikit unsur keberuntungan yang terlibat dalam upaya selamat dari tenggelamnya kapal, beberapa kelompok orang lebih mungkin selamat daripada yang lain, seperti wanita, anak-anak, dan kelas atas.

Dalam project ini, kita diminta untuk menyelesaikan analisis tentang jenis orang yang mungkin selamat.

### Tahap 2: Mengumpulkan Data

Data yang dipakai adalah data dari Kaggle pada link berikut:  
<https://www.kaggle.com/c/titanic/data>

Dataset ini terdiri dari 2 file yang dapat digunakan, File train.csv digunakan sebagai dataset referensi, file ini memiliki 12 kolom. File test.csv digunakan sebagai data uji untuk memprediksi penumpang selamat atau tidak, File ini memiliki 11 kolom (hanya kolom survived yang tidak ada dan kita diminta untuk memberikan hasil prediksi pada kolom survived baru pada file test.csv).

### Tahap 3: Menyiapkan Data untuk Diolah

Langkah pertama yang dilakukan adalah import Library yang dibutuhkan.

Langkah kedua yang dilakukan adalah memahami data secara utuh. Kita dapat gunakan fungsi `info()` dan `sample()` untuk melihat contoh data dan tipe datanya. Dari data Dictionary kita dapat lihat spesifikasi tiap data atribut. Data Dictionary dapat dilihat pada link berikut: <https://www.kaggle.com/c/titanic/data>

**Data Dictionary**

| Variable | Definition                                 | Key                                            |
|----------|--------------------------------------------|------------------------------------------------|
| survival | Survival                                   | 0 = No, 1 = Yes                                |
| pclass   | Ticket class                               | 1 = 1st, 2 = 2nd, 3 = 3rd                      |
| sex      | Sex                                        |                                                |
| Age      | Age in years                               |                                                |
| sibsp    | # of siblings / spouses aboard the Titanic |                                                |
| parch    | # of parents / children aboard the Titanic |                                                |
| ticket   | Ticket number                              |                                                |
| fare     | Passenger fare                             |                                                |
| cabin    | Cabin number                               |                                                |
| embarked | Port of Embarkation                        | C = Cherbourg, Q = Queenstown, S = Southampton |

Gambar 1. Data Dictionary Titanic Dataset

1. Atribut **Survived** adalah hasil atau variabel dependen. Data ini bertipe nominal biner 1 untuk yang selamat dan 0 untuk yang tidak selamat. Semua atribut lainnya adalah prediktor potensial atau atribut independen. Penting untuk dicatat, lebih banyak atribut prediktor tidak selalu akan membuat model menjadi lebih baik
2. Atribut **PassengerID** dan **Ticket** diasumsikan sebagai identifier unik dan acak, yang tidak memiliki dampak pada variabel hasil. Dengan demikian, kedua atribut ini akan dikecualikan dari analisis.
3. Atribut **Pclass** adalah tipe data ordinal untuk kelas tiket, menunjukkan status sosial-ekonomi, yang mewakili 1 = kelas atas, 2 = kelas menengah, dan 3 = kelas bawah.
4. Atribut **Name** adalah tipe data nominal. Atribut ini dapat digunakan dalam rekayasa fitur untuk memperoleh data jenis kelamin dari gelar, jumlah keluarga dari surname (nama keluarga), dan status ekonomi dari gelar seperti doctor atau master. Karena atribut-atribut ini sudah ada, kita akan coba menggunakannya untuk melihat apakah gelar seperti master, membuat perbedaan.

5. **Sex** dan **Embarked** adalah tipe data nominal. Keduanya akan diubah menjadi nilai angka untuk perhitungan matematika.
6. Atribut **Age** dan **Fare** adalah tipe data kuantitatif continues.
7. **SibSp** menunjukkan jumlah saudara kandung/pasangan hidup di atas kapal dan **Parch** menunjukkan jumlah orang tua/anak terkait di atas kapal. Keduanya adalah tipe data kuantitatif discrete. Data ini dapat digunakan untuk rekayasa fitur guna membuat atribut ukuran keluarga dan merupakan atribut tunggal.
8. Atribut **Cabin** adalah tipe data nominal yang dapat digunakan dalam rekayasa fitur untuk perkiraan posisi di kapal saat insiden terjadi dan status sosial ekonomi dari tingkat dek. Namun, karena ada banyak sekali nilai kosong, data ini tidak terlalu mempengaruhi model dan dapat dikecualikan dari analisis.

Setelah memahami atribut yang ada dalam data, Langkah berikutnya yang dilakukan adalah pembersihan data. Pembersihan data ini dikenal dengan istilah 4C (Correcting, Completing, Creating, dan Converting).

1. **Correcting**: Setelah meninjau data Titanic, terlihat tidak ada data yang menyimpang atau aneh. Akan tetapi, kita dapat lihat bahwa ada kemungkinan terdapat potensi data outlier dalam atribut **Age** dan **Fare**. Namun, karena nilainya masih dalam ambang wajar, kita akan lihat hingga setelah menyelesaikan analisis eksploratori untuk menentukan apakah harus mengikutkan atau menghilangkan sampel tersebut dari kumpulan data. Perlu dicatat, bahwa jika nilainya tidak wajar, misalnya usia = 800 bukannya 80, maka mungkin keputusan yang terbaik untuk memperbaikinya sekarang. Namun, kita tetap harus hati-hati saat mengubah data dari nilai aslinya, karena hal tersebut dapat sangat mempengaruhi akurasi dari model yang dibuat.
2. **Completing**: Ada nilai kosong (NULL) atau data yang hilang di atribut Age, Cabin, dan Embarked. Nilai yang kosong bisa jadi buruk karena beberapa algoritma pemodelan tidak tahu cara menangani nilai kosong dan akan gagal. Merupakan hal yang cukup penting untuk memperbaiki dataset sebelum kita mulai menggunakan model, karena kita akan membandingkan dan melakukan evaluasi beberapa model. Ada dua metode umum, menghapus sampel atau mengisi nilai yang hilang menggunakan nilai yang wajar. Tidak disarankan untuk menghapus sampel, terutama sebagian besar sampel, kecuali jika benar-benar mewakili sampel yang tidak lengkap. Sebaliknya, yang terbaik adalah mengisi nilai yang hilang. Metodologi dasar untuk data kualitatif adalah mengimputasikan menggunakan modus. Metodologi dasar untuk data kuantitatif adalah mengimputasikan menggunakan mean, median, atau mean + standard deviasi acak. Metodologi perantara adalah menggunakan metodologi dasar berdasarkan kriteria tertentu;

seperti usia rata-rata menurut kelas atau pelabuhan keberangkatan menurut tarif dan status sosial ekonomi. Ada metodologi yang lebih kompleks, namun sebelum menerapkannya, kita harus yakin metodologi tersebut telah dibandingkan dengan model dasar untuk menentukan apakah kompleksitasnya dapat menambah nilai akurasi model. Untuk kumpulan dataset ini, usia akan diperhitungkan dengan median, atribut kabin akan dihilangkan, dan keberangkatan akan diperhitungkan dengan modus. Iterasi model berikutnya dapat mengubah keputusan ini untuk menentukan apakah hal itu meningkatkan akurasi model.

3. Creating: Feature Engineering adalah saat kita akan menggunakan fitur (atribut) yang sudah ada untuk membuat fitur baru guna menentukan apakah fitur tersebut memberikan luaran baru untuk memprediksi hasil. Untuk dataset ini, kita akan membuat fitur Title untuk menentukan apakah fitur tersebut berperan dalam penentuan Survived.
4. Converting: Tidak ada format tanggal atau mata uang, hanya format Datatypes. Data kategorikal diimpor sebagai objek yang tidak mungkin diproses menggunakan perhitungan matematika.

Kita coba melihat data pada Train dan Test yang memiliki nilai NULL.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn import model_selection

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

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

data_raw = pd.read_csv('/content/drive/MyDrive/Kuliah/2025 Data Mining/Dataset/Titanic/train.csv')
data_val = pd.read_csv('/content/drive/MyDrive/Kuliah/2025 Data Mining/Dataset/Titanic/test.csv')

data1 = data_raw.copy(deep = True)

data_cleaner = [data1, data_val]

print('kolom data Train dengan nilai NULL:\n', data1.isnull().sum())
print("-"*10)

print('kolom data Test dengan nilai NULL:\n', data_val.isnull().sum())
print("-"*10)

data_raw.describe(include = 'all')
```





```
kolom data Train dengan nilai NULL:
PassengerId      0
Survived          0
Pclass            0
Name              0
Sex               0
Age              177
SibSp             0
Parch             0
Ticket            0
Fare              0
Cabin            687
Embarked          2
dtype: int64
-----
kolom data Test dengan nilai NULL:
PassengerId      0
Pclass            0
Name              0
Sex               0
Age              86
SibSp             0
Parch             0
Ticket            0
Fare              1
Cabin            327
Embarked          0
dtype: int64
-----
```

|        | PassengerId | Survived   | Pclass     | Name                             | Sex  | Age        | SibSp      | Parch      | Ticket | Fare       | Cabin      | Embarked |
|--------|-------------|------------|------------|----------------------------------|------|------------|------------|------------|--------|------------|------------|----------|
| count  | 891.000000  | 891.000000 | 891.000000 | 891                              | 891  | 714.000000 | 891.000000 | 891.000000 | 891    | 891.000000 | 204        | 889      |
| unique | NaN         | NaN        | NaN        | 891                              | 2    | NaN        | NaN        | NaN        | 681    | NaN        | 147        | 3        |
| top    | NaN         | NaN        | NaN        | Braund,<br>Mr.<br>Owen<br>Harris | male | NaN        | NaN        | NaN        | 347082 | NaN        | B96<br>B98 | S        |
| freq   | NaN         | NaN        | NaN        | 1                                | 577  | NaN        | NaN        | NaN        | 7      | NaN        | 4          | 644      |
| mean   | 446.000000  | 0.383838   | 2.308642   | NaN                              | NaN  | 29.699118  | 0.523008   | 0.381594   | NaN    | 32.204208  | NaN        | NaN      |
| std    | 257.353842  | 0.486592   | 0.836071   | NaN                              | NaN  | 14.526497  | 1.102743   | 0.806057   | NaN    | 49.693429  | NaN        | NaN      |
| min    | 1.000000    | 0.000000   | 1.000000   | NaN                              | NaN  | 0.420000   | 0.000000   | 0.000000   | NaN    | 0.000000   | NaN        | NaN      |
| 25%    | 223.500000  | 0.000000   | 2.000000   | NaN                              | NaN  | 20.125000  | 0.000000   | 0.000000   | NaN    | 7.910400   | NaN        | NaN      |
| 50%    | 446.000000  | 0.000000   | 3.000000   | NaN                              | NaN  | 28.000000  | 0.000000   | 0.000000   | NaN    | 14.454200  | NaN        | NaN      |
| 75%    | 668.500000  | 1.000000   | 3.000000   | NaN                              | NaN  | 38.000000  | 1.000000   | 0.000000   | NaN    | 31.000000  | NaN        | NaN      |
| max    | 891.000000  | 1.000000   | 3.000000   | NaN                              | NaN  | 80.000000  | 8.000000   | 6.000000   | NaN    | 512.329200 | NaN        | NaN      |

Gambar 2. Kode dan hasil dari data bernilai NULL

## Percobaan yang saya lakukan

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn import model_selection

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

data_raw = pd.read_csv('/content/drive/MyDrive/Data Mining/titanic/train.csv')
data_val = pd.read_csv('/content/drive/MyDrive/Data Mining/titanic/test.csv')

data1 = data_raw.copy(deep = True)

data_cleaner = [data1, data_val]

print('Kolom data Train dengan nilai NULL:\n', data1.isnull().sum())
print("-"*10)

print('Kolom data Test dengan nilai NULL:\n', data_val.isnull().sum())
print("-"*10)

data_raw.describe(include = 'all')
```

```
Kolom data Train dengan nilai NULL:
 PassengerId      0
 Survived         0
 Pclass          0
 Name            0
 Sex             0
 Age            177
 SibSp           0
 Parch           0
 Ticket          0
 Fare            0
 Cabin          687
 Embarked        2
dtype: int64
-----
Kolom data Test dengan nilai NULL:
 PassengerId      0
 Pclass          0
 Name            0
 Sex             0
 Age             86
 SibSp           0
 Parch           0
 Ticket          0
 Fare            1
 Cabin          327
 Embarked        0
dtype: int64
-----
```

|        | PassengerId | Survived   | Pclass     | Name                    | Sex  | Age        | SibSp      | Parch      | Ticket | Fare       | Cabin   | Embarked |
|--------|-------------|------------|------------|-------------------------|------|------------|------------|------------|--------|------------|---------|----------|
| count  | 891.000000  | 891.000000 | 891.000000 | 891                     | 891  | 714.000000 | 891.000000 | 891.000000 | 891    | 891.000000 | 204     | 889      |
| unique | NaN         | NaN        | NaN        | 891                     | 2    | NaN        | NaN        | NaN        | 681    | NaN        | 147     | 3        |
| top    | NaN         | NaN        | NaN        | Braund, Mr. Owen Harris | male | NaN        | NaN        | NaN        | 347082 | NaN        | B96 B98 | S        |
| freq   | NaN         | NaN        | NaN        | 1                       | 577  | NaN        | NaN        | NaN        | 7      | NaN        | 4       | 644      |
| mean   | 446.000000  | 0.383838   | 2.308642   | NaN                     | NaN  | 29.699118  | 0.523008   | 0.381594   | NaN    | 32.204208  | NaN     | NaN      |
| std    | 257.353842  | 0.486592   | 0.836071   | NaN                     | NaN  | 14.526497  | 1.102743   | 0.806057   | NaN    | 49.693429  | NaN     | NaN      |
| min    | 1.000000    | 0.000000   | 1.000000   | NaN                     | NaN  | 0.420000   | 0.000000   | 0.000000   | NaN    | 0.000000   | NaN     | NaN      |
| 25%    | 223.500000  | 0.000000   | 2.000000   | NaN                     | NaN  | 20.125000  | 0.000000   | 0.000000   | NaN    | 7.910400   | NaN     | NaN      |
| 50%    | 446.000000  | 0.000000   | 3.000000   | NaN                     | NaN  | 28.000000  | 0.000000   | 0.000000   | NaN    | 14.454200  | NaN     | NaN      |
| 75%    | 668.500000  | 1.000000   | 3.000000   | NaN                     | NaN  | 38.000000  | 1.000000   | 0.000000   | NaN    | 31.000000  | NaN     | NaN      |
| max    | 891.000000  | 1.000000   | 3.000000   | NaN                     | NaN  | 80.000000  | 8.000000   | 6.000000   | NaN    | 512.329200 | NaN     | NaN      |

Langkah berikutnya adalah melakukan Completing pada nilai NULL.

```
for dataset in data_cleaner:
    #isikan age yang kosong dengan median
    dataset['Age'].fillna(dataset['Age'].median(), inplace = True)

    #isikan embarked dengan modus
    dataset['Embarked'].fillna(dataset['Embarked'].mode()[0], inplace = True)

    #isikan fare dengan median
    dataset['Fare'].fillna(dataset['Fare'].median(), inplace = True)

#hapus atribut cabin dan lainnya yang kita anggap tidak diperlukan
drop_column = ['PassengerId', 'Cabin', 'Ticket']
data1.drop(drop_column, axis=1, inplace = True)

print(data1.isnull().sum())
print("-"*10)
print(data_val.isnull().sum())
```

```
Survived      0
Pclass        0
Name          0
Sex           0
Age           0
SibSp         0
Parch         0
Fare          0
Embarked      0
dtype: int64
-----
PassengerId   0
Pclass        0
Name          0
Sex           0
Age           0
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin         327
Embarked      0
```

Gambar 3. Kode dan hasil setelah dilakukan perbaikan data

### Percobaan yang saya lakukan

```
for dataset in data_cleaner:
    # isikan age yang kosong dengan median
    dataset['Age'].fillna(dataset['Age'].median(), inplace = True)

    # isikan embarked dengan modus
    dataset['Embarked'].fillna(dataset['Embarked'].mode()[0], inplace = True)

    # isikan fare dengan median
    dataset['Fare'].fillna(dataset['Fare'].median(), inplace = True)

# hapus atribut cabin dan lainnya yang kita anggap tidak diperlukan
drop_column = ['PassengerId', 'Cabin', 'Ticket']
data1.drop(drop_column, axis=1, inplace = True)

print(data1.isnull().sum())
print("-"*10)
print(data_val.isnull().sum())
```

```
Survived      0
Pclass        0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Fare           0
Embarked       0
dtype: int64
-----
PassengerId    0
Pclass          0
Name            0
Sex             0
Age             0
SibSp           0
Parch           0
Ticket         0
Fare            0
Cabin          327
Embarked        0
```

Langkah berikutnya adalah membuat atribut baru, contoh akan dibuat atribut Title dan Jumlah keluarga, Fare dan Age akan di buat bentuk kategorikal.

```
for dataset in data_cleaner:
    #variables discrete
    dataset['FamilySize'] = dataset ['SibSp'] + dataset['Parch'] + 1

    dataset['IsAlone'] = 1 #diisikan 1 jika sendiri
    dataset['IsAlone'].loc[dataset['FamilySize'] > 1] = 0 # isikan 0 jika ada relasi yang bersama

    #Pisahkan title dari nama: http://www.pythonforbeginners.com/dictionary/python-split
    dataset['Title'] = dataset['Name'].str.split(" ", expand=True)[1].str.split(".", expand=True)[0]

    #Buat bin Fare menggunakan qcut atau bin Frekuensi:
    dataset['FareBin'] = pd.qcut(dataset['Fare'], 4)

    #buat bin Age menggunakan cut or nilai bin:
    dataset['AgeBin'] = pd.cut(dataset['Age'].astype(int), 5)

#bersihkan title yang jarang ada
stat_min = 10
title_names = (data1['Title'].value_counts() < stat_min)

data1['Title'] = data1['Title'].apply(lambda x: 'Misc' if title_names.loc[x] == True else x)
print(data1['Title'].value_counts())
print("-"*10)

#tampilkan data lagi
data1.info()
data_val.info()
data1.sample(10)
```

| #  | Column     | Non-Null Count | Dtype    |
|----|------------|----------------|----------|
| 0  | Survived   | 891 non-null   | int64    |
| 1  | Pclass     | 891 non-null   | int64    |
| 2  | Name       | 891 non-null   | object   |
| 3  | Sex        | 891 non-null   | object   |
| 4  | Age        | 891 non-null   | float64  |
| 5  | SibSp      | 891 non-null   | int64    |
| 6  | Parch      | 891 non-null   | int64    |
| 7  | Fare       | 891 non-null   | float64  |
| 8  | Embarked   | 891 non-null   | object   |
| 9  | FamilySize | 891 non-null   | int64    |
| 10 | IsAlone    | 891 non-null   | int64    |
| 11 | Title      | 891 non-null   | object   |
| 12 | FareBin    | 891 non-null   | category |
| 13 | AgeBin     | 891 non-null   | category |

dtypes: category(2), float64(2), int64(6), object(4)  
memory usage: 85.9+ KB

| #  | Column      | Non-Null Count | Dtype    |
|----|-------------|----------------|----------|
| 0  | PassengerId | 418 non-null   | int64    |
| 1  | Pclass      | 418 non-null   | int64    |
| 2  | Name        | 418 non-null   | object   |
| 3  | Sex         | 418 non-null   | object   |
| 4  | Age         | 418 non-null   | float64  |
| 5  | SibSp       | 418 non-null   | int64    |
| 6  | Parch       | 418 non-null   | int64    |
| 7  | Ticket      | 418 non-null   | object   |
| 8  | Fare        | 418 non-null   | float64  |
| 9  | Cabin       | 91 non-null    | object   |
| 10 | Embarked    | 418 non-null   | object   |
| 11 | FamilySize  | 418 non-null   | int64    |
| 12 | IsAlone     | 418 non-null   | int64    |
| 13 | Title       | 418 non-null   | object   |
| 14 | FareBin     | 418 non-null   | category |
| 15 | AgeBin      | 418 non-null   | category |

dtypes: category(2), float64(2), int64(6), object(6)  
memory usage: 47.1+ KB

Gambar 4. Kode dan hasil dari melakukan feature engineering yaitu membuat fitur baru



### Percobaan yang saya lakukan

```
for dataset in data_cleaner:
    #variables discrete
    dataset['FamilySize'] = dataset['SibSp'] + dataset['Parch'] + 1

    dataset['IsAlone'] = 1 #diisikan 1 jika sendiri
    dataset['IsAlone'].loc[dataset['FamilySize'] > 1] = 0 # isikan 0 jika ada relasi yang bersama

    #Pisahkan title dari nama: http://www.pythonforbeginners.com/dictionary/python-split
    dataset['Title'] = dataset['Name'].str.split(".", expand=True)[1].str.split(".", expand=True)[0]

    #Buat bin Fare menggunakan qcut atau bin Frekuensi:
    dataset['FareBin'] = pd.qcut(dataset['Fare'], 4)

    #buat bin Age menggunakan cut atau nilai bin:
    dataset['AgeBin'] = pd.cut(dataset['Age'].astype(int), 5)

    #bersihkan title yang jarang ada
    stat_min = 10
    title_names = (data1['Title'].value_counts() < stat_min)

    data1['Title'] = data1['Title'].apply(lambda x: 'Misc' if title_names.loc[x] == True else x)
    print(data1['Title'].value_counts())
    print("-"*10)

    #tampilkan data lagi
    data1.info()
    data_val.info()
    data1.sample(10)
```

```
Title
Mr      517
Miss    182
Mrs     125
Master   40
Misc     27
Name: count, dtype: int64
-----
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Survived    891 non-null   int64
1   Pclass      891 non-null   int64
2   Name        891 non-null   object
3   Sex         891 non-null   object
4   Age         891 non-null   float64
5   SibSp       891 non-null   int64
6   Parch       891 non-null   int64
7   Fare        891 non-null   float64
8   Embarked    891 non-null   object
9   FamilySize  891 non-null   int64
10  IsAlone     891 non-null   int64
11  Title       891 non-null   object
12  FareBin     891 non-null   category
13  AgeBin      891 non-null   category
dtypes: category(2), float64(2), int64(6), object(4)
memory usage: 85.9+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 16 columns):
#   Column      Non-Null Count  Dtype
---  -
0   PassengerId  418 non-null   int64
1   Pclass      418 non-null   int64
2   Name        418 non-null   object
3   Sex         418 non-null   object
4   Age         418 non-null   float64
5   SibSp       418 non-null   int64
6   Parch       418 non-null   int64
7   Ticket      418 non-null   object
8   Fare        418 non-null   float64
9   Cabin       91 non-null    object
10  Embarked    418 non-null   object
11  FamilySize  418 non-null   int64
12  IsAlone     418 non-null   int64
13  Title       418 non-null   object
14  FareBin     418 non-null   category
15  AgeBin      418 non-null   category
dtypes: category(2), float64(2), int64(6), object(6)
memory usage: 47.1+ KB
```

#### Tahap 4: melakukan Analisa Exploratory

Setelah melakukan pembersihan data, Langkah berikutnya yang dapat dilakukan adalah melakukan eksplorasi data secara statistik deskriptif dan grafis dan menyimpulkan kaitan antara variabel independen dan dependen. Kode berikut digunakan untuk melihat statistik deskriptif korelasi antara variabel independen dengan dependen.

```
#Discrete Variable Correlation dengan Survival menggunakan
#group by / pivot table: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.groupby.html
for x in data1_x:
    if data1[x].dtype != 'float64':
        print('Korelasi Survival dengan:', x)
        print(data1[[x, Target[0]]].groupby(x, as_index=False).mean())
        print('-'*10, '\n')

#menggunakan crosstabs: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.crosstab.html
print(pd.crosstab(data1['Title'], data1[Target[0]]))
```

| Korelasi Survival dengan: Sex |          |  |
|-------------------------------|----------|--|
| Sex                           | Survived |  |
| 0 female                      | 0.742038 |  |
| 1 male                        | 0.188908 |  |

| Korelasi Survival dengan: Pclass |          |          |
|----------------------------------|----------|----------|
| Pclass                           | Survived |          |
| 0                                | 1        | 0.629630 |
| 1                                | 2        | 0.472826 |
| 2                                | 3        | 0.242363 |

| Korelasi Survival dengan: Embarked |          |  |
|------------------------------------|----------|--|
| Embarked                           | Survived |  |
| 0 C                                | 0.553571 |  |
| 1 Q                                | 0.389610 |  |
| 2 S                                | 0.339009 |  |

| Korelasi Survival dengan: Title |          |  |
|---------------------------------|----------|--|
| Title                           | Survived |  |
| 0 Master                        | 0.575000 |  |
| 1 Misc                          | 0.444444 |  |
| 2 Miss                          | 0.697802 |  |
| 3 Mr                            | 0.156673 |  |
| 4 Mrs                           | 0.792000 |  |

| Korelasi Survival dengan: SibSp |          |          |
|---------------------------------|----------|----------|
| SibSp                           | Survived |          |
| 0                               | 0        | 0.345395 |
| 1                               | 1        | 0.535885 |
| 2                               | 2        | 0.464286 |
| 3                               | 3        | 0.250000 |
| 4                               | 4        | 0.166667 |
| 5                               | 5        | 0.000000 |
| 6                               | 8        | 0.000000 |

| Korelasi Survival dengan: Parch |          |          |
|---------------------------------|----------|----------|
| Parch                           | Survived |          |
| 0                               | 0        | 0.343658 |
| 1                               | 1        | 0.550847 |
| 2                               | 2        | 0.500000 |
| 3                               | 3        | 0.600000 |
| 4                               | 4        | 0.000000 |
| 5                               | 5        | 0.200000 |
| 6                               | 6        | 0.000000 |

| Korelasi Survival dengan: FamilySize |          |          |
|--------------------------------------|----------|----------|
| FamilySize                           | Survived |          |
| 0                                    | 1        | 0.303538 |
| 1                                    | 2        | 0.552795 |
| 2                                    | 3        | 0.578431 |
| 3                                    | 4        | 0.724138 |
| 4                                    | 5        | 0.200000 |
| 5                                    | 6        | 0.136364 |
| 6                                    | 7        | 0.333333 |
| 7                                    | 8        | 0.000000 |
| 8                                    | 11       | 0.000000 |

| Korelasi Survival dengan: IsAlone |          |          |
|-----------------------------------|----------|----------|
| IsAlone                           | Survived |          |
| 0                                 | 0        | 0.505650 |
| 1                                 | 1        | 0.303538 |

| Survived | 0   | 1   |
|----------|-----|-----|
| Title    |     |     |
| Master   | 17  | 23  |
| Misc     | 15  | 12  |
| Miss     | 55  | 127 |
| Mr       | 436 | 81  |
| Mrs      | 26  | 99  |

Gambar 5. Kode dan hasil dari Analisa deskriptif korelasi antara Survived dengan fitur lain

### Percobaan yang saya lakukan

```
Target = ['Survived']
data1_x = data1.columns.drop(Target)

for x in data1_x:
    if data1[x].dtype != 'float64':
        print('Korelasi Survival dengan:', x)
        print(data1[[x, Target[0]]].groupby(x, as_index=False).mean())
        print('-'*10, '\n')

print(pd.crosstab(data1['Title'], data1[Target[0]]))
```

Korelasi Survival dengan: Pclass

| Pclass | Survived   |
|--------|------------|
| 0      | 1 0.629630 |
| 1      | 2 0.472826 |
| 2      | 3 0.242363 |

Korelasi Survival dengan: Sex

| Sex      | Survived |
|----------|----------|
| 0 female | 0.742038 |
| 1 male   | 0.188908 |

Korelasi Survival dengan: SibSp

| SibSp | Survived   |
|-------|------------|
| 0     | 0 0.345395 |
| 1     | 1 0.535885 |
| 2     | 2 0.464286 |
| 3     | 3 0.250000 |
| 4     | 4 0.166667 |
| 5     | 5 0.000000 |
| 6     | 8 0.000000 |

Korelasi Survival dengan: Parch

| Parch | Survived   |
|-------|------------|
| 0     | 0 0.343658 |
| 1     | 1 0.550847 |
| 2     | 2 0.500000 |
| 3     | 3 0.600000 |
| 4     | 4 0.000000 |
| 5     | 5 0.200000 |
| 6     | 6 0.000000 |

Korelasi Survival dengan: Embarked

| Embarked | Survived |
|----------|----------|
| 0 C      | 0.553571 |
| 1 Q      | 0.389610 |
| 2 S      | 0.339009 |

Korelasi Survival dengan: Name

| Name                                  | Survived |
|---------------------------------------|----------|
| Abbing, Mr. Anthony                   | 0.0      |
| Abbott, Mr. Rossmore Edward           | 0.0      |
| Abbott, Mrs. Stanton (Rosa Hunt)      | 1.0      |
| Abelson, Mr. Samuel                   | 0.0      |
| Abelson, Mrs. Samuel (Hannah Wizosky) | 1.0      |
| ...                                   | ...      |
| de Mulder, Mr. Theodore               | 1.0      |
| de Pelsmaecker, Mr. Alfons            | 0.0      |
| del Carlo, Mr. Sebastiano             | 0.0      |
| van Billiard, Mr. Austin Blyler       | 0.0      |
| van Melkebeke, Mr. Philemon           | 0.0      |

[891 rows x 2 columns]

Korelasi Survival dengan: Embarked

| Embarked | Survived |
|----------|----------|
| 0 C      | 0.553571 |
| 1 Q      | 0.389610 |
| 2 S      | 0.339009 |

Korelasi Survival dengan: FamilySize

| FamilySize | Survived    |
|------------|-------------|
| 0          | 1 0.303538  |
| 1          | 2 0.552795  |
| 2          | 3 0.578431  |
| 3          | 4 0.724138  |
| 4          | 5 0.200000  |
| 5          | 6 0.136364  |
| 6          | 7 0.333333  |
| 7          | 8 0.000000  |
| 8          | 11 0.000000 |

Korelasi Survival dengan: IsAlone

| IsAlone | Survived   |
|---------|------------|
| 0       | 0 0.505650 |
| 1       | 1 0.303538 |

Korelasi Survival dengan: Title

| Title    | Survived |
|----------|----------|
| 0 Master | 0.575000 |
| 1 Misc   | 0.444444 |
| 2 Miss   | 0.697802 |
| 3 Mr     | 0.156673 |
| 4 Mrs    | 0.792000 |

Korelasi Survival dengan: FareBin

| FareBin           | Survived |
|-------------------|----------|
| 0 (-0.001, 7.91]  | 0.197309 |
| 1 (7.91, 14.454]  | 0.303571 |
| 2 (14.454, 31.0]  | 0.454955 |
| 3 (31.0, 512.329] | 0.581081 |

Korelasi Survival dengan: AgeBin

| AgeBin          | Survived |
|-----------------|----------|
| 0 (-0.08, 16.0] | 0.550000 |
| 1 (16.0, 32.0]  | 0.344762 |
| 2 (32.0, 48.0]  | 0.403226 |
| 3 (48.0, 64.0]  | 0.434783 |
| 4 (64.0, 80.0]  | 0.090909 |

| Survived | 0   | 1   |
|----------|-----|-----|
| Title    |     |     |
| Master   | 17  | 23  |
| Misc     | 15  | 12  |
| Miss     | 55  | 127 |
| Mr       | 436 | 81  |
| Mrs      | 26  | 99  |



Kode berikut akan menunjukkan tampilan grafis dalam berbagai bentuk. Perhatikan bentuk grafiknya dan coba pahami arti dari grafik-grafik tersebut.

```
plt.figure(figsize=[16,12])
plt.subplot(231)
plt.boxplot(x=data1['Fare'], showmeans = True, meanline = True)
plt.title('Fare Boxplot')
plt.ylabel('Fare ($)')

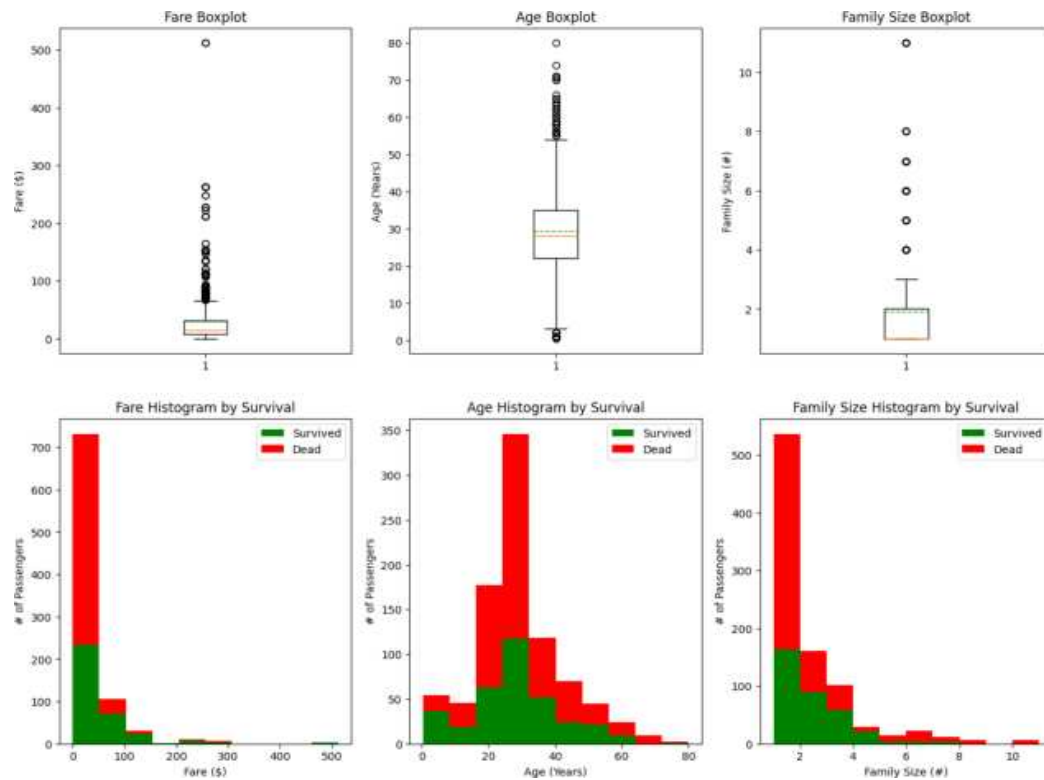
plt.subplot(232)
plt.boxplot(data1['Age'], showmeans = True, meanline = True)
plt.title('Age Boxplot')
plt.ylabel('Age (Years)')

plt.subplot(233)
plt.boxplot(data1['FamilySize'], showmeans = True, meanline = True)
plt.title('Family Size Boxplot')
plt.ylabel('Family Size (#)')

plt.subplot(234)
plt.hist(x = [data1[data1['Survived']==1]['Fare'], data1[data1['Survived']==0]['Fare']],
        stacked=True, color = ['g','r'],label = ['Survived','Dead'])
plt.title('Fare Histogram by Survival')
plt.xlabel('Fare ($)')
plt.ylabel('# of Passengers')
plt.legend()

plt.subplot(235)
plt.hist(x = [data1[data1['Survived']==1]['Age'], data1[data1['Survived']==0]['Age']],
        stacked=True, color = ['g','r'],label = ['Survived','Dead'])
plt.title('Age Histogram by Survival')
plt.xlabel('Age (Years)')
plt.ylabel('# of Passengers')
plt.legend()

plt.subplot(236)
plt.hist(x = [data1[data1['Survived']==1]['FamilySize'], data1[data1['Survived']==0]['FamilySize']],
        stacked=True, color = ['g','r'],label = ['Survived','Dead'])
plt.title('Family Size Histogram by Survival')
plt.xlabel('Family Size (#)')
plt.ylabel('# of Passengers')
plt.legend()
```



Gambar 6. Kode dan Grafik hasil perbandingan dalam boxplot dan histogram dengan matplotlib

*Percobaan yang saya lakukan*

```
plt.figure(figsize=[16,12])

plt.subplot(231)
plt.boxplot(x=data1['Fare'], showmeans = True, meanline = True)
plt.title('Fare Boxplot')
plt.ylabel('Fare ($)')

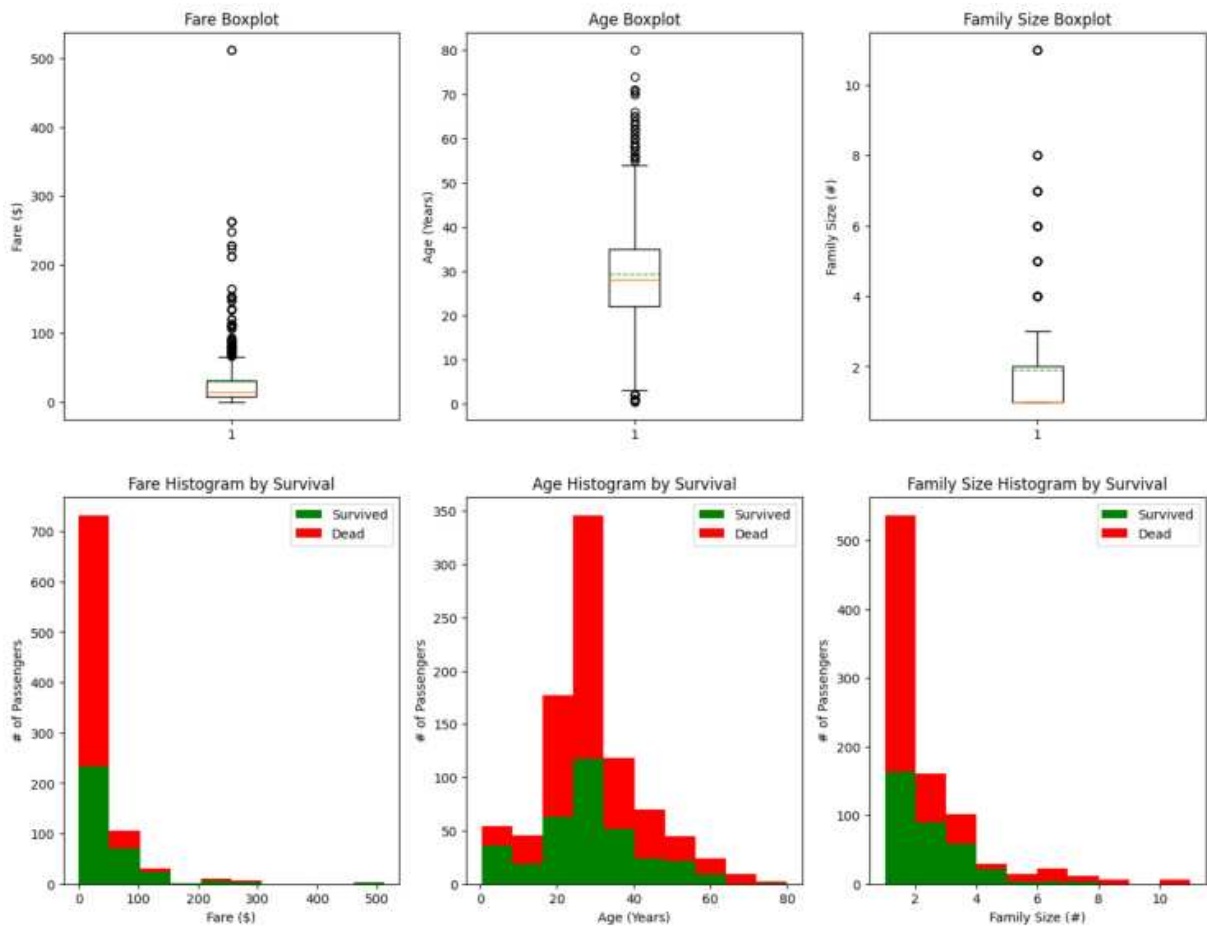
plt.subplot(232)
plt.boxplot(data1['Age'], showmeans = True, meanline = True)
plt.title('Age Boxplot')
plt.ylabel('Age (Years)')

plt.subplot(233)
plt.boxplot(data1['FamilySize'], showmeans = True, meanline = True)
plt.title('Family Size Boxplot')
plt.ylabel('Family Size (#)')

plt.subplot(234)
plt.hist(x = [data1[data1['Survived']==1]['Fare'], data1[data1['Survived']==0]['Fare']],
         stacked=True, color = ['g','r'], label = ['Survived', 'Dead'])
plt.title('Fare Histogram by Survival')
plt.xlabel('Fare ($)')
plt.ylabel('# of Passengers')
plt.legend()

plt.subplot(235)
plt.hist(x = [data1[data1['Survived']==1]['Age'], data1[data1['Survived']==0]['Age']],
         stacked=True, color = ['g','r'], label = ['Survived', 'Dead'])
plt.title('Age Histogram by Survival')
plt.xlabel('Age (Years)')
plt.ylabel('# of Passengers')
plt.legend()

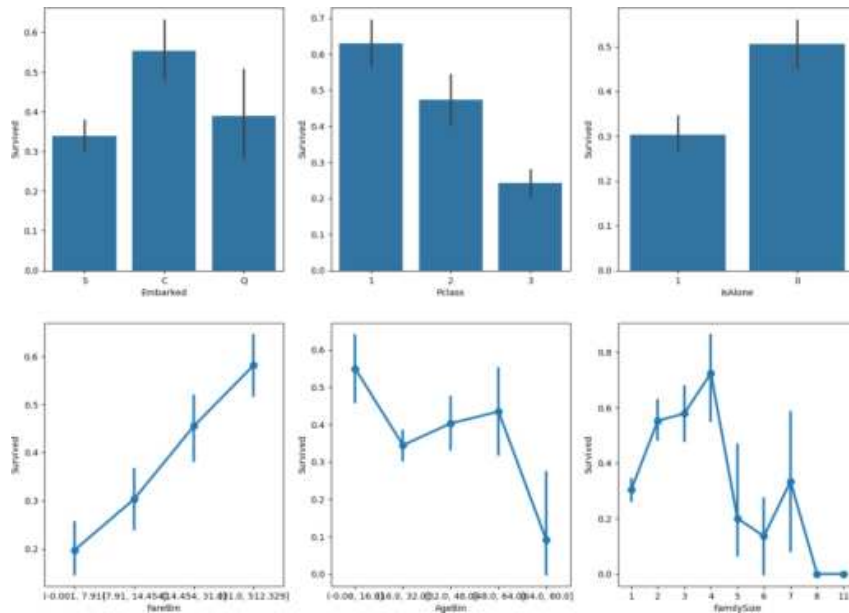
plt.subplot(236)
plt.hist(x = [data1[data1['Survived']==1]['FamilySize'], data1[data1['Survived']==0]['FamilySize']],
         stacked=True, color = ['g','r'], label = ['Survived', 'Dead'])
plt.title('Family Size Histogram by Survival')
plt.xlabel('Family Size (#)')
plt.ylabel('# of Passengers')
plt.legend()
```



```
#kita gunakan seaborn untuk perbandingan antar variabel: https://seaborn.pydata.org/api.html
#graph individual features dengan survival
fig, saxis = plt.subplots(2, 3, figsize=(16,12))

sns.barplot(x = 'Embarked', y = 'Survived', data=data1, ax = saxis[0,0])
sns.barplot(x = 'Pclass', y = 'Survived', order=[1,2,3], data=data1, ax = saxis[0,1])
sns.barplot(x = 'IsAlone', y = 'Survived', order=[1,0], data=data1, ax = saxis[0,2])

sns.pointplot(x = 'FareBin', y = 'Survived', data=data1, ax = saxis[1,0])
sns.pointplot(x = 'AgeBin', y = 'Survived', data=data1, ax = saxis[1,1])
sns.pointplot(x = 'FamilySize', y = 'Survived', data=data1, ax = saxis[1,2])
```



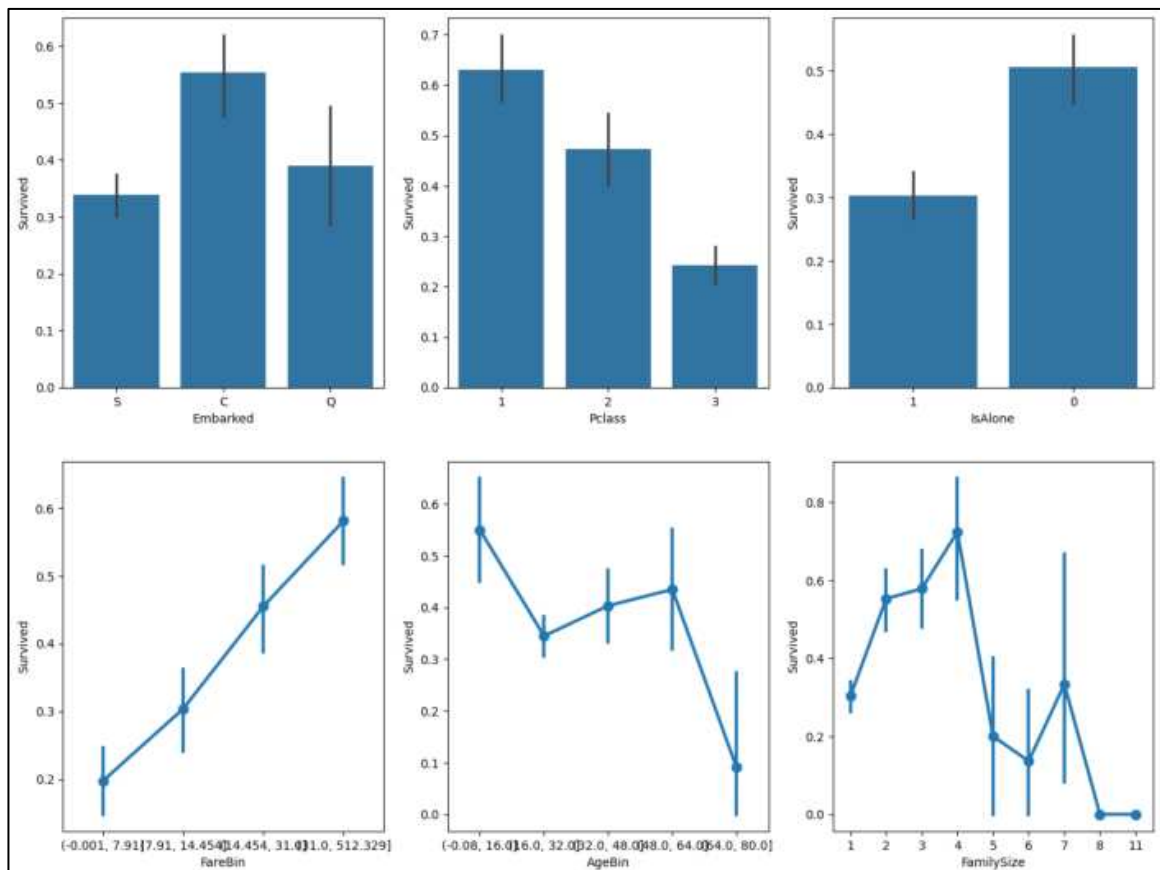
Gambar 7. Kode dan Grafik hasil perbandingan dalam boxplot dan pointplot dengan seaborn

### Percobaan yang saya lakukan

```
fig, saxis = plt.subplots(2, 3, figsize=(16,12))

sns.barplot(x = 'Embarked', y = 'Survived', data=data1, ax = saxis[0,0])
sns.barplot(x = 'Pclass', y = 'Survived', order=[1,2,3], data=data1, ax = saxis[0,1])
sns.barplot(x = 'IsAlone', y = 'Survived', order=[1,0], data=data1, ax = saxis[0,2])

sns.pointplot(x = 'FareBin', y = 'Survived', data=data1, ax = saxis[1,0])
sns.pointplot(x = 'AgeBin', y = 'Survived', data=data1, ax = saxis[1,1])
sns.pointplot(x = 'FamilySize', y = 'Survived', data=data1, ax = saxis[1,2])
```



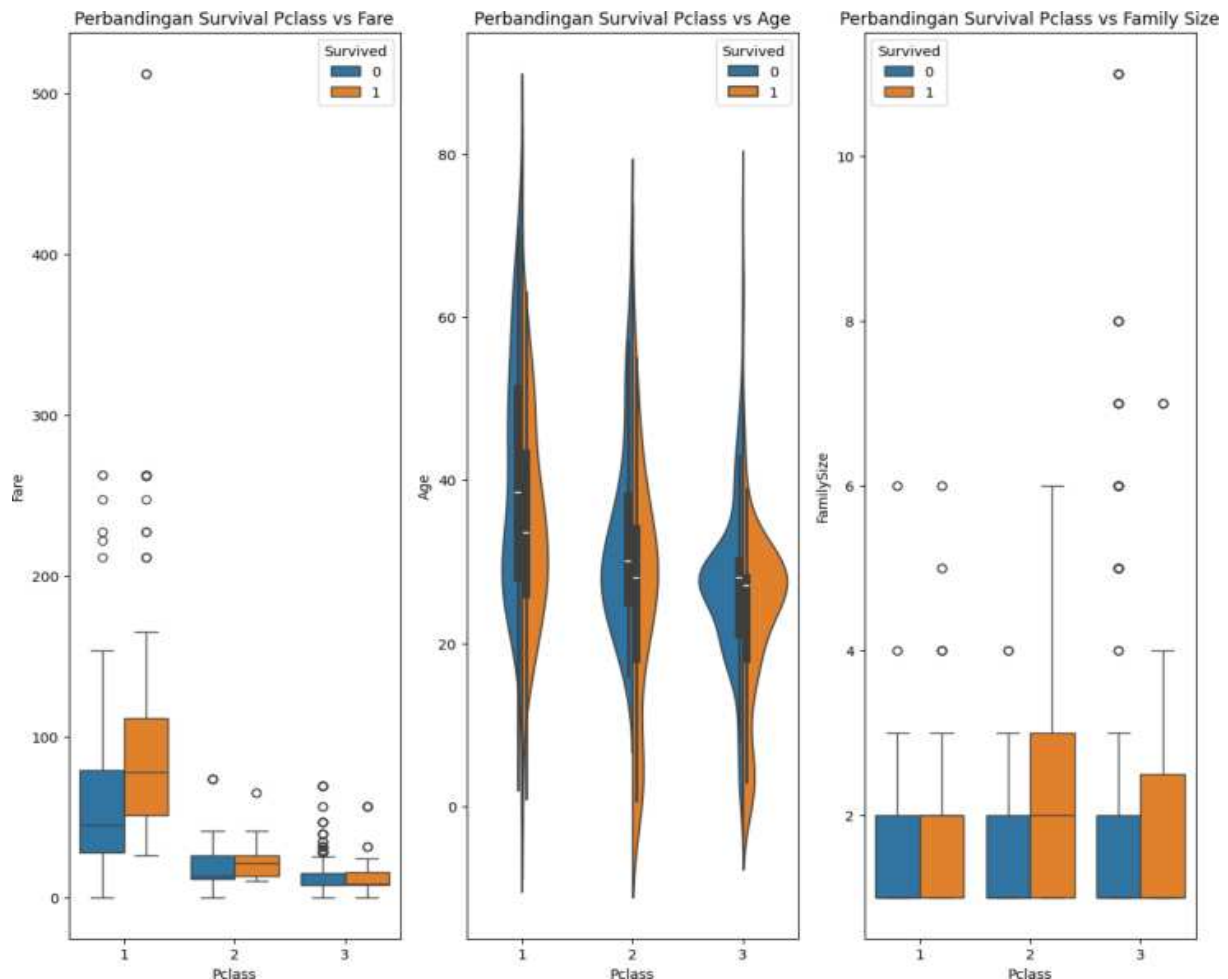


```
#grafik distribusi dari data kualitatif: Pclass
#Pclass cukup tinggi korelasinya dengan survival, berikut akan ditunjukkan perbandingan Pclass dengan bbrp fitur lain
fig, (axis1,axis2,axis3) = plt.subplots(1,3,figsize=(14,12))

sns.boxplot(x = 'Pclass', y = 'Fare', hue = 'Survived', data = data1, ax = axis1)
axis1.set_title('Perbandingan Survival Pclass vs Fare')

sns.violinplot(x = 'Pclass', y = 'Age', hue = 'Survived', data = data1, split = True, ax = axis2)
axis2.set_title('Perbandingan Survival Pclass vs Age')

sns.boxplot(x = 'Pclass', y = 'FamilySize', hue = 'Survived', data = data1, ax = axis3)
axis3.set_title('Perbandingan Survival Pclass vs Family Size')
```



Gambar 8. Kode dan Grafik hasil perbandingan dalam boxplot dan violinplot dengan seaborn

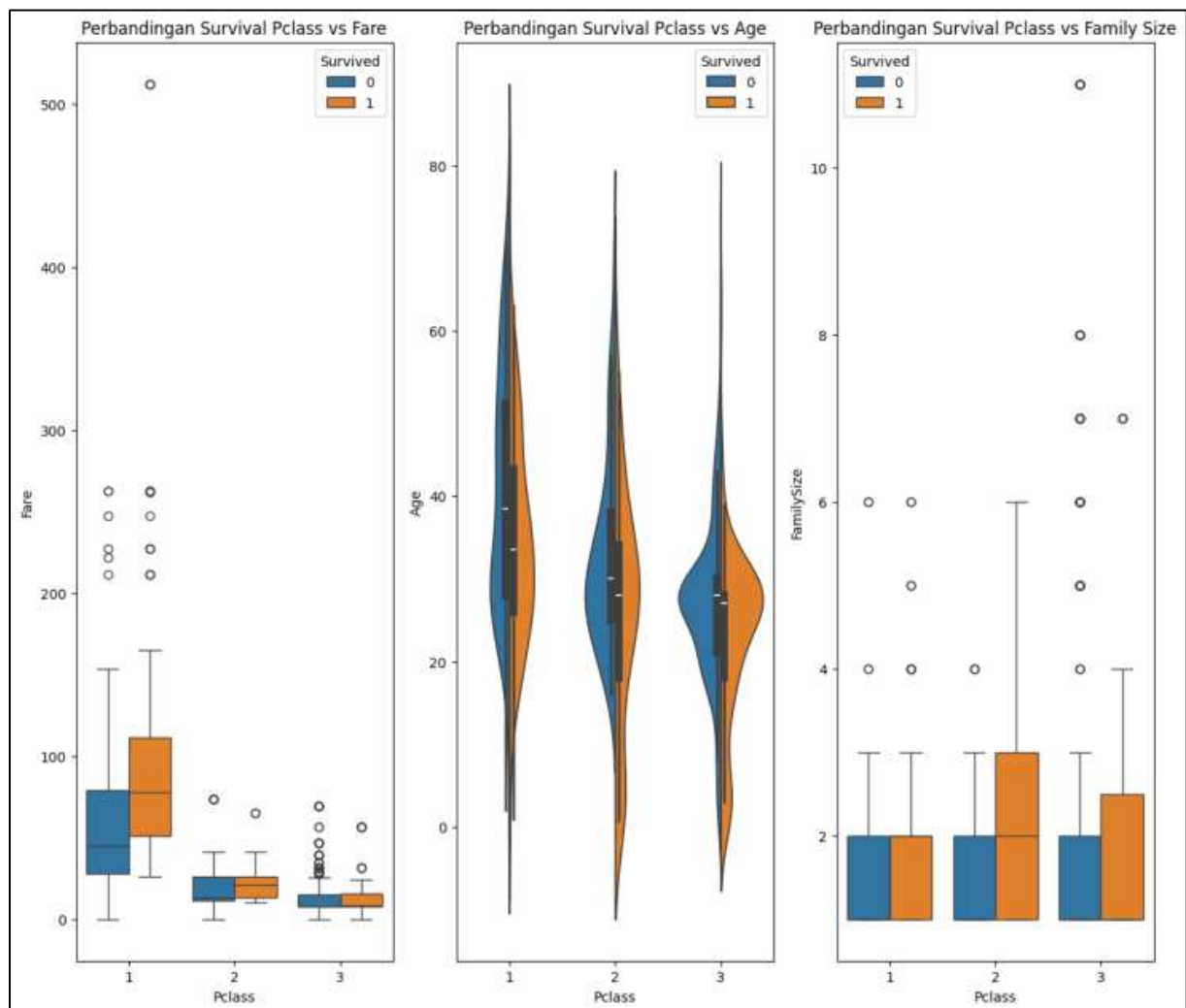
### Percobaan yang saya lakukan

```
fig, (axis1, axis2, axis3) = plt.subplots(1,3,figsize=(14,12))

sns.boxplot(x = 'Pclass', y = 'Fare', hue = 'Survived', data = data1, ax = axis1)
axis1.set_title('Perbandingan Survival Pclass vs Fare')

sns.violinplot(x = 'Pclass', y = 'Age', hue = 'Survived', data = data1, split = True, ax = axis2)
axis2.set_title('Perbandingan Survival Pclass vs Age')

sns.boxplot(x = 'Pclass', y = 'FamilySize', hue = 'Survived', data = data1, ax = axis3)
axis3.set_title('Perbandingan Survival Pclass vs Family Size')
```

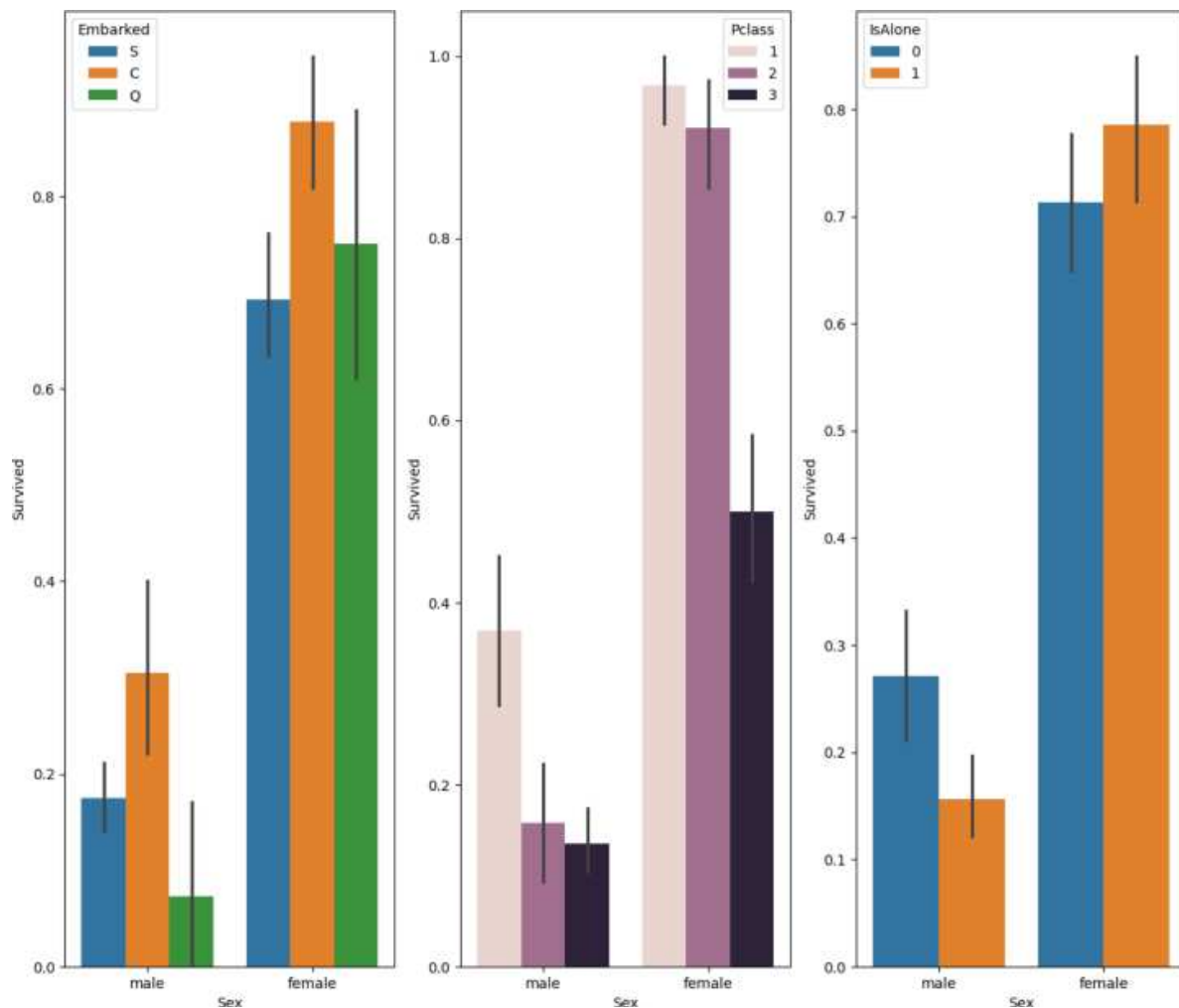


```
#grafik distribusi dari data kualitatif: Sex
#Sex cukup tinggi korelasinya dengan survival, berikut akan ditunjukkan perbandingan Sex dengan bbrp fitur lain
fig, qaxis = plt.subplots(1,3,figsize=(14,12))

sns.barplot(x = 'Sex', y = 'Survived', hue = 'Embarked', data=data1, ax = qaxis[0])
axis1.set_title('Perbandingan Survival Sex vs Embarked')

sns.barplot(x = 'Sex', y = 'Survived', hue = 'Pclass', data=data1, ax = qaxis[1])
axis1.set_title('Perbandingan Survival Sex vs Pclass')

sns.barplot(x = 'Sex', y = 'Survived', hue = 'IsAlone', data=data1, ax = qaxis[2])
axis1.set_title('Perbandingan Survival Sex vs IsAlone')
```



Gambar 9. Kode dan Grafik hasil perbandingan dalam barplot

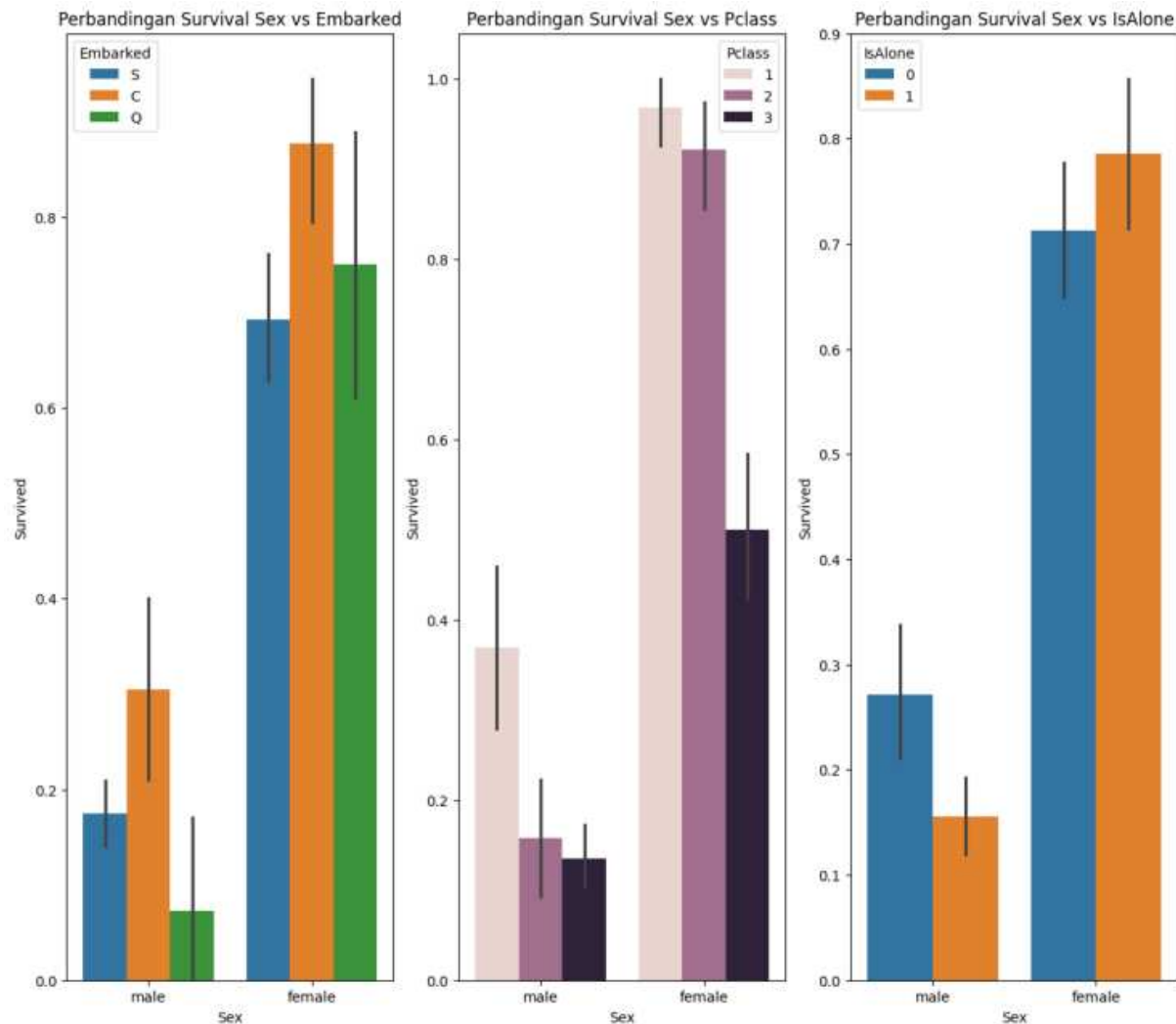
### Percobaan yang saya lakukan

```
fig, qaxis = plt.subplots(1,3,figsize=(14,12))

sns.barplot(x = 'Sex', y = 'Survived', hue = 'Embarked', data=data1, ax = qaxis[0])
qaxis[0].set_title('Perbandingan Survival Sex vs Embarked')

sns.barplot(x = 'Sex', y = 'Survived', hue = 'Pclass', data=data1, ax = qaxis[1])
qaxis[1].set_title('Perbandingan Survival Sex vs Pclass')

sns.barplot(x = 'Sex', y = 'Survived', hue = 'IsAlone', data=data1, ax = qaxis[2])
qaxis[2].set_title('Perbandingan Survival Sex vs IsAlone')
```

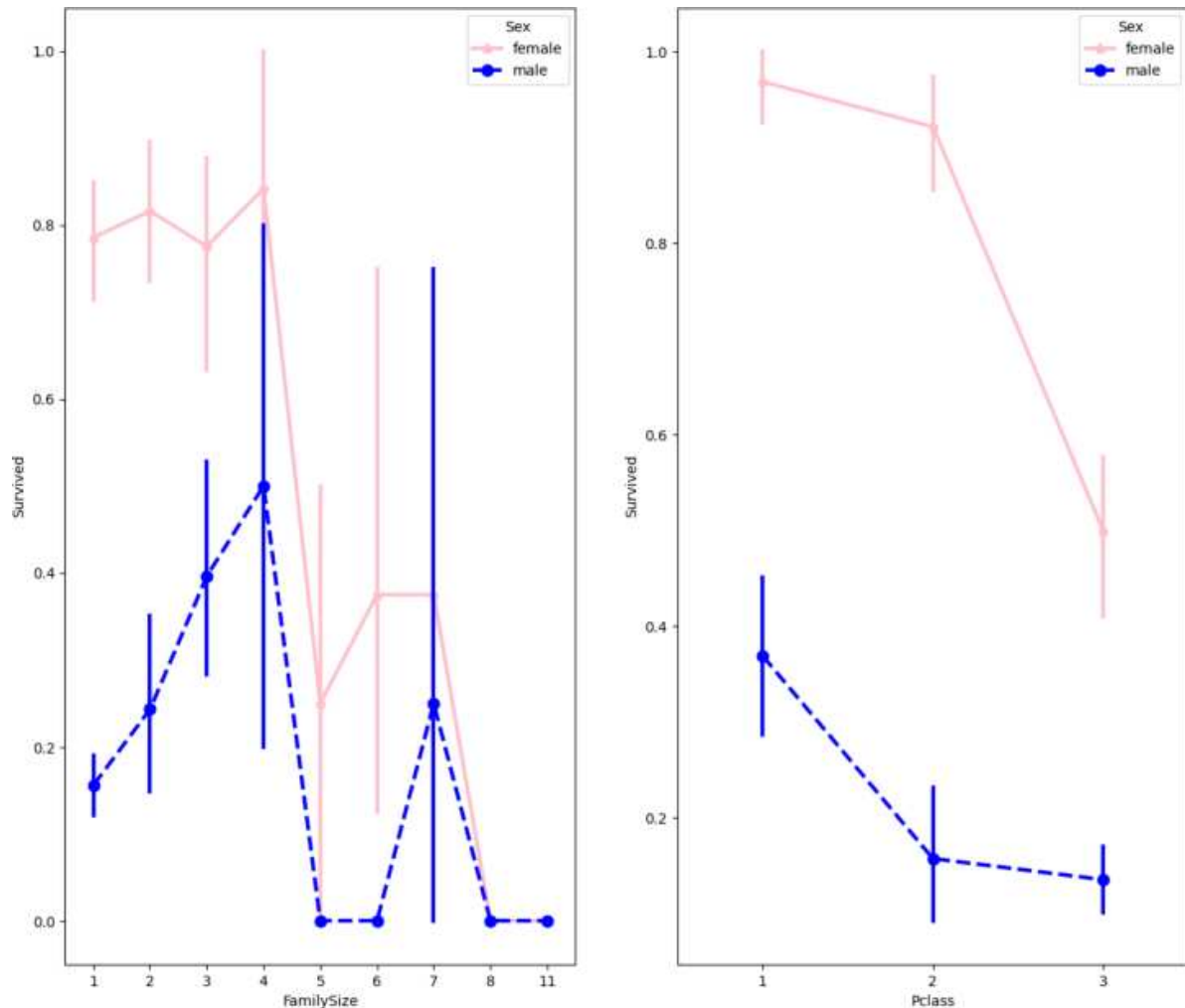


```
#contoh grafik perbandingan lainnya
fig, (maxis1, maxis2) = plt.subplots(1, 2, figsize=(14,12))

#bagaimana pengaruh faktor ukuran keluarga terhadap perbandingan jenis kelamin dan survival
sns.pointplot(x="FamilySize", y="Survived", hue="Sex", data=data1,
              palette={"male": "blue", "female": "pink"},
              markers=["*", "o"], linestyle=["-", "--"], ax = maxis1)

#bagaimana pengaruh faktor class terhadap perbandingan jenis kelamin dan survival
sns.pointplot(x="Pclass", y="Survived", hue="Sex", data=data1,
              palette={"male": "blue", "female": "pink"},
              markers=["*", "o"], linestyle=["-", "--"], ax = maxis2)
```





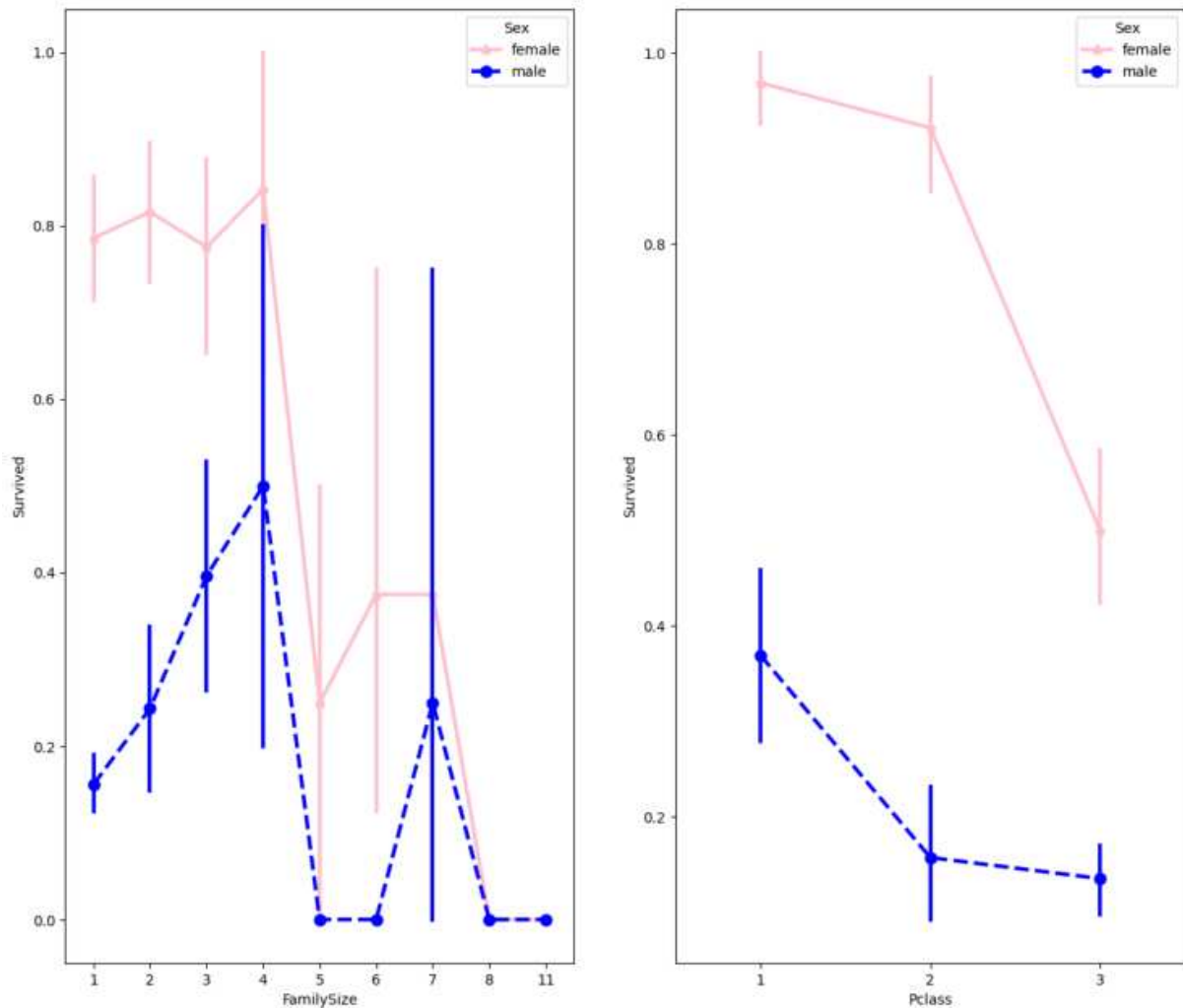
Gambar 10. Kode dan Grafik hasil perbandingan dalam pointplot

### Percobaan yang saya lakukan

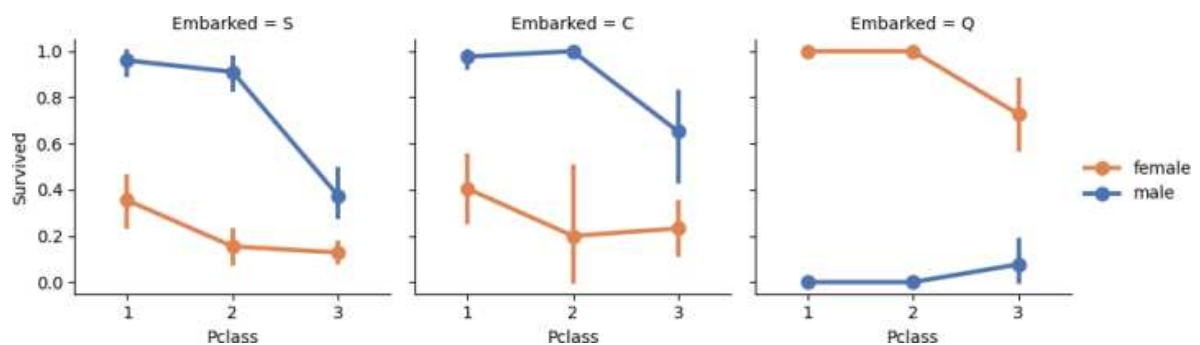
```
#contoh grafik perbandingan lainnya
fig, (maxis1, maxis2) = plt.subplots(1, 2, figsize=(14,12))

#bagaimana pengaruh faktor ukuran keluarga terhadap perbandingan jenis kelamin dan survival
sns.pointplot(x="FamilySize", y="Survived", hue="Sex", data=data1,
              palette={"male": "blue", "female": "pink"},
              markers=["*", "o"], linestyle=["-", "--"], ax = maxis1)

#bagaimana pengaruh faktor class terhadap perbandingan jenis kelamin dan survival
sns.pointplot(x="Pclass", y="Survived", hue="Sex", data=data1,
              palette={"male": "blue", "female": "pink"},
              markers=["*", "o"], linestyle=["-", "--"], ax = maxis2)
```



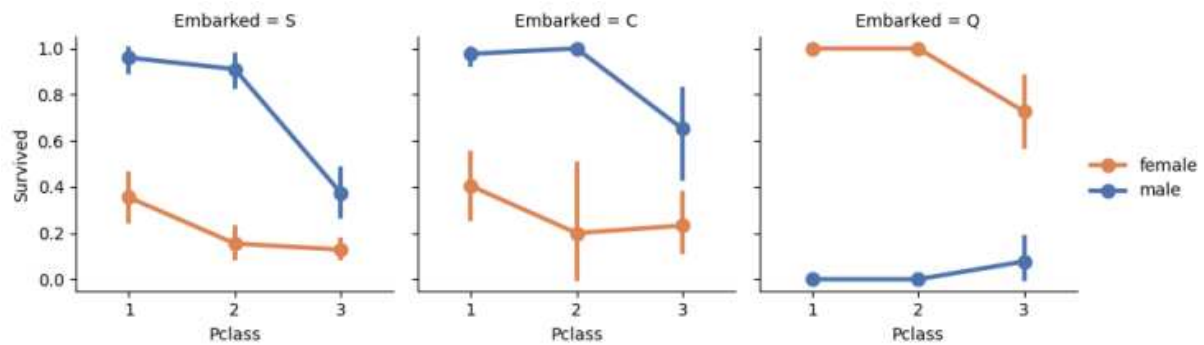
```
#bagaimana pengaruh faktor port keberangkatan terhadap perbandingan Pclass, sex, dan survival
#facetgrid: https://seaborn.pydata.org/generated/seaborn.FacetGrid.html
e = sns.FacetGrid(data1, col = 'Embarked')
e.map(sns.pointplot, 'Pclass', 'Survived', 'Sex', ci=95.0, palette = 'deep')
e.add_legend()
```



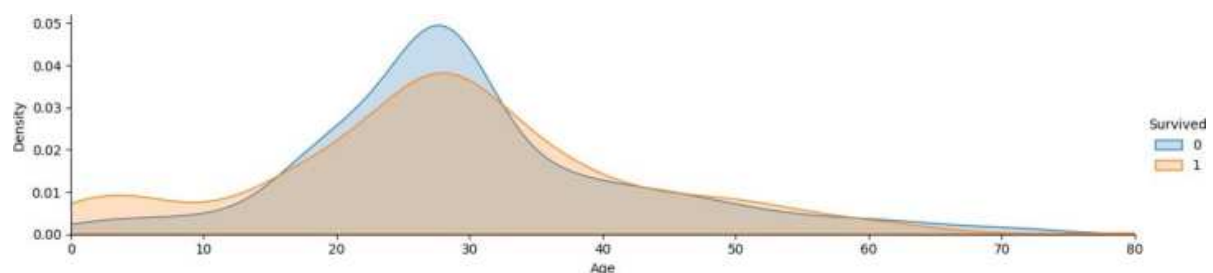
Gambar 11. Kode dan Grafik hasil perbandingan dalam pointplot

### Percobaan yang saya lakukan

```
#bagaimana pengaruh faktor port keberangkatan terhadap perbandingan Pclass, sex, dan survival
#facetgrid: https://seaborn.pydata.org/generated/seaborn.FacetGrid.html
e = sns.FacetGrid(data1, col = 'Embarked')
e.map(sns.pointplot, 'Pclass', 'Survived', 'Sex', ci=95.0, palette = 'deep')
e.add_legend()
```



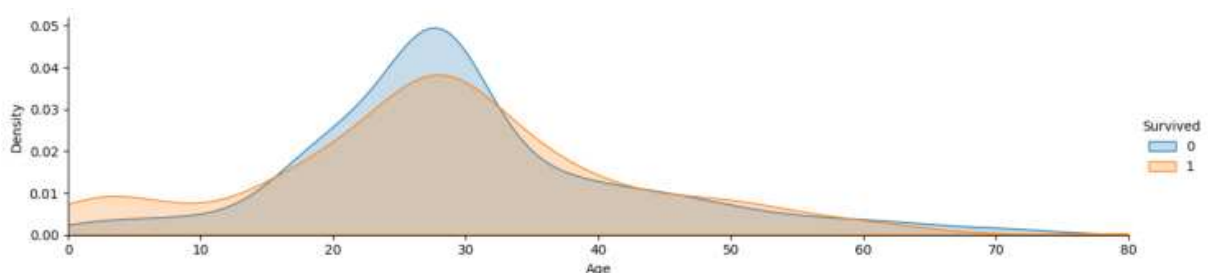
```
#plot distribusi dari usia penumpang yang selamat / tidak selamat
a = sns.FacetGrid( data1, hue = 'Survived', aspect=4 )
a.map(sns.kdeplot, 'Age', shade= True )
a.set(xlim=(0 , data1['Age'].max()))
a.add_legend()
```



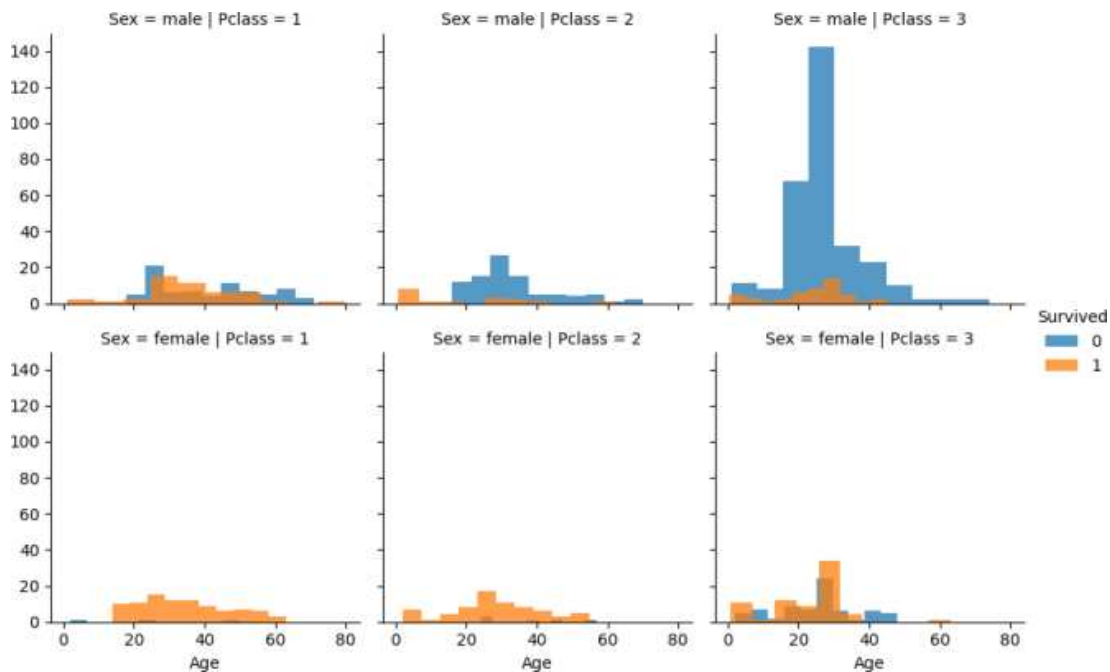
Gambar 12. Kode dan Grafik hasil perbandingan dalam kdeplot

### Percobaan yang saya lakukan

```
#plot distribusi dari usia penumpang yang selamat / tidak selamat
a = sns.FacetGrid( data1, hue = 'Survived', aspect=4 )
a.map(sns.kdeplot, 'Age', shade= True )
a.set(xlim=(0 , data1['Age'].max()))
a.add_legend()
```



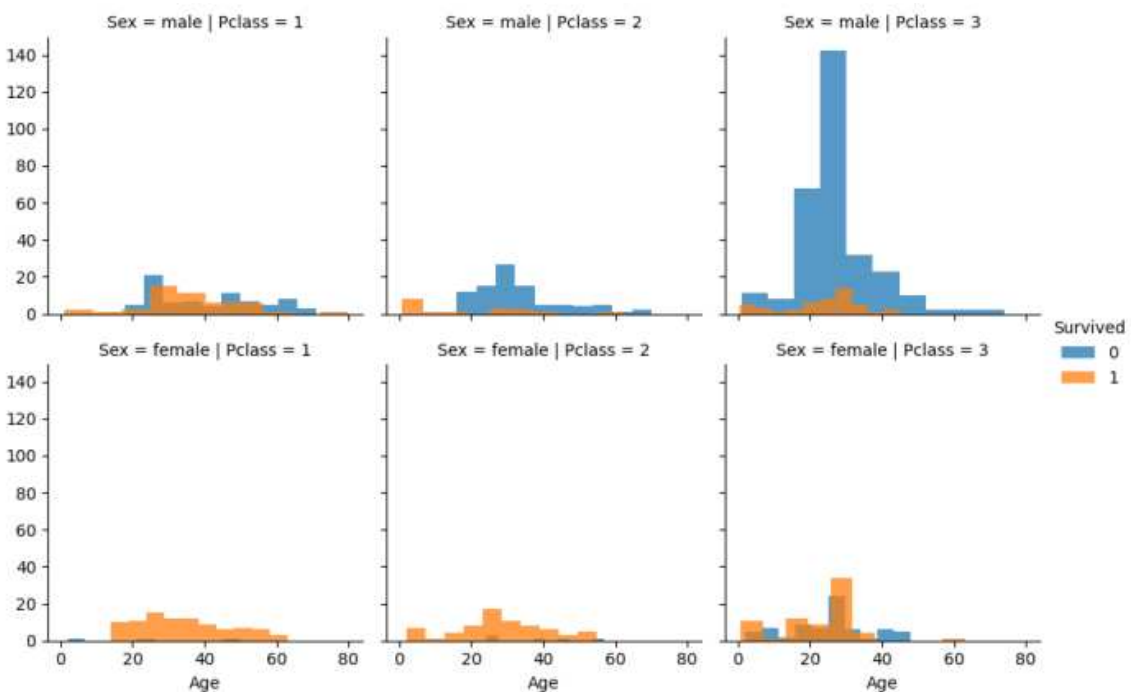
```
#perbandingan histogram dari sex, class, dan age dengan survival
h = sns.FacetGrid(data1, row = 'Sex', col = 'Pclass', hue = 'Survived')
h.map(plt.hist, 'Age', alpha = .75)
h.add_legend()
```



Gambar 13. Kode dan Grafik hasil perbandingan dalam histogram

### Percobaan yang saya lakukan

```
#perbandingan histogram dari sex, class, dan age dengan survival
h = sns.FacetGrid(data1, row = 'Sex', col = 'Pclass', hue = 'Survived')
h.map(plt.hist, 'Age', alpha = .75)
h.add_legend()
```



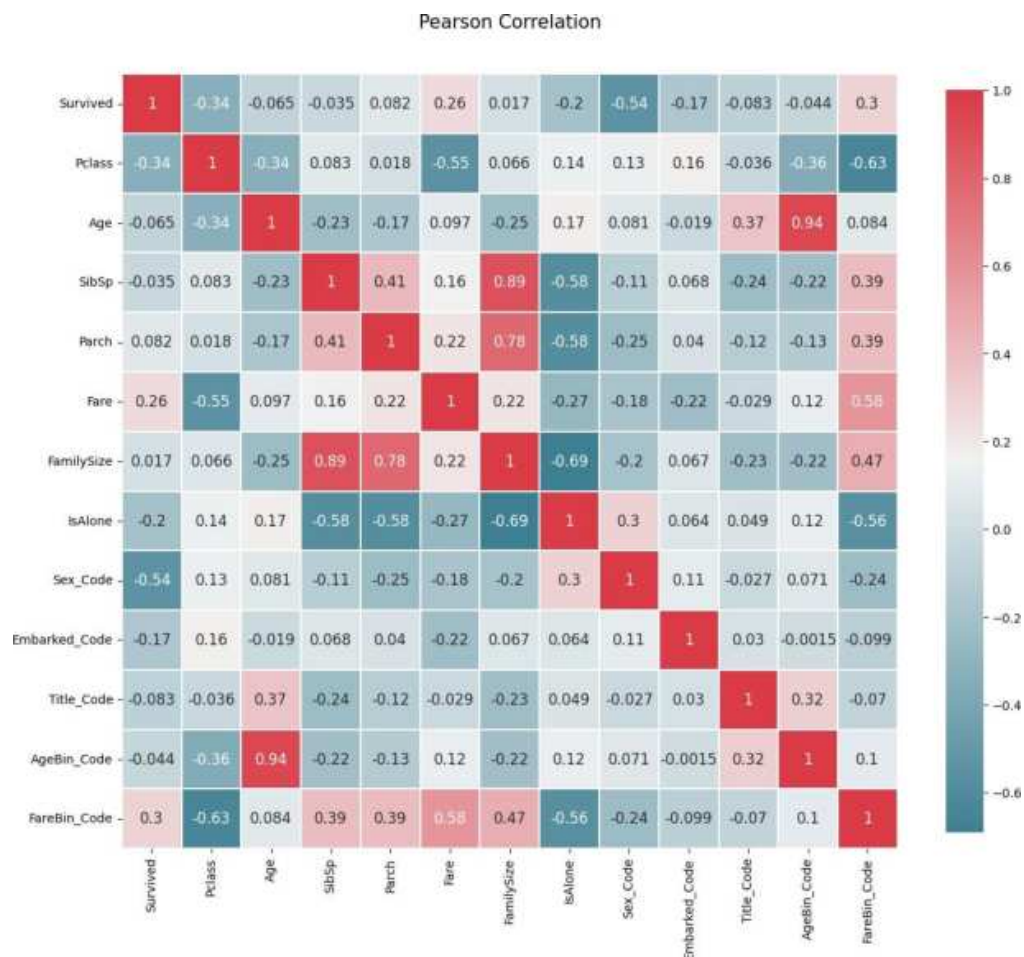
```
#Korelasi heatmap dari dataset
def correlation_heatmap(df):
    _, ax = plt.subplots(figsize=(14, 12))
    colormap = sns.diverging_palette(220, 10, as_cmap = True)

    # Hanya dipilih fitur numerik saja
    numerical_df = df.select_dtypes(include=['number'])

    _ = sns.heatmap(
        numerical_df.corr(),
        cmap = colormap,
        square=True,
        cbar_kws={'shrink':.9 },
        ax=ax,
        annot=True,
        linewidths=0.1, vmax=1.0, linecolor='white',
        annot_kws={'fontsize':12 }
    )

    plt.title('Pearson Correlation', y=1.05, size=15)

correlation_heatmap(data1)
```



Gambar 12. Kode dan Grafik hasil perbandingan dalam heatmap



### Percobaan yang saya lakukan

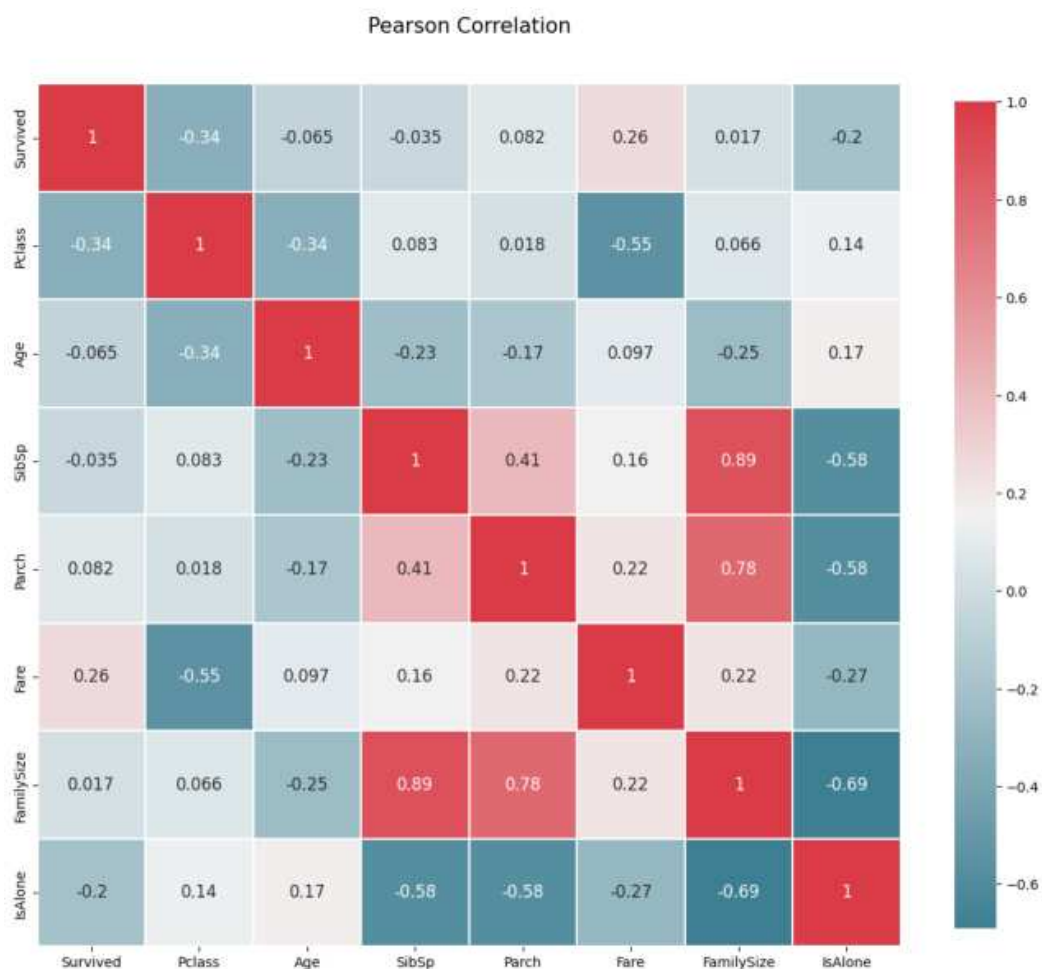
```
#Korelasi heatmap dari dataset
def correlation_heatmap(df):
    _, ax = plt.subplots(figsize=(14, 12))
    colormap = sns.diverging_palette(220, 10, as_cmap = True)

    # Hanya dipilih fitur numerik saja
    numerical_df = df.select_dtypes(include=['number'])

    _ = sns.heatmap(
        numerical_df.corr(),
        cmap = colormap,
        square=True,
        cbar_kws={'shrink':.9 },
        ax=ax,
        annot=True,
        linewidths=0.1, vmax=1.0, linecolor='white',
        annot_kws={'fontsize':12 }
    )

    plt.title('Pearson Correlation', y=1.05, size=15)

correlation_heatmap(data1)
```



## LATIHAN

1. Lakukan praktikum seperti yang ditunjukkan pada kode diatas
2. Lakukan Analisa terhadap yang sudah anda kerjakan diatas, coba tampilkan descriptive grafik lain yang memungkinkan
3. Lakukan perbaikan dan visualisasikan dataset berikut:  
<https://www.kaggle.com/datasets/mathchi/diabetes-data-set>
4. Pada dataset tersebut, fitur bernilai 0 merupakan missing values. Baca deskripsi dari dataset tersebut sebelum anda melakukan perbaikan dan visualisasi data

Jawaban :

### 2. Analisa Dataset Titanic

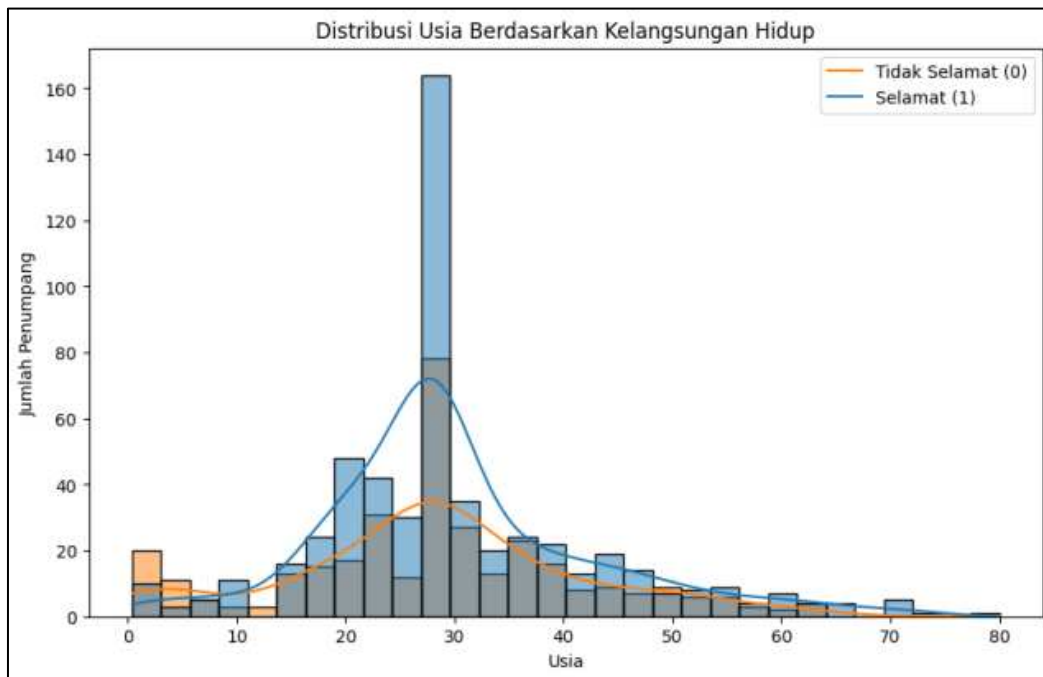
1. **Pclass (Kelas Penumpang)**
  - a. Semakin tinggi kelasnya (Pclass = 1), semakin besar peluang untuk bertahan hidup.
  - b. Hal ini bisa dikaitkan dengan prioritas penyelamatan pada penumpang kelas atas.
2. **Sex (Jenis Kelamin)**
  - a. Perempuan memiliki tingkat kelangsungan hidup lebih tinggi dibandingkan laki-laki.
  - b. Hal ini mendukung aturan penyelamatan "Women and children first."
3. **Fare (Harga Tiket)**
  - a. Semakin tinggi harga tiket (Fare), semakin besar kemungkinan bertahan hidup.
  - b. Ini berhubungan dengan kelas penumpang, di mana Pclass 1 membayar lebih mahal dan lebih diutamakan.
4. **Age (Usia)**
  - a. Anak-anak (usia lebih muda) memiliki kemungkinan selamat yang lebih tinggi dibandingkan orang dewasa.
5. **Embarked (Pelabuhan Keberangkatan)**
  - a. Ada perbedaan tingkat keselamatan berdasarkan titik naik kapal, yang bisa berhubungan dengan kelas ekonomi di setiap lokasi.
6. **FamilySize & IsAlone**
  - a. Penumpang yang bepergian sendirian (IsAlone = 1) memiliki kemungkinan lebih kecil untuk bertahan hidup dibandingkan mereka yang memiliki keluarga (FamilySize > 1).
7. **Korelasi antar Variabel**
  - a. Korelasi heatmap menunjukkan bahwa fitur seperti Sex, Pclass, dan Fare memiliki hubungan kuat terhadap **Survival**.

### Tampilan descriptive grafik lainnya

#### Distribusi Usia Berdasarkan Kelangsungan Hidup

Menunjukkan bagaimana rentang usia mempengaruhi peluang selamat.

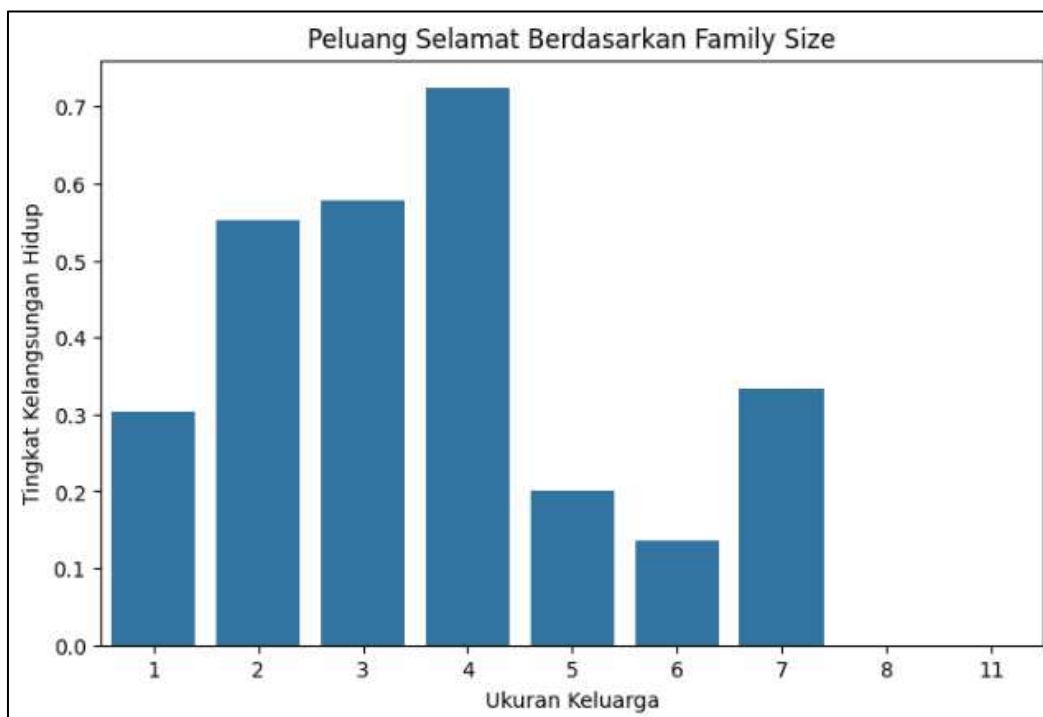
```
plt.figure(figsize=(10,6))
sns.histplot(data=data1, x='Age', hue='Survived', kde=True, bins=30)
plt.title('Distribusi Usia Berdasarkan Kelangsungan Hidup')
plt.xlabel('Usia')
plt.ylabel('Jumlah Penumpang')
plt.legend(['Tidak Selamat (0)', 'Selamat (1)'])
plt.show()
```



### Survival Rate Berdasarkan FamilySize (Ukuran Keluarga)

Untuk melihat apakah bepergian dalam kelompok besar memengaruhi keselamatan.

```
plt.figure(figsize=(8,5))
sns.barplot(x='FamilySize', y='Survived', data=data1, ci=None)
plt.title('Peluang Selamat Berdasarkan Family Size')
plt.xlabel('Ukuran Keluarga')
plt.ylabel('Tingkat Kelangsungan Hidup')
plt.show()
```





### 3. Memperbaiki dan Memvisualisasi Dataset Diabetes

#### Tahap 1: Import dan Memuat Dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

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

# Memuat dataset
data_raw = pd.read_csv('/content/drive/MyDrive/Data Mining/diabetes.csv')
data1 = data_raw.copy(deep=True)

# Menampilkan informasi dataset
print("Informasi dataset:")
print(data1.info())

# Menampilkan jumlah nilai 0 per kolom
print("\nKolom dengan nilai 0 sebagai missing value:")
print((data1 == 0).sum())

# Statistik deskriptif dataset
print("\nStatistik deskriptif:")
print(data1.describe())
```

```
Mounted at /content/drive
Informasi dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Pregnancies            768 non-null   int64
1   Glucose                768 non-null   int64
2   BloodPressure          768 non-null   int64
3   SkinThickness          768 non-null   int64
4   Insulin                768 non-null   int64
5   BMI                   768 non-null   float64
6   DiabetesPedigreeFunction 768 non-null   float64
7   Age                   768 non-null   int64
8   Outcome                768 non-null   int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
None

Kolom dengan nilai 0 sebagai missing value:
Pregnancies      111
Glucose           5
BloodPressure     35
SkinThickness    227
Insulin          374
BMI              11
DiabetesPedigreeFunction 0
Age              0
Outcome          500
dtype: int64
```

Statistik deskriptif:

|       | Pregnancies | Glucose    | BloodPressure | SkinThickness | Insulin    | \ |
|-------|-------------|------------|---------------|---------------|------------|---|
| count | 768.000000  | 768.000000 | 768.000000    | 768.000000    | 768.000000 |   |
| mean  | 3.845052    | 120.894531 | 69.105469     | 20.536458     | 79.799479  |   |
| std   | 3.369578    | 31.972618  | 19.355807     | 15.952218     | 115.244002 |   |
| min   | 0.000000    | 0.000000   | 0.000000      | 0.000000      | 0.000000   |   |
| 25%   | 1.000000    | 99.000000  | 62.000000     | 0.000000      | 0.000000   |   |
| 50%   | 3.000000    | 117.000000 | 72.000000     | 23.000000     | 30.500000  |   |
| 75%   | 6.000000    | 140.250000 | 80.000000     | 32.000000     | 127.250000 |   |
| max   | 17.000000   | 199.000000 | 122.000000    | 99.000000     | 846.000000 |   |

|       | BMI        | DiabetesPedigreeFunction | Age        | Outcome    |
|-------|------------|--------------------------|------------|------------|
| count | 768.000000 | 768.000000               | 768.000000 | 768.000000 |
| mean  | 31.992578  | 0.471876                 | 33.240885  | 0.348958   |
| std   | 7.884160   | 0.331329                 | 11.760232  | 0.476951   |
| min   | 0.000000   | 0.078000                 | 21.000000  | 0.000000   |
| 25%   | 27.300000  | 0.243750                 | 24.000000  | 0.000000   |
| 50%   | 32.000000  | 0.372500                 | 29.000000  | 0.000000   |
| 75%   | 36.600000  | 0.626250                 | 41.000000  | 1.000000   |
| max   | 67.100000  | 2.420000                 | 81.000000  | 1.000000   |

## Tahap 2: Identifikasi dan Perbaikan Data

```
# Daftar kolom yang tidak seharusnya memiliki nilai 0
missing_cols = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']

# Mengisi missing value dengan median
for col in missing_cols:
    data1[col].replace(0, data1[col].median(), inplace=True)

# Menampilkan kembali jumlah nilai 0 setelah perbaikan
print("\nJumlah nilai 0 setelah imputasi:")
print((data1 == 0).sum())
```

```
Jumlah nilai 0 setelah imputasi:
Pregnancies      111
Glucose           0
BloodPressure     0
SkinThickness     0
Insulin           0
BMI               0
DiabetesPedigreeFunction  0
Age               0
Outcome           500
dtype: int64
```

## Tahap 3: Pembuatan Fitur Tambahan

```
# Kategorisasi usia
data1['AgeBin'] = pd.cut(data1['Age'], bins=5, labels=['Young', 'Adult', 'Middle-aged', 'Senior', 'Elderly'])

# Kategorisasi BMI
data1['BMIBin'] = pd.cut(data1['BMI'], bins=[0, 18.5, 24.9, 29.9, 40, 100], labels=['Underweight', 'Normal', 'Overweight', 'Obese', 'Severely Obese'])

# Menampilkan sampel data setelah transformasi
print("\nData setelah kategorisasi:")
print(data1[['Age', 'AgeBin', 'BMI', 'BMIBin']].head())
```

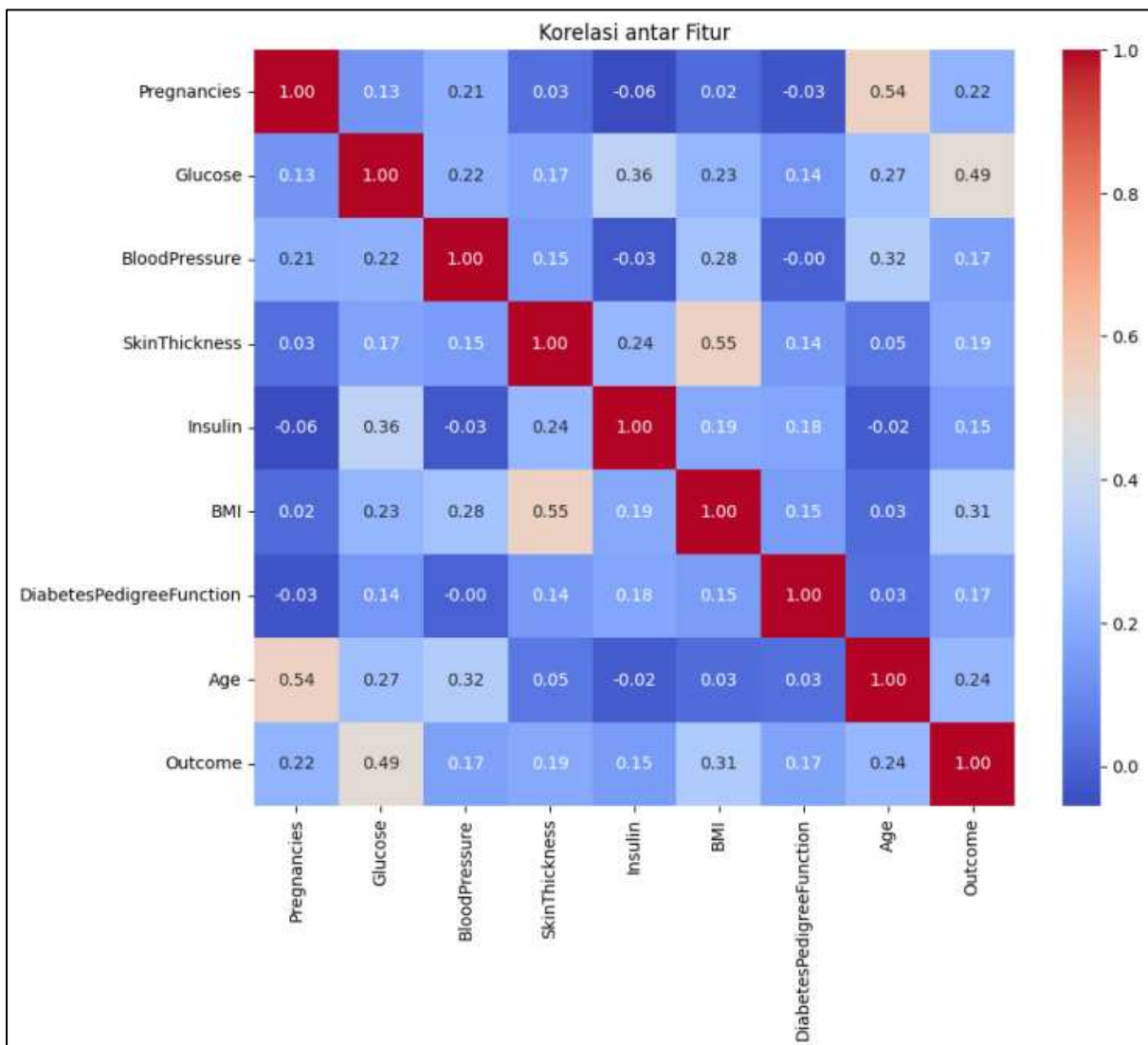
Data setelah kategorisasi:

|   | Age | AgeBin      | BMI  | BMI Bin        |
|---|-----|-------------|------|----------------|
| 0 | 50  | Middle-aged | 33.6 | Obese          |
| 1 | 31  | Young       | 26.6 | Overweight     |
| 2 | 32  | Young       | 23.3 | Normal         |
| 3 | 21  | Young       | 28.1 | Overweight     |
| 4 | 33  | Young       | 43.1 | Severely Obese |

Tahap 4: Analisis Korelasi Fitur dengan Diabetes

```
# Menghapus kolom kategorikal sebelum analisis korelasi
numeric_data = data1.select_dtypes(include=['number'])

# Korelasi antar fitur
plt.figure(figsize=(10, 8))
sns.heatmap(numeric_data.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Korelasi antar Fitur")
plt.show()
```



## Tahap 5: Visualisasi Distribusi Fitur

```
plt.figure(figsize=(16,12))

# Boxplot fitur numerik
plt.subplot(231)
sns.boxplot(y=data1['Glucose'])
plt.title('Glucose Boxplot')

plt.subplot(232)
sns.boxplot(y=data1['BloodPressure'])
plt.title('Blood Pressure Boxplot')

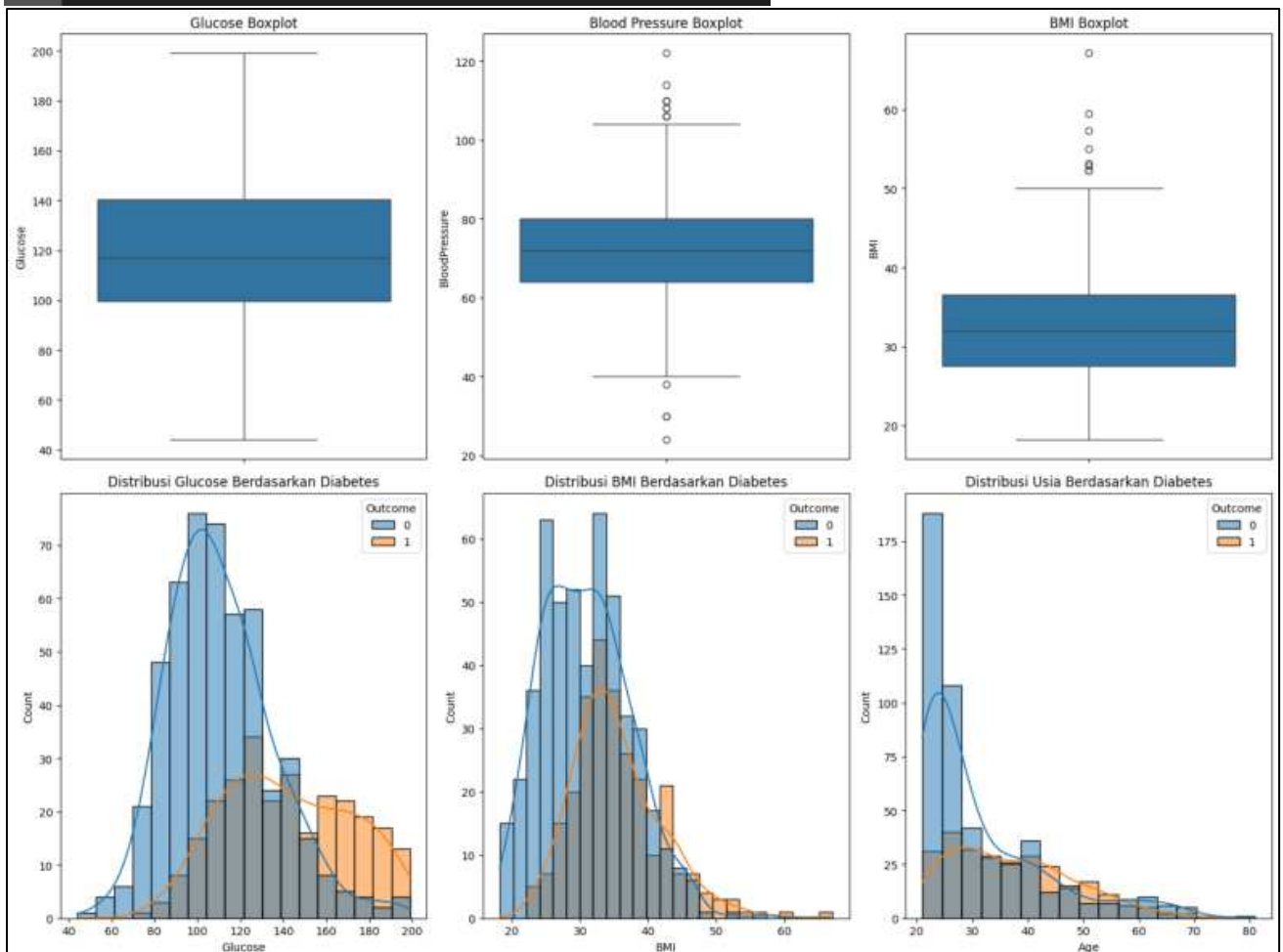
plt.subplot(233)
sns.boxplot(y=data1['BMI'])
plt.title('BMI Boxplot')

# Histogram distribusi berdasarkan outcome diabetes
plt.subplot(234)
sns.histplot(data1, x='Glucose', hue='Outcome', kde=True)
plt.title('Distribusi Glucose Berdasarkan Diabetes')

plt.subplot(235)
sns.histplot(data1, x='BMI', hue='Outcome', kde=True)
plt.title('Distribusi BMI Berdasarkan Diabetes')

plt.subplot(236)
sns.histplot(data1, x='Age', hue='Outcome', kde=True)
plt.title('Distribusi Usia Berdasarkan Diabetes')

plt.tight_layout()
plt.show()
```





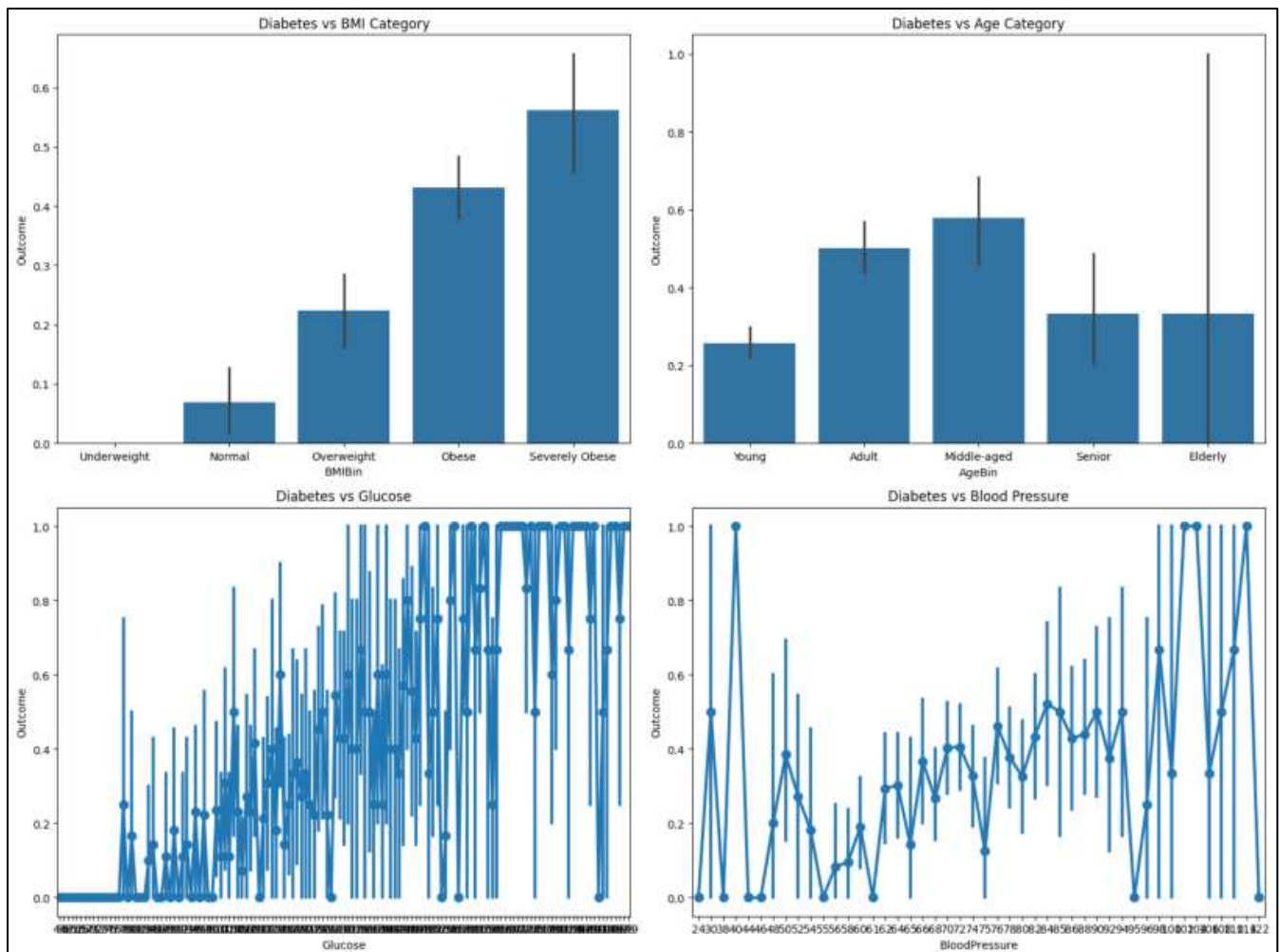
## Tahap 6: Visualisasi Perbandingan Antar Variabel

```
fig, saxis = plt.subplots(2, 2, figsize=(16,12))

sns.barplot(x='BMIBin', y='Outcome', data=data1, ax=saxis[0,0])
sns.barplot(x='AgeBin', y='Outcome', data=data1, ax=saxis[0,1])
sns.pointplot(x='Glucose', y='Outcome', data=data1, ax=saxis[1,0])
sns.pointplot(x='BloodPressure', y='Outcome', data=data1, ax=saxis[1,1])

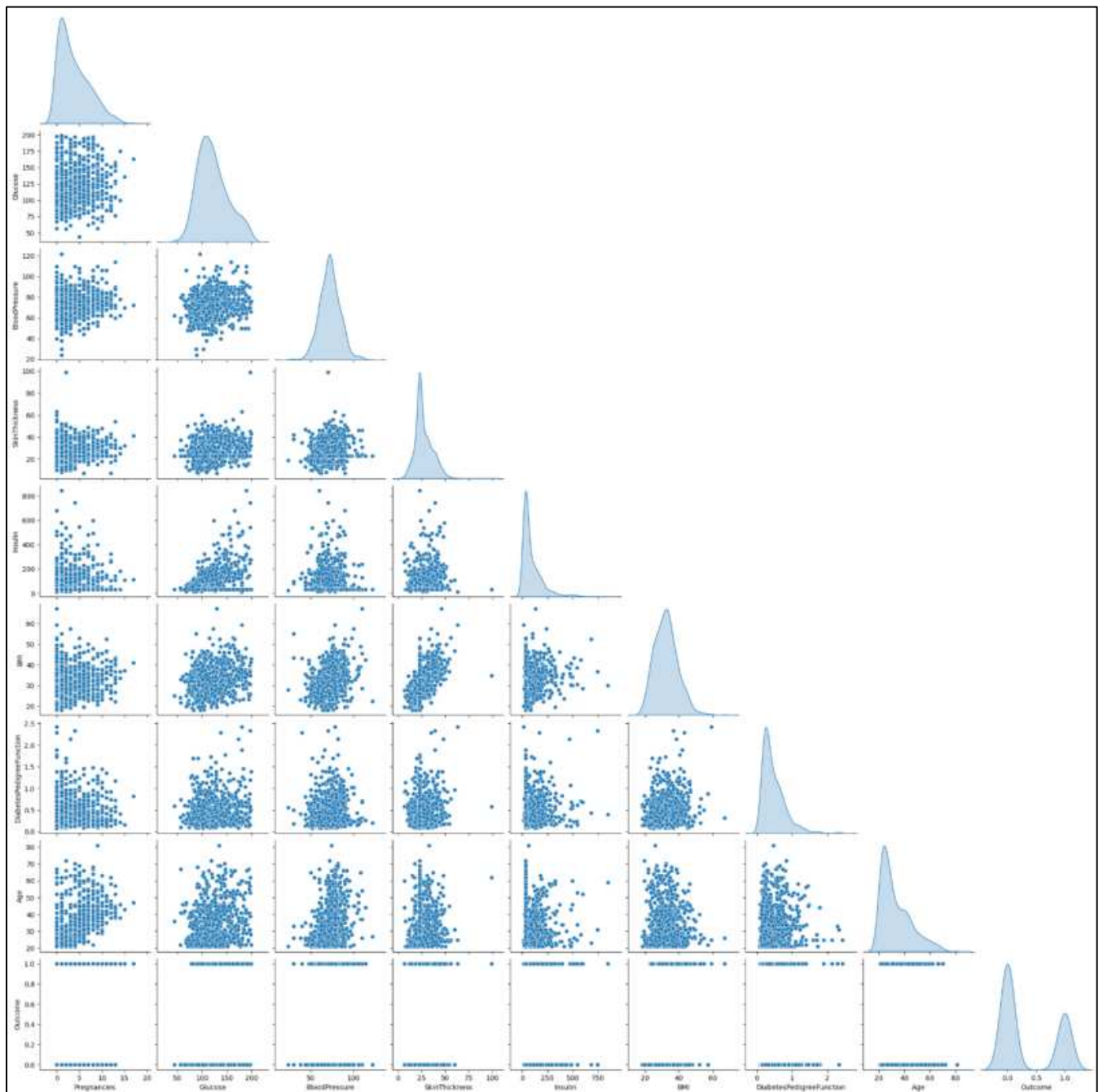
saxis[0,0].set_title('Diabetes vs BMI Category')
saxis[0,1].set_title('Diabetes vs Age Category')
saxis[1,0].set_title('Diabetes vs Glucose')
saxis[1,1].set_title('Diabetes vs Blood Pressure')

plt.tight_layout()
plt.show()
```



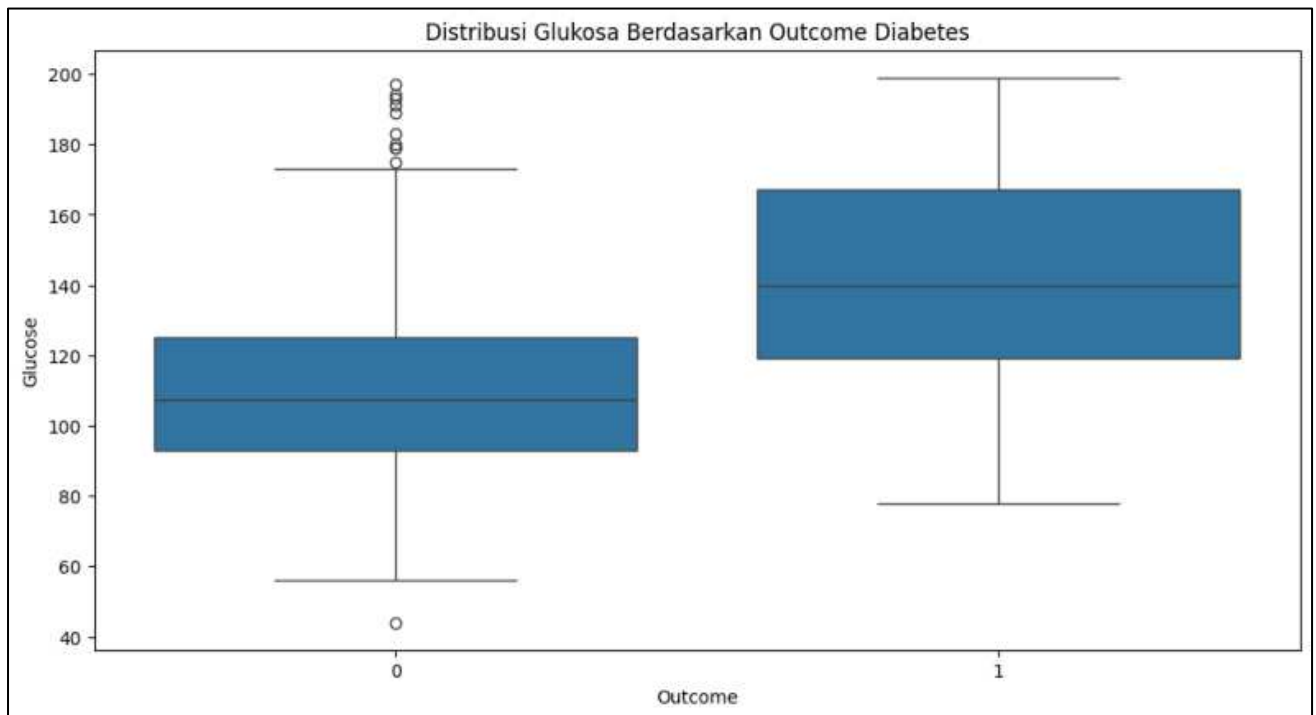
## Tahap 7: Visualisasi Hubungan Antar Variabel Numerik

```
sns.pairplot(numeric_data, diag_kind="kde", corner=True)
plt.show()
```



Tahap 8: Visualisasi distribusi fitur berdasarkan kategori tertentu

```
plt.figure(figsize=(12, 6))
sns.boxplot(x="Outcome", y="Glucose", data=data1)
plt.title("Distribusi Glukosa Berdasarkan Outcome Diabetes")
plt.show()
```



#### 4. Membaca Deskripsi dari Dataset Diabetes

Untuk membaca deskripsi dari dataset ada di tahap 1 pada soal no 3 berikut lebih jelasnya

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

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

# Memuat dataset
data_raw = pd.read_csv('/content/drive/MyDrive/Data Mining/diabetes.csv')
data1 = data_raw.copy(deep=True)

# Menampilkan informasi dataset
print("Informasi dataset:")
print(data1.info())

# Menampilkan jumlah nilai 0 per kolom
print("\nKolom dengan nilai 0 sebagai missing value:")
print((data1 == 0).sum())

# Statistik deskriptif dataset
print("\nStatistik deskriptif:")
print(data1.describe())
```

```

Mounted at /content/drive
Informasi dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Pregnancies            768 non-null   int64
1   Glucose                768 non-null   int64
2   BloodPressure          768 non-null   int64
3   SkinThickness          768 non-null   int64
4   Insulin                768 non-null   int64
5   BMI                   768 non-null   float64
6   DiabetesPedigreeFunction 768 non-null   float64
7   Age                   768 non-null   int64
8   Outcome                768 non-null   int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
None

Kolom dengan nilai 0 sebagai missing value:
Pregnancies      111
Glucose           5
BloodPressure     35
SkinThickness    227
Insulin          374
BMI              11
DiabetesPedigreeFunction 0
Age              0
Outcome          500
dtype: int64

```

```

Statistik deskriptif:

```

|       | Pregnancies | Glucose    | BloodPressure | SkinThickness | Insulin    |
|-------|-------------|------------|---------------|---------------|------------|
| count | 768.000000  | 768.000000 | 768.000000    | 768.000000    | 768.000000 |
| mean  | 3.845052    | 120.894531 | 69.105469     | 20.536458     | 79.799479  |
| std   | 3.369578    | 31.972618  | 19.355807     | 15.952218     | 115.244002 |
| min   | 0.000000    | 0.000000   | 0.000000      | 0.000000      | 0.000000   |
| 25%   | 1.000000    | 99.000000  | 62.000000     | 0.000000      | 0.000000   |
| 50%   | 3.000000    | 117.000000 | 72.000000     | 23.000000     | 30.500000  |
| 75%   | 6.000000    | 140.250000 | 80.000000     | 32.000000     | 127.250000 |
| max   | 17.000000   | 199.000000 | 122.000000    | 99.000000     | 846.000000 |

|       | BMI        | DiabetesPedigreeFunction | Age        | Outcome    |
|-------|------------|--------------------------|------------|------------|
| count | 768.000000 | 768.000000               | 768.000000 | 768.000000 |
| mean  | 31.992578  | 0.471876                 | 33.240885  | 0.348958   |
| std   | 7.884160   | 0.331329                 | 11.760232  | 0.476951   |
| min   | 0.000000   | 0.078000                 | 21.000000  | 0.000000   |
| 25%   | 27.300000  | 0.243750                 | 24.000000  | 0.000000   |
| 50%   | 32.000000  | 0.372500                 | 29.000000  | 0.000000   |
| 75%   | 36.600000  | 0.626250                 | 41.000000  | 1.000000   |
| max   | 67.100000  | 2.420000                 | 81.000000  | 1.000000   |