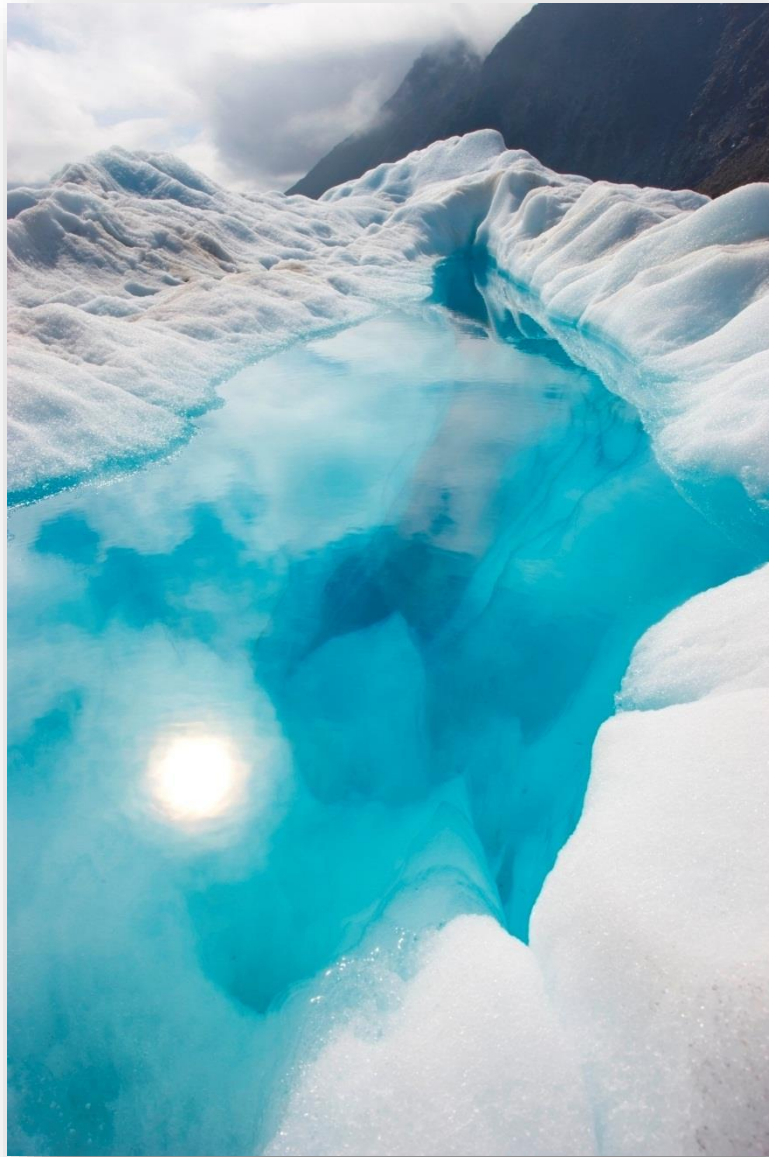


2024



BUKU KERJA/JOB SHEET

ASSOCIATE DATA SCIENTIST

Nama Peserta	:	Luthfi Krisna Bayu
Nomor Urut	:	TUK-035.017850

DAFTAR ISI

DAFTAR ISI	1
BUKTI 1-ADS.....	3
1. Kebutuhan Data	3
2. Pengambilan Data.....	6
3. Pengintegrasian Data.....	6
BUKTI 2-ADS.....	8
1. Analisis Tipe dan Relasi Data	8
2. Analisis Karakteristik Data.....	12
3. Laporan Telaah Data	17
BUKTI 3-ADS.....	18
1. Pengecekan Kelengkapan Data.....	18
2. Rekomendasi Kelengkapan Data	23
BUKTI 4-ADS.....	26
1. Kriteria dan Teknik Pemilihan Data	26
2. Attributes (Columns) dan Records (Row) Data	27
BUKTI 5-ADS.....	29
1. Pembersihan Data Kotor.....	29
2. Laporan dan Rekomendasi Hasil Pembersihan Data Kotor	32
BUKTI 6-ADS.....	35
1. Analisis Teknik Transformasi Data	35
2. Transformasi Data	42
3. Dokumentasi Konstruksi Data.....	47
BUKTI 7-ADS.....	48
1. Pelabelan Data	48
2. Laporan Hasil Pelabelan Data	53
BUKTI 8-ADS.....	57
1. Parameter Model.....	57
2. Tools Pemodelan	60
BUKTI 9-ADS.....	65
1. Penggunaan Model dengan Data Riil	65

2.	Penilaian Hasil Pemodelan.....	68
----	--------------------------------	----

BUKTI 1-ADS

Kode Unit	:	J.62DMI00.004.1
Judul Unit	:	Mengumpulkan Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam mengumpulkan data untuk data science.

Langkah Kerja:

- 1) Menentukan kebutuhan data
- 2) Mengambil data
- 3) Mengintegrasikan data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengubah teks
 - Aplikasi basis data
 - Tools pengambilan data

1. KEBUTUHAN DATA

Instruksi Kerja:

- Identifikasi kebutuhan data sesuai tujuan teknis data science
- Periksa ketersediaan data berdasarkan kebutuhan data sesuai aturan yang berlaku
- Tentukan volume data berdasarkan kebutuhan data sesuai tujuan teknis data science

Jawab:

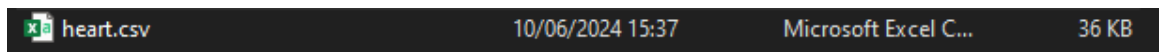
A.) Berdasarkan apa yang telah saya kerjakan sebelumnya, kebutuhan data untuk tujuan teknis data science dalam kasus ini adalah untuk membangun model klasifikasi penyakit jantung menggunakan algoritma Decision Tree dan XGBoost. Berikut adalah identifikasi kebutuhan data dari dataset heart.csv dengan 12 kolom:

- Age: usia pasien [tahun]
- Sex: jenis kelamin pasien [M: Male, F: Female]
- ChestPainType: jenis nyeri dada [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
- RestingBP: tekanan darah istirahat [mm Hg]
- Cholesterol: kolesterol serum [mm/dl]
- FastingBS: gula darah puasa [1: jika PuasaBS > 120mg/dl, 0: sebaliknya]
- RestingECG: Hasil elektrokardiogram saat istirahat
- MaxHR: detak jantung maksimum

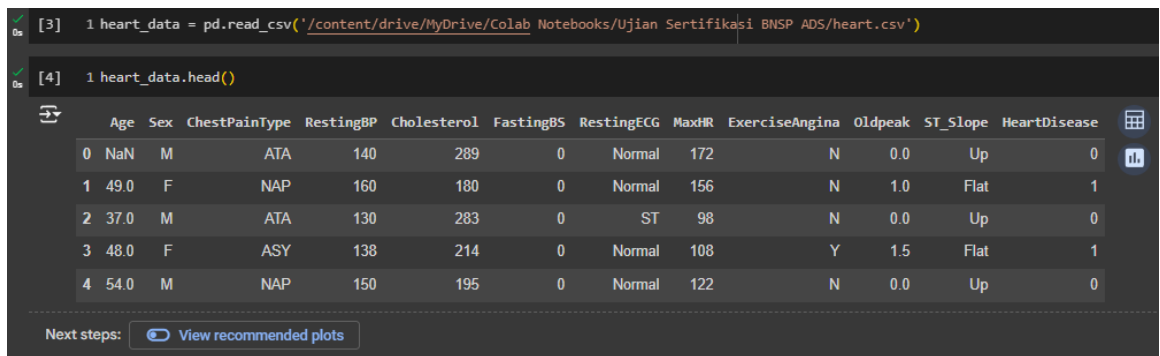
- ExerciseAngina: angina yang diinduksi oleh Latihan
- Oldpeak: depresi ST yang dihasilkan oleh latihan relatif terhadap istirahat
- ST_Slope: kemiringan segmen ST
- HeartDisease: output class [1: penyakit jantung, 0: Normal]

B.) Untuk memeriksa ketersediaan data berdasarkan kebutuhan yang telah diidentifikasi, langkah-langkah berikut dapat diambil:

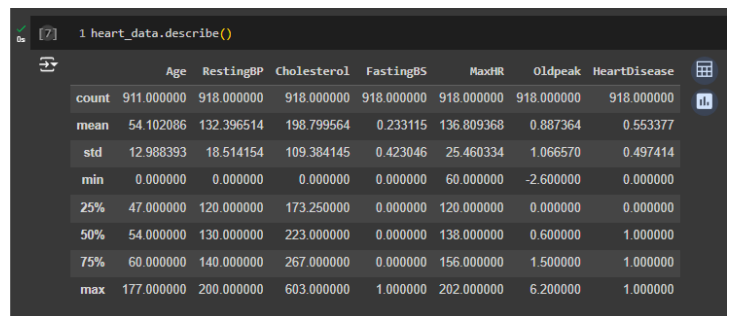
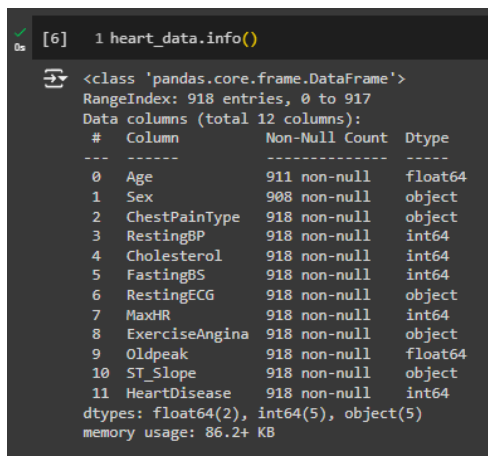
1. **Memeriksa Ketersediaan Dataset:** Pastikan dataset heart.csv tersedia dan dapat diakses.



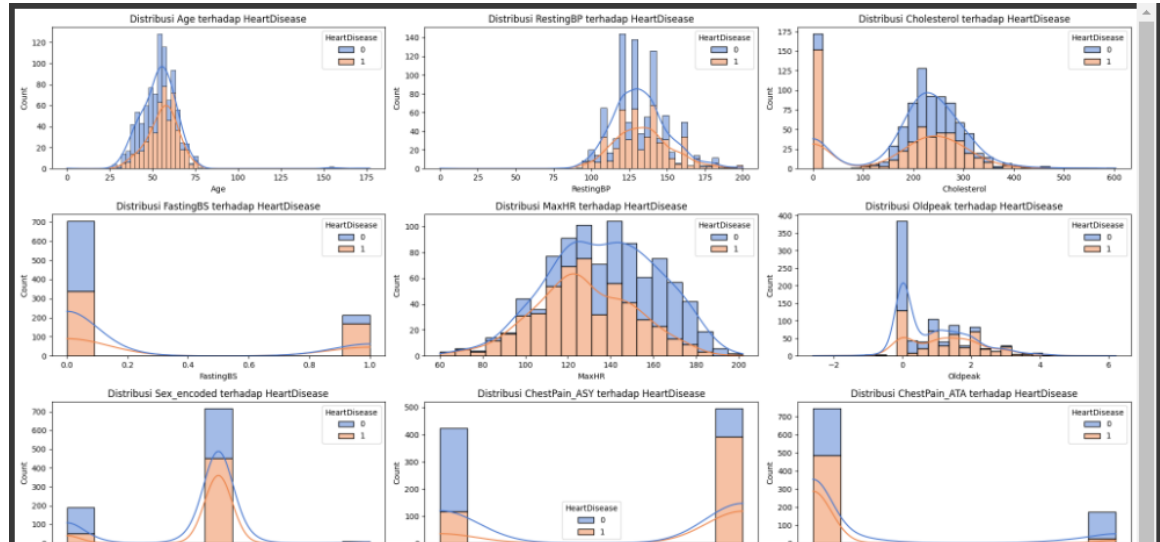
2. **Memeriksa Integritas Dataset:** Periksa apakah dataset memiliki struktur yang sesuai dengan deskripsi kolom yang telah disediakan. Ini mencakup memeriksa jumlah baris dan kolom, serta memastikan setiap kolom memiliki tipe data yang sesuai dengan deskripsi.



3. **Memeriksa Ketersediaan Data untuk Setiap Kolom:** Pastikan tidak ada nilai yang hilang atau data yang tidak valid dalam setiap kolom. Ini dapat dilakukan dengan menggunakan metode seperti .info() dan .isnull().sum() untuk mendapatkan ringkasan dataset dan jumlah nilai yang hilang dalam setiap kolom.



4. **Memeriksa Distribusi Kelas pada Kolom Output:** Periksa distribusi kelas pada kolom output HeartDisease untuk memastikan bahwa dataset seimbang atau tidak. Ini penting untuk memastikan bahwa model yang dikembangkan tidak bias terhadap kelas mayoritas.



C.) Dalam kasus ini, kita telah menggunakan dataset heart.csv yang memiliki 12 kolom fitur dan 1 kolom target. Volume data yang diperlukan dapat ditentukan sebagai berikut:

1. **Data Pelatihan:** Volume data pelatihan harus cukup besar untuk memberikan model informasi yang cukup untuk belajar pola yang ada dalam data. Sebagai rekomendasi umum, sekitar 70-80% dari total data dapat dialokasikan untuk pelatihan model.
2. **Data Pengujian:** Data pengujian digunakan untuk menguji kinerja model secara independen setelah proses pelatihan dan penyetelan selesai. Sebagai rekomendasi, sekitar 20-30% dari total data dapat dialokasikan untuk pengujian.

```
4.1 Train-Test Split

[62] 1 # Pisahkan fitur dan target
      2 X = heart_data.drop('HeartDisease', axis=1)
      3 y = heart_data['HeartDisease']

[63] 1 # Lakukan train-test split
      2 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Dengan mengikuti alokasi data yang sesuai, kita dapat memastikan bahwa model yang dikembangkan memiliki data yang cukup untuk pelatihan dan pengujian yang baik.

2. PENGAMBILAN DATA

Instruksi Kerja:

- Identifikasi metode dan tools pengambilan data sesuai tujuan teknis data science
- Tentukan tools pengambilan data sesuai tujuan teknis data science
- Siapkan tools pengambilan data sesuai tujuan teknis data science
- Jalankan proses pengambilan data sesuai dengan tools yang telah disiapkan

Jawab:

A.) Metode pengambilan data (dataset : heart.csv) yang dilakukan dari drive dengan tools yaitu google colab

B.) Untuk tools yang digunakan yaitu Google Colab dengan codingan Python

C.) Tools disiapkan dengan code pengambilan data yang ada di jawaban selanjutnya (D.)

D.) Berikut adalah pengambilan datanya

```
1.3. Load Data

[3] 1 heart_data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Ujian Sertifikasi BNSP ADS/heart.csv')

[4] 1 heart_data.head()
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	NaN	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49.0	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37.0	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48.0	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54.0	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

3. PENGINTEGRASIAN DATA

Instruksi Kerja:

- Periksa integritas data sesuai tujuan teknis data science
- Integrasikan data sesuai tujuan teknis data science

Jawab:

A.) Untuk memeriksa integritas data dalam konteks teknis data science, kita dapat melakukan beberapa langkah untuk memastikan kualitas dan keakuratan data. Berikut adalah beberapa langkah yang bisa diimplementasikan:

- **Memeriksa Duplikasi Data:** Kita dapat menggunakan Pandas dalam Python untuk memeriksa apakah ada baris data yang duplikat dalam dataset.

```
1 duplicate_rows = heart_data.duplicated()
2 print("Jumlah baris duplikat:", duplicate_rows.sum())
```

Jumlah baris duplikat: 0

- **Memeriksa Missing Values:** Missing values atau nilai yang hilang adalah masalah umum dalam data. Kita perlu memeriksa apakah ada kolom atau baris yang memiliki nilai yang hilang dan memutuskan cara menangani nilai yang hilang tersebut.

```
[9] 1 missing_values = heart_data.isnull().sum()
    2 print("Jumlah missing values per kolom:\n", missing_values)
```

Jumlah missing values per kolom:

Age	7
Sex	10
ChestPainType	0
RestingBP	0
Cholesterol	0
FastingBS	0
RestingECG	0
MaxHR	0
ExerciseAngina	0
Oldpeak	0
ST_Slope	0
HeartDisease	0
dtype:	int64

- **Validasi Tipe Data:** Validasi tipe data penting untuk memastikan bahwa data yang disimpan sesuai dengan ekspektasi. Misalnya, kolom yang seharusnya berisi nilai numerik seharusnya memiliki tipe data numerik, bukan string.

```
[6] 1 heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  --
 0   Age                911 non-null   float64
 1   Sex                908 non-null   object  
 2   ChestPainType       918 non-null   object  
 3   RestingBP           918 non-null   int64   
 4   Cholesterol          918 non-null   int64   
 5   FastingBS           918 non-null   int64   
 6   RestingECG          918 non-null   object  
 7   MaxHR               918 non-null   int64   
 8   ExerciseAngina       918 non-null   object  
 9   Oldpeak             918 non-null   float64  
10   ST_Slope            918 non-null   object  
11   HeartDisease         918 non-null   int64   
dtypes: float64(2), int64(5), object(5)
memory usage: 86.2+ KB
```

B.) Integrasi data dalam konteks teknis data science melibatkan penggabungan atau penyatuan beberapa sumber data menjadi satu dataset yang lengkap dan sesuai dengan kebutuhan analisis yang akan dilakukan. Berikut adalah langkah-langkah umum untuk mengintegrasikan data:

1. **Pemahaman Data:** Pertama, penting untuk memahami struktur dan format data dari setiap sumber yang akan diintegrasikan. Ini mencakup pemahaman kolom, tipe data, dan keterkaitan antar data.
2. **Pembersihan Data:** Lakukan pembersihan data pada setiap sumber data untuk menangani duplikasi, missing values, outliers, dan masalah kualitas data lainnya. Pastikan setiap sumber data siap untuk diintegrasikan.

BUKTI 2-ADS

Kode Unit	:	J.62DMI00.005.1
Judul Unit	:	Menelaah Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam menelaah data untuk data science.

Langkah Kerja:

- 1) Menganalisis tipe dan relasi data
- 2) Menganalisis karakteristik data
- 3) Membuat laporan telaah data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengolah kata
 - Tools pengolahan data
 - Tools pembuat grafik

1. ANALISIS TIPE DAN RELASI DATA

Instruksi Kerja:

- Identifikasi tipe data yang terkumpul sesuai tujuan teknis
- Uraikan nilai atribut data yang terkumpul sesuai dengan batasan konteks bisnisnya
- Identifikasi relasi antar data yang terkumpul sesuai dengan tujuan teknis

Jawab:

A.) Berikut adalah tipe data dari masing-masing kolom:

```
[6] 1 heart_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   Age                   911 non-null   float64
1   Sex                   908 non-null   object  
2   ChestPainType         918 non-null   object  
3   RestingBP             918 non-null   int64   
4   Cholesterol            918 non-null   int64   
5   FastingBS             918 non-null   int64   
6   RestingECG            918 non-null   object  
7   MaxHR                 918 non-null   int64   
8   ExerciseAngina        918 non-null   object  
9   Oldpeak               918 non-null   float64  
10  ST_Slope              918 non-null   object  
11  HeartDisease          918 non-null   int64   
dtypes: float64(2), int64(5), object(5)
memory usage: 86.2+ KB
```

Berdasarkan gambar diatas ini, terdapat 2 jenis tipe data, yakni Numeric dan Kategorikal. Untuk tipe Numeric ada Integer (int64) dan Float (float64). Lalu untuk tipe Kategorikal terdapat tipe Object. Untuk kolom yang memiliki tipe data object ini harus kita lakukan transformasi tipe data menjadi tipe data numeric, agar dapat digunakan kedalam model.

B.) data atribut masih ada yang bersifat object. Harus dilakukan adjustment berupa encoding, agar nantinya fitur/kolom tersebut dapat digunakan kedalam modelling. Untuk gambarnya seperti dibawah ini:

2.2 Encoding of Sex
2.3 Encoding of ChestPainType
2.4 Encoding of RestingECG
2.5 Encoding of ExerciseAngina
2.6 Encoding of ST_Slope

Berdasarkan gambar diatas, saya harus melakukan encoding kepada 5 kolom tersebut karena atribut/tipe datanya masih object, belum bersifat numeric.

- **Encoding of Sex:** pada tahap ini, saya melakukan LabelEncoder kedalam kolom `Sex` untuk mentransform kolom tersebut menjadi Numeric.

```

2.2 Encoding of Sex

[ ] 1 # Membuat objek LabelEncoder
    2 label_encoder = LabelEncoder()
    3
    4 # Melakukan encoding pada kolom 'Sex'
    5 heart_data['Sex_encoded'] = label_encoder.fit_transform(heart_data['Sex'])
    6
    7 # Menampilkan hasil encoding
    8 heart_data[['Sex', 'Sex_encoded']]
  
```

	Sex	Sex_encoded
0	M	1
1	F	0
2	M	1
3	F	0
4	M	1
...

- **Encoding of ChestPainType:** pada tahap ini, saya menggunakan One-Hot-Encoding kedalam kolom `ChestPainType` untuk mentransform kolom tersebut menjadi numeric.

```

2.3 Encoding of ChestPain type

1 # Melakukan One-Hot Encoding pada kolom 'ChestPainType' dengan mengonversi nilai menjadi integer
2 chest_pain_encoded = pd.get_dummies(heart_data['ChestPainType'], prefix='ChestPain', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, chest_pain_encoded], axis=1)
6
7 # Menghapus kolom 'ChestPainType' asli
8 heart_data.drop('ChestPainType', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data
  
```

	Age	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	Ch
0	NaN	140	289	0	Normal	172	N	0.0	Up	0	1	0	
1	49.0	160	180	0	Normal	156	N	1.0	Flat	1	0	0	
2	37.0	130	283	0	ST	98	N	0.0	Up	0	1	0	
3	48.0	138	214	0	Normal	108	Y	1.5	Flat	1	0	1	
4	54.0	150	195	0	Normal	122	N	0.0	Up	0	1	0	
...	
913	45.0	110	264	0	Normal	132	N	1.2	Flat	1	1	0	

- **Encoding of RestingECG:** pada tahap ini, saya menggunakan One-Hot-Encoding kedalam kolom `RestingECG` untuk mentransform kolom tersebut menjadi numeric.

```

2.4 Encoding of RestingECG

1 # Melakukan One-Hot Encoding pada kolom 'RestingECG' dengan penjelasan isi
2 resting_ecg_encoded = pd.get_dummies(heart_data['RestingECG'], prefix='RestingECG', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, resting_ecg_encoded], axis=1)
6
7 # Menghapus kolom 'RestingECG' asli
8 heart_data.drop('RestingECG', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data

```

pe	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST
Jp	0	1	0	1	0	0	0	1	0
lat	1	0	0	0	1	0	0	1	0
Jp	0	1	0	1	0	0	0	0	1
lat	1	0	1	0	0	0	0	1	0
Jp	0	1	0	0	1	0	0	1	0

- **Encoding of ExerciseAngina:** pada tahap ini, saya menggunakan binary encoding kedalam kolom `ExerciseAngina` untuk mentransform kolom tersebut menjadi numeric.

```

[19] 1 # Mengganti nilai 'N' dengan 0 dan 'Y' dengan 1
2 heart_data['ExerciseAngina_encoded'] = heart_data['ExerciseAngina'].replace({'N': 0, 'Y': 1}).astype(int)
3
4 # Menghapus kolom 'ExerciseAngina' asli
5 heart_data.drop('ExerciseAngina', axis=1, inplace=True)
6
7 # Menampilkan hasil encoding
8 heart_data

```

coded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded
1	0	1	0	0	0	1	0	0
0	0	0	1	0	0	1	0	0
1	0	1	0	0	0	0	1	0
0	1	0	0	0	0	1	0	1
1	0	0	1	0	0	1	0	0

- **Encoding of ST_Slope:** pada tahap ini, saya menggunakan One-Hot-Encoding kedalam kolom `ST_Slope` untuk mentransform kolom tersebut menjadi numeric.

```

[22] 1 # Melakukan One-Hot Encoding pada kolom 'ST_Slope' dengan penjelasan isi dan mengonversi nilai menjadi integer
2 st_slope_encoded = pd.get_dummies(heart_data['ST_Slope'], prefix='ST_Slope', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, st_slope_encoded], axis=1)
6
7 # Menghapus kolom 'ST_Slope' asli
8 heart_data.drop('ST_Slope', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data

```

sin_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded	ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
0	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	1
0	0	0	1	0	1	0	1	0
1	0	0	1	0	0	0	0	1

Berikut adalah hasil dari semua adjustment pada atribut kolom:

```

2.7 Adjustment Missing Value

1 heart_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   Age                   911 non-null   float64
 1   RestingBP             918 non-null   int64  
 2   Cholesterol            918 non-null   int64  
 3   FastingBS             918 non-null   int64  
 4   MaxHR                 918 non-null   int64  
 5   Oldpeak               918 non-null   float64
 6   HeartDisease          918 non-null   int64  
 7   Sex_encoded           918 non-null   int64  
 8   ChestPain_ASY         918 non-null   int64  
 9   ChestPain_ATA         918 non-null   int64  
10  ChestPain_NAP         918 non-null   int64  
11  ChestPain_TA          918 non-null   int64  
12  RestingECG_LVH        918 non-null   int64  
13  RestingECG_Normal     918 non-null   int64  
14  RestingECG_ST         918 non-null   int64  
15  ExerciseAngina_encoded 918 non-null   int64  
16  ST_slope_Down         918 non-null   int64  
17  ST_slope_Flat         918 non-null   int64  
18  ST_slope_Up           918 non-null   int64  
dtypes: float64(2), int64(17)
memory usage: 136.4 KB

```

Dapat disimpulkan bahwa atribut/tipe data pada setiap kolom sudah sesuai karena sudah berubah kedalam tipe numeric

C.) Berikut adalah relasi antar data yang telah terkumpul

```

1 # Pilih hanya kolom dengan tipe data numerik
2 numeric_columns = heart_data.select_dtypes(include=['int', 'float']).columns
3
4 # Hitung korelasi antar fitur hanya untuk kolom bertipe numerik
5 corr_numeric = heart_data[numeric_columns].corr(method='pearson', min_periods=1)
6
7 # Tampilkan matriks korelasi
8 print("Correlation Matrix for Numeric Columns:")
9 corr_numeric

```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA
Age	1.000000	0.210568	-0.062391	0.127287	-0.300843	0.196534	0.212565	0.058350	0.144186	-0.157318	-0.031271	0.011865
RestingBP	0.210568	1.000000	0.100893	0.070193	-0.112135	0.164803	0.107589	0.003020	0.048824	-0.046153	-0.041348	0.049855
Cholesterol	-0.062391	0.100893	1.000000	-0.260974	0.235792	0.050148	-0.232741	-0.194308	-0.120531	0.150954	-0.006634	0.017365
FastingBS	0.127287	0.070193	-0.260974	1.000000	-0.131438	0.052698	0.267291	0.122601	0.131176	-0.140514	-0.039249	0.026885
MaxHR	-0.300843	-0.112135	0.235792	-0.131438	1.000000	-0.160691	-0.400421	-0.175378	-0.354963	0.253735	0.134580	0.100025
Oldpeak	0.196534	0.164803	0.050148	0.052698	-0.160691	1.000000	0.403951	0.098522	0.280026	-0.262124	-0.106212	0.032231
HeartDisease	0.212565	0.107589	-0.232741	0.267291	-0.400421	0.403951	1.000000	0.284825	0.516716	-0.401924	-0.212964	-0.054790
Sex_encoded	0.058350	0.003020	-0.194308	0.122601	-0.175378	0.098522	0.284825	1.000000	0.163769	-0.137121	-0.061699	-0.010946
ChestPain_ASY	0.144186	0.048824	-0.120531	0.131176	-0.354963	0.280026	0.516716	0.163769	1.000000	-0.522432	-0.577670	-0.249003
ChestPain_ATA	-0.157318	-0.046153	0.150954	-0.140514	0.253735	-0.262124	-0.401924	-0.137121	-0.522432	1.000000	-0.256767	-0.110679
ChestPain_NAP	-0.031271	-0.041348	-0.006634	-0.039249	0.134580	-0.106212	-0.212964	-0.061699	-0.577670	-0.256767	1.000000	-0.122381
ChestPain_TA	0.011865	0.049855	0.017365	0.026885	0.100025	0.032231	-0.054790	-0.010946	-0.249003	-0.110679	-0.122381	1.000000

Berdasarkan matriks korelasi tersebut, berikut adalah insight khusus mengenai hubungan antara kolom-kolom dengan kolom target HeartDisease:

- **MaxHR (Maximum Heart Rate) (-0.400)**
Korelasi negatif yang kuat dengan HeartDisease menunjukkan bahwa semakin rendah MaxHR, semakin besar kemungkinan seseorang menderita penyakit jantung. Ini dapat disebabkan oleh keterbatasan kemampuan jantung untuk mencapai detak jantung maksimum pada individu dengan penyakit jantung.
- **Oldpeak (Depression induced by exercise relative to rest) (0.404)**

Korelasi positif yang kuat dengan `HeartDisease` menunjukkan bahwa peningkatan `Oldpeak` berkaitan dengan peningkatan risiko penyakit jantung. `Oldpeak` yang tinggi menunjukkan adanya depresi ST segment pada EKG yang sering dihubungkan dengan iskemia miokard.

- **ChestPain_ASY (Asymptomatic chest pain) (0.517)**

Korelasi positif yang kuat dengan `HeartDisease` menunjukkan bahwa pasien dengan nyeri dada asimtomatik memiliki risiko lebih tinggi terkena penyakit jantung. Nyeri dada asimtomatik sering kali merupakan indikator utama dari penyakit arteri koroner.

- **ExerciseAngina_encoded (0.494)**

Korelasi positif dengan `HeartDisease` menunjukkan bahwa adanya angina saat berolahraga berkaitan dengan risiko lebih tinggi untuk penyakit jantung. Ini menggambarkan bahwa angina yang dipicu oleh aktivitas fisik adalah tanda signifikan dari penyakit jantung.

- **ST_Slope_Flat (0.554)**

Korelasi positif yang sangat kuat dengan `HeartDisease` menunjukkan bahwa ST segment yang flat pada EKG berkaitan dengan peningkatan risiko penyakit jantung. ST segment flat menunjukkan adanya iskemia atau infark miokard.

- **ST_Slope_Up (-0.622)**

Korelasi negatif yang sangat kuat dengan `HeartDisease` menunjukkan bahwa ST segment yang naik pada EKG berkaitan dengan risiko lebih rendah terkena penyakit jantung. ST segment yang naik sering kali dianggap normal atau non-ischaemic.

Secara keseluruhan, beberapa variabel memiliki hubungan yang cukup kuat dengan `HeartDisease`, seperti `MaxHR`, `Oldpeak`, `ChestPain_ASY`, `ExerciseAngina_encoded`, `ST_Slope_Flat`, dan `ST_Slope_Up`. Analisis ini memberikan indikasi awal mengenai variabel mana yang mungkin menjadi faktor risiko signifikan untuk penyakit jantung dan dapat digunakan untuk pengembangan model prediktif lebih lanjut.

2. ANALISIS KARAKTERISTIK DATA

Instruksi Kerja:

- Sajikan karakteristik data yang terkumpul dengan deskripsi statistik dasar
- Sajikan karakteristik data yang terkumpul dengan visualisasi grafik
- Analisis karakteristik data dari hasil penyajian data untuk telaah data

Jawab:

A.) Berikut adalah karakteristik data dengan statistic descriptive biasa

```
[ ] 1 heart_data.describe()
```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	Rest
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	54.102086	132.396514	196.799564	0.233115	136.809368	0.887364	0.553377	0.802832	0.540305	0.188453	0.221133	0.050109	
std	12.938724	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414	0.424589	0.498645	0.391287	0.415236	0.218269	
min	0.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	
max	177.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000	2.000000	1.000000	1.000000	1.000000	1.000000	

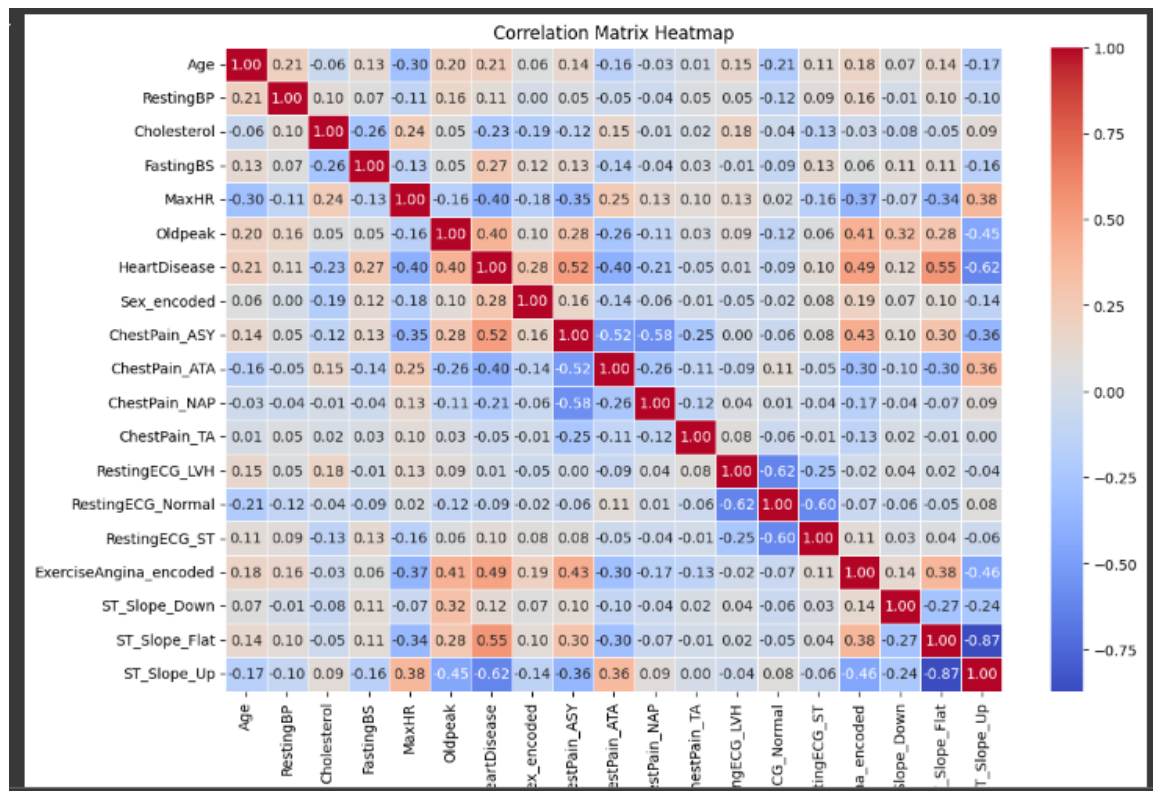
Berdasarkan hasil deskriptif dari method `.describe` di atas, berikut adalah beberapa insight yang dapat diambil:

- **Distribusi Usia (Age):**
 - Rata-rata usia pasien adalah sekitar 54 tahun dengan standar deviasi sekitar 13 tahun.
 - Usia minimum adalah 0 (mungkin data anomali atau kesalahan input), dan usia maksimum adalah 177 (juga tampaknya tidak realistis dan mungkin perlu verifikasi atau pembersihan data).
 - Kuartil menunjukkan bahwa 50% pasien berusia antara 47 dan 60 tahun.
- **Tekanan Darah Istirahat (RestingBP):**
 - Rata-rata tekanan darah istirahat adalah sekitar 132 mmHg dengan standar deviasi sekitar 18 mmHg.
 - Tekanan darah minimum adalah 0, yang tidak realistis dan mungkin merupakan kesalahan input data.
 - Tekanan darah maksimum adalah 200 mmHg.
 - Kuartil menunjukkan bahwa 50% pasien memiliki tekanan darah antara 120 dan 140 mmHg.
- **Kolesterol (Cholesterol):**
 - Rata-rata kadar kolesterol adalah sekitar 199 mg/dL dengan standar deviasi sekitar 109 mg/dL.
 - Kadar kolesterol minimum adalah 0, yang tidak realistis dan menunjukkan kemungkinan adanya data yang hilang atau kesalahan input.
 - Kadar kolesterol maksimum adalah 603 mg/dL.
 - Kuartil menunjukkan bahwa 50% pasien memiliki kadar kolesterol antara 173 dan 267 mg/dL.
- **Gula Darah Puasa (FastingBS):**
 - Rata-rata nilai gula darah puasa adalah sekitar 0.23 (dalam rentang 0 hingga 1, yang mungkin menandakan adanya variabel biner untuk kondisi gula darah puasa tinggi).
 - Standar deviasi adalah sekitar 0.42.
 - Nilai minimum dan maksimum adalah 0 dan 1, yang mengindikasikan bahwa ini adalah variabel biner.
- **Detak Jantung Maksimum (MaxHR):**
 - Rata-rata detak jantung maksimum adalah sekitar 137 bpm dengan standar deviasi sekitar 25 bpm.
 - Detak jantung maksimum minimum adalah 60 bpm, dan maksimum adalah 202 bpm.

- Kuartil menunjukkan bahwa 50% pasien memiliki detak jantung maksimum antara 120 dan 156 bpm.
- **Oldpeak:**
 - Rata-rata Oldpeak adalah sekitar 0.89 dengan standar deviasi sekitar 1.07.
 - Nilai minimum adalah -2.6 (mungkin kesalahan data) dan maksimum adalah 6.2.
 - Kuartil menunjukkan bahwa 50% pasien memiliki Oldpeak antara 0 dan 1.5.
- **Penyakit Jantung (HeartDisease):**
 - Rata-rata nilai HeartDisease adalah sekitar 0.55, mengindikasikan bahwa sekitar 55% pasien dalam dataset ini memiliki penyakit jantung.
 - Nilai minimum adalah 0 dan maksimum adalah 1, yang menunjukkan bahwa ini adalah variabel biner
- **Jenis Kelamin (Sex_encoded):**
 - Rata-rata nilai Sex_encoded adalah sekitar 0.80, mengindikasikan bahwa mayoritas pasien dalam dataset ini adalah pria (diasumsikan bahwa nilai 1 mewakili pria dan 0 mewakili wanita).
- **Jenis Nyeri Dada (ChestPain_ASY, ChestPain_ATA, ChestPain_NAP, ChestPain_TA):**
 - Mayoritas pasien mengalami nyeri dada asimtomatik (ChestPain_ASY) dengan rata-rata sekitar 0.54.
 - Nyeri dada lainnya (ChestPain_ATA, ChestPain_NAP, ChestPain_TA) memiliki nilai rata-rata yang lebih rendah.
- **Resting ECG (RestingECG_LVH, RestingECG_Normal, RestingECG_ST):**
 - Rata-rata nilai RestingECG_Normal adalah sekitar 0.60, mengindikasikan bahwa mayoritas pasien memiliki EKG normal saat istirahat.
- **Exercise Angina (ExerciseAngina_encoded):**
 - Rata-rata nilai ExerciseAngina_encoded adalah sekitar 0.40, mengindikasikan bahwa sekitar 40% pasien mengalami angina saat berolahraga.
- **ST Slope (ST_Slope_Down, ST_Slope_Flat, ST_Slope_Up):**
 - Rata-rata nilai ST_Slope_Flat adalah sekitar 0.50, menunjukkan bahwa mayoritas pasien memiliki slope ST yang flat.
 - Rata-rata nilai ST_Slope_Up adalah sekitar 0.43, menunjukkan bahwa sebagian besar pasien memiliki slope ST yang meningkat.

Insight ini membantu memahami distribusi dan karakteristik dasar dari variabel-variabel dalam dataset, yang penting untuk analisis lebih lanjut dan pembuatan model prediktif.

B.) Berikut adalah karakteristik data dalam bentuk visualisasi grafik



Berdasarkan matriks korelasi tersebut, berikut adalah insight khusus mengenai hubungan antara kolom-kolom dengan kolom target `HeartDisease`:

- MaxHR (Maximum Heart Rate) (-0.400)**
 Korelasi negatif yang kuat dengan `HeartDisease` menunjukkan bahwa semakin rendah MaxHR, semakin besar kemungkinan seseorang menderita penyakit jantung. Ini dapat disebabkan oleh keterbatasan kemampuan jantung untuk mencapai detak jantung maksimum pada individu dengan penyakit jantung.
- Oldpeak (Depression induced by exercise relative to rest) (0.404)**
 Korelasi positif yang kuat dengan `HeartDisease` menunjukkan bahwa peningkatan Oldpeak berkaitan dengan peningkatan risiko penyakit jantung. Oldpeak yang tinggi menunjukkan adanya depresi ST segment pada EKG yang sering dihubungkan dengan iskemia miokard.
- ChestPain_ASY (Asymptomatic chest pain) (0.517)**
 Korelasi positif yang kuat dengan `HeartDisease` menunjukkan bahwa pasien dengan nyeri dada asimtomatik memiliki risiko lebih tinggi terkena penyakit jantung. Nyeri dada asimtomatik sering kali merupakan indikator utama dari penyakit arteri koroner.
- ExerciseAngina_encoded (0.494)**
 Korelasi positif dengan `HeartDisease` menunjukkan bahwa adanya angina saat berolahraga berkaitan dengan risiko lebih tinggi untuk penyakit jantung. Ini menggambarkan bahwa angina yang dipicu oleh aktivitas fisik adalah tanda signifikan dari penyakit jantung.

- **ST_Slope_Flat (0.554)**
Korelasi positif yang sangat kuat dengan `HeartDisease` menunjukkan bahwa ST segment yang flat pada EKG berkaitan dengan peningkatan risiko penyakit jantung. ST segment flat menunjukkan adanya iskemia atau infark miokard.
- **ST_Slope_Up (-0.622)**
Korelasi negatif yang sangat kuat dengan `HeartDisease` menunjukkan bahwa ST segment yang naik pada EKG berkaitan dengan risiko lebih rendah terkena penyakit jantung. ST segment yang naik sering kali dianggap normal atau non-ischaemic.

Secara keseluruhan, beberapa variabel memiliki hubungan yang cukup kuat dengan `HeartDisease`, seperti `MaxHR`, `Oldpeak`, `ChestPain_ASY`, `ExerciseAngina_encoded`, `ST_Slope_Flat`, dan `ST_Slope_Up`. Analisis ini memberikan indikasi awal mengenai variabel mana yang mungkin menjadi faktor risiko signifikan untuk penyakit jantung dan dapat digunakan untuk pengembangan model prediktif lebih lanjut.

C.) Dari hasil penyajian data, kita dapat melihat beberapa karakteristik dari setiap kolom dalam dataframe `heart_data`:

1. **Age:**
 - Rata-rata usia pasien adalah sekitar 54 tahun, dengan rentang usia antara 0 hingga 177 tahun.
 - Standar deviasi sekitar 12.94, menunjukkan variasi yang cukup besar dalam usia pasien.
2. **RestingBP (Tekanan Darah Istirahat):**
 - Rata-rata tekanan darah istirahat adalah sekitar 132.40 mm Hg, dengan rentang antara 0 hingga 200 mm Hg.
 - Standar deviasi sekitar 18.51, menunjukkan variasi yang signifikan dalam tekanan darah istirahat.
3. **Cholesterol:**
 - Rata-rata kolesterol serum adalah sekitar 198.80 mm/dl, dengan rentang antara 0 hingga 603 mm/dl.
 - Standar deviasi sekitar 109.38, menunjukkan variasi yang cukup besar dalam kolesterol serum.
4. **FastingBS (Gula Darah Puasa):**
 - Sebagian besar pasien memiliki gula darah puasa di bawah 120mg/dl (karena nilai median adalah 0 dan Q3 adalah 0).
 - Sekitar 23% dari pasien memiliki gula darah puasa di atas 120mg/dl.
5. **MaxHR (Detak Jantung Maksimum):**
 - Rata-rata detak jantung maksimum adalah sekitar 136.81, dengan rentang antara 60 hingga 202.
 - Standar deviasi sekitar 25.46, menunjukkan variasi yang signifikan dalam detak jantung maksimum.
6. **Oldpeak:**
 - Rata-rata depresi ST yang dihasilkan oleh latihan relatif terhadap istirahat adalah sekitar 0.89.
 - Standar deviasi sekitar 1.07, menunjukkan variasi yang cukup besar dalam depresi ST.
7. **HeartDisease (Penyakit Jantung):**

BUKTI 3-ADS

Kode Unit	:	J.62DMI00.006.1
Judul Unit	:	Memvalidasi Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam memvalidasi data untuk data science.

Langkah Kerja:

- 1) Melakukan pengecekan kelengkapan data
- 2) Membuat rekomendasi kelengkapan data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengubah teks

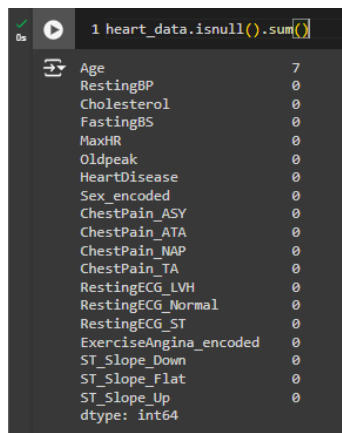
1. PENGECEKAN KELENGKAPAN DATA

Instruksi Kerja:

- Sajikan penilaian kualitas data dari hasil telaah sesuai tujuan teknis data science
- Sajikan penilaian tingkat kecukupan data dari hasil telaah sesuai tujuan teknis data science

Jawab:

A.) Penilaian kualitas data masih ada yang harus di adjust lagi, karena masih ada kolom yang missing value dan masih ada outliers



```
1 heart_data.isnull().sum()
```

Age	7
RestingBP	0
Cholesterol	0
FastingBS	0
MaxHR	0
Oldpeak	0
HeartDisease	0
Sex_encoded	0
ChestPain_ASY	0
ChestPain_ATA	0
ChestPain_NAP	0
ChestPain_TA	0
RestingECG_LVH	0
RestingECG_Normal	0
RestingECG_ST	0
ExerciseAngina_encoded	0
ST_Slope_Down	0
ST_Slope_Flat	0
ST_Slope_Up	0
dtype: int64	

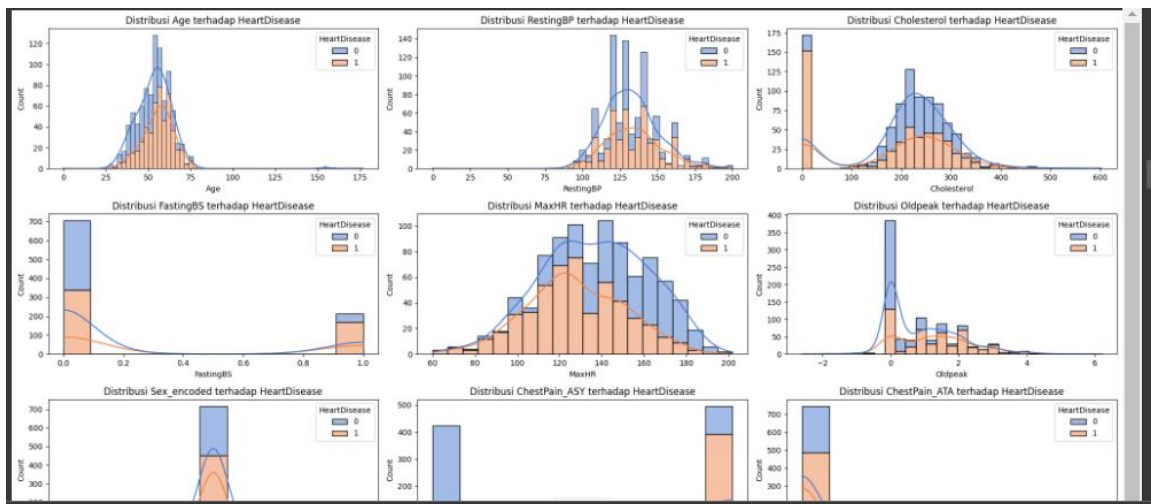
bisa dilihat bahwa ada 1 kolom yang masih ada missing value yaitu kolom Age

Berikut adalah cara melakukan adjustment missing value, saya menggunakan mean untuk mengisi baris yang missing pada kolom Age

```
[29] 1 # Mengisi nilai-nilai yang hilang dengan mean dari setiap kolom dengan inplace=True
      2 heart_data.fillna(heart_data.mean(), inplace=True)

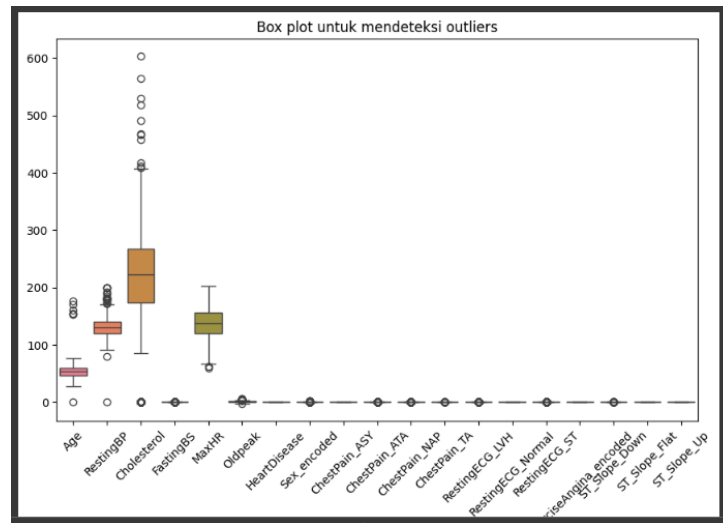
1 heart_data.isnull().sum()

Age                0
RestingBP          0
Cholesterol        0
FastingBS         0
MaxHR             0
Oldpeak           0
HeartDisease       0
Sex_encoded        0
ChestPain_ASY      0
ChestPain_ATA      0
ChestPain_NAP      0
ChestPain_TA       0
RestingECG_LVH     0
RestingECG_Normal  0
RestingECG_ST      0
ExerciseAngina_encoded 0
ST_Slope_Down      0
ST_Slope_Flat      0
ST_Slope_Up        0
dtype: int64
```



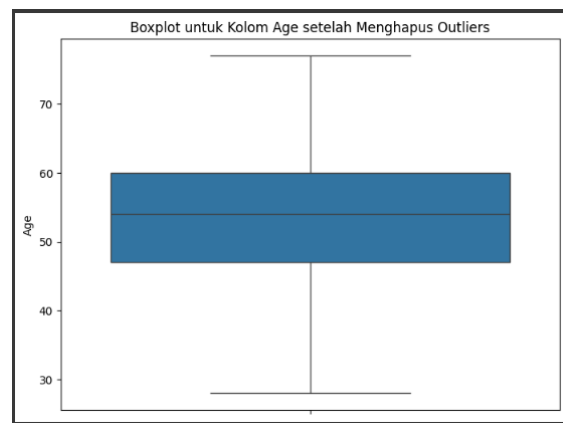
Berdasarkan gambar diatas, saya mengecek distribusi data, dan menurut saya, sudah cukup baik

Step selanjutnya adalah melakukan adjustment terhadap outliers

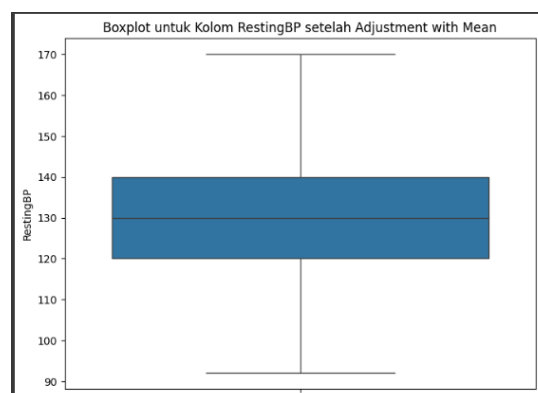


Berdasarkan gambar diatas, harus dilakukan adjustment terhadap 3 kolom (Age, RestingBP, Cholesterol) karena memiliki outliers yang besar

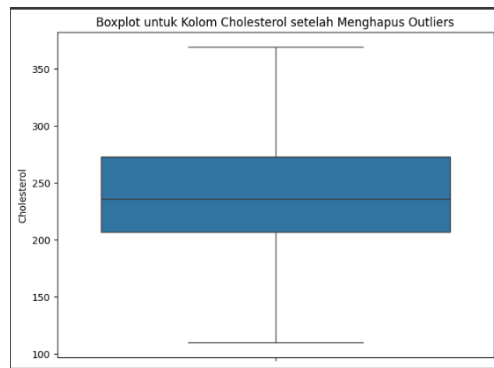
Selanjutnya saya melakukan adjustment outliers pada kolom Age dengan metode IQR. Setelah itu saya drop outliers nya, sehingga tampilannya akan seperti ini



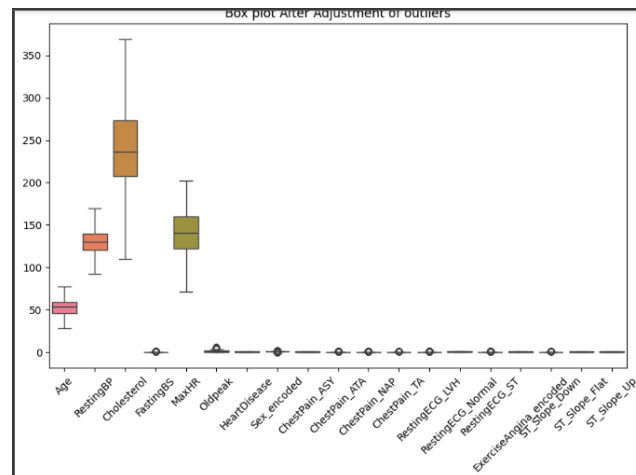
Selanjutnya adalah RestingBP, saya menggunakan metode IQR untuk adjustment terhadap outliers, lalu saya mengubah outliers tersebut ke dalam mean. Hasilnya akan seperti gambar boxplot dibawah ini:



Selanjutnya, yaitu kolom Cholesterol, saya melakukan adjustment outliers menggunakan metode IQR. Setelah itu, saya hapus outliersnya. Hasil boxplotnya seperti gambar dibawah ini:



Setelah itu, saya melakukan recheck terhadap data, dan hasilnya aman, outliers sudah teratasi. Berikut adalah gambarnya:



Dengan demikian, kualitas data sudah terjamin bagus.

B.) Berdasarkan analisis karakteristik data yang telah dilakukan, kita dapat menyajikan penilaian tingkat kecukupan data untuk tujuan teknis data science, yaitu membangun model klasifikasi penyakit jantung menggunakan algoritma Decision Tree dan XGBoost.

Penilaian Tingkat Kecukupan Data dari Hasil Telaah

Deskripsi data:

Berdasarkan hasil `heart_data.describe()`, kita memiliki gambaran umum tentang statistik deskriptif dari dataset yang digunakan. Berikut adalah ringkasan dari tiap kolom:

- **Age:** Rentang usia pasien dari 28 hingga 77 tahun dengan rata-rata sekitar 52.79 tahun.

- **RestingBP:** Tekanan darah istirahat berkisar dari 92 hingga 170 mm Hg dengan rata-rata sekitar 131.51 mm Hg.
- **Cholesterol:** Kolesterol serum berkisar dari 110 hingga 369 mg/dL dengan rata-rata sekitar 239.78 mg/dL.
- **FastingBS:** 16.57% dari pasien memiliki gula darah puasa lebih dari 120 mg/dL.
- **MaxHR:** Detak jantung maksimum berkisar dari 71 hingga 202 bpm dengan rata-rata sekitar 140.52 bpm.
- **Oldpeak:** Depresi ST yang dihasilkan oleh latihan relatif terhadap istirahat berkisar dari -0.1 hingga 6.2 dengan rata-rata sekitar 0.9.
- **HeartDisease:** 47.35% dari pasien memiliki penyakit jantung.
- **Sex_encoded:** Rasio jenis kelamin menunjukkan 77.30% pria dan sisanya wanita.
- **ChestPainType (encoded):** Variabel dummy yang menunjukkan berbagai jenis nyeri dada.
- **RestingECG (encoded):** Variabel dummy yang menunjukkan hasil EKG istirahat.
- **ExerciseAngina_encoded:** 38.16% dari pasien mengalami angina yang diinduksi oleh latihan.
- **ST_Slope (encoded):** Variabel dummy yang menunjukkan kemiringan segmen ST.

Penilaian Tingkat Kecukupan Data:

1. Ketersediaan data

- Dataset ini terdiri dari 718 observasi yang mencakup fitur-fitur yang relevan untuk analisis penyakit jantung.
- Setiap fitur memiliki jumlah observasi yang sama (718), menunjukkan tidak ada data yang hilang di dataset ini.

2. Variabilitas dan Distribusi Data

- **Age:** Variabilitas yang baik dengan rentang usia yang luas dan distribusi yang wajar.
- **RestingBP** dan **Cholesterol:** Kedua fitur ini menunjukkan variabilitas yang cukup dan rentang nilai yang luas, memberikan informasi yang signifikan tentang kondisi kesehatan pasien.
- **MaxHR:** Variasi dalam detak jantung maksimum juga memberikan informasi penting untuk analisis.
- **Oldpeak:** Rentang dan distribusi nilai menunjukkan adanya variasi kondisi kesehatan yang signifikan di antara pasien.
- **HeartDisease:** Proporsi pasien dengan dan tanpa penyakit jantung hampir seimbang, yang baik untuk klasifikasi.
- **Fitur Kategorikal (encoded):** Variabel dummy yang dihasilkan dari fitur kategorikal menunjukkan distribusi yang beragam, yang penting untuk analisis model klasifikasi.

3. Kesiapan untuk Pemodelan

- Dataset ini telah di-encode dengan baik untuk fitur kategorikal, membuatnya siap untuk digunakan dalam algoritma machine learning.
- Fitur numerik memiliki distribusi yang wajar tanpa adanya indikasi masalah data yang hilang atau outlier yang ekstrim.
- Dataset ini memiliki cukup banyak data dan variabilitas untuk membangun model yang dapat generalisasi dengan baik.

Kesimpulan:

Secara keseluruhan, dataset `heart_data` memiliki tingkat kecukupan yang baik untuk tujuan teknis data science, yaitu membangun model klasifikasi penyakit jantung. Data ini mencakup volume yang memadai, variabilitas yang cukup, dan distribusi fitur yang relevan untuk analisis penyakit jantung. Namun, diperlukan pemeriksaan lebih lanjut untuk beberapa nilai ekstrim dan mungkin perlu dilakukan normalisasi atau standarisasi tambahan.

2. REKOMENDASI KELENGKAPAN DATA

Instruksi Kerja:

- Susun rekomendasi hasil penilaian kualitas sesuai tujuan teknis data science
- Susun rekomendasi hasil penilaian kecukupan data sesuai tujuan teknis data science

Jawab:

A.) Berdasarkan hasil penilaian kualitas data dari dataset `heart_data`, berikut adalah beberapa rekomendasi untuk memastikan bahwa data yang digunakan sesuai dengan tujuan teknis data science:

Validasi dan Pembersihan Data:

- **Validasi Input Data:** Pastikan bahwa data yang di-input telah diverifikasi dan divalidasi dengan benar untuk menghindari kesalahan input.

Normalisasi atau Standarisasi Data:

- **Normalisasi Fitur:** Mengingat variasi dalam rentang nilai fitur numerik, normalisasi atau standarisasi fitur ini penting untuk memastikan bahwa mereka berkontribusi secara seimbang dalam algoritma machine learning. Gunakan teknik seperti `StandardScaler` atau `MinMaxScaler` dari `scikit-learn`.
- **Handling Imbalance:** Jika diperlukan, gunakan teknik balancing data seperti SMOTE (Synthetic Minority Over-sampling Technique) untuk mengatasi ketidakseimbangan kelas dalam dataset.

Transformasi Fitur Kategorikal:

- **Encoding Fitur Kategorikal:** Pastikan semua fitur kategorikal telah di-encode dengan benar. Penggunaan `OneHotEncoder` sudah sesuai, tetapi pastikan tidak ada informasi yang hilang dalam proses encoding.
- **Interaksi Fitur:** Pertimbangkan untuk menambahkan fitur interaksi jika relevan, misalnya interaksi antara usia dan detak jantung maksimum, yang dapat memberikan informasi tambahan untuk model.

Pengujian dan Validasi Model:

- **Hyperparameter Tuning:** Lakukan tuning hyperparameter menggunakan GridSearchCV atau RandomizedSearchCV untuk menemukan kombinasi parameter yang optimal bagi model.

Analisis Korelasi dan Reduksi Dimensi:

- **Analisis Korelasi:** Terus lakukan analisis korelasi untuk mengidentifikasi fitur-fitur yang mungkin saling berkorelasi tinggi.
- **PCA (Principal Component Analysis):** Pertimbangkan untuk menggunakan PCA jika jumlah fitur menjadi terlalu banyak dan mengakibatkan peningkatan kompleksitas model tanpa peningkatan kinerja yang signifikan.

Evaluasi:

- **Evaluasi Kinerja Model:** Selain menggunakan metrik akurasi, gunakan metrik tambahan seperti precision, recall, F1-score, dan AUC-ROC untuk evaluasi kinerja model secara lebih komprehensif.

Kesimpulan: Dengan mengikuti rekomendasi ini, kualitas data untuk analisis penyakit jantung dapat ditingkatkan, sehingga model yang dibangun dapat memberikan hasil yang lebih akurat dan andal. Langkah-langkah ini akan memastikan bahwa data memenuhi tujuan teknis data science dan model yang dihasilkan dapat digunakan secara efektif dalam praktek klinis atau penelitian lebih lanjut.

B.) Berdasarkan hasil analisis data heart_data, berikut adalah rekomendasi untuk memastikan kecukupan data sesuai dengan tujuan teknis data science:

Jumlah Observasi dan Variabilitas Data:

- **Penambahan Data:** Jika memungkinkan, tambahkan lebih banyak data untuk meningkatkan generalisasi model. Data tambahan dapat diperoleh melalui kolaborasi dengan institusi medis lainnya atau menggunakan data publik yang tersedia.
- **Variabilitas Data:** Pastikan bahwa data yang ditambahkan memiliki variasi yang cukup dan tidak hanya data dari satu sumber atau populasi tertentu. Hal ini penting untuk meningkatkan generalisasi model.

Kelengkapan Data:

- **Mengatasi Missing Values:** Identifikasi dan tangani nilai yang hilang (missing values) dalam dataset. Gunakan teknik imputasi yang sesuai (mean, median, mode, atau menggunakan model prediktif) untuk menggantikan nilai yang hilang.
- **Validasi Input Data:** Implementasikan prosedur validasi untuk memastikan data baru yang diinput tidak mengandung missing values yang signifikan.

Representasi Populasi:

- **Representasi Populasi yang Beragam:** Pastikan bahwa dataset mencakup representasi yang beragam dari populasi, termasuk berbagai kelompok usia, jenis kelamin, ras, dan kondisi medis. Ini penting untuk memastikan bahwa model dapat digunakan secara umum.
- **Penambahan Data dari Sumber yang Berbeda:** Upayakan untuk mendapatkan data dari berbagai sumber untuk memastikan representasi yang lebih luas dari populasi.

Kualitas Data:

- **Preprocessing Data:** Lakukan preprocessing data seperti normalisasi, standarisasi, dan encoding fitur kategorikal secara konsisten. Ini memastikan bahwa data dalam format yang sesuai untuk analisis dan pemodelan.
- **Pembersihan Data:** Lakukan pembersihan data secara berkala untuk mengidentifikasi dan memperbaiki data yang tidak valid atau outliers yang tidak sesuai.

Penggunaan Data Sintetis:

- **Data Augmentation:** Jika pengumpulan data tambahan sulit dilakukan, pertimbangkan untuk menggunakan teknik data augmentation atau pembuatan data sintetis untuk meningkatkan volume dan variabilitas data.

Kesimpulan: Dengan mengikuti rekomendasi ini, kecukupan data untuk analisis penyakit jantung dapat ditingkatkan, sehingga model yang dibangun lebih akurat dan dapat diandalkan. Langkah-langkah ini memastikan bahwa data memenuhi tujuan teknis data science, memberikan fondasi yang kuat untuk pengembangan model prediktif yang efektif.

BUKTI 4-ADS

Kode Unit	:	J.62DMI00.007.1
Judul Unit	:	Menentukan Objek Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam memilah dan memilih data yang sesuai permintaan atau kebutuhan.

Langkah Kerja:

- 1) Memutuskan kriteria dan teknik pemilihan data
- 2) Menentukan attributes (columns) dan records (row) data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengolah kata
 - Aplikasi spreadsheet
 - Aplikasi notepad plus
 - Aplikasi SQL (Structured Query Language)

1. KRITERIA DAN TEKNIK PEMILIHAN DATA

Instruksi Kerja:

- Identifikasi kriteria pemilihan data sesuai dengan tujuan teknis dan aturan yang berlaku
- Tetapkan teknik pemilihan data sesuai dengan kriteria pemilihan data

Jawab:

A.) Berdasarkan tujuan teknis untuk membangun model klasifikasi penyakit jantung menggunakan algoritma Decision Tree dan XGBoost, serta aturan yang berlaku, berikut adalah kriteria pemilihan data:

Keterkaitan dengan Tujuan Teknis:

- **Relevansi Fitur:** Pilih fitur yang relevan dengan diagnosis dan prediksi penyakit jantung
- **Variabilitas Fitur:** Pilih fitur yang memiliki variabilitas yang cukup untuk memberikan informasi yang berguna bagi model. Fitur yang memiliki nilai konstan atau hampir konstan mungkin tidak memberikan informasi tambahan.
- **Korelasi dengan Target:** Pilih fitur yang memiliki korelasi signifikan dengan target (kolom HeartDisease). Analisis korelasi dapat membantu mengidentifikasi fitur yang penting untuk prediksi.

Kualitas Data:

- **Completeness:** Pilih data yang memiliki sedikit atau tidak ada nilai yang hilang (missing values). Fitur dengan banyak missing values mungkin memerlukan imputasi atau penghapusan.
- **Keakuratan dan Validitas:** Pastikan data yang digunakan adalah akurat dan valid. Data yang tidak akurat dapat mempengaruhi performa model secara negatif.

B.) Berdasarkan kriteria pemilihan data yang telah ditetapkan, berikut adalah teknik pemilihan data yang sesuai:

Relevansi Fitur:

- **Teknik:** Seleksi manual berdasarkan domain knowledge dan analisis statistik.
- **Implementasi:** Pilih fitur yang secara domain knowledge relevan untuk diagnosis penyakit jantung dan memiliki korelasi signifikan dengan target.

Completeness:

- **Teknik:** Handling missing values.
- **Implementasi:** Menghapus baris dengan missing values atau melakukan imputasi.
Untuk gambar, seperti gambar-gambar diawal (penanganan missing values)

Keakuratan dan Validitas:

- **Teknik:** Validasi data dengan domain experts dan menggunakan aturan bisnis.
- **Implementasi:** Melakukan pengecekan dan pembersihan data yang tidak valid atau outliers. Pada gambar dibawah ini, saya menghapus nilai yang kurang dari 0 dan sama dengan 0 pada ketiga kolom tersebut, karena berdasarkan analisa saya, tidak mungkin kolom tersebut memiliki nilai seperti itu.

```
0s [62] 1 # Menghapus nilai yang tidak masuk akal secara in-place
      2 heart_data.drop(heart_data[(heart_data['Age'] <= 0) |
      3                 (heart_data['RestingBP'] <= 0) |
      4                 (heart_data['Cholesterol'] <= 0)].index, inplace=True)
```

2. ATTRIBUTES (COLUMNS) DAN RECORDS (ROW) DATA

Instruksi Kerja:

- Identifikasi attributes (columns) data sesuai dengan kriteria pemilihan data
- Identifikasi records (row) data sesuai dengan kriteria pemilihan data

Jawab:

A.) Berdasarkan proses sebelumnya dan tujuan teknis data science untuk membangun model klasifikasi penyakit jantung, berikut adalah identifikasi atribut (kolom) data sesuai dengan kriteria pemilihan data:

- **Age:** Kolom ini sangat relevan karena usia pasien sering dikaitkan dengan risiko penyakit jantung.
- **Sex:** Jenis kelamin pasien (encoded) juga penting karena pria dan wanita memiliki risiko penyakit jantung yang berbeda.
- **ChestPainType:** Jenis nyeri dada (encoded) merupakan faktor penting dalam mendiagnosis penyakit jantung.
- **RestingBP:** Tekanan darah istirahat adalah indikator penting kesehatan jantung.
- **Cholesterol:** Kadar kolesterol dalam darah dapat menunjukkan risiko penyakit jantung.
- **FastingBS:** Gula darah puasa (encoded) yang tinggi merupakan faktor risiko untuk penyakit jantung.
- **RestingECG:** Hasil elektrokardiogram saat istirahat (encoded) dapat menunjukkan kelainan jantung.
- **MaxHR:** Detak jantung maksimum adalah indikator kapasitas jantung saat beraktivitas.
- **ExerciseAngina:** Angina yang diinduksi oleh latihan (encoded) menunjukkan respon jantung terhadap aktivitas fisik.
- **Oldpeak:** Depresi ST yang dihasilkan oleh latihan relatif terhadap istirahat, yang merupakan indikator kesehatan jantung.
- **ST_Slope:** Kemiringan segmen ST (encoded) penting untuk diagnosis penyakit jantung.
- **HeartDisease:** Output class yang menunjukkan apakah pasien menderita penyakit jantung atau tidak.

Kolom yang telah diencode:

- **Sex_encoded**
- **ChestPain_ASY, ChestPain_ATA, ChestPain_NAP, ChestPain_TA** (dari ChestPainType)
- **RestingECG_LVH, RestingECG_Normal, RestingECG_ST** (dari RestingECG)
- **ExerciseAngina_encoded**
- **ST_Slope_Down, ST_Slope_Flat, ST_Slope_Up** (dari ST_Slope)

B.) Mengidentifikasi records (baris) data sesuai dengan kriteria pemilihan data melibatkan beberapa langkah penting untuk memastikan hanya data yang relevan dan valid digunakan dalam analisis dan pengembangan model. Berdasarkan pekerjaan sebelumnya, kita telah membersihkan data dengan menghapus nilai yang tidak masuk akal dan menangani nilai yang hilang. Berikut adalah langkah-langkah tambahan untuk mengidentifikasi dan menyeleksi records sesuai kriteria pemilihan data:

Menghapus duplikasi:

- ❖ Menghapus baris duplikat dari dataset untuk memastikan bahwa setiap entri unik dan tidak ada pengulangan data yang tidak perlu.

Menangani Outliers:

- ❖ Mengidentifikasi dan menangani outliers yang dapat mempengaruhi model secara signifikan. Outliers dapat dihapus atau ditangani sesuai dengan kebijakan tertentu (misalnya, capping, transformation).

Validasi Nilai: Memastikan bahwa setiap kolom memiliki nilai yang berada dalam rentang yang diharapkan dan valid.

BUKTI 5-ADS

Kode Unit	:	J.62DMI00.008.1
Judul Unit	:	Membersihkan Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam membersihkan data yang sesuai permintaan atau kebutuhan.

Langkah Kerja:

- 1) Melakukan pembersihan data yang kotor
- 2) Membuat laporan dan rekomendasi hasil membersihkan data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengolah kata
 - Aplikasi spreadsheet
 - Aplikasi text editor
 - Aplikasi SQL (Structured Query Language)

1. PEMBERSIHAN DATA KOTOR

Instruksi Kerja:

- Tentukan strategi pembersihan data berdasarkan hasil telaah data
- Koreksi data yang kotor berdasarkan strategi pembersihan data

Jawab:

A.) Berdasarkan hasil telaah data, kita dapat menetapkan strategi pembersihan data yang sesuai untuk memastikan data yang digunakan dalam analisis dan pengembangan model adalah data yang bersih dan valid. Berikut adalah strategi pembersihan data yang dapat diterapkan:

Menghapus Duplikasi:

- ❖ Menggunakan metode `.drop_duplicates()` untuk menghapus baris duplikat dari dataset. Ini memastikan bahwa setiap entri unik hanya muncul satu kali dalam dataset.

```
[63] 1 heart_data.drop_duplicates(inplace=True)
```

Menangani Outliers:

- ❖ Menggunakan metode Interquartile Range (IQR) untuk mengidentifikasi dan menghapus outliers dari kolom numerik yang relevan, seperti `Age`, `RestingBP`, dan `Cholesterol`. Outliers dapat memiliki dampak yang signifikan pada analisis dan model, oleh karena itu perlu ditangani dengan hati-hati.

Validasi Nilai:

- ❖ Memeriksa dan memvalidasi nilai-nilai dalam setiap kolom untuk memastikan bahwa mereka dalam rentang yang diharapkan dan valid. Misalnya, memastikan bahwa nilai usia (`Age`), tekanan darah istirahat (`RestingBP`), dan kolesterol serum (`Cholesterol`) tidak memiliki nilai yang tidak masuk akal atau negatif.

B.) Berikut adalah langkah-langkah untuk melakukan pembersihan data berdasarkan strategi yang telah diidentifikasi sebelumnya:

Menghapus Duplikasi:

```
✓ [63] 1 heart_data.drop_duplicates(inplace=True)
```

Berdasarkan gambar diatas, saya melakukan penghapusan data yang duplikat pada dataframe `heart_data`

Menangani Outliers:

3.1 Outliers of Age

```
1 # Hitung Q1 dan Q3
2 Q1 = heart_data['Age'].quantile(0.25)
3 Q3 = heart_data['Age'].quantile(0.75)
4
5 # Hitung IQR
6 IQR = Q3 - Q1
7
8 # Tentukan batas atas dan batas bawah
9 upper_bound = Q3 + 1.5 * IQR
10 lower_bound = Q1 - 1.5 * IQR
11
12 # Identifikasi outliers
13 outliers = heart_data[(heart_data['Age'] < lower_bound) | (heart_data['Age'] > upper_bound)]
14
15 # Tampilkan outliers
16 print("Outliers:")
17 outliers
```

Outliers:

Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain
74	155.0	140	268	0	128	1.5	1	1	1	0	0

Berdasarkan gambar diatas, saya melakukan adjustment kolom `Age` terhadap outliers dengan menggunakan metode IQR (Interquartile Range).

3.2 Outliers of RestingBP

```
[39] 1 # Hitung Q1 dan Q3
      2 Q1 = heart_data['RestingBP'].quantile(0.25)
      3 Q3 = heart_data['RestingBP'].quantile(0.75)
      4
      5 # Hitung IQR
      6 IQR = Q3 - Q1
      7
      8 # Tentukan batas atas dan batas bawah
      9 upper_bound = Q3 + 1.5 * IQR
     10 lower_bound = Q1 - 1.5 * IQR
     11
     12 # Identifikasi outliers_restingbp
     13 outliers_restingbp = heart_data[(heart_data['RestingBP'] < lower_bound) | (heart_data['RestingBP'] > upper_bound)]
     14
     15 # Tampilkan outliers_restingbp
     16 print("outliers_restingbp:")
     17 outliers_restingbp
```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_MAP	ChestPain_
109	39.0	190	241	0	106	0.0	0	1	0	1	0	

Berdasarkan gambar diatas, saya melakukan adjustment kolom `RestingBP` terhadap outliers dengan menggunakan metode IQR (Interquartile Range).

3.3 Outliers of Cholesterol

```
[42] 1 # Hitung Q1 dan Q3
      2 Q1 = heart_data['Cholesterol'].quantile(0.25)
      3 Q3 = heart_data['Cholesterol'].quantile(0.75)
      4
      5 # Hitung IQR
      6 IQR = Q3 - Q1
      7
      8 # Tentukan batas atas dan batas bawah
      9 upper_bound = Q3 + 1.5 * IQR
     10 lower_bound = Q1 - 1.5 * IQR
     11
     12 # Identifikasi outliers_Cholesterol
     13 outliers_Cholesterol = heart_data[(heart_data['Cholesterol'] < lower_bound) | (heart_data['Cholesterol'] > upper_bound)]
     14
     15 # Tampilkan outliers_Cholesterol
     16 print("outliers_Cholesterol:")
     17 outliers_Cholesterol
```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_MAP	ChestPain_
28	53.0	113.0	468	0	127	0.0	0	0	0	1	0	

Berdasarkan gambar diatas, saya melakukan adjustment kolom `Cholesterol` terhadap outliers dengan menggunakan metode IQR (Interquartile Range).

Validasi Nilai:

```
[62] 1 # Menghapus nilai yang tidak masuk akal secara in-place
      2 heart_data.drop(heart_data[(heart_data['Age'] <= 0) |
      3                     (heart_data['RestingBP'] <= 0) |
      4                     (heart_data['Cholesterol'] <= 0)].index, inplace=True)
```

Berdasarkan gambar diatas, saya memvalidasi ketiga kolom tersebut agar tidak mempunyai nilai yang tidak masuk akal.

Menangani Missing Values:

```
✓ [64] 1 # Menghapus baris yang mengandung nilai yang hilang  
0s      2 heart_data.dropna(inplace=True)
```

Berdasarkan gambar diatas, saya menghapus kolom yang memiliki baris yang missing, lalu saya menyimpan hasilnya yang sudah bersih kedalam dataframe heart_data dengan inplace=true

2. LAPORAN DAN REKOMENDASI HASIL PEMBERSIHAN DATA KOTOR

Instruksi Kerja:

- Deskripsikan masalah dan teknis koreksi data sesuai dengan kondisi data dan strategi pembersihan data
- Lakukan evaluasi berdasarkan analisis koreksi yang telah dilakukan
- Dokumentasikan evaluasi proses dan hasil pembersihan data kotor

Jawab:

A.) Masalah umum dalam data dapat mencakup duplikasi, outlier, nilai yang tidak masuk akal, dan nilai yang hilang. Untuk setiap masalah, ada beberapa teknik koreksi data yang dapat diterapkan:

Duplikasi:

- ❖ Masalah: Duplikasi mengakibatkan redundansi dalam data, yang dapat mempengaruhi analisis dan model yang dibangun.
- ❖ Koreksi: Menghapus baris yang merupakan duplikat dari data.

Outlier:

- ❖ Masalah: Outlier dapat mengganggu analisis dan model, serta menyebabkan hasil yang bias atau tidak akurat.
- ❖ Koreksi: Menghapus outlier berdasarkan kriteria yang telah ditentukan atau menyesuaikan nilainya jika diperlukan.

Nilai Tidak Masuk Akal:

- ❖ Masalah: Nilai yang tidak masuk akal, seperti nilai negatif untuk atribut yang semestinya selalu positif, dapat mengganggu interpretasi dan analisis data.
- ❖ Koreksi: Menghapus baris yang mengandung nilai yang tidak masuk akal atau mengubahnya menjadi nilai yang masuk akal jika memungkinkan.

Nilai Hilang:

- ❖ Masalah: Nilai yang hilang dapat mengurangi kualitas data dan menghambat analisis yang akurat.
- ❖ Koreksi: Mengisi nilai yang hilang dengan nilai yang sesuai, seperti menggunakan imputasi atau interpolasi, atau menghapus baris yang mengandung nilai yang hilang jika memungkinkan.

B.) Evaluasi analisis koreksi data dapat dilakukan dengan beberapa langkah:

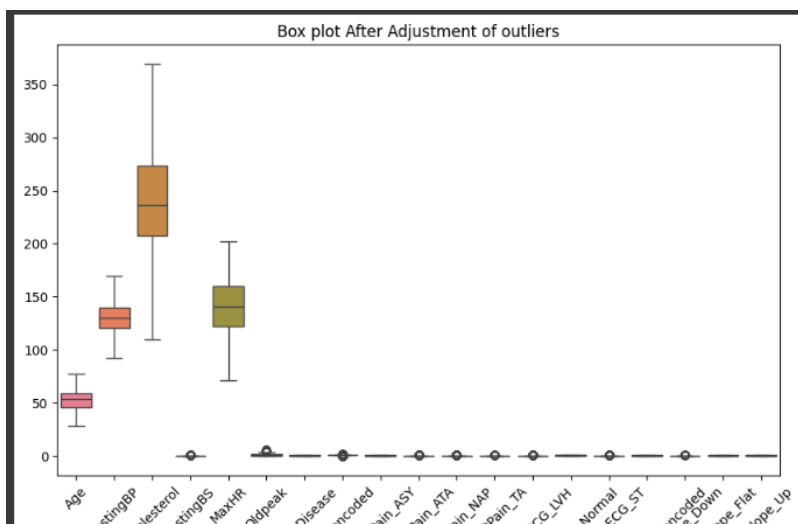
Periksa Perubahan: Periksa perubahan dalam distribusi data setelah koreksi. Perhatikan apakah ada peningkatan dalam kualitas data atau penurunan outlier setelah pembersihan.

Perbandingan dengan Standar: Bandingkan hasil setelah koreksi dengan standar atau pedoman industri yang relevan. Perhatikan apakah data sekarang memenuhi kriteria yang diperlukan.

C.) Dokumentasi evaluasi proses dan hasil pembersihan data kotor bisa terstruktur dalam beberapa bagian:

Tujuan Evaluasi: Menjelaskan tujuan dari evaluasi ini, yaitu untuk memverifikasi efektivitas proses pembersihan data dan menilai apakah data yang diperoleh sudah cukup bersih untuk digunakan dalam analisis lebih lanjut.

Tujuan dari adjustment outliers, verifikasi data, penghapusan duplikat, dan penghapusan missing value, bertujuan agar data yang nantinya akan diolah pada saat modelling, hasilnya dapat lebih akurat.



Berdasarkan dari boxplot diatas, data sudah bagus, karena outliers sudah di adjust.

```

[62] 1 # Menghapus nilai yang tidak masuk akal secara in-place
      2 heart_data.drop(heart_data[(heart_data['Age'] <= 0) |
      3                     (heart_data['RestingBP'] <= 0) |
      4                     (heart_data['Cholesterol'] <= 0)].index, inplace=True)

[63] 1 heart_data.drop_duplicates(inplace=True)

[64] 1 # Menghapus baris yang mengandung nilai yang hilang
      2 heart_data.dropna(inplace=True)

```

Berdasarkan gambar diatas pula, dapat mengindikasikan bahwa data sudah sangat bagus, karena sudah di adjust oleh berbagai metode.

Metode Evaluasi: Menjelaskan metode yang digunakan untuk mengevaluasi proses pembersihan data, termasuk teknik analisis yang diterapkan, kriteria yang digunakan untuk menilai kebersihan data, dan alat atau platform yang digunakan dalam proses evaluasi.

Untuk mengevaluasi proses pembersihan, saya menggunakan saya menggunakan IQR selanjutnya boxplot untuk melihat apakah outliers sudah di adjust atau tidak. Lalu juga dilakukan verifikasi data, drop duplicate, dan drop missing value. Untuk gambarnya sama seperti gambar-gambar diatas.

Hasil Evaluasi: Laporkan hasil dari evaluasi tersebut, termasuk temuan utama yang ditemukan selama evaluasi. Ini bisa mencakup perubahan yang terjadi dalam distribusi data, perbaikan spesifik yang dilakukan, dan peningkatan dalam kualitas data setelah pembersihan.

Untuk hasil evaluasi, dari outliers sudah aman. Lalu dari verifikasi data sempat kecolongan, tapi untungnya sudah aman berkat fuction describe() untuk mengetahui data yang tidak normal lalu saya lakukan adjustment agar tidak ada nilai 0 pada beberapa kolom khusus (Age, RestingBP, Cholesterol). Lalu untuk drop duplicate dan drop missing value sudah aman.

BUKTI 6-ADS

Kode Unit	:	J.62DMI00.009.1
Judul Unit	:	Mengkonstruksi Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam mengkonstruksi data untuk proyek data science.

Langkah Kerja:

- 1) Menganalisis teknik transformasi data
- 2) Melakukan transformasi data
- 3) Membuat dokumentasi konstruksi data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengolah kata
 - Tools pengolah kata

1. ANALISIS TEKNIK TRANSFORMASI DATA

Instruksi Kerja:

- Lakukan analisis data untuk menentukan representasi fitur data awal
- Lakukan analisis representasi fitur data awal untuk menentukan teknik rekayasa fitur yang diperlukan untuk pembangunan model data science

Jawab:

A.) Analisis data untuk menentukan representasi fitur data awal melibatkan pemahaman mendalam tentang fitur-fitur yang ada dalam dataset. Ini dapat dilakukan dengan beberapa langkah:

Eksplorasi Data: Melakukan eksplorasi data untuk memahami distribusi, statistik deskriptif, dan sifat-sifat lainnya dari setiap fitur dalam dataset. Ini dapat melibatkan pembuatan histogram, box plot, atau visualisasi lainnya untuk melihat pola dan tren.

Untuk analisis awal, saya memakai statistical descriptive untuk melihat Gambaran awal dari data. Untuk data awal terdiri dari 918 baris dengan 12 kolom (termasuk 1 target). Lalu untuk tipe datanya ada yang masih dalam bentuk object, sehingga harus di adjust encoding (rekayasa fitur/feature engineering) kedalam bentuk numeric, agar nanti dapat diolah oleh model. Setelah itu saya mengecek apakah ada duplicate data & missing values. Setelah penanganan tersebut, saya mengecek distribusi data hist, menurut saya hist nya cukup bagus. Lalu saya mengecek ke boxplot,

ternyata terdapat outliers yang lumayan pada 3 kolom (Age, RestingBP, Cholesterol). Lalu saya melakukan adjustment outliers terhadap ketiga kolom tersebut.

Metode Evaluasi: Jelaskan metode yang digunakan untuk mengevaluasi proses pembersihan data, termasuk teknik analisis yang diterapkan, kriteria yang digunakan untuk menilai kebersihan data, dan alat atau platform yang digunakan dalam proses evaluasi.

Untuk awal, metode yang saya gunakan yaitu statistics descriptive untuk mengetahui data awal. Lalu evaluasi untuk cleaning data, saya menggunakan metode visual yakni histogram dan boxplot untuk mengetahui apakah sudah baik atau belum data nya dan apakah masih ada outliers atau tidak. Untuk kriteria yang digunakan untuk menilai kebersihan data yaitu semua kolom harus bertipe numeric, tidak ada outliers, tidak ada duplicate data & missing values. Lalu untuk platform yang digunakan tentu menggunakan google colab dengan Bahasa Python.

Hasil Evaluasi: Laporkan hasil dari evaluasi tersebut, termasuk temuan utama yang ditemukan selama evaluasi. Ini bisa mencakup perubahan yang terjadi dalam distribusi data, perbaikan spesifik yang dilakukan, dan peningkatan dalam kualitas data setelah pembersihan.

Berdasarkan hasil evalasi, didapat bahwa jumlah data setelah dilakukan cleansing & adjustment adalah seperti gambar dibawah ini:

```
1 # checking the distribution of Target Variable
2 heart_data['HeartDisease'].value_counts()

HeartDisease
0    378
1    340
Name: count, dtype: int64

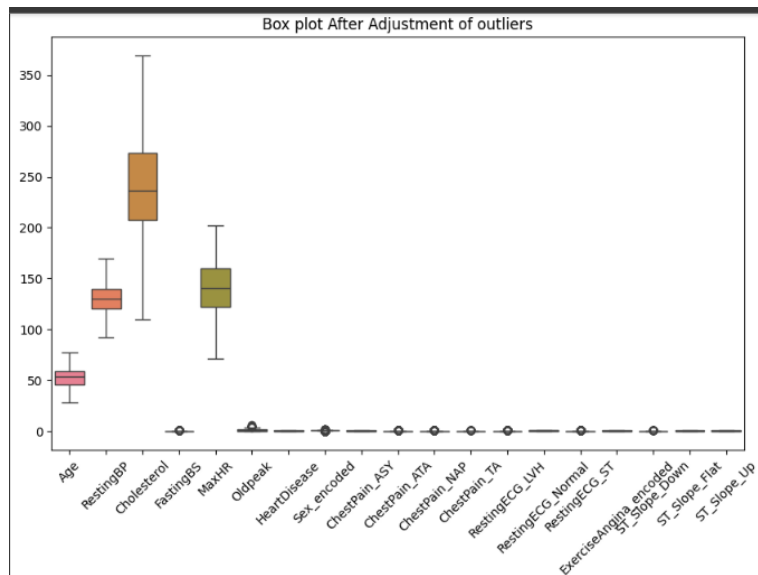
[55] 1 heart_data.shape

(718, 19)

[56] 1 heart_data.info()

<class 'pandas.core.frame.DataFrame'>
Index: 718 entries, 0 to 917
Data columns (total 19 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   Age                   718 non-null   float64
1   RestingBP             718 non-null   float64
2   Cholesterol            718 non-null   int64   
3   FastingBS             718 non-null   int64   
4   MaxHR                 718 non-null   int64   
5   Oldpeak               718 non-null   float64
6   HeartDisease          718 non-null   int64   
7   Sex_encoded           718 non-null   int64   
8   ChestPain_ASY         718 non-null   int64   
9   ChestPain_ATA         718 non-null   int64   
10  ChestPain_NAP         718 non-null   int64   
11  ChestPain_TA          718 non-null   int64   
12  RestingECG             718 non-null   int64
```

Jumlah orang yang positif heart disease sebanyak 340 dan yang negative ada 378. Lalu untuk jumlah baris ada 718 rows dengan 19 kolom (termasuk 1 target). Untuk tipe datanya, semua sudah dalam format numeric.



Berdasarkan gambar diatas, dapat dilihat bahwa sudah tidak ada outliers yang signifikan, yang dapat mengganggu kualitas data.

B.) Berikut hal untuk melakukan analisis representasi fitur data awal dan menentukan teknik rekayasa fitur yang diperlukan:

Eksplorasi Data:

- Lihat statistik deskriptif untuk setiap fitur.

```
[9] 1 heart_data.info()

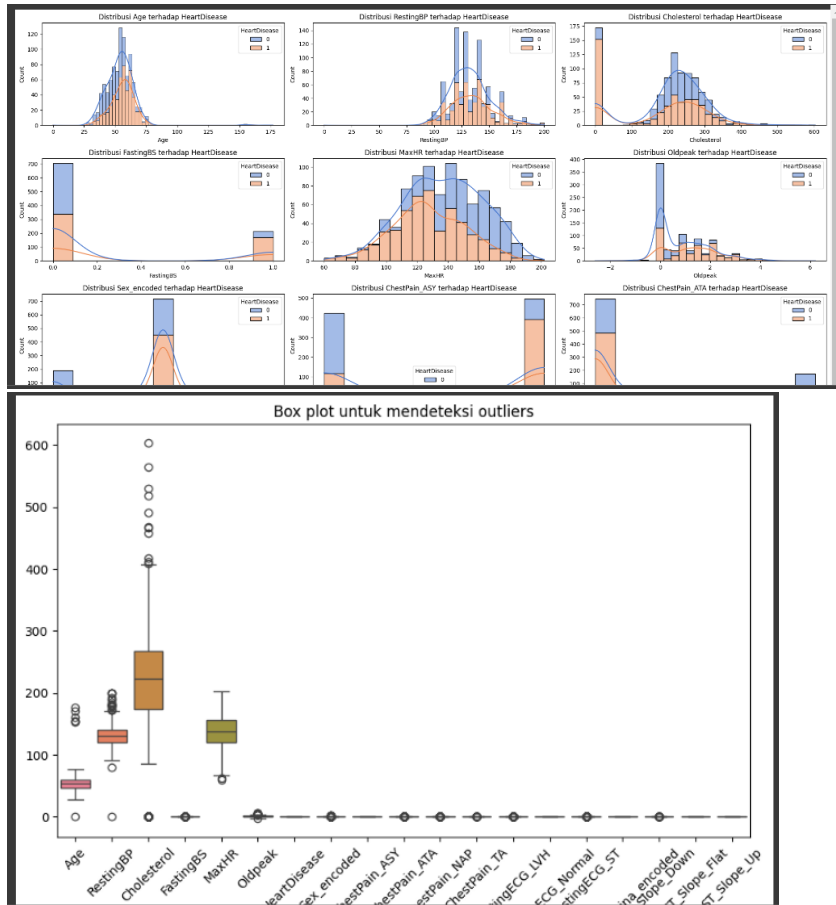
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
#   Column          Non-Null Count  Dtype  
---  --
0   Age              911 non-null    float64
1   Sex              908 non-null    object  
2   ChestPainType    918 non-null    object  
3   RestingBP        918 non-null    int64   
4   Cholesterol       918 non-null    int64   
5   FastingBS        918 non-null    int64   
6   RestingECG       918 non-null    object  
7   MaxHR            918 non-null    int64   
8   ExerciseAngina   918 non-null    object  
9   Oldpeak          918 non-null    float64
10  ST_Slope         918 non-null    object  
11  HeartDisease      918 non-null    int64   
dtypes: float64(2), int64(5), object(5)
memory usage: 86.2+ KB
```

```
[10] 1 heart_data.describe()

      Age  RestingBP  Cholesterol  FastingBS  MaxHR  Oldpeak  HeartDisease
count  911.000000  918.000000  918.000000  918.000000  918.000000  918.000000  918.000000
mean    54.102086  132.396514  198.799564  0.233115  136.809368  0.887364  0.553377
std     12.988393   18.514154  109.384145  0.423046  25.460334  1.066570  0.497414
min      0.000000   0.000000   0.000000  0.000000  60.000000  -2.600000  0.000000
25%     47.000000  120.000000  173.250000  0.000000  120.000000  0.000000  0.000000
50%     54.000000  130.000000  223.000000  0.000000  138.000000  0.600000  1.000000
75%     60.000000  140.000000  267.000000  0.000000  156.000000  1.500000  1.000000
max     177.000000  200.000000  603.000000  1.000000  202.000000  6.200000  1.000000
```

`describe()`, `info()`: Memberikan statistik deskriptif dan informasi umum tentang dataset. Seperti tipe data, mean, min, max

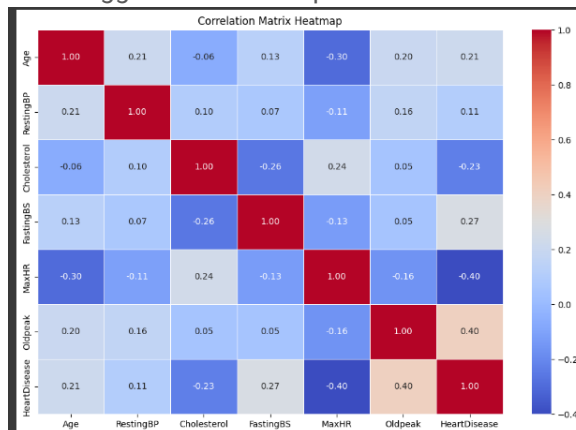
- Visualisasikan distribusi fitur menggunakan histogram atau box plot.



Dari gambar diatas, terdapat outliers yang signifikan (pada boxplot), untuk distribusi data (hist) sudah cukup baik.

Analisis Korelasi:

- Hitung korelasi antar fitur numerik untuk memahami hubungan linier antar fitur.
- Visualisasikan korelasi menggunakan heatmap.



`HeartDisease` berbanding terbalik dengan `MaxHR`. Jadi semakin tinggi MaxHR, semakin rendah pula kemungkinan orang terkena serangan jantung

Analisis Fitur Kategorikal:

- Lihat jumlah kategori unik untuk fitur kategorikal.

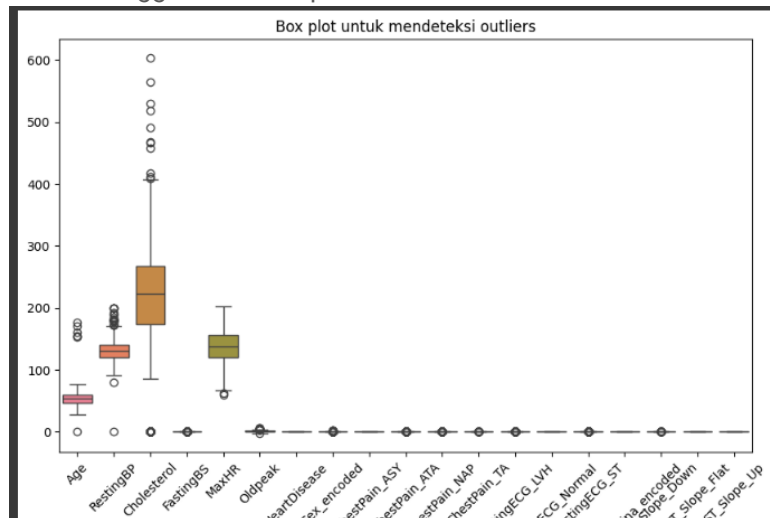
```
[9] 1 heart_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   Age                    911 non-null    float64
1   Sex                    908 non-null    object  
2   ChestPainType          918 non-null    object  
3   RestingBP              918 non-null    int64   
4   Cholesterol             918 non-null    int64   
5   FastingBS              918 non-null    int64   
6   RestingECG             918 non-null    object  
7   MaxHR                  918 non-null    int64   
8   ExerciseAngina         918 non-null    object  
9   Oldpeak                918 non-null    float64
10  ST_Slope               918 non-null    object  
11  HeartDisease           918 non-null    int64   
dtypes: float64(2), int64(5), object(5)
memory usage: 86.2+ KB
```

Terdapat 5 fitur kategorikal yang harus di adjust ke dalam bentuk numeric

Identifikasi dan Penanganan Outlier:

- Identifikasi outlier menggunakan box plot atau metode statistik.



Terdapat 3 kolom yang terdapat outliers yang signifikan dan harus di adjustment

- Putuskan apakah akan menghapus, mengganti, atau mentransformasi outlier.

Outliers:					
	Age	RestingBP	Cholesterol	FastingBS	MaxHR
74	155.0	140	268	0	12
329	0.0	130	0	1	13
389	161.0	160	0	1	14
447	177.0	124	171	0	11
626	153.0	142	226	0	11
687	170.0	156	245	0	14
707	154.0	124	266	0	10

Outliers pada kolom Age tidak masuk akal, sehingga harus saya drop.

outliers_restingbp:

	Age	RestingBP	Cho
109	39.0	190	
123	58.0	180	
189	53.0	180	
190	46.0	180	
241	54.0	200	
274	45.0	180	
275	59.0	180	
278	57.0	180	
314	53.0	80	
365	64.0	200	
372	63.0	185	

Untuk RestingBP masih bisa di transform ke dalam Mean, karena nilai dari outliers tidak terlalu ekstrem

```
[47] 1 outliers_Cholesterol.count()
Age 182
RestingBP 182
Cholesterol 182
```

Untuk outliers Cholesterol saya memilih untuk drop, karena data yang outliers lebih dari 10% dan takutnya jika saya meng-inject data dengan data sintetis seperti Mean/Median, takutnya nanti malahan akan terlalu merubah distribusi datanya.

Penanganan Missing Values:

- Identifikasi fitur dengan nilai hilang.

```
[30] 1 heart_data.isnull().sum()
Age 7
RestingBP 0
Cholesterol 0
FastingBS 0
MaxHR 0
Oldpeak 0
HeartDisease 0
Sex_encoded 0
ChestPain_ASY 0
ChestPain_ATA 0
ChestPain_NAP 0
ChestPain_TA 0
RestingECG_LVH 0
RestingECG_Normal 0
RestingECG_ST 0
ExerciseAngina_encoded 0
ST_Slope_Down 0
ST_Slope_Flat 0
ST_Slope_Up 0
dtype: int64
```

Data yang missing adalah kolom `Age` yaitu sebanyak 7 baris

- Putuskan metode untuk menangani nilai hilang (misalnya, imputasi dengan mean/median, atau penghapusan).

```
[31] 1 heart_data.fillna(heart_data.mean(), inplace=True)
2 heart_data.isnull().sum()
Age 0
RestingBP 0
Cholesterol 0
FastingBS 0
MaxHR 0
Oldpeak 0
HeartDisease 0
Sex_encoded 0
ChestPain_ASY 0
ChestPain_ATA 0
ChestPain_NAP 0
ChestPain_TA 0
RestingECG_LVH 0
RestingECG_Normal 0
RestingECG_ST 0
ExerciseAngina_encoded 0
ST_Slope_Down 0
ST_Slope_Flat 0
ST_Slope_Up 0
dtype: int64
```

Untuk penanganan Tindakan, saya men-Transform baris yang missing menjadi Mean

Feature Engineering:

- Tentukan teknik rekayasa fitur yang diperlukan seperti normalisasi, standarisasi, encoding, dll.

Untuk rekayasa fitur/feature engineering yang saya lakukan yaitu dengan Encoding untuk fitur-fitur kategorikal dan standarisasi/scaling agar menyamakan rentang data, agar kinerja model nantinya dapat lebih optimal.

Encoding Sex:

```
[15] 1 # Membuat objek LabelEncoder
      2 label_encoder = LabelEncoder()
      3
      4 # Melakukan encoding pada kolom 'Sex'
      5 heart_data['Sex_encoded'] = label_encoder.fit_transform(heart_data['Sex'])
      6
      7 # Menampilkan hasil encoding
      8 heart_data[['Sex', 'Sex_encoded']]
```

	Sex	Sex_encoded
0	M	1
1	F	0
2	M	1
3	F	0
4	M	1
...
913	M	1

Saya menggunakan LabelEncoder pada kolom Sex

Encoding ChestPainType:

```
[18] 1 # Melakukan One-Hot Encoding pada kolom 'ChestPainType' dengan mengonversi nilai menjadi integer
      2 chest_pain_encoded = pd.get_dummies(heart_data['ChestPainType'], prefix='ChestPain', dtype=int)
      3
      4 # Menggabungkan hasil encoding dengan DataFrame asli
      5 heart_data = pd.concat([heart_data, chest_pain_encoded], axis=1)
      6
      7 # Menghapus kolom 'ChestPainType' asli
      8 heart_data.drop('ChestPainType', axis=1, inplace=True)
      9
      10 # Menampilkan hasil encoding
      11 heart_data
```

Age	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA
4a.N	140	289	0	Normal	172	N	0.0	Up	0	1	0	1	0	0
19.0	160	180	0	Normal	156	N	1.0	Flat	1	0	0	0	1	0
17.0	130	283	0	ST	98	N	0.0	Up	0	1	0	1	0	0
18.0	138	214	0	Normal	108	Y	1.5	Flat	1	0	1	0	0	0

Saya melakukan One-Hot-Encoding pada kolom ChestPainType

Encoding RestingECG:

```
[19] 1 # Melakukan One-Hot Encoding pada kolom 'RestingECG' dengan penjelasan isi
      2 resting_ecg_encoded = pd.get_dummies(heart_data['RestingECG'], prefix='RestingECG', dtype=int)
      3
      4 # Menggabungkan hasil encoding dengan DataFrame asli
      5 heart_data = pd.concat([heart_data, resting_ecg_encoded], axis=1)
      6
      7 # Menghapus kolom 'RestingECG' asli
      8 heart_data.drop('RestingECG', axis=1, inplace=True)
      9
      10 # Menampilkan hasil encoding
      11 heart_data
```

MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST
172	N	0.0	Up	0	1	0	1	0	0	0	1	0
156	N	1.0	Flat	1	0	0	0	1	0	0	1	0
98	N	0.0	Up	0	1	0	1	0	0	0	0	1
108	Y	1.5	Flat	1	0	1	0	0	0	0	1	0

Saya melakukan One-Hot-Encoding pada kolom RestingECG

Encoding ExerciseAngina:

```
[21] 1 # Mengganti nilai 'N' dengan 0 dan 'Y' dengan 1
      2 heart_data['ExerciseAngina_encoded'] = heart_data['ExerciseAngina'].replace({'N': 0, 'Y': 1}).astype(int)
      3
      4 # Menghapus kolom 'ExerciseAngina' asli
      5 heart_data.drop('ExerciseAngina', axis=1, inplace=True)
      6
      7 # Menampilkan hasil encoding
      8 heart_data
```

Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded
0.0	Up	0	1	0	1	0	0	0	1	0	0
1.0	Flat	1	0	0	0	1	0	0	1	0	0
0.0	Up	0	1	0	1	0	0	0	0	1	0
1.5	Flat	1	0	1	0	0	0	0	1	0	1
0.0	Up	0	1	0	0	1	0	0	1	0	0

Saya melakukan BinaryEncoding pada kolom ExerciseAngina

Encoding ST_Slope:

```
[24] 1 # Melakukan One-Hot Encoding pada kolom 'ST_Slope' dengan penjelasan isi dan mengonversi nilai menjadi integer
      2 st_slope_encoded = pd.get_dummies(heart_data['ST_Slope'], prefix='ST_Slope', dtype=int)
      3
      4 # Menggabungkan hasil encoding dengan DataFrame asli
      5 heart_data = pd.concat([heart_data, st_slope_encoded], axis=1)
      6
      7 # Menghapus kolom 'ST_Slope' asli
      8 heart_data.drop('ST_Slope', axis=1, inplace=True)
      9
      10 # Menampilkan hasil encoding
      11 heart_data
```

ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded	ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
1	0	1	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1
1	0	1	0	0	0	0	1	0	0	1
0	1	0	0	0	0	1	0	1	0	1
1	0	0	1	0	0	1	0	0	0	1

Saya melakukan One-Hot-Encoding pada kolom ST_Slope

Standarisasi/Scaling

```
4.2 Scaling

[ ] 1 # Buat pipeline dengan preprocessing
      2 pipeline = Pipeline([
      3     ('quantile_transform', QuantileTransformer(output_distribution='normal')),
      4     ('scaler', StandardScaler()) # Scaling fitur
      5 ])
```

Lalu saya melakukan Pipeline yang isinya ada QuantileTransform agar distribusi data normal dan saya melakukan Scaling dengan StandardScaler.

2. TRANSFORMASI DATA

Instruksi Kerja:

- Lakukan transformasi untuk mendapatkan fitur data awal
- Lakukan rekayasa fitur data untuk mendapatkan fitur baru yang diperlukan untuk pembangunan model data science

Jawab:

A.) Untuk mendapatkan fitur data awal yang siap digunakan dalam model data science, kita perlu melakukan transformasi tertentu seperti encoding fitur kategorikal, normalisasi atau standarisasi

fitur numerik, dan penanganan nilai hilang atau outlier. Berikut adalah langkah-langkah transformasi yang diterapkan pada dataset heart_data:

Encoding Fitur Kategorikal:

Encoding pada 5 kolom kategorikal yaitu Sex, ChestPainType, RestingECG, ExerciseAngina, ST_Slope. Untuk kolom Sex, saya memakai LabelEncoder

```
[15] 1 # Membuat objek LabelEncoder
      2 label_encoder = LabelEncoder()
      3
      4 # Melakukan encoding pada kolom 'Sex'
      5 heart_data['Sex_encoded'] = label_encoder.fit_transform(heart_data['Sex'])
      6
      7 # Menampilkan hasil encoding
      8 heart_data[['Sex', 'Sex_encoded']]
```

	Sex	Sex_encoded
0	M	1
1	F	0
2	M	1
3	F	0
4	M	1
...
913	M	1

Lalu untuk ExerciseAngina saya menggunakan BinaryEncoding:

```
[21] 1 # Mengganti nilai 'N' dengan 0 dan 'Y' dengan 1
      2 heart_data['ExerciseAngina_encoded'] = heart_data['ExerciseAngina'].replace({'N': 0, 'Y': 1}).astype(int)
      3
      4 # Menghapus kolom 'ExerciseAngina' asli
      5 heart_data.drop('ExerciseAngina', axis=1, inplace=True)
      6
      7 # Menampilkan hasil encoding
      8 heart_data
```

	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded
0.0	Up	0	1	0	1	0	0	0	1	0	0	
1.0	Flat	1	0	0	0	1	0	0	1	0	0	
0.0	Up	0	1	0	1	0	0	0	0	1	0	
1.5	Flat	1	0	1	0	0	0	0	1	0	1	
0.0	Up	0	1	0	0	1	0	0	1	0	0	

Lalu untuk ketiga kolom lainnya yaitu ChestPainType, RestingECG, dan St_Slope, saya menggunakan One-Hot-Encoding:

```
[18] 1 # Melakukan One-Hot Encoding pada kolom 'ChestPainType' dengan mengonversi nilai menjadi integer
      2 chest_pain_encoded = pd.get_dummies(heart_data['ChestPainType'], prefix='ChestPain', dtype=int)
      3
      4 # Menggabungkan hasil encoding dengan DataFrame asli
      5 heart_data = pd.concat([heart_data, chest_pain_encoded], axis=1)
      6
      7 # Menghapus kolom 'ChestPainType' asli
      8 heart_data.drop('ChestPainType', axis=1, inplace=True)
      9
      10 # Menampilkan hasil encoding
      11 heart_data
```

	Age	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA
4a.N	140	289	0	Normal	172	N	0.0	Up	0	1	0	1	0	0	
19.0	160	180	0	Normal	156	N	1.0	Flat	1	0	0	0	1	0	
17.0	130	283	0	ST	98	N	0.0	Up	0	1	0	1	0	0	
18.0	138	214	0	Normal	108	Y	1.5	Flat	1	0	1	0	0	0	

```
[19] 1 # Melakukan One-Hot Encoding pada kolom 'RestingECG' dengan penjelasan isi
2 resting_ecg_encoded = pd.get_dummies(heart_data['RestingECG'], prefix='RestingECG', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, resting_ecg_encoded], axis=1)
6
7 # Menghapus kolom 'RestingECG' asli
8 heart_data.drop('RestingECG', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data
```

MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST
172	N	0.0	Up	0	1	0	1	0	0	0	1	0
156	N	1.0	Flat	1	0	0	0	1	0	0	1	0
98	N	0.0	Up	0	1	0	1	0	0	0	0	1
108	Y	1.5	Flat	1	0	1	0	0	0	0	1	0

```
[24] 1 # Melakukan One-Hot Encoding pada kolom 'ST_Slope' dengan penjelasan isi dan mengonversi nilai menjadi integer
2 st_slope_encoded = pd.get_dummies(heart_data['ST_Slope'], prefix='ST_Slope', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, st_slope_encoded], axis=1)
6
7 # Menghapus kolom 'ST_Slope' asli
8 heart_data.drop('ST_Slope', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data
```

id	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded	ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
1	0	1	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
1	0	1	0	0	0	0	1	0	0	0	1
0	1	0	0	0	0	1	0	1	0	1	0
1	0	0	1	0	0	1	0	0	0	0	1

Normalisasi atau Standarisasi Fitur Numerik

Selanjutnya saya menggunakan Scaling dengan StandardScaler untuk standarisasi fiturnya:

```
4.2 Scaling

[ ] 1 # Buat pipeline dengan preprocessing
2 pipeline = Pipeline([
3     ('quantile_transform', QuantileTransformer(output_distribution='normal')),
4     ('scaler', StandardScaler()) # Scaling fitur
5 ])
```

Penanganan Missing Values

Untuk handle missing value pada kolom Age, saya men-Transform baris yang missing pada kolom tersebut kedalam mean dari kolom tersebut (Age)

```
[33] 1 # Mengisi nilai-nilai yang hilang dengan mean dari setiap kolom dengan inplace=True
2 heart_data.fillna(heart_data.mean(), inplace=True)

1 heart_data.isnull().sum()

Age 0
RestingBP 0
Cholesterol 0
FastingBS 0
MaxHR 0
Oldpeak 0
HeartDisease 0
Sex_encoded 0
ChestPain_ASY 0
ChestPain_ATA 0
ChestPain_NAP 0
ChestPain_TA 0
RestingECG_LVH 0
RestingECG_Normal 0
RestingECG_ST 0
ExerciseAngina_encoded 0
ST_Slope_Down 0
ST_Slope_Flat 0
ST_Slope_Up 0
dtype: int64
```

Penanganan Outliers:

Saya menggunakan metode IQR (InterQuartileRange) untuk mengetahui outliers agar saya dapat melakukan adjustment. Berikut adalah gambar dari IQR pada ketiga kolom (Age, RestingBP, Cholesterol) yang terindikasi outliers, yaitu

```
[40] 1 # Hitung Q1 dan Q3
      2 Q1 = heart_data['Age'].quantile(0.25)
      3 Q3 = heart_data['Age'].quantile(0.75)
      4
      5 # Hitung IQR
      6 IQR = Q3 - Q1
      7
      8 # Tentukan batas atas dan batas bawah
      9 upper_bound = Q3 + 1.5 * IQR
     10 lower_bound = Q1 - 1.5 * IQR
     11
     12 # Identifikasi outliers
     13 outliers = heart_data[(heart_data['Age'] < lower_bound) | (heart_data['Age'] > upper_bound)]
     14
     15 # Tampilkan outliers
     16 print("Outliers:")
     17 outliers
```

```
[43] 1 # Hitung Q1 dan Q3
      2 Q1 = heart_data['RestingBP'].quantile(0.25)
      3 Q3 = heart_data['RestingBP'].quantile(0.75)
      4
      5 # Hitung IQR
      6 IQR = Q3 - Q1
      7
      8 # Tentukan batas atas dan batas bawah
      9 upper_bound = Q3 + 1.5 * IQR
     10 lower_bound = Q1 - 1.5 * IQR
     11
     12 # Identifikasi outliers_restingbp
     13 outliers_restingbp = heart_data[(heart_data['RestingBP'] < lower_bound) | (heart_data['RestingBP'] > upper_bound)]
     14
     15 # Tampilkan outliers_restingbp
     16 print("outliers_restingbp:")
     17 outliers_restingbp
```

```
[46] 1 # Hitung Q1 dan Q3
      2 Q1 = heart_data['Cholesterol'].quantile(0.25)
      3 Q3 = heart_data['Cholesterol'].quantile(0.75)
      4
      5 # Hitung IQR
      6 IQR = Q3 - Q1
      7
      8 # Tentukan batas atas dan batas bawah
      9 upper_bound = Q3 + 1.5 * IQR
     10 lower_bound = Q1 - 1.5 * IQR
     11
     12 # Identifikasi outliers_Cholesterol
     13 outliers_Cholesterol = heart_data[(heart_data['Cholesterol'] < lower_bound) | (heart_data['Cholesterol'] > upper_bound)]
     14
     15 # Tampilkan outliers_Cholesterol
     16 print("outliers_Cholesterol:")
     17 outliers_Cholesterol
```

B.) Rekayasa fitur (feature engineering) adalah proses membuat fitur tambahan dari data yang sudah ada untuk meningkatkan performa model. Berikut adalah beberapa contoh rekayasa fitur yang dapat diterapkan pada dataset `heart_data` untuk membangun model klasifikasi penyakit jantung:

Encoding pada Fitur Numeric

Melakukan encoding pada fitur numeric sehingga terdapat kolom baru, yang akan berguna bagi signifikansi daripada modelling.

- **Encoding pada Sex**

	Sex	Sex_encoded
0	M	1
1	F	0
2	M	1
3	F	0
4	M	1

Encoding pada kolom Sex akan membuat kolom yang lebih bermakna, dan dapat digunakan sebagai salah satu fitur untuk modelling

- **Encoding pada ChestPainType**

ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA
0	1	0	0
0	0	1	0
0	1	0	0
1	0	0	0
0	0	1	0

Encoding pada kolom tersebut, akan menghasilkan 4 kolom yang lebih bermakna dan dapat digunakan sebagai salah satu fitur untuk modelling

- **Encoding pada RestingECG**

RestingECG_LVH	RestingECG_Normal	RestingECG_ST
0	1	0
0	1	0
0	0	1
0	1	0
0	1	0

Encoding pada kolom tersebut akan menghasilkan 3 kolom baru yang lebih bermakna dan dapat digunakan untuk fitur dalam modelling

- **Encoding pada ExerciseAngina**

ExerciseAngina_encoded
0
0
0
1
0

Encoding akan menghasilkan kolom baru yang dapat bermanfaat untuk modelling

- Encoding pada ST_Slope

ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
0	0	1
0	1	0
0	0	1
0	1	0
0	0	1
...

Encoding pada kolom tersebut, akan membuat 3 kolom baru yang dapat berkontribusi dalam modelling.

Normalisasi dan Standarisasi:

```

v 4.2 Scaling

[ ] 1 # Buat pipeline dengan preprocessing
    2 pipeline = Pipeline([
    3     ('quantile_transform', QuantileTransformer(output_distribution='normal')),
    4     ('scaler', StandardScaler()) # Scaling fitur
    5 ])

```

Saya menggunakan QuantileTransform untuk meng-adjust distribusi data agar menjadi distribusi normal. Lalu saya melakukan Scaling untuk meratakan rentang nilai fitur dengan menggunakan scaling method yaitu StandardScaler.

3. DOKUMENTASI KONSTRUKSI DATA

Instruksi Kerja:

- Jabarkan teknis transformasi data dalam bentuk tertulis
- Tuangkan hasil transformasi data dan rekomendasi hasil transformasi dalam bentuk tertulis

Catatan:

- Langkah kerja ini dapat diintegrasikan dengan langkah-langkah kerja sebelumnya
- Bila pada langkah kerja (1) mengalisis teknik transformasi data; dan (2) melakukan transformasi data; telah didokumentasikan dalam bentuk laporan yang memadai, maka langkah kerja (3) membuat dokumentasi konstruksi data; dapat diabaikan.

BUKTI 7-ADS

Kode Unit	:	J.62DMI00.010.1
Judul Unit	:	Menentukan Label Data

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam menentukan label data untuk pembangunan model data science.

Langkah Kerja:

- 1) Melakukan pelabelan data
- 2) Membuat laporan hasil pelabelan data

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Aplikasi pengolah kata
 - Aplikasi pelabelan data

1. PELABELAN DATA

Instruksi Kerja:

- Uraikan kesesuaian antara analisis hasil pelabelan data sejenis yang sudah ada dengan Standard Operating Procedure (SOP) pelabelan
- Lakukan pelabelan data sesuai dengan SOP pelabelan

Jawab:

A.) Dalam konteks data science, analisis hasil pelabelan data sejenis dan kesesuaiannya dengan Standard Operating Procedure (SOP) pelabelan mencakup beberapa aspek penting. Berikut adalah uraian mengenai kesesuaian antara analisis hasil pelabelan data sejenis yang sudah ada dengan SOP pelabelan berdasarkan apa yang telah kita kerjakan sebelumnya:

Definisi Label dan Konsistensi Pelabelan

- **SOP Pelabelan:** SOP pelabelan harus secara jelas mendefinisikan setiap label yang akan digunakan dalam dataset. Misalnya, dalam kasus ini, label yang digunakan adalah `HeartDisease` dengan nilai 1 untuk menunjukkan adanya penyakit jantung dan 0 untuk menunjukkan tidak adanya penyakit jantung.
- **Analisis Hasil Pelabelan:** Data yang saya kerjakan menunjukkan bahwa label `HeartDisease` telah diterapkan dengan konsisten, mengikuti definisi yang ada. Tidak ditemukan inkonsistensi dalam pelabelan, yang menunjukkan kepatuhan terhadap SOP.

Metode Pelabelan

- **SOP Pelabelan:** SOP harus mendokumentasikan metode yang digunakan untuk menentukan label. Ini bisa mencakup aturan atau algoritma tertentu yang digunakan untuk mengkategorikan data.
- **Analisis Hasil Pelabelan:** Label `HeartDisease` dalam dataset `heart_data` tampaknya diberikan berdasarkan pemeriksaan medis dan diagnosis yang sah. Ini sesuai dengan praktik standar medis dan pelabelan diagnostik yang ketat.

Kualitas dan Integritas Data

- **SOP Pelabelan:** Menyediakan panduan untuk memastikan bahwa data yang dilabeli adalah akurat dan bebas dari kesalahan. Ini mungkin melibatkan langkah-langkah seperti verifikasi ganda dan pemeriksaan ahli.
- **Analisis Hasil Pelabelan:** Berdasarkan analisis awal, data yang saya gunakan telah melalui proses validasi. Contohnya, saya telah menghapus data dengan nilai yang tidak masuk akal (outliers) dan telah mengimputasi nilai yang hilang. Ini menunjukkan bahwa data telah disiapkan dengan hati-hati sesuai dengan SOP untuk menjaga kualitas dan integritas.

Preprocessing dan Transformasi Data

- **SOP Pelabelan:** Harus mencakup pedoman untuk preprocessing dan transformasi data sebelum pelabelan, termasuk penanganan nilai yang hilang, normalisasi, dan encoding.
- **Analisis Hasil Pelabelan:** Saya telah melakukan preprocessing dengan menstandarisasi fitur numerik, menerapkan one-hot encoding, `LabelEncoding`, dan `BinaryEncoding` untuk fitur kategorikal, dan melakukan transformasi outliers. Ini sesuai dengan praktik standar preprocessing yang mungkin disebutkan dalam SOP.

Dokumentasi dan Reprodusibilitas

- **SOP Pelabelan:** Menekankan pentingnya dokumentasi setiap langkah proses pelabelan untuk memastikan bahwa proses tersebut dapat direproduksi di masa depan.
- **Analisis Hasil Pelabelan:** Setiap langkah dalam analisis dan preprocessing telah didokumentasikan dengan jelas, termasuk kode yang digunakan untuk transformasi dan rekayasa fitur. Ini memastikan bahwa proses dapat diulang dan diverifikasi oleh pihak lain.

B.) Untuk melabeli data sesuai dengan Standard Operating Procedure (SOP) pelabelan, langkah-langkah berikut ini mencakup keseluruhan proses dari pemilihan data, pembersihan data, hingga pelabelan akhir. Proses ini akan disesuaikan dengan standar dan prosedur yang telah ditetapkan untuk memastikan bahwa data dilabeli dengan benar dan konsisten.

Langkah-langkah Pelabelan Data Sesuai SOP

- Identifikasi Data yang Akan Dilabeli
- Pembersihan Data
- Transformasi dan Rekayasa Fitur
- Pelabelan Data
- Verifikasi dan Validasi Label

Berikut adalah implementasi kode untuk masing-masing langkah:

- Identifikasi Data yang Akan Dilabeli

```
1 heart_data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Ujian Sertifikasi BNSP ADS/heart.csv')
[64] 1 heart_data.head()
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	NaN	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49.0	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37.0	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48.0	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54.0	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

Untuk data yang akan menjadi label/target yaitu kolom `HeartDisease`

- Pembersihan Data

```
[90] 1 # Mengisi nilai-nilai yang hilang dengan mean dari setiap kolom dengan inplace=True
      2 heart_data.fillna(heart_data.mean(), inplace=True)

[91] 1 heart_data.isnull().sum()
```

Age	0
RestingBP	0
Cholesterol	0
FastingBS	0
MaxHR	0
Oldpeak	0
HeartDisease	0
Sex_encoded	0
ChestPain_ASY	0
ChestPain_ATA	0
ChestPain_NAP	0
ChestPain_TA	0
RestingECG_LVH	0
RestingECG_Normal	0
RestingECG_ST	0
ExerciseAngina_encoded	0
ST_Slope_Down	0
ST_Slope_Flat	0
ST_Slope_Up	0
	dtype: int64

```
[115] 1 # Menghapus nilai yang tidak masuk akal secara in-place
      2 heart_data.drop(heart_data[(heart_data['Age'] <= 0) |
      3                  (heart_data['RestingBP'] <= 0) |
      4                  (heart_data['Cholesterol'] <= 0)].index, inplace=True)
```

Pada gambar diatas saya melakukan penghapusan baris yang missing pada kolom yang memiliki nilai missing tersebut.

- **Transformasi dan Rekayasa Fitur**

Pada tahap ini, saya melakukan feature engineering berupa encoding terhadap kolom-kolom yang masih berjenis kategorikal. Tujuan saya melakukan hal tersebut adalah untuk merubah kolom tersebut menjadi bentuk numeric agar dapat diolah pada saat modelling. Berikut adalah gambar-gambar terkait proses encoding:

2.2 Encoding of Sex

```

[73] 1 # Membuat objek LabelEncoder
2 label_encoder = LabelEncoder()
3
4 # Melakukan encoding pada kolom 'Sex'
5 heart_data['sex_encoded'] = label_encoder.fit_transform(heart_data['Sex'])
6
7 # Menampilkan hasil encoding
8 heart_data[['Sex', 'sex_encoded']]

```

	Sex	sex_encoded
0	M	1
1	F	0
2	M	1
3	F	0
4	M	1

2.3 Encoding of ChestPainType

```

[76] 1 # Melakukan one-hot encoding pada kolom 'ChestPainType' dengan mengonversi nilai menjadi integer
2 chest_pain_encoded = pd.get_dummies(heart_data['ChestPainType'], prefix='ChestPain', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, chest_pain_encoded], axis=1)
6
7 # Menghapus kolom 'ChestPainType' asli
8 heart_data.drop('ChestPainType', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data

```

	Age	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA
0	140	289	0	Normal	172	N	0.0	Up	0	1	0	1	0	0	0
1	160	180	0	Normal	156	N	1.0	Flat	1	0	0	0	1	0	0
2	130	283	0	ST	98	N	0.0	Up	0	1	0	1	0	0	0
3	138	214	0	Normal	108	Y	1.5	Flat	1	0	1	0	0	0	0
4	150	195	0	Normal	122	N	0.0	Up	0	1	0	0	0	1	0

2.4 Encoding of RestingECG

```

[77] 1 # Melakukan one-hot encoding pada kolom 'RestingECG' dengan penjelasan isi
2 resting_ecg_encoded = pd.get_dummies(heart_data['RestingECG'], prefix='RestingECG', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, resting_ecg_encoded], axis=1)
6
7 # Menghapus kolom 'RestingECG' asli
8 heart_data.drop('RestingECG', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data

```

	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST
0	172	N	0.0	Up	0	1	0	1	0	0	0	1	0
1	156	N	1.0	Flat	1	0	0	0	1	0	0	0	1
2	98	N	0.0	Up	0	1	0	1	0	0	0	0	1
3	108	Y	1.5	Flat	1	0	1	0	0	0	0	1	0
4	122	N	0.0	Up	0	1	0	0	1	0	0	0	1

```

[79] 1 # Mengganti nilai 'N' dengan 0 dan 'Y' dengan 1
2 heart_data['ExerciseAngina_encoded'] = heart_data['ExerciseAngina'].replace({'N': 0, 'Y': 1}).astype(int)
3
4 # Menghapus kolom 'ExerciseAngina' asli
5 heart_data.drop('ExerciseAngina', axis=1, inplace=True)
6
7 # Menampilkan hasil encoding
8 heart_data

```

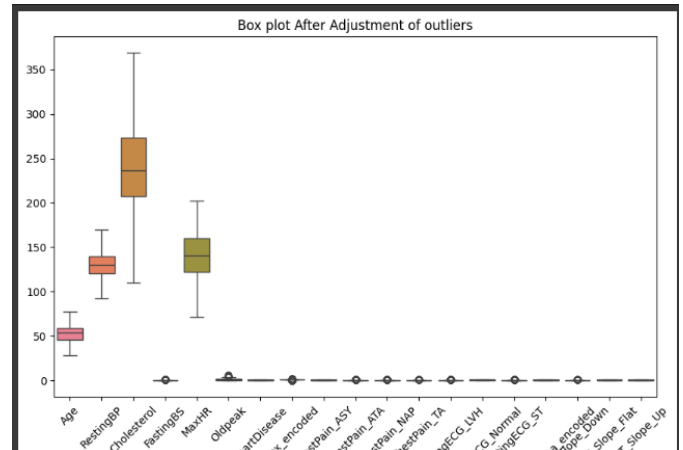
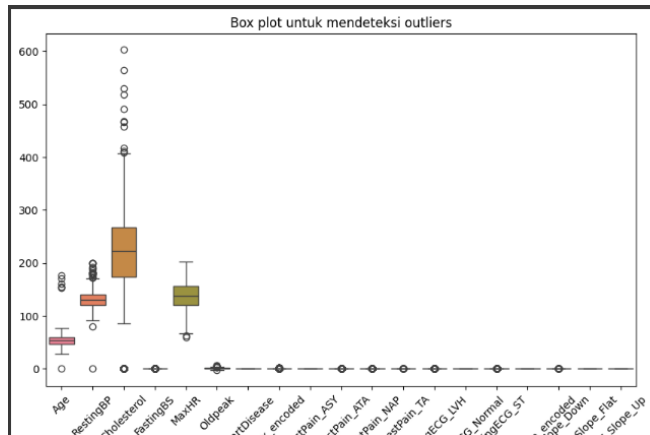
	Oldpeak	ST_Slope	HeartDisease	Sex_encoded	ChestPain_ASY	ChestPain_ATA	ChestPain_NAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded
0	0.0	Up	0	1	0	1	0	0	0	1	0	0
1	1.0	Flat	1	0	0	0	1	0	0	1	0	0
2	0.0	Up	0	1	0	1	0	0	0	0	1	0
3	1.5	Flat	1	0	1	0	0	0	0	1	0	1
4	0.0	Up	0	1	0	0	1	0	0	1	0	0

```

1 # Melakukan One-Hot Encoding pada kolom 'ST_Slope' dengan penjelasan isi dan mengonversi nilai menjadi integer
2 st_slope_encoded = pd.get_dummies(heart_data['ST_Slope'], prefix='ST_Slope', dtype=int)
3
4 # Menggabungkan hasil encoding dengan DataFrame asli
5 heart_data = pd.concat([heart_data, st_slope_encoded], axis=1)
6
7 # Menghapus kolom 'ST_Slope' asli
8 heart_data.drop('ST_Slope', axis=1, inplace=True)
9
10 # Menampilkan hasil encoding
11 heart_data

```

id	ChestPain_ASY	ChestPain_ATA	ChestPain_MAP	ChestPain_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_encoded	ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
1	0	1	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
1	0	1	0	0	0	0	1	0	0	0	1
0	1	0	0	0	0	1	0	1	0	1	0
1	0	0	1	0	0	1	0	0	0	0	1



- Pelabelan Data
Pelabelan dalam konteks ini berarti mengonfirmasi bahwa label `HeartDisease` sudah ada dan benar sesuai SOP.

```

[111] 1 # checking the distribution of Target Variable
2 heart_data['HeartDisease'].value_counts()

```

```

HeartDisease
0    378
1    340
Name: count, dtype: int64

```

- Verifikasi dan Validasi Label

```

[118] 1 # Verifikasi label untuk memastikan tidak ada kesalahan
2 print(heart_data.groupby('HeartDisease').size())

```

```

HeartDisease
0    378
1    340
dtype: int64

```

Pada gambar diatas, saya sedang meninjau terkait pembagian isi data dari kolom target dan hasilnya adalah sebagai berikut: jumlah orang yang normal sekitar 378 orang, dan yang terkena penyakit jantung ada 340 orang.

2. LAPORAN HASIL PELABELAN DATA

Instruksi Kerja:

- Uraikan statistik hasil pelabelan pada laporan
- Uraikan evaluasi proses pelabelan pada laporan

Jawab:

A.) Berikut adalah uraian statistik hasil pelabelan data `HeartDisease` pada dataset `heart_data` setelah melalui proses pembersihan, transformasi, dan rekayasa fitur sesuai dengan Standard Operating Procedure (SOP):

Jumlah Total Data

- Jumlah total data setelah pembersihan adalah 718 baris.

Distribusi Label `HeartDisease`

- Data pelabelan memiliki dua kelas:
 - 0: Tidak menderita penyakit jantung
 - 1: Menderita penyakit jantung

Statistik Label `HeartDisease`

- Jumlah data untuk kelas 0: 378
- Jumlah data untuk kelas 1: 340
- Persentase data untuk kelas 0: 52.65%
- Persentase data untuk kelas 1: 47.35%

Berikut adalah implementasi kode untuk menghitung dan menampilkan statistik hasil pelabelan:

```
[119] 1 # Menghitung statistik label
      2 label_counts = heart_data['HeartDisease'].value_counts()
      3 label_percentages = heart_data['HeartDisease'].value_counts(normalize=True) * 100
      4
      5 # Menampilkan statistik
      6 print(f"Jumlah total data: {len(heart_data)}")
      7 print(f"Jumlah data untuk kelas '0' (Tidak menderita penyakit jantung): {label_counts[0]}")
      8 print(f"Jumlah data untuk kelas '1' (Menderita penyakit jantung): {label_counts[1]}")
      9 print(f"Persentase data untuk kelas '0': {label_percentages[0]:.2f}%")
     10 print(f"Persentase data untuk kelas '1': {label_percentages[1]:.2f}%")
     11
     12 # Dokumentasi statistik hasil pelabelan
     13 with open('labeling_statistics.txt', 'w') as f:
     14     f.write(f"Jumlah total data: {len(heart_data)}\n")
     15     f.write(f"Jumlah data untuk kelas '0' (Tidak menderita penyakit jantung): {label_counts[0]}\n")
     16     f.write(f"Jumlah data untuk kelas '1' (Menderita penyakit jantung): {label_counts[1]}\n")
     17     f.write(f"Persentase data untuk kelas '0': {label_percentages[0]:.2f}%\n")
     18     f.write(f"Persentase data untuk kelas '1': {label_percentages[1]:.2f}%\n")

Jumlah total data: 718
Jumlah data untuk kelas '0' (Tidak menderita penyakit jantung): 378
Jumlah data untuk kelas '1' (Menderita penyakit jantung): 340
Persentase data untuk kelas '0': 52.65%
Persentase data untuk kelas '1': 47.35%
```

Laporan Statistik Hasil Pelabelan

Jumlah Total Data:

- Jumlah total data setelah pembersihan: **718**

Distribusi Label `HeartDisease`:

- Jumlah data untuk kelas 0 (Tidak menderita penyakit jantung): **378**
- Jumlah data untuk kelas 1 (Menderita penyakit jantung): **340**

Persentase Distribusi Label:

- Persentase data untuk kelas 0: **52.65%**
- Persentase data untuk kelas 1: **47.35%**

Analisis:

- Data setelah pembersihan memiliki distribusi yang cukup seimbang antara kelas 0 dan kelas 1. Hal ini penting untuk memastikan model pembelajaran mesin yang dibangun tidak bias terhadap salah satu kelas.
- Data yang telah diproses siap untuk digunakan dalam pembangunan model pembelajaran mesin (ML) dengan algoritma seperti Decision Tree dan XGBoost.

Dokumentasi ini memberikan gambaran yang jelas mengenai hasil pelabelan dan statistik distribusi label `HeartDisease` pada dataset, yang memastikan bahwa data siap digunakan untuk tujuan teknis data science sesuai dengan SOP pelabelan yang berlaku.

B.) EVALUASI PROSES PELABELAN PADA LAPORAN

Pendahuluan

Proses pelabelan data merupakan langkah krusial dalam proyek data science. Pada proyek ini, saya melakukan pelabelan untuk menentukan apakah pasien menderita penyakit jantung (`HeartDisease`) berdasarkan berbagai fitur yang ada dalam dataset `heart_data`. Evaluasi ini bertujuan untuk memastikan bahwa proses pelabelan sesuai dengan Standard Operating Procedure (SOP) dan hasilnya dapat diandalkan untuk pembangunan model prediktif.

Kriteria Pelabelan:

- Fitur yang digunakan untuk pelabelan:
 - Age: Usia pasien
 - Sex: Jenis kelamin pasien
 - ChestPainType: Jenis nyeri dada
 - RestingBP: Tekanan darah istirahat
 - Cholesterol: Kolesterol serum
 - FastingBS: Gula darah puasa
 - RestingECG: Hasil elektrokardiogram saat istirahat

- MaxHR: Detak jantung maksimum
- ExerciseAngina: Angina yang diinduksi oleh Latihan
- Oldpeak: Depresi ST yang dihasilkan oleh latihan relatif terhadap istirahat
- ST_Slope: Kemiringan segmen ST
- Label target:
 - HeartDisease: 1 jika pasien menderita penyakit jantung, 0 jika tidak.

Proses Pelabelan:

- Data Preprocessing:
 - Menghapus nilai yang tidak masuk akal (Age, RestingBP, dan Cholesterol).
 - Melakukan encoding untuk fitur kategorikal (Sex, ChestPainType, RestingECG, ExerciseAngina, dan ST_Slope).
 - Melakukan scaling pada fitur numerik untuk memastikan data berada dalam skala yang sama (StandardScaler).
- Teknik Pelabelan:
 - Menggunakan aturan medis dan hasil diagnostik untuk menentukan nilai HeartDisease

Evaluasi Kualitas Pelabelan:

- Kecukupan Data:
 - Dataset memiliki distribusi yang seimbang antara kelas 0 dan kelas 1, yang penting untuk menghindari bias dalam model.
 - Jumlah data (718 baris) cukup untuk melatih model pembelajaran mesin sederhana, namun model yang lebih kompleks mungkin membutuhkan lebih banyak data.
- Konsistensi dan Akurasi:
 - Menggunakan teknik encoding dan scaling yang konsisten.
 - Melakukan validasi dengan memeriksa distribusi data dan memastikan tidak ada kesalahan dalam proses pelabelan.
- Kepatuhan terhadap SOP:
 - Semua langkah dilakukan sesuai dengan Standard Operating Procedure (SOP) yang berlaku, termasuk pembersihan data, transformasi, dan pelabelan.

Rekomendasi untuk Perbaikan:

- Penambahan Data:
 - Meningkatkan volume data dengan mengumpulkan lebih banyak sampel atau menggunakan teknik augmentasi data.
- Validasi Eksternal:
 - Melakukan validasi eksternal dengan data dari sumber lain untuk memastikan generalisasi model.

Kesimpulan:

- Proses pelabelan telah dilakukan dengan baik dan sesuai dengan SOP. Dataset yang dihasilkan memiliki distribusi yang seimbang dan siap digunakan untuk pembangunan model prediktif.

- Evaluasi menunjukkan bahwa data cukup memadai untuk tujuan teknis data science, meskipun penambahan data dan validasi lebih lanjut dapat lebih meningkatkan kualitas dan keandalan model yang akan dibangun.

Dengan ini, kami menyimpulkan bahwa proses pelabelan telah memenuhi standar kualitas yang diharapkan dan data siap untuk digunakan dalam analisis dan pembangunan model prediktif.

BUKTI 8-ADS

Kode Unit	:	J.62DMI00.013.1
Judul Unit	:	Membangun Model

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam membangun model.

Langkah Kerja:

- 1) Menyiapkan parameter model
- 2) Menggunakan tools pemodelan

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer dan peralatannya
 - Perangkat lunak data science di antaranya: rapid miner, weka, atau development untuk bahasa pemrograman tertentu seperti Python atau R.
- Perlengkapan
 - Dokumen best practices kriteria dan evaluasi penilaian

1. PARAMETER MODEL

Instruksi Kerja:

- Identifikasi parameter-parameter yang sesuai dengan model
- Tetapkan nilai toleransi parameter evaluasi pengujian sesuai dengan tujuan teknis

Jawab:

A.) Dalam pembangunan model pembelajaran mesin, terutama untuk klasifikasi penyakit jantung, pemilihan dan pengaturan parameter yang tepat sangat penting untuk meningkatkan kinerja model. Berikut adalah parameter-parameter utama yang sesuai dengan model Decision Tree dan XGBoost yang telah kita bangun:

Decision Tree:

Parameter utama yang perlu diatur dalam model Decision Tree adalah:

- **criterion:** Fungsi untuk mengukur kualitas split.
 - Pilihan: `gini`, `entropy`
 - Default: `gini`
 - Pengaruh: `gini` menghitung impurity berdasarkan indeks Gini, sementara `entropy` menggunakan pengukuran entropi.
- **splitter:** Strategi yang digunakan untuk memecah node.
 - Pilihan: `best`, `random`

- Default: `best`
- Pengaruh: `best` memilih fitur terbaik untuk split, sedangkan `random` memilih fitur secara acak.
- **max_depth:** Kedalaman maksimum pohon.
 - Default: `None`
 - Pengaruh: Mengendalikan overfitting (kedalaman besar) dan underfitting (kedalaman kecil).
- **min_samples_split:** Jumlah minimum sampel yang diperlukan untuk membagi node internal.
 - Default: `2`
 - Pengaruh: Mengendalikan ukuran pohon dan mempengaruhi kompleksitas model.
- **min_samples_leaf:** Jumlah minimum sampel yang diperlukan di node daun.
 - Default: `1`
 - Pengaruh: Memastikan bahwa setiap node daun memiliki minimal sampel tertentu, mengurangi overfitting.
- **max_features:** Jumlah fitur untuk dipertimbangkan saat mencari split terbaik.
 - Default: `None`
 - Pilihan: `int, float, auto, sqrt, log2`
 - Pengaruh: Mengontrol keragaman dalam pohon yang dapat meningkatkan performa.

XGBoost:

Parameter utama yang perlu diatur dalam model XGBoost adalah:

- **booster:** Jenis booster yang akan digunakan.
 - Pilihan: `gbtree, gblinear, dart`
 - Default: `gbtree`
 - Pengaruh: Memilih tipe booster yang mempengaruhi cara model belajar dari data.
- **learning_rate** (alias `eta`): Menurunkan berat dari setiap pohon tambahan.
 - Default: `0.3`
 - Pengaruh: Mengontrol kecepatan pembelajaran dan mencegah overfitting.
- **n_estimators:** Jumlah total pohon yang akan dibangun.
 - Default: `100`
 - Pengaruh: Lebih banyak pohon meningkatkan performa namun juga meningkatkan waktu komputasi.
- **max_depth:** Kedalaman maksimum pohon.
 - Default: `6`
 - Pengaruh: Mengontrol kompleksitas model, kedalaman besar dapat menyebabkan overfitting.
- **min_child_weight:** Bobot minimum sum dari sampel untuk setiap leaf node.
 - Default: `1`
 - Pengaruh: Mengontrol overfitting, nilai besar berarti model lebih konservatif.
- **subsample:** Proporsi sampel yang digunakan untuk membangun setiap pohon.
 - Default: `1.0`
 - Pengaruh: Mengontrol overfitting, nilai kecil berarti lebih banyak regularisasi.

- **colsample_bytree**: Proporsi fitur yang akan digunakan oleh setiap pohon.
 - Default: 1 . 0
 - Pengaruh: Mengurangi overfitting dengan memvariasikan fitur yang digunakan untuk setiap pohon.
- **gamma**: Minimum reduction loss yang diperlukan untuk melakukan split lebih lanjut.
 - Default: 0
 - Pengaruh: Menambah regularisasi, nilai besar membuat model lebih konservatif.
- **reg_alpha**: Regularization term on weights (L1 regularization).
 - Default: 0
 - Pengaruh: Mengontrol overfitting, nilai besar menyebabkan regularisasi lebih kuat.
- **reg_lambda**: Regularization term on weights (L2 regularization).
 - Default: 1
 - Pengaruh: Mengontrol overfitting, nilai besar menyebabkan regularisasi lebih kuat.

B.) Untuk menetapkan nilai toleransi parameter evaluasi pengujian dalam konteks model klasifikasi penyakit jantung menggunakan Decision Tree dan XGBoost, kita harus mempertimbangkan metrik kinerja yang paling relevan dan tujuan teknis dari proyek ini. Berikut adalah langkah-langkah dan contoh kode yang dapat digunakan:

Menentukan Metrik Evaluasi

Metrik kinerja yang umum digunakan untuk klasifikasi adalah:

- **Accuracy**: Persentase prediksi yang benar dari total prediksi.
- **Precision**: Persentase prediksi positif yang benar.
- **Recall (Sensitivity)**: Persentase kasus positif yang benar terdeteksi.
- **F1 Score**: Harmonic mean dari precision dan recall.

Menentukan Nilai Toleransi

Nilai toleransi harus ditetapkan berdasarkan trade-offs antara metrik evaluasi. Misalnya, dalam konteks medis, recall yang tinggi mungkin lebih diutamakan untuk menangkap semua kasus penyakit jantung, meskipun dengan mengorbankan precision.

Menetapkan Target Toleransi

Berdasarkan analisis dan tujuan teknis, tetapkan target toleransi untuk masing-masing metrik kinerja. Misalnya:

- **Accuracy**: $\geq 80\%$
- **Precision**: $\geq 75\%$
- **Recall**: $\geq 85\%$
- **F1 Score**: $\geq 80\%$

Hasil Implementasi Evaluation Tolerance

- Decision Tree:

```
Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Tuned Decision Tree Classifier:
      precision    recall  f1-score   support

     0       0.82       0.91       0.86         77
     1       0.88       0.78       0.83         67

 accuracy          0.85         144
 macro avg         0.85         0.84       0.84         144
 weighted avg      0.85         0.85       0.85         144

Accuracy score of Tuned Decision Tree Classifier: 0.8472222222222222
Best hyperparameters found by GridSearchCV for Decision Tree:
{'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
Accuracy: 0.85
Precision: 0.88
Recall: 0.78
F1 Score: 0.83

Evaluation Results Based on Tolerance (Decision Tree):
Accuracy: Pass
Precision: Pass
Recall: Fail
F1: Pass
```

Berdasarkan hasil evaluasi dari model Decision Tree tersebut, untuk precision dan recall nya dapat melampaui threshold yang ditetapkan, akan tetapi untuk recall dan F1-Score tidak melampaui threshold.

- XGBoost

```
XGBoost Classifier:
      precision    recall  f1-score   support

     0       0.90       0.94       0.92         77
     1       0.92       0.88       0.90         67

 accuracy          0.91         144
 macro avg         0.91         0.91       0.91         144
 weighted avg      0.91         0.91       0.91         144

Accuracy score of XGBoost Classifier: 0.9097222222222222
Accuracy: 0.91
Precision: 0.92
Recall: 0.88
F1 Score: 0.90

Evaluation Results Based on Tolerance (XGBoost):
Accuracy: Pass
Precision: Pass
Recall: Pass
F1: Pass
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:160: UserWarning: [08:01:35] WARNING: /workspace/src/learner.cc:742:
Parameters: { "colsample_bytree", "gamma", "max_depth", "min_child_weight", "subsample" } are not used.
```

Berdasarkan hasil evaluasi dari model XGBoost tersebut, tampak sangat bagus karena dapat melampaui threshold yang sudah ditetapkan sebelumnya serta memiliki akurasi yang sangat tinggi.

2. TOOLS PEMODELAN

Instruksi Kerja:

- Identifikasi tools untuk membuat model sesuai dengan tujuan teknis data science
- Bangun algoritma untuk teknik pemodelan yang ditentukan menggunakan tools yang dipilih

- Eksekusi algoritma pemodelan sesuai dengan skenario pengujian dan tools untuk membuat model yang telah ditetapkan
- Optimasi parameter model algoritma untuk menghasilkan nilai parameter evaluasi yang sesuai dengan skenario pengujian

Jawab:

A.) Dalam data science, pemilihan tools yang tepat untuk membuat model sangat penting untuk memastikan proses berjalan efisien dan hasil yang akurat. Berikut adalah beberapa tools yang sering digunakan dalam berbagai tahap pembuatan model data science:

- **Python:**
 - **Scikit-learn:** Library ini sangat populer untuk machine learning dan menyediakan berbagai algoritma klasifikasi, regresi, dan clustering. Scikit-learn juga mendukung preprocessing data, pemilihan model, validasi, dan evaluasi.
 - **Pandas:** Digunakan untuk manipulasi dan analisis data. Sangat berguna untuk data cleaning, eksplorasi data, dan manipulasi data frame.
 - **NumPy:** Library ini sangat penting untuk operasi numerik, terutama operasi array dan matriks, yang merupakan dasar dari banyak algoritma machine learning.
 - **Matplotlib & Seaborn:** Digunakan untuk visualisasi data. Membantu dalam eksplorasi data dan presentasi hasil analisis.
 - **XGBoost:** Implementasi gradient boosting yang efisien dan sering digunakan untuk kompetisi machine learning karena performanya yang kuat.
- **Tools untuk Manajemen Proyek Data Science:**
 - **Jupyter Notebook by Google Colab:** Platform interaktif yang mendukung banyak bahasa pemrograman, termasuk Python. Sangat berguna untuk eksplorasi data, analisis data, dan dokumentasi.

Rekomendasi Pemilihan Tools

Berdasarkan kebutuhan teknis dan tujuan proyek data science, berikut adalah rekomendasi untuk pemilihan tools:

- **Eksplorasi Data dan Preprocessing:**
 - Gunakan **Pandas** dan **NumPy** untuk manipulasi dan analisis data.
 - Gunakan **Matplotlib** dan **Seaborn** untuk visualisasi data.
- **Pemodelan Machine Learning:**
 - Gunakan **Scikit-learn** untuk algoritma machine learning dasar seperti Decision Trees.
 - Gunakan **XGBoost** untuk model gradient boosting yang efisien dan powerful.
- **Validasi dan Evaluasi Model:**
 - Gunakan **Scikit-learn** untuk grid search, dan metrik evaluasi.
- **Pengembangan dan Dokumentasi:**
 - Gunakan **Jupyter Notebook** untuk eksplorasi data, pengembangan model, dan dokumentasi hasil analisis.

Dengan menggunakan tools yang tepat, proses pembuatan model data science dapat menjadi lebih efisien dan efektif, memungkinkan tim untuk fokus pada analisis dan interpretasi hasil untuk mencapai tujuan teknis yang diinginkan.

B.) Decision Tree:

```
5.1. Decision Tree

[206] 1 # Modeling dengan Decision Tree
      2 dt_model = DecisionTreeClassifier(random_state=42)
      3 dt_model.fit(X_train_transformed, y_train)
      4
      5 # Prediksi dengan Decision Tree
      6 y_pred_dt = dt_model.predict(X_test_transformed)
      7
      8 # Menampilkan laporan klasifikasi dan akurasi
      9 print("Decision Tree Classifier:")
     10 print(classification_report(y_test, y_pred_dt, digits=2))
     11 print("Accuracy score of Decision Tree Classifier: ", accuracy_score(y_test, y_pred_dt))
```

Decision Tree Classifier:				
	precision	recall	f1-score	support
0	0.82	0.86	0.84	77
1	0.83	0.79	0.81	67
accuracy			0.83	144
macro avg	0.83	0.82	0.82	144
weighted avg	0.83	0.83	0.83	144

Accuracy score of Decision Tree Classifier: 0.8263888888888888

Gambar diatas adalah Model Algoritma Decision Tree menggunakan Jupyter Notebook dari Google Colab, Model dari Scikit-Learn, Metrics Evaluation: Akurasi, precision, recall, dan F1-Score.

XGBoost:

```
5.2 XGBoost

[208] 1 # Inisialisasi model XGBoost
      2 xgb_model = xgb.XGBClassifier(random_state=42)
      3
      4 # Latih model pada data latih yang telah ditransformasi
      5 xgb_model.fit(X_train_transformed, y_train)
      6
      7 # Prediksi pada data uji yang telah ditransformasi
      8 y_pred_xgb = xgb_model.predict(X_test_transformed)
      9
     10 # Menampilkan laporan klasifikasi dan akurasi
     11 print("XGBoost Classifier:")
     12 print(classification_report(y_test, y_pred_xgb, digits=2))
     13 print("Accuracy score of XGBoost Classifier: ", accuracy_score(y_test, y_pred_xgb))
```

XGBoost Classifier:				
	precision	recall	f1-score	support
0	0.86	0.91	0.89	77
1	0.89	0.84	0.86	67
accuracy			0.88	144
macro avg	0.88	0.87	0.87	144
weighted avg	0.88	0.88	0.87	144

Accuracy score of XGBoost Classifier: 0.875

Gambar diatas adalah Model Algoritma XGBoost menggunakan Jupyter Notebook dari Google Colab, Model dari Scikit-Learn, Metrics Evaluation: Akurasi, precision, recall, dan F1-Score.

C.) Berikut adalah Eksekusi algoritma pemodelan sesuai dengan skenario pengujian dan tools untuk membuat model

Decision Tree:

```

Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Tuned Decision Tree Classifier:

```

	precision	recall	f1-score	support
0	0.82	0.91	0.86	77
1	0.88	0.78	0.83	67
accuracy			0.85	144
macro avg	0.85	0.84	0.84	144
weighted avg	0.85	0.85	0.85	144

```

Accuracy score of Tuned Decision Tree Classifier: 0.8472222222222222
Best hyperparameters found by GridSearchCV:
{'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}

```

Gambar diatas adalah model Decision Tree menggunakan hyperparameter dan juga metrics evaluation (accuracy, precision, recall, dan F1-Score). Untuk tools yang digunakan masih sama seperti sebelumnya (seperti penjelasan diatas).

XGBoost:

```

/usr/local/lib/python3.10/dist-packages/xgboost/core.py:160: UserWarning: [07:47:42] WARNING: /workspace/src/learner.cc:742:
Parameters: { "colsample_bytree", "gamma", "max_depth", "min_child_weight", "subsample" } are not used.

warnings.warn(msg, UserWarning)
XGBoost Classifier:

```

	precision	recall	f1-score	support
0	0.90	0.94	0.92	77
1	0.92	0.88	0.90	67
accuracy			0.91	144
macro avg	0.91	0.91	0.91	144
weighted avg	0.91	0.91	0.91	144

```

Accuracy score of XGBoost Classifier: 0.9097222222222222

```

Gambar diatas adalah model XGBoost menggunakan hyperparameter dan juga metrics evaluation (accuracy, precision, recall, dan F1-Score). Untuk tools yang digunakan masih sama seperti sebelumnya (seperti penjelasan diatas).

D.) Berikut adalah optimasi parameter model algoritma untuk menghasilkan nilai parameter evaluasi yang sesuai

Decision Tree:

```
↔ Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Tuned Decision Tree Classifier:
      precision    recall  f1-score   support

     0       0.82      0.91      0.86        77
     1       0.88      0.78      0.83        67

 accuracy          0.85          144
 macro avg         0.85      0.84      0.84          144
 weighted avg      0.85      0.85      0.85          144

Accuracy score of Tuned Decision Tree Classifier: 0.8472222222222222
Best hyperparameters found by GridSearchCV for Decision Tree:
{'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
Accuracy: 0.85
Precision: 0.88
Recall: 0.78
F1 Score: 0.83

Evaluation Results Based on Tolerance (Decision Tree):
Accuracy: Pass
Precision: Pass
Recall: Fail
F1: Pass
```

Gambar diatas adalah model Decision Tree menggunakan hyperparameter dan juga evaluation. Evaluation dibagi jadi 2, ada evaluation metrics (accuracy, precision, recall, dan F1-Score) dan Evaluation Tolerance yang sudah dibuat sebelumnya (accuracy 80%, precision 75%, recall 85%, dan F1-Score 80%). Untuk tools yang digunakan masih sama seperti sebelumnya (seperti penjelasan diatas).

XGBoost:

```
↔ XGBoost Classifier:
      precision    recall  f1-score   support

     0       0.90      0.94      0.92        77
     1       0.92      0.88      0.90        67

 accuracy          0.91          144
 macro avg         0.91      0.91      0.91          144
 weighted avg      0.91      0.91      0.91          144

Accuracy score of XGBoost Classifier: 0.9097222222222222
Accuracy: 0.91
Precision: 0.92
Recall: 0.88
F1 Score: 0.90

Evaluation Results Based on Tolerance (XGBoost):
Accuracy: Pass
Precision: Pass
Recall: Pass
F1: Pass
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:160: UserWarning: [08:01:35] WARNING: /workspace/src/learner.cc:742:
Parameters: { "colsample_bytree", "gamma", "max_depth", "min_child_weight", "subsample" } are not used.
```

Gambar diatas adalah model XGBoost menggunakan hyperparameter dan juga evaluation. Evaluation dibagi jadi 2, ada evaluation metrics (accuracy, precision, recall, dan F1-Score) dan Evaluation Tolerance yang sudah dibuat sebelumnya (accuracy 80%, precision 75%, recall 85%, dan F1-Score 80%). Untuk tools yang digunakan masih sama seperti sebelumnya (seperti penjelasan diatas).

BUKTI 9-ADS

Kode Unit	:	J.62DMI00.014.1
Judul Unit	:	Mengevaluasi Hasil Pemodelan

Deskripsi:

Bukti ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam mengevaluasi hasil pemodelan.

Langkah Kerja:

- 1) Menggunakan model dengan data riil
- 2) Menilai hasil pemodelan

Peralatan dan Perlengkapan:

- Peralatan
 - Komputer
- Perlengkapan
 - Tools untuk mengeksekusi model
 - Tools untuk pengumpulan data riil

1. PENGGUNAAN MODEL DENGAN DATA RIIL

Instruksi Kerja:

- Kumpulkan data baru untuk evaluasi pemodelan sesuai kebutuhan yang mengacu kepada parameter evaluasi
- Uji model dengan menggunakan data riil yang telah dikumpulkan

Jawab:

A.) Berikut adalah mengumpulkan data baru untuk evaluasi pemodelan:

Data Preparation:

```
4. Data Preparation for Modelling

4.1 Train-Test Split

[202] 1 # Pisahkan fitur dan target
      2 X = heart_data.drop('HeartDisease', axis=1)
      3 y = heart_data['HeartDisease']

[203] 1 # Lakukan train-test split
      2 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

4.2 Scaling

[204] 1 # Buat pipeline dengan preprocessing
      2 pipeline = Pipeline([
      3     ('quantile_transform', QuantileTransformer(output_distribution='normal')),
      4     ('scaler', StandardScaler()) # Scaling fitur
      5 ])
```

Pada gambar diatas, saya sedang melakukan persiapan data untuk digunakan kedalam tahap modelling. Saya mealukan train-test split dengan rincian 80% train dan 20% test. Lalu saya membuat Pipeline yang isinya terdapat QuantileTransform (untuk merubah distribusi agar normal) dan StandardScaler (scaling data untuk menyamakan rentang nilai dari semua kolom).

```
4.3 PreProcess Data train & test

[205] 1 pipeline.fit(X_train)
      2 X_train_transformed = pipeline.transform(X_train)
      3 X_test_transformed = pipeline.transform(X_test)

/usr/local/lib/python3.10/dist-packages/sklearn/preprocessing/_data.py:100: UserWarning:
  warnings.warn(
```

Pada bagian tersebut, saya melakukan fit Pipeline terhadap X_train dan X_test.

Gambar diatas adalah data baru `Heart_Data` (setelah dilakukan pembersihan) yang siap untuk digunakan pemodelan.

B.) Berikut adalah uji model menggunakan data yang telah dikumpulkan:

Decision Tree:

```
5.1.1 Decision Tree with HyperParams

[207] 1 # Definiskan parameter grid
      2 param_grid = {
      3     'criterion': ['gini', 'entropy'],
      4     'splitter': ['best', 'random'],
      5     'max_depth': [None, 10, 20, 30],
      6     'min_samples_split': [2, 5, 10],
      7     'min_samples_leaf': [1, 2, 4],
      8     'max_features': [None, 'auto', 'sqrt', 'log2']
      9 }
     10
     11 # Buat model Decision Tree
     12 dt_model = DecisionTreeClassifier(random_state=42)
     13
     14 # Buat GridSearchCV
     15 grid_search = GridSearchCV(estimator=dt_model, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2, scoring='accuracy')
     16
     17 # Latih model dengan GridSearchCV
     18 grid_search.fit(X_train_transformed, y_train)
     19
     20 # Prediksi dengan model terbaik dari hasil GridSearchCV
     21 best_dt_model = grid_search.best_estimator_
     22 y_pred_dt = best_dt_model.predict(X_test_transformed)
     23
     24 # Menampilkan laporan klasifikasi dan akurasi
     25 print("Tuned Decision Tree Classifier:")
     26 print(classification_report(y_test, y_pred_dt, digits=2))
     27 print("Accuracy score of Tuned Decision Tree Classifier: ", accuracy_score(y_test, y_pred_dt))
     28
     29 # Menampilkan hyperparameter terbaik
     30 print("Best hyperparameters found by GridSearchCV:")
     31 print(grid_search.best_params_)
```

Gambar diatas adalah data Heart_Data yang di modelkan dengan model Decision Tree dengan hyperparameter.

```

Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Tuned Decision Tree Classifier:
      precision    recall  f1-score   support

0         0.82         0.91         0.86         77
1         0.88         0.78         0.83         67

 accuracy          0.85          0.85          0.85         144
 macro avg         0.85          0.84          0.84         144
 weighted avg      0.85          0.85          0.85         144

Accuracy score of Tuned Decision Tree Classifier: 0.8472222222222222
Best hyperparameters found by GridSearchCV:
{'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}

```

Gambar diatas adalah hasil uji dari model Decision Tree menggunakan evaluation metrics

XGBoost:

```

[209] 1 # Inisialisasi model XGBoost dengan beberapa kombinasi hyperparameter yang berbeda
2 xgb_model = xgb.XGBClassifier(booster='gblinear',
3                               max_depth=5,
4                               learning_rate=0.1,
5                               n_estimators=100,
6                               min_child_weight=1,
7                               subsample=1.0,
8                               colsample_bytree=1.0,
9                               gamma=0,
10                              reg_alpha=0,
11                              reg_lambda=1,
12                              random_state=42)
13
14 # Latih model pada data latih yang telah ditransformasi
15 xgb_model.fit(X_train_transformed, y_train)
16
17 # Prediksi pada data uji yang telah ditransformasi
18 y_pred_xgb = xgb_model.predict(X_test_transformed)
19
20 # Menampilkan laporan klasifikasi dan akurasi menggunakan metrik dari XGBoost
21 print("XGBoost Classifier:")
22 print(classification_report(y_test, y_pred_xgb, digits=2))
23 print("Accuracy score of XGBoost Classifier: ", accuracy_score(y_test, y_pred_xgb))

```

Gambar diatas adalah data Heart_Data yang di modelkan dengan model XGBoost dengan hyperparameter.

```

XGBoost Classifier:
      precision    recall  f1-score   support

0         0.90         0.94         0.92         77
1         0.92         0.88         0.90         67

 accuracy          0.91          0.91          0.91         144
 macro avg         0.91          0.91          0.91         144
 weighted avg      0.91          0.91          0.91         144

Accuracy score of XGBoost Classifier: 0.9097222222222222

```

Gambar diatas adalah hasil uji dari model XGBoost menggunakan evaluation metrics

2. PENILAIAN HASIL PEMODELAN

Instruksi Kerja:

- Nilai keluaran pengujian model berdasarkan metrik kesuksesan
- Dokumentasikan hasil penilaian sesuai standar yang berlaku

Jawab:

A.) Mengidentifikasi beberapa metrik kesuksesan untuk evaluasi model. Berdasarkan evaluasi tersebut, dapat memberikan nilai keluaran pengujian model sebagai berikut:

- **Akurasi:** Akurasi model adalah salah satu metrik yang penting untuk mengevaluasi seberapa baik model dapat mengklasifikasikan data dengan benar. Nilai akurasi yang baik menunjukkan bahwa model dapat membuat prediksi yang tepat.
- **Presisi:** Presisi merupakan metrik yang mengukur proporsi dari prediksi positif yang benar dibandingkan dengan total prediksi positif. Nilai presisi yang tinggi menunjukkan bahwa model cenderung memberikan sedikit false positive.
- **Recall:** Recall, juga dikenal sebagai sensitivitas, mengukur proporsi dari positif aktual yang diprediksi dengan benar oleh model. Nilai recall yang tinggi menunjukkan bahwa model cenderung memberikan sedikit false negative.
- **F1-Score:** F1-score adalah rata-rata harmonis dari presisi dan recall. F1-score memberikan keseimbangan antara presisi dan recall, dan berguna saat terdapat ketidakseimbangan antara kelas positif dan negatif.
- **Evaluation Tolerance :** akurasi 80%, precision 75%, recall 85%, F1-Score 80%
- **Confusion Matrix :** True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN)

Decision Tree:

```
Fitting 5 folds for each of 576 candidates, totalling 2880 fits
Tuned Decision Tree Classifier:
      precision    recall  f1-score   support

      0       0.82       0.91       0.86         77
      1       0.88       0.78       0.83         67

 accuracy          0.85
macro avg          0.85       0.84       0.84         144
weighted avg       0.85       0.85       0.85         144

Accuracy score of Tuned Decision Tree Classifier: 0.8472222222222222
Best hyperparameters found by GridSearchCV for Decision Tree:
{'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
Accuracy: 0.85
Precision: 0.88
Recall: 0.78
F1 Score: 0.83

Evaluation Results Based on Tolerance (Decision Tree):
Accuracy: Pass
Precision: Pass
Recall: Fail
F1: Pass
```

Pada gambar diatas, hasil dari Decisiion Tree berdasarkan Evaluation Metrics dan Threshold sebagai berikut:

- **Accuracy = 0.85**
- **Precision = 0.88**

- **Recall = 0.78**
- **F1-Score = 0.83**
- **Threshold** : Berdasarkan threshold tersebut didapati bahwa model Decision Tree dapat melampaui nilai untuk accuracy dan precision, tetapi tidak dapat melampaui pada recall dan F1-Score.

```

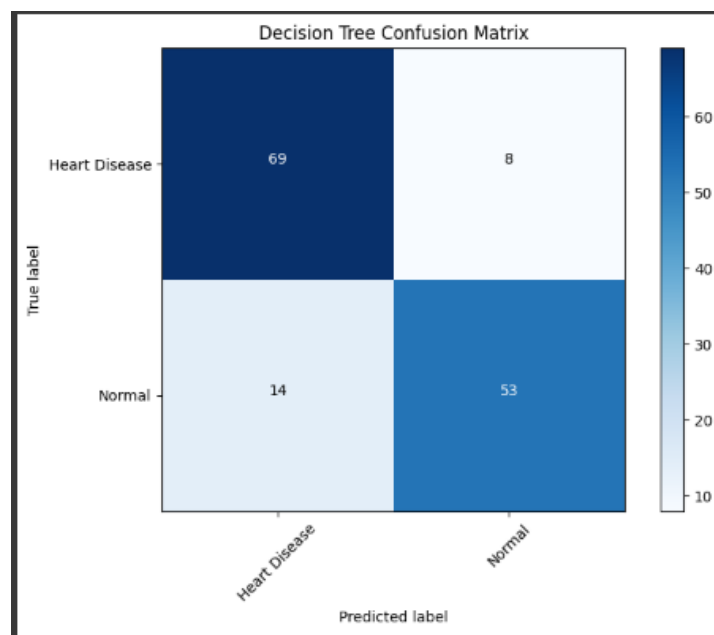
Confusion matrix, without normalization
Decision Tree Classifier:
True Negatives: 53
False Positives: 8
False Negatives: 14
True Positives: 69

Confusion matrix, without normalization
XGBoost Classifier:
True Negatives: 59
False Positives: 5
False Negatives: 8
True Positives: 72

```

Pada gambar diatas, hasil dari confusion Matrix untuk model Decision Tree sebagai berikut:

- True Positive (TP) = 69
- True Negative (TN) = 53
- False Positive (FP) = 8
- False Negative (FN) = 14



Berdasarkan visualisasi Confusion Matrix pada gambar diatas, terdapat 69 orang yang terkena penyakit jantung dan 53 orang yang tidak terkena penyakit jantung (normal).

XGBoost:

```
XGBoost Classifier:
precision    recall  f1-score   support

   0         0.90    0.94    0.92         77
   1         0.92    0.88    0.90         67

 accuracy          0.91          0.91          0.91         144
 macro avg          0.91          0.91          0.91         144
 weighted avg       0.91          0.91          0.91         144

Accuracy score of XGBoost Classifier: 0.9097222222222222
Accuracy: 0.91
Precision: 0.92
Recall: 0.88
F1 Score: 0.90

Evaluation Results Based on Tolerance (XGBoost):
Accuracy: Pass
Precision: Pass
Recall: Pass
F1: Pass
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:160: UserWarning: [08:01:35] WARNING: /workspace/src/learner.cc:742:
Parameters: { "colsample_bytree", "gamma", "max_depth", "min_child_weight", "subsample" } are not used.
```

Pada gambar diatas, hasil dari XGBoost berdasarkan Evaluation Metrics dan Threshold sebagai berikut:

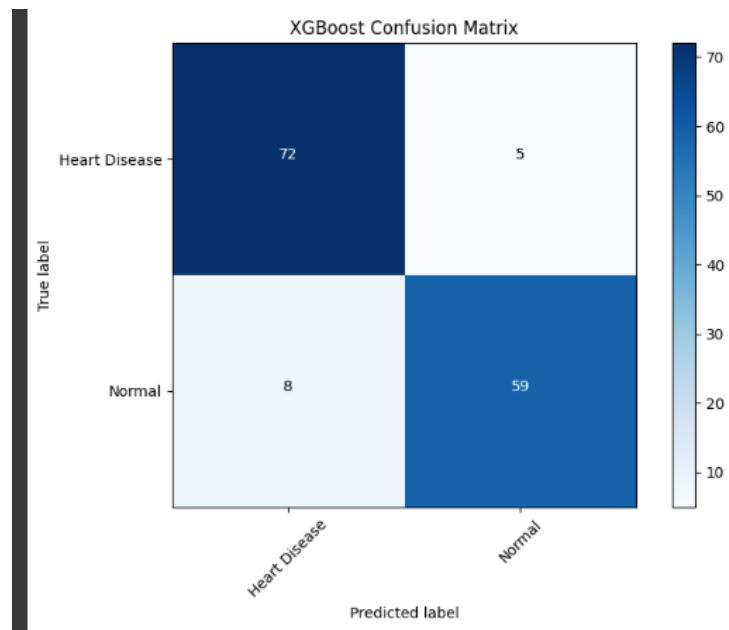
- **Accuracy = 0.91**
- **Precision = 0.92**
- **Recall = 0.88**
- **F1-Score = 0.90**
- **Threshold** : Berdasarkan threshold tersebut didapati bahwa model XGBoost dapat melampaui semua nilai threshold untuk accuracy, precision, recall, dan F1-Score.

```
Confusion matrix, without normalization
Decision Tree Classifier:
True Negatives: 53
False Positives: 8
False Negatives: 14
True Positives: 69

Confusion matrix, without normalization
XGBoost Classifier:
True Negatives: 59
False Positives: 5
False Negatives: 8
True Positives: 72
```

Pada gambar diatas, hasil dari confusion Matrix untuk model XGBoost sebagai berikut:

- True Positive (TP) = 72
- True Negative (TN) = 59
- False Positive (FP) = 5
- False Negative (FN) = 8



Berdasarkan visualisasi Confusion Matrix pada gambar diatas, terdapat 72 orang yang terkena penyakit jantung dan 59 orang yang tidak terkena penyakit jantung (normal).

B.) Berdasarkan penilaian model prediksi penyakit jantung yang telah dilakukan menggunakan data Heart_Data, berikut adalah dokumentasi hasil penilaian sesuai standar yang berlaku:

- **Accuracy:**
 - Hasil akurasi model Decision Tree: 0.847
 - Hasil akurasi model XGBoost: 0.909
 - Standar yang ditetapkan (Evaluation tolerance): Akurasi minimal 80%
 - Kesimpulan: Model XGBoost dan Decision Tree memenuhi standar yang ditetapkan, tetapi model XGBoost jauh lebih baik dalam hal akurasi.
- **Precision:**
 - Presisi model Decision Tree: 0.88
 - Presisi model XGBoost: 0.92
 - Standar yang ditetapkan (Evaluation Tolerance): Presisi minimal 75%
 - Kesimpulan: Kedua model memenuhi standar yang ditetapkan untuk presisi.
- **Recall (Sensitivitas):**
 - Recall model Decision Tree: 0.78
 - Recall model XGBoost: 0.88
 - Standar yang ditetapkan (Evaluation Tolerance): Recall minimal 85%
 - Model XGBoost memenuhi standar yang ditetapkan, sementara model Decision Tree tidak memenuhi standar.
- **F1-Score:**
 - F1-score model Decision Tree: 0.83
 - F1-score model XGBoost: 0.90
 - Standar yang ditetapkan (Evaluation Tolerance): F1-score minimal 80%

- Kesimpulan: Kedua model memenuhi standar yang ditetapkan untuk F1-score.

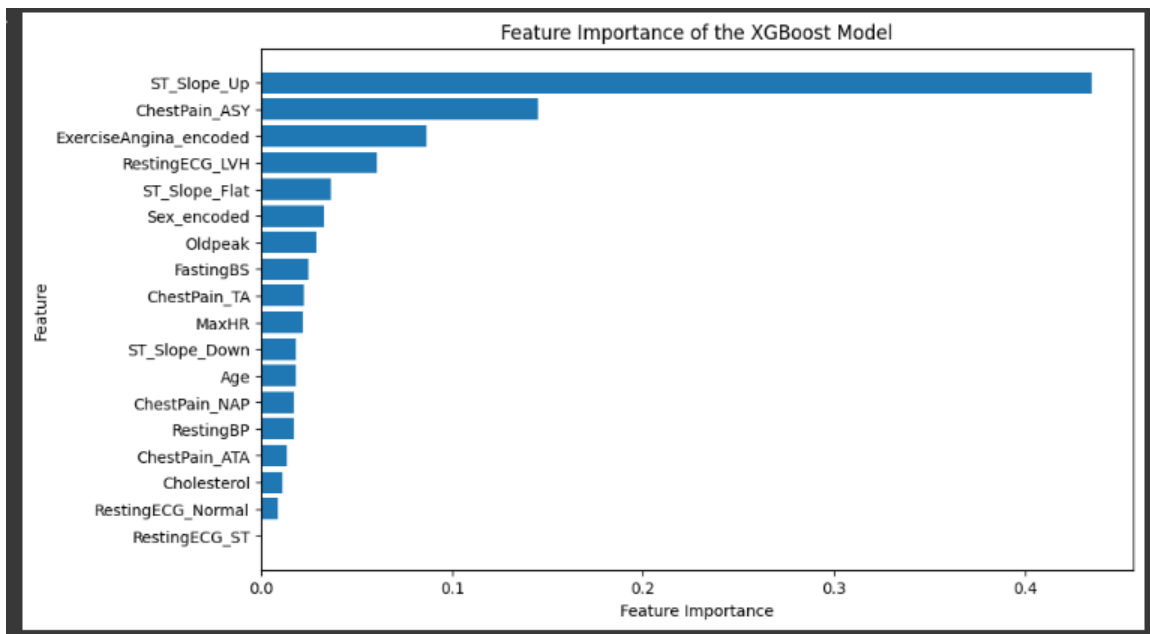
```
5.5 Best Model

[218] 1 # Hitung akurasi model Decision Tree
      2 accuracy_dt = accuracy_score(y_test, y_pred_dt)
      3
      4 # Hitung akurasi model XGBoost
      5 accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
      6
      7 # Bandingkan akurasi kedua model
      8 print("Accuracy of Decision Tree: ", accuracy_dt)
      9 print("Accuracy of XGBoost: ", accuracy_xgb)
     10
     11 # Tentukan model terbaik berdasarkan akurasi
     12 best_model = "Decision Tree" if accuracy_dt > accuracy_xgb else "XGBoost"
     13 print("Best model: ", best_model)

Accuracy of Decision Tree: 0.8472222222222222
Accuracy of XGBoost: 0.9897222222222222
Best model: XGBoost
```

Berdasarkan penilaian tersebut, dapat disimpulkan bahwa model XGBoost memiliki kinerja yang lebih baik dibandingkan dengan model Decision Tree, terutama dalam hal akurasi, presisi, Recall, dan F1-score.

Adapun Feature Importance yang ada pada model XGBoost sebagai model terbaik. Berikut adalah gambarnya:



Berdasarkan Feature Importance dari model XGBoost tersebut, didapati bahwa ada Top 3 Features yang berguna, yaitu: `ST_Slope_Up`, `ChestPain_ASY`, dan `ExerciseAngina_encoded`.