

```
from google.colab import files
uploaded = files.upload()
```



Pilih File diabetes_data_upload.csv

- **diabetes_data_upload.csv**(text/csv) - 34682 bytes, last modified: 22/1/2025 - 100% done
Saving diabetes_data_upload.csv to diabetes_data_upload.csv

✓ kodingan 1C keseluruhan

```
# Mengimpor library yang diperlukan
from google.colab import files
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# 1. Mengupload file CSV
uploaded = files.upload()

# 2. Membaca dataset
df = pd.read_csv('diabetes_data_upload.csv') # Ganti dengan nama file yang di-upload

# 3. Memeriksa tipe data dan nilai null
print("Informasi Tipe Data dan Nilai Null:")
df.info() # Menampilkan tipe data kolom dan apakah ada nilai null

# Memeriksa jumlah nilai null per kolom
print("\nJumlah Nilai Null per Kolom:")
print(df.isnull().sum())

# 4. Mengganti nama kolom
# Menyesuaikan nama kolom sesuai dengan laporan dan dataset Anda
df.columns = ['Age', 'Gender', 'Polyuria', 'Polydipsia', 'Sudden_Weight_Loss',
              'Weakness', 'Polyphagia', 'Genital_Thrush', 'Visual_Blurring',
              'Itching', 'Irritability', 'Delayed_Healing', 'Partial_Paresis',
              'Muscle_Stiffness', 'Alopecia', 'Obesity', 'Class']

# Menampilkan nama kolom setelah perubahan
print("\nNama Kolom Setelah Diganti:")
print(df.columns)

# 5. Menampilkan Statistik Deskriptif (Summary Statistik)
print("\nSummary Statistik Dataset:")
print(df.describe()) # Menampilkan ringkasan statistik seperti mean, std, min, max, dll.

# 6. Matriks Korelasi Antar Variabel
# Memilih hanya kolom numerik untuk korelasi
```

```
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns

# Menghitung matriks korelasi hanya untuk kolom numerik
correlation_matrix = df[numeric_cols].corr()

# Menampilkan heatmap matriks korelasi
plt.figure(figsize=(12,8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Matriks Korelasi Antar Fitur")
plt.show()
```



Pilih File diabetes_data_upload.csv

- **diabetes_data_upload.csv**(text/csv) - 34682 bytes, last modified: 22/1/2025 - 100% done

Saving diabetes_data_upload.csv to diabetes_data_upload (9).csv

Informasi Tipe Data dan Nilai Null:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 520 entries, 0 to 519

Data columns (total 17 columns):

#	Column	Non-Null Count	Dtype
0	Age	520 non-null	int64
1	Gender	520 non-null	object
2	Polyuria	520 non-null	object
3	Polydipsia	520 non-null	object
4	sudden weight loss	520 non-null	object
5	weakness	520 non-null	object
6	Polyphagia	520 non-null	object
7	Genital thrush	520 non-null	object
8	visual blurring	520 non-null	object
9	Itching	520 non-null	object
10	Irritability	520 non-null	object
11	delayed healing	520 non-null	object
12	partial paresis	520 non-null	object
13	muscle stiffness	520 non-null	object
14	Alopecia	520 non-null	object
15	Obesity	520 non-null	object
16	class	520 non-null	object

dtypes: int64(1), object(16)

memory usage: 69.2+ KB

Jumlah Nilai Null per Kolom:

Age	0
Gender	0
Polyuria	0
Polydipsia	0
sudden weight loss	0
weakness	0
Polyphagia	0
Genital thrush	0
visual blurring	0
Itching	0
Irritability	0
delayed healing	0
partial paresis	0
muscle stiffness	0
Alopecia	0
Obesity	0
class	0

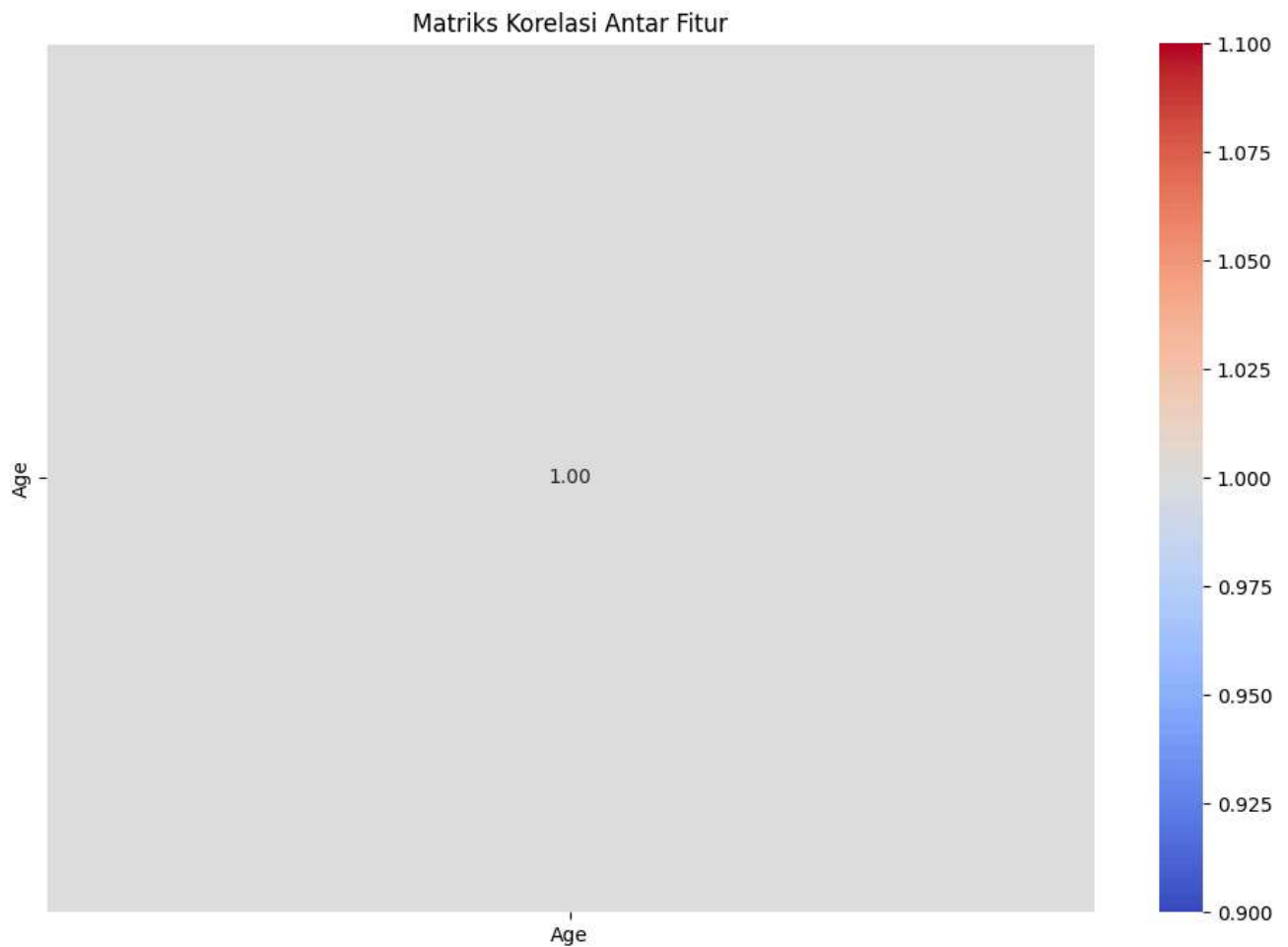
dtype: int64

Nama Kolom Setelah Diganti:

```
Index(['Age', 'Gender', 'Polyuria', 'Polydipsia', 'Sudden_Weight_Loss',
      'Weakness', 'Polyphagia', 'Genital_Thrush', 'Visual_Blurring',
      'Itching', 'Irritability', 'Delayed_Healing', 'Partial_Paresis',
      'Muscle_Stiffness', 'Alopecia', 'Obesity', 'Class'],
      dtype='object')
```

Summary Statistik Dataset:

Age
count 520.000000
mean 48.028846
std 12.151466
min 16.000000
25% 39.000000
50% 47.500000
75% 57.000000
max 90.000000



✓ Kodingan 1D keseluruhan

```

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Menggunakan LabelEncoder untuk mengubah data kategorikal menjadi numerik
le = LabelEncoder()

# Mengonversi kolom-kolom kategorikal menjadi numerik
df['Gender'] = le.fit_transform(df['Gender']) # 'Male' menjadi 1, 'Female' menjadi 0
df['Polyuria'] = le.fit_transform(df['Polyuria']) # 'Yes' menjadi 1, 'No' menjadi 0
df['Polydipsia'] = le.fit_transform(df['Polydipsia']) # 'Yes' menjadi 1, 'No' menjadi 0
df['Sudden_Weight_Loss'] = le.fit_transform(df['Sudden_Weight_Loss']) # 'Yes' menjadi 1,
df['Weakness'] = le.fit_transform(df['Weakness']) # 'Yes' menjadi 1, 'No' menjadi 0
df['Partial_Paresis'] = le.fit_transform(df['Partial_Paresis']) # 'Yes' menjadi 1, 'No' m
df['Visual_Blurring'] = le.fit_transform(df['Visual_Blurring']) # 'Yes' menjadi 1, 'No' m
df['Class'] = le.fit_transform(df['Class']) # 'Positive' menjadi 1, 'Negative' menjadi 0

# Mengonversi kolom yang memiliki string ke numerik, seperti 'Polyphagia', 'Genital_Thrush'
categorical_columns = ['Polyphagia', 'Genital_Thrush', 'Itching', 'Irritability', 'Delayed
                        'Muscle_Stiffness', 'Alopecia', 'Obesity']

for col in categorical_columns:
    df[col] = le.fit_transform(df[col]) # 'Yes' menjadi 1, 'No' menjadi 0

# Memeriksa tipe data untuk memastikan semua kolom numerik
print(df.dtypes)

# Memisahkan fitur (X) dan target (y)
X = df.drop('Class', axis=1) # Menghapus kolom 'Class' untuk fitur
y = df['Class'] # Kolom 'Class' adalah target

# Membagi dataset menjadi data latih dan data uji
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Membuat model Random Forest
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Melatih model
model.fit(X_train, y_train)

# Memprediksi hasil pada data uji
y_pred = model.predict(X_test)

# Menghitung akurasi model
accuracy = accuracy_score(y_test, y_pred)

```

```
# Menampilkan laporan klasifikasi
report = classification_report(y_test, y_pred)

# Menampilkan hasil
print(f"Akurasi Model: {accuracy:.4f}")
print("\nLaporan Klasifikasi:\n", report)
```

```
➡ Age                int64
   Gender             int64
   Polyuria           int64
   Polydipsia         int64
   Sudden_Weight_Loss int64
   Weakness           int64
   Polyphagia         int64
   Genital_Thrush     int64
   Visual_Blurring    int64
   Itching            int64
   Irritability       int64
   Delayed_Healing    int64
   Partial_Paresis    int64
   Muscle_Stiffness   int64
   Alopecia           int64
   Obesity            int64
   Class              int64
dtype: object
Akurasi Model: 0.9904
```

Laporan Klasifikasi:

	precision	recall	f1-score	support
0	0.97	1.00	0.99	33
1	1.00	0.99	0.99	71
accuracy			0.99	104
macro avg	0.99	0.99	0.99	104
weighted avg	0.99	0.99	0.99	104

✓ Kodingan 1E keseluruhan

```
# Menampilkan hasil prediksi
print("Hasil Prediksi Model:")
print(y_pred[:10]) # Menampilkan 10 hasil prediksi pertama

# Menampilkan akurasi model
print(f"Akurasi Model: {accuracy:.4f}")

# Menampilkan laporan klasifikasi lebih lanjut
print("\nLaporan Klasifikasi:")
```

```
print(report)
```

```
➞ Hasil Prediksi Model:
[0 1 1 1 1 1 0 1 0]
Akurasi Model: 0.9904
```

Laporan Klasifikasi:

	precision	recall	f1-score	support
0	0.97	1.00	0.99	33
1	1.00	0.99	0.99	71
accuracy			0.99	104
macro avg	0.99	0.99	0.99	104
weighted avg	0.99	0.99	0.99	104

soal 1 C

poin-poin dari soal 1 C

Lakukan pre-processing data dengan memeriksa tipe data, mengganti nama kolom, memeriksa nilai null, mengubah tipe data (agar bisa diproses), menampilkan summary, dan menampilkan matriks kolerasinya menggunakan metode yang pernah dipelajari.

✓ Memeriksa Tipe Data

```
# Memeriksa tipe data setiap kolom
print(df.dtypes)
```

```
➞ Age                int64
   Gender            int64
   Polyuria          int64
   Polydipsia         int64
   Sudden_Weight_Loss int64
   Weakness           int64
```

```

Polyphagia          int64
Genital_Thrush       int64
Visual_Blurring     int64
Itching             int64
Irritability        int64
Delayed_Healing     int64
Partial_Paresis     int64
Muscle_Stiffness    int64
Alopecia            int64
Obesity             int64
Class               int64
dtype: object

```

✓ Mengganti Nama Kolom

```

# Mengganti nama kolom agar lebih mudah dipahami
df.columns = ['Age', 'Gender', 'Polyuria', 'Polydipsia', 'Sudden_Weight_Loss',
              'Weakness', 'Polyphagia', 'Genital_Thrush', 'Visual_Blurring',
              'Itching', 'Irritability', 'Delayed_Healing', 'Partial_Paresis',
              'Muscle_Stiffness', 'Alopecia', 'Obesity', 'Class']

# Menampilkan nama kolom setelah perubahan
print("\nNama Kolom Setelah Diganti:")
print(df.columns)

```



```

Nama Kolom Setelah Diganti:
Index(['Age', 'Gender', 'Polyuria', 'Polydipsia', 'Sudden_Weight_Loss',
       'Weakness', 'Polyphagia', 'Genital_Thrush', 'Visual_Blurring',
       'Itching', 'Irritability', 'Delayed_Healing', 'Partial_Paresis',
       'Muscle_Stiffness', 'Alopecia', 'Obesity', 'Class'],
      dtype='object')

```

✓ Memeriksa Nilai Null

```

# Memeriksa nilai null per kolom
print("\nJumlah Nilai Null per Kolom:")
print(df.isnull().sum()) # Memeriksa jumlah nilai null di setiap kolom

```



```

Jumlah Nilai Null per Kolom:
Age          0
Gender       0
Polyuria     0
Polydipsia   0

```



```

Sudden_Weight_Loss    0
Weakness               0
Polyphagia             0
Genital_Thrush         0
Visual_Blurring        0
Itching               0
Irritability           0
Delayed_Healing        0
Partial_Paresis        0
Muscle_Stiffness       0
Alopecia               0
Obesity                0
Class                  0
dtype: int64

```

✓ Mengubah Tipe Data

```

# Mengubah tipe data kolom yang memiliki tipe data salah
df['Age'] = pd.to_numeric(df['Age'], errors='coerce') # Mengubah 'Age' menjadi numerik, jika
df['Gender'] = pd.to_numeric(df['Gender'], errors='coerce') # Mengubah 'Gender' menjadi numerik
# Mengubah data kategorikal menjadi numerik jika diperlukan, misalnya 'Yes' menjadi 1, 'No'
df['Polyuria'] = df['Polyuria'].map({'Yes': 1, 'No': 0})
df['Polydipsia'] = df['Polydipsia'].map({'Yes': 1, 'No': 0})
df['Sudden_Weight_Loss'] = df['Sudden_Weight_Loss'].map({'Yes': 1, 'No': 0})
df['Weakness'] = df['Weakness'].map({'Yes': 1, 'No': 0})

# Memeriksa tipe data setelah perubahan
print("\nTipe Data Setelah Perubahan:")
print(df.dtypes)

```



Tipe Data Setelah Perubahan:

```

Age                int64
Gender             int64
Polyuria           float64
Polydipsia         float64
Sudden_Weight_Loss float64
Weakness           float64
Polyphagia         int64
Genital_Thrush     int64
Visual_Blurring    int64
Itching            int64
Irritability       int64
Delayed_Healing    int64
Partial_Paresis    int64
Muscle_Stiffness   int64
Alopecia           int64
Obesity            int64
Class              int64
dtype: object

```

✓ Menampilkan Summary Statistik

```
# Menampilkan Summary Statistik Dataset
print("\nSummary Statistik Dataset:")
print(df.describe()) # Menampilkan statistik deskriptif dari kolom numerik
```



Summary Statistik Dataset:

	Age	Gender	Polyuria	Polydipsia	Sudden_Weight_Loss	\
count	520.000000	520.000000	0.0	0.0	0.0	
mean	48.028846	0.630769	NaN	NaN	NaN	
std	12.151466	0.483061	NaN	NaN	NaN	
min	16.000000	0.000000	NaN	NaN	NaN	
25%	39.000000	0.000000	NaN	NaN	NaN	
50%	47.500000	1.000000	NaN	NaN	NaN	
75%	57.000000	1.000000	NaN	NaN	NaN	
max	90.000000	1.000000	NaN	NaN	NaN	

	Weakness	Polyphagia	Genital_Thrush	Visual_Blurring	Itching	\
count	0.0	520.000000	520.000000	520.000000	520.000000	
mean	NaN	0.455769	0.223077	0.448077	0.486538	
std	NaN	0.498519	0.416710	0.497776	0.500300	
min	NaN	0.000000	0.000000	0.000000	0.000000	
25%	NaN	0.000000	0.000000	0.000000	0.000000	
50%	NaN	0.000000	0.000000	0.000000	0.000000	
75%	NaN	1.000000	0.000000	1.000000	1.000000	
max	NaN	1.000000	1.000000	1.000000	1.000000	

	Irritability	Delayed_Healing	Partial_Paresis	Muscle_Stiffness	\
count	520.000000	520.000000	520.000000	520.000000	
mean	0.242308	0.459615	0.430769	0.375000	
std	0.428892	0.498846	0.495661	0.484589	
min	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	
75%	0.000000	1.000000	1.000000	1.000000	
max	1.000000	1.000000	1.000000	1.000000	

	Alopecia	Obesity	Class
count	520.000000	520.000000	520.000000
mean	0.344231	0.169231	0.615385
std	0.475574	0.375317	0.486973
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	0.000000	0.000000	1.000000
75%	1.000000	0.000000	1.000000
max	1.000000	1.000000	1.000000

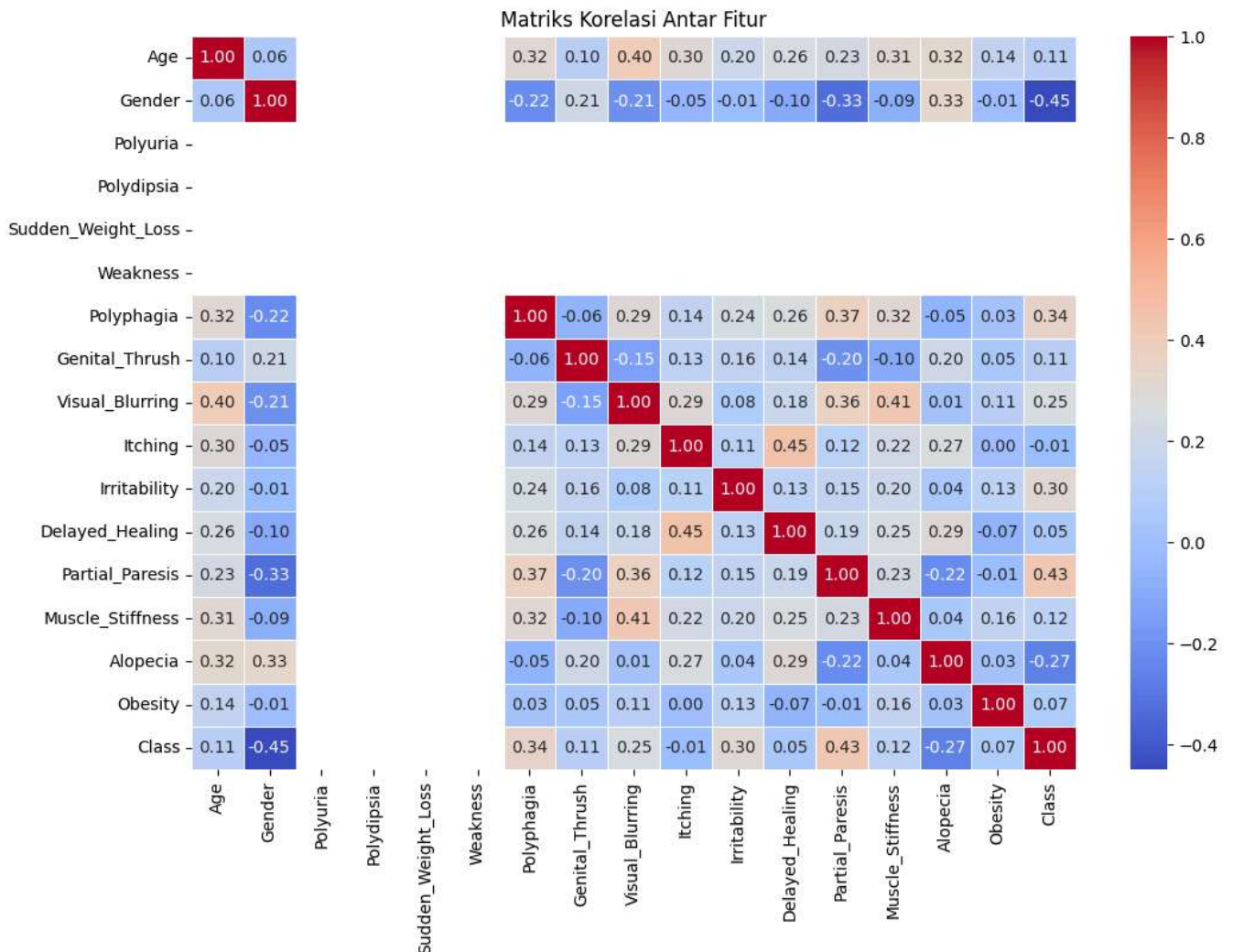
✓ Matriks Korelasi

```
# Mengimpor library untuk visualisasi
import seaborn as sns
import matplotlib.pyplot as plt

# Memilih kolom numerik untuk menghitung korelasi
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns

# Menghitung matriks korelasi antar kolom numerik
correlation_matrix = df[numeric_cols].corr()

# Menampilkan heatmap dari matriks korelasi
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Matriks Korelasi Antar Fitur")
plt.show()
```



✓ soal 1 D


poin-poin dari soal 1 D

Gunakan exploratory dan analysis (EDA) untuk melihat sudut pandang yang ada mengenai data (minimal 4) dua diantaranya bar dan pie chart, 2 di antaranya bebas. Berikan penjelasannya.

Klik dua kali (atau tekan Enter) untuk mengedit

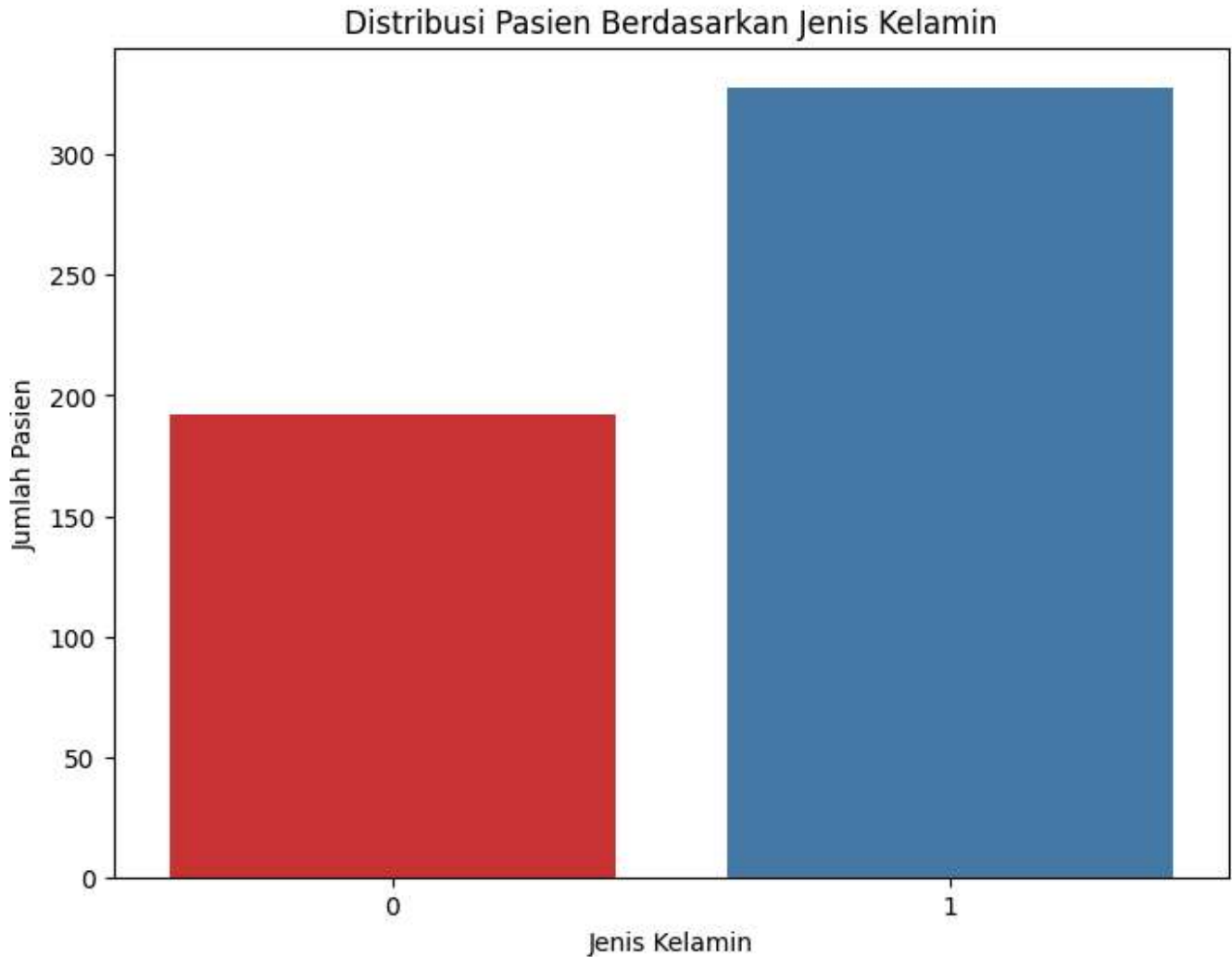
✓ **Bar Chart** - Visualisasi Distribusi Pasien Berdasarkan Jenis Kelamin

```
# Bar Chart: Distribusi Pasien Berdasarkan Jenis Kelamin
plt.figure(figsize=(8, 6))
sns.countplot(x='Gender', data=df, palette='Set1')
plt.title("Distribusi Pasien Berdasarkan Jenis Kelamin")
plt.xlabel("Jenis Kelamin")
plt.ylabel("Jumlah Pasien")
plt.show()
```

 <ipython-input-29-de7bd2c293d5>:3: FutureWarning:

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

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

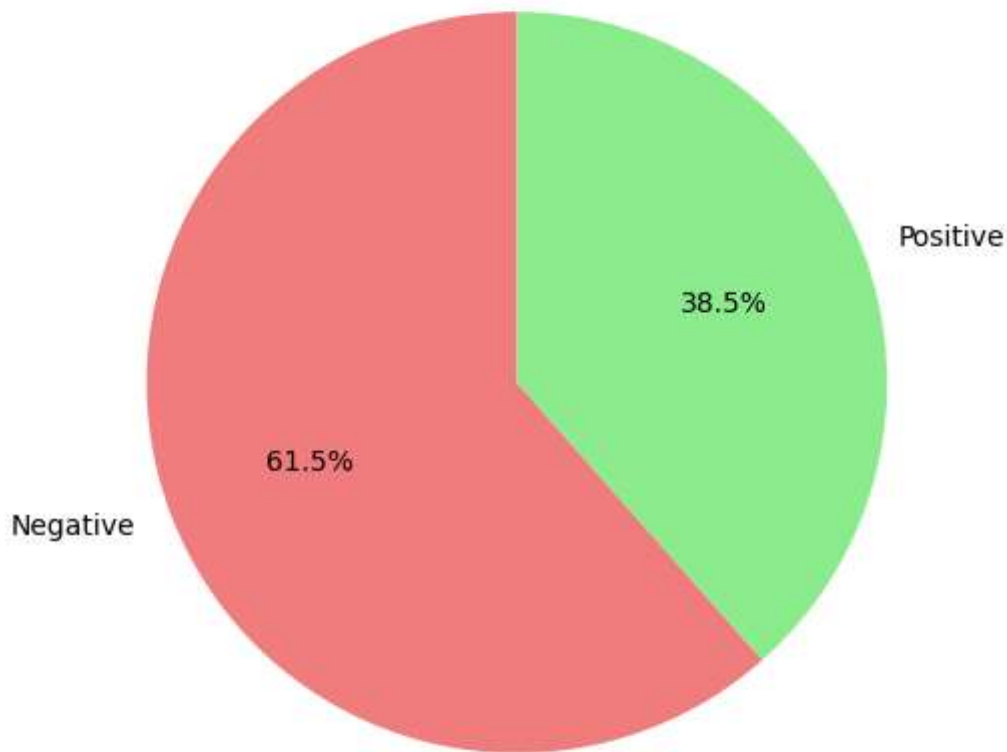


✓ **Pie Chart** - Visualisasi Pasien yang Menderita Diabetes (Positif) dan Tidak (Negatif)

```
# Pie Chart: Persentase Pasien yang Menderita Diabetes
class_counts = df['Class'].value_counts()
plt.figure(figsize=(8, 6))
plt.pie(class_counts, labels=['Negative', 'Positive'], autopct='%1.1f%%', startangle=90, col
plt.title("Persentase Pasien yang Menderita Diabetes (Positif vs Negatif)")
plt.show()
```

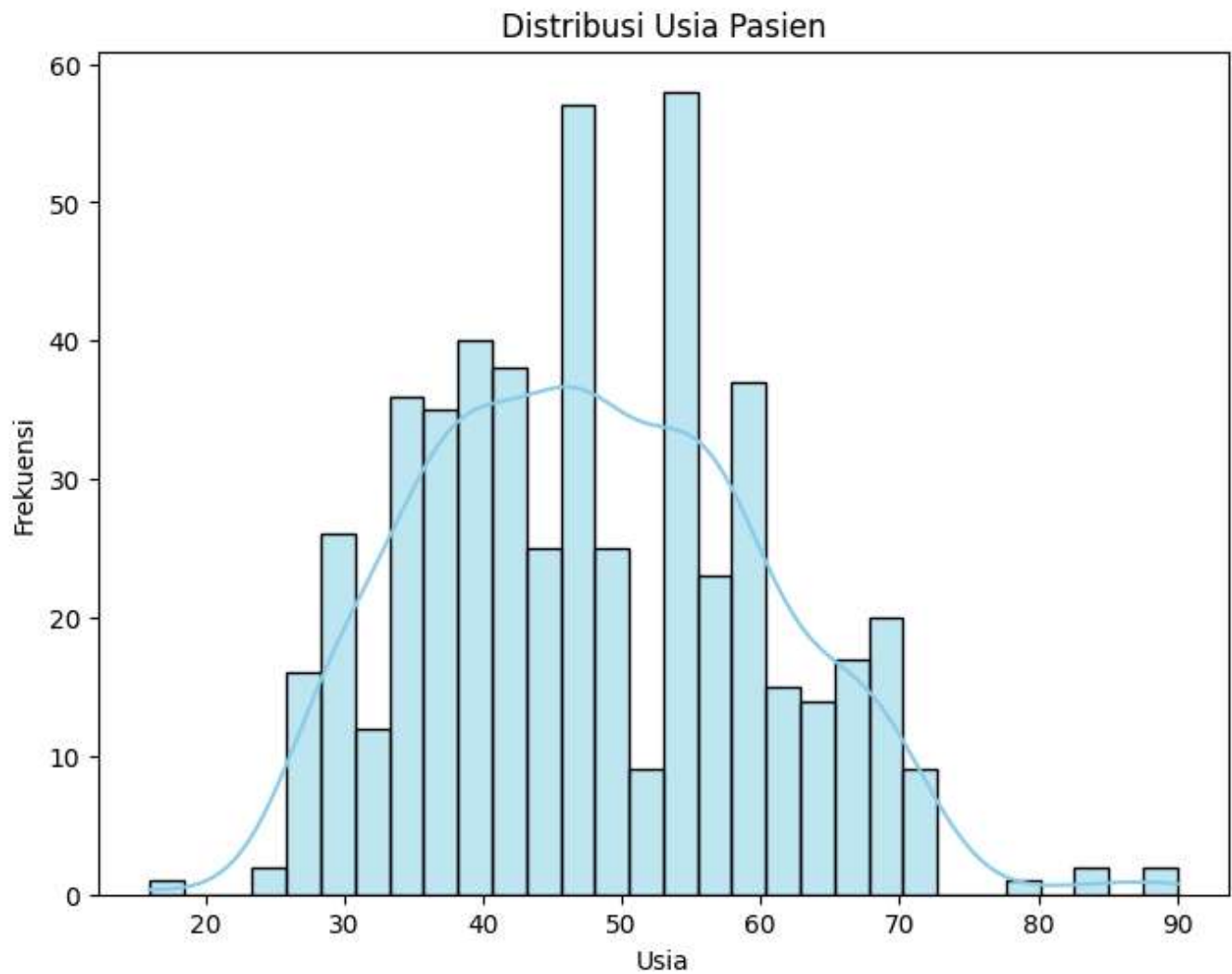


Persentase Pasien yang Menderita Diabetes (Positif vs Negatif)




✓ Histogram - Age (Distribusi Usia Pasien)

```
# Membuat histogram untuk kolom 'Age' (Usia Pasien)
plt.figure(figsize=(8, 6))
sns.histplot(df['Age'], kde=True, bins=30, color='skyblue')
plt.title("Distribusi Usia Pasien")
plt.xlabel("Usia")
plt.ylabel("Frekuensi")
plt.show()
```



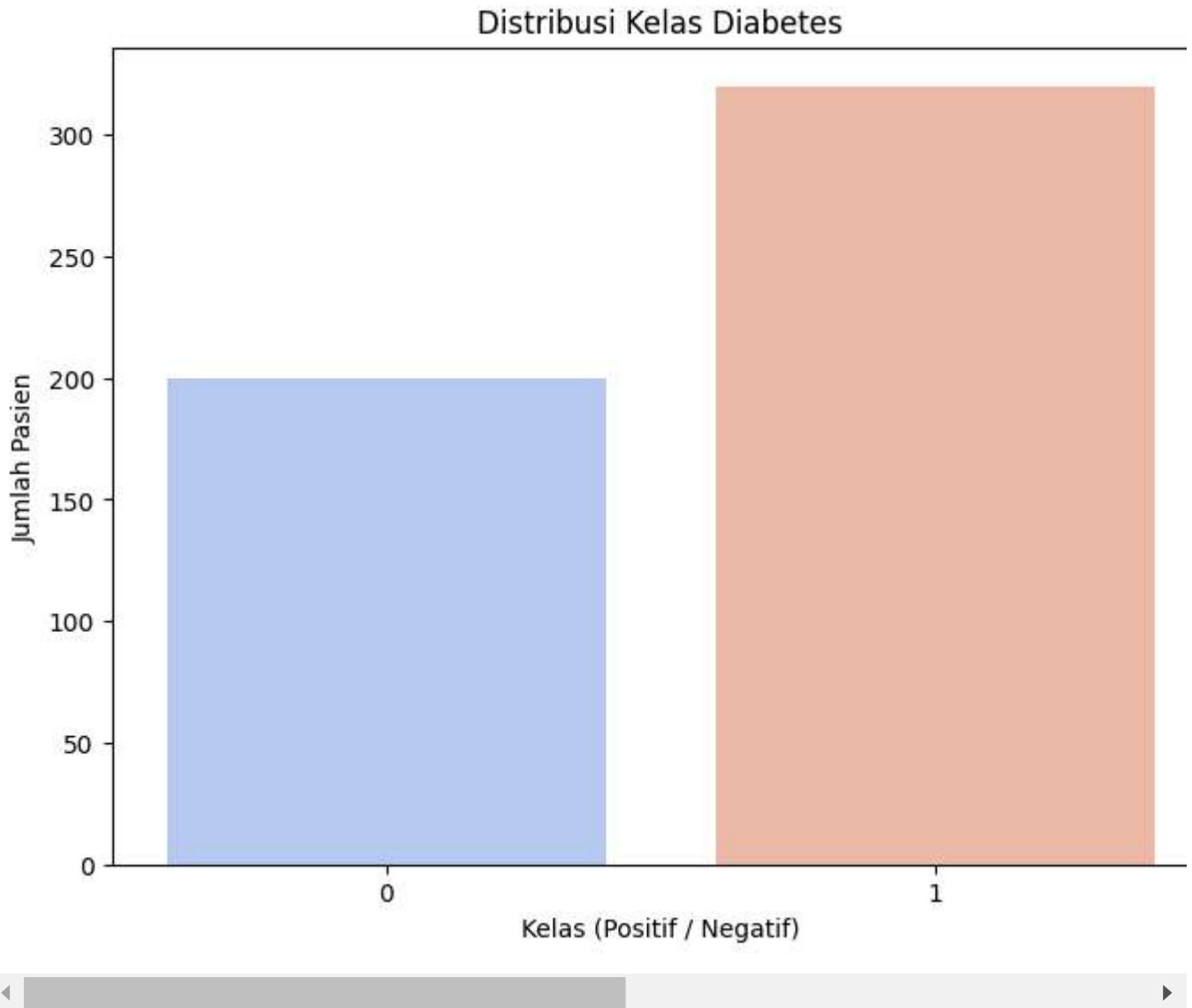
✓ Histogram - Class (Distribusi Kelas Diabetes)

```
# Membuat histogram untuk kolom 'Class' (Distribusi Kelas Diabetes)
plt.figure(figsize=(8, 6))
sns.countplot(x='Class', data=df, palette='coolwarm')
plt.title("Distribusi Kelas Diabetes")
plt.xlabel("Kelas (Positif / Negatif)")
plt.ylabel("Jumlah Pasien")
plt.show()
```

 <ipython-input-36-03c11a7957c9>:3: FutureWarning:

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

```
sns.countplot(x='Class', data=df, palette='coolwarm')
```



Klik dua kali (atau tekan Enter) untuk mengedit

✓ **Boxlot** - Untuk melihat hubungan antara kadar insulin dan hasil (apakah positif diabetes atau tidak).

```
# Menggunakan boxplot untuk melihat hubungan antara kadar insulin dan hasil (apakah positif
plt.figure(figsize=(8, 6))
sns.boxplot(x='Class', y='Age', data=df, palette='coolwarm') # Ganti 'Age' dengan kolom kac
plt.title("Hubungan Antara Kadar Insulin dan Hasil Diabetes")
plt.xlabel("Kelas (Positif / Negatif)")
```

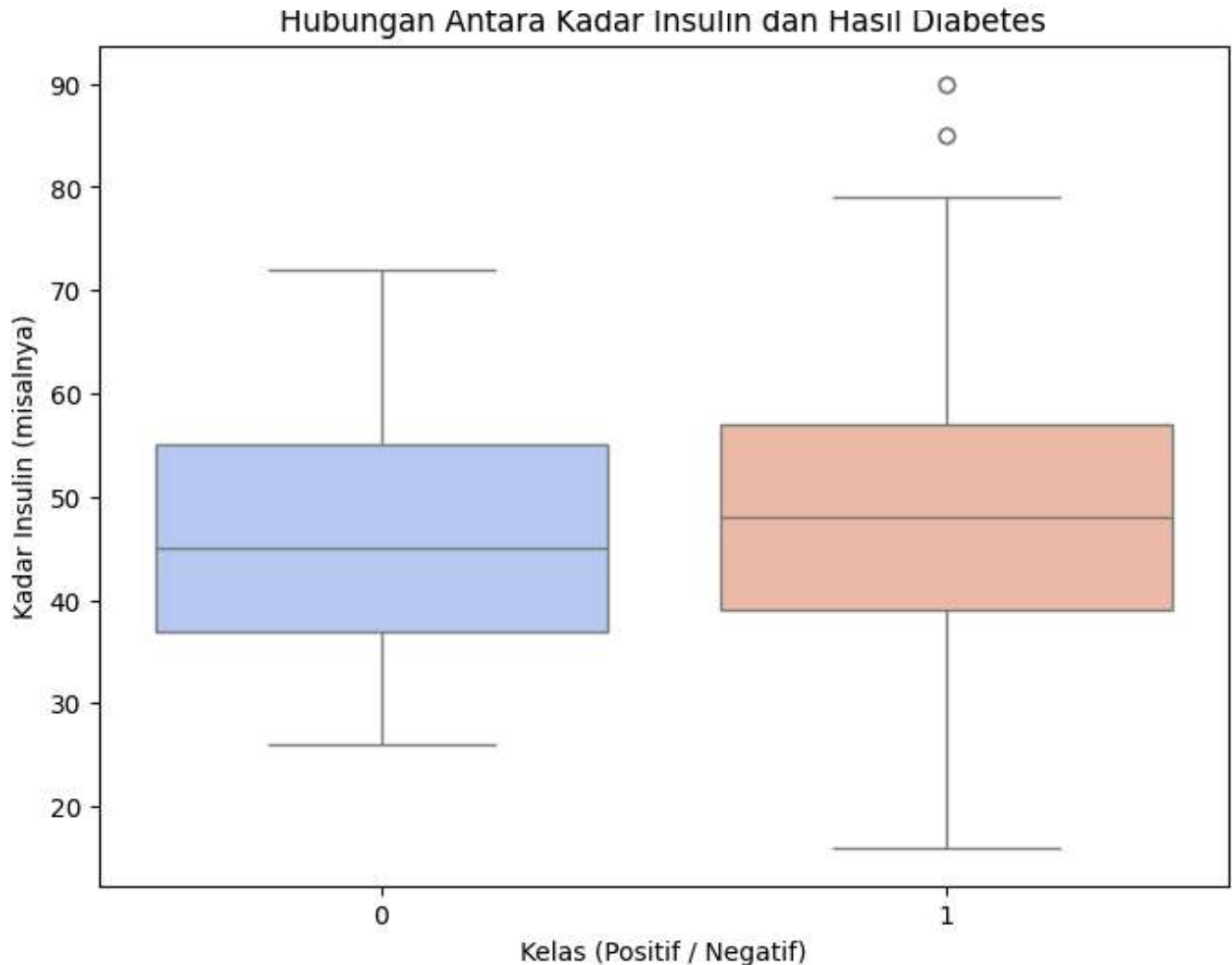


```
plt.ylabel("Kadar Insulin (misalnya)")
plt.show()
```

↳ <ipython-input-37-573628da1b70>:3: FutureWarning:

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

```
sns.boxplot(x='Class', y='Age', data=df, palette='coolwarm') # Ganti 'Age' dengan kol
```



✓ Untuk Point soal no 2 a

```
# Mengimpor library yang diperlukan
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score
```

```

# Misalnya X dan y sudah terdefinisi sebelumnya

# Membagi dataset menjadi data latih dan data uji
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 1. Logistic Regression
log_reg_model = LogisticRegression(max_iter=1000)
log_reg_model.fit(X_train, y_train)
log_reg_pred = log_reg_model.predict(X_test)
log_reg_accuracy = accuracy_score(y_test, log_reg_pred)

# 2. Random Forest Classifier
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)
rf_accuracy = accuracy_score(y_test, rf_pred)

# 3. Gradient Boosting Classifier
gb_model = GradientBoostingClassifier(n_estimators=100, random_state=42)
gb_model.fit(X_train, y_train)
gb_pred = gb_model.predict(X_test)
gb_accuracy = accuracy_score(y_test, gb_pred)

# 4. Support Vector Machine (SVM)
svm_model = SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)
svm_pred = svm_model.predict(X_test)
svm_accuracy = accuracy_score(y_test, svm_pred)

# Menampilkan hasil akurasi untuk setiap model
print(f"Akurasi Logistic Regression: {log_reg_accuracy:.4f}")
print(f"Akurasi Random Forest: {rf_accuracy:.4f}")
print(f"Akurasi Gradient Boosting: {gb_accuracy:.4f}")
print(f"Akurasi SVM: {svm_accuracy:.4f}")

```

```

➞ Akurasi Logistic Regression: 0.9231
   Akurasi Random Forest: 0.9904
   Akurasi Gradient Boosting: 0.9712
   Akurasi SVM: 0.8912

```