major4

August 29, 2025

[1]:

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

#  Define sample dataset manually
data = {
    'age': [63, 37, 41, 56, 57],
    'sex': [1, 1, 0, 1, 0],
    'cp': [3, 2, 1, 1, 0],
    'trestbps': [145, 130, 130, 120, 120],
    'chol': [233, 250, 204, 236, 354],
    'fbs': [1, 0, 0, 0, 0],
    'restecg': [0, 1, 0, 1, 0],
    'thalach': [150, 187, 172, 178, 163],
    'exang': [0, 0, 0, 0, 1],
    'oldpeak': [2.3, 3.5, 1.4, 0.8, 0.6],
    'slope': [0, 0, 2, 2, 2],
    'ca': [0, 0, 0, 0, 0],
    'thal': [1, 2, 2, 2, 2],
    'target': [1, 1, 1, 1, 0]
}

df = pd.DataFrame(data)
print("Sample dataset defined successfully.")

#  Quick overview
print("\n First 5 rows of the dataset:")
print(df.head())

print("\n Dataset info:")
print(df.info())

# Check for missing values
print("\n Missing values in each column:")
missing = df.isnull().sum()
print(missing)
```

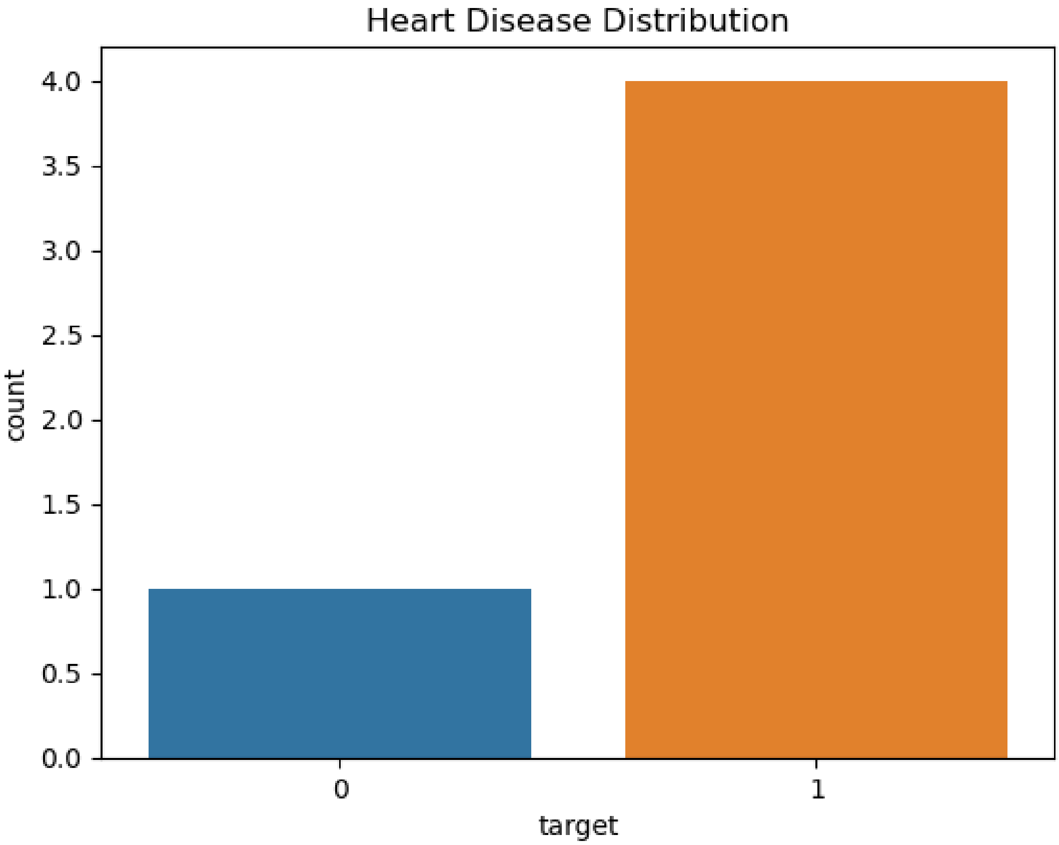
```
# Optional: Drop or fill missing values if any
if missing.sum() > 0:
    df = df.dropna()
    print("\n Missing values detected and dropped.")

#  Encode categorical variables if any
df = pd.get_dummies(df, drop_first=True)

#  Confirm 'target' column exists
if 'target' not in df.columns:
    print("  Error: 'target' column not found in dataset. Please verify your␣
 ↪label column.")
    exit()

# Feature scaling
scaler = StandardScaler()
X = scaler.fit_transform(df.drop('target', axis=1))
y = df['target']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

print("\n Preprocessing complete. Data is ready for modeling.")
print(f"Training samples: {X_train.shape[0]}, Testing samples: {X_test.
 ↪shape[0]}")
```

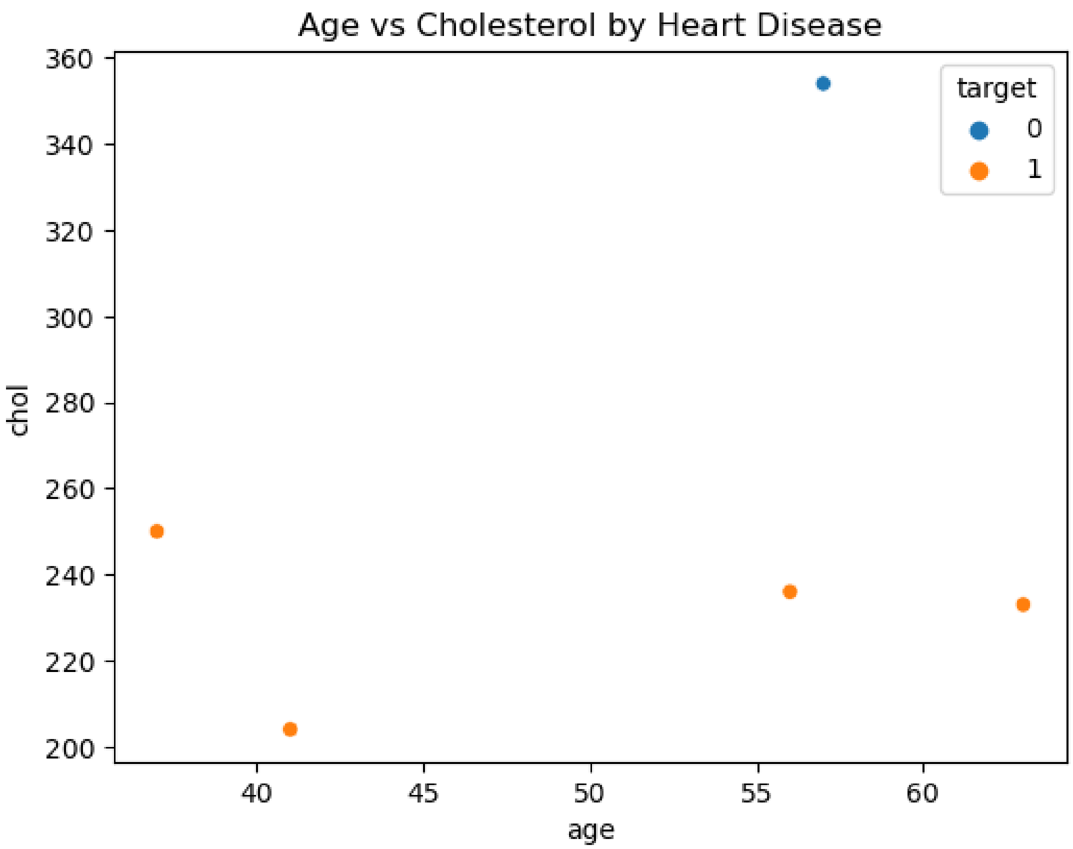
```
Sample dataset defined successfully.

 First 5 rows of the dataset:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   3       145   233    1        0      150      0      2.3      0
1   37    1   2       130   250    0        1      187      0      3.5      0
2   41    0   1       130   204    0        0      172      0      1.4      2
3   56    1   1       120   236    0        1      178      0      0.8      2
4   57    0   0       120   354    0        0      163      1      0.6      2

   ca  thal  target
0   0     1       1
1   0     2       1
2   0     2       1
3   0     2       1
4   0     2       0

 Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
```

```
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       5 non-null      int64
 1   sex       5 non-null      int64
 2   cp        5 non-null      int64
 3   trestbps  5 non-null      int64
 4   chol      5 non-null      int64
 5   fbs       5 non-null      int64
 6   restecg   5 non-null      int64
 7   thalach   5 non-null      int64
 8   exang     5 non-null      int64
 9   oldpeak   5 non-null      float64
 10  slope     5 non-null      int64
 11  ca        5 non-null      int64
 12  thal      5 non-null      int64
 13  target    5 non-null      int64
dtypes: float64(1), int64(13)
memory usage: 688.0 bytes
None

 Missing values in each column:
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64

 Preprocessing complete. Data is ready for modeling.
Training samples: 4, Testing samples: 1
```

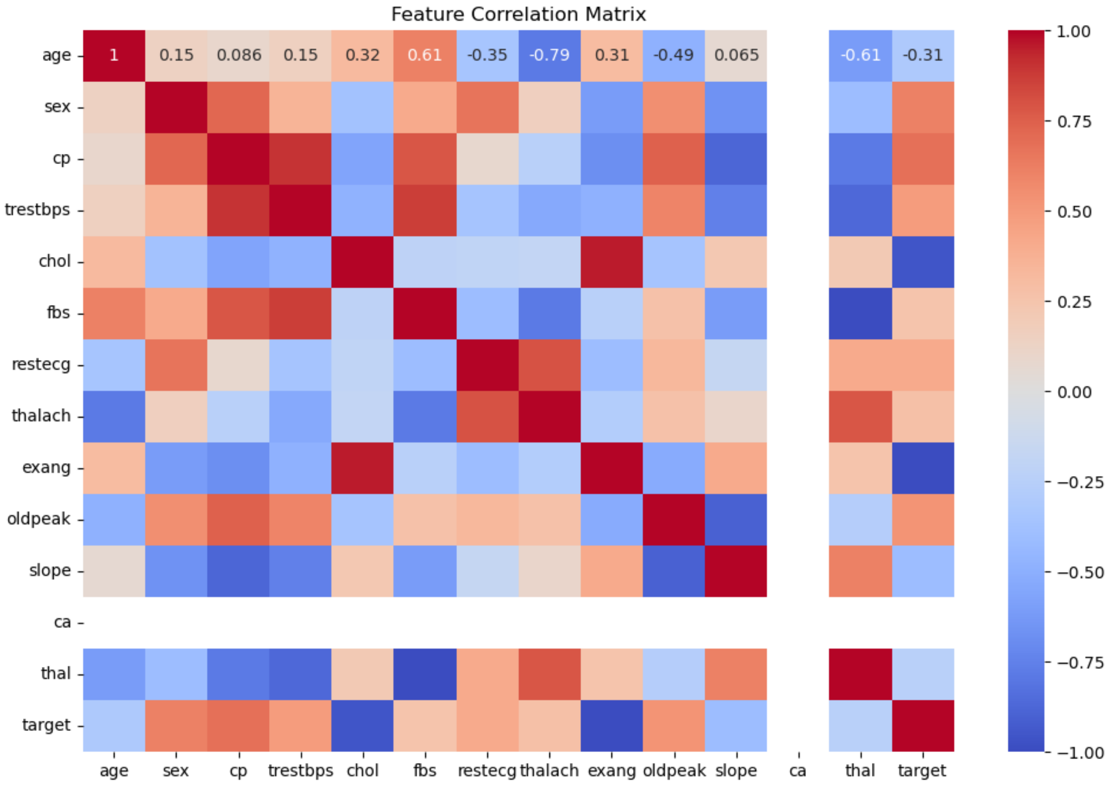
[2]:

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Distribution of target variable
sns.countplot(x='target', data=df)
plt.title('Heart Disease Distribution')
plt.show()

# Age vs. Cholesterol
sns.scatterplot(x='age', y='chol', hue='target', data=df)
plt.title('Age vs Cholesterol by Heart Disease')
plt.show()

# Correlation matrix
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Feature Correlation Matrix')
plt.show()
```





```
/opt/conda/envs/anaconda-ai-2024.04-py310/lib/python3.10/site-
packages/seaborn/matrix.py:260: FutureWarning: Format strings passed to
MaskedConstant are ignored, but in future may error or produce different
behavior
  annotation = ("{:" + self.fmt + "}").format(val)
```



[3]:
```
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier

models = {
    'SVM': SVC(),
    'KNN': KNeighborsClassifier(n_neighbors=3),  # Reduced n_neighbors to be less than n_samples
    'Decision Tree': DecisionTreeClassifier(),
    'Logistic Regression': LogisticRegression(),
    'Random Forest': RandomForestClassifier()
}

results = {}
for name, model in models.items():
    model.fit(X_train, y_train)
    preds = model.predict(X_test)
    acc = accuracy_score(y_test, preds)
    results[name] = acc
```

```python

# Display results
for model_name, accuracy in results.items():
    print(f"{model_name}: {accuracy:.2f}")
```

```
/opt/conda/envs/anaconda-ai-2024.04-py310/lib/python3.10/site-
packages/joblib/externals/loky/backend/context.py:110: UserWarning: Could not
find the number of physical cores for the following reason:
found 0 physical cores < 1
Returning the number of logical cores instead. You can silence this warning by
setting LOKY_MAX_CPU_COUNT to the number of cores you want to use.
  warnings.warn(
  File "/opt/conda/envs/anaconda-ai-2024.04-py310/lib/python3.10/site-
packages/joblib/externals/loky/backend/context.py", line 217, in
_count_physical_cores
    raise ValueError(
```

```
SVM: 1.00
KNN: 1.00
Decision Tree: 1.00
Logistic Regression: 1.00
Random Forest: 1.00
```

[4]:

```python
import joblib

best_model = RandomForestClassifier()
best_model.fit(X_train, y_train)
joblib.dump(best_model, 'heart_disease_model.pkl')
```

[4]:

```
['heart_disease_model.pkl']
```

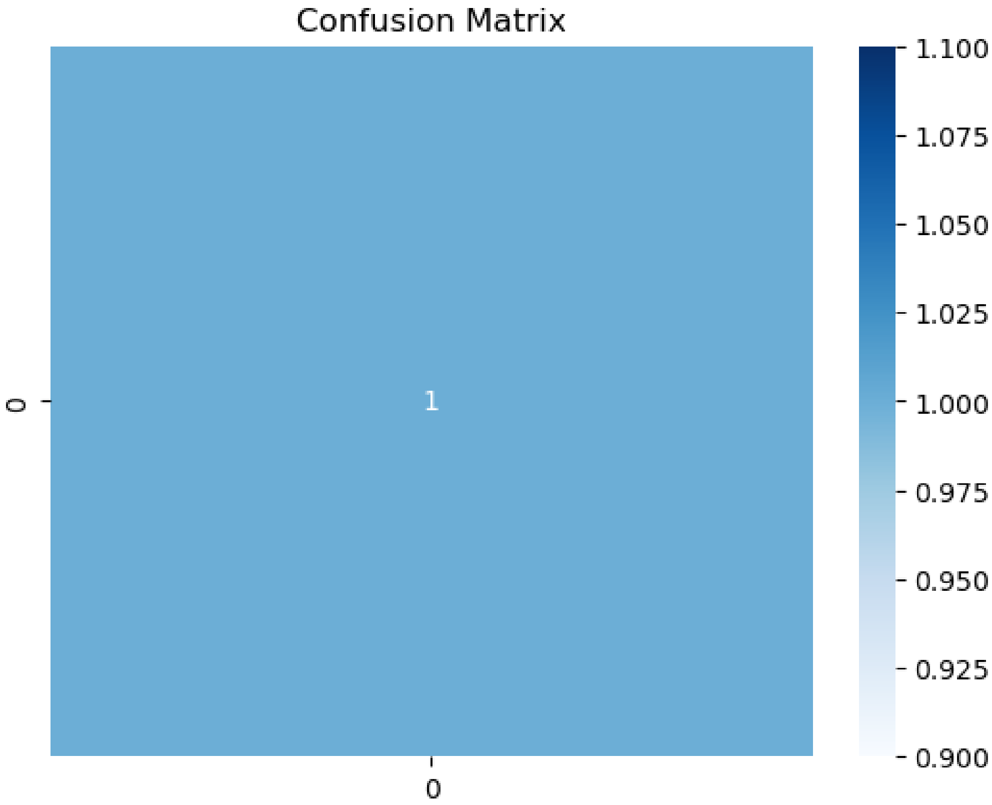
[5]:

```python
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
import numpy as np

# Reduce the number of folds to match your smallest class size
# For example, if your smallest class has 3 samples, use cv=3 or less
cv_scores = cross_val_score(best_model, X, y, cv=3)  # Reduced from 5 to 3
print(f"Cross-validation accuracy: {cv_scores.mean():.2f}")

# Alternatively, you can use StratifiedKFold with shuffle=True
# from sklearn.model_selection import StratifiedKFold
# cv = StratifiedKFold(n_splits=3, shuffle=True, random_state=42)
# cv_scores = cross_val_score(best_model, X, y, cv=cv)

# Confusion matrix
y_pred = best_model.predict(X_test)
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
```

```
plt.title('Confusion Matrix')
plt.show()
```

```
/opt/conda/envs/anaconda-ai-2024.04-py310/lib/python3.10/site-
packages/sklearn/model_selection/_split.py:725: UserWarning: The least populated
class in y has only 1 members, which is less than n_splits=3.
  warnings.warn(
```

```
Cross-validation accuracy: 0.83
```



[ ]:

```

```