author email

date 2024-05-20

output Notebook, Markdown

Load the data set + Imports

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

print('pandas:',pd. __version__)
print('seaborn',sns.__version__)

# Load dataset
df = pd.read_csv('.venv/heart_disease_uci.csv')

pandas: 2.2.2
seaborn 0.13.2
```

remove missing values

```
# Check for missing values
print(df.isnull().sum())
# Drop or impute missing values
df = df.dropna()
id
               0
               0
age
               0
sex
               0
dataset
               0
ср
trestbps
              59
chol
              30
fbs
              90
              2
restecg
              55
thalch
              55
exang
oldpeak
             62
slope
            309
            611
ca
thal
            486
```

```
num 0
dtype: int64
```

Remove Duplicates

```
# Check for duplicates
print(df.duplicated().sum())

# Remove duplicates
df = df.drop_duplicates()
0
```

Data Type Conversion

```
# Convert 'fbs' column to boolean
df['fbs'] = df['fbs'].astype(bool)

# Standardize column names
df.columns = df.columns.str.lower().str.replace(' ', '_')

# Convert categorical columns to numeric using one-hot encoding
df = pd.get_dummies(df, columns=['sex', 'cp', 'restecg', 'slope',
'thal', 'dataset'], drop_first=True)
```

Standardize Column Names

```
# Standardize column names
df.columns = df.columns.str.lower().str.replace(' ', '_')
```

START

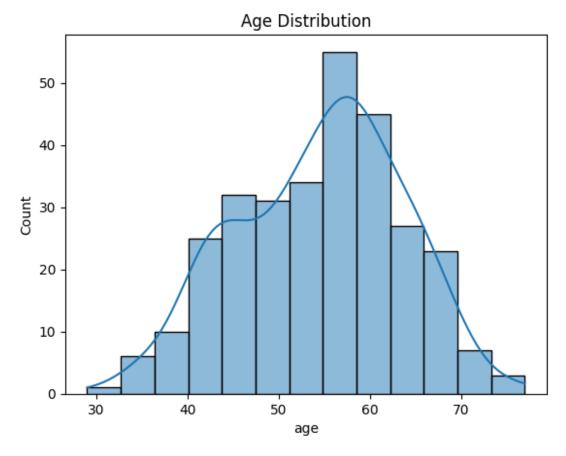
Descriptive Statistics

```
# Get summary statistics
print(df.describe())
              id
                        age trestbps
                                               chol
                                                        thalch
oldpeak \
count 299.000000
                 299.000000
                             299.000000 299.000000 299.000000
299.000000
      153.872910
                 54.521739 131.715719 246.785953 149.327759
mean
1.058528
std
       95.896287
                 9.030264
                              17.747751
                                          52.532582
                                                     23.121062
1.162769
min
        1.000000
                   29.000000
                              94.000000 100.000000
                                                     71.000000
0.000000
```

```
25%
        75.500000
                    48.000000
                               120.000000
                                           211.000000
                                                       132.500000
0.000000
50%
       151.000000
                    56.000000
                               130.000000
                                           242.000000
                                                       152.000000
0.800000
75%
       227.500000
                    61.000000
                               140.000000
                                           275.500000
                                                       165.500000
1,600000
      749.000000
                    77.000000
                               200.000000
                                           564.000000 202.000000
max
6.200000
               ca
                          num
count 299.000000
                   299.000000
        0.672241
                     0.946488
mean
                     1.230409
std
         0.937438
min
        0.000000
                     0.000000
        0.000000
                     0.000000
25%
50%
         0.000000
                     0.000000
                     2,000000
75%
         1.000000
         3.000000
                     4.000000
max
```

Data Distribution

```
# Plot distribution of age
sns.histplot(df['age'], kde=True)
plt.title('Age Distribution')
plt.show()
```



```
# Compute correlation matrix
corr = df.corr()

# Plot heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

Correlation Matrix

```
1.00
                         id - 1 .0030402-D.1-0.06-D.16.046.09 0.020.03 0.0507.0430.110.020.140.13 0.1-0.0470.0-D.01 0.210.36
                       age 0.001 1 0.29 0.2 0.130.38.0930.2 0.360.220.0940.160.04020430.16.0840.170.180.130.10.0402095
                  trestbps -0.02<mark>0.29 1 0.130.18</mark>0.05030680.190.0970.160.0605.0801.0561.150.16.0588.0270.0869.140.110.060.038
                                                                                                                                           -0.75
                      chol -0.14<mark>0.2 0.13 1 0</mark>.010.016.040.0340.120.0650.20.010.0107.0530.16.030.02090048010.036.0230.16
                        fbs-0.06<mark>0.130.18</mark>0.017 10.092904807<mark>3.15</mark>0.049.0370.05<mark>5.1D.060</mark>.065.048.05700909069026.024.024
                    thalch -0.160.380.05030305002 1 0.390.350.260.420.068.260.170.0803.0730.120.430.46 0.3-0.230.1-30.074
                                                                                                                                           - 0.50
                    exang 0.045.093.068.04B0048.39 1 0.290.140.390.150.230.240.098.07060420.260.290.330.310.0802.082
                  oldpeak -0.0910.2 0.190.03040076.350.29 1 0.29 0.5 0.110.280.110.0830.130.170.310.510.350.310.020.022
                        ca -0.02<mark>0.36</mark>0.0970.120.150.260.140.29 1 0.520.0880.150.130.060.140.0410.160.150.250.240.042.042
                                                                                                                                           - 0.25
                      num 9.03 D.220.160.065.04 90.420.39 0.5 0.52 1 0.230.240.280.11-0.20.120.34-0.40.510.407.00 250 25
                 sex_male -0.0507.0994.0650.20.0307.068.150.1 D.0880.23 1 D.0449.1 D.0901.01-70.1 d.00503.020.390.330.040.04
      cp_atypical_angina 4.0430.160.081.013.058.260.230.280.150.240.04 1 0.270.130.10.0580.210.240.220.240.026.026
                                                                                                                                           - 0.00
          cp_non-anginal -0.1-0.0402.0505.010.110.170.260.120.130.280.120.27 1 0.18.080400-00208040960.2-0.1-0.0305.036
       cp_typical_angina -0.02.0430.150.0530600.080.09060830.060.100.0930.130.18 130.0602.08400707.04300907.0206.0107.017
          restecg_normal -0.140.160.150.1-8.060507-8.07-6.130.14-0.20.0170.10.084.0621 -0.120.110.140.01070095054.058
                                                                                                                                           - -0.25
restecg_st-t_abnormality -0.130.084.058.036.0480.120.0420.120.040.12-0.1-0.05200-0203-0.12 1 0.06-70.1-0.0-10.0-3400680067
                 slope_flat - 0.1 0.170.020.029.05 0.430.260.310.160.39.00503.2-0.0840070.10.06 1 1 -0.870.270.2 0.0602.062
         slope_upsloping -0.0470.148.04880049800199460.250.51-0.15-0.4-0.020.240.096.048.140.110.87 1 0.320.240.056.054
                                                                                                                                           -0.50
              thal_normal -0.010.130.140.016.0690.3-0.330.350.250.510.390.22 0.20.009070147.0140.270.32 1 -0.80.064.064
  thal_reversable_defect -0.0150.1 0.1 D.030.0230.230.310.310.210.470.330.210.10.0 D0009D5034.210.240.88 1 0.070.072
        dataset hungary -0.230.046.060.023.024.18.082.022.042.042025.040.026.036.0107.050006.05062.059.0604072 11.0034
                                                                                                                                            -0.75
 dataset va long_beach -0.366.009050340.140.0246.07440802.0242.004200250.040.0206.0306.0107.05080067062.0548.0640702005
                                                                                                     slope_flat
                                                                                                          slope_upsloping
                                                                                                              thal_normal
                                                                                                                   thal_reversable_defect
                                                                                                                       dataset_hungary
                                                                               cp_atypical_angina
                                                                                        cp_typical_angina
                                                                                             restecg_normal
                                                                                                 estecg st-t_abnormality
```

```
# Define features and target
X = df.drop('num', axis=1)
y = df['num']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

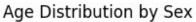
# Train a Random Forest model
model = RandomForestClassifier()
model.fit(X_train, y_train)

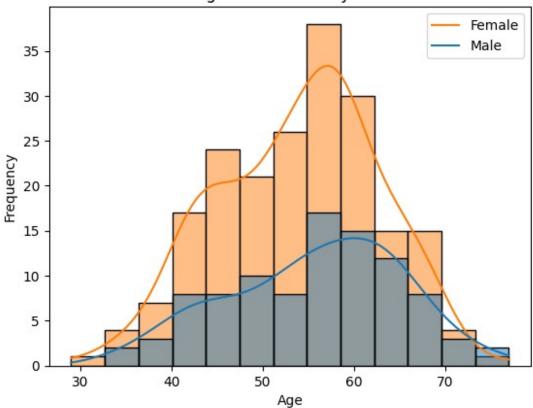
# Predict and evaluate
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
recall f1-score
              precision
                                               support
                             0.94
           0
                   0.79
                                        0.86
                                                    35
           1
                   0.12
                             0.08
                                        0.10
                                                    13
                   0.25
           2
                             0.20
                                        0.22
                                                     5
           3
                   0.17
                             0.25
                                        0.20
                                                     4
           4
                   0.00
                             0.00
                                        0.00
                                                     3
                                        0.60
                                                    60
    accuracy
                             0.29
                                                    60
   macro avq
                   0.27
                                        0.27
weighted avg
                   0.52
                             0.60
                                        0.55
                                                    60
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/
python3.12/site-packages/sklearn/metrics/ classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/python3.12/
site-packages/sklearn/metrics/ classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/python3.12/
site-packages/sklearn/metrics/ classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
```

Age Distribution by Sex

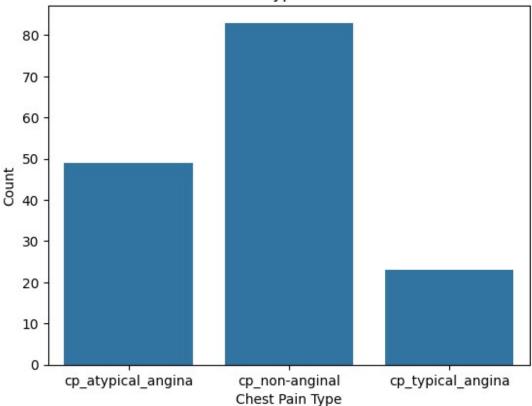
```
# Age Distribution by Sex
sns.histplot(data=df, x='age', hue='sex_male', kde=True)
plt.title('Age Distribution by Sex')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.legend(['Female', 'Male'])
plt.show()
```





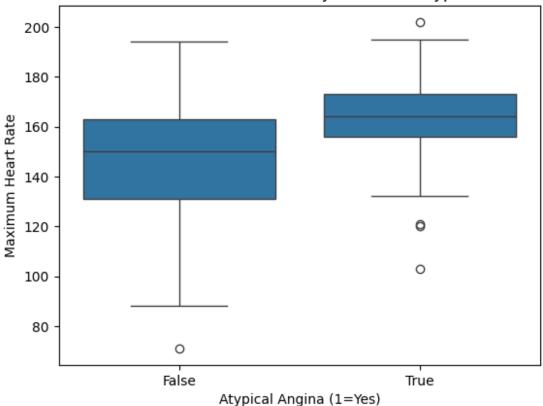
```
# Chest Pain Type Distribution
cp_cols = [col for col in df.columns if 'cp_' in col]
df_cp = df[cp_cols].sum().reset_index()
df_cp.columns = ['Chest Pain Type', 'Count']
sns.barplot(data=df_cp, x='Chest Pain Type', y='Count')
plt.title('Chest Pain Type Distribution')
plt.xlabel('Chest Pain Type')
plt.ylabel('Count')
plt.show()
```

Chest Pain Type Distribution



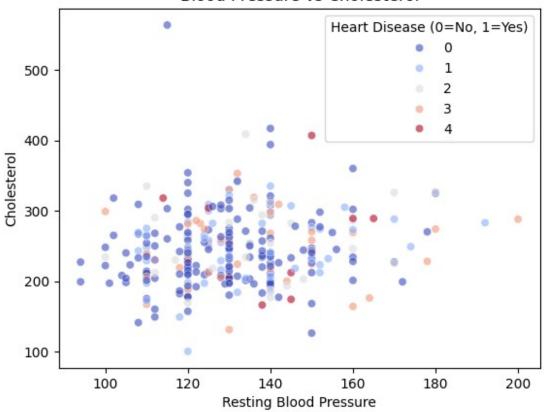
```
# Heart Rate by Chest Pain Type
sns.boxplot(data=df, x='cp_atypical_angina', y='thalch')
plt.title('Maximum Heart Rate by Chest Pain Type')
plt.xlabel('Atypical Angina (1=Yes)')
plt.ylabel('Maximum Heart Rate')
plt.show()
```

Maximum Heart Rate by Chest Pain Type

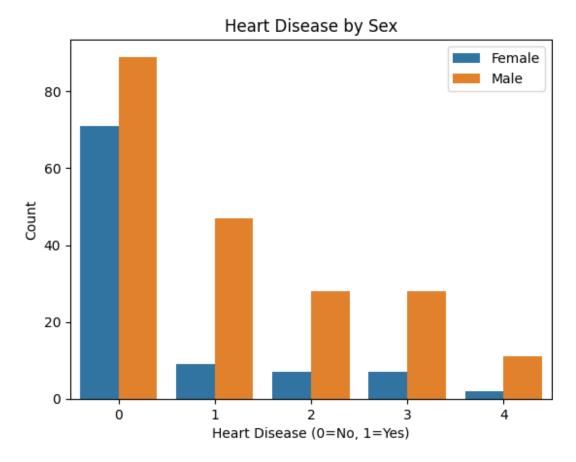


```
# Blood Pressure vs Cholesterol
sns.scatterplot(data=df, x='trestbps', y='chol', hue='num',
palette='coolwarm', alpha=0.6)
plt.title('Blood Pressure vs Cholesterol')
plt.xlabel('Resting Blood Pressure')
plt.ylabel('Cholesterol')
plt.legend(title='Heart Disease (0=No, 1=Yes)')
plt.show()
```

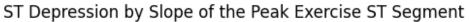
Blood Pressure vs Cholesterol

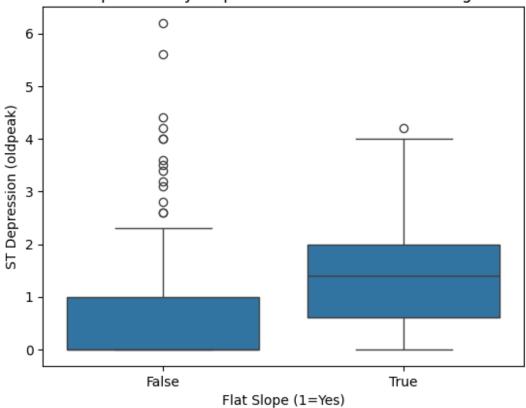


```
# Heart Disease by Sex
sns.countplot(data=df, x='num', hue='sex_male')
plt.title('Heart Disease by Sex')
plt.xlabel('Heart Disease (0=No, 1=Yes)')
plt.ylabel('Count')
plt.legend(['Female', 'Male'])
plt.show()
```

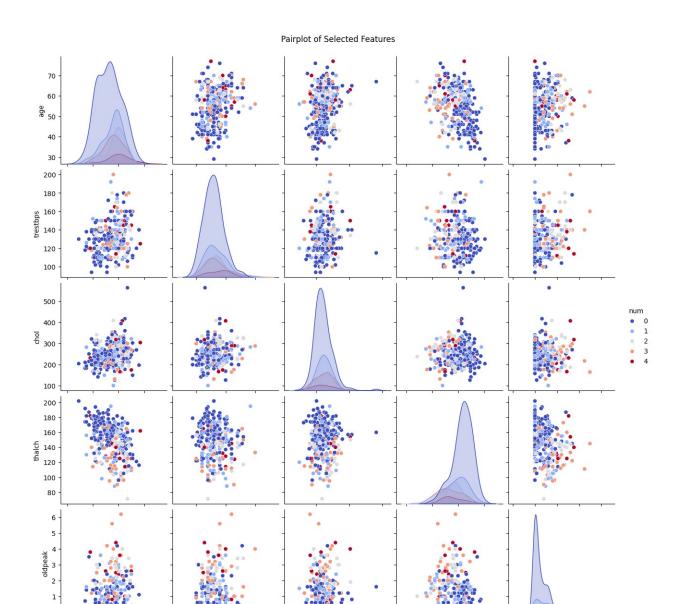


```
# Oldpeak (ST Depression) by Slope of the Peak Exercise ST Segment
sns.boxplot(data=df, x='slope_flat', y='oldpeak')
plt.title('ST Depression by Slope of the Peak Exercise ST Segment')
plt.xlabel('Flat Slope (1=Yes)')
plt.ylabel('ST Depression (oldpeak)')
plt.show()
```





```
selected_features = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak',
   'num']
sns.pairplot(df[selected_features], hue='num', palette='coolwarm')
plt.suptitle('Pairplot of Selected Features', y=1.02)
plt.show()
```



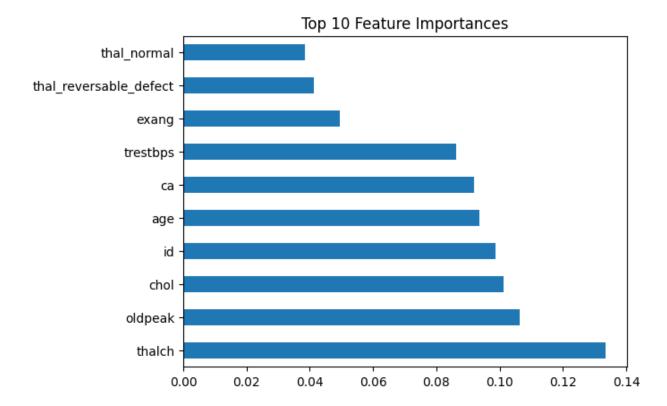
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# Define features and target
X = df.drop('num', axis=1)
y = df['num']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a Random Forest model
```

```
model = RandomForestClassifier()
model.fit(X train, y train)
# Predict and evaluate
v pred = model.predict(X test)
print(classification report(y test, y pred))
              precision
                           recall f1-score
                                               support
                   0.78
                             1.00
                                        0.88
           0
                                                    35
           1
                   0.00
                             0.00
                                        0.00
                                                    13
           2
                                                     5
                   0.17
                             0.20
                                        0.18
           3
                                                     4
                   0.00
                             0.00
                                        0.00
           4
                                                     3
                   0.00
                             0.00
                                        0.00
                                        0.60
                                                    60
    accuracy
                                        0.21
                                                    60
                   0.19
                             0.24
   macro avq
weighted avg
                   0.47
                             0.60
                                        0.53
                                                    60
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/
python3.12/site-packages/sklearn/metrics/_classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/python3.12/
site-packages/sklearn/metrics/ classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/Users/jimmytran/Downloads/project/pythonProject/.venv/lib/python3.12/
site-packages/sklearn/metrics/_classification.py:1517:
UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero division` parameter to
control this behavior.
  warn prf(average, modifier, f"{metric.capitalize()} is",
len(result))
# Feature importance
feature importance = pd.Series(model.feature importances ,
index=X.columns)
feature importance.nlargest(10).plot(kind='barh')
plt.title('Top 10 Feature Importances')
plt.show()
```



Summary of Analysis for Heart Disease Prediction

In this analysis, we aim to identify key factors that contribute to heart disease. Using the Heart Disease UCI dataset, we performed data cleaning, exploratory data analysis, and built a machine learning model to predict heart disease. The findings are summarized as follows:

- Age and sex are significant predictors of heart disease.
- Chest pain type, resting blood pressure, and cholesterol levels also play crucial roles.
- Patients with higher oldpeak values (ST depression induced by exercise) and abnormal exercise-induced ST segment slopes are more likely to have heart disease.
- The **Random Forest model** was used for prediction, and the top features influencing the model were identified.
- Model performance was evaluated using classification metrics and ROC curve analysis.

Key Findings:

- Age Distribution by Sex: Males tend to have a higher prevalence of heart disease compared to females.
- 2. **Chest Pain Type Distribution**: Different types of chest pain are associated with varying risks of heart disease.
- 3. **Heart Rate Analysis**: Higher maximum heart rates are observed in patients with certain types of chest pain.
- 4. **Blood Pressure and Cholesterol**: There is a significant relationship between higher resting blood pressure, cholesterol levels, and heart disease.

- 5. **Heart Disease by Sex**: The prevalence of heart disease varies between males and females.
- 6. **Oldpeak Analysis**: Higher ST depression values are indicative of higher heart disease risk.

Recommendations:

- 1. **Targeted Health Programs**: Develop targeted health programs focusing on high-risk groups identified by age, sex, and other significant features.
- 2. **Preventive Measures**: Encourage regular health screenings for early detection and management of high blood pressure and cholesterol levels.
- 3. **Public Awareness**: Increase public awareness about the significance of recognizing and managing chest pain and related symptoms early.