Assignment 1

# Predicting Heart Disease

Data Science and Machine Learning Bootcamp

Sajith Nuranga Satharsinghe

## Introduction

The UCI Heart Disease Data Set has 303 observations with 14 variables about patients with and without heart disease. The objective of this analysis is to predict a patient has heart disease based on their attributes. This report analyse this dataset to understand the factors that affect the likelihood of a patient having heart disease.

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | num |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63.0 | 1.0 | 1.0 | 145.0 | 233.0 | 1.0 | 2.0 | 150.0 | 0.0 | 2.3 | 3.0 | 0.0 | 6.0 | 0 |
| **1** | 67.0 | 1.0 | 4.0 | 160.0 | 286.0 | 0.0 | 2.0 | 108.0 | 1.0 | 1.5 | 2.0 | 3.0 | 3.0 | 2 |
| **2** | 67.0 | 1.0 | 4.0 | 120.0 | 229.0 | 0.0 | 2.0 | 129.0 | 1.0 | 2.6 | 2.0 | 2.0 | 7.0 | 1 |
| **3** | 37.0 | 1.0 | 3.0 | 130.0 | 250.0 | 0.0 | 0.0 | 187.0 | 0.0 | 3.5 | 3.0 | 0.0 | 3.0 | 0 |
| **4** | 41.0 | 0.0 | 2.0 | 130.0 | 204.0 | 0.0 | 2.0 | 172.0 | 0.0 | 1.4 | 1.0 | 0.0 | 3.0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **298** | 45.0 | 1.0 | 1.0 | 110.0 | 264.0 | 0.0 | 0.0 | 132.0 | 0.0 | 1.2 | 2.0 | 0.0 | 7.0 | 1 |
| **299** | 68.0 | 1.0 | 4.0 | 144.0 | 193.0 | 1.0 | 0.0 | 141.0 | 0.0 | 3.4 | 2.0 | 2.0 | 7.0 | 2 |
| **300** | 57.0 | 1.0 | 4.0 | 130.0 | 131.0 | 0.0 | 0.0 | 115.0 | 1.0 | 1.2 | 2.0 | 1.0 | 7.0 | 3 |
| **301** | 57.0 | 0.0 | 2.0 | 130.0 | 236.0 | 0.0 | 2.0 | 174.0 | 0.0 | 0.0 | 2.0 | 1.0 | 3.0 | 1 |
| **302** | 38.0 | 1.0 | 3.0 | 138.0 | 175.0 | 0.0 | 0.0 | 173.0 | 0.0 | 0.0 | 1.0 | ? | 3.0 | 0 |

303 rows × 14 columns

## Data Exploration

Data set has been checked for null values, missing values and cleaned and prepare for the data for analysis.

```
# Check for missing values
print(DF_1.isnull().sum())
```

```
DF_1.isnull().any()
```

```
DF_1 = DF_1.replace("?", np.nan)
DF_1 = DF_1.dropna()
print(DF_1)
```

Data set has been divided into Categorical and Numerical datasets for further Analysis.

```
CATEGORICAL_COLS = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'thal', 'ca']
NUMERICAL_COLS = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']


C_DF = DF_1[CATEGORICAL_COLS]
N_DF = DF_1[NUMERICAL_COLS]
```

Manupulate the num Field to differentiate sick and Healthy

Value 0 – Healthy

Values 1,2,3,4,- Sick

In [337]:
```
DF_1.loc[DF_1['num'] > 0, 'num'] = 1
DF_1
```

Out[337]:

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | num |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63.0 | 1.0 | 1.0 | 145.0 | 233.0 | 1.0 | 2.0 | 150.0 | 0.0 | 2.3 | 3.0 | 0.0 | 6.0 | 0 |
| **1** | 67.0 | 1.0 | 4.0 | 160.0 | 286.0 | 0.0 | 2.0 | 108.0 | 1.0 | 1.5 | 2.0 | 3.0 | 3.0 | 1 |
| **2** | 67.0 | 1.0 | 4.0 | 120.0 | 229.0 | 0.0 | 2.0 | 129.0 | 1.0 | 2.6 | 2.0 | 2.0 | 7.0 | 1 |
| **3** | 37.0 | 1.0 | 3.0 | 130.0 | 250.0 | 0.0 | 0.0 | 187.0 | 0.0 | 3.5 | 3.0 | 0.0 | 3.0 | 0 |
| **4** | 41.0 | 0.0 | 2.0 | 130.0 | 204.0 | 0.0 | 2.0 | 172.0 | 0.0 | 1.4 | 1.0 | 0.0 | 3.0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **297** | 57.0 | 0.0 | 4.0 | 140.0 | 241.0 | 0.0 | 0.0 | 123.0 | 1.0 | 0.2 | 2.0 | 0.0 | 7.0 | 1 |
| **298** | 45.0 | 1.0 | 1.0 | 110.0 | 264.0 | 0.0 | 0.0 | 132.0 | 0.0 | 1.2 | 2.0 | 0.0 | 7.0 | 1 |
| **299** | 68.0 | 1.0 | 4.0 | 144.0 | 193.0 | 1.0 | 0.0 | 141.0 | 0.0 | 3.4 | 2.0 | 2.0 | 7.0 | 1 |
| **300** | 57.0 | 1.0 | 4.0 | 130.0 | 131.0 | 0.0 | 0.0 | 115.0 | 1.0 | 1.2 | 2.0 | 1.0 | 7.0 | 1 |
| **301** | 57.0 | 0.0 | 2.0 | 130.0 | 236.0 | 0.0 | 2.0 | 174.0 | 0.0 | 0.0 | 2.0 | 1.0 | 3.0 | 1 |

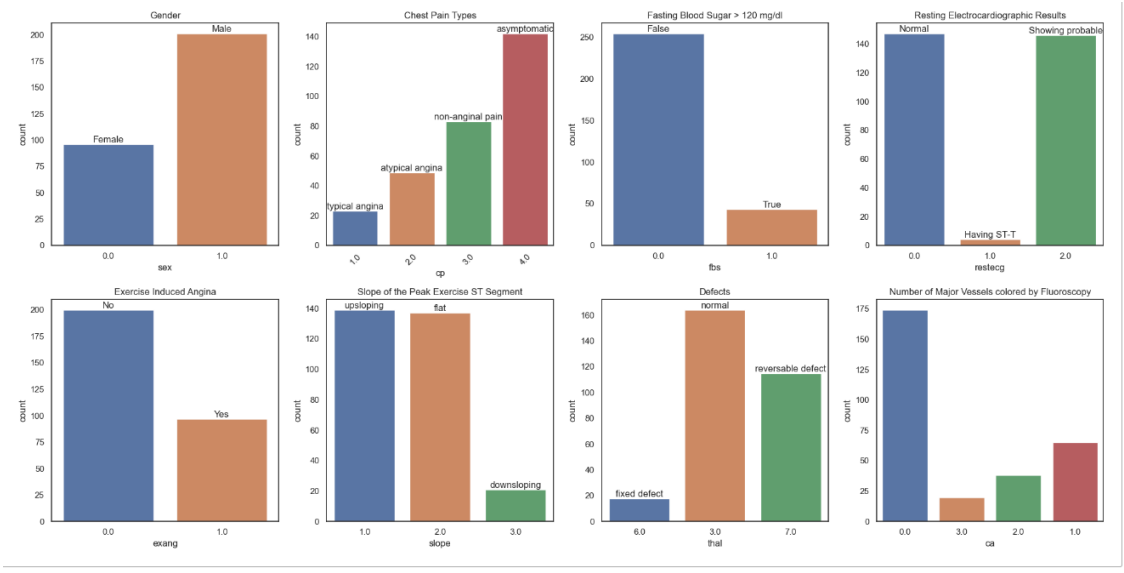
297 rows × 14 columns

## Data Visualization

Descriptive Statistics for Numerical Data

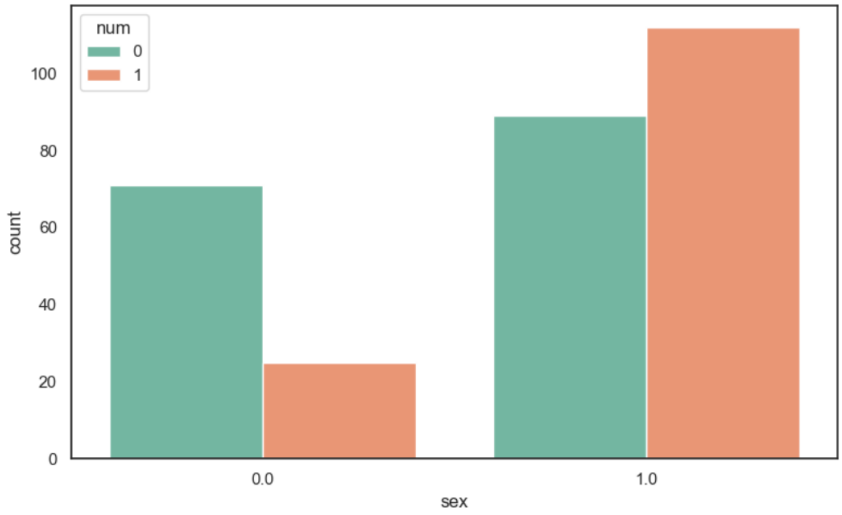
| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **age** | 297.000000 | 54.542088 | 9.049736 | 29.000000 | 48.000000 | 56.000000 | 61.000000 | 77.000000 |
| **trestbps** | 297.000000 | 131.693603 | 17.762806 | 94.000000 | 120.000000 | 130.000000 | 140.000000 | 200.000000 |
| **chol** | 297.000000 | 247.350168 | 51.997583 | 126.000000 | 211.000000 | 243.000000 | 276.000000 | 564.000000 |
| **thalach** | 297.000000 | 149.599327 | 22.941562 | 71.000000 | 133.000000 | 153.000000 | 166.000000 | 202.000000 |
| **oldpeak** | 297.000000 | 1.055556 | 1.166123 | 0.000000 | 0.000000 | 0.800000 | 1.600000 | 6.200000 |

Bar Chats are generated for Categorical Data – Gender, Chest Pain, Fasting Blood Sugar, Resting Electrocardiographic Results, Exercise included Angina, Slope, Thal and ca



Gender and Heart disease Distribution

In [373]:
```
fig, ax = plt.subplots(figsize=(8,5))
ax = sns.countplot(x='sex', hue='num', data=DF_1, palette='Set2')
plt.tight_layout()
```

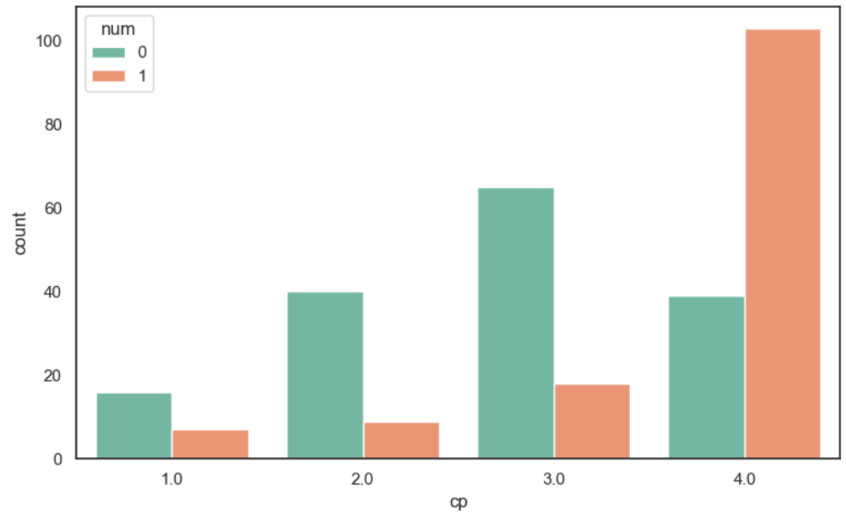



According to the Bar graph Male has higher vulnerability of heart disease than Female.

Chest Pain Type and Hart disease Distribution

sex

In [374]:
```
fig, ax = plt.subplots(figsize=(8,5))
ax = sns.countplot(x='cp', hue='num', data=DF_1, palette='Set2')
plt.tight_layout()
```

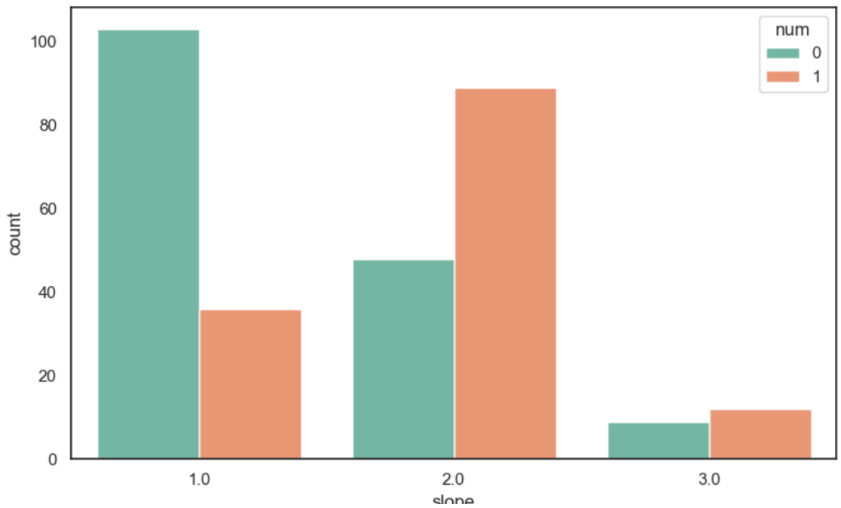



Chest pain Type Asymptomatic has a higher amount of heart disease.

Slope and Heart disease Distribution

cp

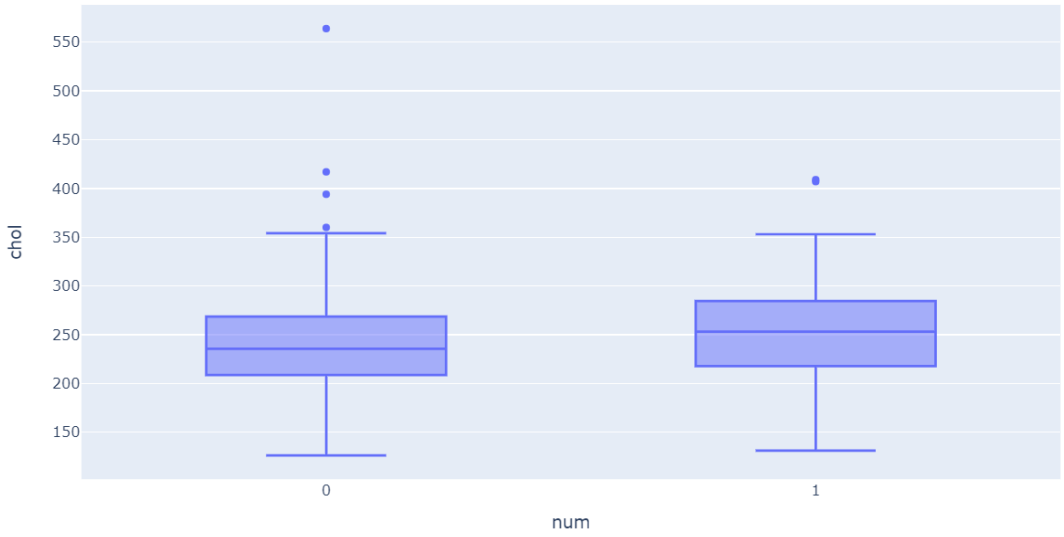
```
In [376]: fig, ax = plt.subplots(figsize=(8,5))
          ax = sns.countplot(x='slope', hue='num', data=DF_1, palette='Set2')
          plt.tight_layout()
```




Box plots for Numerical type data

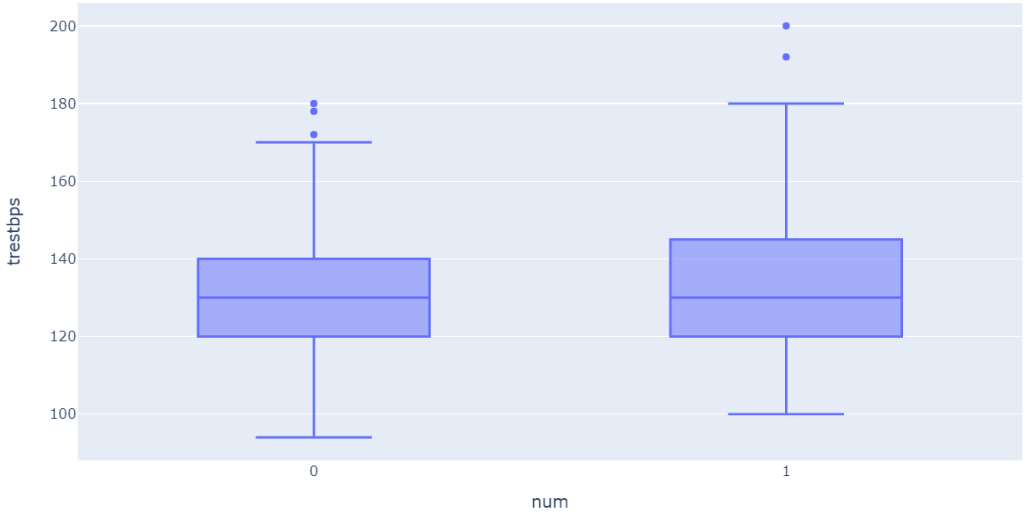
Age

```
fig = px.box(DF_1, x='num', y='age')
fig.show()
```



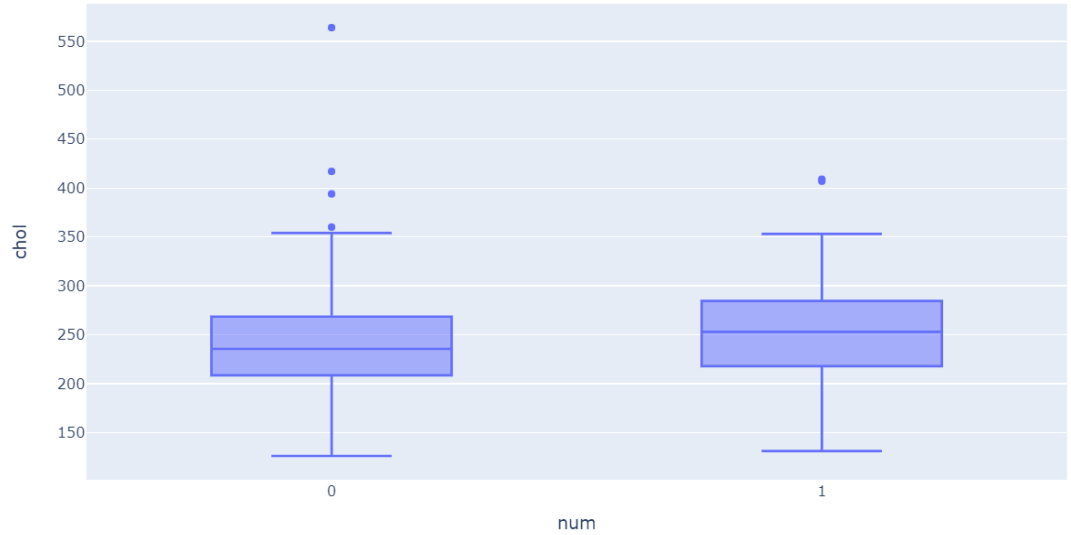
## Resting blood pressure

```
fig = px.box(DF_1, x='num', y='trestbps')
fig.show()
```



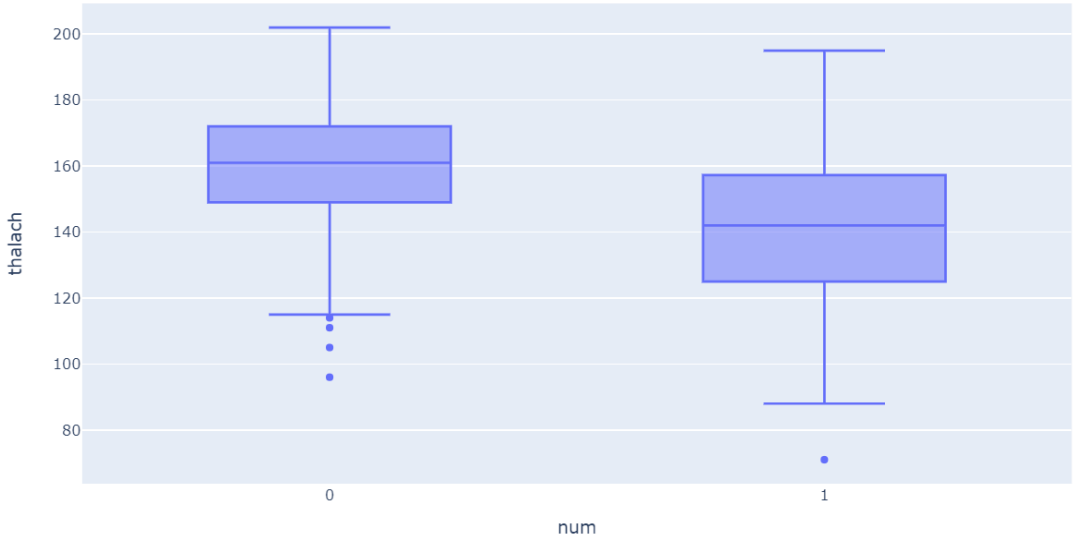

## Cholesterol

```
fig = px.box(DF_1, x='num', y='chol')
fig.show()
```



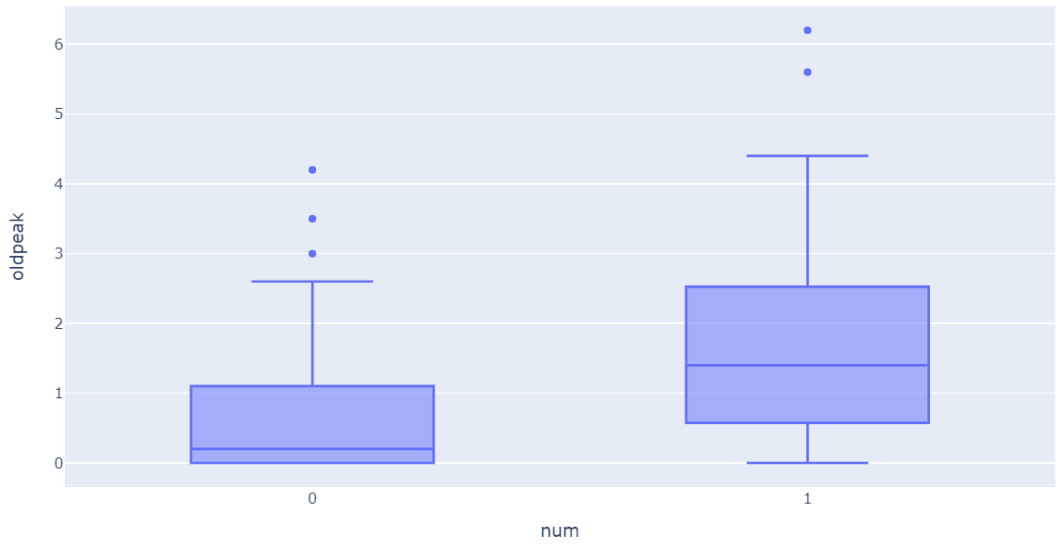
Maximum heart rate achieved.

```
fig = px.box(DF_1, x='num', y='thalach')
fig.show()
```




ST depression induced by exercise relative to rest

```
fig = px.box(DF_1, x='num', y='oldpeak')
fig.show()
```



Analyse the dot plots and remove outliers.

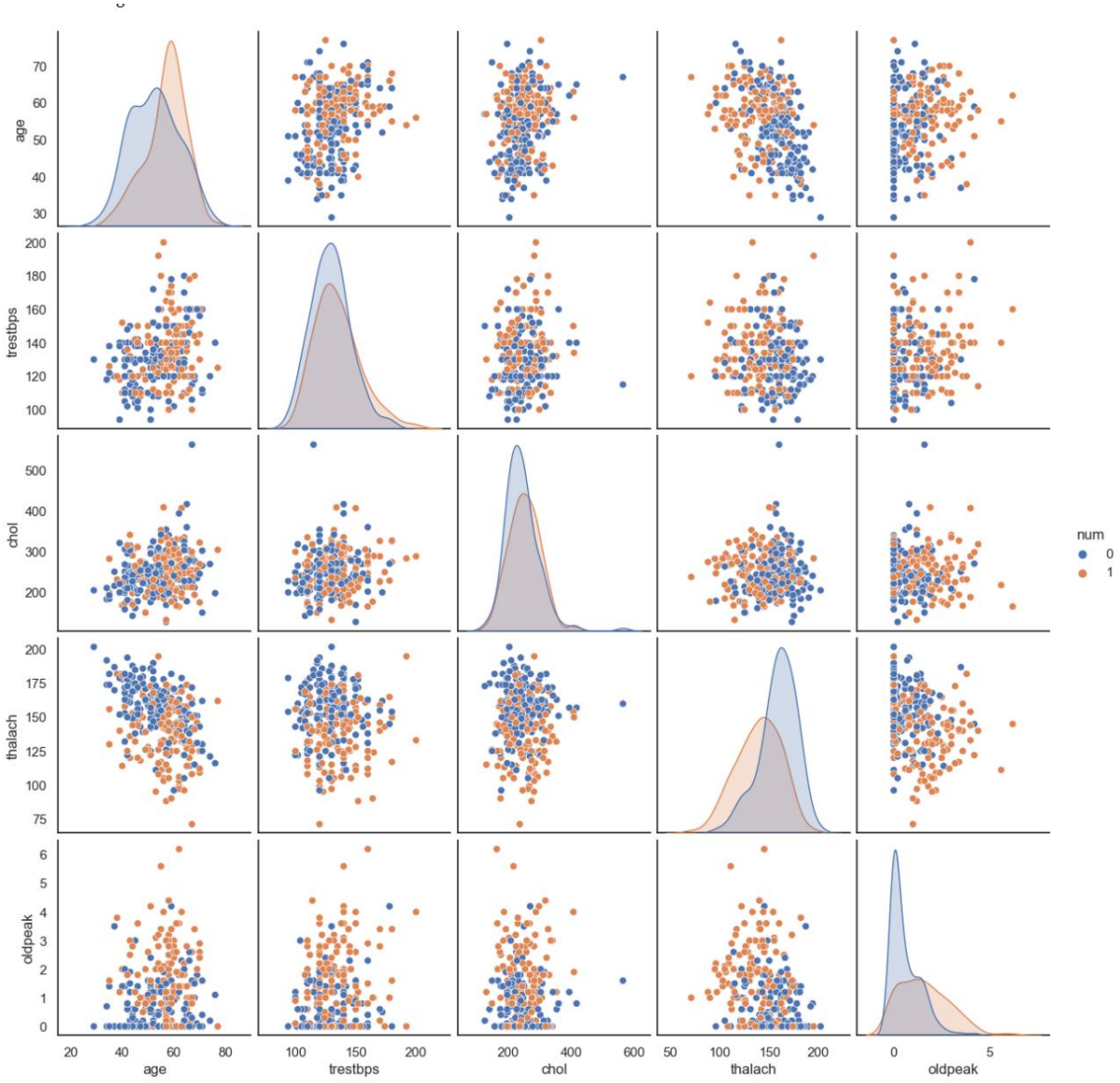
```
continous_features = ['age','trestbps','chol','thalach','oldpeak']
def outliers(df_out, drop = False):
    for each_feature in df_out.columns:
        feature_data = df_out[each_feature]
        Q1 = np.percentile(feature_data, 25.) # 25th percentile of the data of the given feature
        Q3 = np.percentile(feature_data, 75.) # 75th percentile of the data of the given feature
        IQR = Q3-Q1 #Interquartile Range
        outlier_step = IQR * 1.5 #That's we were talking about above
        outliers = feature_data[~((feature_data >= Q1 - outlier_step) & (feature_data <= Q3 + outlier_step))].index.tolist()
        if not drop:
            print('For the feature {}, No of Outliers is {}'.format(each_feature, len(outliers)))
        if drop:
            df.drop(outliers, inplace = True, errors = 'ignore')
            print('Outliers from {} feature removed'.format(each_feature))

outliers(DF_1[continous_features])
```

```
For the feature age, No of Outliers is 0
For the feature trestbps, No of Outliers is 9
For the feature chol, No of Outliers is 5
For the feature thalach, No of Outliers is 1
For the feature oldpeak, No of Outliers is 5
```

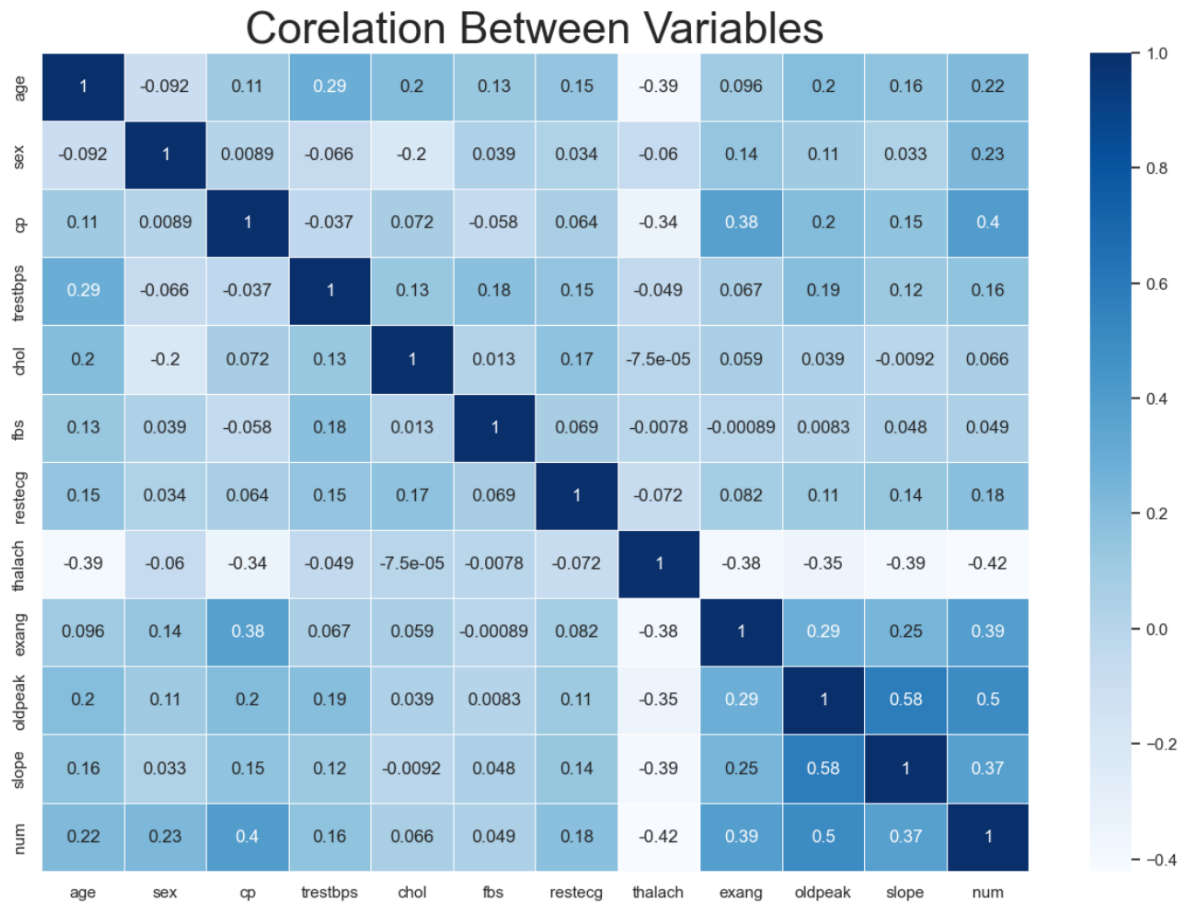
```
#Drop the Outliers

outliers(DF_1[continous_features], drop= True)
```

```
Outliers from age feature removed
Outliers from trestbps feature removed
Outliers from chol feature removed
Outliers from thalach feature removed
Outliers from oldpeak feature removed
```

Numerical Data Distribution




Oldpeak and Thalach have linear separation than other numerical variables




Based on above Correlation Heat Map, it clearly shows CP, Exang, Slop has positive Correlation with num variable while thalach has a good negative  Correlation. Fbs,chol and tresbps has only low Correlation

## Model Building and Evaluation

Logistic Regression Model has used for evaluation

```
data = pd.get_dummies(data_tmp, drop_first=False)
data.columns
```

```
Index(['age', 'sex', 'chest_pain_type', 'resting_blood_pressure',
       'cholesterol', 'fasting_blood_sugar', 'max_heart_rate_achieved',
       'exercise_induced_angina', 'st_depression', 'st_slope_type', 'target',
       'num_major_vessels_0.0', 'num_major_vessels_1.0',
       'num_major_vessels_2.0', 'num_major_vessels_3.0',
       'thalassemia_type_3.0', 'thalassemia_type_6.0', 'thalassemia_type_7.0'],
      dtype='object')
```

```
y = data['target']
X = data.drop('target', axis = 1)


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
print(f'Shape of X_train: {X_train.shape}')
print(f'Shape of y_train: {y_train.shape}')
print(f'Shape of X_test: {X_test.shape}')
print(f'Shape of y_test: {y_test.shape}')
```

```
Shape of X_train: (237, 17)
Shape of y_train: (237,)
Shape of X_test: (60, 17)
Shape of y_test: (60,)
```

```
X_train=(X_train-np.min(X_train))/(np.max(X_train)-np.min(X_train)).values
X_test=(X_test-np.min(X_test))/(np.max(X_test)-np.min(X_test)).values
```

Logistic model accuracy shows as 0.82

```
print('The Accuracy Score is: ', accuracy_score(y_test,y_pred))
```

```
The Accuracy Score is:  0.8166666666666667
```