Exploratory Data Analysis Project

Heart Attack Dataset EDA

Jean Paul Tuyikunde

May 20, 2021

I. Data Description

- Age : Age of the patient
- Sex : Sex of the patient
- exang: exercise induced angina (1 = yes; 0 = no)
- ca: number of major vessels (0-3)
- cp : Chest Pain type chest pain type

```
Value 1: typical angina
Value 2: atypical angina
Value 3: non-anginal pain
Value 4: asymptomatic
```

- trtbps : resting blood pressure (in mm Hg)
- chol : cholestoral in mg/dl fetched via BMI sensor
- fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- rest_ecg : resting electrocardiographic results

```
Value 0: normal
Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or        d
epression of > 0.05 mV)
Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
```

- thalach : maximum heart rate achieved
- target : 0= less chance of heart attack 1= more chance of heart attack.

II. Data Exploration

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [2]:
```
# Load data
data = pd.read_csv('heart.csv')
data.head()
```

Out[2]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [3]:
```
data.shape
```

Out[3]:
```
(303, 14)
```

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trtbps    303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalachh  303 non-null    int64
 8   exng      303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slp       303 non-null    int64
 11  caa       303 non-null    int64
 12  thall     303 non-null    int64
 13  output    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

In [5]:
```
data.dtypes.value_counts().to_frame()
```

Out[5]:

	0
int64	13
float64	1

In [6]:
```
# Checking missing values
data.isnull().sum().sort_values().to_frame()
```

Out[6]:

	0
age	0
sex	0
cp	0
trtbps	0
chol	0
fbs	0
restecg	0
thalachh	0
exng	0
oldpeak	0
slp	0
caa	0
thall	0
output	0

In [7]:
```
data.describe()
```

Out[7]:

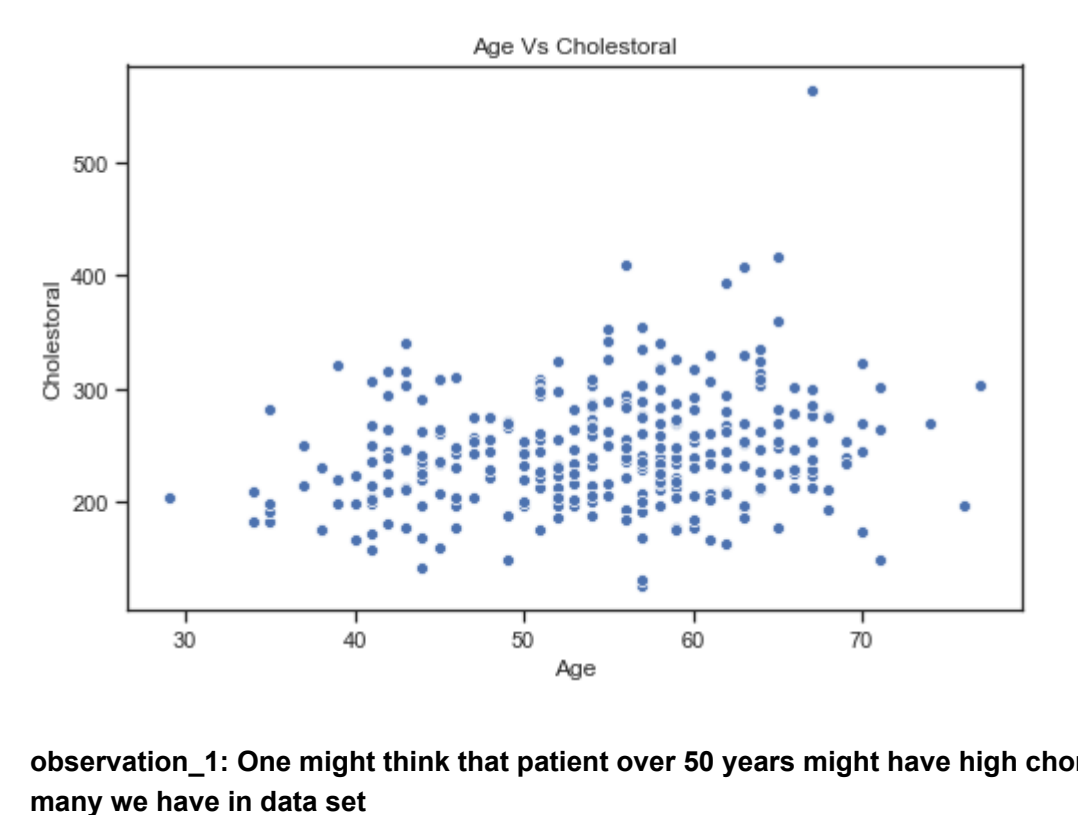
	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000(
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.3993
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.6162
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000(
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000(
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000(
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000(
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000(

III. Exploratory Data Analysis

Make a scatter plot of Age of patient (`age`) Vs Cholestoral (`chol`)

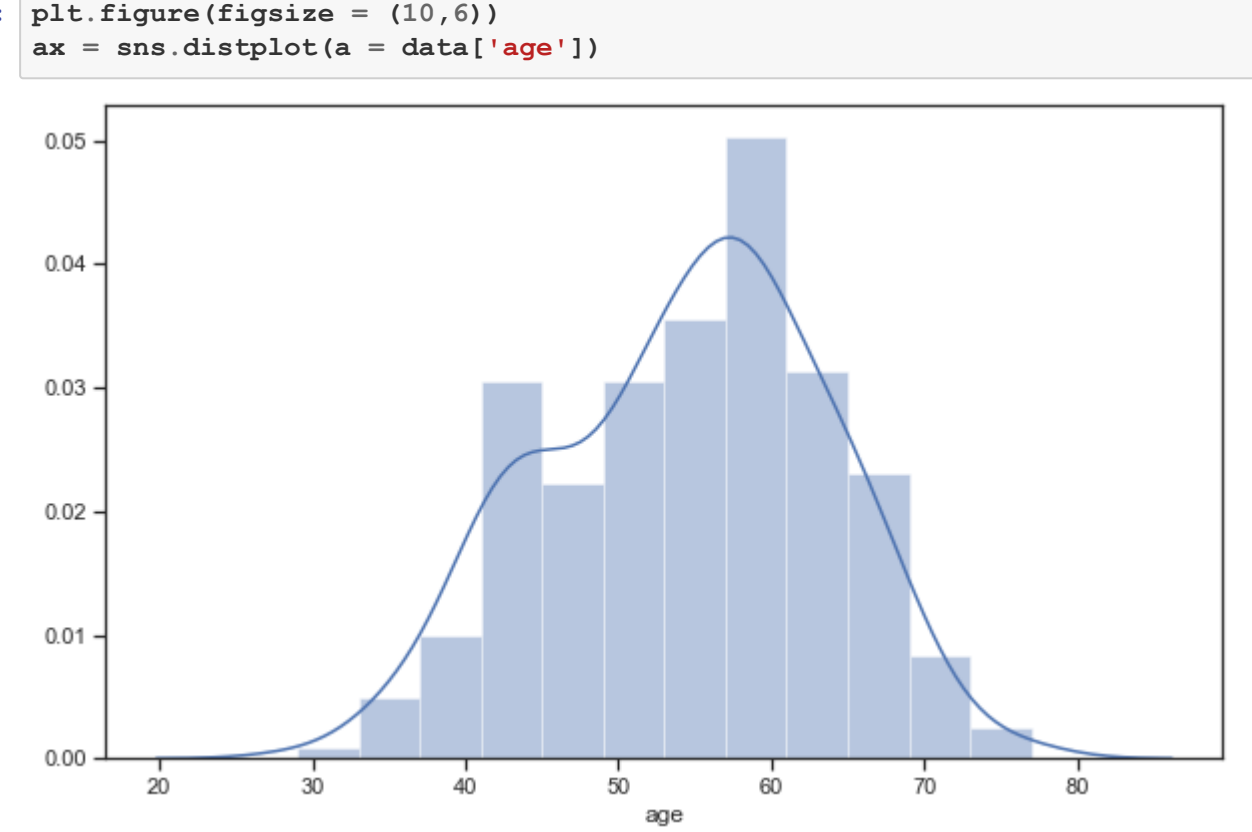
In [8]:
```
#Scatter plot
f = plt.figure(figsize=(8,5))
ax = sns.scatterplot(x="age", y="chol", data= data)
ax.set(xlabel='Age', ylabel='Cholestoral', Title = 'Age Vs Cholestoral')
```

Out[8]:
```
[Text(0, 0.5, 'Cholestoral'),
 Text(0.5, 0, 'Age'),
 Text(0.5, 1.0, 'Age Vs Cholestoral')]
```



observation_1: One might think that patient over 50 years might have high chorestoral but this led to onother question to see how many we have in data set

In [9]:
```
plt.figure(figsize = (10,6))
ax = sns.distplot(a = data['age'])
```



In [10]:
```
print('Less than 50 years: ',len([age for age in data.age if age <50]))
print('Greater than 50 years: ',len([age for age in data.age if age >=50]))
```

```
Less than 50 years:  88
Greater than 50 years:  215
```

observation_2: patients of ages greater than *50 years* seem to have a large number in our data set

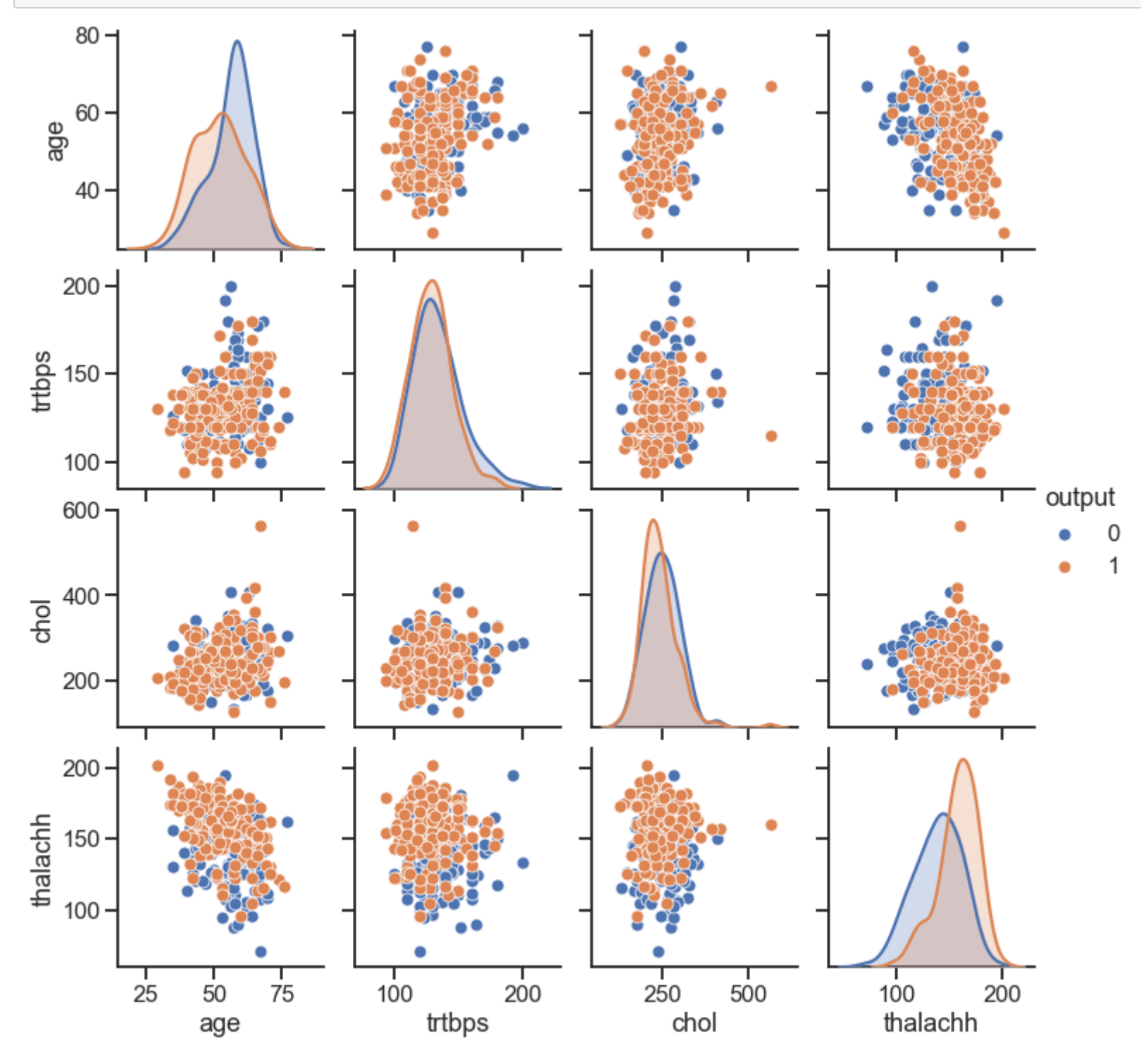
Make a plot of Sex of patient (`sex`)

In [11]:
```
#Barplot
plt.figure(figsize=(8, 6))
plt.title("Female Vs Male")
ax = sns.countplot(x = data['sex'])
ax.set(ylabel = 'Number of Patients')
```

Out[11]:
```
[Text(0, 0.5, 'Number of Patients')]
```

observation_3: There is a huge number of male patients than female

In [12]:
```
cols =['age','trtbps','chol','thalachh','output']
sns.set_context('talk')
sns.pairplot(data[cols], hue = 'output');
```



The mean of each measurement. The median of each measurements.

In [13]:
```
data.groupby('output').agg(['mean', 'median'])
```

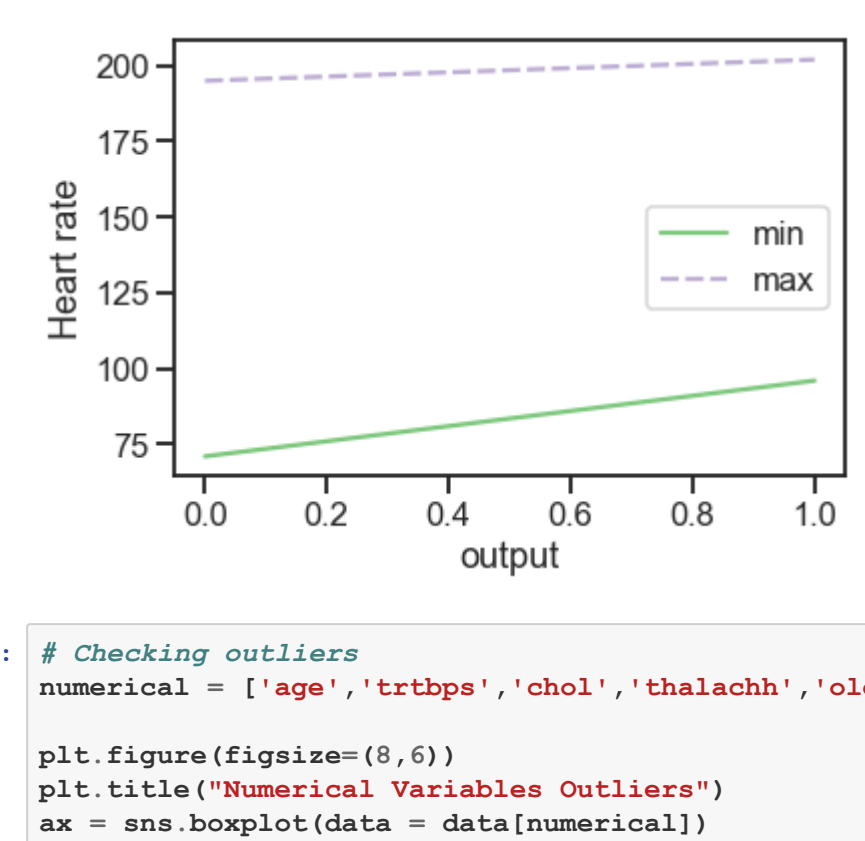
Out[13]:

	age		sex		cp		trtbps		chol		...	exng		oldpea
	mean	median	mean	median	mean	median	mean	median	mean	median	...	mean	median	mean
output														
0	56.601449	58	0.826087	1	0.478261	0	134.398551	130	251.086957	249	...	0.550725	1	1.5855
1	52.496970	52	0.563636	1	1.375758	2	129.303030	130	242.230303	234	...	0.139394	0	0.5830

2 rows × 26 columns

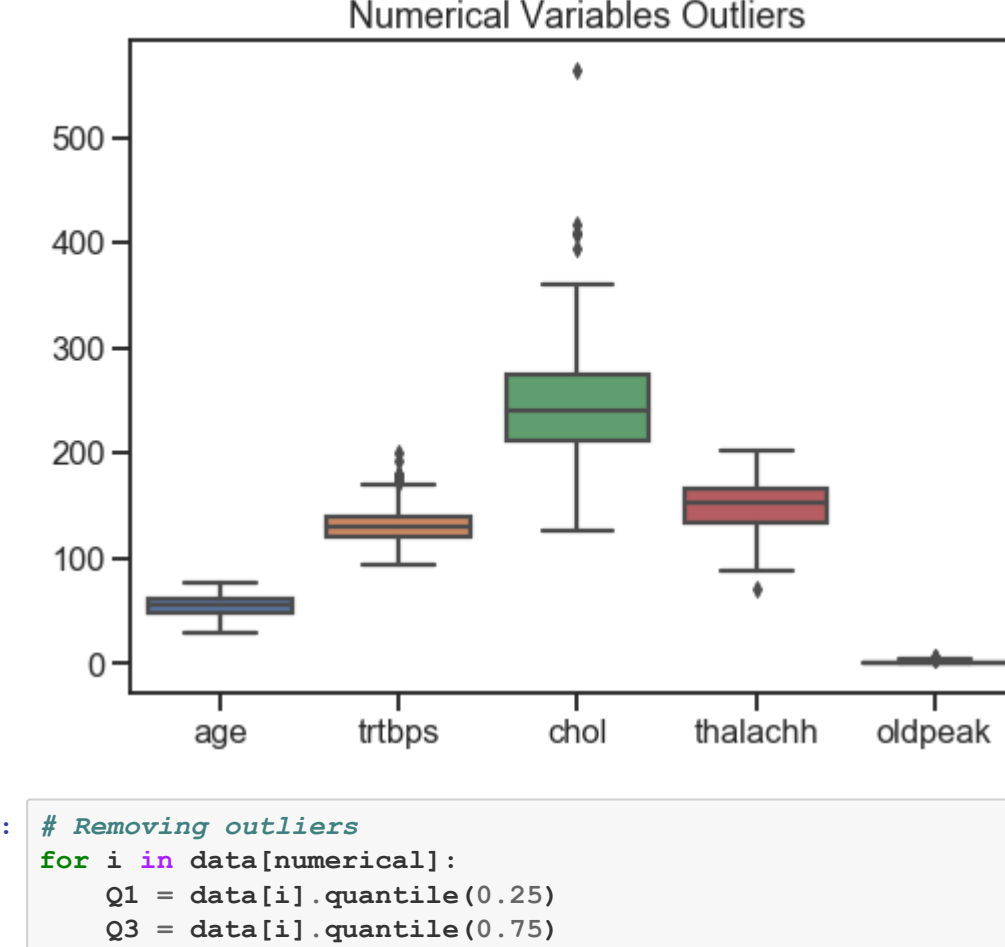
In [14]:
```
df = data.groupby('output')['thalachh'].agg(['min','max'])
ax2 = sns.lineplot(data=df, palette="Accent")
ax2.set(xlabel='output', ylabel='Heart rate')
```

Out[14]:
```
[Text(0, 0.5, 'Heart rate'), Text(0.5, 0, 'output')]
```



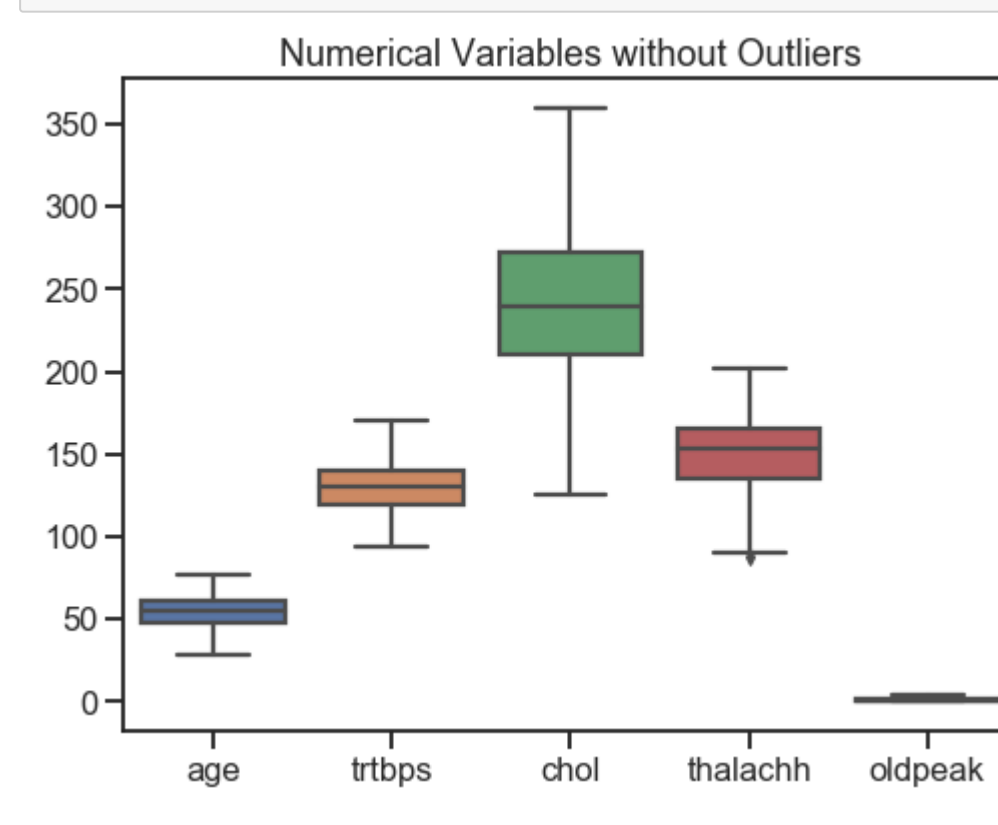
In [15]:
```
# Checking outliers
numerical = ['age','trtbps','chol','thalachh','oldpeak']

plt.figure(figsize=(8,6))
plt.title("Numerical Variables Outliers")
ax = sns.boxplot(data = data[numerical])
```



In [16]:
```
# Removing outliers
for i in data[numerical]:
    Q1 = data[i].quantile(0.25)
    Q3 = data[i].quantile(0.75)
    interQ = Q3-Q1
    LowerT = Q1 - 1.5 * interQ
    UpperT = Q3 + 1.5 * interQ
    median = np.median(data[i])
    for j in data[i]:
        if j > UpperT or j < LowerT:
            data[i] = data[i].replace(j, median)
```

In [17]:
```
plt.figure(figsize=(8, 6))
plt.title("Numerical Variables without Outliers")
ax = sns.boxplot(data = data[numerical])
```



IV. Feature engineering

In [18]:
```
# separating features from target
X = data.drop(['output'],axis=1)
Y = data[['output']]
```

In [19]:
```
#Selecting categorical columns using cardinality
low_cardinality_cols = [colName for colName in X.columns if X[colName].nunique() < 6 or X[colName].dty
pe == 'object']
low_cardinality_cols
numerical_cols = numerical
mycols = low_cardinality_cols + numerical_cols
```

One-hot encoding

In [20]:
```
X = pd.get_dummies(X, columns=low_cardinality_cols, drop_first=True)
df.describe().T
```

Out[20]:

	count	mean	std	min	25%	50%	75%	max
min	2.0	83.5	17.677670	71.0	77.25	83.5	89.75	96.0
max	2.0	198.5	4.949747	195.0	196.75	198.5	200.25	202.0

In [21]:
```
print("columns for dummy: {}\ncolumns for non-dummy:{}".format(len(df.columns), len(X.columns)))
```

```
columns for dummy: 2
columns for non-dummy:22
```

Polynomial Features

In [22]:
```
from sklearn.preprocessing import PolynomialFeatures
pf = PolynomialFeatures(degree = 2)
```

In [23]:
```
features = ['chol', 'trtbps'] #cholestoral and Blood pressure
poly_features = pf.fit(X[features])
```

In [24]:
```
featured = pd.DataFrame(poly_features.transform(X[features]),
                        columns = poly_features.get_feature_names(input_features = features))
new_cols = []
for i in featured.columns[3:]:
    new_cols.append(i)
```

In [25]:
```
X_featured = X.join(featured[new_cols])
X_featured
```

Out[25]:

	age	trtbps	chol	thalachh	oldpeak	sex_1	cp_1	cp_2	cp_3	fbs_1	...	caa_1	caa_2	caa_3	caa_4	thall_1	thall_2	thall_3	
0	63	145	233	150	2.3	1	0	0	1	1	...	0	0	0	0	1	0	0	5
1	37	130	250	187	3.5	1	0	1	0	0	...	0	0	0	0	0	1	0	6
2	41	130	204	172	1.4	0	1	0	0	0	...	0	0	0	0	0	1	0	4
3	56	120	236	178	0.8	1	1	0	0	0	...	0	0	0	0	0	1	0	5
4	57	120	354	163	0.6	0	0	0	0	0	...	0	0	0	0	0	1	0	12
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
298	57	140	241	123	0.2	0	0	0	0	0	...	0	0	0	0	0	0	1	5
299	45	110	264	132	1.2	1	0	0	1	0	...	0	0	0	0	0	0	1	6
300	68	144	193	141	3.4	1	0	0	0	1	...	0	1	0	0	0	0	1	3
301	57	130	131	115	1.2	1	0	0	0	0	...	1	0	0	0	0	0	1	1
302	57	130	236	174	0.0	0	1	0	0	0	...	1	0	0	0	0	1	0	5

303 rows × 25 columns

V. Hypothesis Testing

Hypothesis:

- The increase of Cholesterol will increase chance of having Heart attack
- High blood pressure will increase the Heart attack
- High heart rate will increase the chance of heart attack

Signficance testing

H0: Cholesterol will increase the Heart Attack.

HA: Cholesterol will not increase the chance of having the heart attack.

In [26]:
```
from sklearn.linear_model import LinearRegression as LinearReg
from sklearn.feature_selection import f_regression as fReg
import statsmodels.api as sm
```

In [27]:
```
x = X_featured['chol'].to_numpy().reshape(-1, 1)
y = Y.to_numpy().reshape(-1, )

reg = LinearReg().fit(x, y)

r_sq = reg.score(x, y)
p_val = fReg(x, y)[1][0]

print("The R-squared is {} \nThe p_value is {}".format(r_sq, p_val))
```

```
The R-squared is 0.013435157198994687
The p_value is 0.04378991393694867
```

Result

The p value is 0.04378991393694867 and is smaller than 0.05. Therefore H0 is rejected, that the higher Cholestoral will lead to chance of having the heart attack.

Next

The data used has no missing values. It contains large number of males than females which could lead to biased results. Eventhough the data is good but it contains less amount of observation.