IBM Unsupervised Learning

Lucian Popa

February 21st 2022

Outline

- Introduction;
- Dataset description;
- Methodology;
- K-Means;
- Conclusion.

Introduction

- The heart is an amazing organ. It continuously pumps oxygen and nutrient-rich blood throughout your body to sustain life. This fist-sized powerhouse beats (expands and contracts) 100,000 times per day pumping 23,000 liters (5,000 gallons) of blood every day. To work properly, the heart (just like any other muscle) needs a good blood supply.
- WHO announced that cardiovascular diseases is the top one killer over the world. There are seventeen million people died from it every year, especially heart disease. Prevention is better than cure. If we can evaluate the risk of every patient who probably has heart disease, that is, not only patients but also everyone can do something earlier to keep illness away.
- A heart attack (also known as myocardial infarction; MI) is defined as the sudden blockage of blood flow to a portion of the heart. Some of the heart muscle begins to die during a heart attack, and without early medical treatment, the loss of the muscle could be permanent.
- Conditions such as high blood pressure, high blood cholesterol, obesity, and diabetes can raise the risk of a heart attack. Behaviors such as an unhealthy diet, low levels of physical activity, smoking, and excessive alcohol consumption can contribute to the conditions that can cause heart attacks. Some factors, such as age and family history of heart disease, cannot be modified but are associated with a higher risk of a heart attack.

The goal:

- The aim of this project is to apply unsupervised learning techniques to find whether an individual will develop a heart attack risk or not. More specifically, after some feature engineering and exploratory data analysis, the k-means and agglomerative clusteing algorithms will be explored.

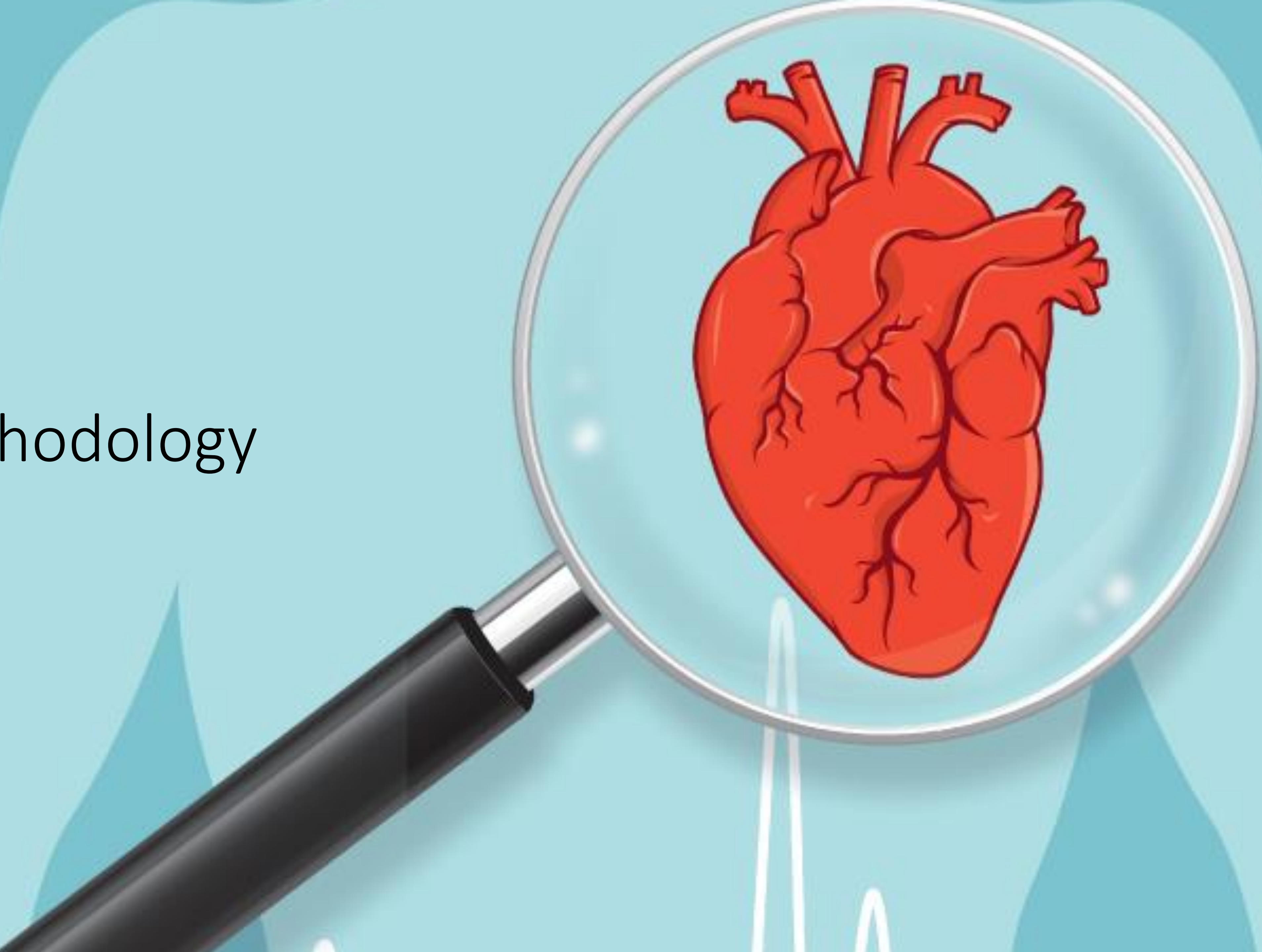
Dataset description

For the exploration of the risk a person has to develop a heart attack, the Heart Attack Analysis & Prediction Dataset from *kaggle.com* was utilized.

There are thirteen features and one target as below:

- age: The person's age in years
- sex: The person's sex (1 = male, 0 = female)
- cp: The chest pain experienced (Value 1: typical angina, Value 2: atypical angina, Value 3: non-anginal pain, Value 4: asymptomatic)
- trestbps: The person's resting blood pressure (mm Hg on admission to the hospital)
- chol: The person's cholesterol measurement in mg/dl
- fbs: The person's fasting blood sugar (> 120 mg/dl, 1 = true; 0 = false)
- restecg: Resting electrocardiographic measurement (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)
- thalach: The person's maximum heart rate achieved
- exang: Exercise induced angina (1 = yes; 0 = no)
- oldpeak: ST depression induced by exercise relative to rest
- slope: the slope of the peak exercise ST segment (Value 1: upsloping, Value 2: flat, Value 3: downsloping)
- ca: The number of major vessels (0-3)
- thal: A blood disorder called thalassemia (3 = normal; 6 = fixed defect; 7 = reversable defect)
- target: Heart disease (0 = no, 1 = yes)

Methodology



Data cleaning and preprocessing

Before we continue we need to continue the exploration of the data, remove duplicates, if any and remove outliers.

In [15]:
```
# check for duplicates and remove them
duplicate = df[df.duplicated()]

print("Duplicate Rows :")

duplicate
```

```
Duplicate Rows :
```

Out[15]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
164	38	1	2	138	175	0	1	173	0	0.0	2	4	2	1

In [16]:
```
# drop duplicates
clean_df = df.drop_duplicates()
```

Data cleaning and preprocessing

Transform numerical columns into categorical features and normalize the target variable.

In [20]:
```
# transform numerical columns into categorical
categorical = ['sex', 'exang', 'ca', 'cp', 'thal', 'fbs', 'restecg', 'slope', 'target']

for cat in categorical:
    clean_df[cat] = clean_df[cat].astype('category')
```

In [21]:
```
clean_df.info()
```

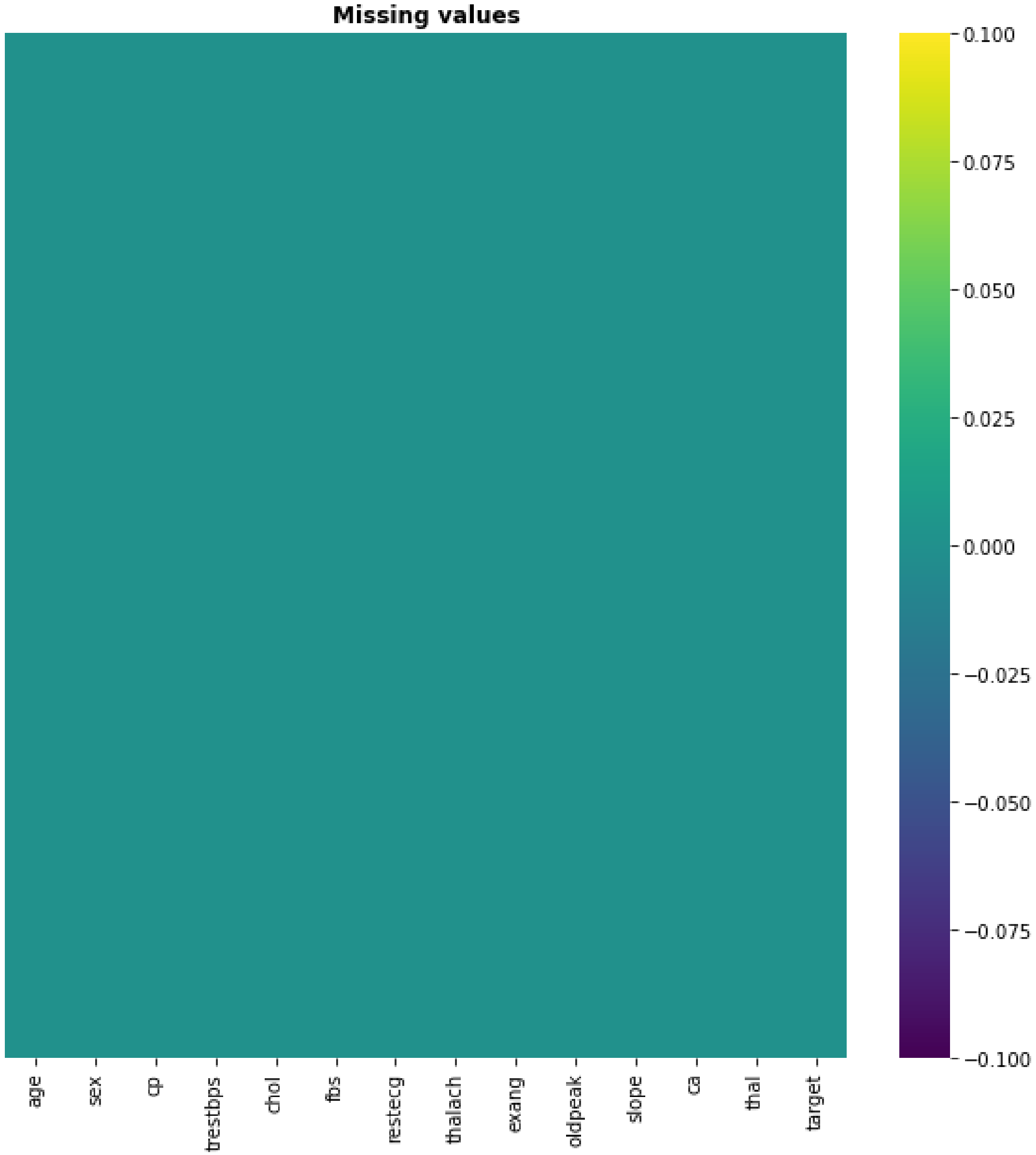
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 302 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype   
---  ------    --------------  -----   
 0   age       302 non-null    int64   
 1   sex       302 non-null    category
 2   cp        302 non-null    category
 3   trestbps  302 non-null    int64   
 4   chol      302 non-null    int64   
 5   fbs       302 non-null    category
 6   restecg   302 non-null    category
 7   thalach   302 non-null    int64   
 8   exang     302 non-null    category
 9   oldpeak   302 non-null    float64 
 10  slope     302 non-null    category
 11  ca        302 non-null    category
 12  thal      302 non-null    category
 13  target    302 non-null    category
dtypes: category(9), float64(1), int64(4)
memory usage: 18.2 KB
```

In [22]:
```
# set the target column and normalize data
clean_df['target'].value_counts(normalize=True)
```

Out[22]:
```
1    0.543046
0    0.456954
Name: target, dtype: float64
```

EDA and insights

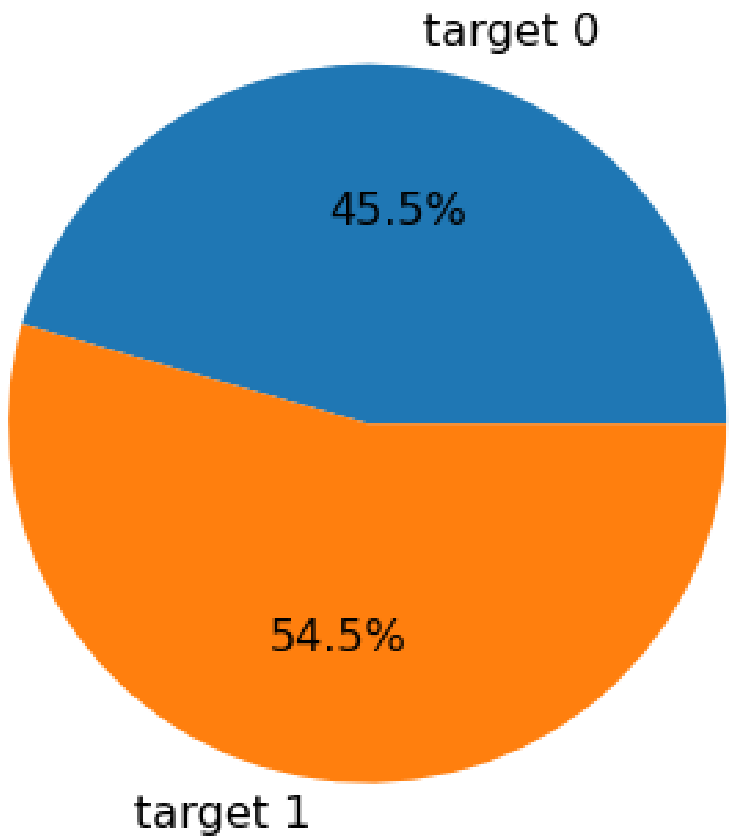
- Check for missing values



EDA and insights

Percentage of the people with heart disease

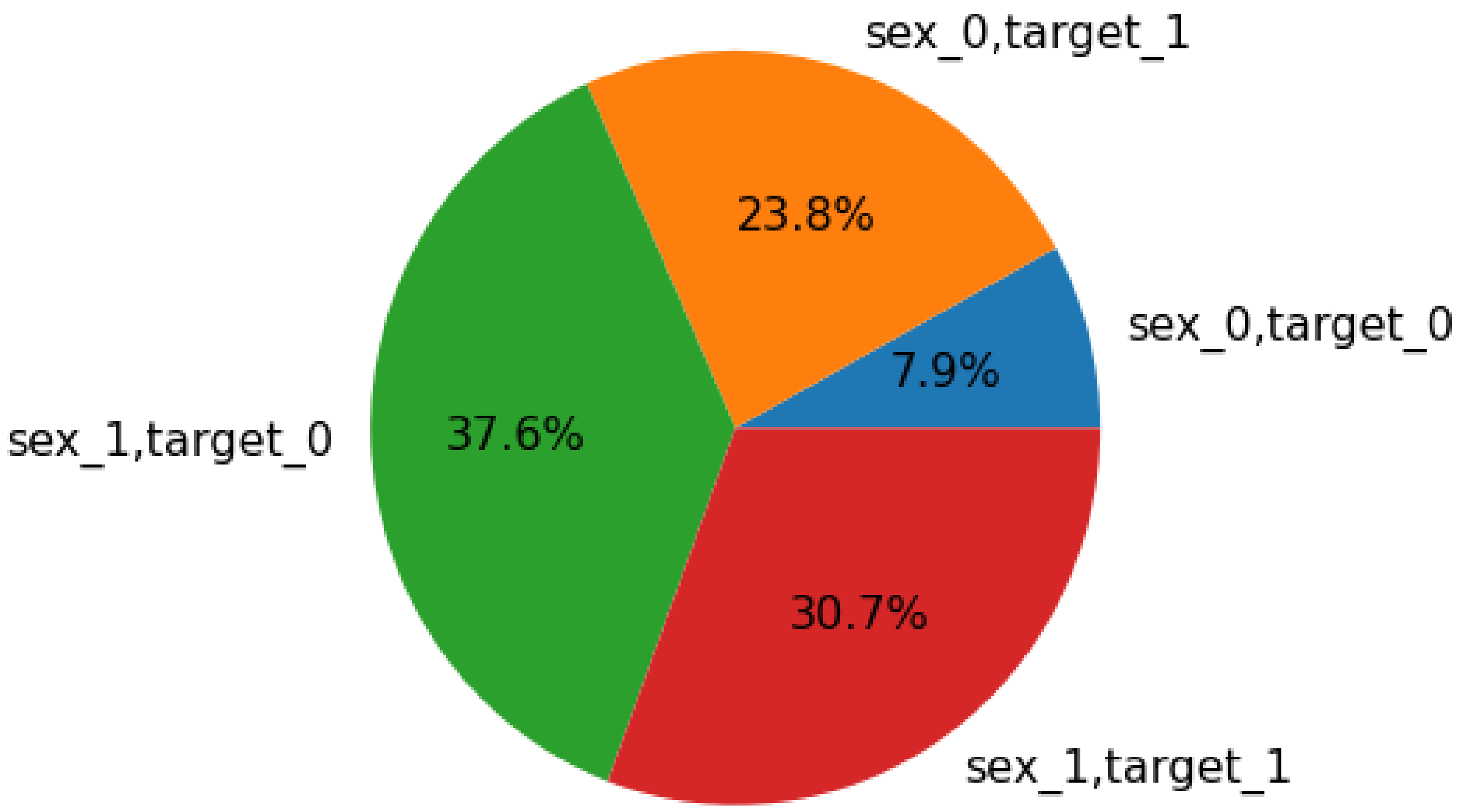
54.5 % of the people in this dataset were diagnosed with heart disease



EDA and insights

- Percentages of diagnosed by sex

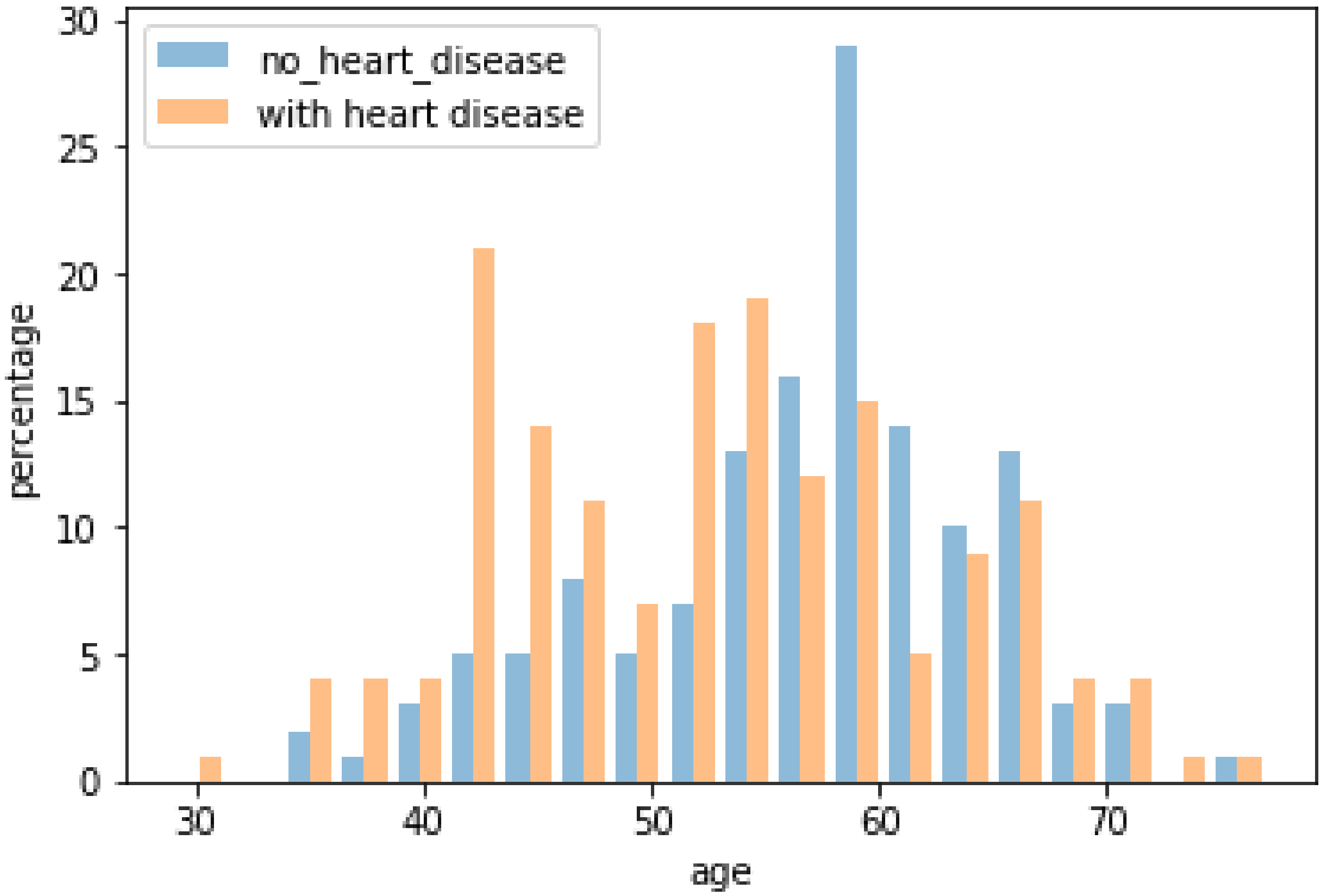
The percentage of male diagnosed with heart disease is higher, 30.7% are male and 23.8% are female.



EDA and insights

- Distribution by age of diagnosed people

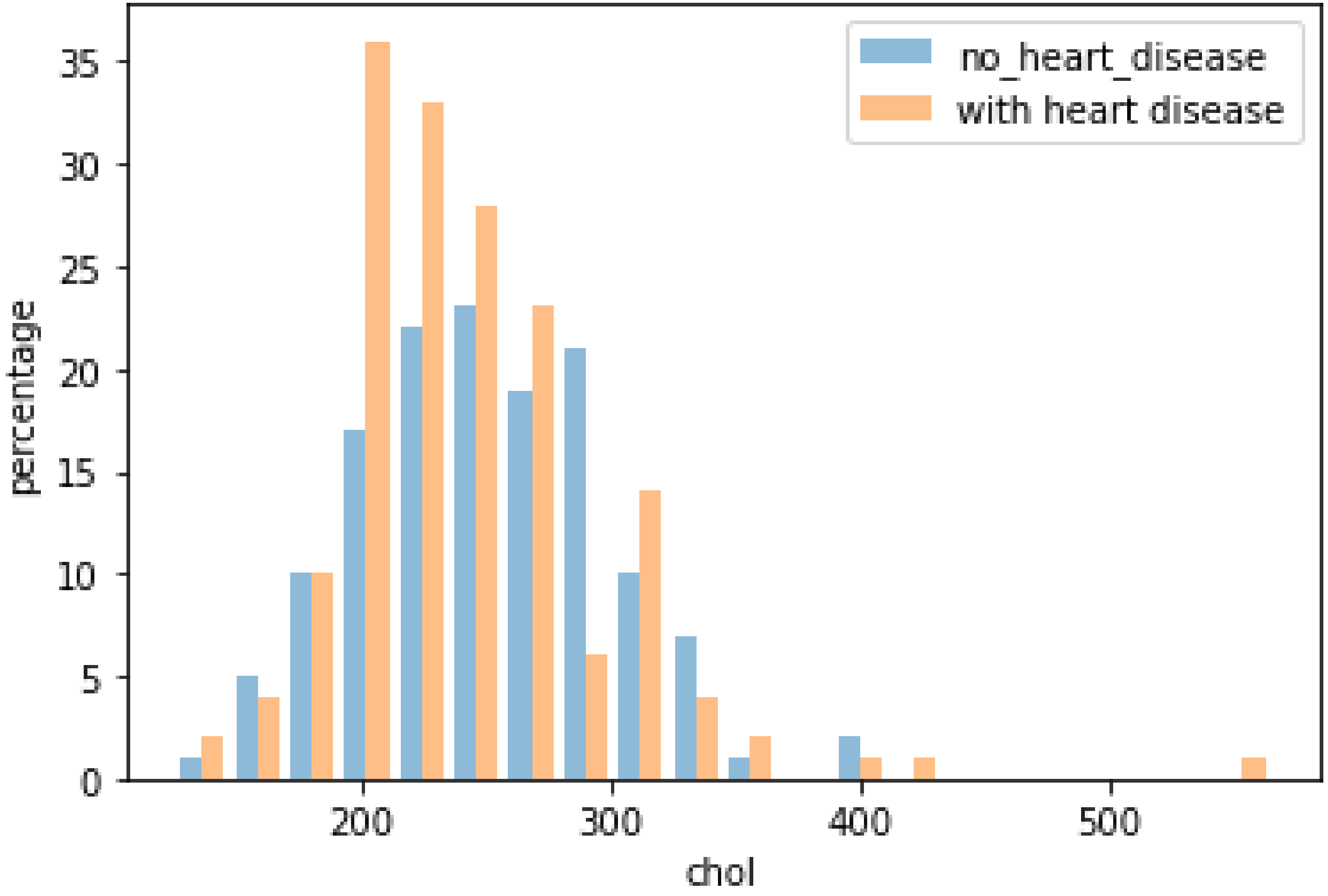
According to this plot, people over 40 years old have a higher change of being diagnosed with heart disease.



EDA and insights

- Distribution of cholesterol

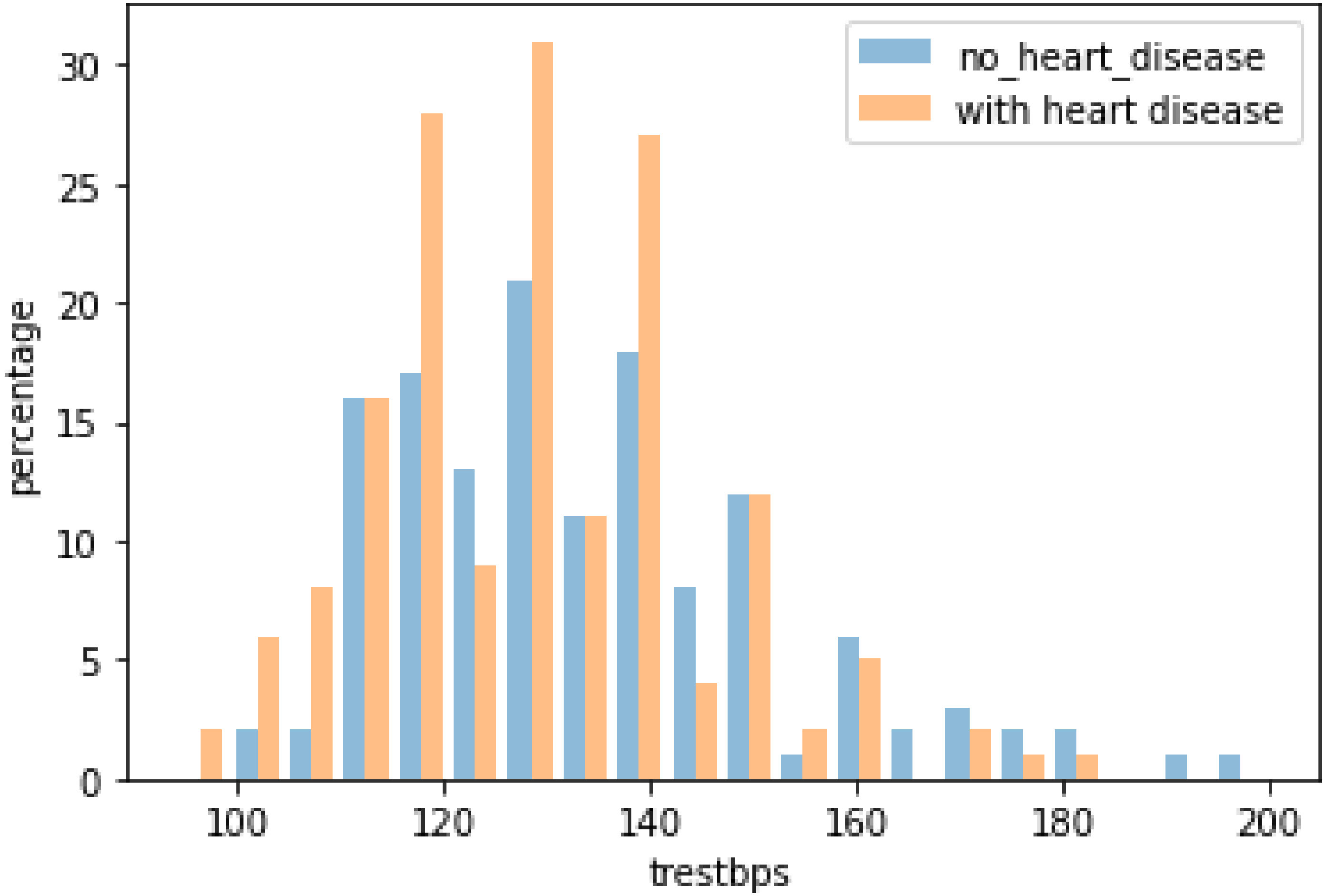
According to the research, the normal value of cholesterol should be lower than 200mg/dl. The number of people with heart disease spikes up when cholesterol goes above 200mg/dl.



EDA and insights

- Distribution of blood pressure while resting

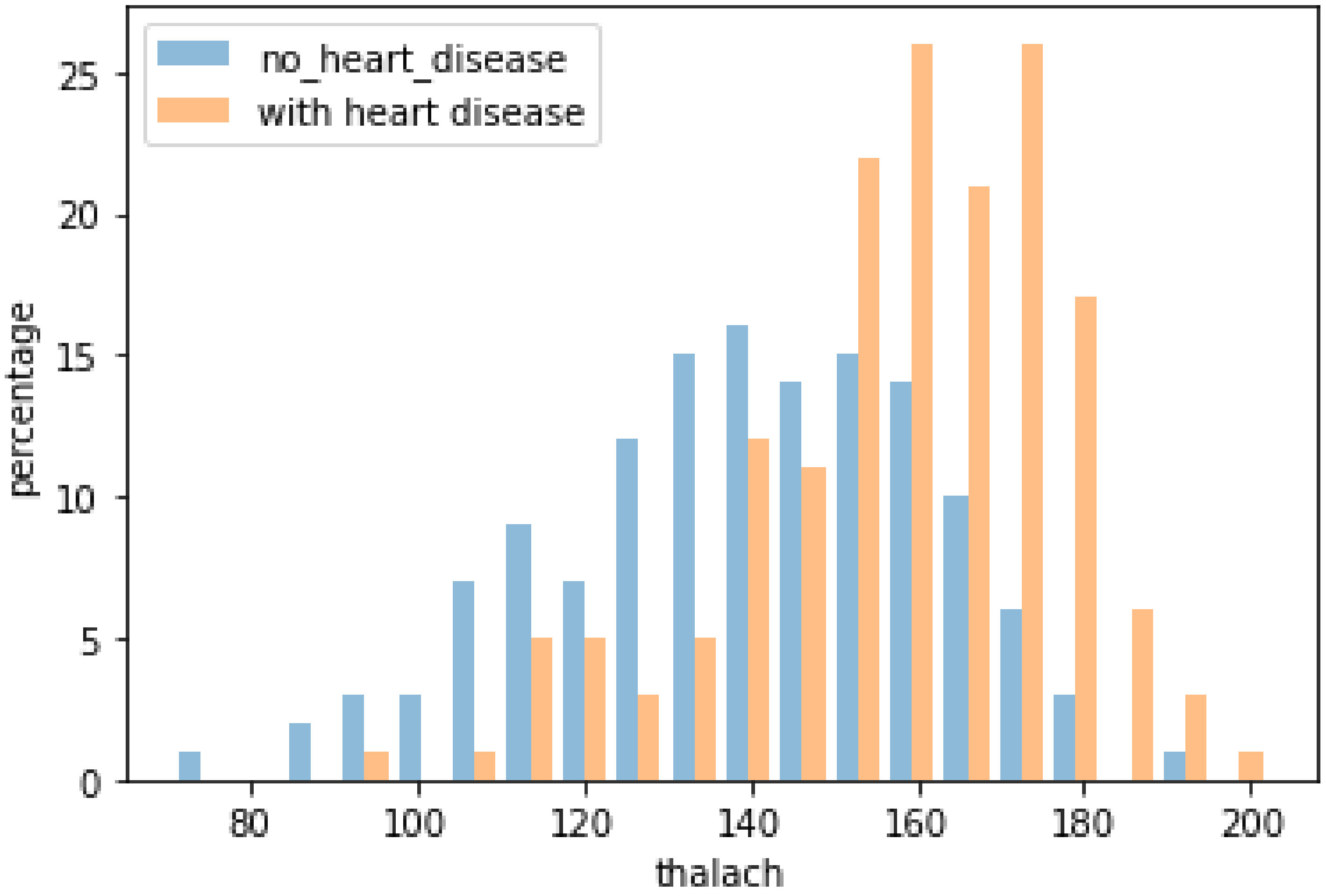
The ideal blood pressure should be lower than 120 mmHg.



EDA and insights

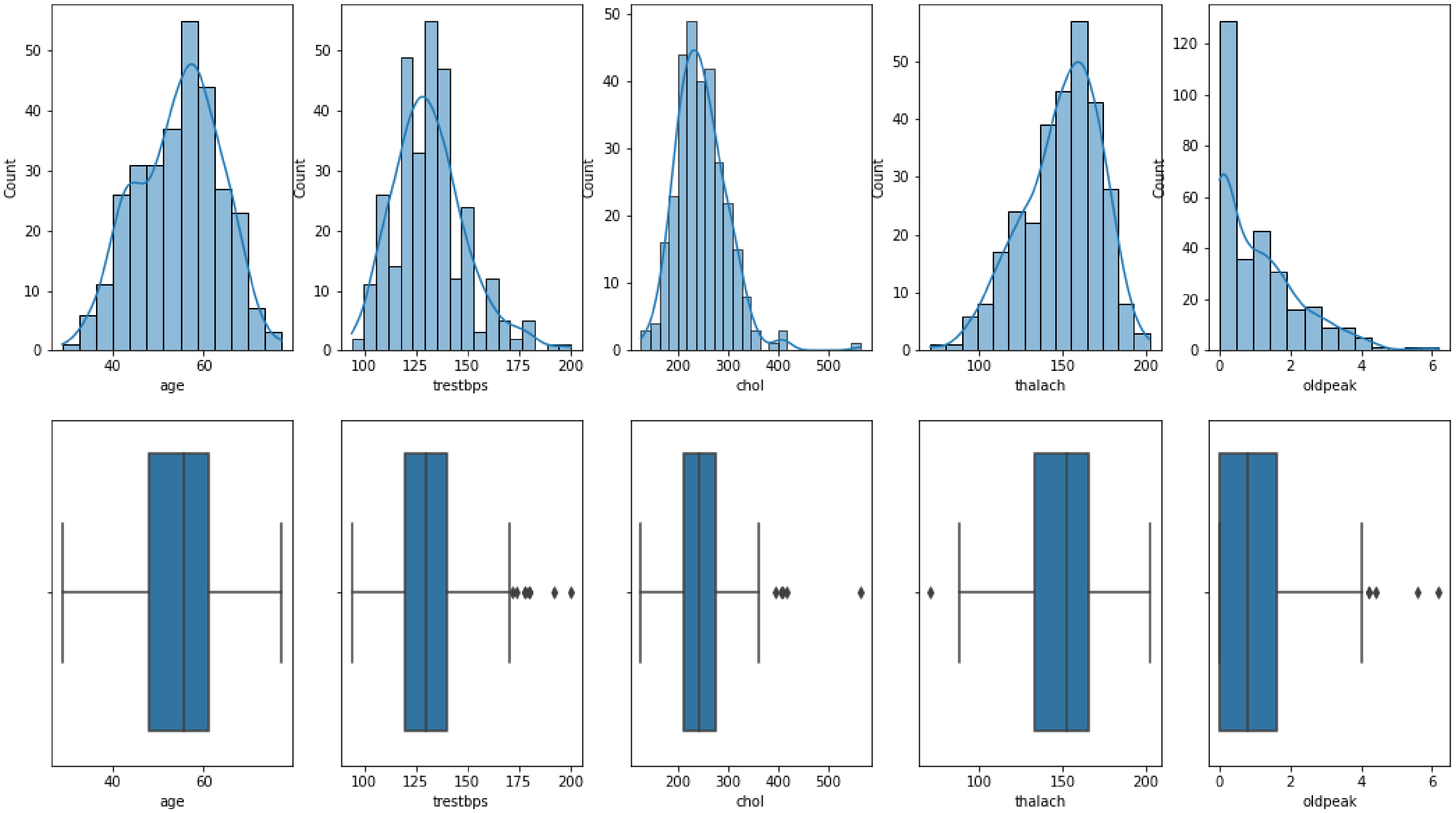
- Distribution of maximum heart rate (which is negatively related to the age)

Seems like the heart rate is really high for those with heart disease.



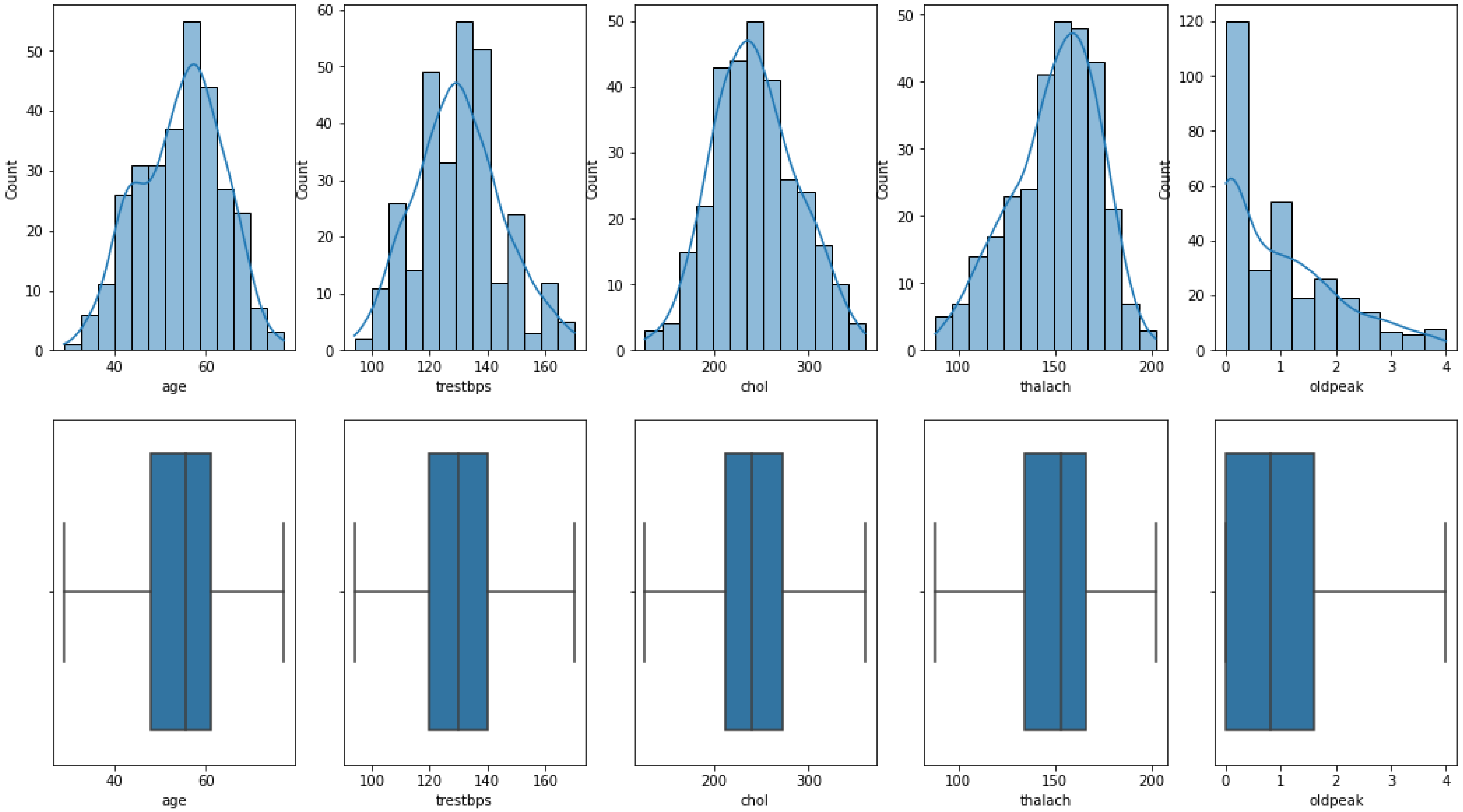
EDA and insights

- Removing the outliers: Data with outliers



EDA and insights

- Data without ouliers



Log transformation and scaling the data

```
In [34]: # check to see skewed data and perform transformation of numerical columns
         log_columns = clean_df[numerical].skew().sort_values(ascending=False)
         log_columns = log_columns.loc[log_columns > 0.75]

         log_columns

Out[34]: oldpeak    0.96995
         dtype: float64
```

```
In [35]: # perform log transformations
         for col in log_columns.index:
             clean_df[col] = np.log1p(clean_df[col])
```

```
In [37]: # scale the data
         sc = StandardScaler()
         feature_columns = [x for x in clean_df.columns if x not in categorical]
         for col in feature_columns:
             clean_df[col] = sc.fit_transform(clean_df[[col]])

         clean_df.head()
```

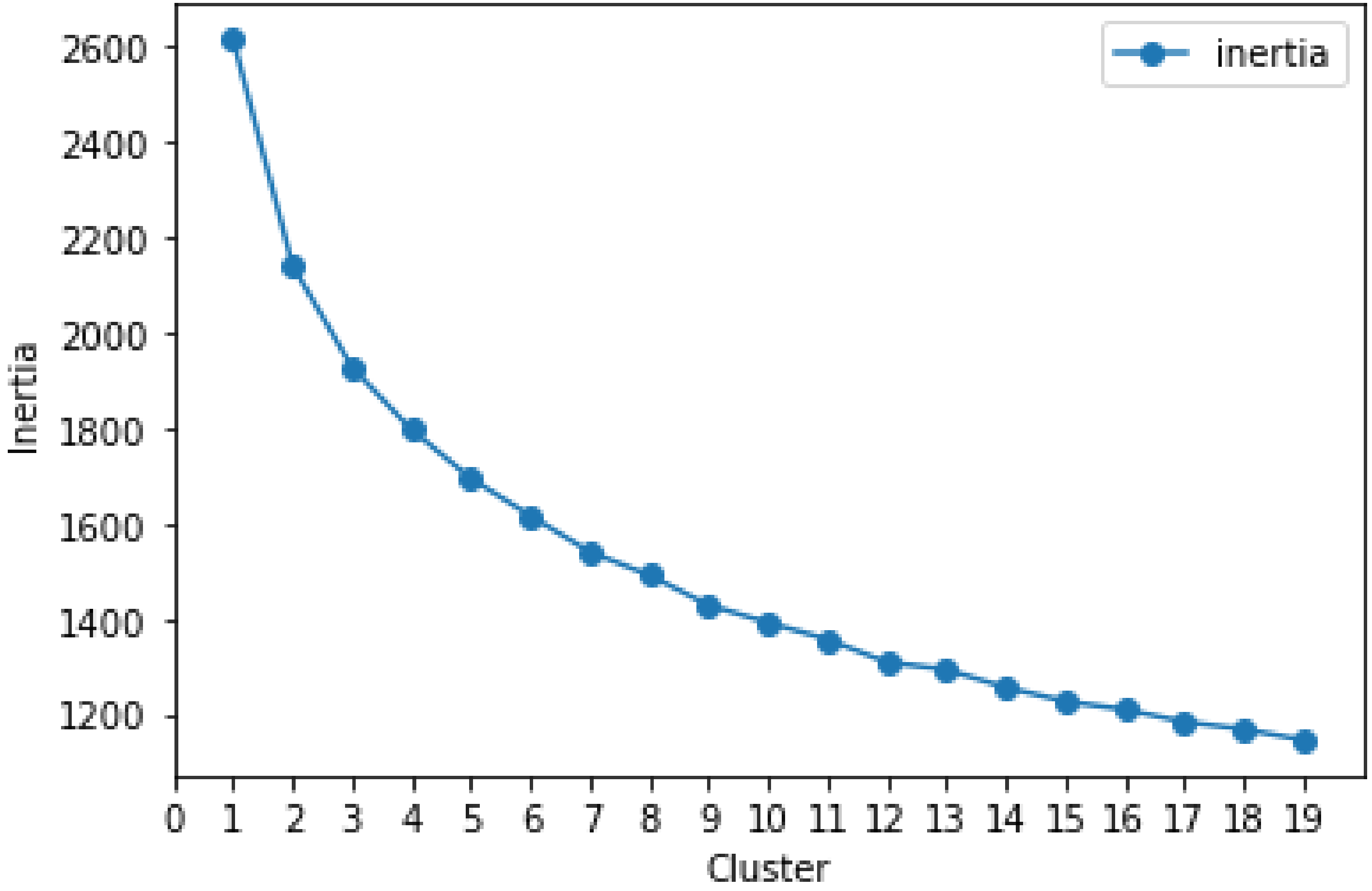
Out[37]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	0.949794	1	3	0.987461	-0.229564	1	0	0.007165	0	1.284737	0	0	1	1
1	-1.928548	1	2	-0.004379	0.152039	0	1	1.657982	0	1.905745	0	0	2	1
2	-1.485726	0	1	-0.004379	-0.880534	0	0	0.988732	0	0.647114	2	0	2	1
3	0.174856	1	1	-0.665606	-0.162222	0	1	1.256432	0	0.071103	2	0	2	1
4	0.285561	0	0	-0.665606	2.486552	0	1	0.587182	1	-0.164728	2	0	2	1

K-Means Clustering

- diagram below shows a plot of inertia versus clusters

The plot of inertia versus number of clusters shows an elbow at number of clusters equal to 2, that is k =2.



K-Means Clustering

In [42]:
```
# check to see how well it predicts with n_clusters set to 2
km = KMeans(n_clusters=2, random_state=42)
km = km.fit(clean_df.drop('target', axis=1))

clean_df['kmeans'] = km.predict(clean_df.drop('target', axis=1))
(clean_df[['target','kmeans']]
             .groupby(['kmeans','target'])
             .size()
             .to_frame()
             .rename(columns={0:'number'}))
```

Out[42]:

kmeans	target	number
0	0	40
	1	142
1	0	98
	1	22

Agglomerative Clustering

- Score using ward linkage

```
for linkage in ['complete', 'ward']:
    ag = AgglomerativeClustering(n_clusters=2, linkage=linkage, compute_full_tree=True)
    ag = ag.fit(clean_df.drop('target', axis=1))
    clean_df[str('agglom_'+linkage)] = ag.fit_predict(clean_df.drop('target', axis=1))
```

```
# score using ward linkage
(clean_df[['target','agglom_ward']]
        .groupby(['target','agglom_ward'])
        .size()
        .to_frame()
        .rename(columns={0:'number'}))
```

target	agglom_ward	number
0	0	37
	1	101
1	0	131
	1	33

```
# score using complete linkage
(clean_df[['target','agglom_complete']]
        .groupby(['target','agglom_complete'])
        .size()
        .to_frame()
        .rename(columns={0:'number'}))
```

Agglomerative Clustering

- Score Using complete linkage

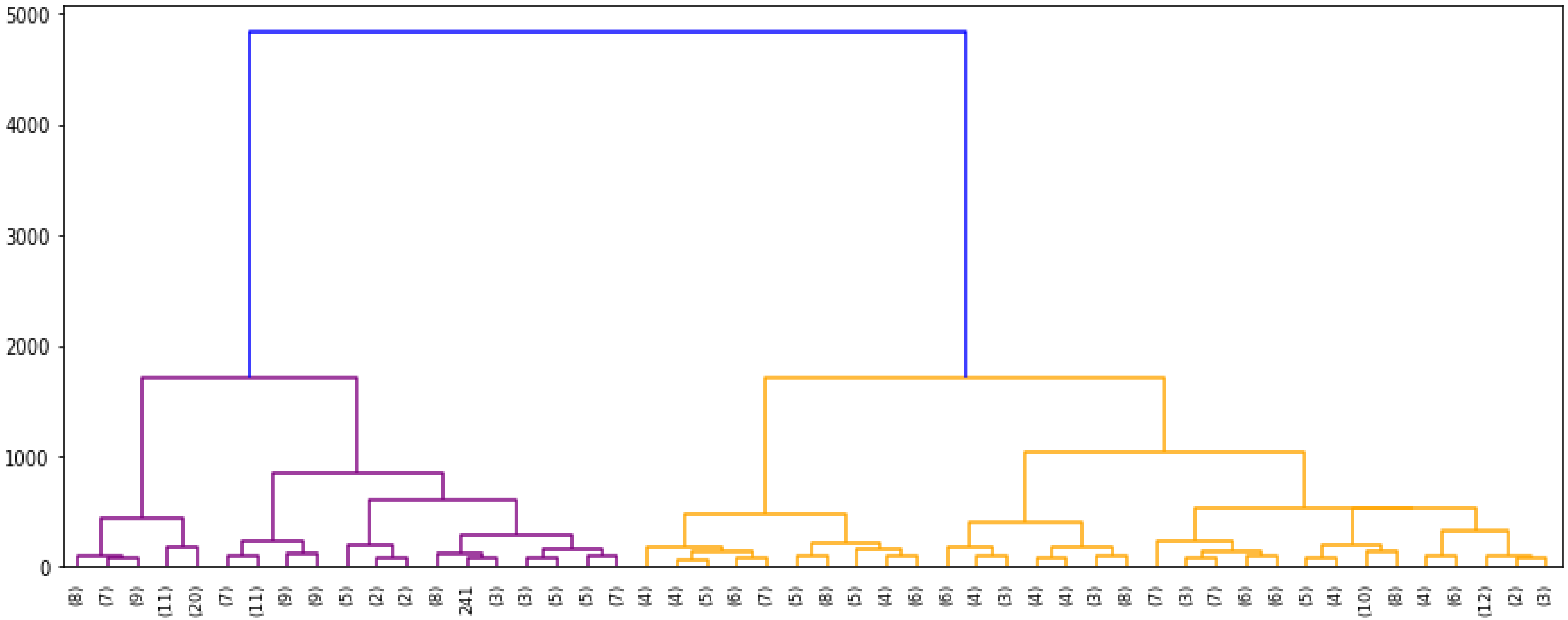
```
: # score using complete linkage
(clean_df[['target','agglom_complete']]
          .groupby(['target','agglom_complete'])
          .size()
          .to_frame()
          .rename(columns={0:'number'}))
```

:

target	agglom_complete	number
0	0	93
	1	45
1	0	18
	1	146

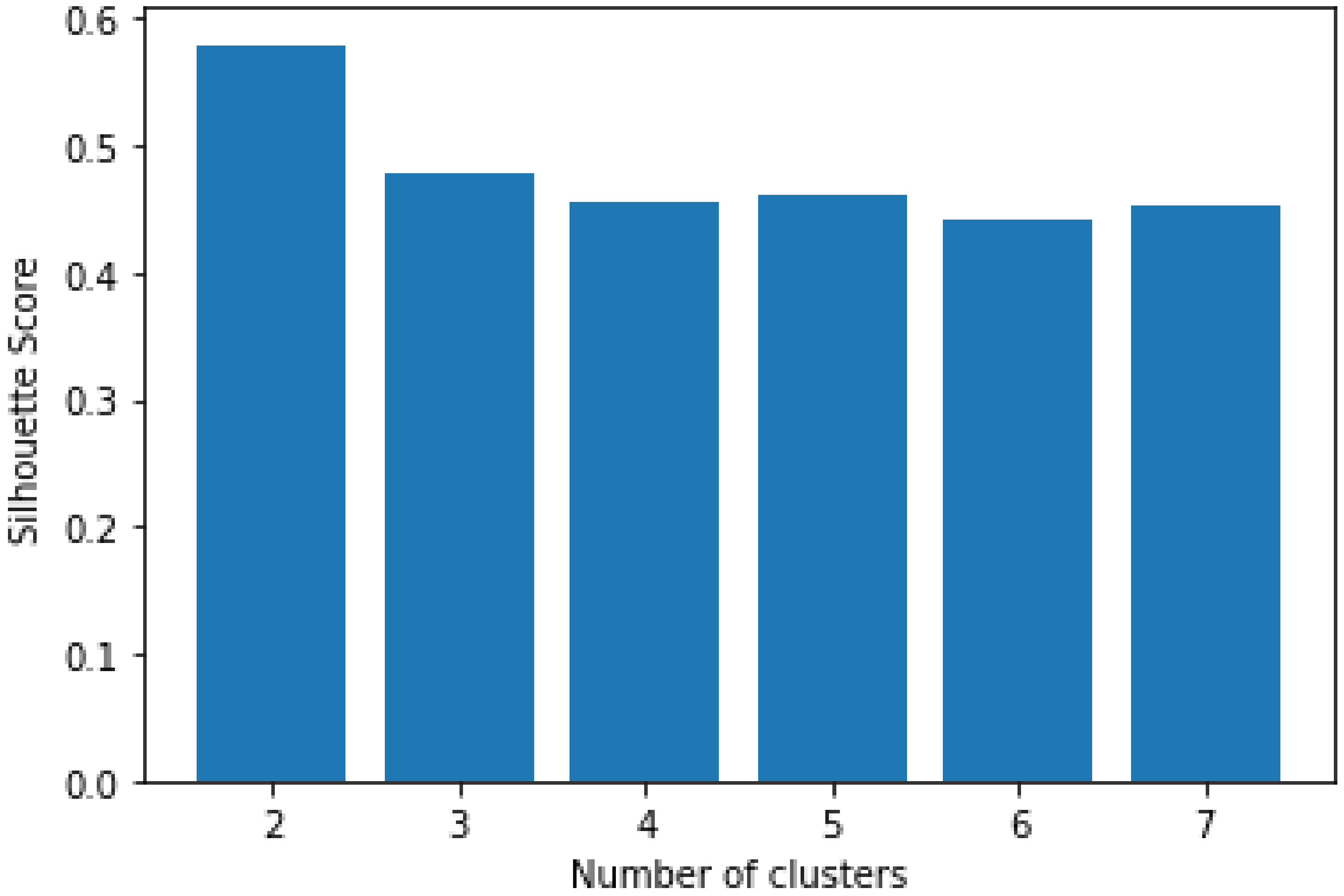
K-Means Clustering

- Diagram below shows the dendrogram for the Hierarchical Agglomerative Clustering model



K-Means Clustering

- Silhouette_scores by number of clusters



Conclusion

- From the abovementioned analysis, a few important findings can be outlined. Performing both K-means and agglomerative clustering algorithms, one could observe that the best model for the prediction of a potential myocardial infarction is the **Complete-link agglomerative technique**. On the contrary, for predicting those cases that there won't be any implications, the most suitable is the **Ward-link agglomerative clustering**. From both the dendrogram and the silhoute score plots, it is evident that the optimal number of the clusters is **two**.

Next steps

- As a further suggestion, a DBSCAN could be implemented, following a Principal Component Analysis.

- Link to the Github Repo: https://github.com/LucianPopaLVP/Project-4---Unsupervised-Machine-Learning


Thank you!