Machine Learning

Course-End Project Problem Statement

Get Certified. Get Ahead.

Course-End Project: Healthcare

Problem statement:

Cardiovascular diseases are the leading cause of death globally. It is therefore necessary to identify the causes and develop a system to predict heart attacks in an effective manner. The data below has the information about the factors that might have an impact on cardiovascular health.

Dataset description:

Dataset name: **CEP 1_ Dataset.xlsx**

Variable	**Description**
Age	Age in years
Sex	1 = male; 0 = female
cp|	Chest pain type
trestbps	Resting blood pressure (in mm Hg on admission to the hospital)
chol	Serum cholesterol in mg/dl
fbs	Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
restecg	Resting electrocardiographic results
thalach	Maximum heart rate achieved
exang	Exercise induced angina (1 = yes; 0 = no)
oldpeak	ST depression induced by exercise relative to rest
slope	Slope of the peak exercise ST segment
ca	Number of major vessels (0-3) colored by fluoroscopy
thal	3 = normal; 6 = fixed defect; 7 = reversible defect

Target	1 or 0

<u>Task to be performed:</u>

1. Preliminary analysis:
    a. Perform preliminary data inspection and report the findings on the structure of the data, missing values, duplicates, etc.
    b. Based on these findings, remove duplicates (if any) and treat missing values using an appropriate strategy

2. Prepare a report about the data explaining the distribution of the disease and the related factors using the steps listed below:
    a. Get a preliminary statistical summary of the data and explore the measures of central tendencies and spread of the data
    b. Identify the data variables which are categorical and describe and explore these variables using the appropriate tools, such as count plot
    c. Study the occurrence of CVD across the Age category
    d. Study the composition of all patients with respect to the Sex category
    e. Study if one can detect heart attacks based on anomalies in the resting blood pressure (trestbps) of a patient
    f. Describe the relationship between cholesterol levels and a target variable
    g. State what relationship exists between peak exercising and the occurrence of a heart attack
    h. Check if thalassemia is a major cause of CVD
    i. List how the other factors determine the occurrence of CVD
    j. Use a pair plot to understand the relationship between all the given variables
3. Build a baseline model to predict the risk of a heart attack using a logistic regression and random forest and explore the results while using correlation analysis and logistic regression (leveraging standard error and p-values from statsmodels) for feature selection

<u>Solution :</u>

1. Preliminary analysis:

a) Perform preliminary data inspection and report the findings on the structure of the data, missing values, duplicates, etc.

- Import libraries and read file

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import warnings
warnings.filterwarnings('ignore')

df=pd.read_excel("C:/Users/david/Desktop/personal/AI/course_3_PG AIML - Machine Learning/proj/healthcare/1645792390_cep1_dataset.xlsx")
```

```
df.head()
```

```
df.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
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

```
df.info()
```

```
df.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   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

```
df.shape
```

```
df.shape
```

```
(303, 14)
```

```
# Missing values
df.isnull().sum()
```

```
# Missing values
df.isnull().sum()
```

```
age          0
sex          0
cp           0
trestbps     0
chol        71
fbs          0
restecg      0
thalach      0
exang        0
oldpeak      0
slope        0
ca           0
thal         0
dtype: int64
```

No missing values in data set.

```
df.duplicated().sum()
```

```
: df.duplicated().sum()

: 1
```

One duplicate value present.

b) Based on these findings, remove duplicates (if any) and treat missing values using an appropriate strategy

- Remove the duplicates

```
df2=df[df.duplicated()]
df2
```

```
df=df.drop_duplicates()
df.duplicated().sum()
```

```
df2=df[df.duplicated()]
df2
```

	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

```
df=df.drop_duplicates()
df.duplicated().sum()
```

```
0
```

```
# removed duplicate value
```

2. Prepare a report about the data explaining the distribution of the disease and the related factors using the steps listed below:
   a. Get a preliminary statistical summary of the data and explore the measures of central tendencies and spread of the data

```
df.describe().T
```

```
df.describe().T
```

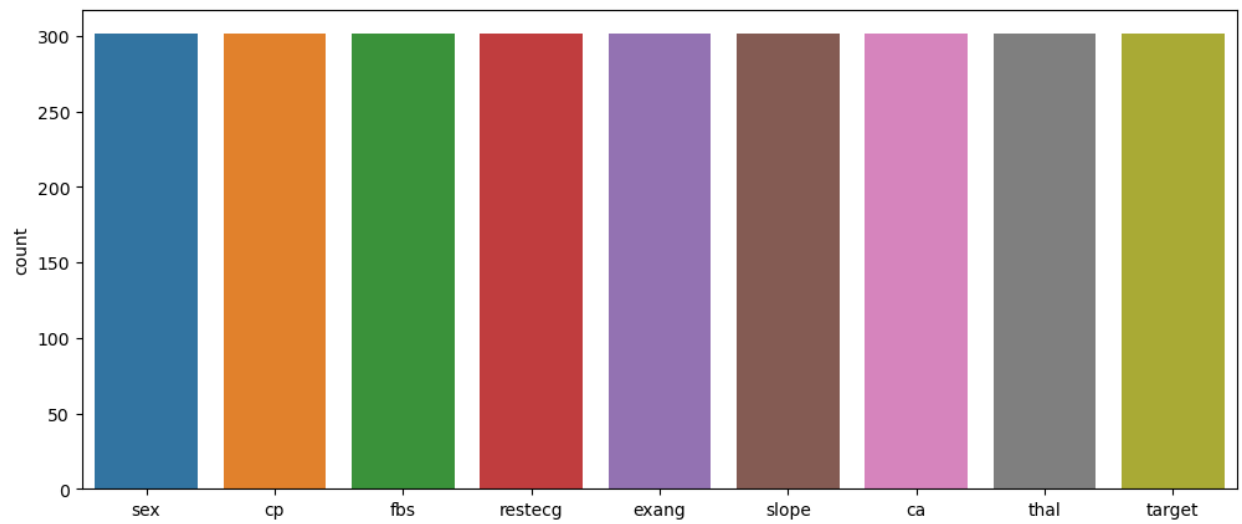
	count	mean	std	min	25%	50%	75%	max
age	303.0	54.366337	9.082101	29.0	47.5	55.0	61.0	77.0
sex	303.0	0.683168	0.466011	0.0	0.0	1.0	1.0	1.0
cp	303.0	0.966997	1.032052	0.0	0.0	1.0	2.0	3.0
trestbps	303.0	131.623762	17.538143	94.0	120.0	130.0	140.0	200.0
chol	303.0	246.264026	51.830751	126.0	211.0	240.0	274.5	564.0
fbs	303.0	0.148515	0.356198	0.0	0.0	0.0	0.0	1.0
restecg	303.0	0.528053	0.525860	0.0	0.0	1.0	1.0	2.0
thalach	303.0	149.646865	22.905161	71.0	133.5	153.0	166.0	202.0
exang	303.0	0.326733	0.469794	0.0	0.0	0.0	1.0	1.0
oldpeak	303.0	1.039604	1.161075	0.0	0.0	0.8	1.6	6.2
slope	303.0	1.399340	0.616226	0.0	1.0	1.0	2.0	2.0
ca	303.0	0.729373	1.022606	0.0	0.0	0.0	1.0	4.0
thal	303.0	2.313531	0.612277	0.0	2.0	2.0	3.0	3.0
target	303.0	0.544554	0.498835	0.0	0.0	1.0	1.0	1.0

so here categorical variables are sex , cp , fbs ,restecg , exang ,slope , ca,thal ,target.

b. Identify the data variables which are categorical and describe and explore these variables using the appropriate tools, such as count plot

```
cat_df=df[['sex', 'cp', 'fbs', 'restecg','exang', 'slope', 'ca',
'thal', 'target']]
plt.figure(figsize=(5,5))
sns.countplot(data=cat_df)
plt.show()
```

```
cat_df=df[['sex', 'cp', 'fbs', 'restecg','exang', 'slope', 'ca', 'thal', 'target']]
plt.figure(figsize=(12,5))
sns.countplot(data=cat_df)
plt.show()
```



c. Study the occurrence of CVD across the Age category

```
age_df=df[['age', 'target']]
age_df.groupby(['target']).mean()
```

```
age_df=df[['age', 'target']]
age_df.groupby(['target']).mean()
```

	age
target	
0	56.601449
1	52.496970

age has more impact target 0.

d. Study the composition of all patients with respect to the Sex category

```
df.groupby(['sex']).mean()
```

```
df.groupby(['sex']).mean()
```

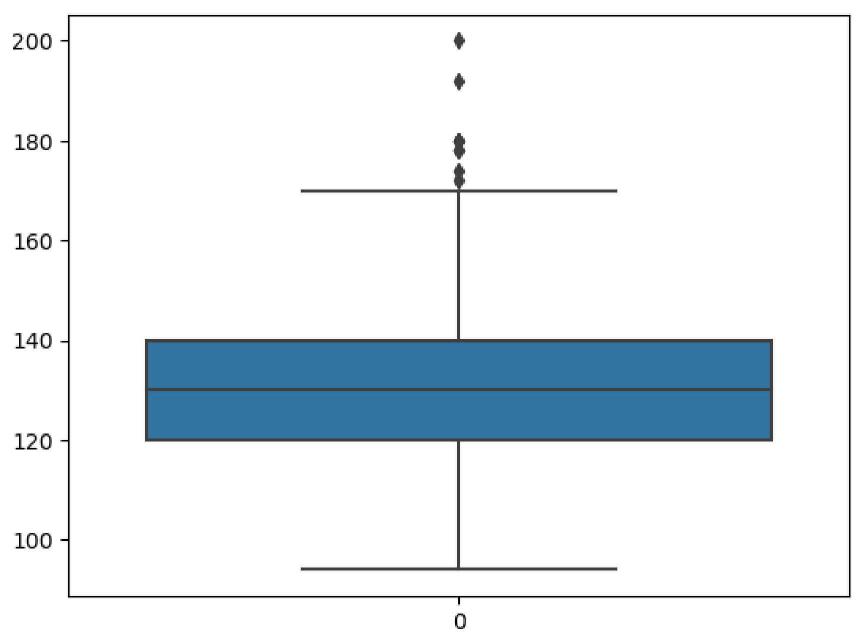
sex	age	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	55.677083	1.041667	133.083333	261.302083	0.12500	0.572917	151.125000	0.229167	0.876042	1.427083	0.552083	2.125000	0.750000
1	53.758454	0.932367	130.946860	239.289855	0.15942	0.507246	148.961353	0.371981	1.115459	1.386473	0.811594	2.400966	0.449275

```
#sex has more impact on target 0
```

Sex has more impact on target 0.

e. Study if one can detect heart attacks based on anomalies in the resting blood pressure (trestbps) of a patient.

```
sns.boxplot(df['trestbps'])
```



here occurence of outliers are at 170.

```
df[df['trestbps']>170]['target'].value_counts()
df[df['trestbps']>180]['target'].value_counts()
```

```
df[df['trestbps']>170]['target'].value_counts()
```

```
0    6
1    3
Name: target, dtype: int64
```

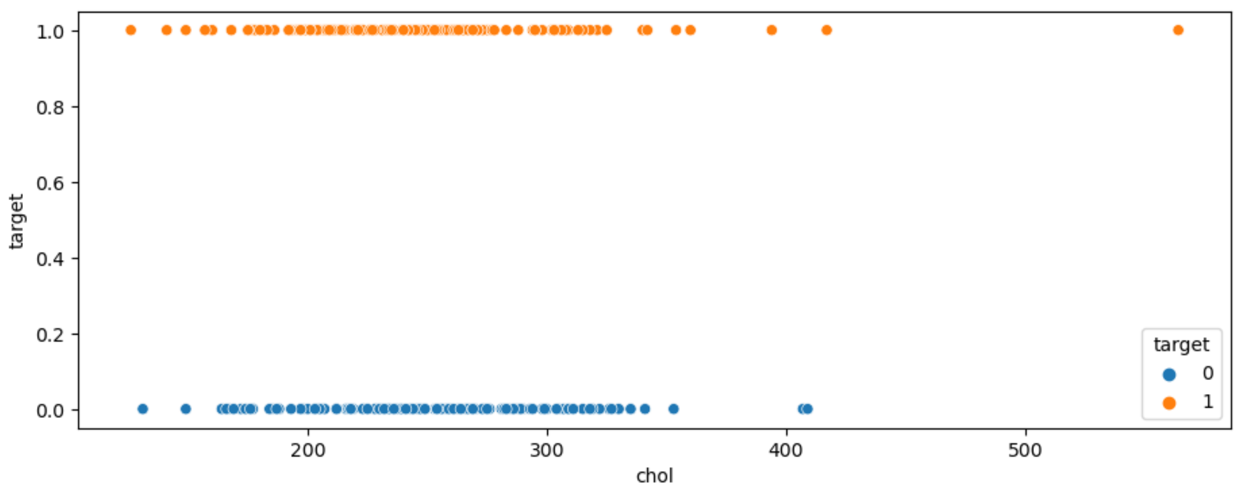
```
df[df['trestbps']>180]['target'].value_counts()
```

```
0    2
Name: target, dtype: int64
```

```
#it has high impact on target 0
```

f. Describe the relationship between cholesterol levels and a target variable

```
plt.figure(figsize=(11,4))
sns.scatterplot(data=df,x='chol',y='target',hue='target')
plt.show()
```




```
chol_df=df[['chol', 'target']]
chol_df.groupby(['target']).mean()
```

```python
chol_df=df[['chol', 'target']]
chol_df.groupby(['target']).mean()
```

target	chol
0	251.086957
1	242.230303

```python
#chol has more impact on target 0
```

g. State what relationship exists between peak exercising and the occurrence of a heart attack

```python
slope_df=df[['slope', 'target']]
slope_df.groupby(['target']).mean()
```

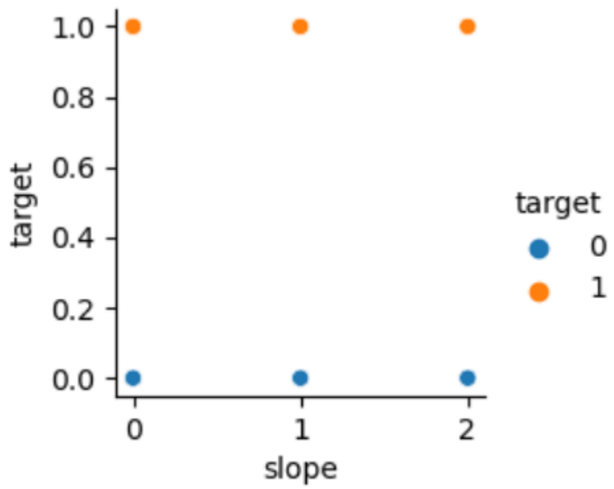
```python
slope_df=df[['slope', 'target']]
slope_df.groupby(['target']).mean()
```

target	slope
0	1.166667
1	1.593939

```python
plt.figure(figsize=(11,4))
sns.pairplot(data=df,x_vars='slope',y_vars='target',hue='target'
)
plt.show()
```

```python
plt.figure(figsize=(11,4))
sns.pairplot(data=df,x_vars='slope',y_vars='target',hue='target')
plt.show()
```

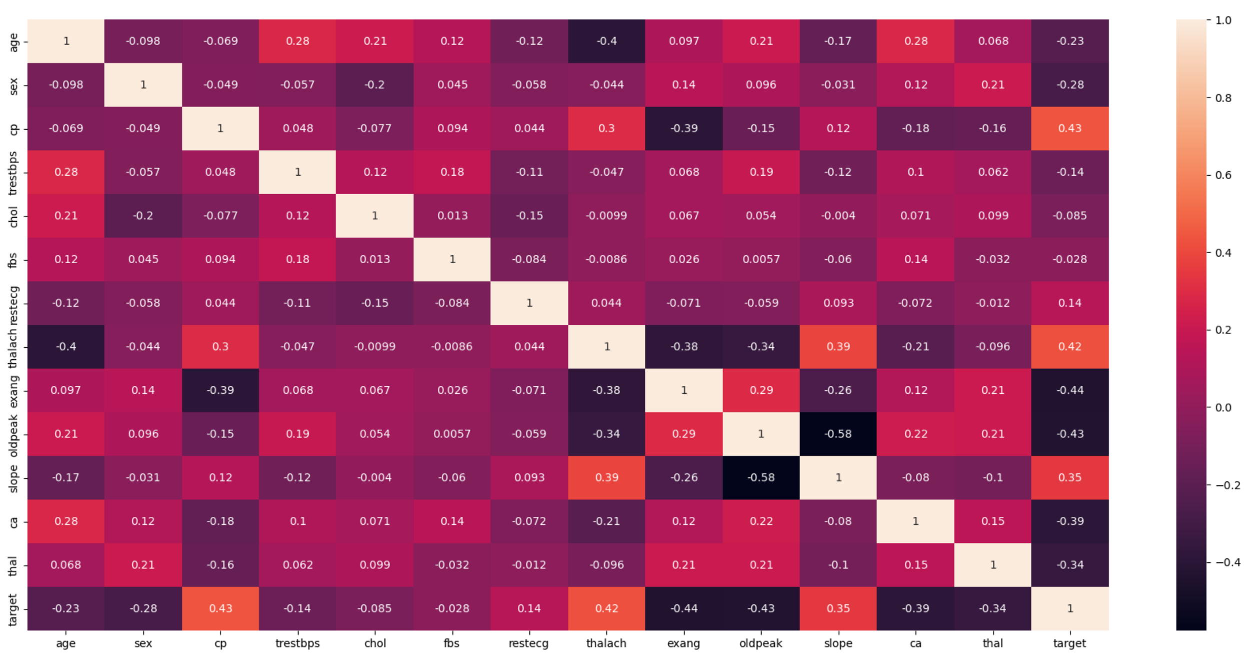
```
<Figure size 1100x400 with 0 Axes>
```



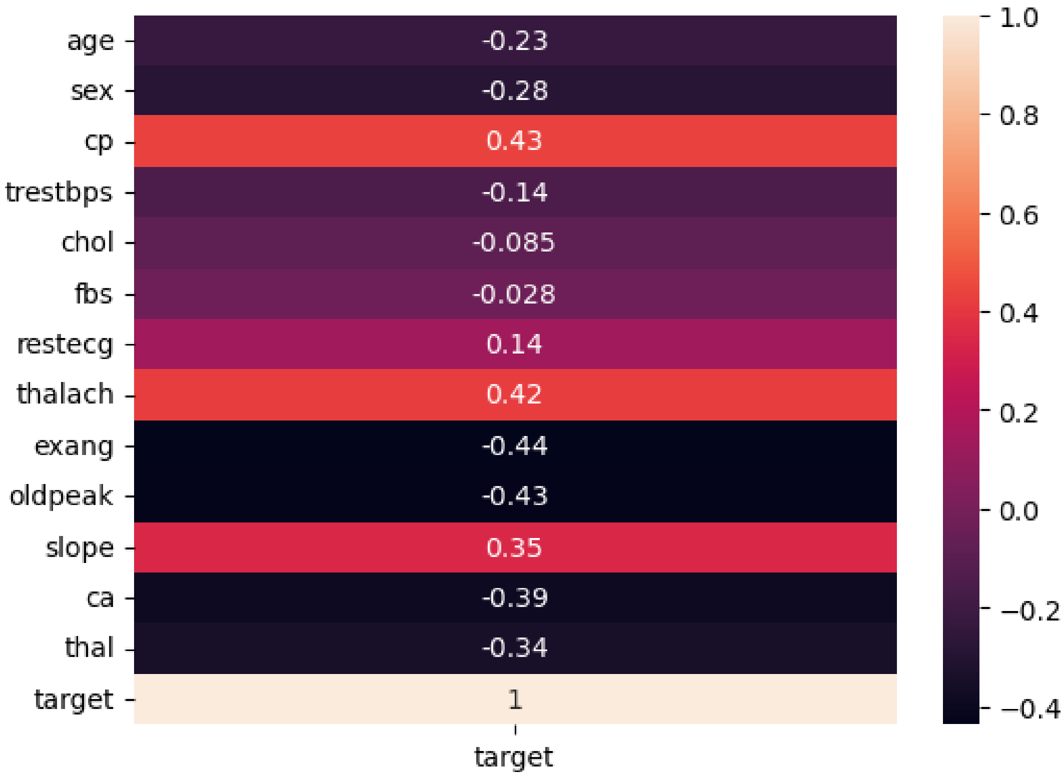

```python
# slope that us peak exercising has very less impact on both target 0 and 1
```

h. Check if thalassemia is a major cause of CVD

```python
plt.figure(figsize = (22,10))
sns.heatmap(df.corr(),annot=True)
```



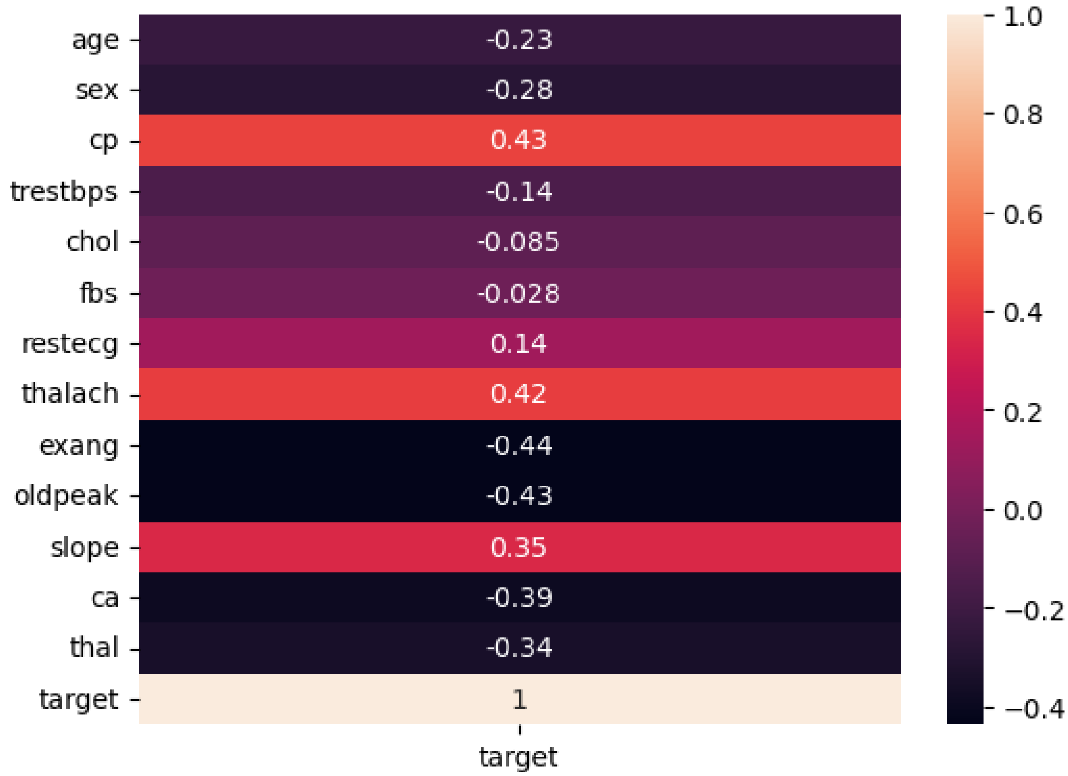
```
sns.heatmap(df.corr()[['target']], annot = True)
```



‘thal' has less impact (-0.34), thalassemia is not a major cause of CVD.

i. List how the other factors determine the occurrence of CVD

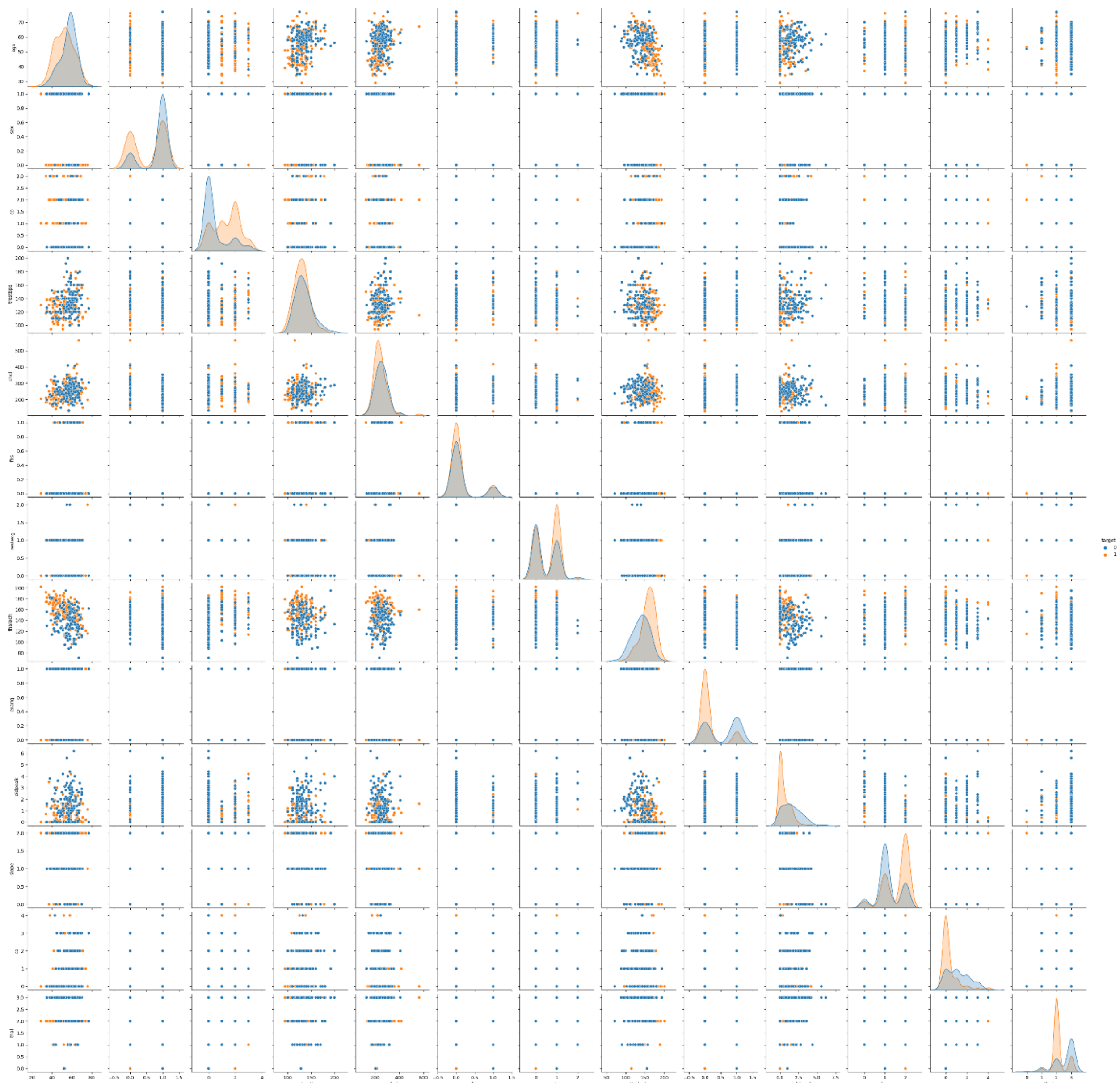
```
sns.heatmap(df.corr()[['target']], annot = True)
```



features ‘cp’ and ‘thalach’ as major impact on target 1 and features ‘exang ‘and ‘oldpeak ‘ have major impact on target 0

j. Use a pair plot to understand the relationship between all the given variables

```
plt.figure(figsize=(11,4))
sns.pairplot(data=df,hue='target')
plt.show()
```



3. Build a baseline model to predict the risk of a heart attack using a logistic regression and random forest and explore the results while using correlation analysis and logistic regression (leveraging standard error and p-values from statsmodels) for feature selection

    - Identify x and y

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

```
x = df[features]
y = df[target]
```

```
#Identify x and y
```

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

```
x = df[features]
y = df[target]
```

- Splitting data

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y,
test_size=0.33, random_state=42)
```

```
#splitting
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)
```

- Using statsmodel

```
import statsmodels.api as sm

c= sm.add_constant(X_train)
model = sm.Logit(y_train, X_train).fit()
model.summary()
```

```
import statsmodels.api as sm
```

```
c= sm.add_constant(X_train)
model = sm.Logit(y_train, X_train).fit()
model.summary()
```

Logit Regression Results

Dep. Variable:	target	**No. Observations:**	202
Model:	Logit	**Df Residuals:**	195
Method:	MLE	**Df Model:**	6
Date:	Mon, 12 Jun 2023	**Pseudo R-squ.:**	0.5288
Time:	21:11:03	**Log-Likelihood:**	-65.806
converged:	True	**LL-Null:**	-139.66
Covariance Type:	nonrobust	**LLR p-value:**	2.363e-29

	coef	std err	z	P>|z|	[0.025	0.975]
sex	-1.3318	0.514	-2.592	0.010	-2.339	-0.325
cp	0.8178	0.229	3.566	0.000	0.368	1.267
thalach	0.0200	0.006	3.161	0.002	0.008	0.032
exang	-1.2170	0.482	-2.525	0.012	-2.162	-0.272
slope	1.2692	0.380	3.340	0.001	0.524	2.014
ca	-1.5515	0.329	-4.716	0.000	-2.196	-0.907
thal	-1.4290	0.365	-3.919	0.000	-2.144	-0.714

```
#selecting features with p value <0.05
x=['sex','cp','thalach','exang','slope','ca','thal']
X_train=X_train[x]
X_test=X_test[x]
```

```
#selecting features with p value <0.05

x=['sex','cp','thalach','exang','slope','ca','thal']
X_train=X_train[x]
X_test=X_test[x]
```

- Model building – logistic regression

```
from sklearn.linear_model import LogisticRegression

logreg=LogisticRegression()

logreg.fit(X_train,y_train)

pred=logreg.predict(X_test)

logreg.score(X_train,y_train)
```

```
logreg.score(X_test,y_test)
```

```
# model building
```

```
from sklearn.linear_model import LogisticRegression
```

```
logreg=LogisticRegression()
```

```
logreg.fit(X_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
pred=logreg.predict(X_test)
```

```
logreg.score(X_train,y_train)
```

```
0.8613861386138614
```

```
logreg.score(X_test,y_test)
```

```
0.82
```

- Model testing

```
#testing
from sklearn.metrics import confusion_matrix,
classification_report
confusion_matrix(y_test, pred)

print(classification_report(y_test, pred))
```

```
print(classification_report())
```

```
#testing
from sklearn.metrics import confusion_matrix, classification_report
confusion_matrix(y_test, pred)
```

```
array([[35,  8],
       [10, 47]], dtype=int64)
```

```
print(classification_report(y_test, pred))

print(classification_report())
```

```
              precision    recall  f1-score   support

           0       0.77      0.79      0.78        42
           1       0.84      0.83      0.83        58

    accuracy                           0.81       100
   macro avg       0.80      0.81      0.81       100
weighted avg       0.81      0.81      0.81       100
```

- Random forest model

```
from sklearn.ensemble import RandomForestClassifier

clf_rf = RandomForestClassifier()

clf_rf.fit(X_train, y_train)

clf_rf.score(X_test, y_test)
```

```
clf_rf.score(X_train, y_train)
```

```
#random forest model
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
clf_rf = RandomForestClassifier()
```

```
clf_rf.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
clf_rf.score(X_test, y_test)
```

```
0.81
```

```
clf_rf.score(X_train, y_train)
```

```
0.9900990099009901
```

- Testing the model

```
from sklearn.metrics import confusion_matrix,
classification_report

predictions = clf_rf.predict(X_test)

confusion_matrix(y_test, predictions)

print(classification_report(y_test, predictions))
```

```
predictions = clf_rf.predict(X_test)
```

```
confusion_matrix(y_test, predictions)
```

```
array([[32, 10],
       [10, 48]], dtype=int64)
```

```
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.76      0.76      0.76        42
           1       0.83      0.83      0.83        58

    accuracy                           0.80       100
   macro avg       0.79      0.79      0.79       100
weighted avg       0.80      0.80      0.80       100
```