

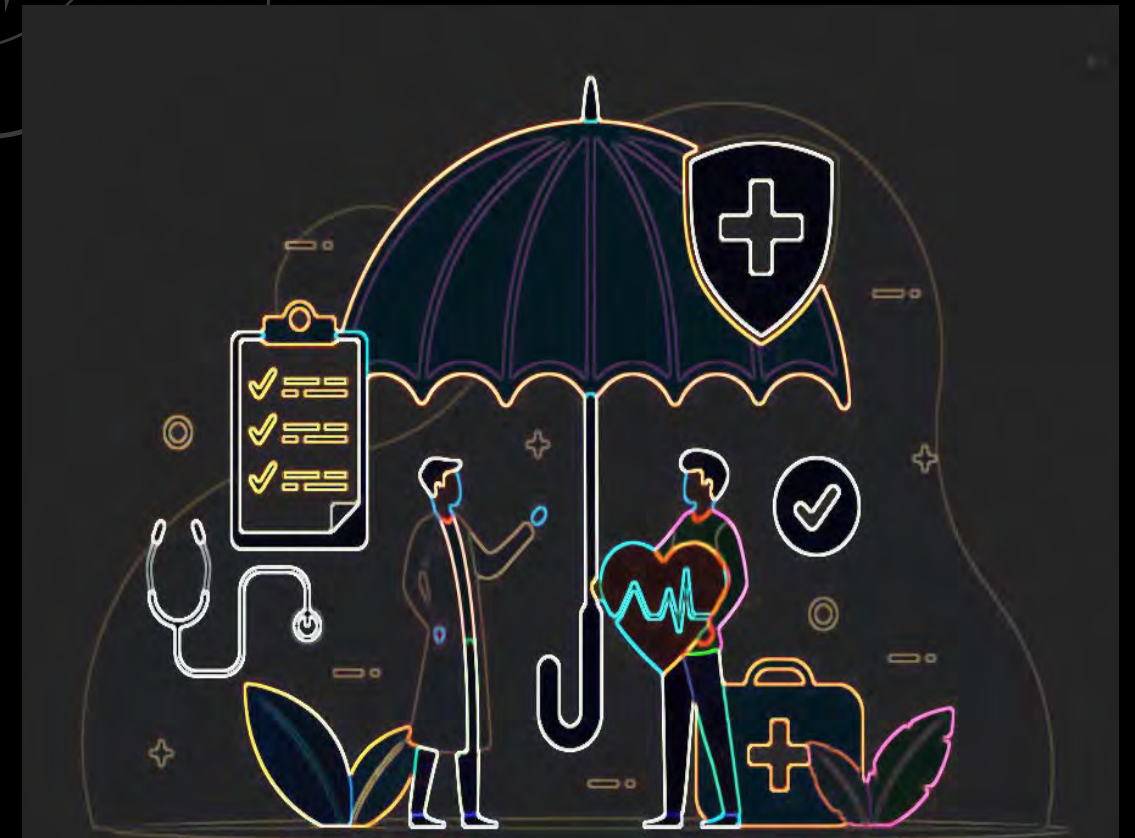
# Supervised Machine Learning: Classification

## Cardiovascular Disorder Prediction

M. Saqib Khan

# Contents

- Dataset Description
- Main objectives of the analysis.
- Applying various classification models.
- Machine learning analysis and findings.
- Models flaws and advanced steps.



# Data Description

# Motivation

- Predicting and diagnosing heart disease is one of the major challenges in the medical industry.
- It depends on several factors including physical examination and various symptoms and signs present in the patient.
- Heart disease is considered as one of the deadliest disease in the world for human life.
- The ML Algorithms play an major role in heart disease prediction.

# Motivation (*contd.*)

- There are several factors affecting heart disease including but not limited to:
  - ☐ cholesterol levels,
  - ☐ obesity,
  - ☐ family history of diseases,
  - ☐ blood pressure,
  - ☐ work environment,
  - ☐ smoking habits.

# Motivation (*contd.*)

- Rapid advances in technology allow ML to integrate with Big Data tools to manage the exponentially growing unstructured data that includes medical data for patients around the world.
- Cardiovascular disease can be predicted based on different symptoms like age, gender, heart rate etc. which in turn reduces the death rate for heart patients.
- The aim of this project is to use ML algorithms to predict these cardiovascular diseases.

# Dataset Description

- **age**: The person's age in years
- **sex**: The person's sex  
(1 = male, 0 = female)
- **cp**: chest pain type:
  - Value 0: asymptomatic
  - Value 1: atypical angina
  - Value 2: non-anginal pain
  - Value 3: typical angina
- **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).

	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



# Dataset Description

- **restecg**: resting electrocardiographic results  
Value 0: showing probable or definite left ventricular hypertrophy by Estes' criteria  
Value 1: normal  
Value 2: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of  $> 0.05$  mV).
- **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 ('ST' relates to positions on the ECG plot. See more here)



# Dataset Description

- **slope**: the slope of the peak exercise ST segment  
**0**: upsloping;  
**1**: flat;  
**2**: down sloping.
- **ca**: The number of major vessels (0–3)
- **thal**: A blood disorder called thalassemia  
**Value 0**: NULL (dropped from the dataset previously)  
**Value 1**: fixed defect (no blood flow in some part of the heart)  
**Value 2**: normal blood flow  
**Value 3**: reversible defect (a blood flow is observed but it is not normal)
- **target**: Heart disease (**0** = no, **1**= yes)

# Dataset Description

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00	303.00
mean	54.37	0.68	0.97	131.62	246.26	0.15	0.53	149.65	0.33	1.04	1.40	0.73	2.31	0.54
std	9.08	0.47	1.03	17.54	51.83	0.36	0.53	22.91	0.47	1.16	0.62	1.02	0.61	0.50
min	29.00	0.00	0.00	94.00	126.00	0.00	0.00	71.00	0.00	0.00	0.00	0.00	0.00	0.00
25%	47.50	0.00	0.00	120.00	211.00	0.00	0.00	133.50	0.00	0.00	1.00	0.00	2.00	0.00
50%	55.00	1.00	1.00	130.00	240.00	0.00	1.00	153.00	0.00	0.80	1.00	0.00	2.00	1.00
75%	61.00	1.00	2.00	140.00	274.50	0.00	1.00	166.00	1.00	1.60	2.00	1.00	3.00	1.00
max	77.00	1.00	3.00	200.00	564.00	1.00	2.00	202.00	1.00	6.20	2.00	4.00	3.00	1.00

# No missing values!!!

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

# Aim of the Project

- The aim is to show correlation between the features to find the most influence features on our target which is Target (Cardiovascular Disease Existence).
- The second step is to build appropriate Classification models using techniques such as Grid Search, pipelines, and hyperparameters tuning to get the best predictive model in terms of accuracy.
- Finally some pitfalls of this approach will be discussed.

# Data Analysis

# Data Analysis

Categorical Features : ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']

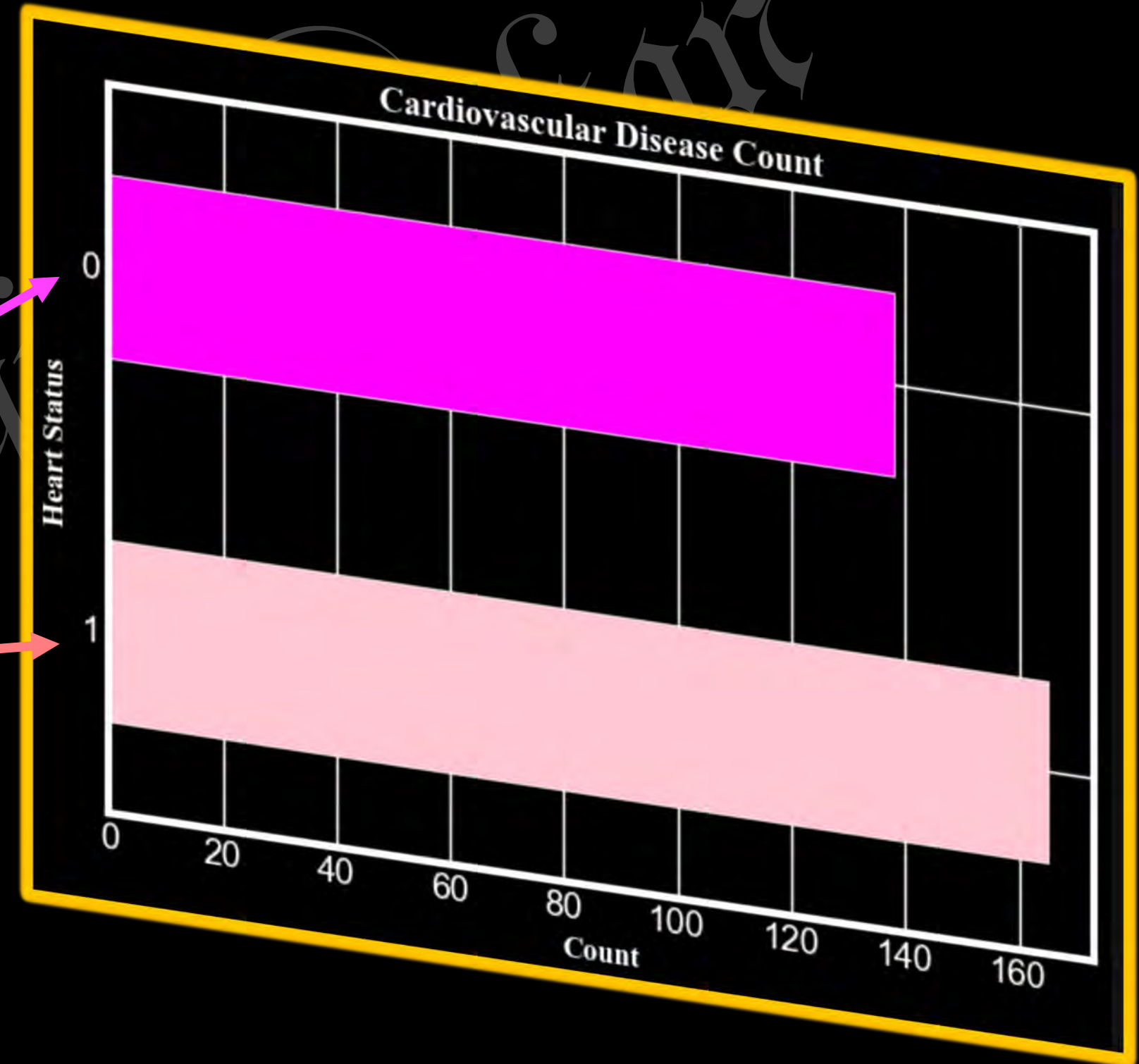
Continuous Features : ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

- Identifying:
  - ☐ categorical features
  - ☐ continuous features

# Viewing the status of people in the data set

- Data for the target variable to predict is in balance, since:

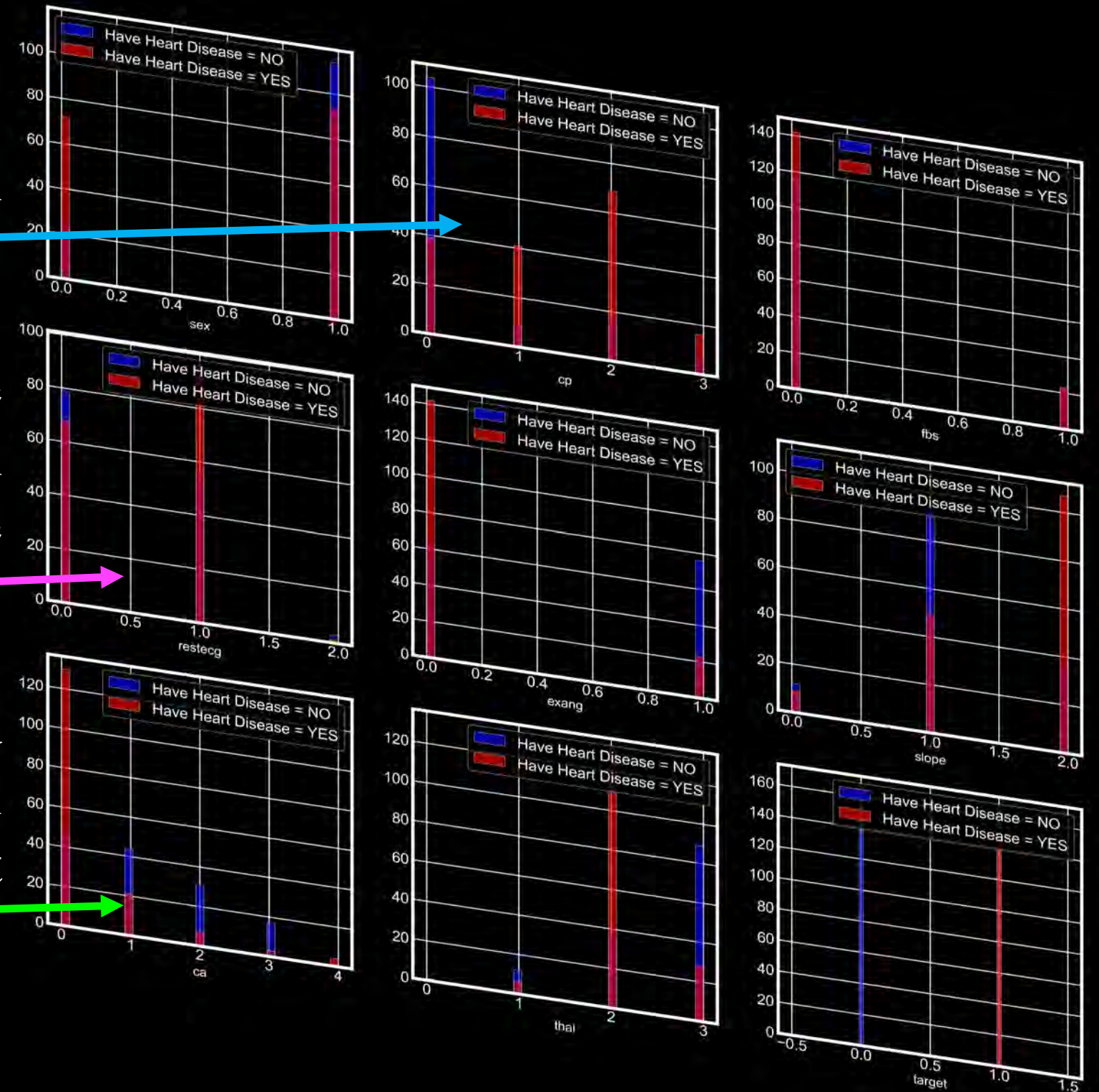
- ☐ 138 healthy people.
- ☐ 165 people with heart disease.





# Relationship b/w categorical features & heart disease

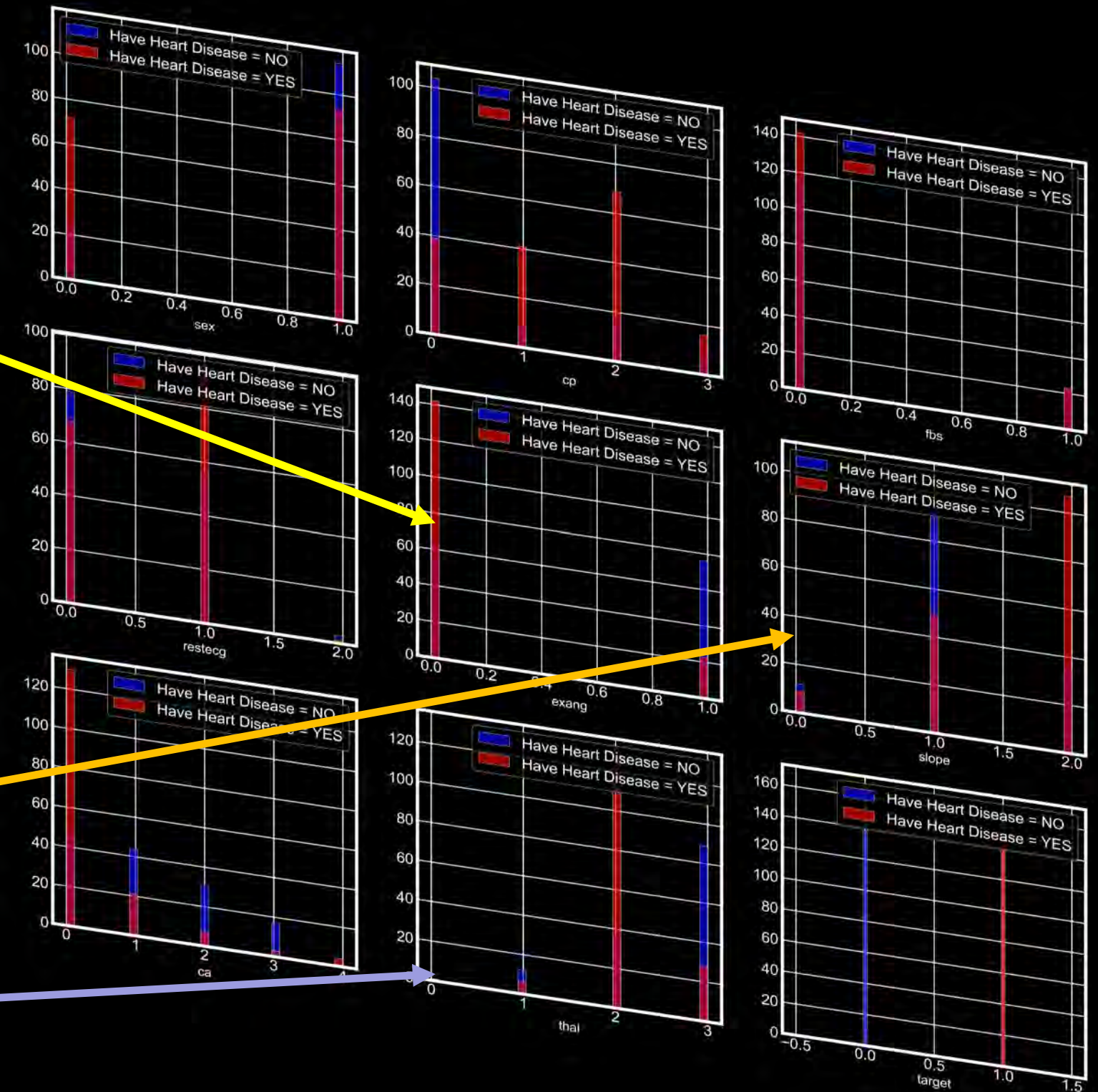
- **cp (chest pain)**: people with chest pain of the type: cp: [1, 2, 3] tend to have more heart disease than people without any chest pain cp: 0
- **restecg (resting ECG results)**: People with a value of 1 (having an abnormal heart rhythm, which can range from mild symptoms to severe problems) are more likely to develop heart disease.
- **ca (number of blood vessels (0-3))**: the more blood flow the better heart, so people with a vessel number ca equal to 0 are more likely to have heart disease.





# Relationship b/w categorical features & heart disease

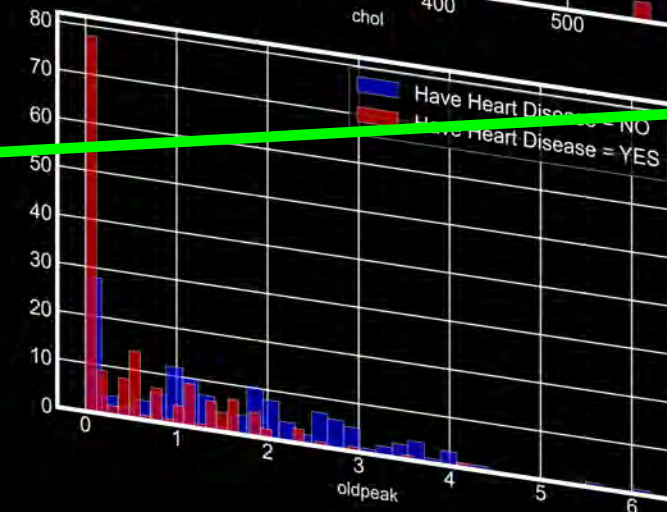
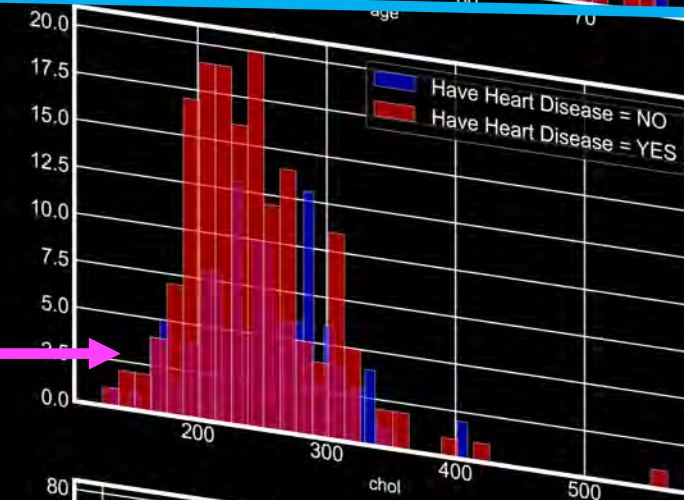
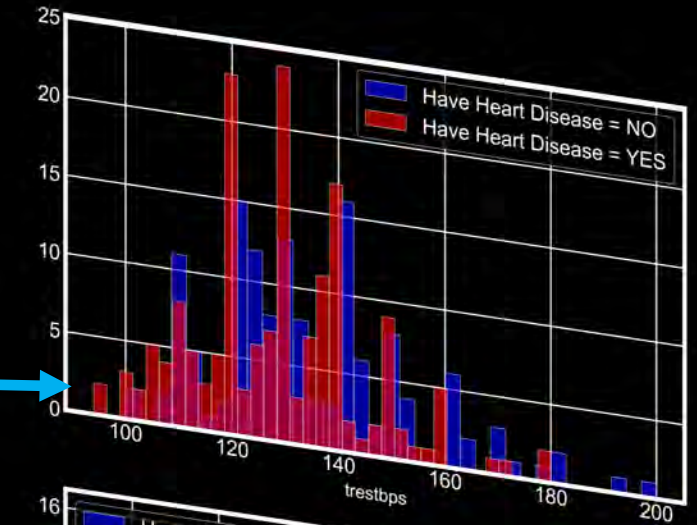
- **exang (exercise-induced angina)**: People with non-exercise-induced angina who have a value of 0 are more likely to have heart disease than those who have exercise-induced angina with a value of 1.
- **Slope (rectal slope for the ST segment of peak exercise)**: People with a down sloping slope of 2 have signs of an unhealthy heart therefore they more likely to have heart disease than people with an upsloping of 0 or a flat slope. A value of 1: minimal change (typical healthy heart).
- **thal (a blood disorder called thalassemia)**: People with a thal value = 2 are more likely to have heart disease.





# Relationship of continuous features & heart disease

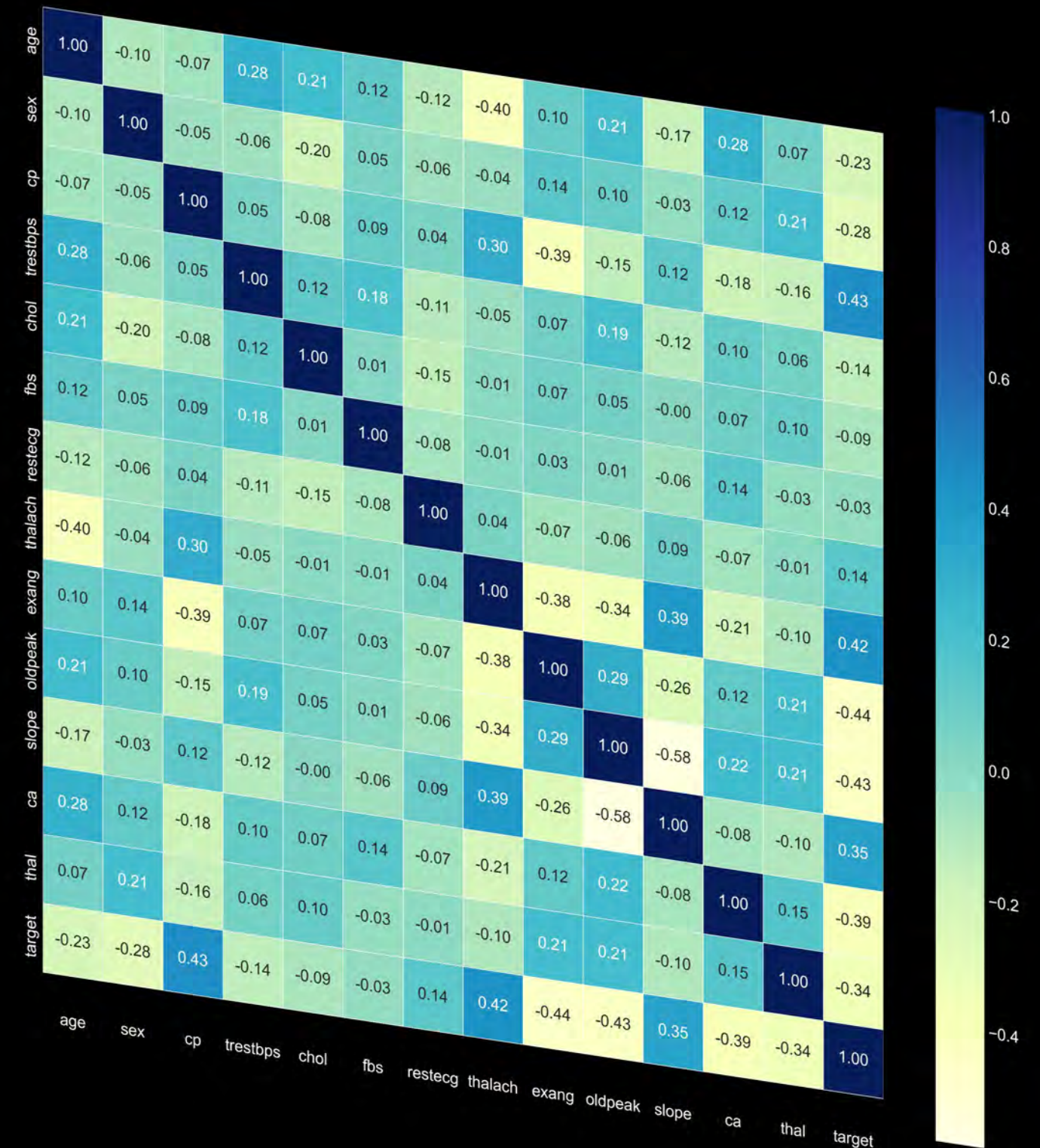
- **trestbps**: When blood pressure is higher than 130-140 mm Hg, this is a cause for concern.
- **chol**: When cholesterol is higher than 200 mg/dL, it is a very dangerous indicator, as shown in the graphic above.
- **thalach**: People with a heart rate above 140 are more likely to have heart disease.





# Heatmap of correlations between features

- Most important for us is the relationship between the target variable (knowing whether a person has a heart disease or not) and the rest of the features,
- Thus, our focus will be on the last row from the matrix.

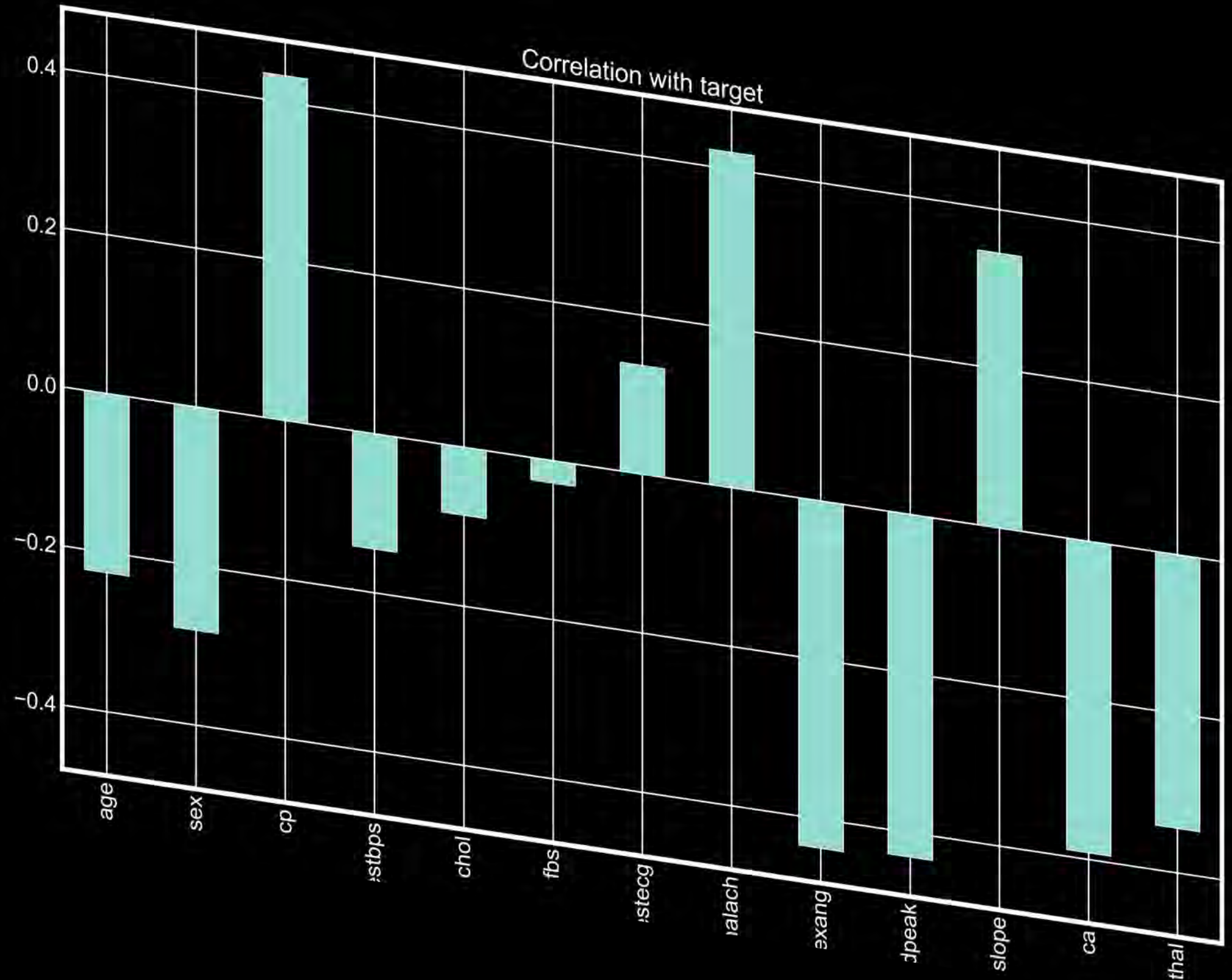


# Heatmap of correlations between features

- Note that:

- ☐ **fbs** and **chol** are the features least related to the target variable.

- ☐ All other features have a high correlation with the target variable.



# Converting Categorical features into Numerical

	age	trestbps	chol	thalach	oldpeak	target	sex_0	sex_1	cp_0	cp_1	...	slope_2	ca_0	ca_1	ca_2	ca_3	ca_4	thal_0	thal_1	thal_2	thal_3
0	0.95	0.76	-0.26	0.02	1.09	1	0	1	0	0	...	0	1	0	0	0	0	0	1	0	0
1	-1.92	-0.09	0.07	1.63	2.12	1	0	1	0	0	...	0	1	0	0	0	0	0	0	1	0
2	-1.47	-0.09	-0.82	0.98	0.31	1	1	0	0	1	...	1	1	0	0	0	0	0	0	1	0
3	0.18	-0.66	-0.20	1.24	-0.21	1	0	1	0	1	...	1	1	0	0	0	0	0	0	1	0
4	0.29	-0.66	2.08	0.58	-0.38	1	1	0	1	0	...	1	1	0	0	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	0.29	0.48	-0.10	-1.17	-0.72	0	1	0	1	0	...	0	1	0	0	0	0	0	0	0	1
299	-1.03	-1.23	0.34	-0.77	0.14	0	0	1	0	0	...	0	1	0	0	0	0	0	0	0	1
300	1.50	0.71	-1.03	-0.38	2.04	0	0	1	1	0	...	0	0	0	1	0	0	0	0	0	1
301	0.29	-0.09	-2.23	-1.52	0.14	0	0	1	1	0	...	0	0	1	0	0	0	0	0	0	1
302	0.29	-0.09	-0.20	1.06	-0.90	0	1	0	0	1	...	0	0	1	0	0	0	0	0	1	0

303 rows × 31 columns



# Analysis & Results

- My analysis will compare between 4 different classification models to predict the cardiovascular disease.
  - ☐ Logistic Regression,
  - ☐ SVM,
  - ☐ KNN,
  - ☐ XGBoost.
- Whereas the following techniques will be used to develop robust models:
  - ☐ Standard scaling,
  - ☐ cross-validation method,
  - ☐ Grid Search,
- I have used metric measurements such as:
  - ☐ accuracy.
  - ☐ precision.
  - ☐ F1 Score.



# Data Splitting

```
from sklearn.model_selection import StratifiedShuffleSplit
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay
feature_cols = [col_name for col_name in dataset.columns if col_name != 'target']

# Get the split indexes
strat_shuf_split = StratifiedShuffleSplit(n_splits=1,
                                         test_size=0.3,
                                         random_state=42)

train_idx, test_idx = next(strat_shuf_split.split(dataset[feature_cols], dataset.target))
# Create the dataframes
X_train = dataset.loc[train_idx, feature_cols]
y_train = dataset.loc[train_idx, 'target']

X_test = dataset.loc[test_idx, feature_cols]
y_test = dataset.loc[test_idx, 'target']
```

# Logistic Regression Model

```
### BEGIN SOLUTION
from sklearn.linear_model import LogisticRegression

# Standard logistic regression
lr = LogisticRegression(solver='liblinear').fit(X_train, y_train)
y_pred_0 = lr.predict(X_test)
clf_report = pd.DataFrame(classification_report(y_test, y_pred_0, output_dict=True))
clf_report
```

	0	1	accuracy	macro avg	weighted avg
precision	0.78	0.80	0.79	0.79	0.79
recall	0.76	0.82	0.79	0.79	0.79
f1-score	0.77	0.81	0.79	0.79	0.79
support	41.00	50.00	0.79	91.00	91.00

# Logistic Regression with L1 penalty

```
from sklearn.linear_model import LogisticRegressionCV

# L1 regularized logistic regression
lr_l1 = LogisticRegressionCV(Cs=10, cv=4, penalty='l1', solver='liblinear').fit(X_train, y_train)
y_pred_1 = lr_l1.predict(X_test)
clf_report = pd.DataFrame(classification_report(y_test, y_pred_1, output_dict=True))
clf_report
```

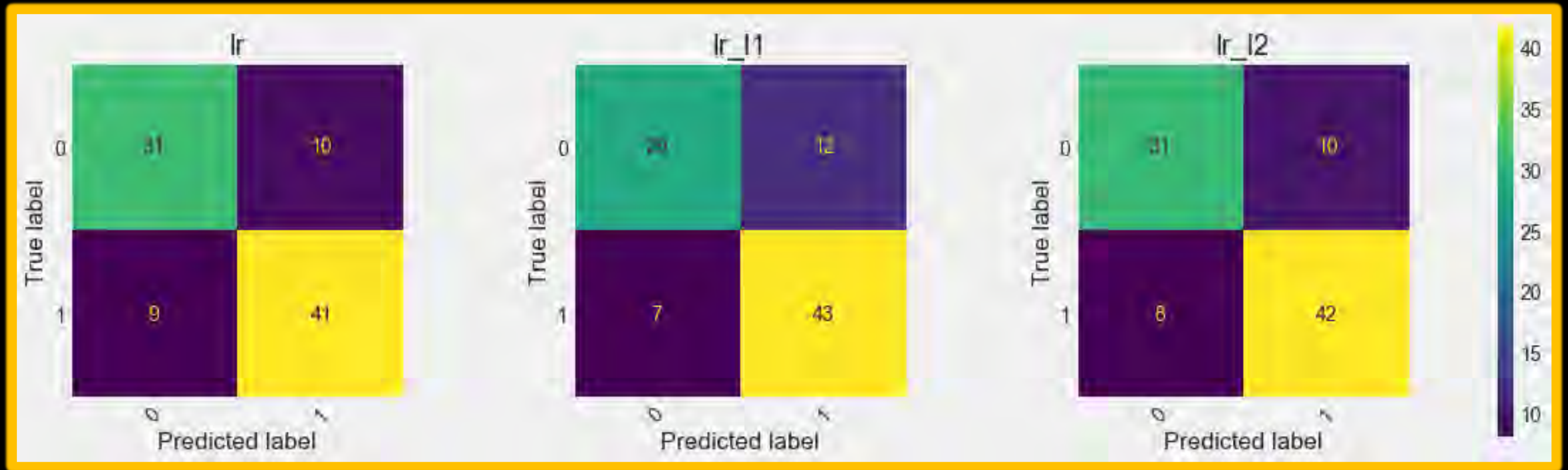
	0	1	accuracy	macro avg	weighted avg
precision	0.81	0.78	0.79	0.79	0.79
recall	0.71	0.86	0.79	0.78	0.79
f1-score	0.75	0.82	0.79	0.79	0.79
support	41.00	50.00	0.79	91.00	91.00

# Logistic Regression with L2 penalty

```
# L2-regularized logistic regression
lr_l2 = LogisticRegressionCV(Cs=10, cv=4, penalty='l2', solver='liblinear').fit(X_train, y_train)
y_pred_2 = lr_l2.predict(X_test)
clf_report = pd.DataFrame(classification_report(y_test, y_pred_2, output_dict=True))
clf_report
```

	0	1	accuracy	macro avg	weighted avg
precision	0.79	0.81	0.80	0.80	0.80
recall	0.76	0.84	0.80	0.80	0.80
f1-score	0.77	0.82	0.80	0.80	0.80
support	41.00	50.00	0.80	91.00	91.00

# Logistic Regression Results



The best model in terms of prediction performance is Logistic Regression with L2 penalty:

- Accuracy : 80%.
- Precision : 80%.
- Recall : 80%.
- F1 Score : 80%.
- Support : 91%.

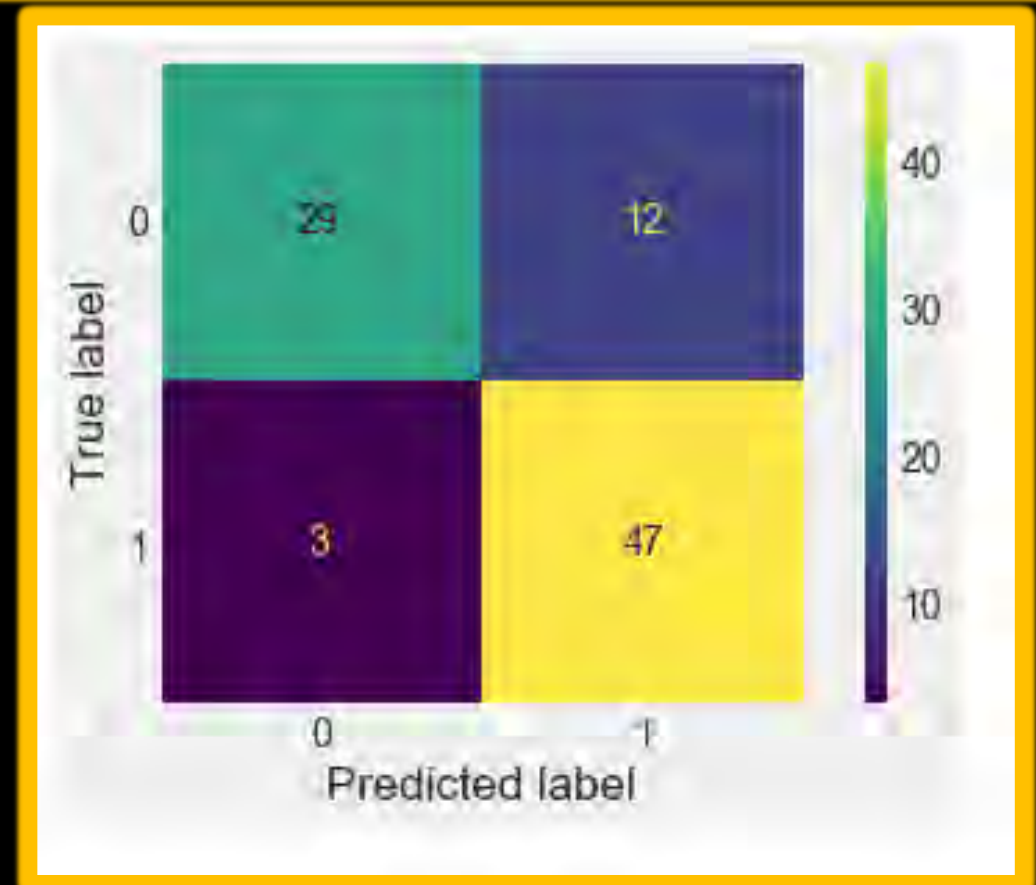
# Using KNN Algorithm

```
knn = KNeighborsClassifier(n_neighbors=25, weights='distance')  
knn = knn.fit(X_train, y_train)  
y_pred = knn.predict(X_test)
```

```
KNN_report = pd.DataFrame(classification_report(y_test, y_pred, output_dict=True))  
KNN_report
```

	0	1	accuracy	macro avg	weighted avg
precision	0.91	0.80	0.84	0.85	0.85
recall	0.71	0.94	0.84	0.82	0.84
f1-score	0.79	0.86	0.84	0.83	0.83
support	41.00	50.00	0.84	91.00	91.00

- ❑ Accuracy : 84%. ❑ Precision : 85%. ❑ Recall : 84%.
- ❑ F1 Score : 83%. ❑ Support : 91%.



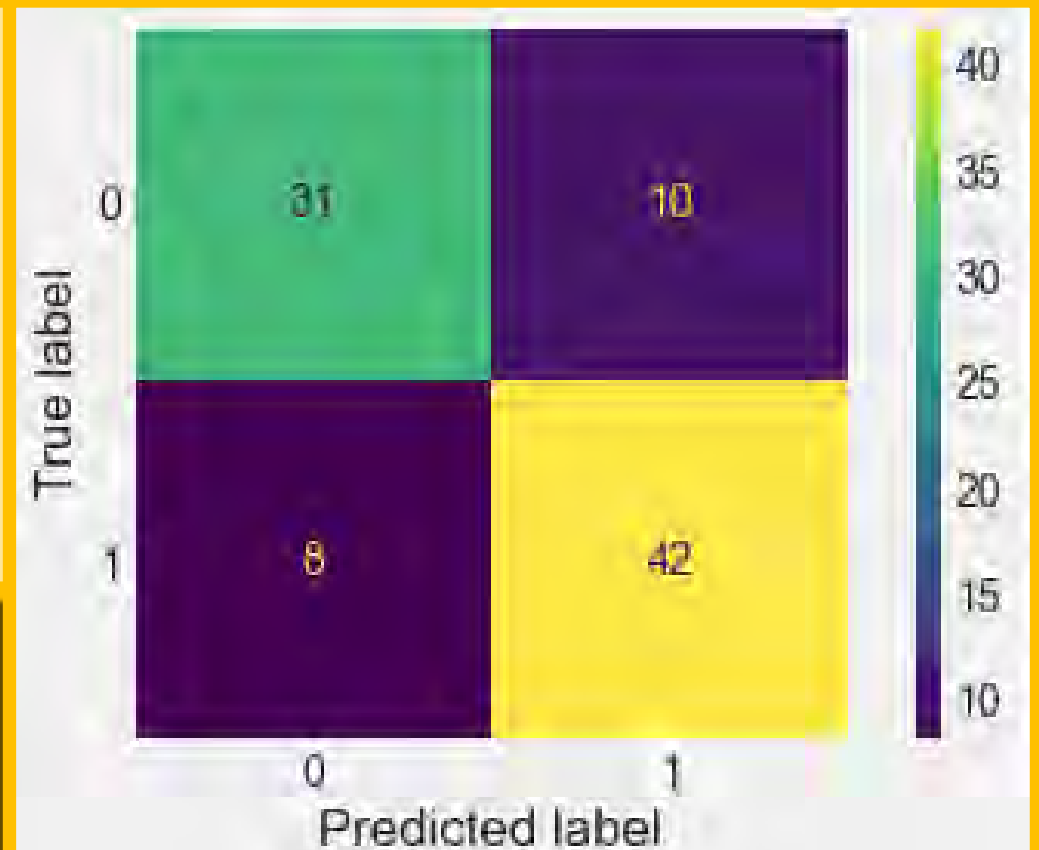
# Using Support Vector Machine

```
from sklearn.svm import SVC

kwargs = {'kernel': 'rbf'}
svc = SVC(**kwargs)

SVC_cl = svc.fit(X_train, y_train)
y_pred = SVC_cl.predict(X_test)
SVC_cl_report = pd.DataFrame(classification_report(y_test, y_pred, output_dict=True))
SVC_cl_report
```

	0	1	accuracy	macro avg	weighted avg
precision	0.79	0.81	0.80	0.80	0.80
recall	0.76	0.84	0.80	0.80	0.80
f1-score	0.77	0.82	0.80	0.80	0.80
support	41.00	50.00	0.80	91.00	91.00



- ❑ Accuracy : 80%.
- ❑ Precision : 80%.
- ❑ Recall : 80%.
- ❑ F1 Score : 80%.
- ❑ Support : 91%.



# Using XGBoost Algorithm

```
import xgboost as xgb
from sklearn.model_selection import GridSearchCV

param_grid = {
    "max_depth": [5],
    "learning_rate": [0.05],
    "gamma": [0, 0.25, 1, 10],
    "reg_lambda": [0],
    "scale_pos_weight": [1, 3, 5, 7, 10],
    "subsample": [0.1, 0.2, 0.3, 0.4, 0.5, 0.8],
    "colsample_bytree": [0.5, 0.7],
}

# Init classifier
xgb_cl = xgb.XGBClassifier(objective="binary:logistic")

# Init Grid Search
grid_cv = GridSearchCV(xgb_cl, param_grid, n_jobs=-1, cv=3, scoring="roc_auc")

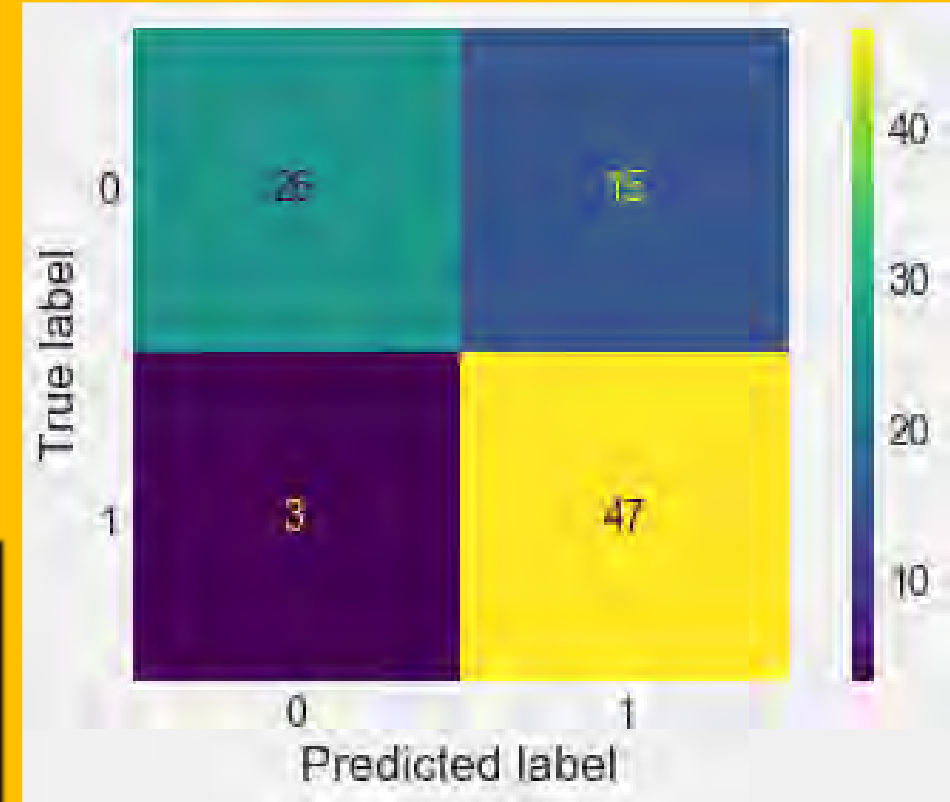
# Fit
_ = grid_cv.fit(X_train, y_train)
```

# Using XGBoost Algorithm

```
final_xgb_cl = xgb.XGBClassifier(  
    **grid_cv.best_params_,  
    objective="binary:logistic",  
)  
  
_ = final_xgb_cl.fit(X_train, y_train)  
  
y_pred = final_xgb_cl.predict(X_test)  
  
xgb_report = pd.DataFrame(classification_report(y_test, y_pred, output_dict=True))
```

	0	1	accuracy	macro avg	weighted avg
precision	0.90	0.76	0.80	0.83	0.82
recall	0.63	0.94	0.80	0.79	0.80
f1-score	0.74	0.84	0.80	0.79	0.80
support	41.00	50.00	0.80	91.00	91.00

- ❑ Accuracy : 80%.    ❑ Precision : 82%.    ❑ Recall : 80%.
- ❑ F1 Score : 80%.    ❑ Support : 91%.



# Models Comparison

	0	1	accuracy	macro avg	weighted avg
precision	0.91	0.80	0.84	0.85	0.85
recall	0.71	0.94	0.84	0.82	0.84
f1-score	0.79	0.86	0.84	0.83	0.83
support	41.00	50.00	0.84	91.00	91.00

As can be seen in the table above, the following models performed according to the descending order

1. KNN
2. XGBoost
3. Logistic Regression with L2
4. Support Vector Machine

Clearly the winner is:



**KNN**

# The pitfalls & advantages of using each model

## ❖ Logistic Regression

Provides high predictive results and it is the simplest and fastest Model in terms of parameters and training.

## ❖ KNN

it is providing the best results, but it is slower in terms of prediction process because it requires to calculate the distance between all the points in the dataset to classify every single point.

# The pitfalls & advantages of using each model

## ❖ XGBoost

- ❑ Performs was very good as well but
- ❑ In contrast of KNN it takes longer time in the training process since we used grid search technique to search about best fitting parameters,
- ❑ In the end it is a tradeoff because in bigger dataset the performance will be higher with this models, but the training process will take a longer time.