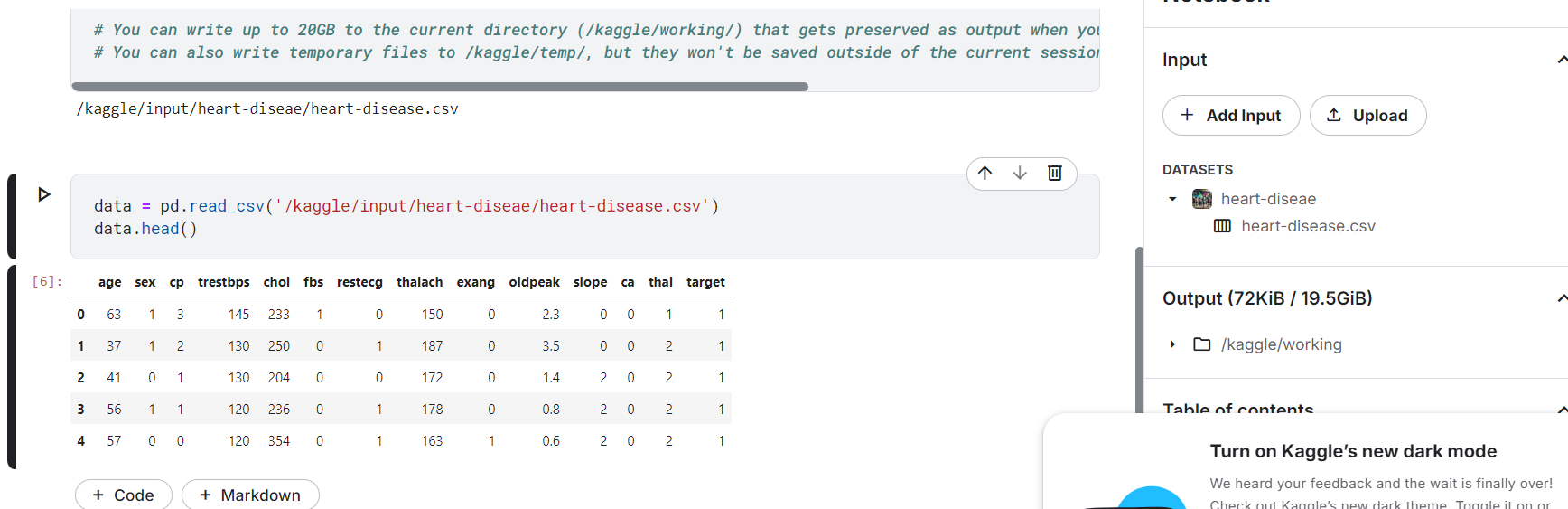
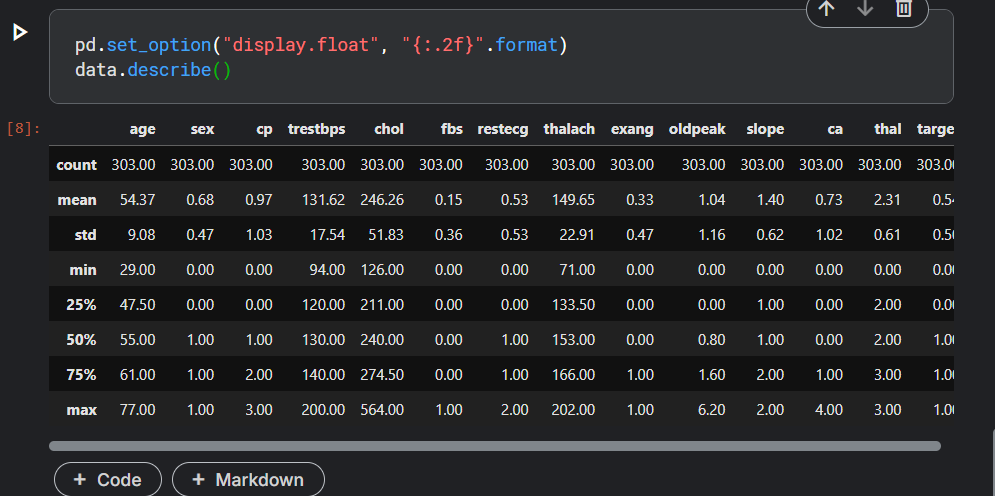
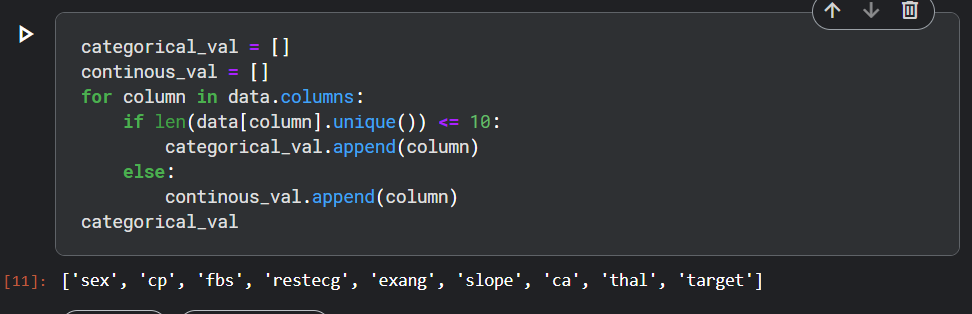
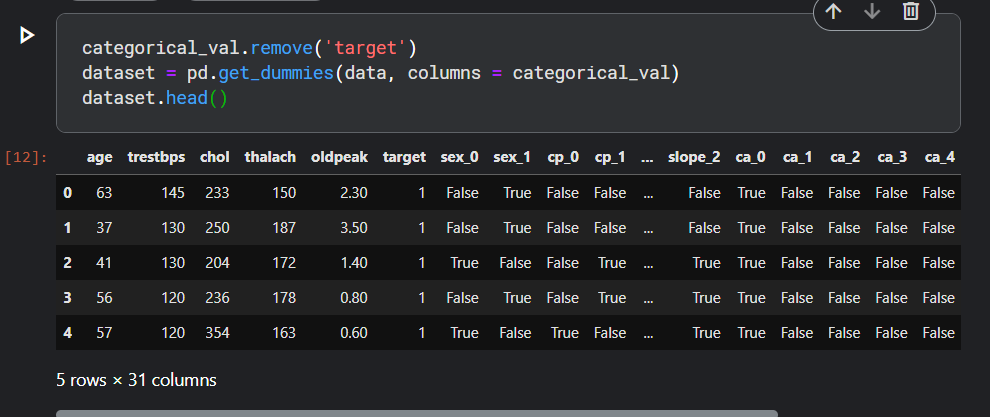
1) Kaggle dataset -> new notebook

2) read data using the path of dataset and display first few records

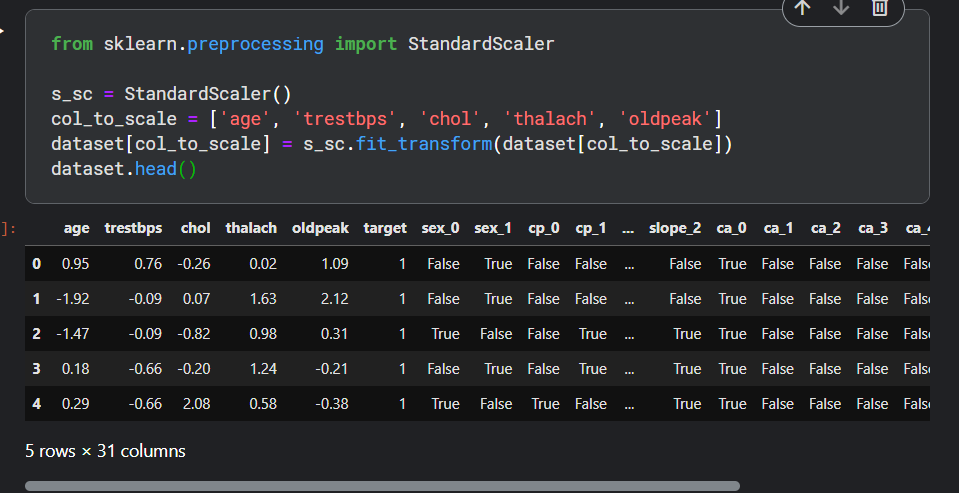
3) display floating-point numbers with two decimal places, making the summary easier to read and more consistent in appearance

4) Separate categorial and continuous columns

Data processing

1) Convert categorical data to dummy variables. Use the get\_dummies method to create dummy columns for categorical variables.

2) Standardise columns –> mean = 0, std deviation = 1



3) Function to evaluate the model’s performance and split dataset

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

def print\_score(clf, X\_train, y\_train, X\_test, y\_test, train=True):

if train:

pred = clf.predict(X\_train)

clf\_report = pd.DataFrame(classification\_report(y\_train, pred, output\_dict=True))

print("Train Result:\n================================================")

print(f"Accuracy Score: {accuracy\_score(y\_train, pred) \* 100:.2f}%")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print(f"CLASSIFICATION REPORT:\n{clf\_report}")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print(f"Confusion Matrix: \n {confusion\_matrix(y\_train, pred)}\n")

elif train==False:

pred = clf.predict(X\_test)

clf\_report = pd.DataFrame(classification\_report(y\_test, pred, output\_dict=True))

print("Test Result:\n================================================")

print(f"Accuracy Score: {accuracy\_score(y\_test, pred) \* 100:.2f}%")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print(f"CLASSIFICATION REPORT:\n{clf\_report}")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print(f"Confusion Matrix: \n {confusion\_matrix(y\_test, pred)}\n")

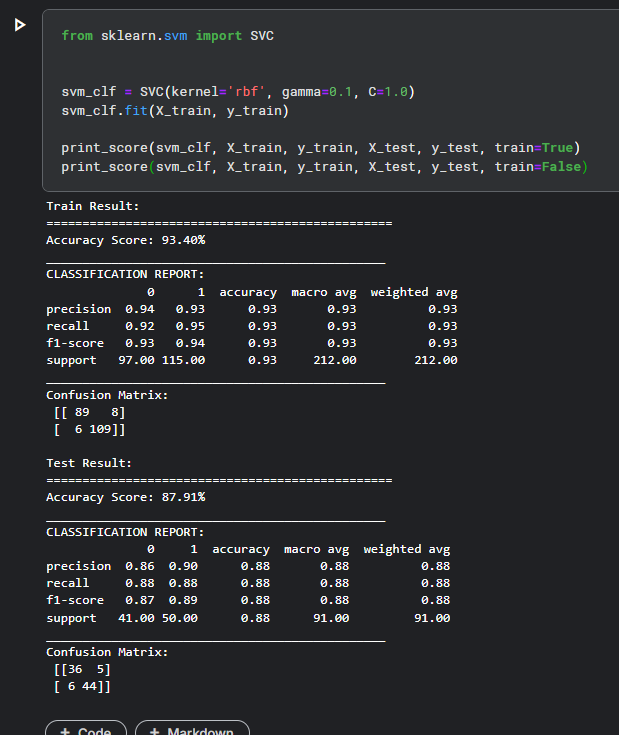
from sklearn.model\_selection import train\_test\_split

X = dataset.drop('target', axis=1)

y = dataset.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

Support Vector Machine



Linear regression

1) View data as plot