1. **Can you get the feature names and the descriptions of the toy diabetes data set via python code? Please write down the code and the output of the code. [1 point]**

**Code:**

from sklearn.datasets import load\_diabetes

data = load\_diabetes()

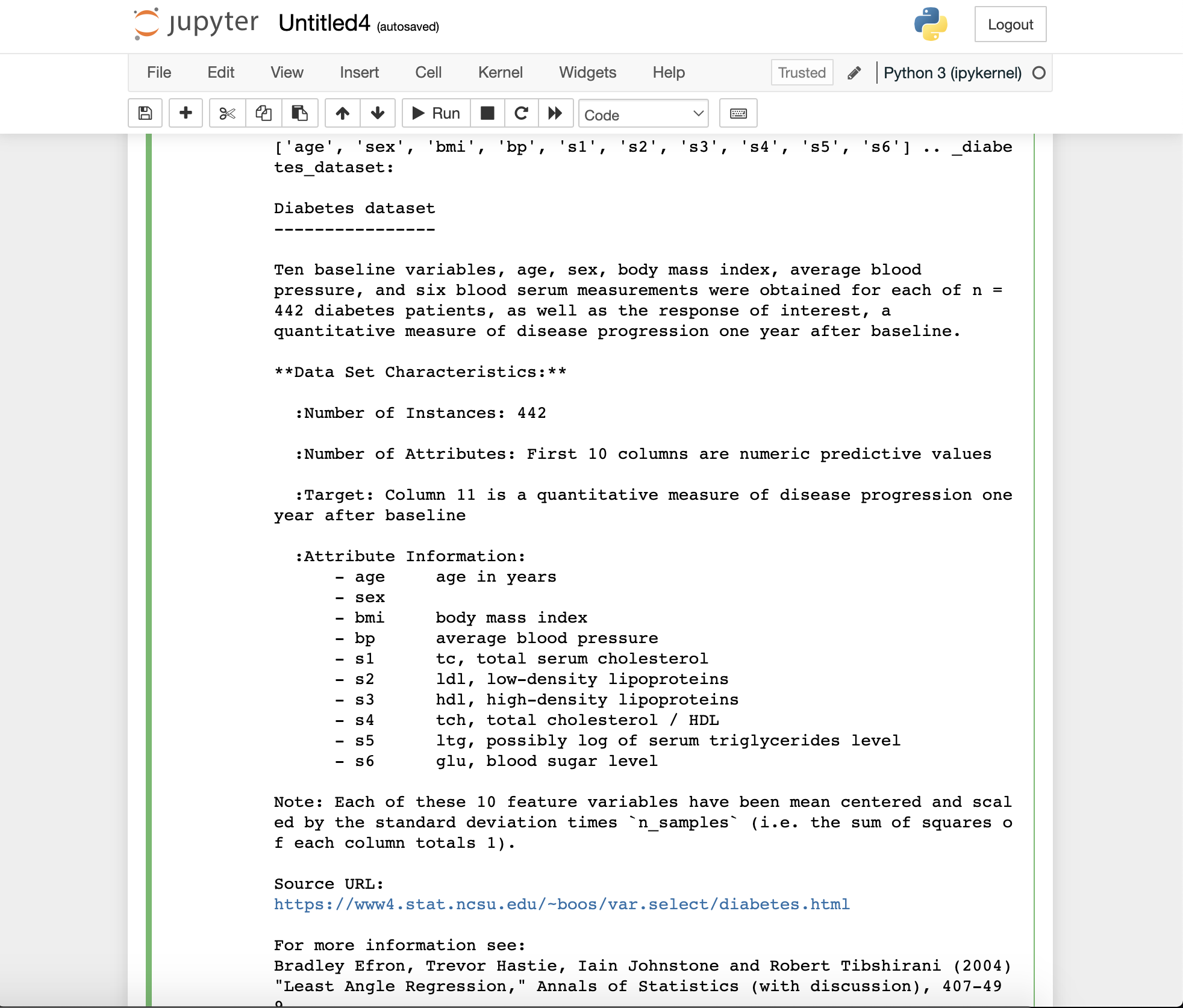
features = data.feature\_names

description = data.DESCR

print(features, description)

Reference: [https://scikit-](https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_diabetes.html#sklearn.datasets.load_diabetes) learn.org/stable/modules/generated/sklearn.datasets.load\_diabetes.html#sklearn.datasets.load\_diabetes

**Output:**



1. **It is time to play with another data set of scikit\_learn: the wine data set. Use load\_wine() to load the wine data set and apply logistic regression on it. We split the data set into 80/20. 80% of data points are used for model training, and 20% for test purposes. Please write the code and output (including mean accuracy and confusion matrix) in the answer. [2 points]**

**Code:**

from sklearn.datasets import load\_wine

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

import numpy as np

from sklearn.linear\_model import LogisticRegression

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

data\_bunch = load\_wine()

print(data\_bunch.feature\_names)

print(data\_bunch.target\_names)

print()

wine\_X, wine\_y = data\_bunch.data, data\_bunch.target

def filter\_class\_2(X, y):

new\_X, new\_y = [], []

for x, y in zip(X, y):

if y == 2:

continue

else:

new\_X.append(x)

new\_y.append(y)

return np.array(new\_X), np.array(new\_y)

wine\_X, wine\_y = filter\_class\_2(wine\_X, wine\_y)

# split the data into train/test (80/20)

wine\_X\_train, wine\_X\_test, wine\_y\_train, wine\_y\_test = train\_test\_split(wine\_X, wine\_y, test\_size=0.20, random\_state=15)

# logistic regression model

logistic\_regression\_model = LogisticRegression()

# train

logistic\_regression\_model.fit(wine\_X\_train, wine\_y\_train)

# prediction

wine\_y\_pred = logistic\_regression\_model.predict(wine\_X\_test)

accuracy = accuracy\_score(wine\_y\_test, wine\_y\_pred)

confusion\_matrix = confusion\_matrix(wine\_y\_test, wine\_y\_pred)

print("accuracy:", accuracy \* 100)

print("confusion\_matrix:", confusion\_matrix)

**Output:**

