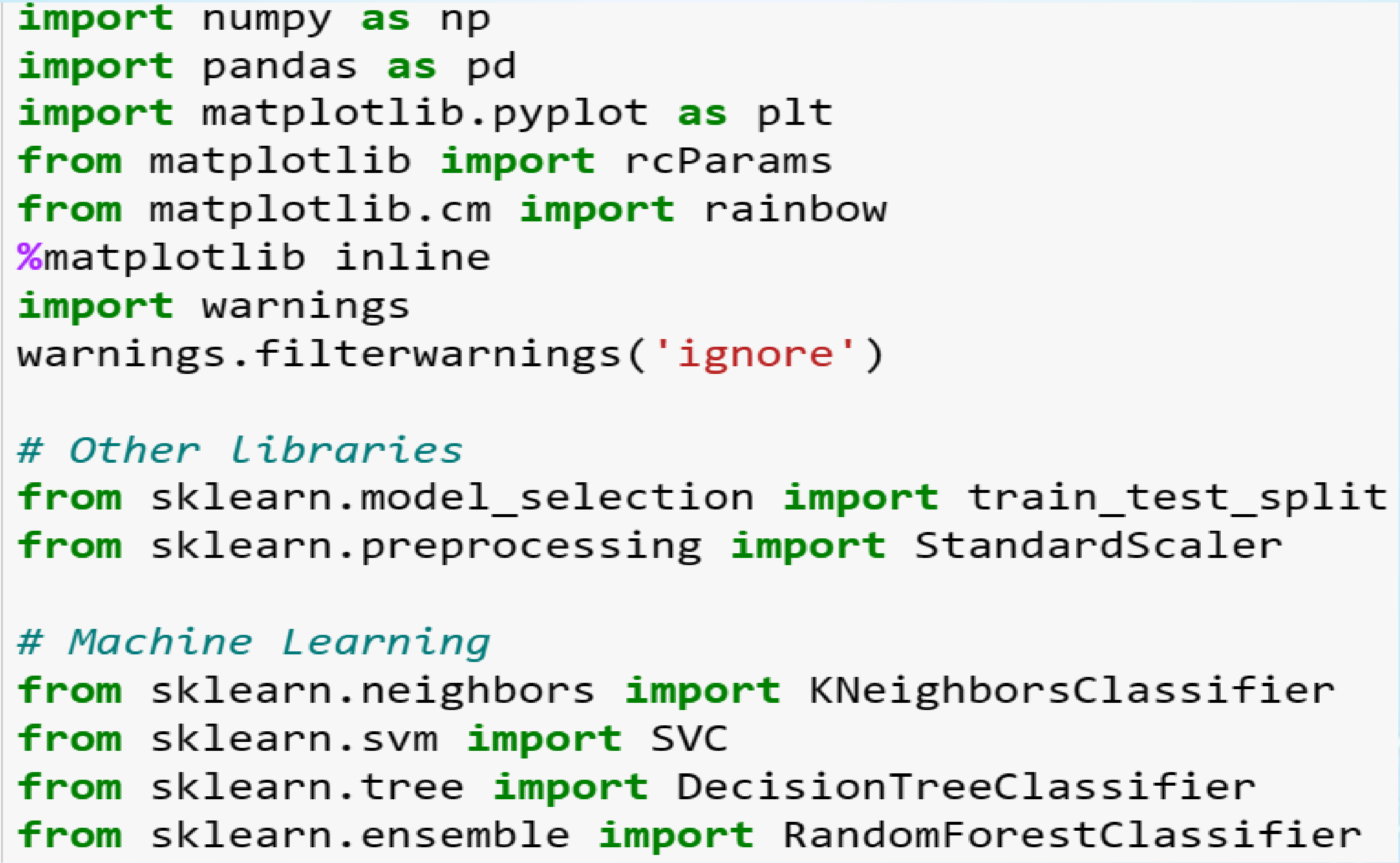
SECOND PRESENTATION IS ON WHAT ARE THE THINGS I USE IN MY PREJECT

imported several libraries for the project:



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

# Other Libraries
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Machine Learning
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

- **Matplotlib:**To create charts using pyplot, define parameters using rcParams and color them with cm.rainbow

- **StandardScaler:** To scale all the features, so that the Machine Learning model better adapts to the dataset

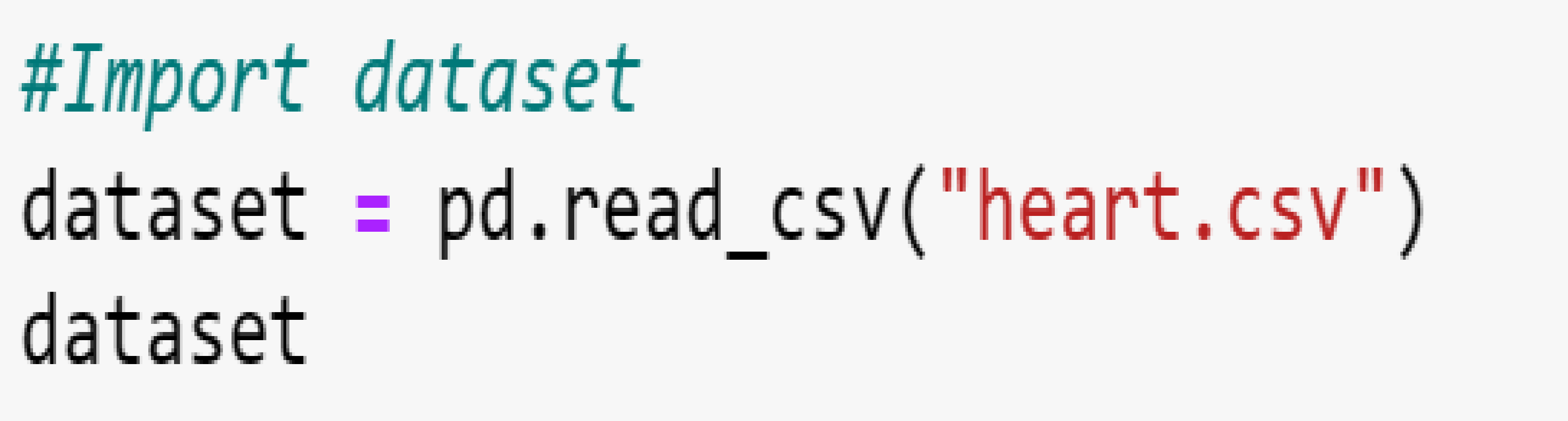
- numpy: To work with arrays

- pandas: To work with csv files and dataframes

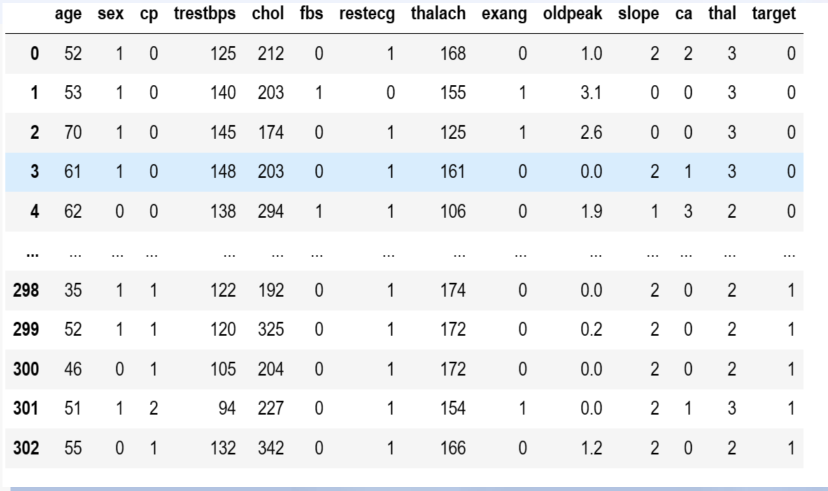
- train_test_split: To split the dataset into training and testing data

IMPORT DATASET

- ❑ After downloading the dataset from Kaggle, I saved it to my working directory with the name dataset.csv. Next, I used read_csv() to read the dataset and save it to the dataset variable.



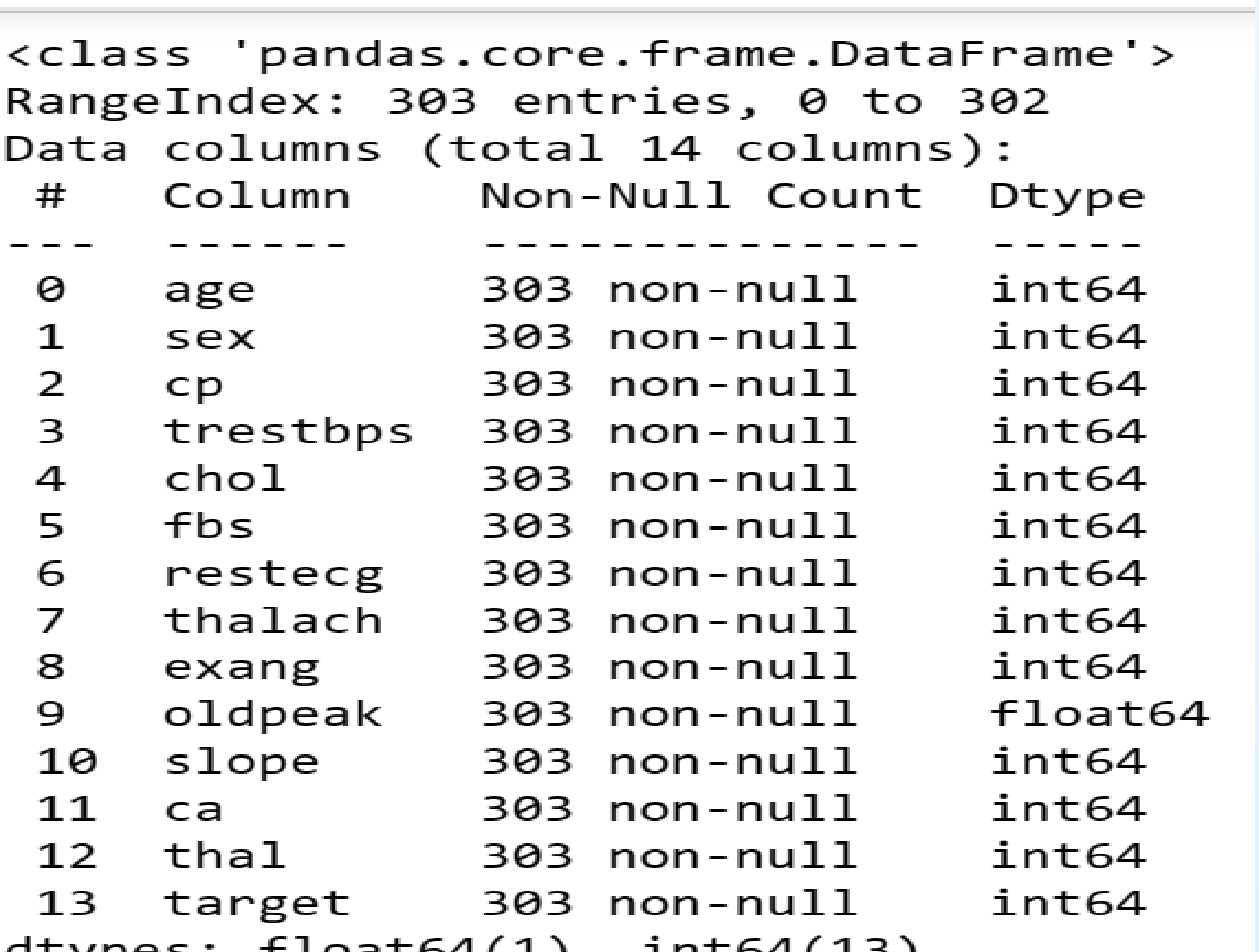
```
#Import dataset
dataset = pd.read_csv("heart.csv")
dataset
```



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	35	1	1	122	192	0	1	174	0	0.0	2	0	2	1
299	52	1	1	120	325	0	1	172	0	0.2	2	0	2	1
300	46	0	1	105	204	0	1	172	0	0.0	2	0	2	1
301	51	1	2	94	227	0	1	154	1	0.0	2	1	3	1
302	55	0	1	132	342	0	1	166	0	1.2	2	0	2	1

Before any analysis, I just wanted to take a look at the data. So, I used the info() method.

INFO() METHOD

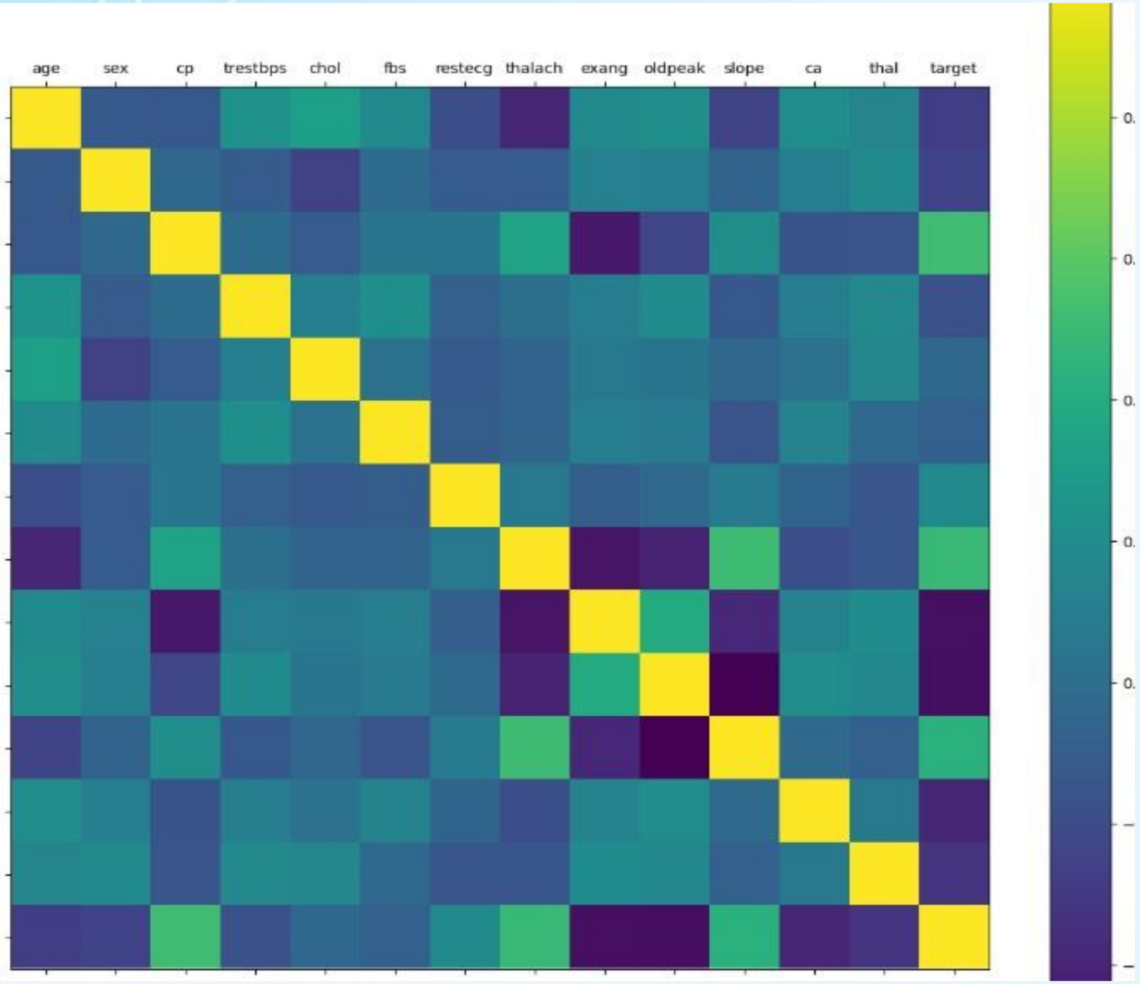


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  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)
```

As you can see from the output above, there are a total of 13 features and 1 target variable. Also, there are no missing values so we don't need to take care of any null values.

Understanding the data

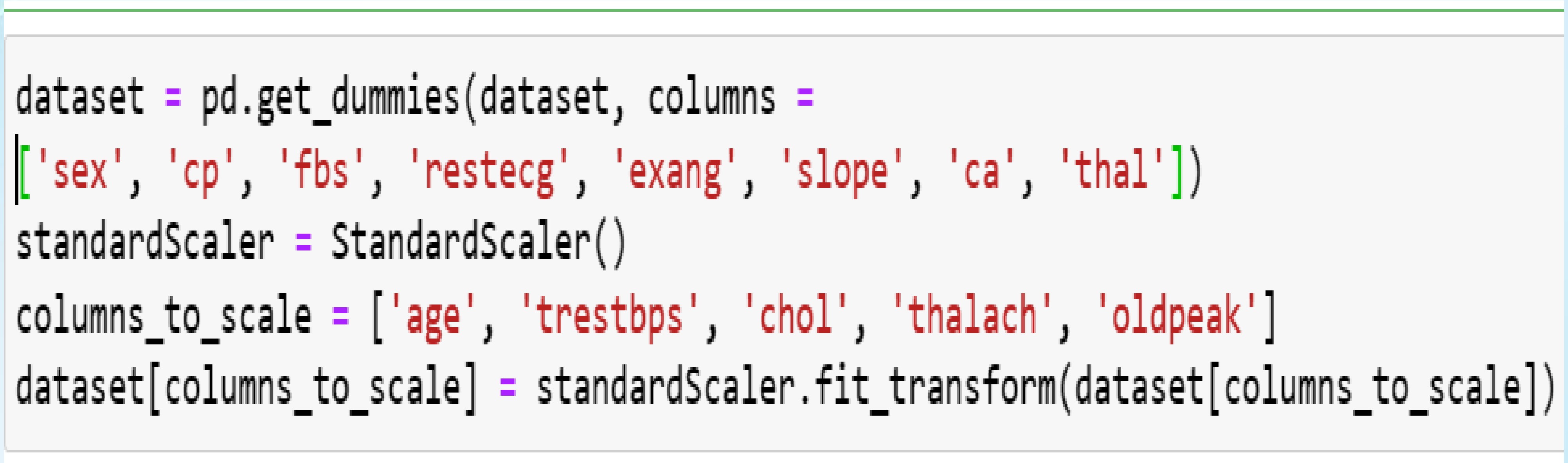
Correlation Matrix



It’s easy to see that there is no single feature that has a very high correlation with our target value. Also, some of the features have a negative correlation with the target value and some have positive

Data Processing

To work with categorical variables, we should break each categorical column into dummy columns with 1s and 0s.

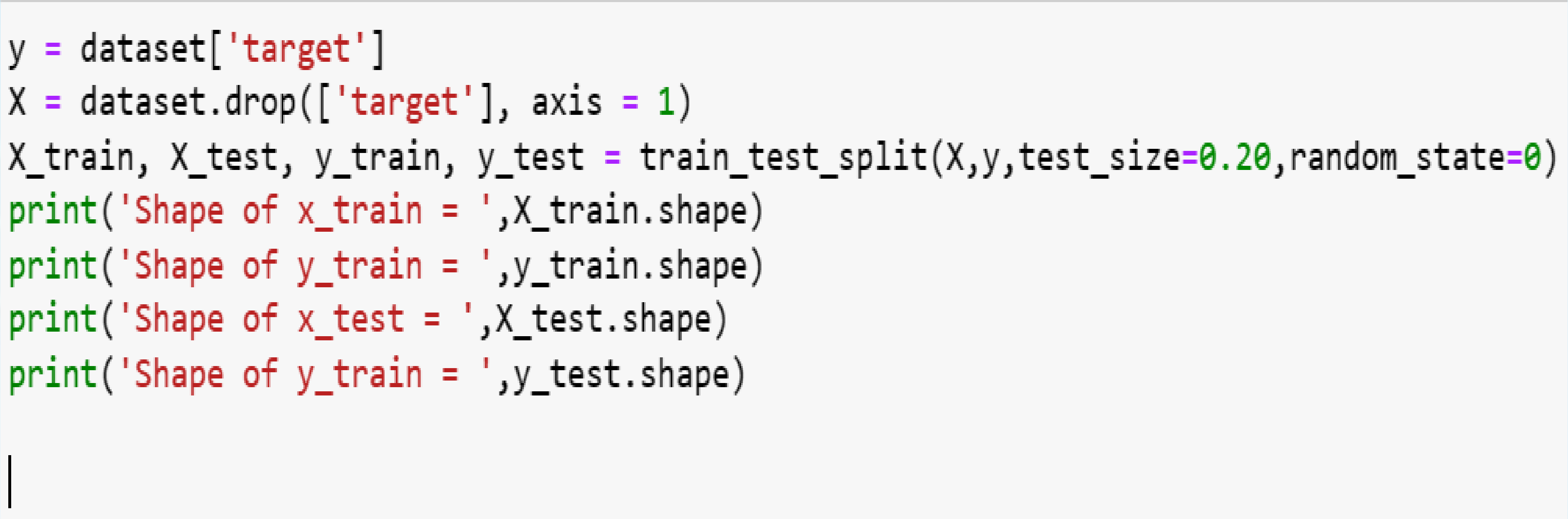


```
dataset = pd.get_dummies(dataset, columns =
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal'])
standardScaler = StandardScaler()
columns_to_scale = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
dataset[columns_to_scale] = standardScaler.fit_transform(dataset[columns_to_scale])
```

- The dataset is now ready. We can begin with training our models.

TRAINING AND TESTING THE DATA

I split the dataset into 80% training data and 20% testing data.



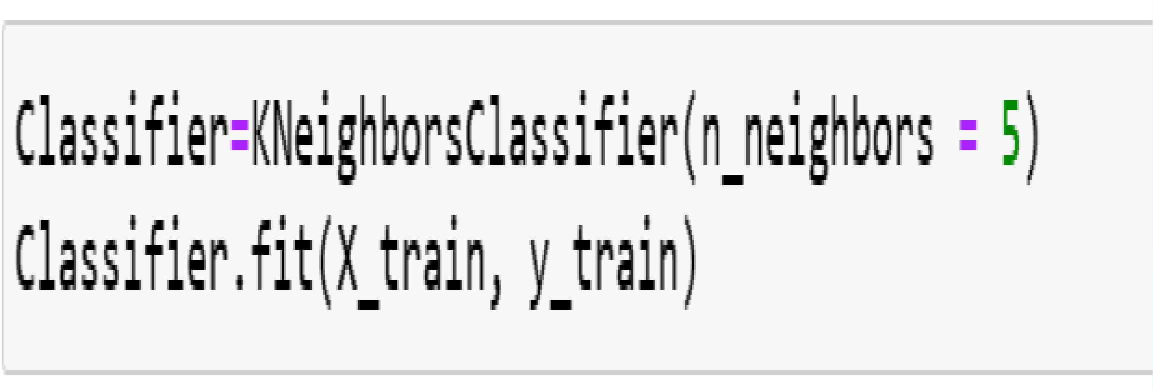
```
y = dataset['target']
X = dataset.drop(['target'], axis = 1)
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20,random_state=0)
print('Shape of x_train = ',X_train.shape)
print('Shape of y_train = ',y_train.shape)
print('Shape of x_test = ',X_test.shape)
print('Shape of y_train = ',y_test.shape)
```

In this project, I took 4 algorithms and varied their various parameters and compared the final models.

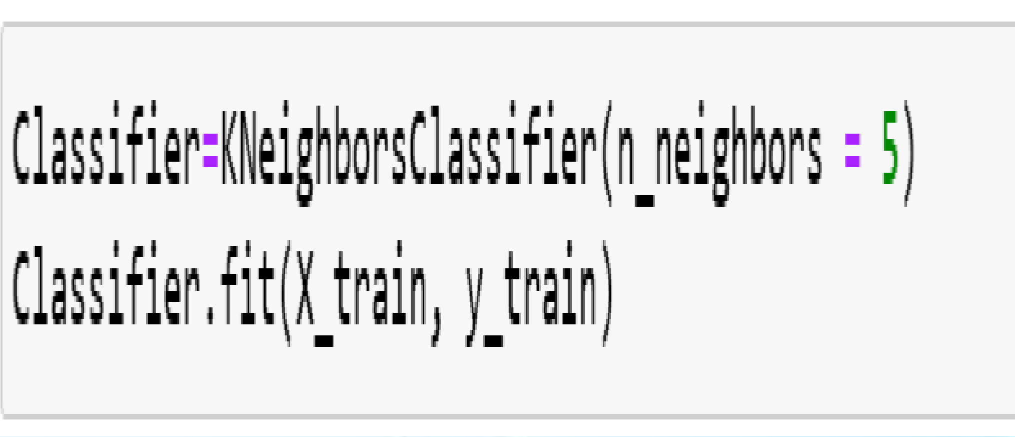
WHICH ARE THE ALGORITHM I USED

I've used a variety of Machine Learning algorithms, implemented in Python, to predict the presence of heart disease in a patient. This is a classification problem, with input features as a variety of parameters, and the target variable as a binary variable, predicting whether heart disease is present or not.

FIRST ONE I UESED KNEAREST-Neighbors

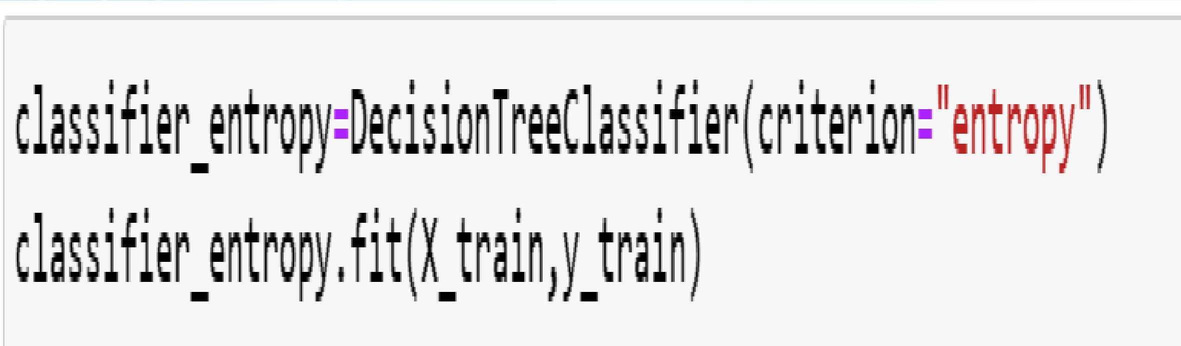


```
Classifier=KNeighborsClassifier(n_neighbors = 5)
Classifier.fit(X_train, y_train)
```

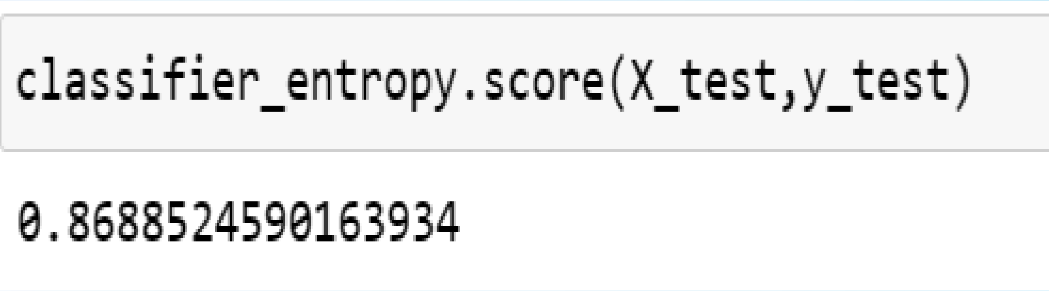


```
Classifier=KNeighborsClassifier(n_neighbors = 5)
Classifier.fit(X_train, y_train)
```

SECOND I UESED IS DECISIONTREE CLASSIFIER



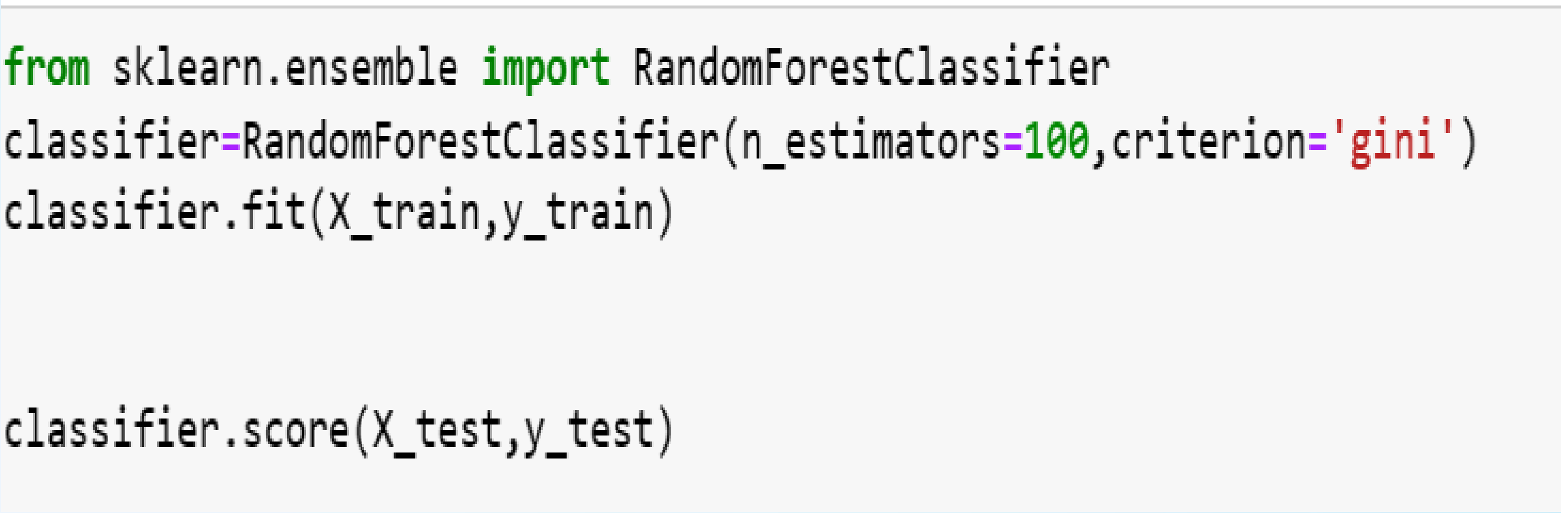
```
classifier_entropy=DecisionTreeClassifier(criterion="entropy")
classifier_entropy.fit(X_train,y_train)
```



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

```
0.8688524590163934
```

THIRD I UESED IS RANDOMFORESTCLASSIFIER



```
from sklearn.ensemble import RandomForestClassifier
classifier=RandomForestClassifier(n_estimators=100,criterion='gini')
classifier.fit(X_train,y_train)



classifier.score(X_test,y_test)
```

CONCLUSION

❑ The project involved analysis of the heart disease patient dataset with proper data processing. Then, 3 models were trained and tested with maximum scores as follows:

1. **K Neighbors Classifier: 80%**
2. **Decision Tree Classifier: 86%**
3. **Random Forest Classifier: 91%**