C02_FinalProject_Heart

April 14, 2023

1 Predicting Heart Disease Using Clinical Variables

https://www.kaggle.com/datasets/thedevastator/predicting-heart-disease-risk-using-clinical-var

The Heart Disease Prediction dataset provides vital insight in the relationship between risk factors and cardiac health. This dataset contains 270 case studies of individuals classified as either having or not having heart disease based on results from cardiac catheterizations - the gold standard in heart health assessment. Each patient is identified by 13 independent predictive variables revealing their age, sex, chest pain type, blood pressure measurements, cholesterol levels, electrocardiogram results, exercise-induced angina symptoms, and the number of vessels seen on fluoroscopy showing narrowing of their coronary arteries. Provided with this data set is an opportunity to evaluate how these characteristics interact with each other in order to determine an individual's level of risk for developing cardiovascular problems that lead to heart failure or stroke. With this knowledge we can create preventative strategies beyond what traditional medical treatment can do by identifying those at risk earlier and aid our healthcare professionals in treating them better. By analyzing a combination of clinical variables explained here, we have a powerful tool at our fingertips to try and combat cardiovascular illness before it even has the chance to take root!

Column name Description

- Age The age of the patient. (Numeric)
- Sex The gender of the patient. (Categorical)
- Chest pain type The type of chest pain experienced by the patient. (Categorical)
- BP The blood pressure level of the patient. (Numeric)
- Cholesterol The cholesterol level of the patient. (Numeric)
- FBS over 120 The fasting blood sugar test results over 120 mg/dl. (Numeric)
- EKG results The electrocardiogram results of the patient. (Categorical)
- Max HR The maximum heart rate levels achieved during exercise testing. (Numeric)
- Exercise angina The angina experienced during exercise testing. (Categorical)
- ST depression The ST depression on an Electrocardiogram. (Numeric)
- Slope of ST The slope of ST segment electrocardiogram readings. (Categorical)
- Number of vessels fluro The amount vessels seen in Fluoroscopy images. (Numeric)
- Thallium The Thallium Stress test findings. (Categorical)

- Heart Disease Whether or not the patient has been diagnosed with Heart Disease. (Categorical)

1.1 Objectives

The objectives of this exercise are: - Explore the dataset - Visualize the data - Compare standard Logistic Regression with Neural Network

1.2 Imports

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import zipfile
import os
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪ConfusionMatrixDisplay
from sklearn.preprocessing import Normalizer, MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import  Dense, Input, InputLayer
import seaborn as sns
```

1.3 Load the Data

[2]:
```
df = pd.read_csv('archive.zip', index_col='index')
```

[3]:
```
df.head()
```

[3]:
```
       Age  Sex  Chest pain type   BP  Cholesterol  FBS over 120  EKG results  \
index
0       70    1                4  130          322             0            2
1       67    0                3  115          564             0            2
2       57    1                2  124          261             0            0
3       64    1                4  128          263             0            0
4       74    0                2  120          269             0            2

       Max HR  Exercise angina  ST depression  Slope of ST  \
index
0         109                0            2.4            2
1         160                0            1.6            2
2         141                0            0.3            1
3         105                1            0.2            2
4         121                1            0.2            1

       Number of vessels fluro  Thallium Heart Disease
index
```

```
0                          3         3      Presence
1                          0         7       Absence
2                          0         7      Presence
3                          1         7       Absence
4                          1         3       Absence
```

[4]:
```
df.replace({'Heart Disease': {'Presence': 1, 'Absence': 0}}, inplace=True)
df['Heart Disease'] = df['Heart Disease'].astype(int)
df.head()
```

[4]:
```
        Age  Sex  Chest pain type   BP  Cholesterol  FBS over 120  EKG results  \
index
0        70    1                4  130          322             0            2
1        67    0                3  115          564             0            2
2        57    1                2  124          261             0            0
3        64    1                4  128          263             0            0
4        74    0                2  120          269             0            2

        Max HR  Exercise angina  ST depression  Slope of ST  \
index
0          109                0            2.4            2
1          160                0            1.6            2
2          141                0            0.3            1
3          105                1            0.2            2
4          121                1            0.2            1

        Number of vessels fluro  Thallium  Heart Disease
index
0                             3         3              1
1                             0         7              0
2                             0         7              1
3                             1         7              0
4                             1         3              0
```

[5]:
```
df.dtypes
```

[5]:
```
Age                       int64
Sex                       int64
Chest pain type           int64
BP                        int64
Cholesterol               int64
FBS over 120              int64
EKG results               int64
Max HR                    int64
Exercise angina           int64
ST depression           float64
Slope of ST               int64
```

```
Number of vessels fluro      int64
Thallium                     int64
Heart Disease                int32
dtype: object
```

[6]:
```
df.isna().any(axis=0)
```

[6]:
```
Age                          False
Sex                          False
Chest pain type              False
BP                           False
Cholesterol                  False
FBS over 120                 False
EKG results                  False
Max HR                       False
Exercise angina              False
ST depression                False
Slope of ST                  False
Number of vessels fluro      False
Thallium                     False
Heart Disease                False
dtype: bool
```

[7]:
```
df.describe()
```

[7]:

	Age	Sex	Chest pain type	BP	Cholesterol
count	270.000000	270.000000	270.000000	270.000000	270.000000
mean	54.433333	0.677778	3.174074	131.344444	249.659259
std	9.109067	0.468195	0.950090	17.861608	51.686237
min	29.000000	0.000000	1.000000	94.000000	126.000000
25%	48.000000	0.000000	3.000000	120.000000	213.000000
50%	55.000000	1.000000	3.000000	130.000000	245.000000
75%	61.000000	1.000000	4.000000	140.000000	280.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000

	FBS over 120	EKG results	Max HR	Exercise angina	ST depression
count	270.000000	270.000000	270.000000	270.000000	270.00000
mean	0.148148	1.022222	149.677778	0.329630	1.05000
std	0.355906	0.997891	23.165717	0.470952	1.14521
min	0.000000	0.000000	71.000000	0.000000	0.00000
25%	0.000000	0.000000	133.000000	0.000000	0.00000
50%	0.000000	2.000000	153.500000	0.000000	0.80000
75%	0.000000	2.000000	166.000000	1.000000	1.60000
max	1.000000	2.000000	202.000000	1.000000	6.20000

	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
count	270.000000	270.000000	270.000000	270.000000

```
mean      1.585185                0.670370    4.696296       0.444444
std       0.614390                0.943896    1.940659       0.497827
min       1.000000                0.000000    3.000000       0.000000
25%       1.000000                0.000000    3.000000       0.000000
50%       2.000000                0.000000    3.000000       0.000000
75%       2.000000                1.000000    7.000000       1.000000
max       3.000000                3.000000    7.000000       1.000000
```

1.4 Data Cleanup

We will keep only the numeric data.

[8]:
```
numCols = ['Age', 'BP', 'Cholesterol', 'FBS over 120', 'Max HR', 'ST␣
 ↪depression', 'Number of vessels fluro']
allCols = ['Age', 'BP', 'Cholesterol', 'FBS over 120', 'Max HR', 'ST␣
 ↪depression', 'Number of vessels fluro', 'Heart Disease']
```

[9]:
```
df = df[allCols]
```

[10]:
```
df.head()
```

[10]:
```
       Age   BP  Cholesterol  FBS over 120  Max HR  ST depression  \
index
0       70  130          322             0     109            2.4
1       67  115          564             0     160            1.6
2       57  124          261             0     141            0.3
3       64  128          263             0     105            0.2
4       74  120          269             0     121            0.2

       Number of vessels fluro  Heart Disease
index
0                            3              1
1                            0              0
2                            0              1
3                            1              0
4                            1              0
```

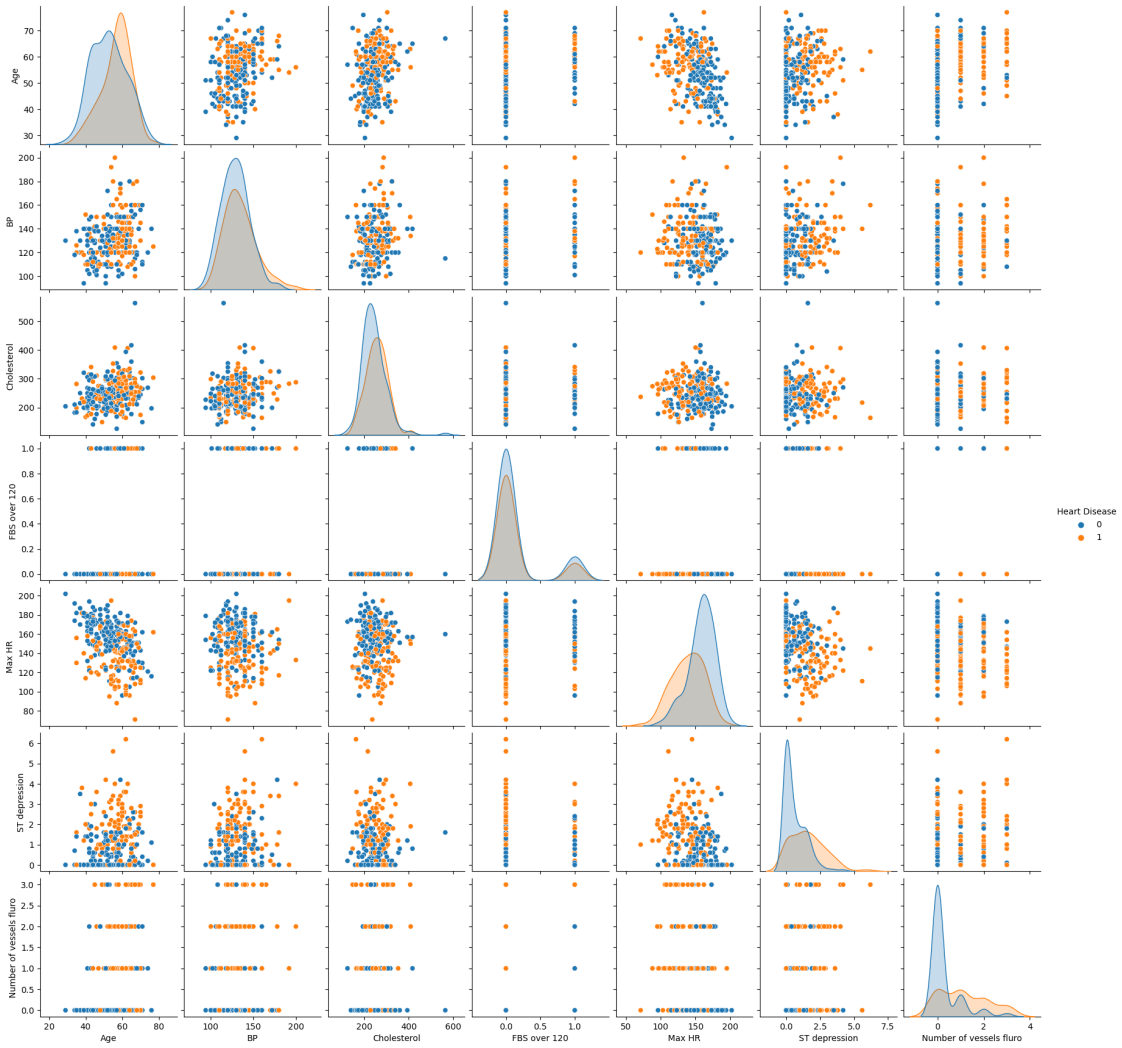
[11]:
```
df.describe()
```

[11]:
```
              Age          BP  Cholesterol  FBS over 120      Max HR  \
count  270.000000  270.000000   270.000000    270.000000  270.000000
mean    54.433333  131.344444   249.659259      0.148148  149.677778
std      9.109067   17.861608    51.686237      0.355906   23.165717
min     29.000000   94.000000   126.000000      0.000000   71.000000
25%     48.000000  120.000000   213.000000      0.000000  133.000000
50%     55.000000  130.000000   245.000000      0.000000  153.500000
75%     61.000000  140.000000   280.000000      0.000000  166.000000
max     77.000000  200.000000   564.000000      1.000000  202.000000
```

	ST depression	Number of vessels fluro	Heart Disease
count	270.00000	270.000000	270.000000
mean	1.05000	0.670370	0.444444
std	1.14521	0.943896	0.497827
min	0.00000	0.000000	0.000000
25%	0.00000	0.000000	0.000000
50%	0.80000	0.000000	0.000000
75%	1.60000	1.000000	1.000000
max	6.20000	3.000000	1.000000

[12]:
```
sns.pairplot(df, hue='Heart Disease')
```

[12]: <seaborn.axisgrid.PairGrid at 0x25de85121a0>



1.5 Train/Test Split and Normalization

[13]:
```
X = df.drop(['Heart Disease'], axis = 1)
y = df['Heart Disease']
```

[14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    train_size=0.7,
                                                    random_state=42)
```

[15]:
```
transformer = MinMaxScaler().fit(X_train)
```

[16]:
```
X_train_norm = transformer.transform(X_train)
X_test_norm = transformer.transform(X_test)

X_train_norm = pd.DataFrame(X_train_norm)
X_train_norm.columns = numCols

X_test_norm = pd.DataFrame(X_test_norm)
X_test_norm.columns = numCols
```

[17]:
```
(X_train_norm).describe()
```

[17]:

	Age	BP	Cholesterol	FBS over 120	Max HR	\
count	189.000000	189.000000	189.000000	189.000000	189.000000	
mean	0.572252	0.356694	0.261298	0.137566	0.537872	
std	0.202942	0.169361	0.124243	0.345359	0.200679	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.422222	0.245283	0.174941	0.000000	0.385965	
50%	0.600000	0.339623	0.243499	0.000000	0.578947	
75%	0.733333	0.433962	0.333333	0.000000	0.684211	
max	1.000000	1.000000	1.000000	1.000000	1.000000	

	ST depression	Number of vessels fluro
count	189.000000	189.000000
mean	0.186380	0.232804
std	0.192856	0.322345
min	0.000000	0.000000
25%	0.000000	0.000000
50%	0.161290	0.000000
75%	0.290323	0.333333
max	1.000000	1.000000

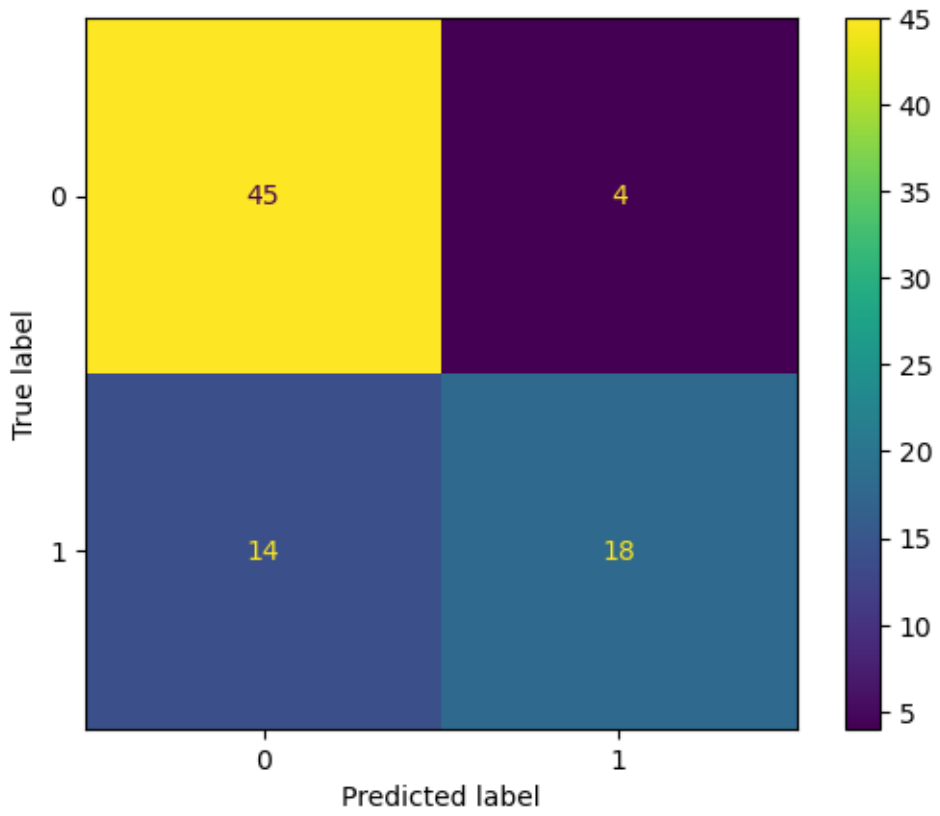
1.6 Logistic Regression

[18]:
```
lr = LogisticRegression()
lr.fit(X_train_norm, y_train)
y_pred_lr = lr.predict(X_test_norm)
lr_accuracy = accuracy_score(y_test, y_pred_lr)
```

```
lr_cm = confusion_matrix(y_test, y_pred_lr)
lr_cmd = ConfusionMatrixDisplay(lr_cm)
print(f"Accuracy: {lr_accuracy:.2f}")
lr_cmd.plot()
```

```
Accuracy: 0.78
```

[18]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x25debe992d0>



1.7 Neural Network

[24]:
```
model = Sequential([
    Input(shape=X_train_norm.shape[1:]),
    Dense(8, 'relu'),
    Dense(1, 'sigmoid')
])

model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
```

```
=================================================================
 dense_3 (Dense)             (None, 8)                 64

 dense_4 (Dense)             (None, 1)                 9

=================================================================
Total params: 73
Trainable params: 73
Non-trainable params: 0
_________________________________________________________________
```

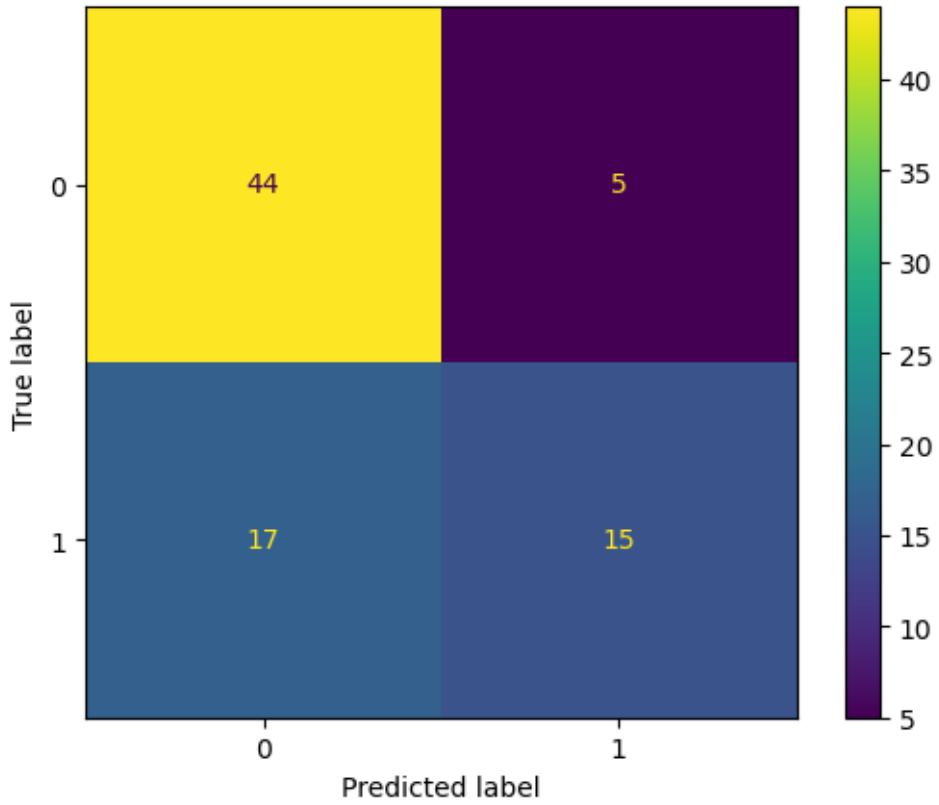
[25]:
```
model.compile(loss='binary_crossentropy', optimizer='adam',␣
 ↪metrics=['accuracy'])
```

[26]:
```
history = model.fit(X_train_norm,
                    y_train,
                    validation_split=0.0,
                    epochs = 200,
                    verbose=0)
```

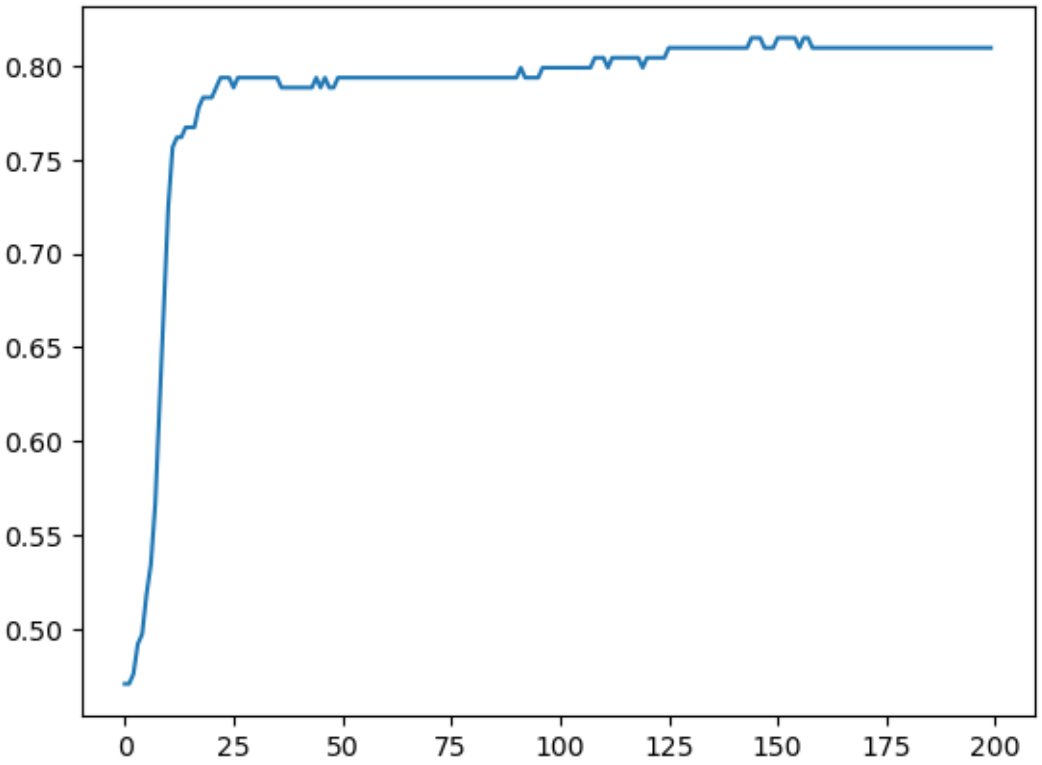
[27]:
```
y_pred_tf = (model.predict(X_test_norm) > 0.5).astype("int32")
tf_accuracy = accuracy_score(y_test, y_pred_tf)
tf_cm = confusion_matrix(y_test, y_pred_tf)
tf_cmd = ConfusionMatrixDisplay(tf_cm)
print(f"Accuracy: {tf_accuracy:.2f}")
tf_cmd.plot()
```

```
3/3 [==============================] - 0s 5ms/step
Accuracy: 0.73
```

[27]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x25deca3e980>
```



[28]:
```
plt.plot(history.history['accuracy'])
plt.show()
```



1.8 Conclusions

In this case a basic Logistic Regression can perform better than a mmore complex Neural Network.

[ ]:

[ ]:

[ ]: