kzzhuuzhp

August 2, 2023

[1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

[2]:
```python
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[3]:
```python
df=pd.read_csv("/content/drive/MyDrive/mydatasets/C4_framingham.csv")
df
```

[3]:
```
      male  age  education  currentSmoker  cigsPerDay  BPMeds  \
0        1   39        4.0              0         0.0     0.0
1        0   46        2.0              0         0.0     0.0
2        1   48        1.0              1        20.0     0.0
3        0   61        3.0              1        30.0     0.0
4        0   46        3.0              1        23.0     0.0
…      …    …          …              …           …      …
4233     1   50        1.0              1         1.0     0.0
4234     1   51        3.0              1        43.0     0.0
4235     0   48        2.0              1        20.0     NaN
4236     0   44        1.0              1        15.0     0.0
4237     0   52        2.0              0         0.0     0.0

      prevalentStroke  prevalentHyp  diabetes  totChol  sysBP  diaBP    BMI  \
0                   0             0         0    195.0  106.0   70.0  26.97
1                   0             0         0    250.0  121.0   81.0  28.73
2                   0             0         0    245.0  127.5   80.0  25.34
3                   0             1         0    225.0  150.0   95.0  28.58
4                   0             0         0    285.0  130.0   84.0  23.10
…                 …             …       …        …      …      …      …
4233                0             1         0    313.0  179.0   92.0  25.97
4234                0             0         0    207.0  126.5   80.0  19.71
4235                0             0         0    248.0  131.0   72.0  22.00
```

```
4236                0             0         0     210.0  126.5    87.0  19.16
4237                0             0         0     269.0  133.5    83.0  21.47

      heartRate  glucose  TenYearCHD
0          80.0     77.0           0
1          95.0     76.0           0
2          75.0     70.0           0
3          65.0    103.0           1
4          85.0     85.0           0
…           …       …         …
4233       66.0     86.0           1
4234       65.0     68.0           0
4235       84.0     86.0           0
4236       86.0      NaN           0
4237       80.0    107.0           0

[4238 rows x 16 columns]
```

[4]:
```
df.head()
```

[4]:
```
   male  age  education  currentSmoker  cigsPerDay  BPMeds  prevalentStroke  \
0     1   39        4.0              0         0.0     0.0                0
1     0   46        2.0              0         0.0     0.0                0
2     1   48        1.0              1        20.0     0.0                0
3     0   61        3.0              1        30.0     0.0                0
4     0   46        3.0              1        23.0     0.0                0

   prevalentHyp  diabetes  totChol  sysBP  diaBP    BMI  heartRate  glucose  \
0             0         0    195.0  106.0   70.0  26.97       80.0     77.0
1             0         0    250.0  121.0   81.0  28.73       95.0     76.0
2             0         0    245.0  127.5   80.0  25.34       75.0     70.0
3             1         0    225.0  150.0   95.0  28.58       65.0    103.0
4             0         0    285.0  130.0   84.0  23.10       85.0     85.0

   TenYearCHD
0           0
1           0
2           0
3           1
4           0
```

1 Data Cleaning and Data Preprocessing

[5]:
```
df.dropna(inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3656 entries, 0 to 4237
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   male             3656 non-null   int64
 1   age              3656 non-null   int64
 2   education        3656 non-null   float64
 3   currentSmoker    3656 non-null   int64
 4   cigsPerDay       3656 non-null   float64
 5   BPMeds           3656 non-null   float64
 6   prevalentStroke  3656 non-null   int64
 7   prevalentHyp     3656 non-null   int64
 8   diabetes         3656 non-null   int64
 9   totChol          3656 non-null   float64
 10  sysBP            3656 non-null   float64
 11  diaBP            3656 non-null   float64
 12  BMI              3656 non-null   float64
 13  heartRate        3656 non-null   float64
 14  glucose          3656 non-null   float64
 15  TenYearCHD       3656 non-null   int64
dtypes: float64(9), int64(7)
memory usage: 485.6 KB
```

[6]:
```
df.describe()
```

[6]:

	male	age	education	currentSmoker	cigsPerDay	\
count	3656.000000	3656.000000	3656.000000	3656.000000	3656.000000	
mean	0.443654	49.557440	1.979759	0.489059	9.022155	
std	0.496883	8.561133	1.022657	0.499949	11.918869	
min	0.000000	32.000000	1.000000	0.000000	0.000000	
25%	0.000000	42.000000	1.000000	0.000000	0.000000	
50%	0.000000	49.000000	2.000000	0.000000	0.000000	
75%	1.000000	56.000000	3.000000	1.000000	20.000000	
max	1.000000	70.000000	4.000000	1.000000	70.000000	

	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	\
count	3656.000000	3656.000000	3656.000000	3656.000000	3656.000000	
mean	0.030361	0.005744	0.311543	0.027079	236.873085	
std	0.171602	0.075581	0.463187	0.162335	44.096223	
min	0.000000	0.000000	0.000000	0.000000	113.000000	
25%	0.000000	0.000000	0.000000	0.000000	206.000000	

```
50%       0.000000         0.000000      0.000000      0.000000    234.000000
75%       0.000000         0.000000      1.000000      0.000000    263.250000
max       1.000000         1.000000      1.000000      1.000000    600.000000

             sysBP        diaBP          BMI    heartRate      glucose  \
count  3656.000000  3656.000000  3656.000000  3656.000000  3656.000000
mean    132.368025    82.912062    25.784185    75.730580    81.856127
std      22.092444    11.974825     4.065913    11.982952    23.910128
min      83.500000    48.000000    15.540000    44.000000    40.000000
25%     117.000000    75.000000    23.080000    68.000000    71.000000
50%     128.000000    82.000000    25.380000    75.000000    78.000000
75%     144.000000    90.000000    28.040000    82.000000    87.000000
max     295.000000   142.500000    56.800000   143.000000   394.000000

         TenYearCHD
count  3656.000000
mean      0.152352
std       0.359411
min       0.000000
25%       0.000000
50%       0.000000
75%       0.000000
max       1.000000
```

[7]:
```
df.columns
```

[7]:
```
Index(['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose', 'TenYearCHD'],
      dtype='object')
```

[8]:
```
feature_matrix = df.iloc[:,0:15]
target_vector = df.iloc[:,-1]
```

[9]:
```
fs = StandardScaler().fit_transform(feature_matrix)
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

[9]:
```
LogisticRegression()
```

[10]:
```
observation=[[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]]
prediction = logr.predict(observation)
print(prediction)
```

```
[1]
```

[11]:
```
logr.classes_
```

[11]: array([0, 1])

[12]:
```
logr.predict_proba(observation)
```

[12]: array([[2.21478351e-04, 9.99778522e-01]])

Random Forest

[18]:
```
x = df.iloc[:,0:15]
y = df.iloc[:,-1]
```

[19]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

[20]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
```

[20]: RandomForestClassifier()

[21]:
```
parameters = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,15,20,25],
              'n_estimators': [10,20,30,40,50]
              }
```

[22]:
```
from sklearn.model_selection import GridSearchCV
grid_search =␣
 ↪GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

[22]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

[23]:
```
grid_search.best_score_
```

[23]: 0.8499410550234558

[24]:
```
rfc_best = grid_search.best_estimator_
```

[25]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(89,40))
plot_tree(rfc_best.estimators_[5], feature_names=x.columns, class_names=['Yes',␣
 ↪'No'], filled=True)
```

[25]:
```
[Text(0.5625, 0.9166666666666666, 'glucose <= 146.0\ngini = 0.259\nsamples =
1637\nvalue = [2167, 392]\nclass = Yes'),
 Text(0.5267857142857143, 0.75, 'prevalentHyp <= 0.5\ngini = 0.25\nsamples =
1616\nvalue = [2157, 370]\nclass = Yes'),
 Text(0.2857142857142857, 0.5833333333333334, 'age <= 48.5\ngini =
0.186\nsamples = 1110\nvalue = [1545, 179]\nclass = Yes'),
 Text(0.14285714285714285, 0.4166666666666667, 'cigsPerDay <= 9.5\ngini =
0.113\nsamples = 660\nvalue = [955, 61]\nclass = Yes'),
 Text(0.07142857142857142, 0.25, 'glucose <= 91.5\ngini = 0.071\nsamples =
356\nvalue = [521, 20]\nclass = Yes'),
 Text(0.03571428571428571, 0.08333333333333333, 'gini = 0.052\nsamples =
323\nvalue = [473, 13]\nclass = Yes'),
 Text(0.10714285714285714, 0.08333333333333333, 'gini = 0.222\nsamples =
33\nvalue = [48, 7]\nclass = Yes'),
 Text(0.21428571428571427, 0.25, 'heartRate <= 90.5\ngini = 0.158\nsamples =
304\nvalue = [434, 41]\nclass = Yes'),
 Text(0.17857142857142858, 0.08333333333333333, 'gini = 0.168\nsamples =
284\nvalue = [403, 41]\nclass = Yes'),
 Text(0.25, 0.08333333333333333, 'gini = 0.0\nsamples = 20\nvalue = [31,
0]\nclass = Yes'),
 Text(0.42857142857142855, 0.4166666666666667, 'cigsPerDay <= 2.5\ngini =
0.278\nsamples = 450\nvalue = [590, 118]\nclass = Yes'),
 Text(0.35714285714285715, 0.25, 'sysBP <= 145.5\ngini = 0.2\nsamples =
260\nvalue = [369, 47]\nclass = Yes'),
 Text(0.32142857142857145, 0.08333333333333333, 'gini = 0.183\nsamples =
245\nvalue = [352, 40]\nclass = Yes'),
 Text(0.39285714285714285, 0.08333333333333333, 'gini = 0.413\nsamples =
15\nvalue = [17, 7]\nclass = Yes'),
 Text(0.5, 0.25, 'glucose <= 63.5\ngini = 0.368\nsamples = 190\nvalue = [221,
71]\nclass = Yes'),
 Text(0.4642857142857143, 0.08333333333333333, 'gini = 0.0\nsamples = 19\nvalue
= [28, 0]\nclass = Yes'),
 Text(0.5357142857142857, 0.08333333333333333, 'gini = 0.393\nsamples =
171\nvalue = [193, 71]\nclass = Yes'),
 Text(0.7678571428571429, 0.5833333333333334, 'age <= 54.5\ngini =
0.363\nsamples = 506\nvalue = [612, 191]\nclass = Yes'),
 Text(0.6785714285714286, 0.4166666666666667, 'sysBP <= 176.75\ngini =
0.259\nsamples = 269\nvalue = [360, 65]\nclass = Yes'),
 Text(0.6428571428571429, 0.25, 'male <= 0.5\ngini = 0.222\nsamples = 250\nvalue
= [343, 50]\nclass = Yes'),
 Text(0.6071428571428571, 0.08333333333333333, 'gini = 0.123\nsamples =
123\nvalue = [184, 13]\nclass = Yes'),
 Text(0.6785714285714286, 0.08333333333333333, 'gini = 0.306\nsamples =
127\nvalue = [159, 37]\nclass = Yes'),
 Text(0.7142857142857143, 0.25, 'gini = 0.498\nsamples = 19\nvalue = [17,
15]\nclass = Yes'),
 Text(0.8571428571428571, 0.4166666666666667, 'glucose <= 69.5\ngini =
```

```
0.444\nsamples = 237\nvalue = [252, 126]\nclass = Yes'),
 Text(0.7857142857142857, 0.25, 'heartRate <= 75.5\ngini = 0.124\nsamples =
28\nvalue = [42, 3]\nclass = Yes'),
 Text(0.75, 0.08333333333333333, 'gini = 0.153\nsamples = 16\nvalue = [22,
2]\nclass = Yes'),
 Text(0.8214285714285714, 0.08333333333333333, 'gini = 0.091\nsamples =
12\nvalue = [20, 1]\nclass = Yes'),
 Text(0.9285714285714286, 0.25, 'totChol <= 318.5\ngini = 0.466\nsamples =
209\nvalue = [210, 123]\nclass = Yes'),
 Text(0.8928571428571429, 0.08333333333333333, 'gini = 0.452\nsamples =
191\nvalue = [198, 104]\nclass = Yes'),
 Text(0.9642857142857143, 0.08333333333333333, 'gini = 0.475\nsamples =
18\nvalue = [12, 19]\nclass = No'),
 Text(0.5982142857142857, 0.75, 'gini = 0.43\nsamples = 21\nvalue = [10,
22]\nclass = No')]
```