6pjgzkdea

August 2, 2023

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

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

```
[3]: df=pd.read_csv("/content/drive/MyDrive/mydatasets/C6_bmi.csv")
     df
```

```
[3]:      Gender  Height  Weight  Index
     0      Male     174      96      4
     1      Male     189      87      2
     2    Female     185     110      4
     3    Female     195     104      3
     4      Male     149      61      3
     ..      …       …       …      …
     495  Female     150     153      5
     496  Female     184     121      4
     497  Female     141     136      5
     498    Male     150      95      5
     499    Male     173     131      5

     [500 rows x 4 columns]
```

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

```
[4]:   Gender  Height  Weight  Index
     0    Male     174      96      4
     1    Male     189      87      2
     2  Female     185     110      4
     3  Female     195     104      3
     4    Male     149      61      3
```

1 Data Cleaning and Data Preprocessing

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Gender  500 non-null    object
 1   Height  500 non-null    int64
 2   Weight  500 non-null    int64
 3   Index   500 non-null    int64
dtypes: int64(3), object(1)
memory usage: 15.8+ KB
```

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

[6]:
```
           Height      Weight       Index
count  500.000000  500.000000  500.000000
mean   169.944000  106.000000    3.748000
std     16.375261   32.382607    1.355053
min    140.000000   50.000000    0.000000
25%    156.000000   80.000000    3.000000
50%    170.500000  106.000000    4.000000
75%    184.000000  136.000000    5.000000
max    199.000000  160.000000    5.000000
```

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

[7]:
```
Index(['Gender', 'Height', 'Weight', 'Index'], dtype='object')
```

[8]:
```
feature_matrix = df.iloc[:,1:3]
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]]
prediction = logr.predict(observation)
print(prediction)
```

```
[5]
```

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

[11]:
```
array([0, 1, 2, 3, 4, 5])
```

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

[12]:
```
array([[5.59566976e-11, 6.05990036e-10, 1.19071465e-07, 4.99471797e-05,
        2.03791363e-02, 9.79570797e-01]])
```

Random Forest

[22]:
```
df
```

[22]:
```
     Gender  Height  Weight  Index
0      Male     174      96      4
1      Male     189      87      2
2    Female     185     110      4
3    Female     195     104      3
4      Male     149      61      3
..      …       …       …     …
495  Female     150     153      5
496  Female     184     121      4
497  Female     141     136      5
498    Male     150      95      5
499    Male     173     131      5

[500 rows x 4 columns]
```

[23]:
```
g1={"Gender":{"Male":1, "Female":2}}
df=df.replace(g1)
df
```

[23]:
```
     Gender  Height  Weight  Index
0         1     174      96      4
1         1     189      87      2
2         2     185     110      4
3         2     195     104      3
4         1     149      61      3
..      …       …       …     …
495       2     150     153      5
496       2     184     121      4
497       2     141     136      5
498       1     150      95      5
499       1     173     131      5

[500 rows x 4 columns]
```

[25]:
```
x=df.drop('Gender', axis=1)
y=df['Gender']
```

[26]:
```
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)
```

[27]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
```

[27]:
```
RandomForestClassifier()
```

[28]:
```
parameters = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,15,20,25],
              'n_estimators': [10,20,30,40,50]
              }
```

[29]:
```
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)
```

[29]:
```
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')
```

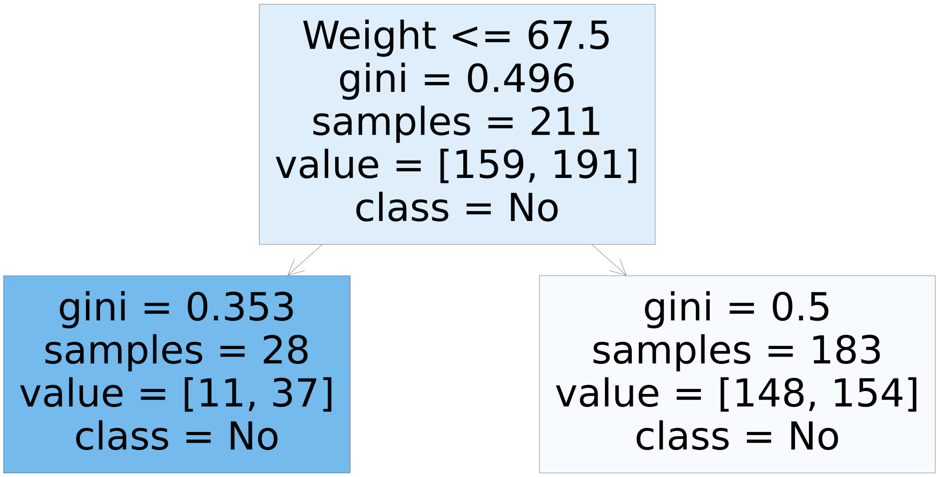
[30]:
```
grid_search.best_score_
```

[30]:
```
0.5628571428571428
```

[31]:
```
rfc_best = grid_search.best_estimator_
```

[32]:
```
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)
```

[32]:
```
[Text(0.5, 0.75, 'Weight <= 67.5\ngini = 0.496\nsamples = 211\nvalue = [159,
191]\nclass = No'),
 Text(0.25, 0.25, 'gini = 0.353\nsamples = 28\nvalue = [11, 37]\nclass = No'),
 Text(0.75, 0.25, 'gini = 0.5\nsamples = 183\nvalue = [148, 154]\nclass = No')]
```


Weight <= 67.5
gini = 0.496
samples = 211
value = [159, 191]
class = No
gini = 0.353
samples = 28
value = [11, 37]
class = No
gini = 0.5
samples = 183
value = [148, 154]
class = No