```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = sns.load_dataset("iris")

df.head(4)
```

| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |

```
df["species"].unique()
```

```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

```
name = {
    "setosa" : 1,
     "versicolor":2,
     "virginica":3}

df["species"]=df["species"].map(name)

df["species"].unique()
```

```
array([1, 2, 3])
```

```
 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    int64  
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

```
df.head()
```

| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | 1 |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | 1 |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | 1 |

```
3            4.6           3.1            1.5           0.2         1
4            5.0           3.6            1.4           0.2         1
```

```
x = df.iloc[:,:-1]
```

```
x.head(3)
```

```
   sepal_length  sepal_width  petal_length  petal_width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
```

```
y = df["species"]
y
```

```
0      1
1      1
2      1
3      1
4      1
      ..
145    3
146    3
147    3
148    3
149    3
Name: species, Length: 150, dtype: int64
```

```
y.head(4)
```

```
0    1
1    1
2    1
3    1
Name: species, dtype: int64
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=.20,random_state = 34)
```

```
x_train.shape
```

```
(120, 4)
```

```
x_test.shape
```

```
(30, 4)
```

```
y_train.shape
```

```
(120,)
```

```
y_test.shape
```

```
(30,)
```

```
from sklearn.tree import DecisionTreeClassifier

cls = DecisionTreeClassifier()

cls.fit(x_train,y_train)
```

```
DecisionTreeClassifier()
```

```
y_pred =cls.predict(x_test)
y_pred
```

```
array([2, 3, 1, 2, 2, 2, 1, 1, 2, 2, 1, 2, 1, 2, 1, 2, 1, 1, 2, 2, 1,
2,
       2, 3, 1, 2, 2, 1, 2, 2])
```

```
y_test
```

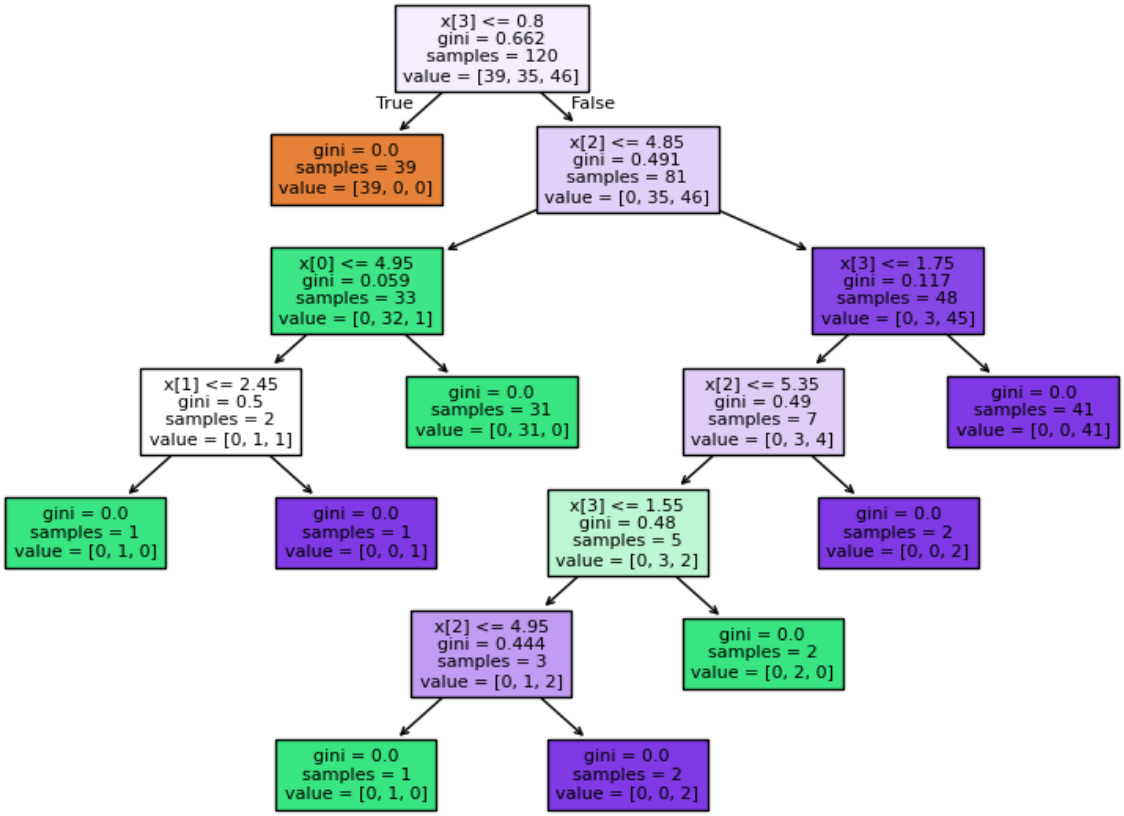
```
97     2
149    3
27     1
60     2
138    3
55     2
10     1
32     1
86     2
67     2
12     1
78     2
7      1
82     2
33     1
94     2
26     1
40     1
52     2
126    3
3      1
65     2
50     2
108    3
46     1
54     2
64     2
22     1
56     2
```

```
53     2
Name: species, dtype: int64
```

```
from sklearn.metrics import accuracy_score

accuracy_score(y_test,y_pred)
```

```
0.9333333333333333
```

```
from sklearn import tree

plt.figure(figsize =((10,7)))
tree.plot_tree(cls,filled = True,fontsize = 8)
plt.show()
```

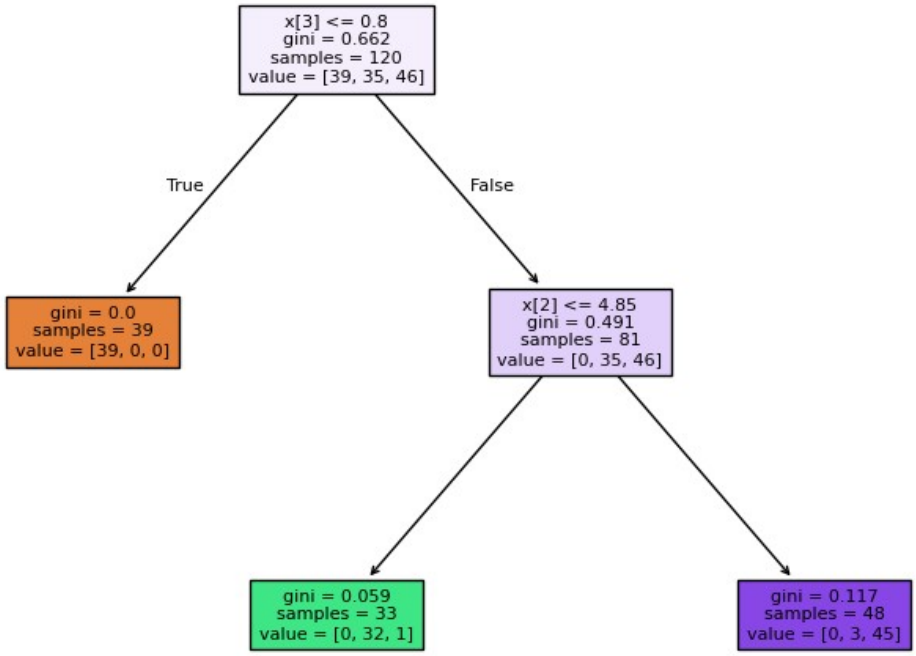



```
cls2 = DecisionTreeClassifier(max_depth = 2)

cls2.fit(x_train,y_train)
```

```
DecisionTreeClassifier(max_depth=2)
```

```
y_pred2 = cls2.predict(x_test)

y_pred2
```

```
array([2, 3, 1, 2, 2, 2, 1, 1, 2, 2, 1, 2, 1, 2, 1, 2, 1, 1, 3, 2, 1,
2,
       2, 3, 1, 2, 2, 1, 2, 2])

accuracy_score(y_test,y_pred2)

0.9

plt.figure(figsize =((10,7)))
tree.plot_tree(cls2,filled = True,fontsize = 8)
plt.show()
```




```
from sklearn.metrics import classification_report

print(classification_report(y_pred2,y_test))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 1.00 | 1.00 | 1.00 | 11 |
| 2 | 0.93 | 0.88 | 0.90 | 16 |
| 3 | 0.50 | 0.67 | 0.57 | 3 |
| accuracy | | | 0.90 | 30 |
| macro avg | 0.81 | 0.85 | 0.82 | 30 |

```
weighted avg       0.91      0.90      0.91        30


print(classification_report(y_pred,y_test))

              precision    recall  f1-score   support

           1       1.00      1.00      1.00        11
           2       1.00      0.88      0.94        17
           3       0.50      1.00      0.67         2

    accuracy                           0.93        30
   macro avg       0.83      0.96      0.87        30
weighted avg       0.97      0.93      0.94        30


# preprunning
import pandas as pd
df = pd.read_csv("bcd.csv")

df.head()

   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0            6      148             72             35        0  33.6

1            1       85             66             29        0  26.6

2            8      183             64              0        0  23.3

3            1       89             66             23       94  28.1

4            0      137             40             35      168  43.1


   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1

x1 = df.iloc[:,:-1]
x1

     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
\
0              6      148             72             35        0  33.6

1              1       85             66             29        0  26.6

2              8      183             64              0        0  23.3
```

```
3               1       89              66              23        94  28.1
4               0      137              40              35       168  43.1
..            ...      ...             ...             ...      ...   ...
763            10      101              76              48       180  32.9
764             2      122              70              27         0  36.8
765             5      121              72              23       112  26.2
766             1      126              60               0         0  30.1
767             1       93              70              31         0  30.4
```

```
     DiabetesPedigreeFunction  Age
0                       0.627   50
1                       0.351   31
2                       0.672   32
3                       0.167   21
4                       2.288   33
..                        ...  ...
763                     0.171   63
764                     0.340   27
765                     0.245   30
766                     0.349   47
767                     0.315   23

[768 rows x 8 columns]
```

```
y1 = df[['Outcome']]

y1
```

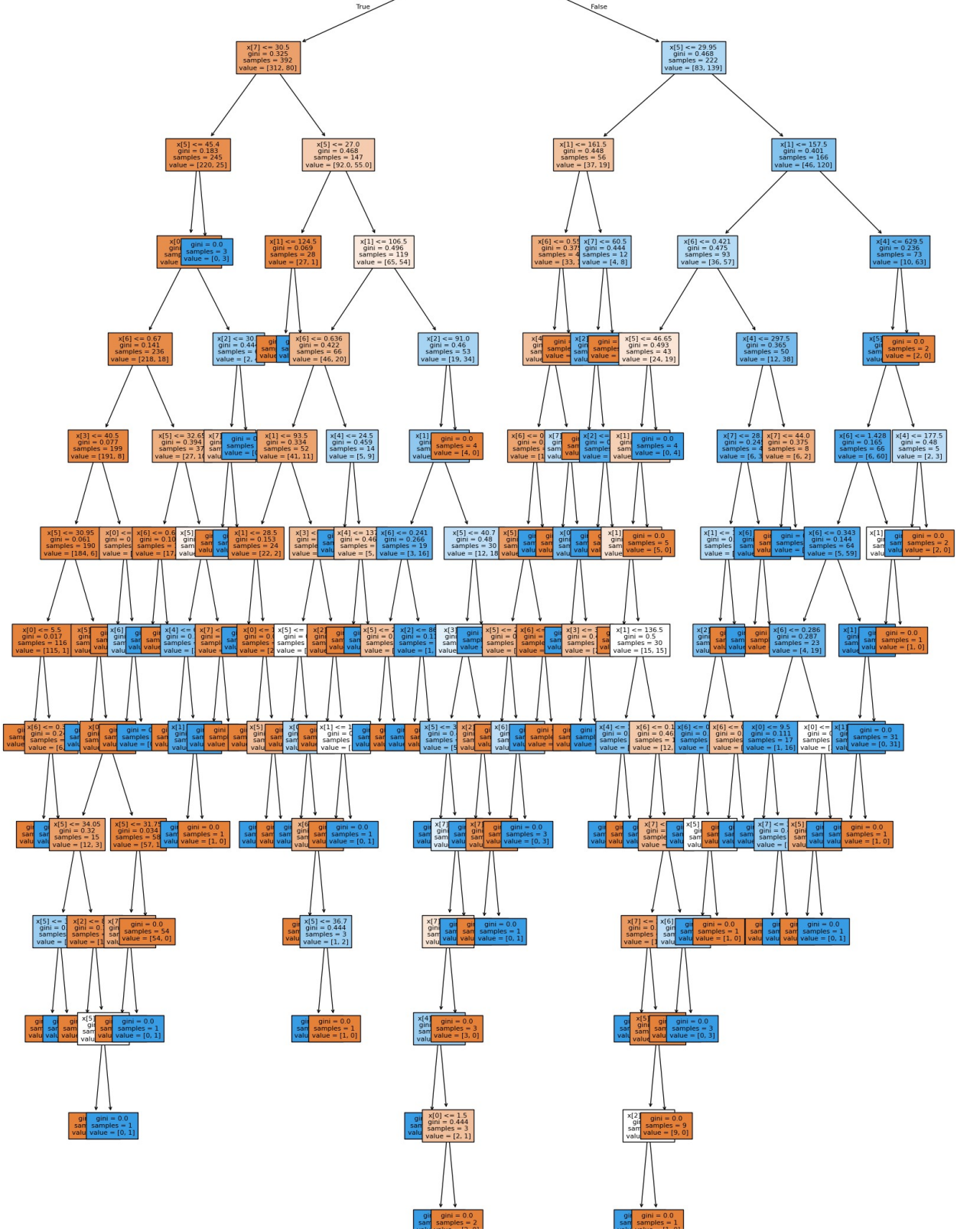
```
     Outcome
0          1
1          0
2          1
3          0
4          1
..       ...
763        0
764        0
765        0
766        1
767        0

[768 rows x 1 columns]
```

```
from sklearn.model_selection import train_test_split
x1_train,x1_test,y1_train,y1_test =
train_test_split(x1,y1,test_size=.20,random_state = 34)

from sklearn.tree import DecisionTreeClassifier
cls3 = DecisionTreeClassifier()
cls3.fit(x1_train,y1_train)
```

```
DecisionTreeClassifier()
```

```
y_pred3 = cls3.predict(x1_test)
print(classification_report(y_pred3,y1_test))
```

```
              precision    recall  f1-score   support

           0       0.79      0.79      0.79       105
           1       0.55      0.55      0.55        49

    accuracy                           0.71       154
   macro avg       0.67      0.67      0.67       154
weighted avg       0.71      0.71      0.71       154


plt.figure(figsize =((20,30)))
tree.plot_tree(cls3,filled = True,fontsize = 8)
plt.show()
```


x[1] <= 127.5
gini = 0.459
samples = 614
value = [395.0, 219.0]
True
False
x[7] <= 30.5
gini = 0.325
samples = 392
value = [312, 80]
x[5] <= 29.95
gini = 0.468
samples = 222
value = [83, 139]
x[5] <= 45.4
gini = 0.183
samples = 245
value = [220, 25]
x[5] <= 27.0
gini = 0.468
samples = 147
value = [92.0, 55.0]
x[1] <= 161.5
gini = 0.448
samples = 56
value = [37, 19]
x[1] <= 157.5
gini = 0.401
samples = 166
value = [46, 120]
gini = 0.0
samples = 3
value = [0, 3]
x[1] <= 124.5
gini = 0.069
samples = 28
value = [27, 1]
x[1] <= 106.5
gini = 0.496
samples = 119
value = [65, 54]
x[6] <= 0.559
gini = 0.375
samples = 4
value = [3, 1]
x[7] <= 60.5
gini = 0.444
samples = 12
value = [4, 8]
x[6] <= 0.421
gini = 0.475
samples = 93
value = [36, 57]
x[4] <= 629.5
gini = 0.236
samples = 73
value = [10, 63]
x[6] <= 0.67
gini = 0.141
samples = 236
value = [218, 18]
x[6] <= 0.636
gini = 0.422
samples = 66
value = [46, 20]
x[2] <= 91.0
gini = 0.46
samples = 53
value = [19, 34]
x[5] <= 46.65
gini = 0.493
samples = 43
value = [24, 19]
x[4] <= 297.5
gini = 0.365
samples = 50
value = [12, 38]
gini = 0.0
samples = 2
value = [2, 0]
x[3] <= 40.5
gini = 0.077
samples = 199
value = [191, 8]
x[1] <= 93.5
gini = 0.334
samples = 52
value = [41, 11]
x[4] <= 24.5
gini = 0.459
samples = 14
value = [5, 9]
gini = 0.0
samples = 4
value = [4, 0]
gini = 0.0
samples = 4
value = [0, 4]
x[7] <= 44.0
gini = 0.375
samples = 8
value = [6, 2]
x[6] <= 1.428
gini = 0.165
samples = 66
value = [6, 60]
x[4] <= 177.5
gini = 0.48
samples = 5
value = [2, 3]
x[5] <= 30.95
gini = 0.061
samples = 190
value = [184, 6]
x[1] <= 28.5
gini = 0.153
samples = 24
value = [22, 2]
x[4] <= 13
gini = 0.46
samples = 5
value = [5,
x[6] <= 0.241
gini = 0.266
samples = 19
value = [3, 16]
x[5] <= 40.7
gini = 0.48
samples = 30
value = [12, 18
gini = 0.0
samples = 5
value = [5, 0]
x[6] <= 0.343
gini = 0.144
samples = 64
value = [5, 59]
gini = 0.0
samples = 2
value = [2, 0]
x[0] <= 5.5
gini = 0.017
samples = 116
value = [115, 1]
x[1] <= 136.5
gini = 0.5
samples = 30
value = [15, 15]
x[6] <= 0.286
gini = 0.287
samples = 23
value = [4, 19]
gini = 0.0
samples = 1
value = [1, 0]
x[5] <= 34.05
gini = 0.32
samples = 15
value = [12, 3]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 3
value = [0, 3]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 54
value = [54, 0]
x[5] <= 36.7
gini = 0.444
samples = 3
value = [1, 2]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 3
value = [3, 0]
x[0] <= 1.5
gini = 0.444
samples = 3
value = [2, 1]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 9
value = [9, 0]

```
from sklearn.model_selection import GridSearchCV

cls3 = DecisionTreeClassifier()

cls3
```

```
DecisionTreeClassifier()
```

```
parameter = {"criterion":["gini","entropy","log_loss"],
             "splitter":["best","random"],
              "max_depth":[4,5,6,7]}


Gridsearch = GridSearchCV(cls3,parameter,cv=5)
Gridsearch.fit(x1_train,y1_train)
```

```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy', 'log_loss'],
                         'max_depth': [4, 5, 6, 7],
                         'splitter': ['best', 'random']})
```

```
Gridsearch.best_params_
```

```
{'criterion': 'log_loss', 'max_depth': 6, 'splitter': 'random'}
```

```
y_predg = Gridsearch.predict(x1_test)
print(classification_report(y_predg,y1_test))
```

```
              precision    recall  f1-score   support

           0       0.90      0.81      0.85       118
           1       0.53      0.72      0.61        36

    accuracy                           0.79       154
   macro avg       0.72      0.76      0.73       154
weighted avg       0.82      0.79      0.80       154
```

```
cls4 = DecisionTreeClassifier(criterion = "entropy",max_depth=
4,splitter = "best")
cls4.fit(x1_train,y1_train)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=4)
```

```
import joblib

joblib.dump(cls4,"dtc.pkl")
```

```
['dtc.pkl']
```