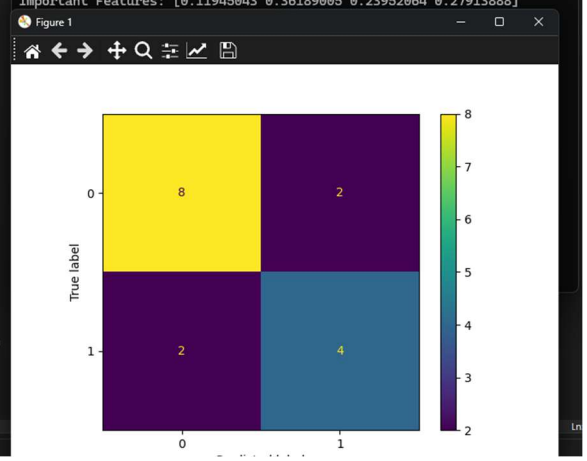
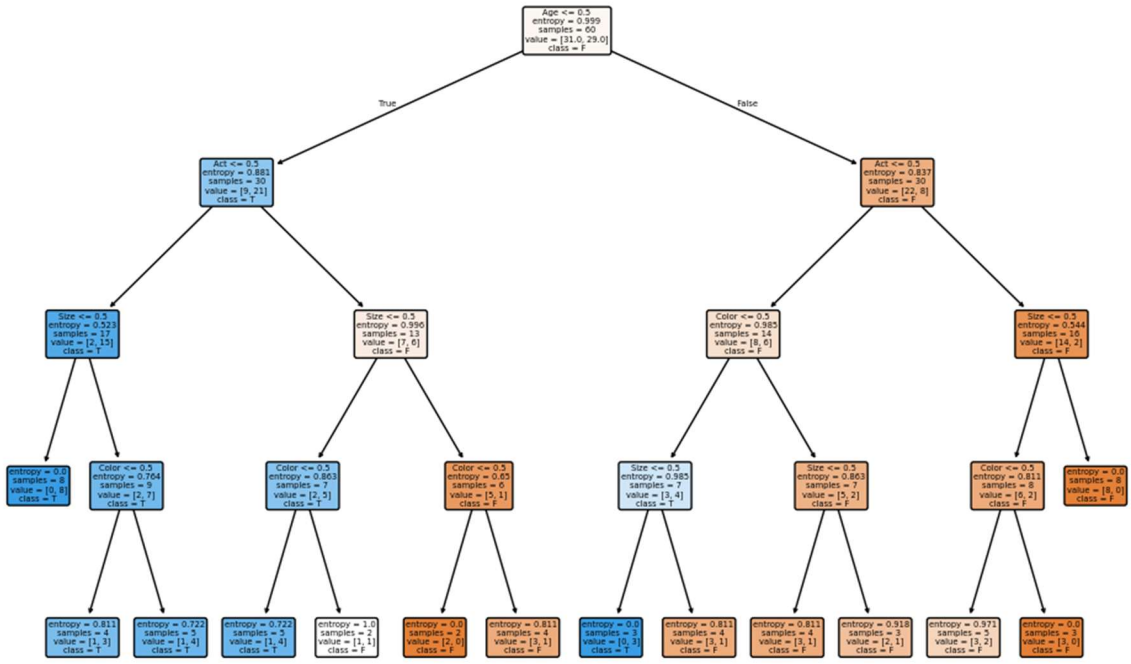
1.

```python
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, plot_tree, export_text
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score,ConfusionMatrixDisplay, classification_report

data = pd.read_csv('balloons_extended.csv')
data['Color'] = data['Color'].map({'YELLOW': 0, 'PURPLE': 1})
data['size'] = data['size'].map({'SMALL': 0, 'LARGE': 1})
data['act'] = data['act'].map({'STRETCH': 0, 'DIP': 1})
data['age'] = data['age'].map({'ADULT': 0, 'CHILD': 1})
data['inflated'] = data['inflated'].map({'T': 1, 'F': 0})

X = data[['Color', 'size', 'act', 'age']]
y = data['inflated']

# Train Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20,random_state=0)

# Train Decision Tree

dt = DecisionTreeClassifier(criterion='entropy', random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
confusionMatrix = confusion_matrix(y_test, y_pred)

# text representations
print(f'Classification Report:\n{classification_report(y_test, y_pred)}')
print('\nAccuracy score: ', accuracy)
print(f'Important Features: {dt.feature_importances_}')


# Visual Confusion Matrix
disp = ConfusionMatrixDisplay(confusion_matrix=confusionMatrix,display_labels=dt.classes_)
disp.plot()
plt.show()

# dt text representations
text_representation = export_text(dt)
print(text_representation)

# Plot the tree
plt.figure(figsize=(12, 8))
plot_tree(dt, feature_names=['Color', 'Size', 'Act', 'Age'], class_names=['F','T'], filled=True, rounded=True)
plt.show()
```

No issues found

```
C:\WINDOWS\system32\cmd.

Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.80      0.80        10
           1       0.67      0.67      0.67         6

    accuracy                           0.75        16
   macro avg       0.73      0.73      0.73        16
weighted avg       0.75      0.75      0.75        16


Accuracy score:  0.75
Important Features: [0.11945043 0.36189005 0.23952064 0.27913888]
```



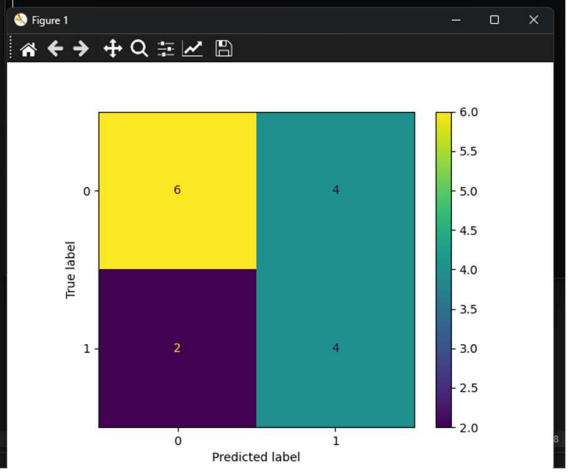


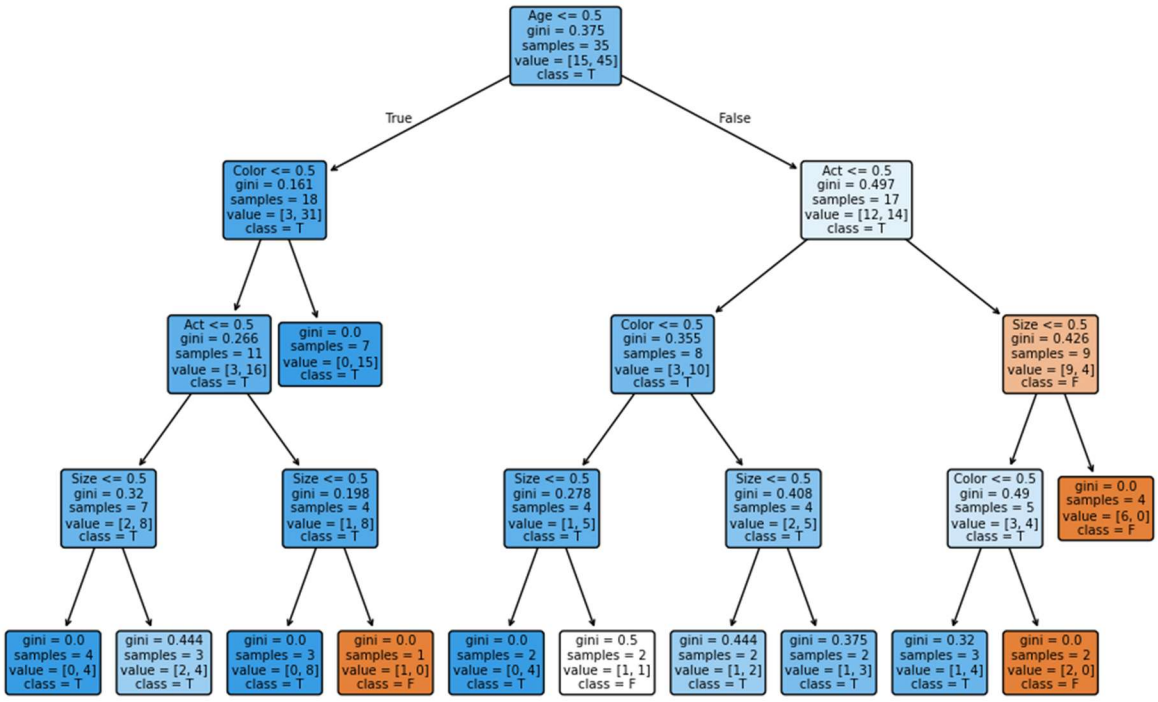
```
|--- feature_3 <= 0.50
|   |--- feature_2 <= 0.50
|   |   |--- feature_1 <= 0.50
|   |   |   |--- class: 1
|   |   |--- feature_1 >  0.50
|   |   |   |--- feature_0 <= 0.50
|   |   |   |   |--- class: 1
|   |   |   |--- feature_0 >  0.50
|   |   |   |   |--- class: 1
|   |--- feature_2 >  0.50
|   |   |--- feature_1 <= 0.50
|   |   |   |--- feature_0 <= 0.50
|   |   |   |   |--- class: 1
|   |   |   |--- feature_0 >  0.50
|   |   |   |   |--- class: 0
|   |   |--- feature_1 >  0.50
|   |   |   |--- feature_0 <= 0.50
|   |   |   |   |--- class: 0
|   |   |   |--- feature_0 >  0.50
|   |   |   |   |--- class: 0
|--- feature_3 >  0.50
|   |--- feature_2 <= 0.50
|   |   |--- feature_0 <= 0.50
|   |   |   |--- feature_1 <= 0.50
|   |   |   |   |--- class: 1
|   |   |   |--- feature_1 >  0.50
```

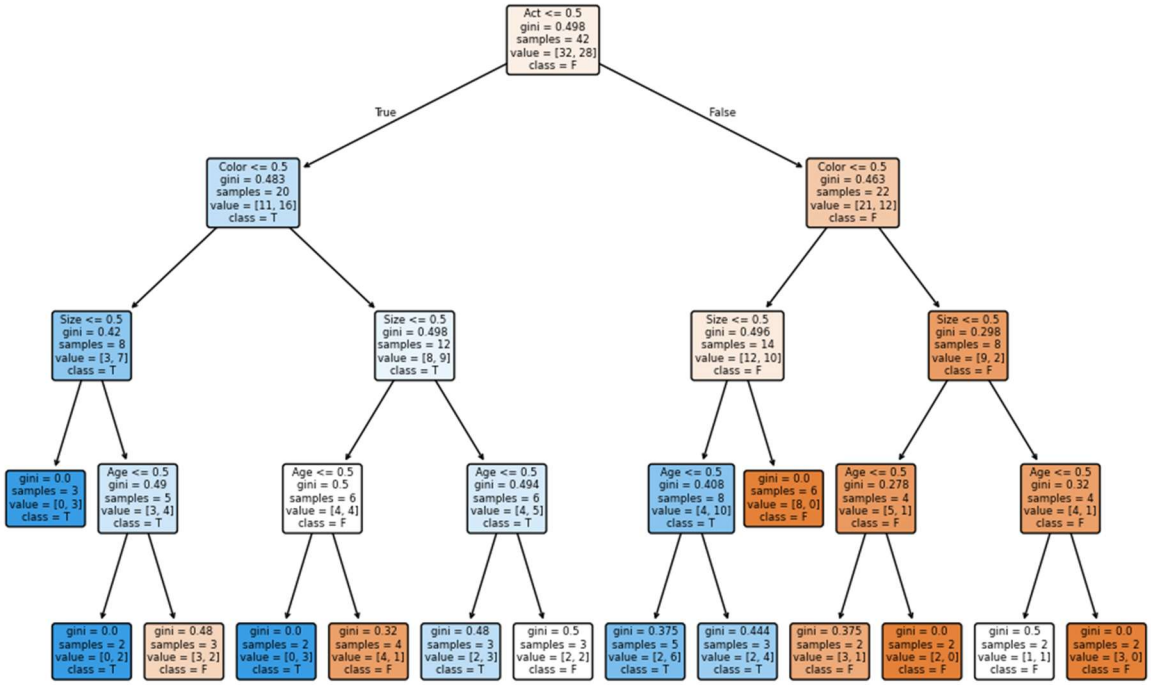
2.

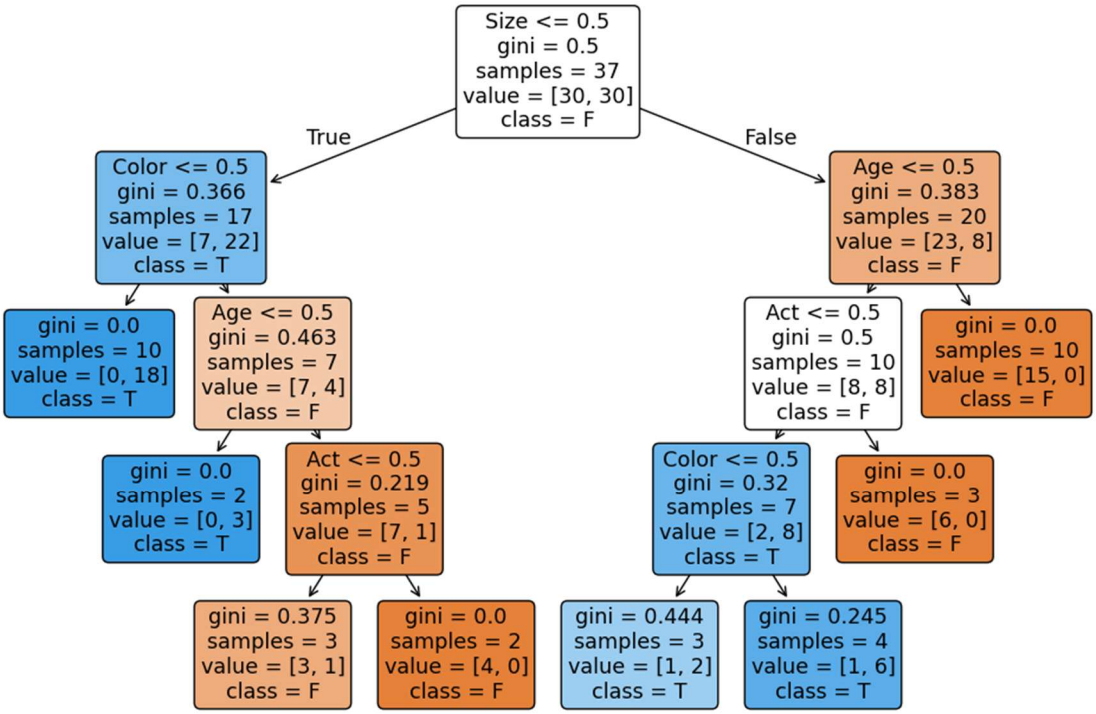
```python
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, plot_tree, export_text
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score,ConfusionMatrixDisplay, classification_report
from sklearn.ensemble import RandomForestClassifier

data = pd.read_csv('balloons_extended.csv')
data['Color'] = data['Color'].map({'YELLOW': 0, 'PURPLE': 1})
data['size'] = data['size'].map({'SMALL': 0, 'LARGE': 1})
data['act'] = data['act'].map({'STRETCH': 0, 'DIP': 1})
data['age'] = data['age'].map({'ADULT': 0, 'CHILD': 1})
data['inflated'] = data['inflated'].map({'T': 1, 'F': 0})

X = data[['Color', 'size', 'act', 'age']]
y = data['inflated']

# Train Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.28,random_state=0)

# Train Random Forest Classifier Tree

RF = RandomForestClassifier(n_estimators=100)
RF.fit(X_train, y_train)

y_pred = RF.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
confusionMatrix = confusion_matrix(y_test, y_pred)

# text representations
print(f'Classification Report:\n{classification_report(y_test, y_pred)}')
print('\nAccuracy score: ', accuracy)
print(f'Important Features: {RF.feature_importances_}')

# Visual Confusion Matrix
disp = ConfusionMatrixDisplay(confusion_matrix=confusionMatrix,display_labels=RF.classes_)
disp.plot()
plt.show()

#RF Visualization
for i in range(5):
    # Extract an individual decision tree from the Random Forest
    dt = RF.estimators_[i]

    # Visualize the decision tree using Matplotlib's plot_tree method
    # Plot the tree
    plt.figure(figsize=(12, 8))
    plot_tree(dt, feature_names=['Color', 'Size', 'Act', 'Age'], class_names=['F','T'], filled=True, rounded=True)
    plt.show()
```

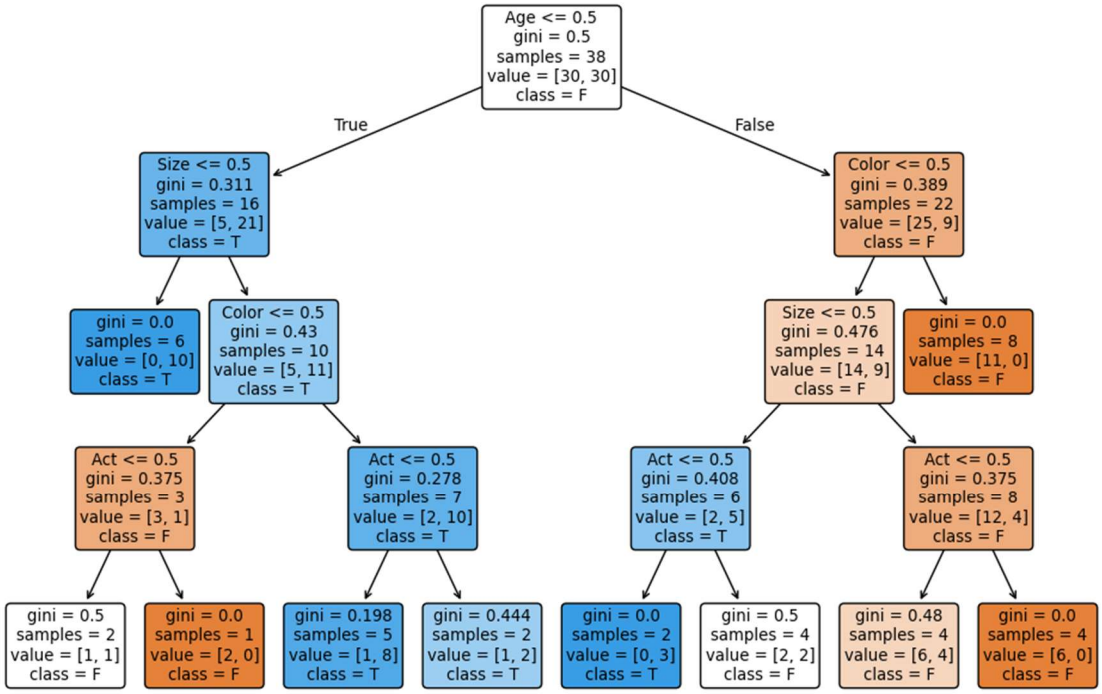
```
Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.60      0.67        10
           1       0.50      0.67      0.57         6

    accuracy                           0.62        16
   macro avg       0.62      0.63      0.62        16
weighted avg       0.66      0.62      0.63        16


Accuracy score:  0.625
Important Features: [0.13782671 0.25174334 0.29680135 0.3136286 ]
```

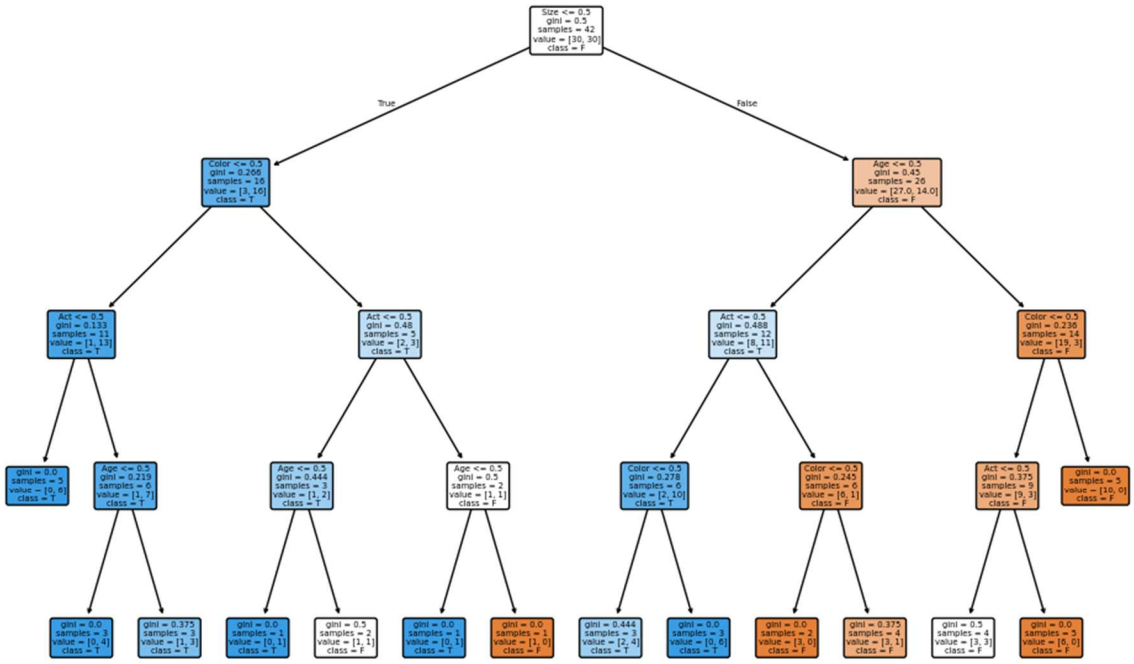





Act <= 0.5
gini = 0.498
samples = 42
value = [32, 28]
class = F
True
False
Color <= 0.5
gini = 0.483
samples = 20
value = [11, 16]
class = T
Color <= 0.5
gini = 0.463
samples = 22
value = [21, 12]
class = F
Size <= 0.5
gini = 0.42
samples = 8
value = [3, 7]
class = T
Size <= 0.5
gini = 0.498
samples = 12
value = [8, 9]
class = T
Size <= 0.5
gini = 0.496
samples = 14
value = [12, 10]
class = F
Size <= 0.5
gini = 0.298
samples = 8
value = [9, 2]
class = F
gini = 0.0
samples = 3
value = [0, 3]
class = T
Age <= 0.5
gini = 0.49
samples = 5
value = [3, 4]
class = T
Age <= 0.5
gini = 0.5
samples = 6
value = [4, 4]
class = F
Age <= 0.5
gini = 0.494
samples = 6
value = [4, 5]
class = T
Age <= 0.5
gini = 0.408
samples = 8
value = [4, 10]
class = T
gini = 0.0
samples = 6
value = [8, 0]
class = F
Age <= 0.5
gini = 0.278
samples = 4
value = [5, 1]
class = F
Age <= 0.5
gini = 0.32
samples = 4
value = [4, 1]
class = F
gini = 0.0
samples = 2
value = [0, 2]
class = T
gini = 0.48
samples = 3
value = [3, 2]
class = F
gini = 0.0
samples = 2
value = [0, 3]
class = T
gini = 0.32
samples = 4
value = [4, 1]
class = F
gini = 0.48
samples = 3
value = [2, 3]
class = T
gini = 0.5
samples = 3
value = [2, 2]
class = F
gini = 0.375
samples = 5
value = [2, 6]
class = T
gini = 0.444
samples = 3
value = [2, 4]
class = T
gini = 0.375
samples = 2
value = [3, 1]
class = F
gini = 0.0
samples = 2
value = [2, 0]
class = F
gini = 0.5
samples = 2
value = [1, 1]
class = F
gini = 0.0
samples = 2
value = [3, 0]
class = F


Size <= 0.5
gini = 0.5
samples = 37
value = [30, 30]
class = F
True
False
Color <= 0.5
gini = 0.366
samples = 17
value = [7, 22]
class = T
Age <= 0.5
gini = 0.383
samples = 20
value = [23, 8]
class = F
gini = 0.0
samples = 10
value = [0, 18]
class = T
Age <= 0.5
gini = 0.463
samples = 7
value = [7, 4]
class = F
Act <= 0.5
gini = 0.5
samples = 10
value = [8, 8]
class = F
gini = 0.0
samples = 10
value = [15, 0]
class = F
gini = 0.0
samples = 2
value = [0, 3]
class = T
Act <= 0.5
gini = 0.219
samples = 5
value = [7, 1]
class = F
Color <= 0.5
gini = 0.32
samples = 7
value = [2, 8]
class = T
gini = 0.0
samples = 3
value = [6, 0]
class = F
gini = 0.375
samples = 3
value = [3, 1]
class = F
gini = 0.0
samples = 2
value = [4, 0]
class = F
gini = 0.444
samples = 3
value = [1, 2]
class = T
gini = 0.245
samples = 4
value = [1, 6]
class = T

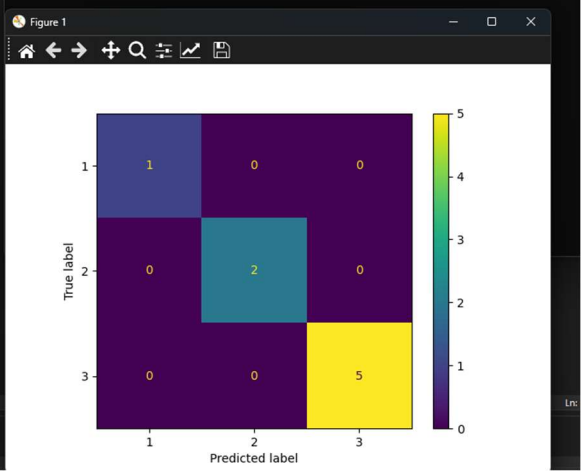

Age <= 0.5
gini = 0.5
samples = 38
value = [30, 30]
class = F
True
False
Size <= 0.5
gini = 0.311
samples = 16
value = [5, 21]
class = T
Color <= 0.5
gini = 0.389
samples = 22
value = [25, 9]
class = F
gini = 0.0
samples = 6
value = [0, 10]
class = T
Color <= 0.5
gini = 0.43
samples = 10
value = [5, 11]
class = T
Size <= 0.5
gini = 0.476
samples = 14
value = [14, 9]
class = F
gini = 0.0
samples = 8
value = [11, 0]
class = F
Act <= 0.5
gini = 0.375
samples = 3
value = [3, 1]
class = F
Act <= 0.5
gini = 0.278
samples = 7
value = [2, 10]
class = T
Act <= 0.5
gini = 0.408
samples = 6
value = [2, 5]
class = T
Act <= 0.5
gini = 0.375
samples = 8
value = [12, 4]
class = F
gini = 0.5
samples = 2
value = [1, 1]
class = F
gini = 0.0
samples = 1
value = [2, 0]
class = F
gini = 0.198
samples = 5
value = [1, 8]
class = T
gini = 0.444
samples = 2
value = [1, 2]
class = T
gini = 0.0
samples = 2
value = [0, 3]
class = T
gini = 0.5
samples = 4
value = [2, 2]
class = F
gini = 0.48
samples = 4
value = [6, 4]
class = F
gini = 0.0
samples = 4
value = [6, 0]
class = F

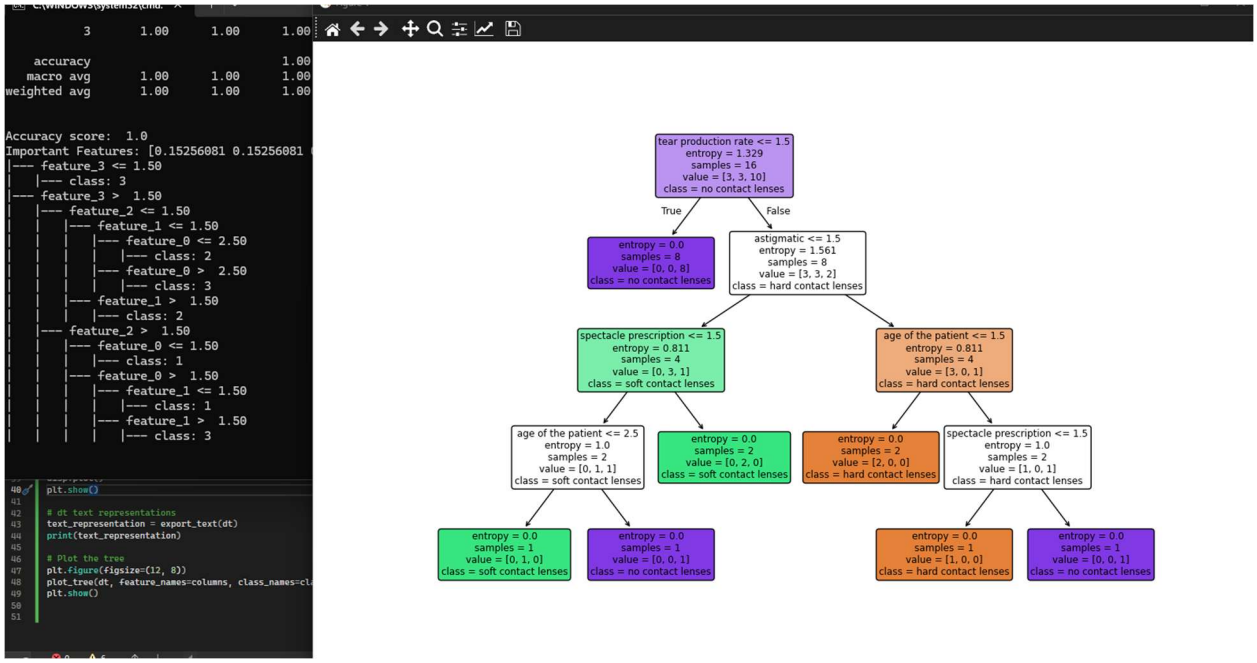


3.

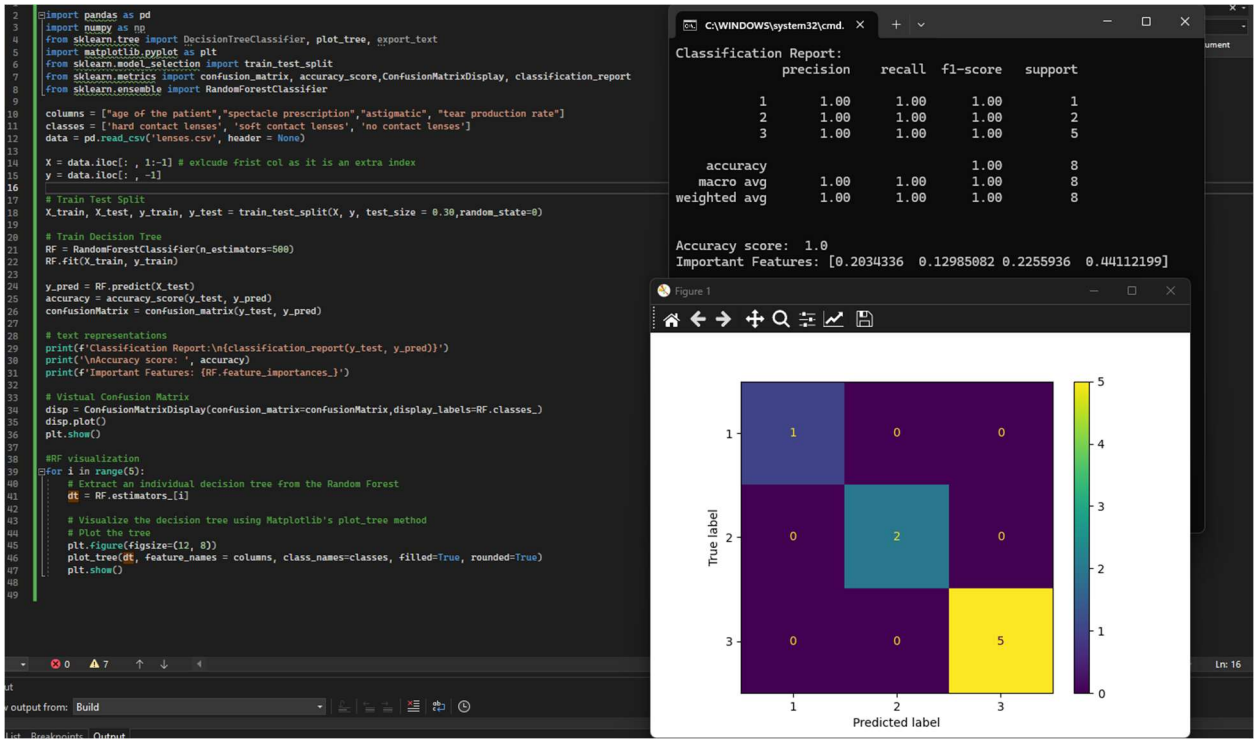
```python
from email import header
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, plot_tree, export_text
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score,ConfusionMatrixDisplay, classification_report


#columns = ["age of the patient","spectacle prescription","astigmatic", "tear production rate"]
classes = ['hard contact lenses', 'soft contact lenses', 'no contact lenses']
data = pd.read_csv('lenses.csv', header = None)

X = data.iloc[: , 1:-1]# exlcude frist col as it is an extra index
y = data.iloc[: , -1]


# Train Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30,random_state=0)

# Train Decision Tree
dt = DecisionTreeClassifier(criterion='entropy', random_state=42)
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
confusionMatrix = confusion_matrix(y_test, y_pred)

# text representations
print(f'Classification Report:\n{classification_report(y_test, y_pred)}')
print('\nAccuracy score: ', accuracy)
print(f'Important Features: {dt.feature_importances_}')


# Visual Confusion Matrix
disp = ConfusionMatrixDisplay(confusion_matrix=confusionMatrix,display_labels=dt.classes_)
disp.plot()
plt.show()

# dt text representations
text_representation = export_text(dt)
print(text_representation)

# Plot the tree
plt.figure(figsize=(12, 8))
plot_tree(dt, feature_names=['Color', 'Size', 'Act', 'Age'], class_names=['F','T'], filled=True, rounded=True)
plt.show()
```

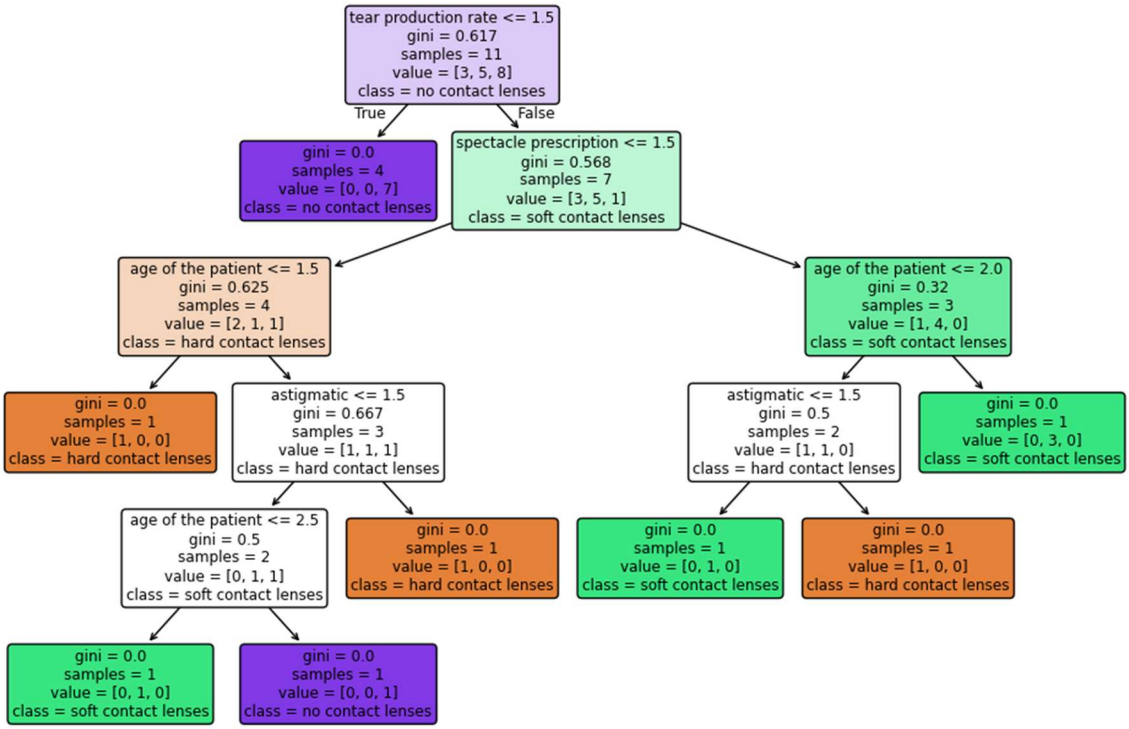
```
C:\WINDOWS\system32\cmd.
Classification Report:
              precision    recall  f1-score   support

           1       1.00      1.00      1.00         1
           2       1.00      1.00      1.00         2
           3       1.00      1.00      1.00         5

    accuracy                           1.00         8
   macro avg       1.00      1.00      1.00         8
weighted avg       1.00      1.00      1.00         8


Accuracy score:  1.0
Important Features: [0.15256081 0.15256081 0.28207493 0.41280345]
```

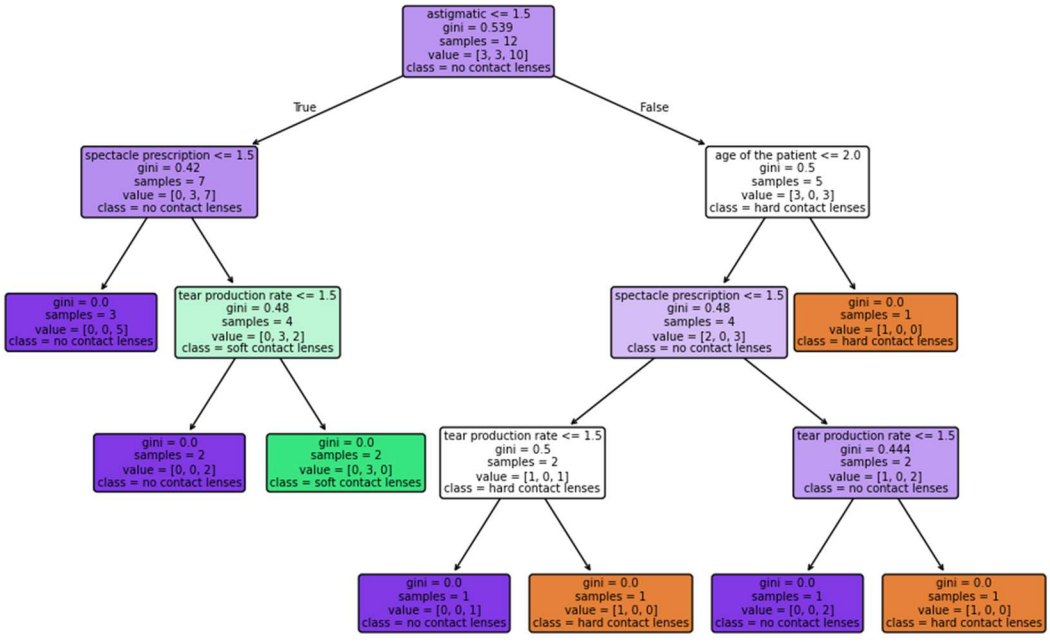


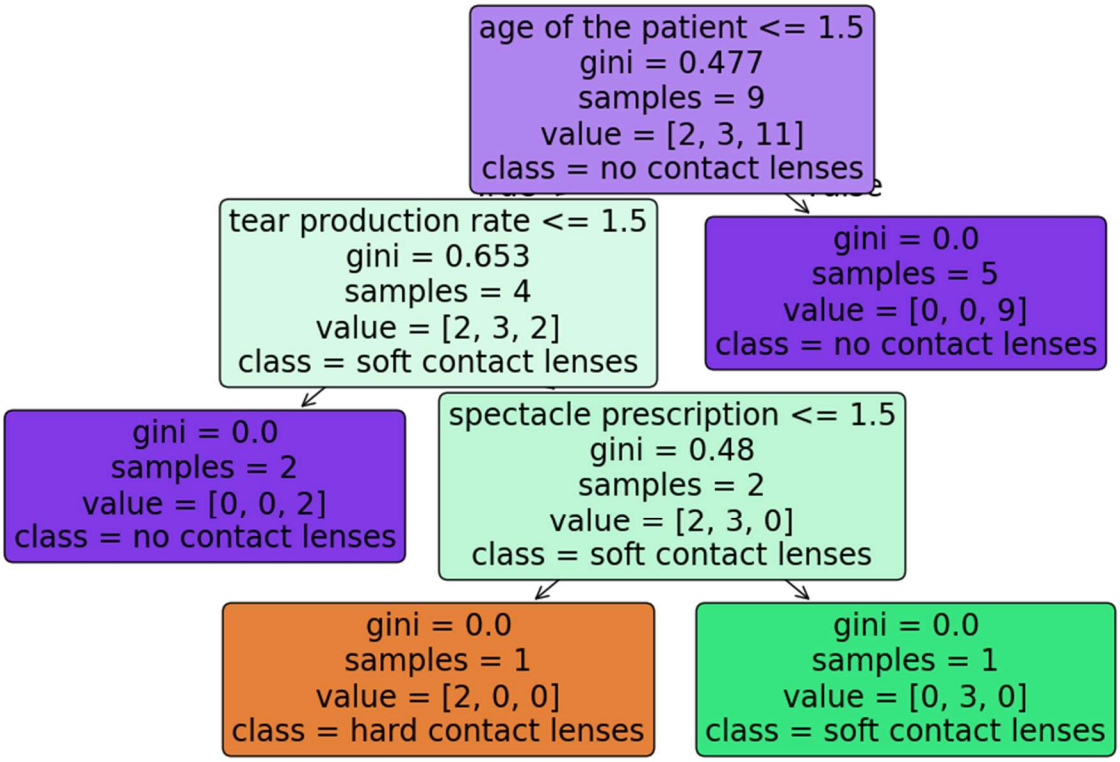

3 1.00 1.00 1.00
accuracy 1.00
macro avg 1.00 1.00 1.00
weighted avg 1.00 1.00 1.00
Accuracy score: 1.0
Important Features: [0.15256081 0.15256081
|--- feature_3 <= 1.50
| |--- class: 3
|--- feature_3 > 1.50
| |--- feature_2 <= 1.50
| | |--- feature_1 <= 1.50
| | | |--- feature_0 <= 2.50
| | | | |--- class: 2
| | | |--- feature_0 > 2.50
| | | | |--- class: 3
| | |--- feature_1 > 1.50
| | | |--- class: 2
| |--- feature_2 > 1.50
| | |--- feature_0 <= 1.50
| | | |--- class: 1
| | |--- feature_0 > 1.50
| | | |--- feature_1 <= 1.50
| | | | |--- class: 1
| | | |--- feature_1 > 1.50
| | | | |--- class: 3
plt.show()
# dt text representations
text_representation = export_text(dt)
print(text_representation)
# Plot the tree
plt.figure(figsize=(12, 8))
plot_tree(dt, feature_names=columns, class_names=cl
plt.show()
tear production rate <= 1.5
entropy = 1.329
samples = 16
value = [3, 3, 10]
class = no contact lenses
True
False
entropy = 0.0
samples = 8
value = [0, 0, 8]
class = no contact lenses
astigmatic <= 1.5
entropy = 1.561
samples = 8
value = [3, 3, 2]
class = hard contact lenses
spectacle prescription <= 1.5
entropy = 0.811
samples = 4
value = [0, 3, 1]
class = soft contact lenses
age of the patient <= 1.5
entropy = 0.811
samples = 4
value = [3, 0, 1]
class = hard contact lenses
age of the patient <= 2.5
entropy = 1.0
samples = 2
value = [0, 1, 1]
class = soft contact lenses
entropy = 0.0
samples = 2
value = [0, 2, 0]
class = soft contact lenses
entropy = 0.0
samples = 2
value = [2, 0, 0]
class = hard contact lenses
spectacle prescription <= 1.5
entropy = 1.0
samples = 2
value = [1, 0, 1]
class = hard contact lenses
entropy = 0.0
samples = 1
value = [0, 1, 0]
class = soft contact lenses
entropy = 0.0
samples = 1
value = [0, 0, 1]
class = no contact lenses
entropy = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
entropy = 0.0
samples = 1
value = [0, 0, 1]
class = no contact lenses

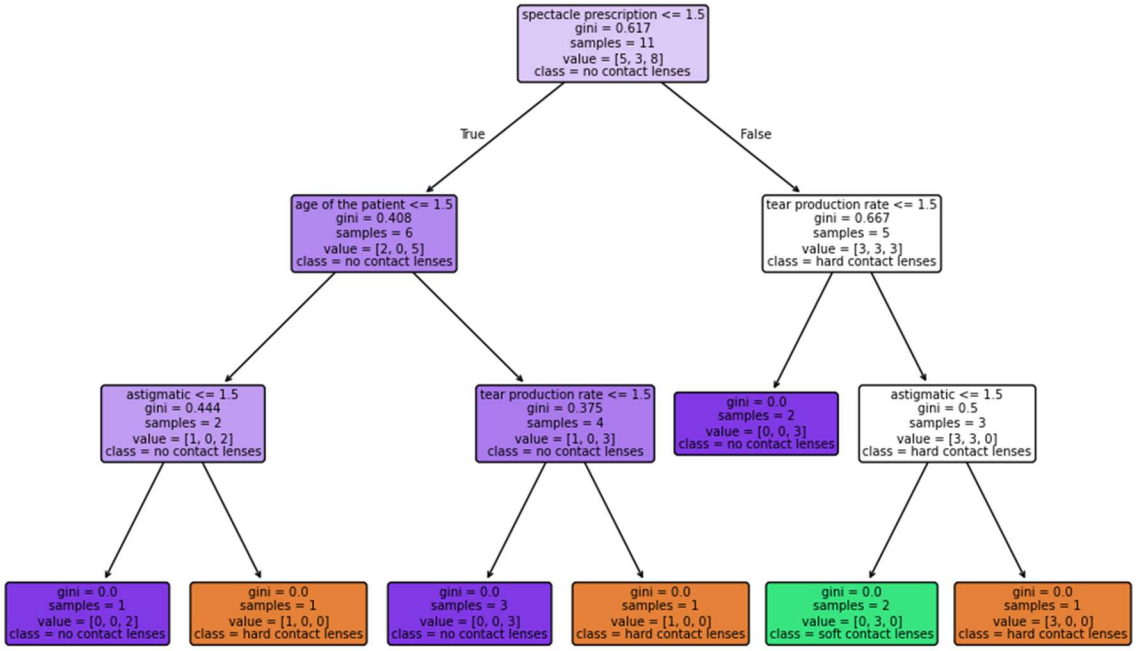
4.

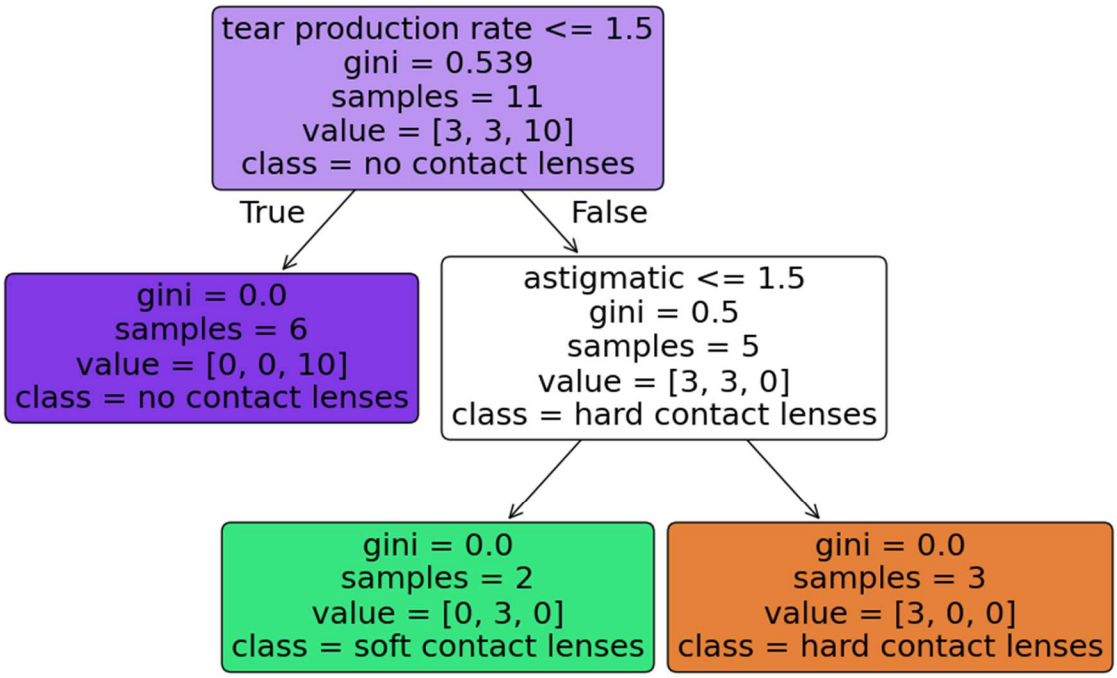

import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, plot_tree, export_text
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score,ConfusionMatrixDisplay, classification_report
from sklearn.ensemble import RandomForestClassifier
columns = ["age of the patient","spectacle prescription","astigmatic", "tear production rate"]
classes = ['hard contact lenses', 'soft contact lenses', 'no contact lenses']
data = pd.read_csv('lenses.csv', header = None)
X = data.iloc[: , 1:-1] # exlcude frist col as it is an extra index
y = data.iloc[: , -1]
# Train Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30,random_state=0)
# Train Decision Tree
RF = RandomForestClassifier(n_estimators=500)
RF.fit(X_train, y_train)
y_pred = RF.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
confusionMatrix = confusion_matrix(y_test, y_pred)
# text representations
print(f'Classification Report:\n{classification_report(y_test, y_pred)}')
print('\nAccuracy score: ', accuracy)
print(f'Important Features: {RF.feature_importances_}')
# Visual Confusion Matrix
disp = ConfusionMatrixDisplay(confusion_matrix=confusionMatrix,display_labels=RF.classes_)
disp.plot()
plt.show()
#RF visualization
for i in range(5):
# Extract an individual decision tree from the Random Forest
dt = RF.estimators_[i]
# Visualize the decision tree using Matplotlib's plot_tree method
# Plot the tree
plt.figure(figsize=(12, 8))
plot_tree(dt, feature_names = columns, class_names=classes, filled=True, rounded=True)
plt.show()
C:\WINDOWS\system32\cmd.
Classification Report:
precision recall f1-score support
1 1.00 1.00 1.00 1
2 1.00 1.00 1.00 2
3 1.00 1.00 1.00 5
accuracy 1.00 8
macro avg 1.00 1.00 1.00 8
weighted avg 1.00 1.00 1.00 8
Accuracy score: 1.0
Important Features: [0.2034336 0.12985082 0.2255936 0.44112199]
Figure 1
True label
Predicted label
1
2
3
1
0
0
0
2
0
0
0
5
5
4
3
2
1
0
Ln: 16
output from: Build


tear production rate <= 1.5
gini = 0.617
samples = 11
value = [3, 5, 8]
class = no contact lenses
True
False
gini = 0.0
samples = 4
value = [0, 0, 7]
class = no contact lenses
spectacle prescription <= 1.5
gini = 0.568
samples = 7
value = [3, 5, 1]
class = soft contact lenses
age of the patient <= 1.5
gini = 0.625
samples = 4
value = [2, 1, 1]
class = hard contact lenses
age of the patient <= 2.0
gini = 0.32
samples = 3
value = [1, 4, 0]
class = soft contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
astigmatic <= 1.5
gini = 0.667
samples = 3
value = [1, 1, 1]
class = hard contact lenses
astigmatic <= 1.5
gini = 0.5
samples = 2
value = [1, 1, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 3, 0]
class = soft contact lenses
age of the patient <= 2.5
gini = 0.5
samples = 2
value = [0, 1, 1]
class = soft contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 1, 0]
class = soft contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 1, 0]
class = soft contact lenses
gini = 0.0
samples = 1
value = [0, 0, 1]
class = no contact lenses


astigmatic <= 1.5
gini = 0.539
samples = 12
value = [3, 3, 10]
class = no contact lenses
True
False
spectacle prescription <= 1.5
gini = 0.42
samples = 7
value = [0, 3, 7]
class = no contact lenses
age of the patient <= 2.0
gini = 0.5
samples = 5
value = [3, 0, 3]
class = hard contact lenses
gini = 0.0
samples = 3
value = [0, 0, 5]
class = no contact lenses
tear production rate <= 1.5
gini = 0.48
samples = 4
value = [0, 3, 2]
class = soft contact lenses
spectacle prescription <= 1.5
gini = 0.48
samples = 4
value = [2, 0, 3]
class = no contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 2
value = [0, 0, 2]
class = no contact lenses
gini = 0.0
samples = 2
value = [0, 3, 0]
class = soft contact lenses
tear production rate <= 1.5
gini = 0.5
samples = 2
value = [1, 0, 1]
class = hard contact lenses
tear production rate <= 1.5
gini = 0.444
samples = 2
value = [1, 0, 2]
class = no contact lenses
gini = 0.0
samples = 1
value = [0, 0, 1]
class = no contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 0, 2]
class = no contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses


age of the patient <= 1.5
gini = 0.477
samples = 9
value = [2, 3, 11]
class = no contact lenses
tear production rate <= 1.5
gini = 0.653
samples = 4
value = [2, 3, 2]
class = soft contact lenses
gini = 0.0
samples = 5
value = [0, 0, 9]
class = no contact lenses
gini = 0.0
samples = 2
value = [0, 0, 2]
class = no contact lenses
spectacle prescription <= 1.5
gini = 0.48
samples = 2
value = [2, 3, 0]
class = soft contact lenses
gini = 0.0
samples = 1
value = [2, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 3, 0]
class = soft contact lenses


spectacle prescription <= 1.5
gini = 0.617
samples = 11
value = [5, 3, 8]
class = no contact lenses
True
False
age of the patient <= 1.5
gini = 0.408
samples = 6
value = [2, 0, 5]
class = no contact lenses
tear production rate <= 1.5
gini = 0.667
samples = 5
value = [3, 3, 3]
class = hard contact lenses
astigmatic <= 1.5
gini = 0.444
samples = 2
value = [1, 0, 2]
class = no contact lenses
tear production rate <= 1.5
gini = 0.375
samples = 4
value = [1, 0, 3]
class = no contact lenses
gini = 0.0
samples = 2
value = [0, 0, 3]
class = no contact lenses
astigmatic <= 1.5
gini = 0.5
samples = 3
value = [3, 3, 0]
class = hard contact lenses
gini = 0.0
samples = 1
value = [0, 0, 2]
class = no contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 3
value = [0, 0, 3]
class = no contact lenses
gini = 0.0
samples = 1
value = [1, 0, 0]
class = hard contact lenses
gini = 0.0
samples = 2
value = [0, 3, 0]
class = soft contact lenses
gini = 0.0
samples = 1
value = [3, 0, 0]
class = hard contact lenses


tear production rate <= 1.5
gini = 0.539
samples = 11
value = [3, 3, 10]
class = no contact lenses
True
False
gini = 0.0
samples = 6
value = [0, 0, 10]
class = no contact lenses
astigmatic <= 1.5
gini = 0.5
samples = 5
value = [3, 3, 0]
class = hard contact lenses
gini = 0.0
samples = 2
value = [0, 3, 0]
class = soft contact lenses
gini = 0.0
samples = 3
value = [3, 0, 0]
class = hard contact lenses