HW - Decision Trees

Please follow the instructions given during the class/demo

Load sklearn's wine dataset and perform classification using Decision trees. Try different pruning techniques, criteria to split the tree which gives you best results. Plot the tree and comment on your findings. Also plot the feature importances of each tree.

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html)

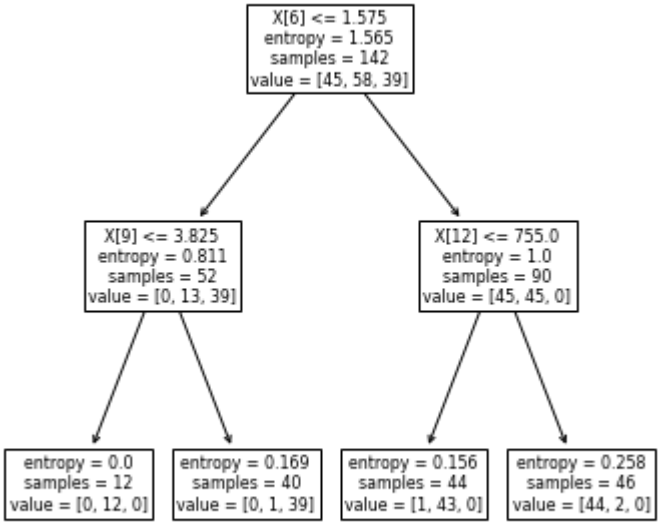
In [123]:
```
from sklearn.datasets import load_wine
from  sklearn.model_selection import train_test_split
from sklearn import tree
import numpy as np
from sklearn import metrics
import matplotlib.pyplot as plt

data = load_wine()
X = data.data
Y = data.target

#print(len(data.feature_names))
#print(data.target_names)
#print(X)
#print(Y)

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_s
```

In [138]:
```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(criterion="entropy",max_depth=2)
model = model.fit(x_train,y_train)
y_pred = model.predict(x_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9666666666666667
```

In [125]:
```
fig = plt.figure(figsize=(6,6))
tree.plot_tree(model)
```

Out[125]:
```
[Text(167.4, 271.8, 'X[6] <= 1.575\nentropy = 1.565\nsamples = 142\nvalue = [45, 58, 39]'),
 Text(83.7, 163.08, 'X[9] <= 3.825\nentropy = 0.811\nsamples = 52\nvalue = [0, 13, 39]'),
 Text(41.85, 54.360000000000014, 'entropy = 0.0\nsamples = 12\nvalue = [0, 12, 0]'),
 Text(125.55000000000001, 54.360000000000014, 'entropy = 0.169\nsamples = 40\nvalue = [0, 1, 39]'),
 Text(251.10000000000002, 163.08, 'X[12] <= 755.0\nentropy = 1.0\nsamples = 90\nvalue = [45, 45, 0]'),
 Text(209.25, 54.360000000000014, 'entropy = 0.156\nsamples = 44\nvalue = [1, 43, 0]'),
 Text(292.95, 54.360000000000014, 'entropy = 0.258\nsamples = 46\nvalue = [44, 2, 0]')]
```

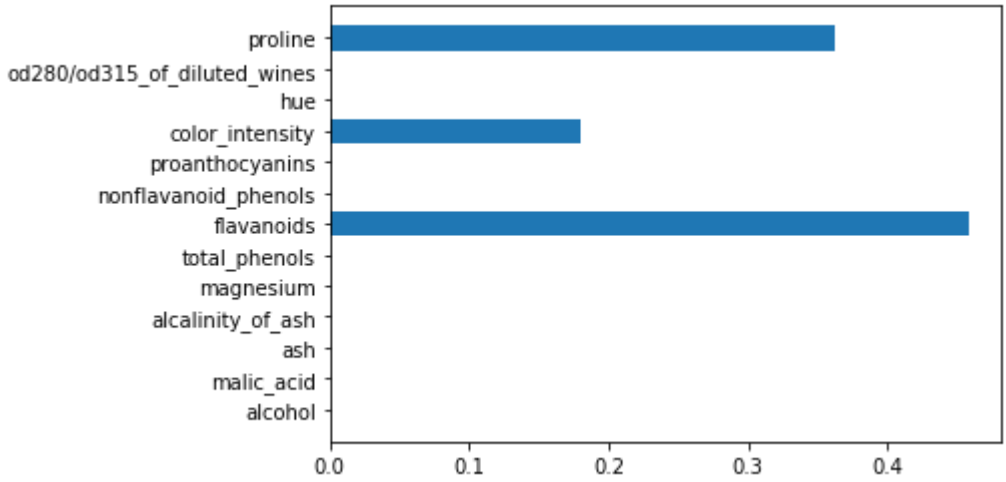


In [126]:
```
model.feature_importances_
```

Out[126]:
```
array([0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.45772282, 0.        , 0.        , 0.18013981,
       0.        , 0.        , 0.36213737])
```

In [127]:
```
plt.barh(data.feature_names, model.feature_importances_)
```

Out[127]: <BarContainer object of 13 artists>

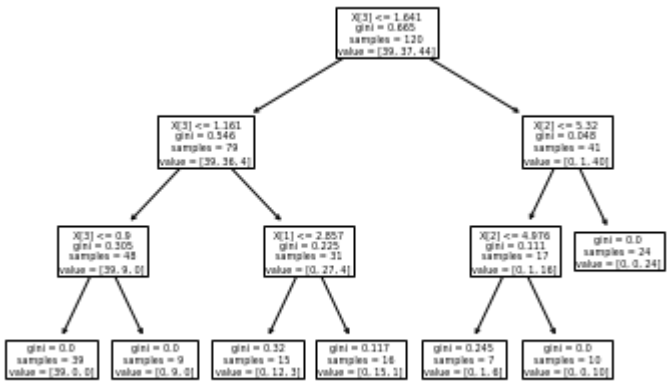


In [141]:
```
model = DecisionTreeClassifier(criterion="gini",splitter="random", max_depth=3)
model = model.fit(x_train,y_train)
y_pred = model.predict(x_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9666666666666667
```

In [142]:
```
tree.plot_tree(model)
```

Out[142]:
```
[Text(193.15384615384616, 190.26, 'X[3] <= 1.641\ngini = 0.665\nsamples = 120\n
value = [39, 37, 44]'),
 Text(103.01538461538462, 135.9, 'X[3] <= 1.161\ngini = 0.546\nsamples = 79\nva
lue = [39, 36, 4]'),
 Text(51.50769230769231, 81.53999999999999, 'X[3] <= 0.9\ngini = 0.305\nsamples
= 48\nvalue = [39, 9, 0]'),
 Text(25.753846153846155, 27.180000000000007, 'gini = 0.0\nsamples = 39\nvalue
= [39, 0, 0]'),
 Text(77.26153846153846, 27.180000000000007, 'gini = 0.0\nsamples = 9\nvalue =
[0, 9, 0]'),
 Text(154.52307692307693, 81.53999999999999, 'X[1] <= 2.857\ngini = 0.225\nsamp
les = 31\nvalue = [0, 27, 4]'),
 Text(128.76923076923077, 27.180000000000007, 'gini = 0.32\nsamples = 15\nvalue
= [0, 12, 3]'),
 Text(180.27692307692308, 27.180000000000007, 'gini = 0.117\nsamples = 16\nvalu
e = [0, 15, 1]'),
 Text(283.2923076923077, 135.9, 'X[2] <= 5.32\ngini = 0.048\nsamples = 41\nvalu
e = [0, 1, 40]'),
 Text(257.53846153846155, 81.53999999999999, 'X[2] <= 4.976\ngini = 0.111\nsamp
les = 17\nvalue = [0, 1, 16]'),
 Text(231.7846153846154, 27.180000000000007, 'gini = 0.245\nsamples = 7\nvalue
= [0, 1, 6]'),
 Text(283.2923076923077, 27.180000000000007, 'gini = 0.0\nsamples = 10\nvalue =
[0, 0, 10]'),
 Text(309.04615384615386, 81.53999999999999, 'gini = 0.0\nsamples = 24\nvalue =
[0, 0, 24]')]
```

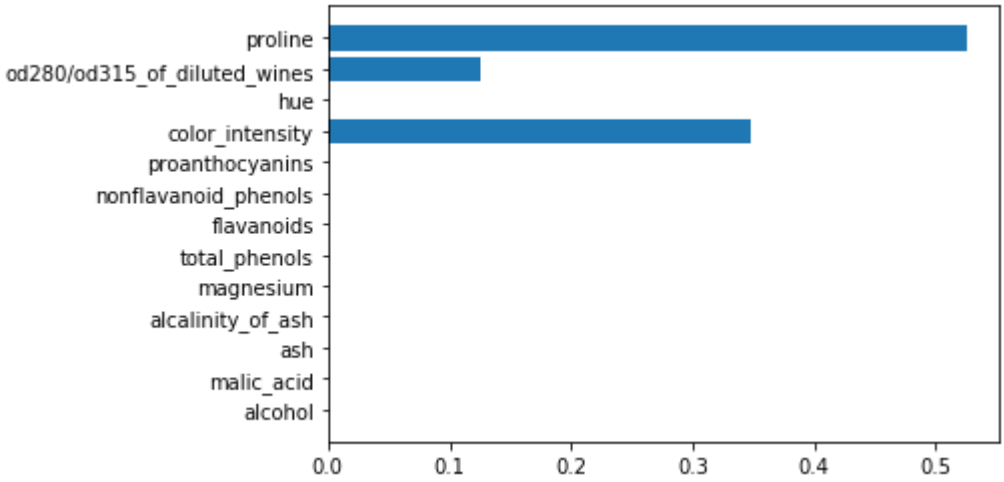


In [130]:
```
model.feature_importances_
```

Out[130]:
```
array([0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.34861755,
       0.        , 0.12620213, 0.52518031])
```

In [131]:
```
plt.barh(data.feature_names, model.feature_importances_)
```

Out[131]: <BarContainer object of 13 artists>



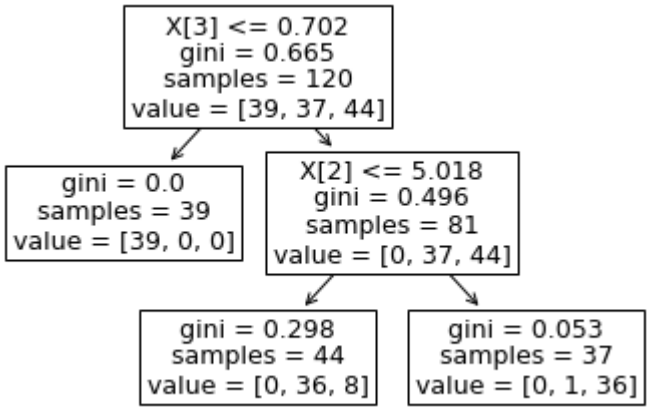
In [132]:
```
model = DecisionTreeClassifier(criterion="gini",splitter="random", max_depth=3)
model = model.fit(x_train,y_train)
y_pred = model.predict(x_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9166666666666666
```

In [145]:
```
model = DecisionTreeClassifier(criterion="gini",splitter="random", max_depth=3,mi
model = model.fit(x_train,y_train)
y_pred = model.predict(x_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
tree.plot_tree(model)
```

```
Accuracy: 0.9666666666666667
```

Out[145]:
```
[Text(133.92000000000002, 181.2, 'X[3] <= 0.702\ngini = 0.665\nsamples = 120\nv
alue = [39, 37, 44]'),
 Text(66.96000000000001, 108.72, 'gini = 0.0\nsamples = 39\nvalue = [39, 0,
0]'),
 Text(200.88000000000002, 108.72, 'X[2] <= 5.018\ngini = 0.496\nsamples = 81\nv
alue = [0, 37, 44]'),
 Text(133.92000000000002, 36.23999999999998, 'gini = 0.298\nsamples = 44\nvalue
= [0, 36, 8]'),
 Text(267.84000000000003, 36.23999999999998, 'gini = 0.053\nsamples = 37\nvalue
= [0, 1, 36]')]
```



HW - Random Forest

Load sklearn's iris dataset and perform classification using Random Forest. Also plot the feature importances. Write down your comments.

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)

In [134]:
```
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier

data = load_iris()
X = data.data
Y = data.target

#print(len(data.feature_names))
#print(Y)


x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_s
```

In [135]:
```
model = RandomForestClassifier(criterion="entropy", n_estimators=5)
model.fit(x_train,y_train)
y_pred = model.predict(x_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

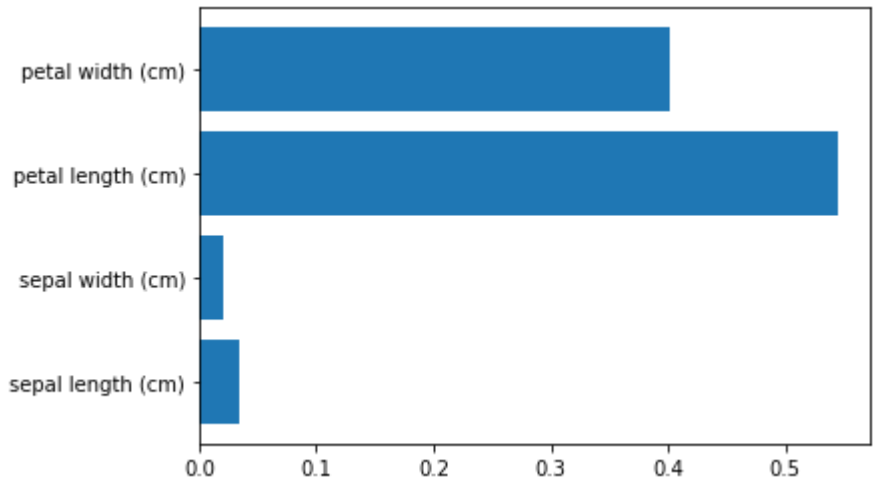
```
Accuracy: 0.9666666666666667
```

In [136]:
```
model.feature_importances_
```

Out[136]:
```
array([0.03430188, 0.02024956, 0.54462578, 0.40082278])
```

In [137]:
```
plt.barh(data.feature_names, model.feature_importances_)
```

Out[137]:
```
<BarContainer object of 4 artists>
```



We notice that petal length is most influencing the decision as it is most important feature followed by petal width and when n_estimator is set to 1 Random forest and Decision tree have same accuracy, as we go on increasing n_estimators the accuracy increases

Write four differences between Random Forests and Decision Trees ¶

Decision tree relies on a singular decision while Random Forest assembles randomized decisions from decision trees in forest and makes the final decision based on the majority.

Decision trees are easy to build and fast while Random forests are more precise, requires more time for building and training as they undergo rigorous training hence slow.

Decision tree is build on whole dataset while Random forest picks up random rows and features to build multiple decision trees and averages the outputs

Random forest reduces the variance part of error unlike decision tree which reduces the bias part. Random forest gives better accuracy on unexpected validation datasets while decision tree gives more accuracy when test dataset is sure to be part of the training dataset or overlapping.

In [ ]: