Installation of Packages

First install packages like numpy, scikit-learn, matplotlib

In [ ]:

```
!pip install numpy scikit-learn matplotlib graphviz pydotplus
```

```
Requirement already satisfied: numpy in /home/john/contributions/.venv/lib/pytho
n3.10/site-packages (1.23.2)
Requirement already satisfied: scikit-learn in /home/john/contributions/.venv/li
b/python3.10/site-packages (1.1.2)
Requirement already satisfied: matplotlib in /home/john/contributions/.venv/lib/
python3.10/site-packages (3.6.1)
Requirement already satisfied: graphviz in /home/john/contributions/.venv/lib/py
thon3.10/site-packages (0.20.1)
Collecting pydotplus
  Using cached pydotplus-2.0.2.tar.gz (278 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: threadpoolctl>=2.0.0 in /home/john/contribution
s/.venv/lib/python3.10/site-packages (from scikit-learn) (3.1.0)
Requirement already satisfied: joblib>=1.0.0 in /home/john/contributions/.venv/l
ib/python3.10/site-packages (from scikit-learn) (1.1.0)
Requirement already satisfied: scipy>=1.3.2 in /home/john/contributions/.venv/li
b/python3.10/site-packages (from scikit-learn) (1.9.1)
Requirement already satisfied: python-dateutil>=2.7 in /home/john/contribution
s/.venv/lib/python3.10/site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: packaging>=20.0 in /home/john/contributions/.ven
v/lib/python3.10/site-packages (from matplotlib) (21.3)
Requirement already satisfied: fonttools>=4.22.0 in /home/john/contributions/.ve
nv/lib/python3.10/site-packages (from matplotlib) (4.37.4)
Requirement already satisfied: pillow>=6.2.0 in /home/john/contributions/.venv/l
ib/python3.10/site-packages (from matplotlib) (9.2.0)
Requirement already satisfied: pyparsing>=2.2.1 in /home/john/contributions/.ven
v/lib/python3.10/site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: contourpy>=1.0.1 in /home/john/contributions/.ven
v/lib/python3.10/site-packages (from matplotlib) (1.0.5)
Requirement already satisfied: cycler>=0.10 in /home/john/contributions/.venv/li
b/python3.10/site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /home/john/contributions/.ve
nv/lib/python3.10/site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: six>=1.5 in /home/john/contributions/.venv/lib/py
thon3.10/site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Using legacy 'setup.py install' for pydotplus, since package 'wheel' is not inst
alled.
Installing collected packages: pydotplus
  Running setup.py install for pydotplus ... done
Successfully installed pydotplus-2.0.2
```

Importation of packages

We import the necessary packages

In [ ]:

```
import numpy as np
from sklearn import datasets, metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plot
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn import tree
import graphviz
import pydotplus
from IPython.display import Image, display
```

Load Dataset

We load the necessary IRIS dataset.

In [ ]:

```
wine = datasets.load_wine()
```

Description of the Dataset

Input features

In [ ]:

```
wine.feature_names
```

Out[ ]:

```
['alcohol',
 'malic_acid',
 'ash',
 'alcalinity_of_ash',
 'magnesium',
 'total_phenols',
 'flavanoids',
 'nonflavanoid_phenols',
 'proanthocyanins',
 'color_intensity',
 'hue',
 'od280/od315_of_diluted_wines',
 'proline']
```

Target feature

In [ ]:

```
wine.target_names
```

Out[ ]:

```
array(['class_0', 'class_1', 'class_2'], dtype='<U7')
```

Verify number of records

In [ ]:

```
print(f"Number of Input Records: {len(wine.data)}")
print(f"Number of Target Records: {len(wine.target)}")
```

```
Number of Input Records: 178
Number of Target Records: 178
```

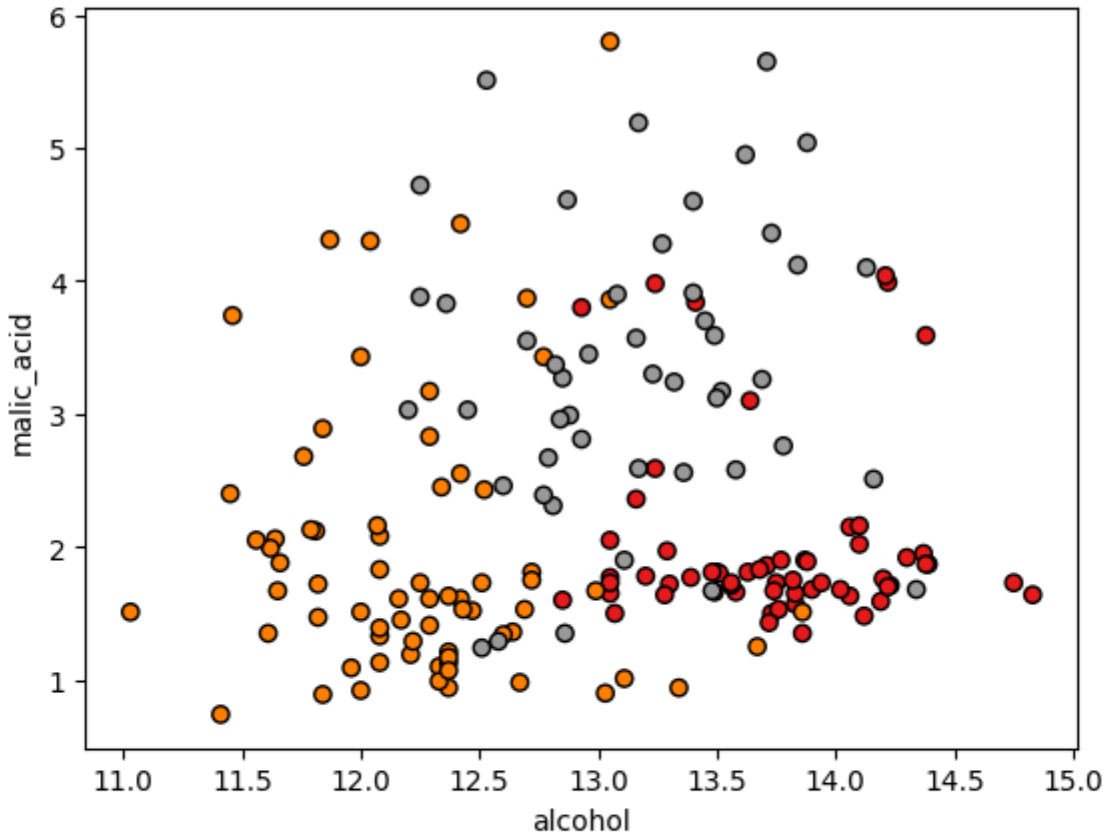
Visulizing the dataset

In [ ]:

```
x = wine.data
y = wine.target
```

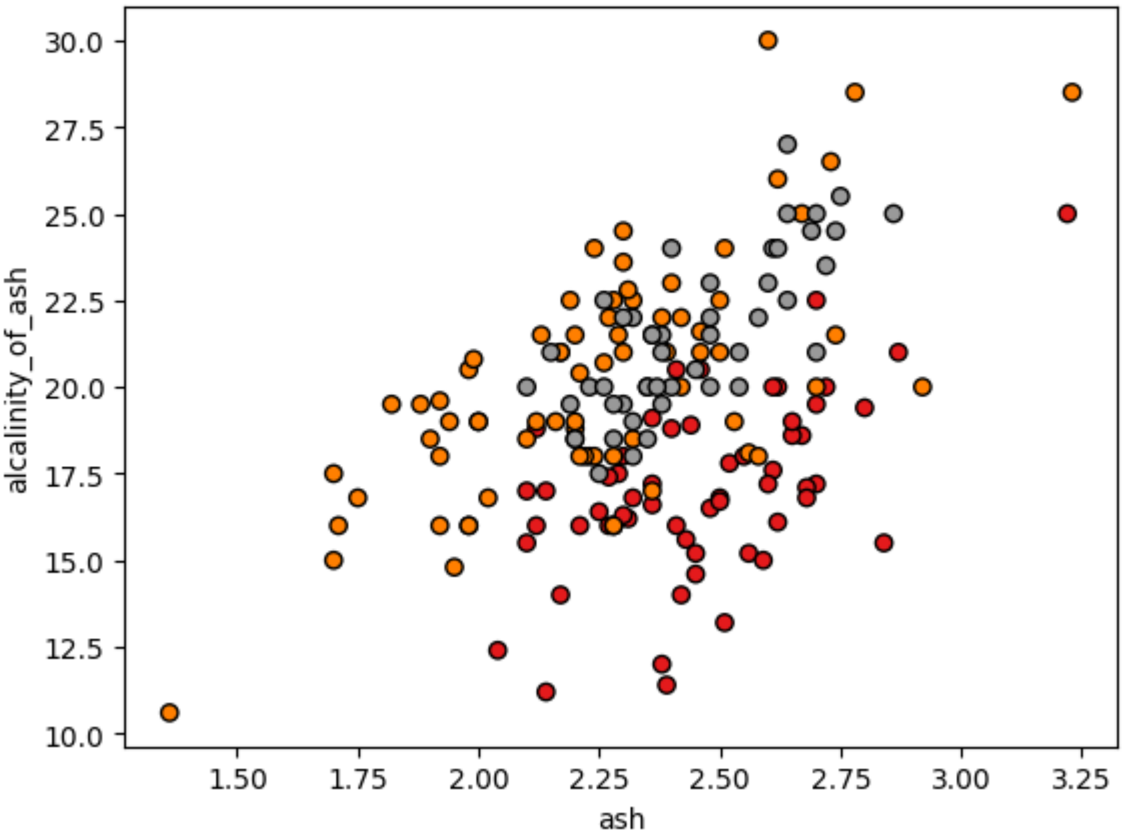
In [ ]:

```
plot.scatter(x[:, 0], x[:, 1], c=y, cmap=plot.cm.Set1, edgecolor="k")
plot.xlabel(wine.feature_names[0])
plot.ylabel(wine.feature_names[1])
plot.show()
```



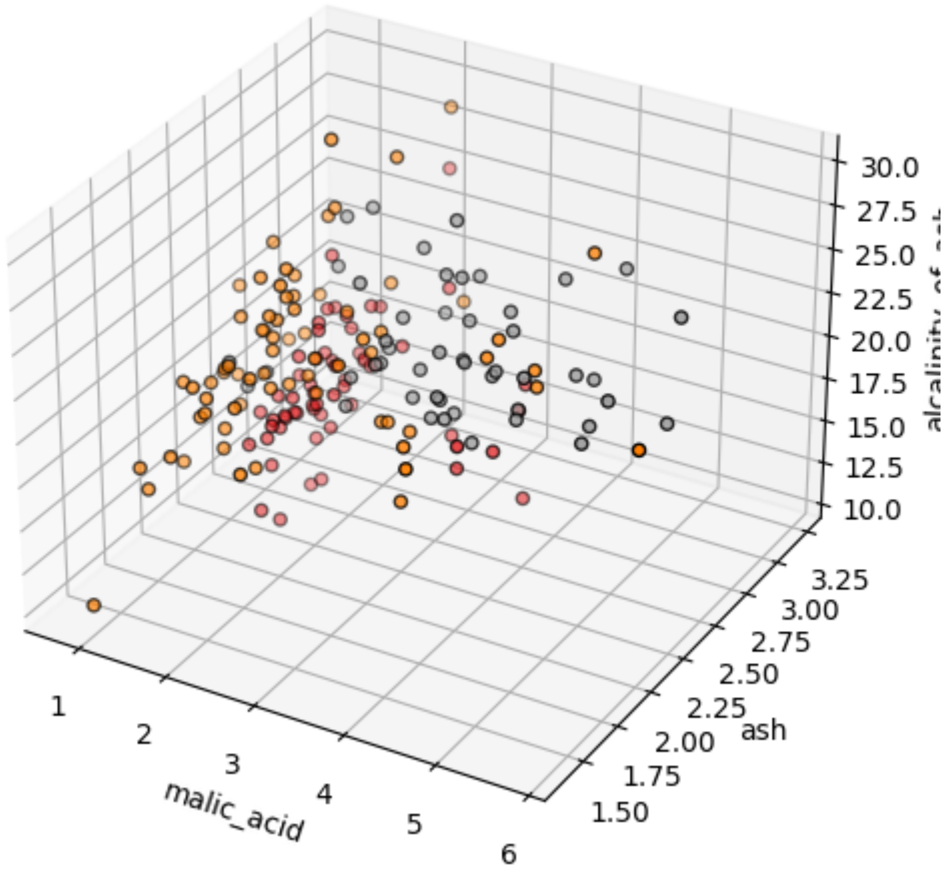
In [ ]:

```
plot.scatter(x[:, 2], x[:, 3], c=y, cmap=plot.cm.Set1, edgecolor="k")
plot.xlabel(wine.feature_names[2])
plot.ylabel(wine.feature_names[3])
plot.show()
```



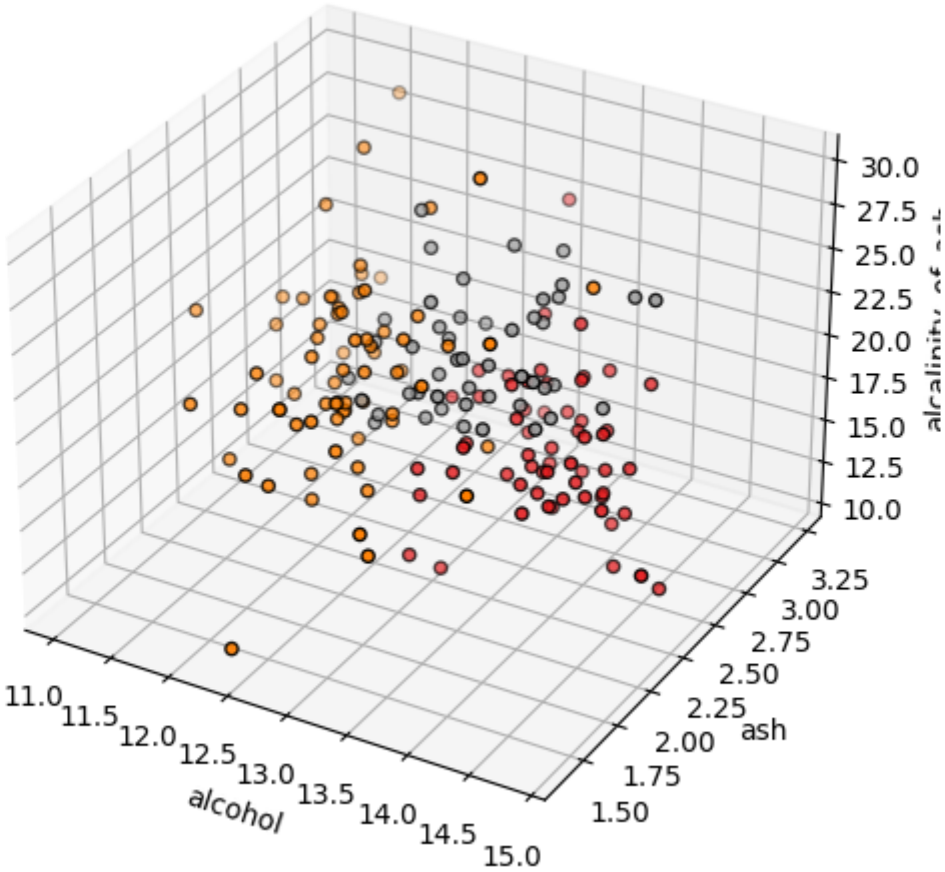
In [ ]:

```
fig = plot.figure(figsize=(6, 6))
ax = fig.add_subplot(projection="3d")

ax.scatter(x[:, 1], x[:, 2], x[:, 3], c=y, cmap=plot.cm.Set1, edgecolor="k")
ax.set_xlabel(wine.feature_names[1])
ax.set_ylabel(wine.feature_names[2])
ax.set_zlabel(wine.feature_names[3])
plot.show()
```



In [ ]:

```
fig = plot.figure(figsize=(6, 6))
ax = fig.add_subplot(projection="3d")

ax.scatter(x[:, 0], x[:, 2], x[:, 3], c=y, cmap=plot.cm.Set1, edgecolor="k")
ax.set_xlabel(wine.feature_names[0])
ax.set_ylabel(wine.feature_names[2])
ax.set_zlabel(wine.feature_names[3])
plot.show()
```



Training

In [ ]:
```
x = wine.data
y = wine.target

x_train, x_test, y_train, y_test = train_test_split(
    x, y, train_size=0.7, random_state=12, stratify=y
)
```

In [ ]:
```
print(f"Number of Training Records (input): {len(x_train)}")
print(f"Number of Training Records (target): {len(y_train)}")

print(f"Number of Test Records (input): {len(x_test)}")
print(f"Number of Test Records (input): {len(x_test)}")
```

```
Number of Training Records (input): 124
Number of Training Records (target): 124
Number of Test Records (input): 54
Number of Test Records (input): 54
```

Standardization of features

In [ ]:
```
sc = StandardScaler()
sc.fit(x_train)
print(f"Mean: {sc.mean_} \nVariance={sc.var_}")
```

```
Mean: [1.30047581e+01 2.37379032e+00 2.35193548e+00 1.94088710e+01
 9.95161290e+01 2.29951613e+00 2.00870968e+00 3.52903226e-01
 1.58637097e+00 4.97782257e+00 9.67870968e-01 2.62653226e+00
 7.34419355e+02]
Variance=[6.14365264e-01 1.36032838e+00 7.36639958e-02 1.13516149e+01
 2.25798127e+02 3.83523959e-01 9.57067690e-01 1.61738293e-02
 3.23856991e-01 5.47239437e+00 5.11881769e-02 4.75337168e-01
 8.90159532e+04]
```

In [ ]:

```
x_train_std = sc.transform(x_train)
x_test_std = sc.transform(x_test)
```

In [ ]:

```
classifier = tree.DecisionTreeClassifier()

# training
classifier.fit(x_train_std, y_train)
```

Out[ ]:

```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

Classification report

In [ ]:

```
predicted_target = classifier.predict(x_test_std)

# classification report
print(metrics.classification_report(y_test, predicted_target))
```

```
              precision    recall  f1-score   support

           0       1.00      0.94      0.97        18
           1       0.91      0.95      0.93        21
           2       0.93      0.93      0.93        15

    accuracy                           0.94        54
   macro avg       0.95      0.94      0.94        54
weighted avg       0.95      0.94      0.94        54
```

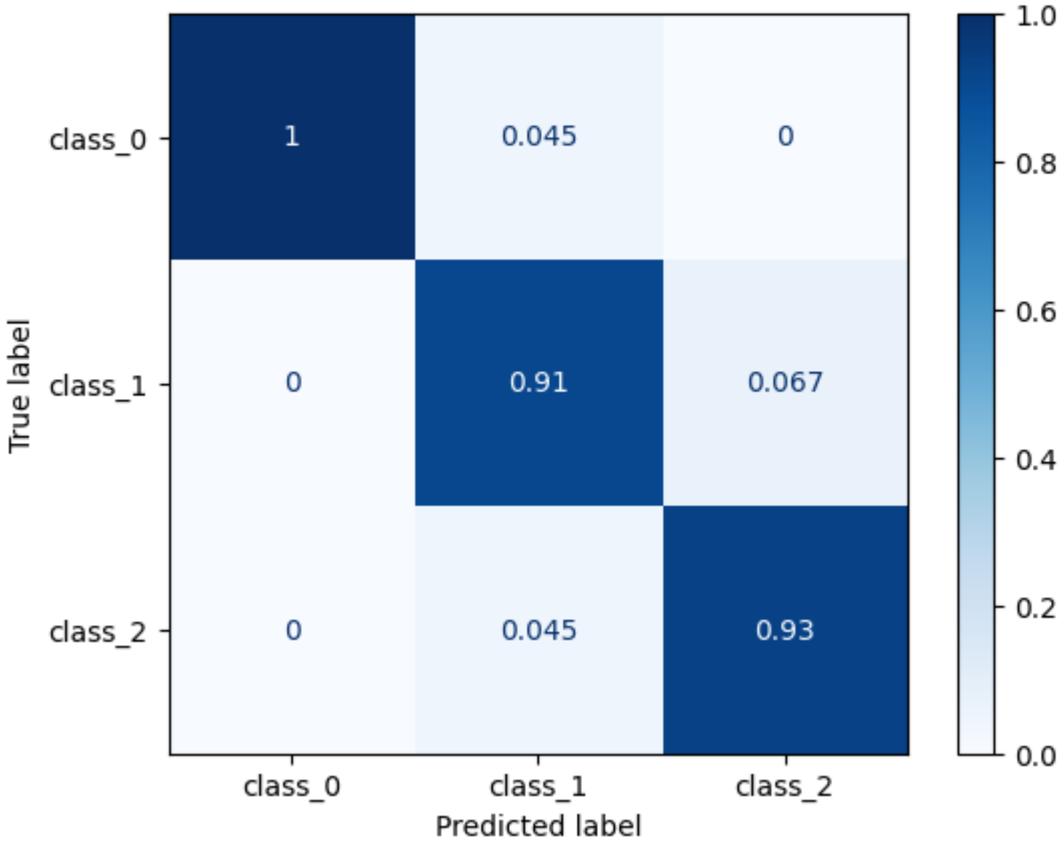
Confusion matrix

In [ ]:

```
cm = confusion_matrix(y_test, predicted_target, normalize="pred")
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=wine.target_na
disp.plot(cmap=plot.cm.Blues)
```

Out[ ]:

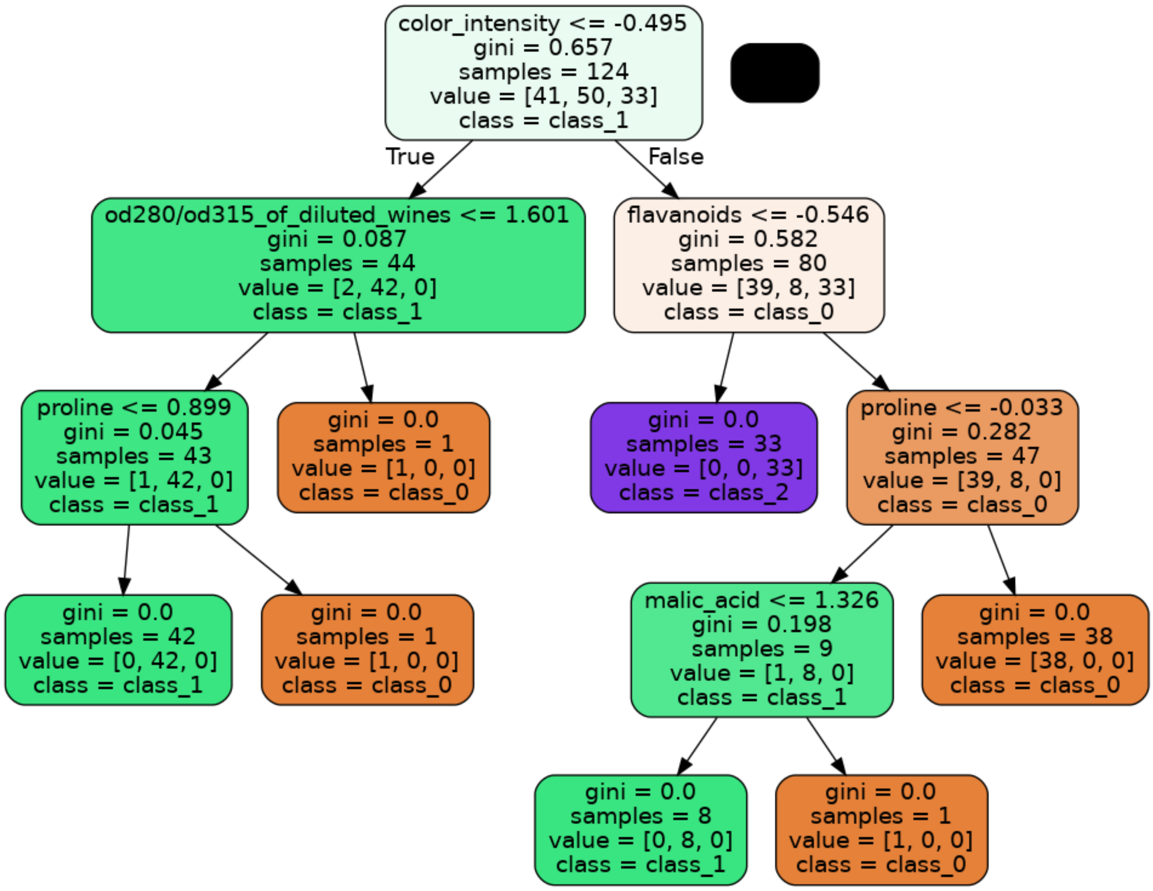
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fe336d114b0
>
```



```
In [ ]: cm = confusion_matrix(y_test, predicted_target, normalize="true")
        disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=wine.target_na
        disp.plot(cmap=plot.cm.Blues)
```

Visualization of Decision tree

```
In [ ]: dot_data = tree.export_graphviz(
            classifier,
            out_file=None,
            feature_names=wine.feature_names,
            filled=True,
            rounded=True,
            class_names=wine.target_names,
        )
        graph = graphviz.Source(dot_data)
        pydot_graph = pydotplus.graph_from_dot_data(dot_data)
        img = Image(pydot_graph.create_png())
        display(img)
```



References

- The Iris Dataset
- 3D scatterplot
- sklearn.preprocessing.StandardScaler
- sklearn.model_selection.train_test_split
- Iris classification with sklearn perceptron
- plot_confusion_matrix without estimator
- sklearn.neural_network.MLPClassifier