```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Example 5.1 Decision Tree Classifier

@author: Austin R.J. Downey
"""

import IPython as IP
IP.get_ipython().run_line_magic('reset', '-sf')

import numpy as np
import matplotlib.pyplot as plt
import sklearn as sk
from sklearn.datasets import load_iris
from sklearn.tree import export_graphviz
import graphviz as graphviz

plt.close('all')


#%% Load your data

# We will use the Iris data set.  This dataset was created by biologist Ronald
# Fisher in his 1936 paper "The use of multiple measurements in taxonomic
# problems" as an example of linear discriminant analysis
iris = sk.datasets.load_iris()

# for simplicity, extract some of the data sets
X = iris['data'] # this contains the length of the petals and sepals
Y = iris['target'] # contains what type of flower it is
Y_names = iris['target_names'] # contains the name that aligns with the type of the
flower
feature_names = iris['feature_names'] # the names of the features

#%% Build the model

# train the decision tree
tree_clf = sk.tree.DecisionTreeClassifier(max_depth=3)
X_petal = X[:,2:]
tree_clf.fit(X_petal, Y)


#%% Visualize the decision tree

#create the export file for graphviz and export it. The file is exported as a
#.DOT file and can be viewed in an online viewer
https://dreampuf.github.io/GraphvizOnline/
export_graphviz(
        tree_clf,
        out_file="tree_clf.dot",
        feature_names=feature_names[2:],
        class_names=Y_names,
        rounded=True,
        filled=True
    )

# We can load the file back in
s = graphviz.Source.from_file('tree_clf.dot')

# look at what is inside it. Also, just typing s in the console will diplay the image
print(s)

# export the image to a jpg
s.render('tree_clf', format='jpg',view=True)

#%% Predict the class for any petal size
```

```
66
67  size = [[7, 2.5]]
68  print(tree_clf.predict_proba(size))
69  print(iris.target_names)
70
71
72  # plot the new data point over the Iris dataset
73  plt.figure()
74  plt.grid(True)
75  plt.scatter(X[Y==0,2],X[Y==0,3],marker='o',label=Y_names[0],zorder=2)
76  plt.scatter(X[Y==1,2],X[Y==1,3],marker='s',label=Y_names[1],zorder=2)
77  plt.scatter(X[Y==2,2],X[Y==2,3],marker='d',label=Y_names[2],zorder=2)
78  plt.scatter(size[0][0],size[0][1],s=300,marker='*',label='new data point',zorder=2)
79  plt.xlabel(feature_names[2])
80  plt.ylabel(feature_names[3])
81  plt.legend(framealpha=1)
82  plt.tight_layout()
83
84
85
86
87
```