Chapter 7: Demo RandomForestClassifier

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
# %cd '/content/gdrive/My Drive/LDS6_MachineLearning/practice/Chapter7_Random_For
```

In [2]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn import datasets
from IPython.display import Image
from sklearn import tree
import pydotplus
import pandas as pd
```

In [3]:
```
iris = pd.read_excel("Iris.xls")
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepallength    150 non-null float64
sepalwidth     150 non-null float64
petallength    150 non-null float64
petalwidth     150 non-null float64
iris           150 non-null object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [4]:
```
X = iris[['sepallength', 'sepalwidth', 'petallength', 'petalwidth']]
y = iris['iris']
```

In [5]:
```
X.head()
```

Out[5]:

	sepallength	sepalwidth	petallength	petalwidth
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
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [6]:
```
y[:5]
```

Out[6]:
```
0    Iris-setosa
1    Iris-setosa
2    Iris-setosa
3    Iris-setosa
4    Iris-setosa
Name: iris, dtype: object
```

In [7]:
```
from sklearn.model_selection import train_test_split
```

In [8]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.3)
```

In [9]:
```
clf = RandomForestClassifier(n_estimators=100) # so cay trong rung
# lam voi: 30, 50, 70, 100, 150, 200 => chon Rung phu hop theo so luong cay (scor
# thuc hien vong lap theo so cay trong rung lst_cay = [30, 50, 70, 100]
# => tao rung theo tung gia tri trong lst_cay
# => do do chinh xac acc, neu do chinh xac = nhau => so sanh train/test r^2
# => chon model random forest co do chinh xac cao nhat, chenh lech giua train/tes
# Train model
model = clf.fit(X_train, y_train)
```

In [10]:
```
# How to show information of trees in random forest
model.estimators_[:3] # 0..99
```

Out[10]:
```
[DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                        max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, presort=False,
                        random_state=1912111053, splitter='best'),
 DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                        max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, presort=False,
                        random_state=995370552, splitter='best'),
 DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                        max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, presort=False,
                        random_state=704366320, splitter='best')]
```

In [11]:
```
y_pred = model.predict(X_test)
```

In [12]:
```
from sklearn import metrics
```

In [13]:
```
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9555555555555556
```

In [14]:
```
# Make new prediction
import numpy as np
X_new = np.array([[4.7, 3.2, 1.3, 0.2],
                  [6.6, 3. , 4.4, 1.4],
                  [5.9, 3. , 5.1, 1.8]])
```

In [15]:
```
yhat_new = model.predict(X_new)
yhat_new
```

Out[15]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

Find important features in sklearn

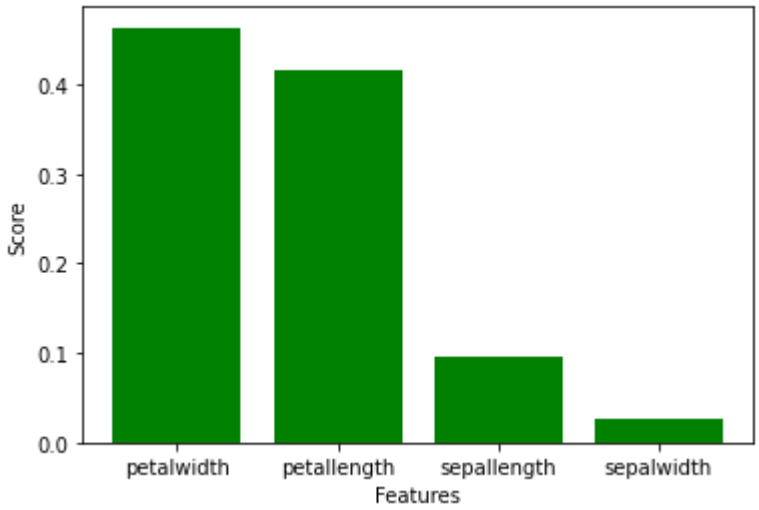
In [16]:
```
imp_features = pd.Series(model.feature_importances_,
                         index=X.columns).sort_values(ascending = False)
```

In [17]:
```
imp_features
```

Out[17]:
```
petalwidth     0.463280
petallength    0.415965
sepallength    0.094915
sepalwidth     0.025840
dtype: float64
```

In [18]:
```
import matplotlib.pyplot as plt
```

In [19]:
```
plt.bar(imp_features.index, imp_features, color="g")
plt.xlabel("Features")
plt.ylabel("Score")
plt.show()
```



In [20]:
```
# Build model with 2 important features
X = iris[['petallength', 'petalwidth']]
y = iris['iris']
```

In [21]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.3)
```

In [22]:
```
model_new = RandomForestClassifier(n_estimators=100)
# Train model
model_new.fit(X_train, y_train)
```

Out[22]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=None, max_features='auto', max_leaf_nodes=Non
e,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

In [23]:
```
y_pred_new = model_new.predict(X_test)
```

In [24]:
```
print("Accuracy:", metrics.accuracy_score(y_test, y_pred_new))
```

```
Accuracy: 0.9555555555555556
```

In [25]:
```
# Make new prediction
import numpy as np
X_new = np.array([[1.3, 0.2],
                  [4.4, 1.4],
                  [5.1, 1.8]])
```

In [26]:
```
yhat_new = model_new.predict(X_new)
yhat_new
```

Out[26]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [27]:
```
# Avg max depth
max_depth = list()
for tree in model_new.estimators_:
    max_depth.append(tree.tree_.max_depth)

print("avg max depth %0.1f" % (sum(max_depth) / len(max_depth)))
```

```
avg max depth 4.4
```