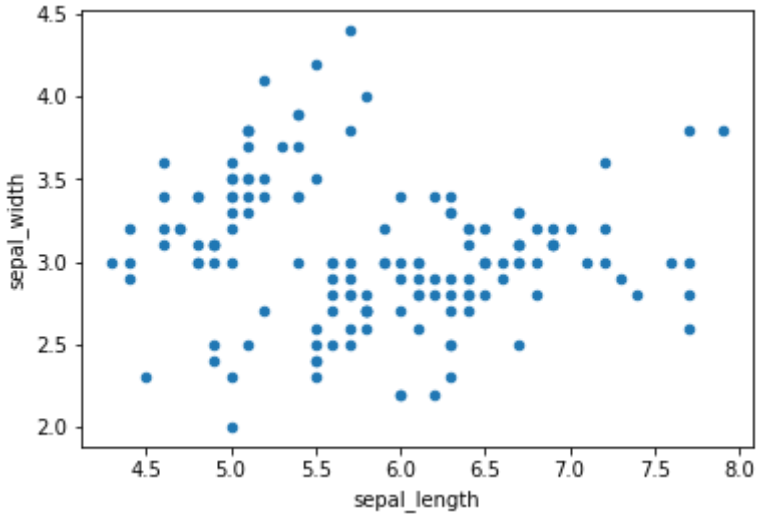
In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
iris =   pd.read_csv("IRIS.csv")
print(iris.head()) #prints first 5 values
```

```
   sepal_length  sepal_width  petal_length  petal_width      species
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa
```

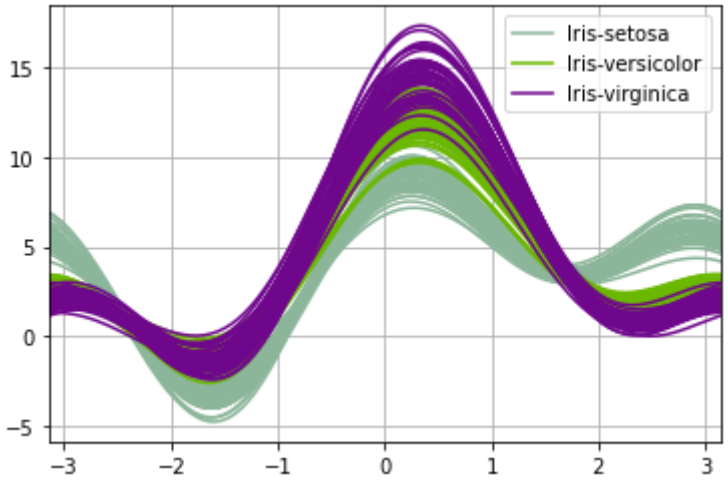
In [2]:
```
print(iris.describe()) #prints some basic statistical data

```

```
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

In [3]:
```
iris.plot(kind="scatter", x="sepal_length",y="sepal_width")
plt.show()
```



In [4]:
```
from pandas.plotting import andrews_curves
andrews_curves(iris.drop(index=1, axis=1), "species")
plt.show()
```



In [5]:
```
from sklearn.model_selection import train_test_split
x = iris.iloc[:, :-1].values #last column values excluded
y = iris.iloc[:,   -1].values #last column value
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

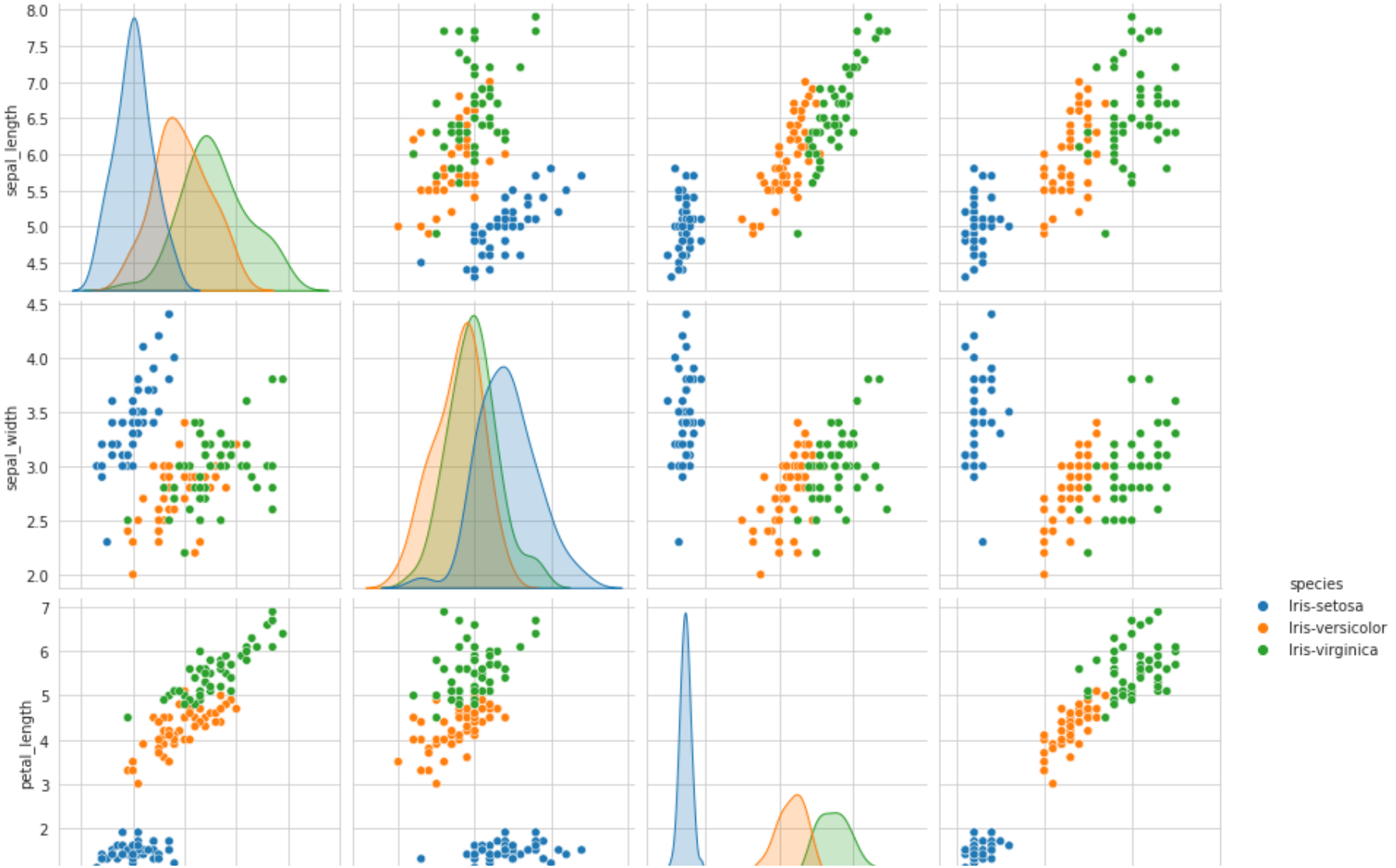
In [6]:
```
from sklearn.tree import   DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

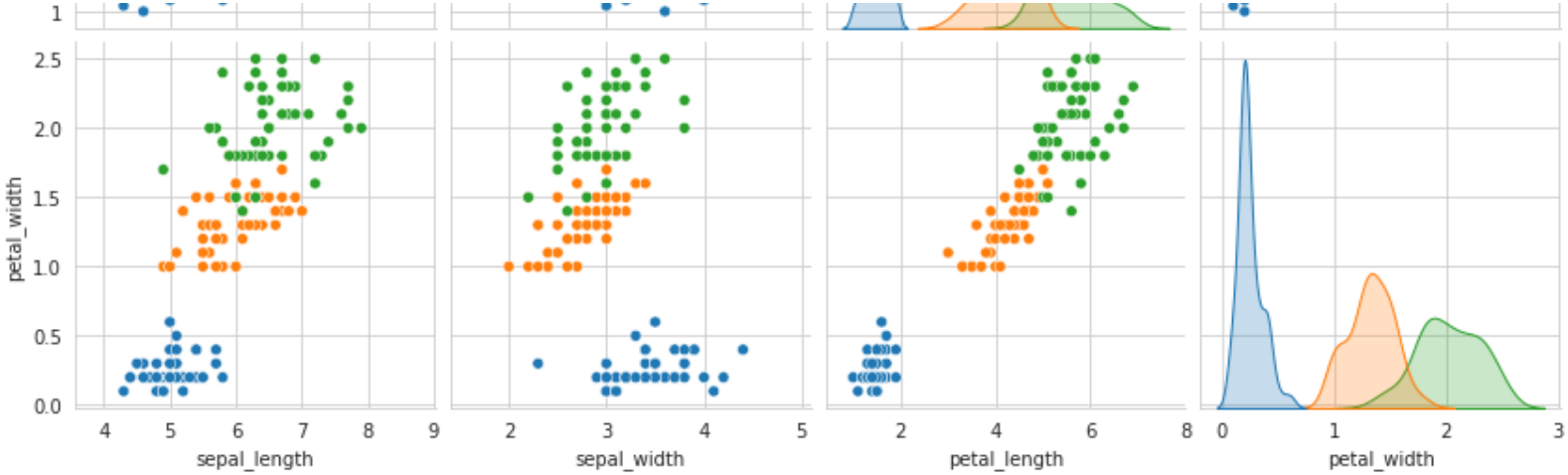
In [7]:
```
classifier   = DecisionTreeClassifier()
classifier.fit(x_train,   y_train) #training the classifier
y_pred   = classifier.predict(x_test) #making precdictions
print('accuracy   is',accuracy_score(y_pred,y_test)) #Accuracy   score
```

```
accuracy   is 1.0
```

In [17]:
```
import seaborn as sns
sns.set_style("whitegrid")
sns.pairplot(iris,hue="species",size=3);
plt.show()
```

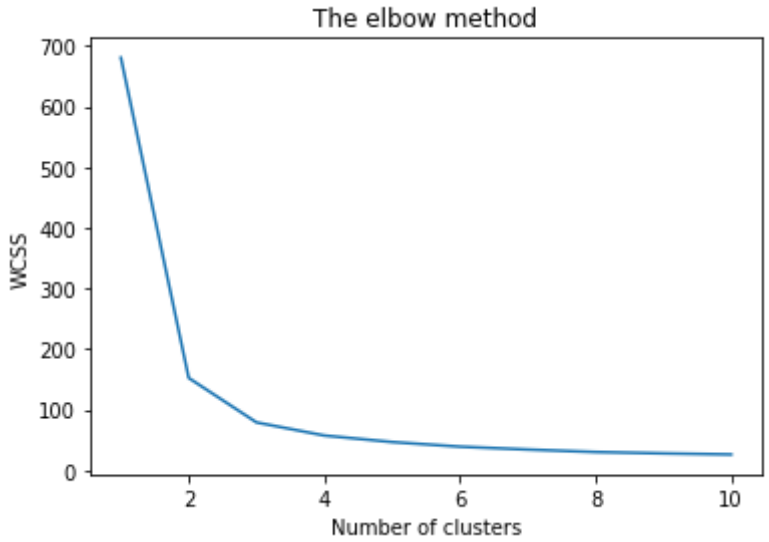
```
/home/praveen/anaconda3/lib/python3.8/site-packages/seaborn/axisgrid.py:1912: UserWarning: The `size` param
eter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```





In [8]:
```
from sklearn.cluster import KMeans
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

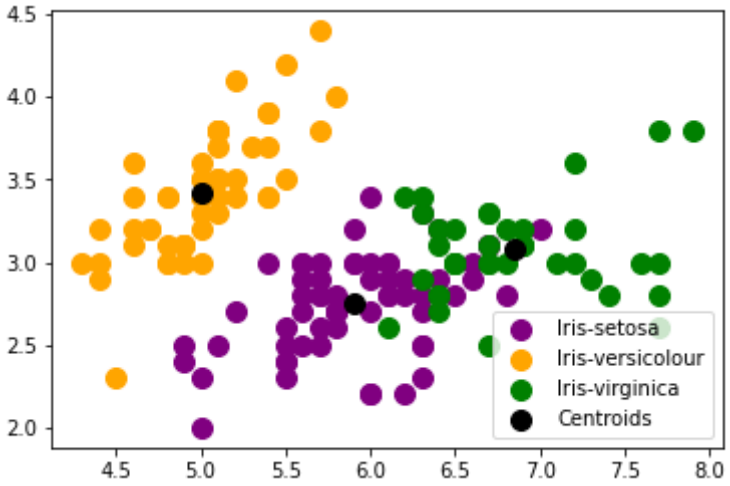
In [9]:
```
plt.plot(range(1, 11), wcss)
plt.title('The elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS') #within cluster sum of squares
plt.show()
```



In [10]:
```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(x)
```

In [12]:
```
#Visualising the clusters
plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'purple', label = 'Iris-setosa')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'orange', label = 'Iris-versicolour')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Iris-virginica')

#Plotting the centroids of the clusters
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,1], s = 100, c = 'black', label = 'Cent

plt.legend()
```

Out[12]: <matplotlib.legend.Legend at 0x7f7e434337c0>



In [ ]: