In [ ]:
```
#importing the basic libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [38]:
```
#importing the dataset using read_csv
iris = pd.read_csv(r'C:\Users\vinee\OneDrive\Desktop\FProjects\IRIS.csv')
```

In [39]:
```
#quick view of top 5 rows
iris.head()
```

Out[39]:

	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 [40]:
```
#checking the data types of the columns
iris.dtypes
```

Out[40]:
```
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
species          object
dtype: object
```

In [41]:
```
#How many unique species are there in the flower
iris['species'].unique()
```

Out[41]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [42]:
```
iris.groupby(['species']).count()
```

Out[42]:

	sepal_length	sepal_width	petal_length	petal_width
species				
Iris-setosa	50	50	50	50
Iris-versicolor	50	50	50	50
Iris-virginica	50	50	50	50

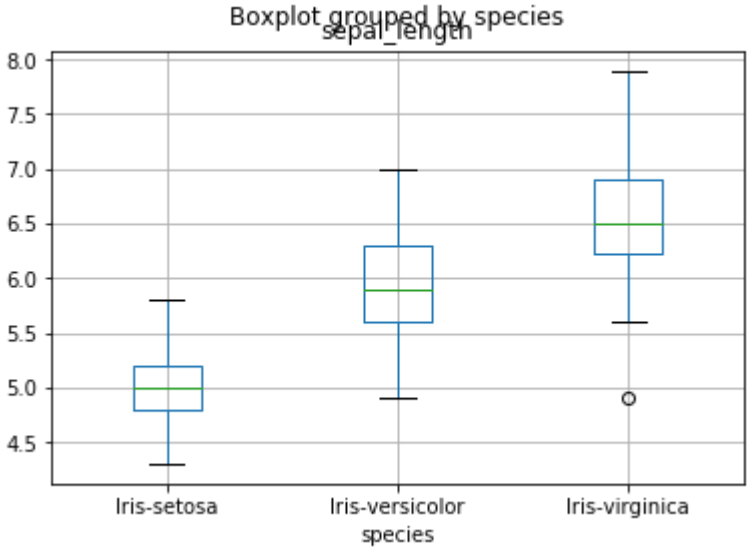
In [43]:
```
iris.describe()
```

Out[43]:

	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

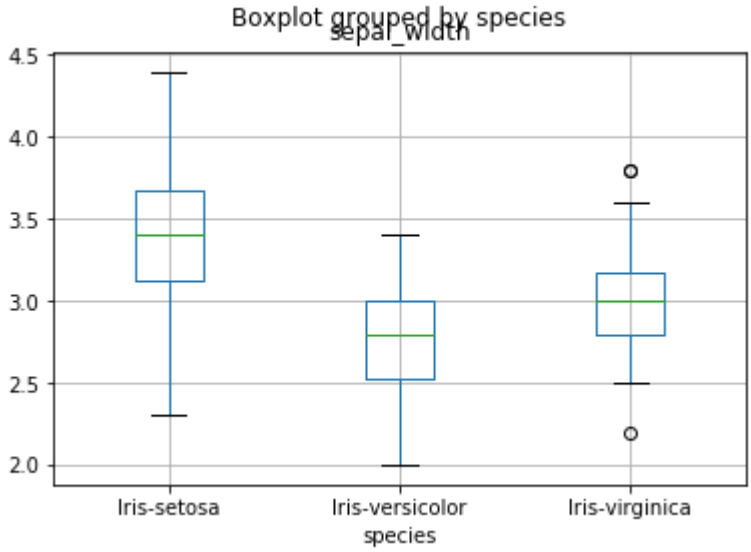
In [44]:
```
#plotting boxplots for each species to see the outliers
iris.boxplot(column='sepal_length', by='species')
#There are no outliers in setosa and versicolor, but there is one outlier in v
irginica when we plot against sepal_length
```

Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x1ba9649a400>



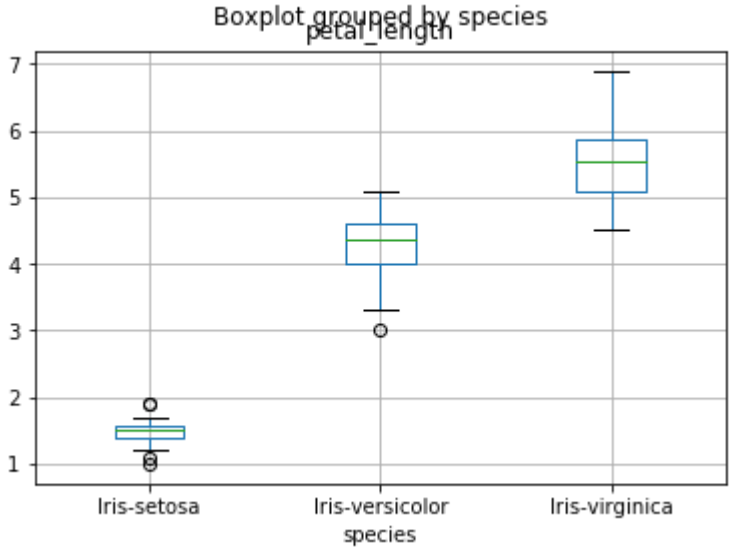
In [45]:
```
iris.boxplot(column='sepal_width', by='species')
#there are two outliers in virginica when we plot against sepal_width
```

Out[45]: <matplotlib.axes._subplots.AxesSubplot at 0x1ba9650ce10>



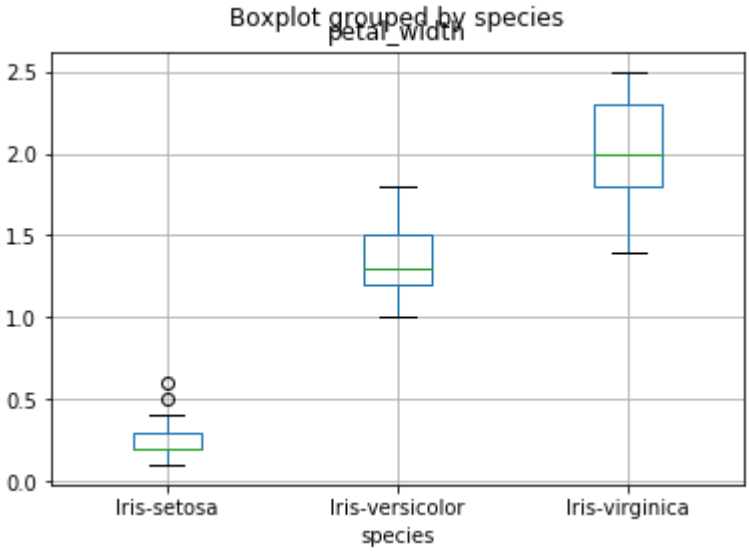
In [46]:
```
iris.boxplot(column='petal_length', by='species')
#There are two outliers in setosa and one in versicolor when we plot against p
etal_length
```

Out[46]: <matplotlib.axes._subplots.AxesSubplot at 0x1ba95b2fc50>



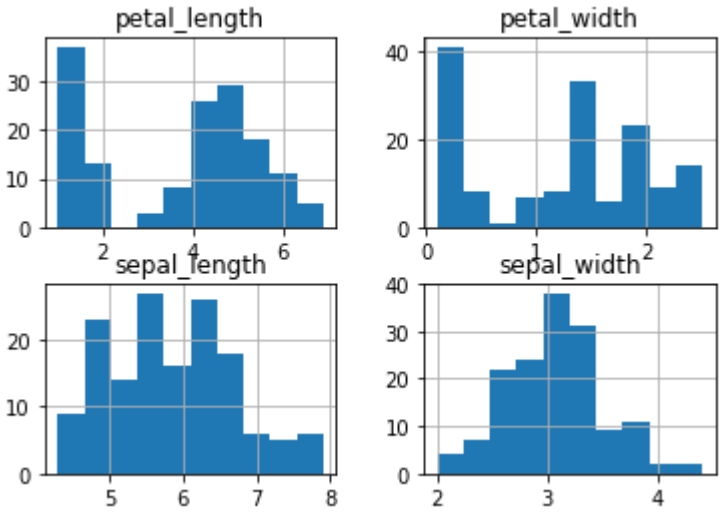
In [47]:
```
iris.boxplot(column='petal_width', by='species')
#when the plot is against the petalk_width the outliers are just observed in s
etosa
```

Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1ba965de828>



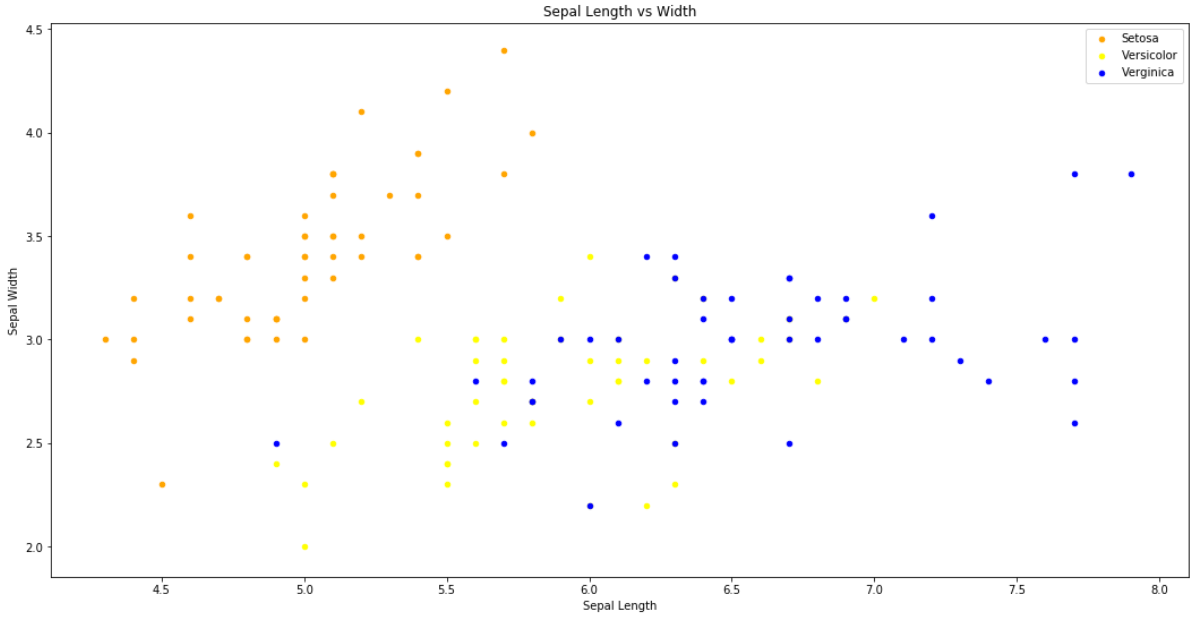
In [48]:
```
#plotting histogram to see the frequency of the 'petal_length','petal_widt
h','sepal_length' and 'sepal_width'.
iris.hist()
```

Out[48]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001BA965B42B0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001BA966EDC50
>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001BA96722BE0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001BA9675DCC0
>]],
      dtype=object)
```



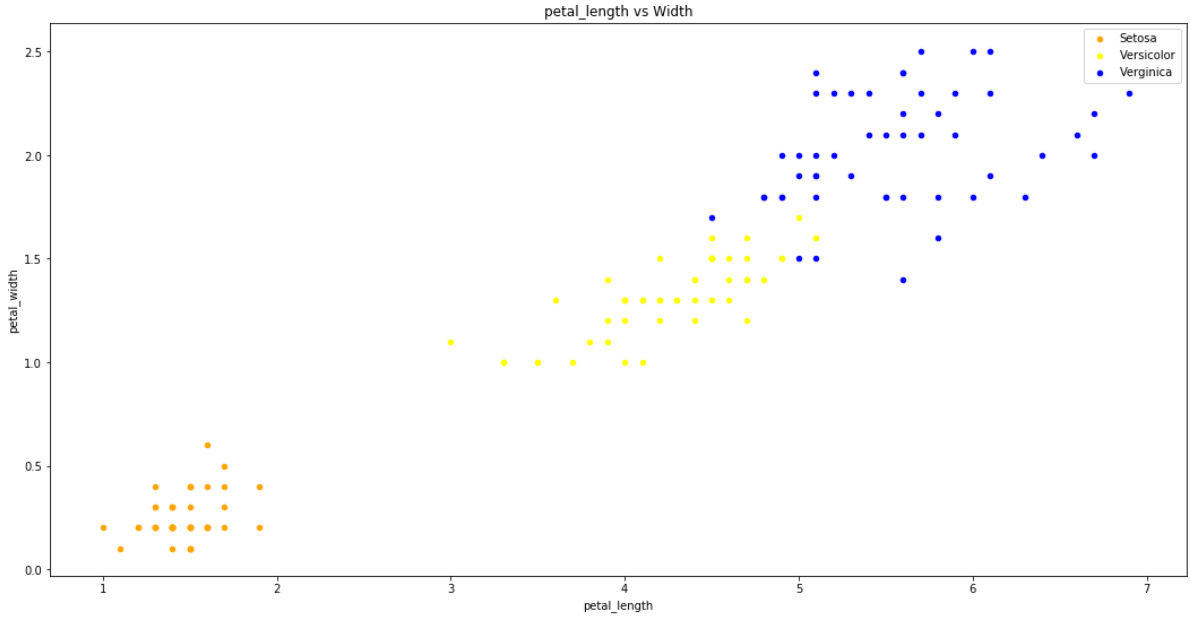
In [49]:
```
#scatter plot to see the correlation between length and width
fig = iris[iris['species'] == 'Iris-setosa'].plot(kind='Scatter', x='sepal_len
gth', y='sepal_width', color='orange', label='Setosa')
iris[iris['species'] == 'Iris-versicolor'].plot(kind='Scatter', x='sepal_lengt
h', y='sepal_width', color='yellow', label='Versicolor', ax=fig)
iris[iris['species'] == 'Iris-virginica'].plot(kind='Scatter', x='sepal_lengt
h', y='sepal_width', color='blue', label='Verginica', ax=fig)
fig.set_ylabel('Sepal Width')
fig.set_xlabel('Sepal Length')
fig.set_title('Sepal Length vs Width')

fig = plt.gcf()
fig.set_size_inches(18, 9)
plt.show()
```



In [50]:
```
fig = iris[iris['species'] == 'Iris-setosa'].plot(kind='Scatter', x='petal_len
gth', y='petal_width', color='orange', label='Setosa')
iris[iris['species'] == 'Iris-versicolor'].plot(kind='Scatter', x='petal_lengt
h', y='petal_width', color='yellow', label='Versicolor', ax=fig)
iris[iris['species'] == 'Iris-virginica'].plot(kind='Scatter', x='petal_lengt
h', y='petal_width', color='blue', label='Verginica', ax=fig)
fig.set_ylabel('petal_width')
fig.set_xlabel('petal_length')
fig.set_title('petal_length vs Width')

fig = plt.gcf()
fig.set_size_inches(18, 9)
plt.show()
```



In [51]:
```
iris['petal_area'] = iris.apply(lambda row: (row['petal_length'] * row['petal_
width']), axis=1)
```

In [52]:
```
iris.head()
```

Out[52]:

	sepal_length	sepal_width	petal_length	petal_width	species	petal_area
0	5.1	3.5	1.4	0.2	Iris-setosa	0.28
1	4.9	3.0	1.4	0.2	Iris-setosa	0.28
2	4.7	3.2	1.3	0.2	Iris-setosa	0.26
3	4.6	3.1	1.5	0.2	Iris-setosa	0.30
4	5.0	3.6	1.4	0.2	Iris-setosa	0.28

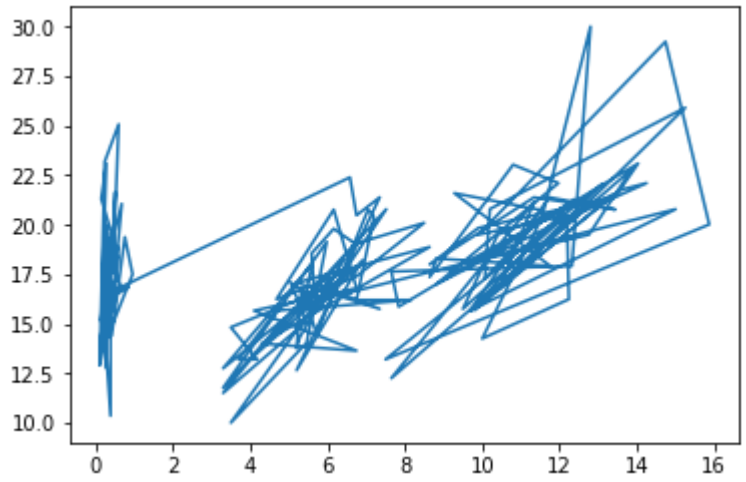
In [53]:
```
iris['sepal_area'] = iris.apply(lambda row: (row['sepal_length'] * row['sepal_
width']), axis=1)
```

In [54]:
```
iris.head()
```

Out[54]:

	sepal_length	sepal_width	petal_length	petal_width	species	petal_area	sepal_area
0	5.1	3.5	1.4	0.2	Iris-setosa	0.28	17.85
1	4.9	3.0	1.4	0.2	Iris-setosa	0.28	14.70
2	4.7	3.2	1.3	0.2	Iris-setosa	0.26	15.04
3	4.6	3.1	1.5	0.2	Iris-setosa	0.30	14.26
4	5.0	3.6	1.4	0.2	Iris-setosa	0.28	18.00

In [55]:
```
plt.plot(iris['petal_area'], iris['sepal_area'])
plt.show()
#clearly there is no correlation between two areas
```



In [56]:
```
iris['species'].groupby(pd.qcut(iris['petal_area'], 3)).value_counts()
```

Out[56]:
```
petal_area      species
(0.109, 2.52]   Iris-setosa        50
(2.52, 7.713]   Iris-versicolor    47
                Iris-virginica      3
(7.713, 15.87]  Iris-virginica     47
                Iris-versicolor     3
Name: species, dtype: int64
```

In [57]:
```
iris['species'].groupby(pd.qcut(iris['sepal_area'], 3)).value_counts()
#the sepal area is more varied than the petal_area, so it is better to take th
e differnece in areas for future purpose.
```

Out[57]:
```
sepal_area      species
(9.999, 16.2]   Iris-versicolor    24
                Iris-setosa        19
                Iris-virginica      8
(16.2, 19.38]   Iris-setosa        22
                Iris-versicolor    17
                Iris-virginica     12
(19.38, 30.02]  Iris-virginica     30
                Iris-setosa         9
                Iris-versicolor     9
Name: species, dtype: int64
```

In [58]:
```
iris['area_diff'] = iris.apply(lambda row: (row['sepal_area'] - row['petal_are
a']), axis=1)
```

In [59]:
```
iris.head()
#including all these variables can give a better accuracy by applying Random f
orest algorithm
```

Out[59]:

	sepal_length	sepal_width	petal_length	petal_width	species	petal_area	sepal_area
0	5.1	3.5	1.4	0.2	Iris-setosa	0.28	17.85
1	4.9	3.0	1.4	0.2	Iris-setosa	0.28	14.70
2	4.7	3.2	1.3	0.2	Iris-setosa	0.26	15.04
3	4.6	3.1	1.5	0.2	Iris-setosa	0.30	14.26
4	5.0	3.6	1.4	0.2	Iris-setosa	0.28	18.00

In [65]:
```
#importing labelencoder to label the species column
from sklearn.preprocessing import LabelEncoder
```

In [66]:
```
label_encoder = LabelEncoder()
```

In [67]:
```
iris['species'] = label_encoder.fit_transform(iris['species'])
```

In [68]:
```
iris.head()
#clearly we can see below the labelencoder has done a good job for the species column
```

Out[68]:

	sepal_length	sepal_width	petal_length	petal_width	species	petal_area	sepal_area
0	5.1	3.5	1.4	0.2	0	0.28	17.85
1	4.9	3.0	1.4	0.2	0	0.28	14.70
2	4.7	3.2	1.3	0.2	0	0.26	15.04
3	4.6	3.1	1.5	0.2	0	0.30	14.26
4	5.0	3.6	1.4	0.2	0	0.28	18.00

In [69]:
```
from sklearn.model_selection import train_test_split
#import train_test_split to split the data into training and testing
```

In [70]:
```
x = iris.drop('species',axis=1)
# x includes all the columns except species column
```

In [71]:
```
y = iris['species']
# y includes only species column
```

In [72]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3)
# now the data is split into x_train, y_train, x_test, y_test
```

In [73]:
```
from sklearn.ensemble import RandomForestClassifier
#import randomforest classifier from sklearn
```

In [74]:
```
rfc = RandomForestClassifier(n_estimators=200)
```

In [75]:
```
rfc.fit(x_train,y_train)
#fit the imported randomforest classifier for x_train and y_tarin
```

Out[75]:
```
RandomForestClassifier(bootstrap=True, 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, n_estimators=200, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [76]:
```
rfc_pred = rfc.predict(x_test)
#x_test is used for predicting the values and check with the y_test
```

In [78]:
```
from sklearn.metrics import confusion_matrix
```

In [79]:
```
print(confusion_matrix(y_test,rfc_pred))
```

```
[[13  0  0]
 [ 0 16  0]
 [ 0  0 16]]
```

In [81]:
```
from sklearn.metrics import classification_report
```

In [82]:
```
print(classification_report(y_test,rfc_pred))
#below classification report shows the good scores of precission and recall
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        16
           2       1.00      1.00      1.00        16

   micro avg       1.00      1.00      1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

In [85]:
```
from sklearn import metrics
pscore = metrics.accuracy_score(y_test, rfc_pred)
pscore
#accuracy score of 1.0 indicates that the algorithm we applied is perfect in p
redicting the flowers species on the given data
```

Out[85]: 1.0