In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
iris=sns.load_dataset('iris')
```

In [3]:
```
iris.head()
```

Out[3]:

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

In [4]:
```
iris.isna().sum()
```

Out[4]:
```
sepal_length  0
sepal_width   0
petal_length  0
petal_width   0
species       0
dtype: int64
```

In [5]:
```
iris.describe(include='all')
```

Out[5]:

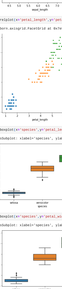
| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| **count** | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150 |
| **unique** | NaN | NaN | NaN | NaN | 3 |
| **top** | NaN | NaN | NaN | NaN | setosa |
| **freq** | NaN | NaN | NaN | NaN | 50 |
| **mean** | 5.843333 | 3.057333 | 3.758000 | 1.199333 | NaN |
| **std** | 0.828066 | 0.435866 | 1.765298 | 0.762238 | NaN |
| **min** | 4.300000 | 2.000000 | 1.000000 | 0.100000 | NaN |
| **25%** | 5.100000 | 2.800000 | 1.600000 | 0.300000 | NaN |
| **50%** | 5.800000 | 3.000000 | 4.350000 | 1.300000 | NaN |
| **75%** | 6.400000 | 3.300000 | 5.100000 | 1.800000 | NaN |
| **50%** | 7.900000 | 4.400000 | 6.900000 | 2.500000 | NaN |

In [6]:
```
iris.info()
```

Out[6]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   sepal_length 150 non-null    float64
 1   sepal_width  150 non-null    float64
 2   petal_length 150 non-null    float64
 3   petal_width  150 non-null    float64
 4   species      150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [7]:
```
sns.relplot('sepal_length','petal_width',data=iris,hue='species',style='species')
```

Out[7]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0f2fac0>
```




In [8]:
```
sns.relplot('petal_length','petal_width',data=iris,hue='species',style='species')
```

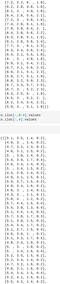
Out[8]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0e1ed40>
```

In [9]:
```
sns.boxplot('species','petal_length',data=iris)
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0e1ed40>
```

In [10]:
```
sns.boxplot('species','petal_width',data=iris)
```

Out[10]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0e1ed40>
```

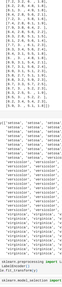



In [11]:
```
sns.boxplot('species','sepal_width',data=iris)
```

Out[11]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0e1ed40>
```

In [12]:
```
sns.boxplot('species','sepal_length',data=iris)
```

Out[12]:
```
<seaborn.axisgrid.FacetGrid at 0x7ef3b0e1ed40>
```




In [13]:
```
iris.iloc[:,0:4].values
```

Out[13]:
```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3.0, 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5.0, 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5.0, 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3.0, 1.4, 0.1],
       [4.3, 3.0, 1.1, 0.1],
       [5.8, 4.0, 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
       [5.4, 3.4, 1.7, 0.2],
       [5.1, 3.7, 1.5, 0.4],
       [4.6, 3.6, 1.0, 0.2],
       [5.1, 3.3, 1.7, 0.5],
       [4.8, 3.4, 1.9, 0.2],
       [5.0, 3.0, 1.6, 0.2],
       [5.0, 3.4, 1.6, 0.4],
       [5.2, 3.5, 1.5, 0.2],
       [5.2, 3.4, 1.4, 0.2],
       [4.7, 3.2, 1.6, 0.2],
       [4.8, 3.1, 1.6, 0.2],
       [5.4, 3.4, 1.5, 0.4],
       [5.2, 4.1, 1.5, 0.1],
       [5.5, 4.2, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.0, 3.2, 1.2, 0.2],
       [5.5, 3.5, 1.3, 0.2],
       [4.9, 3.6, 1.4, 0.1],
       [4.4, 3.0, 1.3, 0.2],
       [5.1, 3.4, 1.5, 0.2],
       [5.0, 3.5, 1.3, 0.3],
       [4.5, 2.3, 1.3, 0.3],
       [4.4, 3.2, 1.3, 0.2],
       [5.0, 3.5, 1.6, 0.6],
       [5.1, 3.8, 1.9, 0.4],
       [4.8, 3.0, 1.4, 0.3],
       [5.1, 3.8, 1.6, 0.2],
       [4.6, 3.2, 1.4, 0.2],
       [5.3, 3.7, 1.5, 0.2],
       [5.0, 3.3, 1.4, 0.2],
       [7.0, 3.2, 4.7, 1.4],
       [6.4, 3.2, 4.5, 1.5],
       [6.9, 3.1, 4.9, 1.5],
       [5.5, 2.3, 4.0, 1.3],
       [6.5, 2.8, 4.6, 1.5],
       [5.7, 2.8, 4.5, 1.3],
       [6.3, 3.3, 4.7, 1.6],
       [4.9, 2.4, 3.3, 1.0],
       [6.6, 2.9, 4.6, 1.3],
       [5.2, 2.7, 3.9, 1.4],
       [5.0, 2.0, 3.5, 1.0],
       [5.9, 3.0, 4.2, 1.5],
       [6.0, 2.2, 4.0, 1.0],
       [6.1, 2.9, 4.7, 1.4],
       [5.6, 2.9, 3.6, 1.3],
       [6.7, 3.1, 4.4, 1.4],
       [5.6, 3.0, 4.5, 1.5],
       [5.8, 2.7, 4.1, 1.0],
       [6.2, 2.2, 4.5, 1.5],
       [5.6, 2.5, 3.9, 1.1],
       [5.9, 3.2, 4.8, 1.8],
       [6.1, 2.8, 4.0, 1.3],
       [6.3, 2.5, 4.9, 1.5],
       [6.1, 2.8, 4.7, 1.2],
       [6.4, 2.9, 4.3, 1.3],
       [6.6, 3.0, 4.4, 1.4],
       [6.8, 2.8, 4.8, 1.4],
       [6.7, 3.0, 5.0, 1.7],
       [6.0, 2.9, 4.5, 1.5],
       [5.7, 2.6, 3.5, 1.0],
       [5.5, 2.4, 3.8, 1.1],
       [5.5, 2.4, 3.7, 1.0],
       [5.8, 2.7, 3.9, 1.2],
       [6.0, 2.7, 5.1, 1.6],
       [5.4, 3.0, 4.5, 1.5],
       [6.0, 3.4, 4.5, 1.6],
       [6.7, 3.1, 4.7, 1.5],
       [6.3, 2.3, 4.4, 1.3],
       [5.6, 3.0, 4.1, 1.3],
       [5.5, 2.5, 4.0, 1.3],
       [5.5, 2.6, 4.4, 1.2],
       [6.1, 3.0, 4.6, 1.4],
       [5.8, 2.6, 4.0, 1.2],
       [5.0, 2.3, 3.3, 1.0],
       [5.6, 2.7, 4.2, 1.3],
       [5.7, 3.0, 4.2, 1.2],
       [5.7, 2.9, 4.2, 1.3],
       [6.2, 2.9, 4.3, 1.3],
       [5.1, 2.5, 3.0, 1.1],
       [5.7, 2.8, 4.1, 1.3],
       [6.3, 3.3, 6.0, 2.5],
       [5.8, 2.7, 5.1, 1.9],
       [7.1, 3.0, 5.9, 2.1],
       [6.3, 2.9, 5.6, 1.8],
       [6.5, 3.0, 5.8, 2.2],
       [7.6, 3.0, 6.6, 2.1],
       [4.9, 2.5, 4.5, 1.7],
       [7.3, 2.9, 6.3, 1.8],
       [6.7, 2.5, 5.8, 1.8],
       [7.2, 3.6, 6.1, 2.5],
       [6.5, 3.2, 5.1, 2.0],
       [6.4, 2.7, 5.3, 1.9],
       [6.8, 3.0, 5.5, 2.1],
       [5.7, 2.5, 5.0, 2.0],
       [5.8, 2.8, 5.1, 2.4],
       [6.4, 3.2, 5.3, 2.3],
       [6.5, 3.0, 5.5, 1.8],
       [7.7, 3.8, 6.7, 2.2],
       [7.7, 2.6, 6.9, 2.3],
       [6.0, 2.2, 5.0, 1.5],
       [6.9, 3.2, 5.7, 2.3],
       [5.6, 2.8, 4.9, 2.0],
       [7.7, 2.8, 6.7, 2.0],
       [6.3, 2.7, 4.9, 1.8],
       [6.7, 3.3, 5.7, 2.1],
       [7.2, 3.2, 6.0, 1.8],
       [6.2, 2.8, 4.8, 1.8],
       [6.1, 3.0, 4.9, 1.8],
       [6.4, 2.8, 5.6, 2.1],
       [7.2, 3.0, 5.8, 1.6],
       [7.4, 2.8, 6.1, 1.9],
       [7.9, 3.8, 6.4, 2.0],
       [6.4, 2.8, 5.6, 2.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.1, 2.6, 5.6, 1.4],
       [7.7, 3.0, 6.1, 2.3],
       [6.3, 3.4, 5.6, 2.4],
       [6.4, 3.1, 5.5, 1.8],
       [6.0, 3.0, 4.8, 1.8],
       [6.9, 3.1, 5.4, 2.1],
       [6.7, 3.1, 5.6, 2.4],
       [6.9, 3.1, 5.1, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [6.8, 3.2, 5.9, 2.3],
       [6.7, 3.3, 5.7, 2.5],
       [6.7, 3.0, 5.2, 2.3],
       [6.3, 2.5, 5.0, 1.9],
       [6.5, 3.0, 5.2, 2.0],
       [6.2, 3.4, 5.4, 2.3],
       [5.9, 3.0, 5.1, 1.8]])
```

In [14]:
```
iris.iloc[:,-1].values
```

Out[14]:
```
array(['setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'setosa', 'setosa', 'setosa', 'setosa', 'setosa', 'setosa',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'versicolor', 'versicolor',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica'], dtype=object)
```

In [15]:
```
from sklearn.preprocessing import LabelEncoder
y = le.fit_transform(y)
```

In [16]:
```
from sklearn.model_selection import train_test_split
```

In [17]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=0)
```

In [18]:
```
from sklearn.naive_bayes import GaussianNB
```

In [19]:
```
gaussian = GaussianNB()
```

In [20]:
```
gaussian.fit(X_train, y_train)
```

Out[20]:
```
GaussianNB
GaussianNB()
```

In [21]:
```
y_pred = gaussian.predict(X_test)
```

In [22]:
```
y_pred
```

Out[22]:
```
array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0,
       2, 0, 2, 2, 2, 2, 1, 1, 2, 0, 0, 2, 1, 2, 2, 2, 2, 1, 2, 1, 1,
       2])
```

In [23]:
```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score ,precision_score,recall_score,f1_score
```

In [24]:
```
cm = confusion_matrix(y_test, y_pred)
```

In [25]:
```
print("Confusion Matrix\n",cm)
```

Out[25]:
```
Confusion Matrix
 [[ 0 16  0]
 [ 0 14  0]
 [ 0  5 14]]
```

In [26]:
```
accuracy = accuracy_score(y_test,y_pred)
precision =precision_score(y_test, y_pred,average='micro')
print('accuracy = %.3f',accuracy)
print('precision= %.3f',precision)
```

Out[26]:
```
accuracy = 0.667
precision= 0.667
```

```
accuracy : 1.000
precision 1.000
```