SUPPORT VECTOR MACHINE

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn import svm
```

In [3]:

```
data = pd.read_csv('C:\\Users\\MARVEL SAMUEL JACOB\\Desktop\\marvel.csv')
```

In [4]:

```
data
```

Out[4]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [5]:

```
data.head()
```

Out[5]:

	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 [6]:

```
data.isnull().sum()
```

Out[6]:

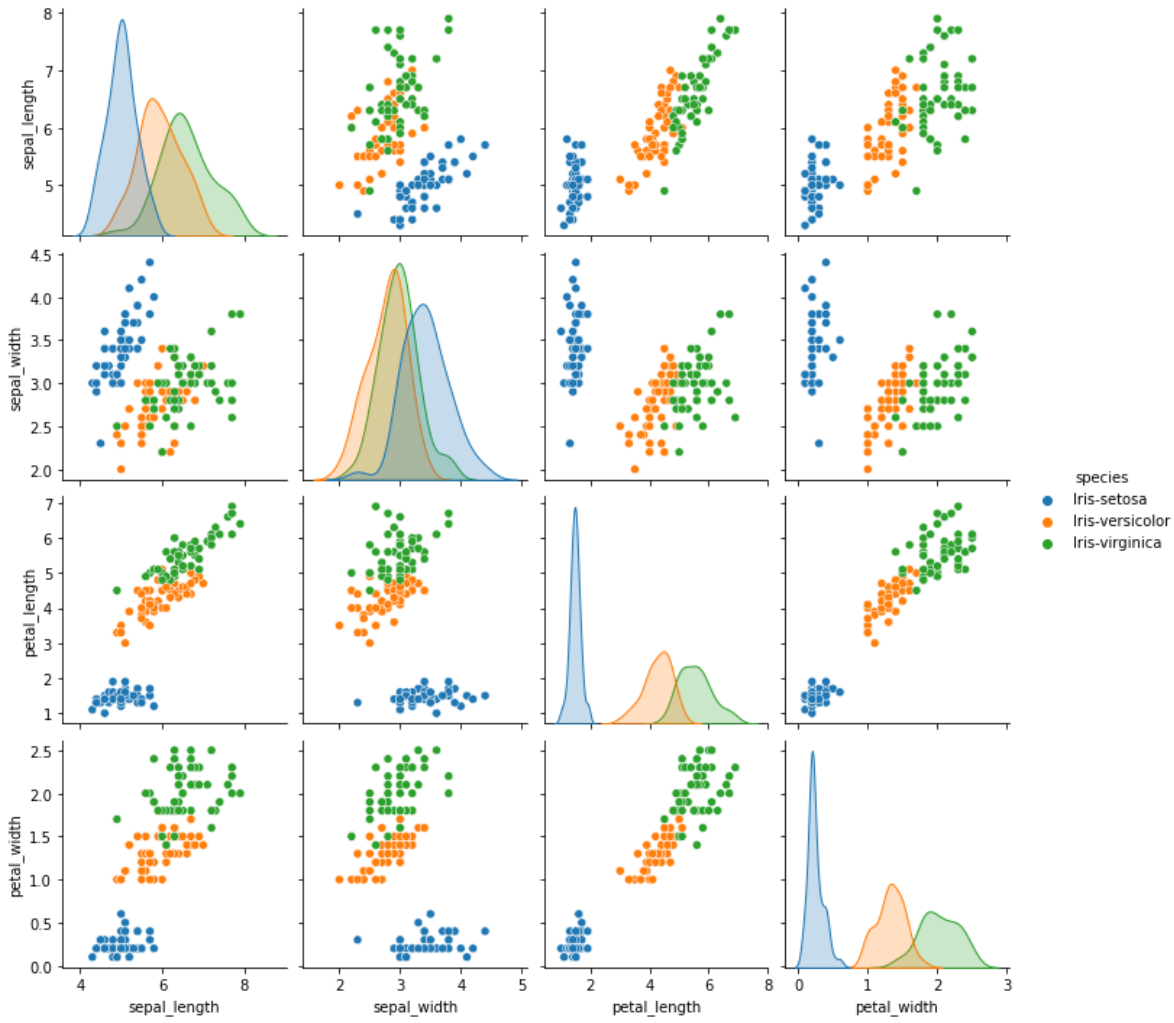
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

In [8]:

```
x=data.drop('species', axis = 1)
y = data['species']
```

In [9]:

```
import seaborn as sns
sns.pairplot(data, hue = 'species');
```



In [69]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=30, random_state= 89)
```

In [70]:

```
from sklearn.svm import SVC
model = svm.SVC()
model.fit(x_train,y_train)
```

Out[70]:

```
SVC()
```

In [71]:

```
pred = model.predict(x_test)
```

In [72]:

```
from sklearn.metrics import classification_report ,confusion_matrix
print("classification_report:\n",classification_report(y_test,pred))
print("confusion_matrix:\n",confusion_matrix(y_test,pred))
```

```
classification_report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       0.78      1.00      0.88         7
           2       1.00      0.78      0.88         9

    accuracy                           0.93        30
   macro avg       0.93      0.93      0.92        30
weighted avg       0.95      0.93      0.93        30

confusion_matrix:
 [[14  0  0]
 [ 0  7  0]
 [ 0  2  7]]
```

In [76]:

```
model.score(x_test,y_test)*100
```

Out[76]:

```
93.33333333333333
```

In [ ]:

DECISION TREE

In [12]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
```

In [14]:

```
data = pd.read_csv('C:\\Users\\MARVEL SAMUEL JACOB\\Desktop\\marvel.csv')
```

In [15]:

```
data
```

Out[15]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [16]:

```
X = iris.data
y = iris.target
```

In [17]:

```
class_names = iris.target_names
```

In [18]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

In [19]:

```
DTM = DecisionTreeClassifier()
```

In [20]:

```
y_final = DTM.fit(X_train, y_train).predict(X_test)
```

In [21]:

```
print()
print(classification_report(y_test, y_final, target_names=class_names))
```

```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       0.94      1.00      0.97        16
   virginica       1.00      0.95      0.97        19

    accuracy                           0.98        45
   macro avg       0.98      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```

In [37]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=90)
```

In [38]:

```
DTM.fit(X_train,y_train)
```

Out[38]:

```
DecisionTreeClassifier()
```

In [39]:

```
y_predict=DTM.predict(X_test)
```

In [40]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

In [41]:

```
y_final = DTM.fit(X_train, y_train).predict(X_test)
```

In [42]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

In [43]:

```
y_final = DTM.fit(X_train, y_train).predict(X_test)
```

In [44]:

```
print("Accuracy of the model by DT is :-",accuracy_score(y_test,y_predict)*100)
```

```
Accuracy of the model by DT is :- 35.55555555555556
```

K NEAREST NEIGHBOURS

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn import svm
import missingno as msn
```

In [2]:

```
data = pd.read_csv('C:\\Users\\MARVEL SAMUEL JACOB\\Desktop\\marvel.csv')
```

In [3]:

```
data
```

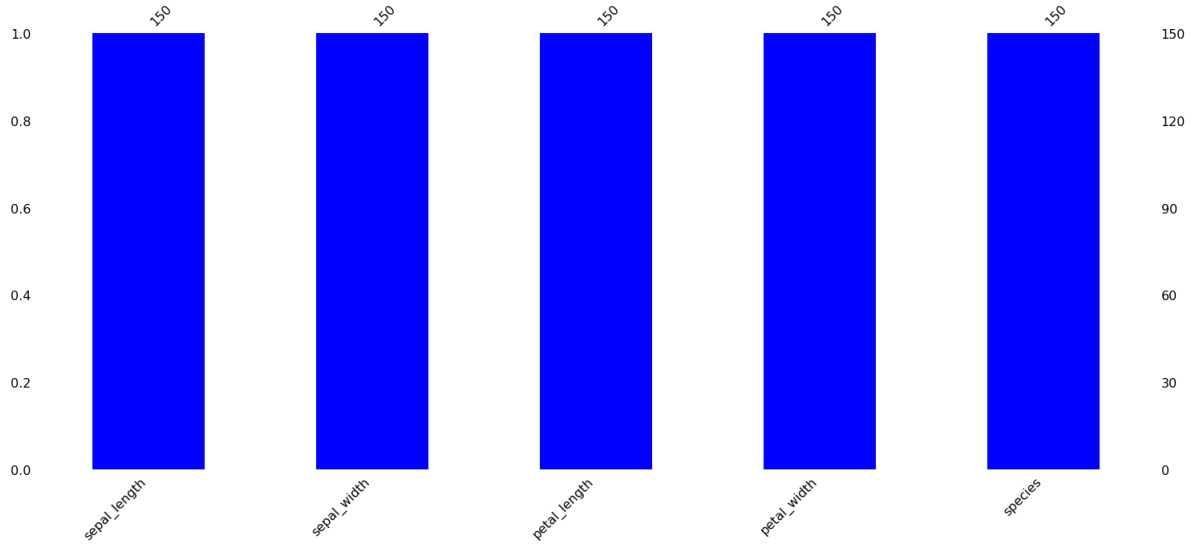
Out[3]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [4]:

```
msn.bar(data,color='blue');
```



In [5]:

```
data['species'].unique()
```

Out[5]:

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

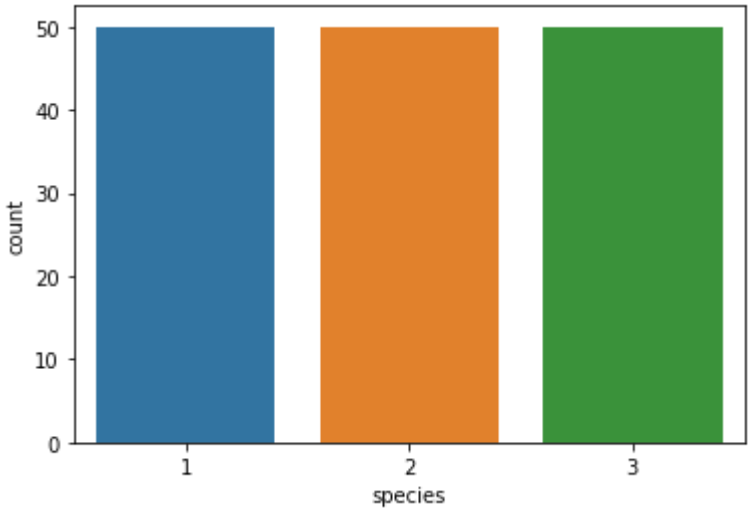
In [6]:

```
data['species']=data['species'].map({'Iris-setosa':1, 'Iris-versicolor':2, 'Iris-virginica'
```

In [7]:

```
sns.countplot(data['species']);
```

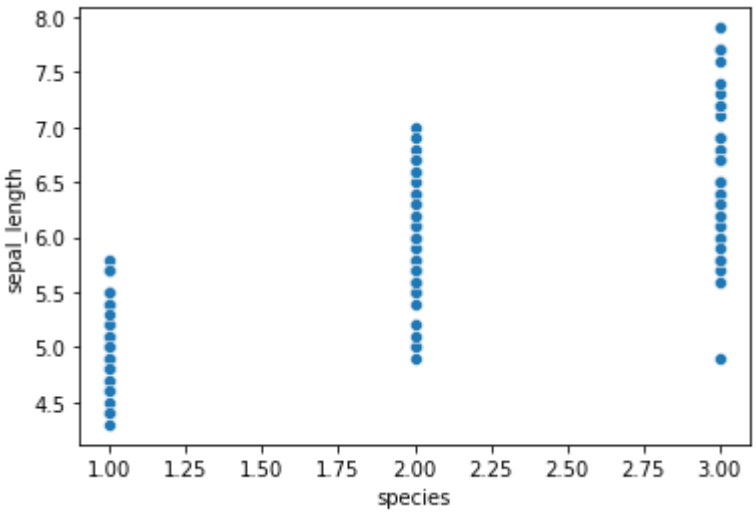
```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variable as a keyword arg: x. Fro
m version 0.12, the only valid positional argument will be `data`, and passi
ng other arguments without an explicit keyword will result in an error or mi
sinterpretation.
  warnings.warn(
```



In [8]:

```
sns.scatterplot(data['species'],data['sepal_length']);
```

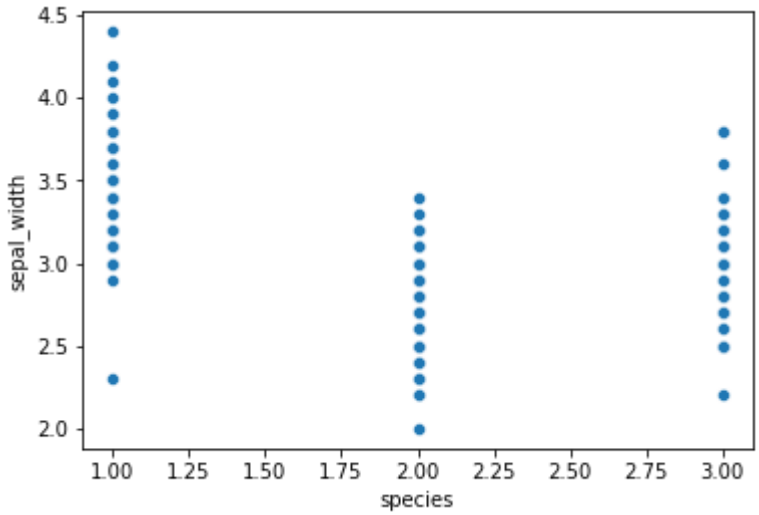
```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



In [9]:

```
sns.scatterplot(data['species'],data['sepal_width']);
```

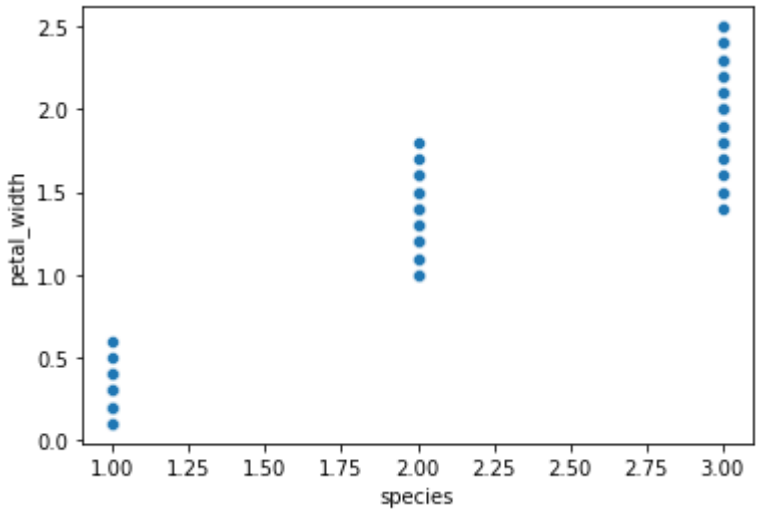
```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



In [10]:

```
sns.scatterplot(data['species'],data['petal_width']);
```

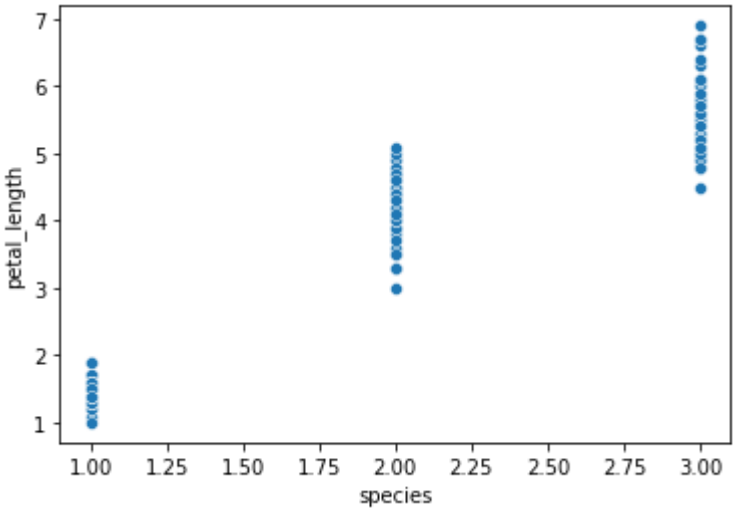
```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



In [11]:

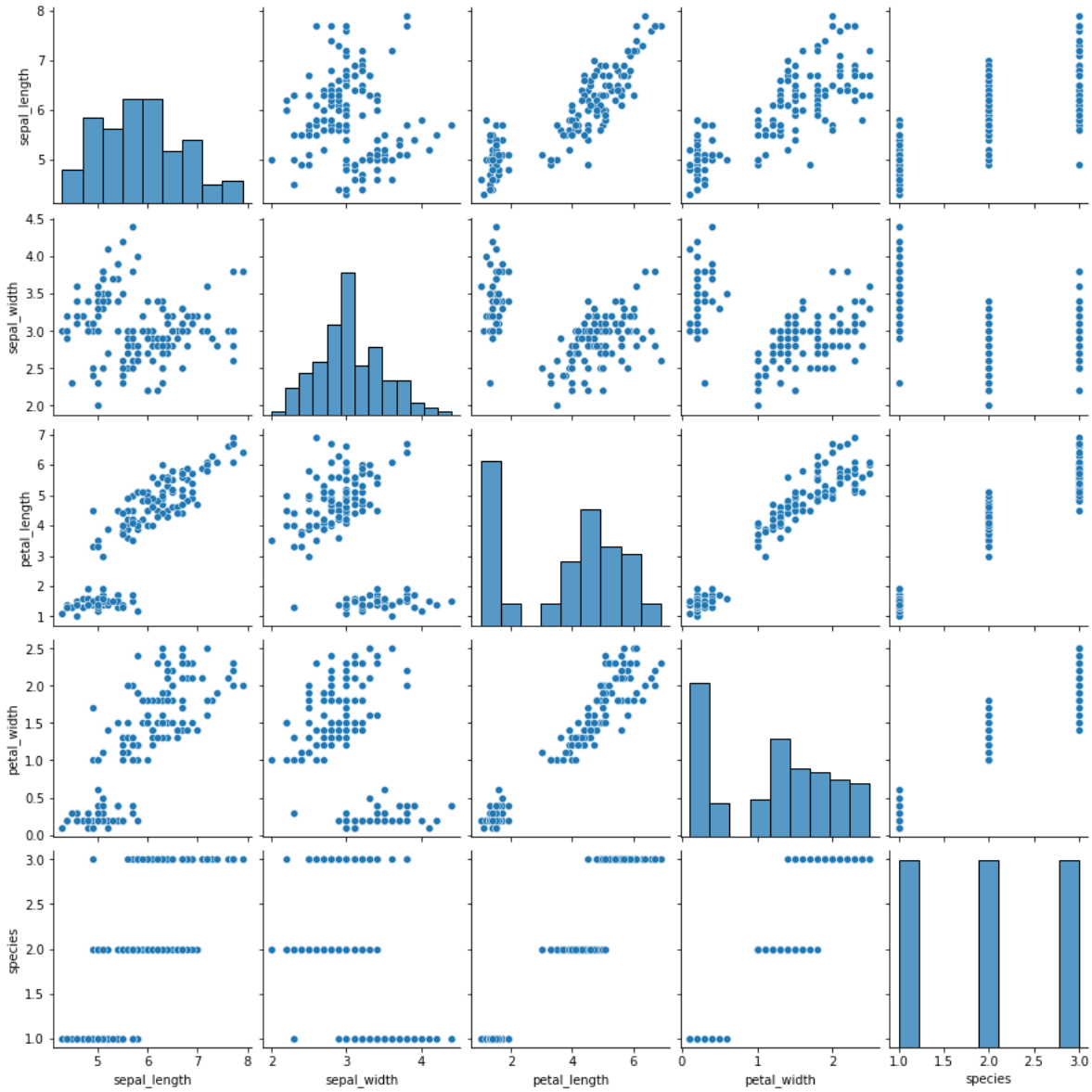
```
sns.scatterplot(data['species'],data['petal_length']);
```

```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`, and pa
ssing other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



In [12]:

```
sns.pairplot(data);
```

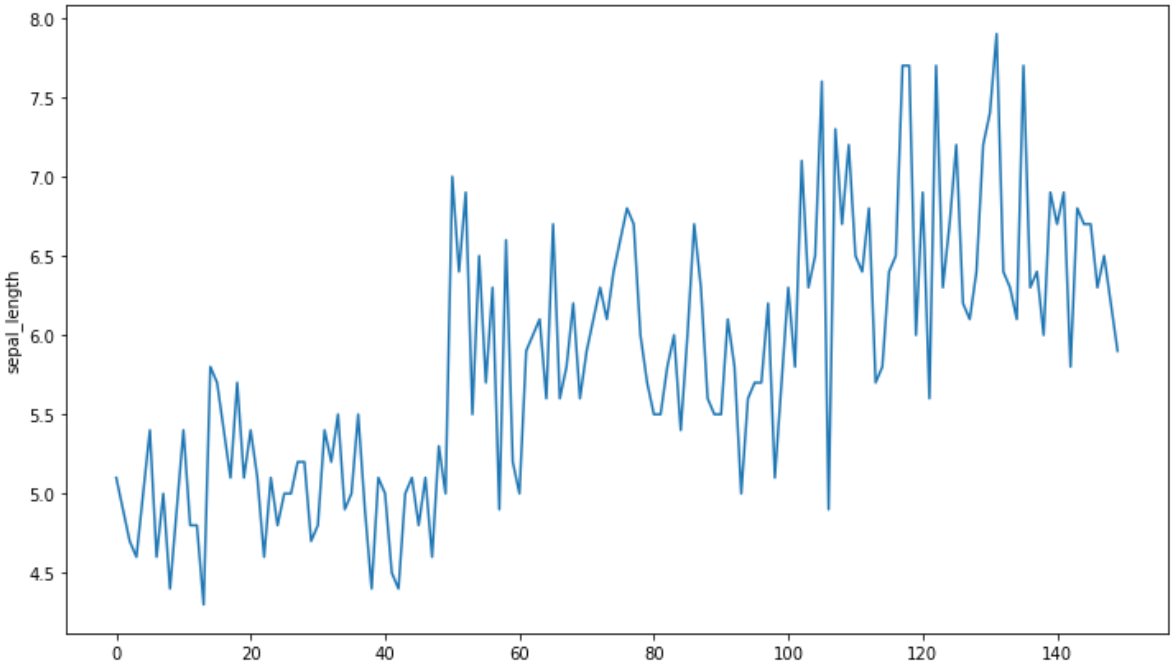


In [13]:

```
import matplotlib.pyplot as plt
```

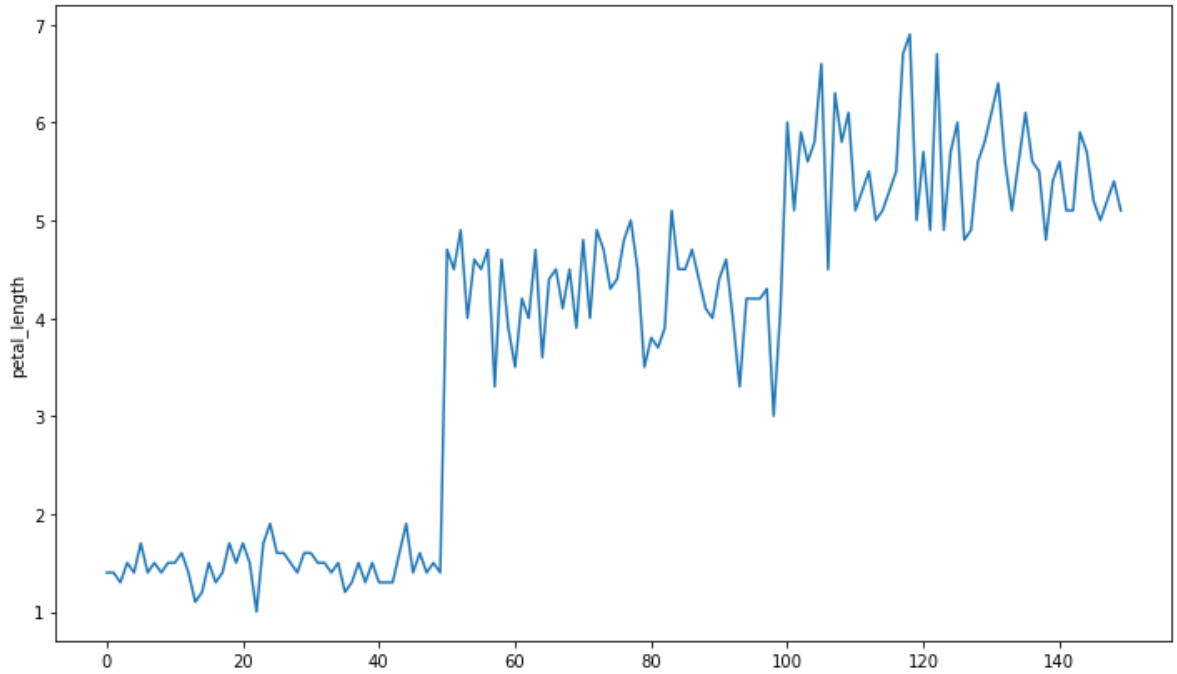
In [14]:

```
fig,ax=plt.subplots(figsize=(12,7));
sns.lineplot(data=data['sepal_length']);
```



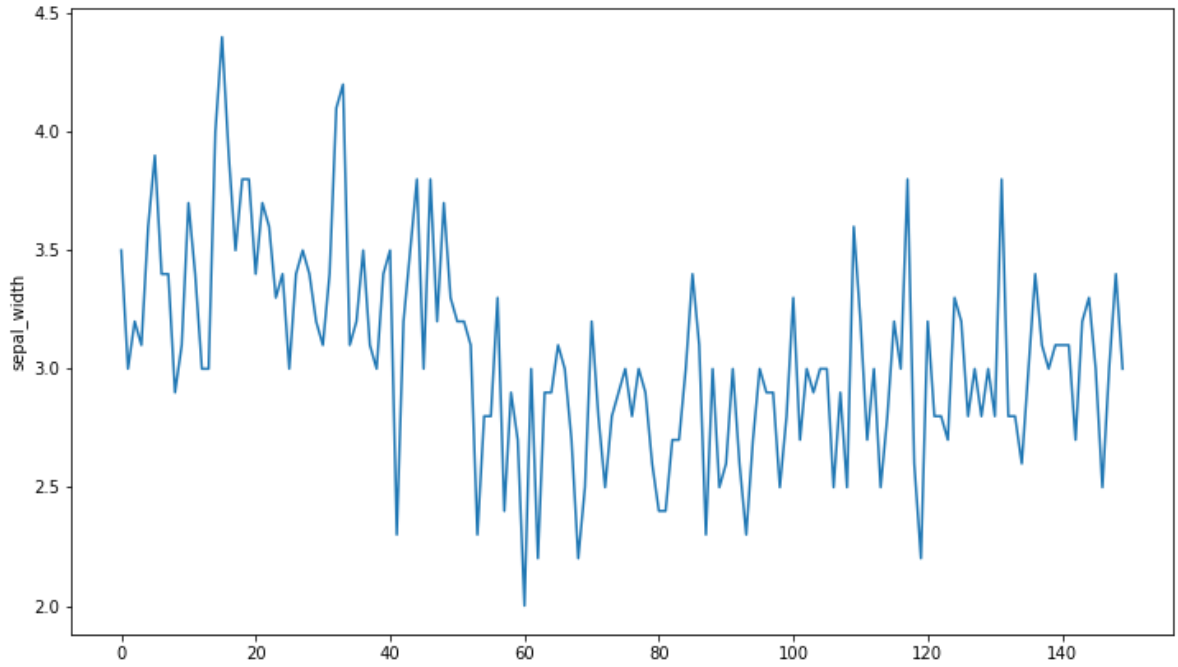
In [15]:

```
fig,ax=plt.subplots(figsize=(12,7));
sns.lineplot(data=data['petal_length']);
```



In [16]:

```
fig,ax=plt.subplots(figsize=(12,7));
sns.lineplot(data=data['sepal_width'],);
```



In [17]:

```
data.columns
```

Out[17]:

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [18]:

```
X=data[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']]
```

In [19]:

```
y=data['species']
```

In [61]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=78)
```

In [62]:

```
from sklearn.neighbors import KNeighborsClassifier
model=KNeighborsClassifier()
```

In [63]:

```
model.fit(X_train,y_train)
```

Out[63]:

```
KNeighborsClassifier()
```

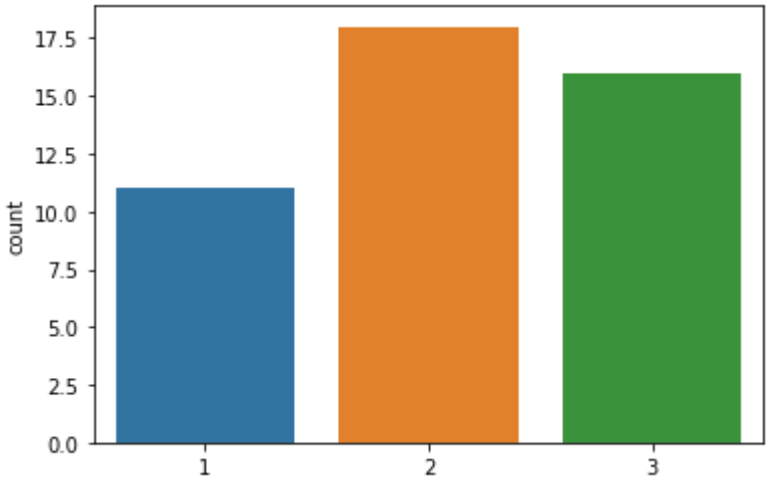
In [64]:

```
y_predict=model.predict(X_test)
```

In [65]:

```
sns.countplot(y_predict);
```

```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\seaborn\_decorator
s.py:36: FutureWarning: Pass the following variable as a keyword arg: x. Fro
m version 0.12, the only valid positional argument will be `data`, and passi
ng other arguments without an explicit keyword will result in an error or mi
sinterpretation.
  warnings.warn(
```



In [66]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

In [68]:

```
class_names = data.species
```

In [69]:

```
KNN = KNeighborsClassifier()
```

In [70]:

```
y_final = model.fit(X_train, y_train).predict(X_test)
```

In [71]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

In [72]:

```
y_final = KNN.fit(X_train, y_train).predict(X_test)
```

In [73]:

```
print("Accuracy of the model by KNN is :-",accuracy_score(y_test,y_predict)*100)
```

```
Accuracy of the model by KNN is :- 35.55555555555556
```

In [ ]:

```

```

LOGISTIC REGRESSION

In [8]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn import svm
```

In [9]:

```
data = pd.read_csv('C:\\Users\\MARVEL SAMUEL JACOB\\Desktop\\marvel.csv')
```

In [10]:

```
data
```

Out[10]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [15]:

```
X=data[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']]
```

In [16]:

```
y=data['species']
```

In [37]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=68)
```

In [38]:

```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
model.fit(X_train,y_train)
```

```
C:\Users\MARVEL SAMUEL JACOB\anaconda3\lib\site-packages\sklearn\linear_mode
l\_logistic.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
```

Out[38]:

```
LogisticRegression()
```

In [39]:

```
y_pre=model.predict(X_test)
```

In [40]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

In [41]:

```
print("Accuracy of the model by Logistic Regression is :-",accuracy_score(y_test,y_pre)*100
```

```
Accuracy of the model by Logistic Regression is :- 97.77777777777777
```

MODELS	ACCURACY
SUPPORT VECTOR MACHINE	93.33333333333333
KNN	42.22222222222222
DECISION TREE	35.55555555555556
LOGISTIC REGRESSION	97.77777777777777

FROM HERE, WE CAME TO AN CONCLUSION THAT THE MODEL LOGISTIC REGRESSION HAS THE HIGHEST ACCURACY AMONG SVM, KNN AND DECISION TREE WHIL USING IRIS FLOWERS DATASET