```
<bound method NDFrame.head of         CRIM    ZN  INDUS  CHAS    NOX     RM   AGE     DIS  RAD  TAX  \
0    0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296
1    0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242
2    0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242
3    0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222
4    0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222
..       ...   ...    ...   ...    ...    ...   ...     ...  ...  ...
501  0.06263   0.0  11.93     0  0.573  6.593  69.1  2.4786    1  273
502  0.04527   0.0  11.93     0  0.573  6.120  76.7  2.2875    1  273
503  0.06076   0.0  11.93     0  0.573  6.976  91.0  2.1675    1  273
504  0.10959   0.0  11.93     0  0.573  6.794  89.3  2.3889    1  273
505  0.04741   0.0  11.93     0  0.573  6.030  80.8  2.5050    1  273

     PTRATIO       B  LSTAT  MEDV
0       15.3  396.90   4.98  24.0
1       17.8  396.90   9.14  21.6
2       17.8  392.83   4.03  34.7
3       18.7  394.63   2.94  33.4
4       18.7  396.90   5.33  36.2
..       ...     ...    ...   ...
501     21.0  391.99   9.67  22.4
502     21.0  396.90   9.08  20.6
503     21.0  396.90   5.64  23.9
504     21.0  393.45   6.48  22.0
505     21.0  396.90   7.88  11.9

[506 rows x 14 columns]>
```

jupyter Iris Classification1 Last Checkpoint: 12 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [38]:
```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(random_state = 0)

model.fit(X_train, y_train)

ypred = model.predict(X_test)
ypred

model.score(X_test, y_test)
0.5779699824126867

plt.figure(figsize=(15, 10))
tree.plot_tree(model, filled=True)
```

Out[38]:
```
[Text(0.40550798384209114, 0.9772727272727273, 'X[0] <= 7.685\nsquared_error = 84.094\nsamples = 455\nvalue = 22.654'),
 Text(0.12481816757676126, 0.9318181818181818, 'X[0] <= 4.65\nsquared_error = 79.0\nsamples = 141\nvalue = 31.906'),
 Text(0.06337839047533793, 0.8863636363636364, 'X[0] <= 4.15\nsquared_error = 68.357\nsamples = 47\nvalue = 39.657'),
 Text(0.03652314027392355, 0.8409090909090909, 'X[0] <= 3.325\nsquared_error = 59.164\nsamples = 34\nvalue = 41.544'),
 Text(0.0186196401396473, 0.7954545454545454, 'X[0] <= 3.145\nsquared_error = 41.18\nsamples = 17\nvalue = 43.635'),
 Text(0.0128905200966789, 0.75, 'X[0] <= 1.855\nsquared_error = 42.374\nsamples = 13\nvalue = 42.177'),
 Text(0.0100259600751947, 0.7045454545454546, 'squared_error = 0.0\nsamples = 1\nvalue = 50.0'),
 Text(0.015755080118163102, 0.7045454545454546, 'X[0] <= 2.875\nsquared_error = 40.38\nsamples = 12\nvalue = 41.525'),
 Text(0.0057291200429684, 0.6590909090909091, 'X[0] <= 2.225\nsquared_error = 8.509\nsamples = 3\nvalue = 37.667'),
 Text(0.0028645600214842, 0.6136363636363636, 'squared_error = 0.0\nsamples = 1\nvalue = 34.9'),
 Text(0.0085936800644526, 0.6136363636363636, 'X[0] <= 2.67\nsquared_error = 7.023\nsamples = 2\nvalue = 39.05'),
 Text(0.0057291200429684, 0.5681818181818182, 'squared_error = 0.0\nsamples = 1\nvalue = 41.7'),
 Text(0.0114582400859368, 0.5681818181818182, 'squared_error = 0.0\nsamples = 1\nvalue = 36.4'),
 Text(0.0257810401933578, 0.6590909090909091, 'X[0] <= 2.975\nsquared_error = 44.388\nsamples = 9\nvalue = 42.811'),
 Text(0.0200519201503894, 0.6136363636363636, 'X[0] <= 2.95\nsquared_error = 51.667\nsamples = 4\nvalue = 45.85'),
 Text(0.0171873601289052, 0.5681818181818182, 'X[0] <= 2.91\nsquared_error = 68.89\nsamples = 2\nvalue = 41.7'),
 Text(0.014322800107421001, 0.5227272727272727, 'squared_error = 0.0\nsamples = 1\nvalue = 50.0'),
 Text(0.0200519201503894, 0.5227272727272727, 'squared_error = 0.0\nsamples = 1\nvalue = 33.4'),
```

jupyter Iris Classification2 Last Checkpoint: 13 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

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

In [2]:
```
iris=pd.read_csv('Iris.csv')
```

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

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [4]:
```
iris['Species'].unique()
```

Out[4]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

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

Out[5]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
count	150.000000	150.000000	150.000000	150.000000	150.000000	150

jupyter Iris Classification2 Last Checkpoint: 13 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

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

Out[5]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
count	150.000000	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	NaN	Iris-setosa
freq	NaN	NaN	NaN	NaN	NaN	50
mean	75.500000	5.843333	3.054000	3.758667	1.198667	NaN
std	43.445368	0.828066	0.433594	1.764420	0.763161	NaN
min	1.000000	4.300000	2.000000	1.000000	0.100000	NaN
25%	38.250000	5.100000	2.800000	1.600000	0.300000	NaN
50%	75.500000	5.800000	3.000000	4.350000	1.300000	NaN
75%	112.750000	6.400000	3.300000	5.100000	1.800000	NaN
max	150.000000	7.900000	4.400000	6.900000	2.500000	NaN

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
```

Labs/ | Iris Classification3 - Jupyter Not | Iris Classification2 - Jupyter Not | Iris Classification1 - Jupyter Not

localhost:8888/notebooks/Labs/Iris%20Classification2.ipynb

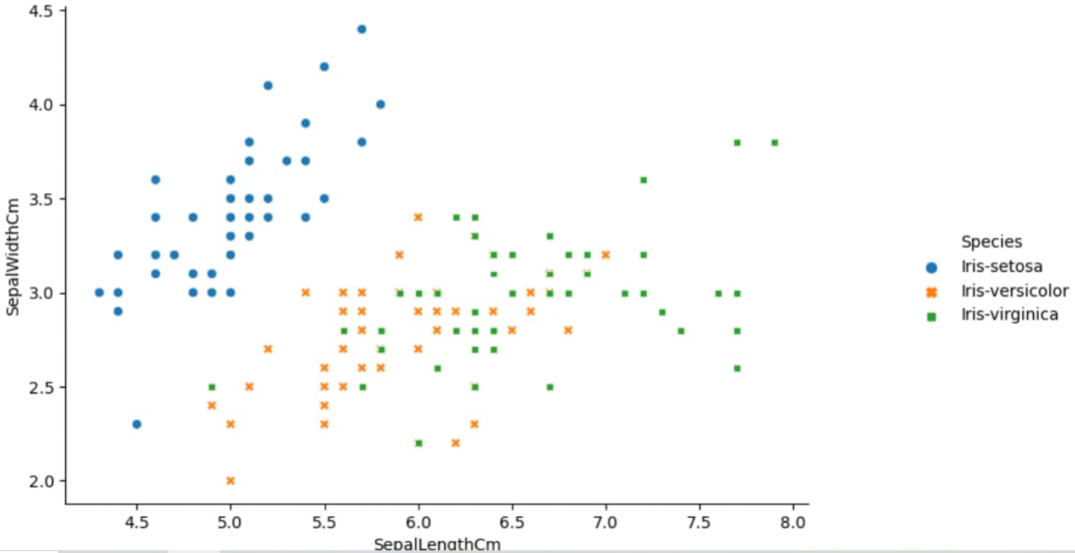
Addons Store | AliExpress | Booking.com | Facebook | Google | YouTube | Neon light button... | All Modules - Front... | techprivacy/fullstac... | CampK12 Live | Boo... | World English Instit... | Newton School

jupyter Iris Classification2 Last Checkpoint: 15 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [7]:
```
g=sns.relplot(x='SepalLengthCm',y='SepalWidthCm',data=iris,hue='Species',style='Species')
g.fig.set_size_inches(10,5)
plt.show()
```



Type here to search

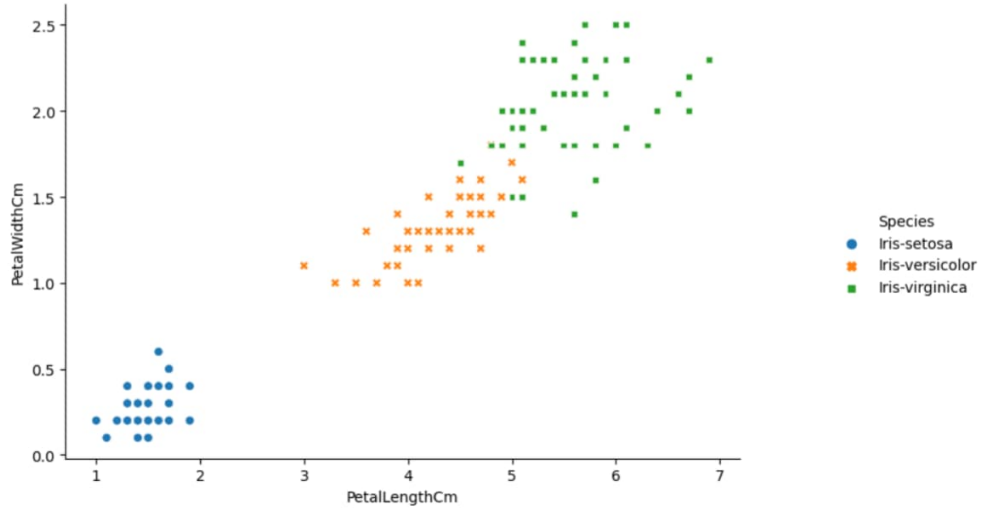
27°C Smoke ENG 17:08 22-11-2023

jupyter Iris Classification2 Last Checkpoint: 15 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [8]:
```
g=sns.relplot(x='PetalLengthCm',y='PetalWidthCm',data=iris,hue='Species',style='Species')
g.fig.set_size_inches(10,5)
plt.show()
```



In [9]: sns.pairplot(iris, hue="Species")

Labs/ | Iris Classification3 - Jupyter Not | Iris Classification2 - Jupyter Not | Iris Classification1 - Jupyter Not

localhost:8888/notebooks/Labs/Iris%20Classification2.ipynb

Addons Store | AliExpress | Booking.com | Facebook | Google | YouTube | Neon light button... | All Modules - Front... | techprivacy/fullstac... | CampK12 Live | Boo... | World English Instit... | Newton School

jupyter Iris Classification2 Last Checkpoint: 17 minutes ago (autosaved)

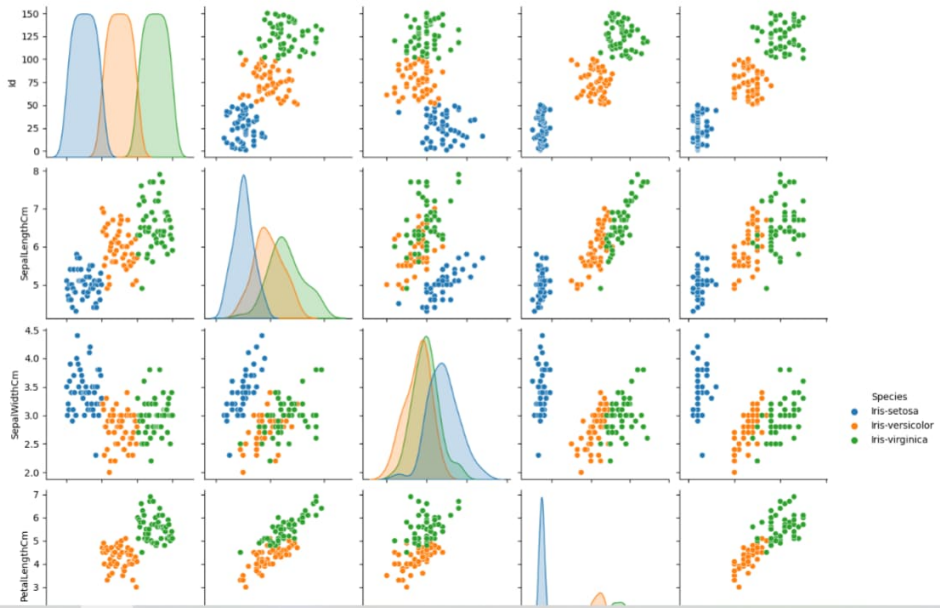
Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run | Code

In [9]:
```
sns.pairplot(iris,hue="Species")
plt.show()
```

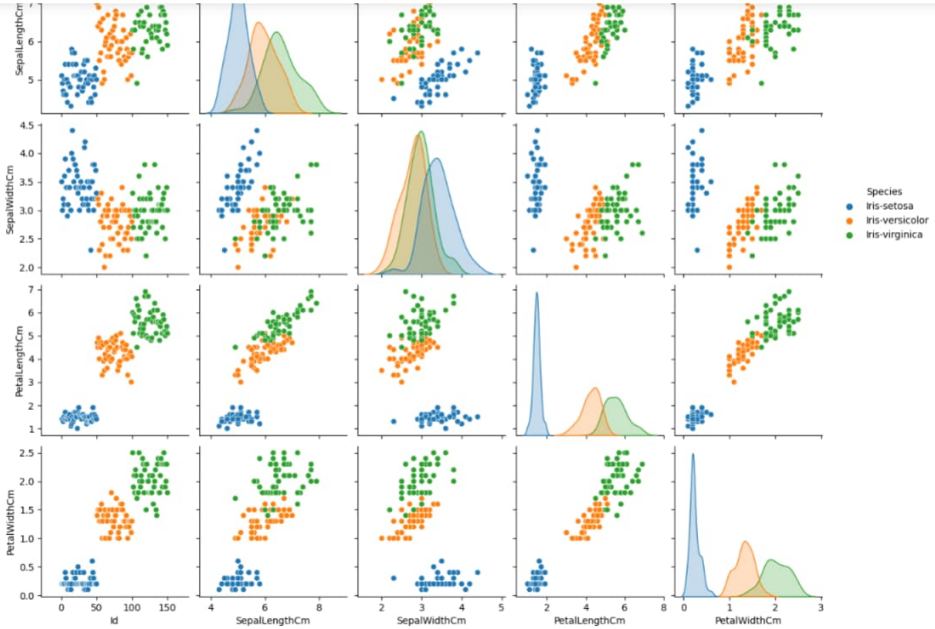


Type here to search

27°C Smoke

ENG

17:11
22-11-2023


SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Id
Species
Iris-setosa
Iris-versicolor
Iris-virginica

Labs/ | Iris Classification3 - Jupyter Not | Iris Classification2 - Jupyter Not | Iris Classification1 - Jupyter Not

localhost:8888/notebooks/Labs/Iris%20Classification2.ipynb

Addons Store | AliExpress | Booking.com | Facebook | Google | YouTube | Neon light button... | All Modules - Front... | techprivacy/fullstac... | CampK12 Live | Boo... | World English Instit... | Newton School

jupyter Iris Classification2 Last Checkpoint: 18 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [10]:
```
X=iris.iloc[:,0:4].values
y=iris.iloc[:,4].values
```

In [11]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

In [12]:
```
from sklearn.model_selection import KFold,train_test_split,cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
```

In [13]:
```
#Train and Test split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=0)
```

In [14]:
```
from sklearn.metrics import make_scorer, accuracy_score,precision_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score ,precision_score,recall_score,f1_score
```

In [15]:
```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
Y_pred = gaussian.predict(X_test)
accuracy_nb=round(accuracy_score(y_test,Y_pred)* 100, 2)
acc_gaussian = round(gaussian.score(X_train, y_train) * 100, 2)

cm = confusion_matrix(y_test, Y_pred)
```

Type here to search 27°C Smoke ENG 17:12 22-11-2023

jupyter Iris Classification2 Last Checkpoint: 19 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
f1 = f1_score(y_test,Y_pred,average='micro')
print('Confusion matrix for Naive Bayes\n',cm)
print('accuracy_Naive Bayes: %.3f' %accuracy)
print('precision_Naive Bayes: %.3f' %precision)
print('recall_Naive Bayes: %.3f' %recall)
print('f1-score_Naive Bayes : %.3f' %f1)
```

```
Confusion matrix for Naive Bayes
 [[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [1 4 1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 2 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 2 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 2 0 0 1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 4 2 1 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 2 0 0 1 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]]
accuracy_Naive Bayes: 0.289
precision_Naive Bayes: 0.289
recall_Naive Bayes: 0.289
f1-score_Naive Bayes : 0.289
```

F1-score_Naive Bayes : 0.289

In [16]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [17]:
```
# Import the dataset using Seaborn library
from sklearn.datasets import load_iris
iris = load_iris()
iris = sns.load_dataset('iris')
```

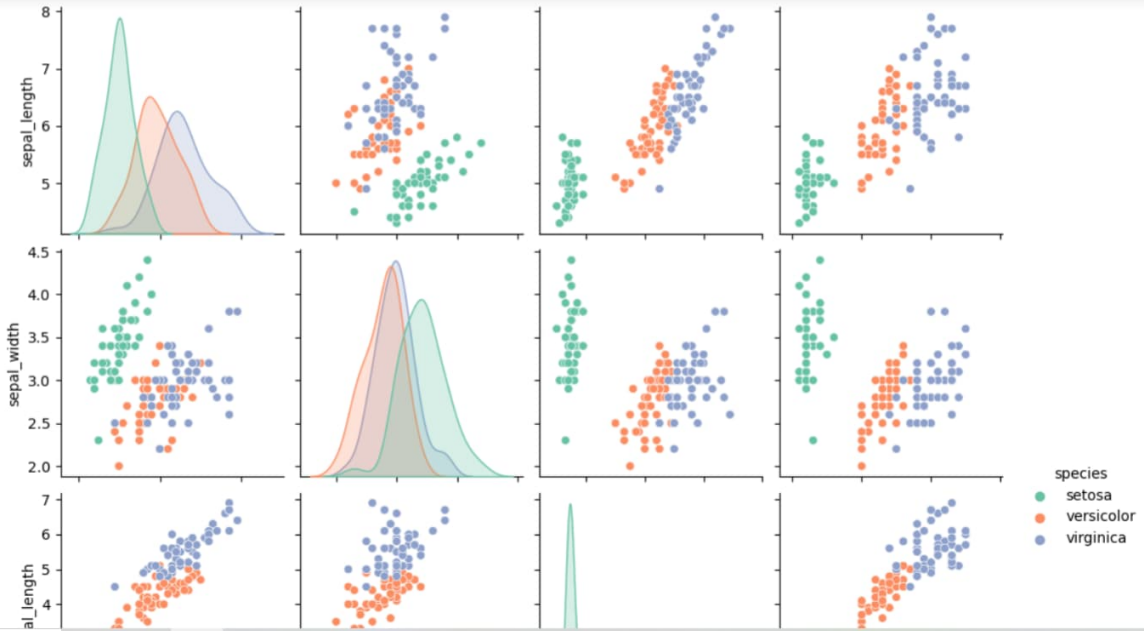
In [18]:
```
# Checking the dataset
iris.head()
```

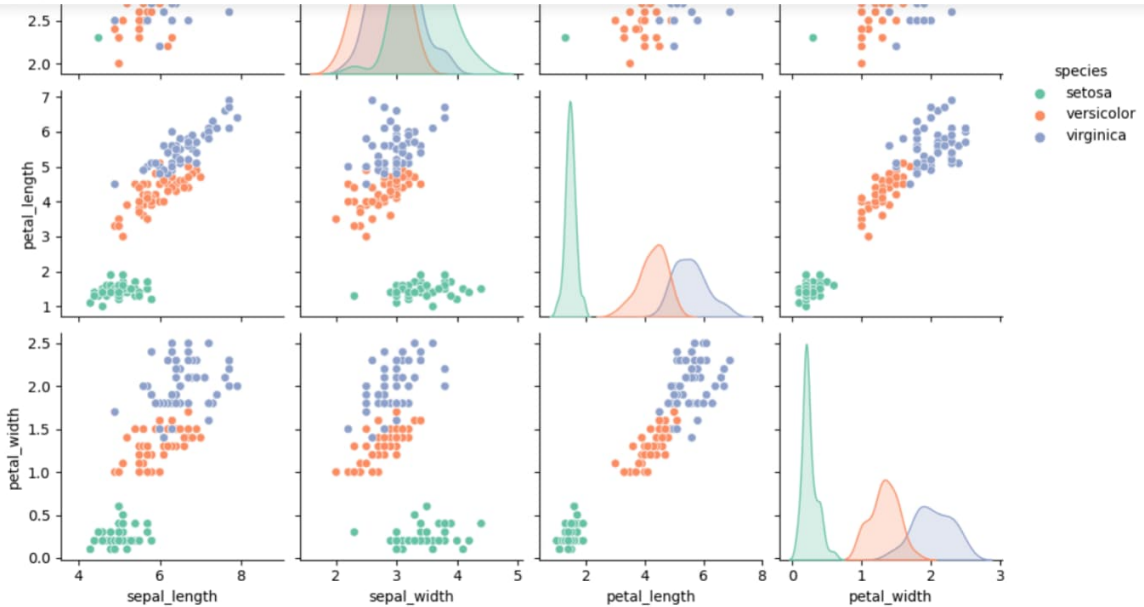
Out[18]:

	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 [19]:
```
# Creating a pairplot to visualize the similarities and especially difference between the species
sns.pairplot(data=iris, hue='species', palette='Set2')
```

Out[19]:
```
<seaborn.axisgrid.PairGrid at 0x20feeb739a0>
```


sepal_length
sepal_width
species
setosa
versicolor
virginica


species
setosa
versicolor
virginica
petal_length
petal_width
sepal_length
sepal_width
petal_length
petal_width

jupyter Iris Classification2 Last Checkpoint: 22 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [20]:
```
from sklearn.model_selection import train_test_split
# Separating the independent variables from dependent variables
x=iris.iloc[:,:-1]
y=iris.iloc[:,4]
x_train,x_test, y_train, y_test=train_test_split(x,y,test_size=0.30)
```

In [21]:
```
from sklearn.svm import SVC
model=SVC()
```

In [22]:
```
model.fit(x_train, y_train)
```

Out[22]:
```
SVC()
```

In [23]:
```
pred=model.predict(x_test)
```

In [24]:
```
# Importing the classification report and confusion matrix
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,pred))
print(classification_report(y_test, pred))
```

```
[[16  0  0]
 [ 0 15  0]
 [ 0  0 14]]
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        16
  versicolor       1.00      1.00      1.00        15
   virginica       1.00      1.00      1.00        14
```

jupyter Iris Classification2 Last Checkpoint: 22 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

```
              accuracy                           1.00        45
             macro avg       1.00      1.00      1.00        45
          weighted avg       1.00      1.00      1.00        45
```

In [25]:
```
import pandas as pd
import sklearn as sk
from sklearn import datasets
from sklearn.datasets import fetch_california_housing

housing_data = sk.datasets.fetch_california_housing()
housing_data
```

Out[25]:
```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55555556,
           37.88      , -122.23      ],
        [   8.3014    ,   21.        ,    6.23813708, ...,    2.10984183,
           37.86      , -122.22      ],
        [   7.2574    ,   52.        ,    8.28813559, ...,    2.80225989,
           37.85      , -122.24      ],
        ...,
        [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
           39.43      , -121.22      ],
        [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
           39.43      , -121.32      ],
        [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
           39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
```

```
        [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
           39.43      , -121.22      ],
        [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
           39.43      , -121.32      ],
        [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
           39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing dataset\n--------------------------\n\n**Data Set Characteristics:**\n\n    :Number of Instances: 20640\n\n    :Number of Attributes: 8 numeric, predictive attributes and the target\n\n    :Attribute Information:\n        - MedInc        median income in block group\n        - HouseAge      median house age in block group\n        - AveRooms      average number of rooms per household\n        - AveBedrms     average number of bedrooms per household\n        - Population    block group population\n        - AveOccup      average number of household members\n        - Latitude      block group latitude\n        - Longitude     block group longitude\n\n    :Missing Attribute Values: None\n\nThis dataset was obtained from the StatLib repository.\nhttps://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html\n\nThe target variable is the median house value for California districts,\nexpressed in hundreds of thousands of dollars ($100,000).\n\nThis dataset was derived from the 1990 U.S. census, using one row per census\nblock group. A block group is the smallest geographical unit for which the U.S.\nCensus Bureau publishes sample data (a block group typically has a population\nof 600 to 3,000 people).\n\nAn household is a group of people residing within a home. Since the average\nnumber of rooms and bedrooms in this dataset are provided per household, these\ncolumns may take surpinsingly large values for block groups with few households\nand many empty houses, such as vacation resorts.\n\nIt can be downloaded/loaded using the\n:func:`sklearn.datasets.fetch_california_housing` function.\n\n.. topic:: References\n\n    - Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregressions,\n      Statistics and Probability Letters, 33 (1997) 291-297\n'}
```

jupyter Iris Classification2 Last Checkpoint: 23 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [26]:
```
X = pd.DataFrame(housing_data.data, columns=housing_data.feature_names)
y = housing_data.target
print(X)
print(y)
```

```
       MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0      8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1      8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2      7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3      5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4      3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85
...       ...       ...       ...        ...         ...       ...       ...
20635  1.5603      25.0  5.045455   1.133333       845.0  2.560606     39.48
20636  2.5568      18.0  6.114035   1.315789       356.0  3.122807     39.49
20637  1.7000      17.0  5.205543   1.120092      1007.0  2.325635     39.43
20638  1.8672      18.0  5.329513   1.171920       741.0  2.123209     39.43
20639  2.3886      16.0  5.254717   1.162264      1387.0  2.616981     39.37

       Longitude
0        -122.23
1        -122.22
2        -122.24
3        -122.25
4        -122.25
...          ...
20635    -121.09
20636    -121.21
20637    -121.22
20638    -121.32
20639    -121.24

[20640 rows x 8 columns]
```

```
1        -122.22
2        -122.24
3        -122.25
4        -122.25
...          ...
20635    -121.09
20636    -121.21
20637    -121.22
20638    -121.32
20639    -121.24

[20640 rows x 8 columns]
[4.526 3.585 3.521 ... 0.923 0.847 0.894]
```

In [27]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.10, random_state=0)
```

In [28]:
```
from sklearn.svm import SVR
regressor = SVR(kernel = 'rbf')
regressor.fit(X, y)
```

Out[28]:
```
SVR()
```

In [29]:
```
y_pred = regressor.predict(X_test)
print(y_pred)
```

```
[1.7554298  1.84194121 1.77758078 ... 1.79164334 1.92946582 1.77307878]
```

In [ ]:

Labs/ | Iris Classification3 - Jupyter Not | Iris Classification2 - Jupyter Not | Iris Classification1 - Jupyter Not

localhost:8888/notebooks/Labs/Iris%20Classification3.ipynb

Addons Store | AliExpress | Booking.com | Facebook | Google | YouTube | Neon light button... | All Modules - Front... | techprivacy/fullstac... | CampK12 Live | Boo... | World English Instit... | Newton School

jupyter Iris Classification3 Last Checkpoint: 24 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import MinMaxScaler
from sklearn.datasets import load_iris
```

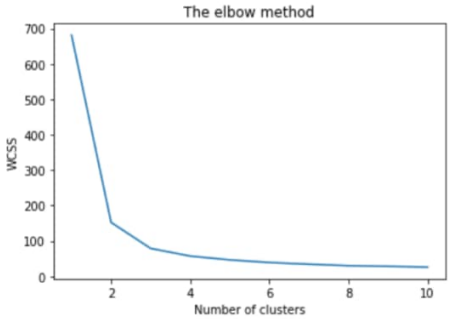
In [ ]:
```
iris = load_iris()
iris = sns.load_dataset('iris')
iris.head()
x = iris.iloc[:, [0, 1, 2, 3]].values
print(x)
```

In [ ]:
```
#Finding the optimum number of clusters for k-means classification
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)
    print(kmeans.cluster_centers_)
    wcss.append(kmeans.inertia_)

score = silhouette_score(x, kmeans.labels_, metric='euclidean')
print(score)
```

Using the elbow method to determine the optimal number of clusters for k-means clustering

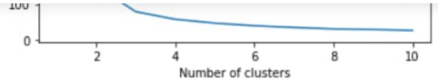
Type here to search 27°C Smoke ENG 17:17 22-11-2023

In [ ]:
```
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()
```



HIERARCHICAL CLUSTERING - SINGLE METHOD

In [ ]:
```
from scipy.cluster.hierarchy import dendrogram, linkage
iris = load_iris()
iris = sns.load_dataset('iris')
iris.head()
```



HIERARCHICAL CLUSTERING - SINGLE METHOD

In [ ]:
```
from scipy.cluster.hierarchy import dendrogram, linkage
iris = load_iris()
iris = sns.load_dataset('iris')
iris.head()
```

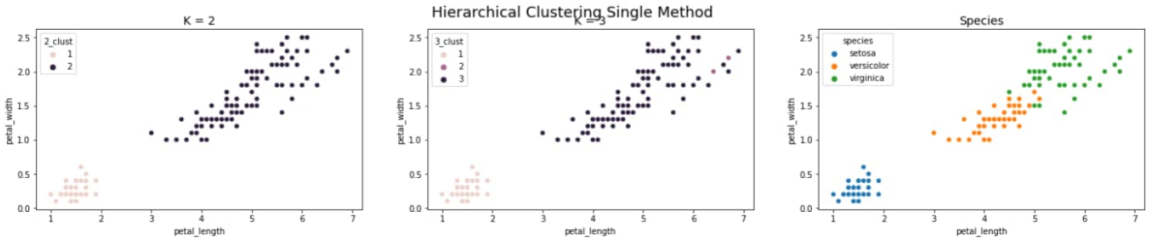
Out[8]:

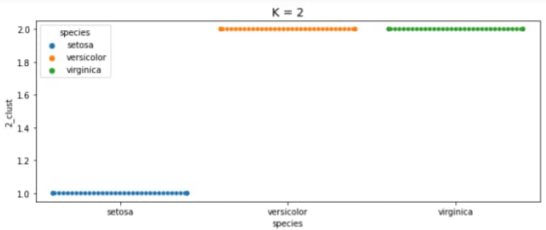
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

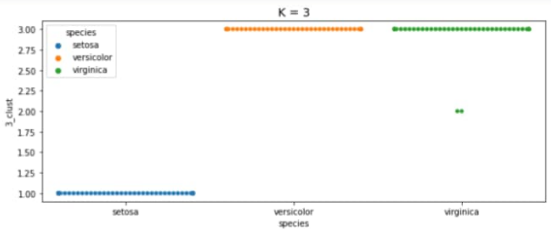
In [ ]:
```
dist_sin = linkage(iris.loc[:,["sepal_length","sepal_width","petal_length","petal_width"]],method="single")
#print(dist_sin)
plt.figure(figsize=(18,6))
dendrogram(dist_sin, leaf_rotation=90)
plt.xlabel('Index')
plt.ylabel('Distance')
plt.suptitle("DENDROGRAM SINGLE METHOD",fontsize=18)
plt.show()
```

In [ ]:
```
plt.figure(figsize=(24,4))

plt.suptitle("Hierarchical Clustering Single Method",fontsize=18)

plt.subplot(1,3,1)
plt.title("K = 2",fontsize=14)
sns.scatterplot(x="petal_length",y="petal_width", data=iris_SM, hue="2_clust")

plt.subplot(1,3,2)
plt.title("K = 3",fontsize=14)
sns.scatterplot(x="petal_length",y="petal_width", data=iris_SM, hue="3_clust")

plt.subplot(1,3,3)
plt.title("Species",fontsize=14)
sns.scatterplot(x="petal_length",y="petal_width", data=iris_SM, hue="species")
```

Out[12]: <Axes: title={'center': 'Species'}, xlabel='petal_length', ylabel='petal_width'>

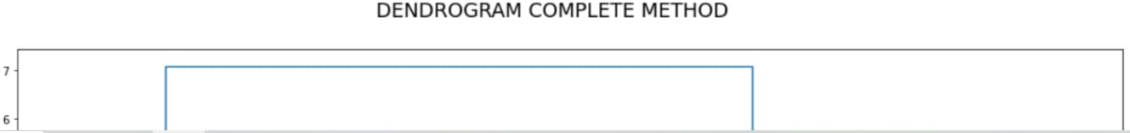


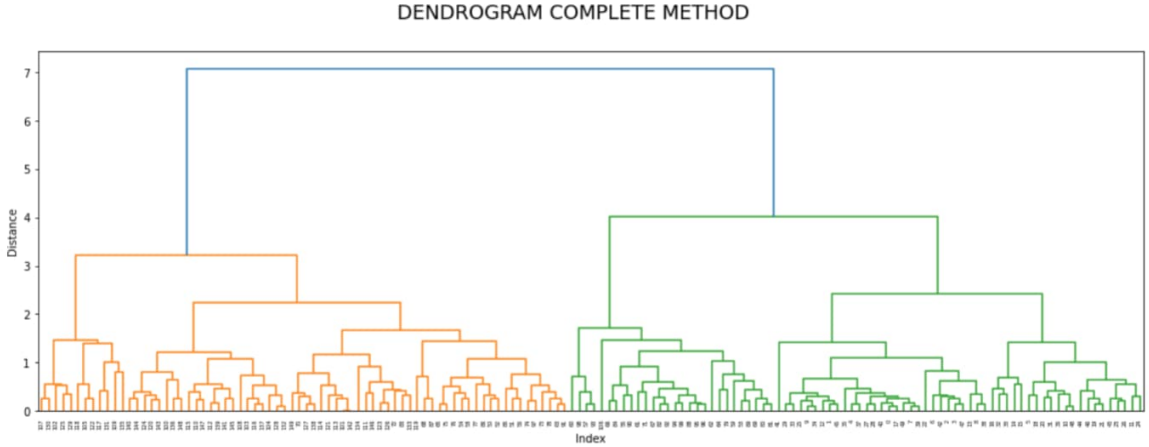




HIERARCHICAL CLUSTERING - COMPLETE METHOD

In [ ]:
```
dist_comp = linkage(iris.loc[:,["sepal_length","sepal_width","petal_length","petal_width"]],method="complete")

plt.figure(figsize=(18,6))
dendrogram(dist_comp, leaf_rotation=90)
plt.xlabel('Index')
plt.ylabel('Distance')
plt.suptitle("DENDROGRAM COMPLETE METHOD",fontsize=18)
plt.show()
```





In [ ]:
```
iris_CM=iris.copy()
iris_CM['2_clust']=fcluster(dist_comp,2, criterion='maxclust')
iris_CM['3_clust']=fcluster(dist_comp,3, criterion='maxclust')
iris_CM.head()
```

Out[15]:

	sepal_length	sepal_width	petal_length	petal_width	species	2_clust	3_clust

Out[15]:

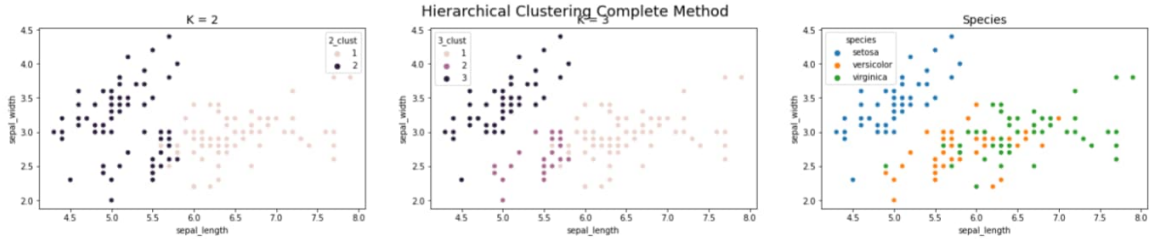
	sepal_length	sepal_width	petal_length	petal_width	species	2_clust	3_clust
0	5.1	3.5	1.4	0.2	setosa	2	3
1	4.9	3.0	1.4	0.2	setosa	2	3
2	4.7	3.2	1.3	0.2	setosa	2	3
3	4.6	3.1	1.5	0.2	setosa	2	3
4	5.0	3.6	1.4	0.2	setosa	2	3

In [ ]:
```
plt.figure(figsize=(24,4))

plt.suptitle("Hierarchical Clustering Complete Method",fontsize=18)

plt.subplot(1,3,1)
plt.title("K = 2",fontsize=14)
sns.scatterplot(x="sepal_length",y="sepal_width", data=iris_CM, hue="2_clust")

plt.subplot(1,3,2)
plt.title("K = 3",fontsize=14)
sns.scatterplot(x="sepal_length",y="sepal_width", data=iris_CM, hue="3_clust")

plt.subplot(1,3,3)
plt.title("Species",fontsize=14)
sns.scatterplot(x="sepal_length",y="sepal_width", data=iris_CM, hue="species")
```

Out[16]: <Axes: title={'center': 'Species'}, xlabel='sepal_length', ylabel='sepal_width'>

```
plt.subplot(1,3,2)
plt.title("K = 3",fontsize=14)
sns.scatterplot(x="sepal_length",y="sepal_width", data=iris_CM, hue="3_clust")

plt.subplot(1,3,3)
plt.title("Species",fontsize=14)
sns.scatterplot(x="sepal_length",y="sepal_width", data=iris_CM, hue="species")
```

Out[16]: <Axes: title={'center': 'Species'}, xlabel='sepal_length', ylabel='sepal_width'>



In [ ]:

```
plt.figure(figsize=(24,4))
plt.subplot(1,2,1)
plt.title("K = 2",fontsize=14)
sns.swarmplot(x="species",y="2_clust", data=iris_CM, hue="species")

plt.subplot(1,2,2)
plt.title("K = 3",fontsize=14)
sns.swarmplot(x="species",y="3_clust", data=iris_CM, hue="species")
```

```
want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.9/dist-packages/seaborn/categorical.py:3544: UserWarning: 12.0% of the points cannot be placed; you may
want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

Out[17]: <Axes: title={'center': 'K = 3'}, xlabel='species', ylabel='3_clust'>