Support Vector Machines - Iris Flower Classification Project

Nawal Obaid

9/7/2022

Import the needed data

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

Get the data - it is a seaborn built in dataset

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.info()
```

```
<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 [5]:
```
iris['species'].unique()
```

Out[5]:
```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

so what we need to do here is to classify the data based on the three species we have in the dataset

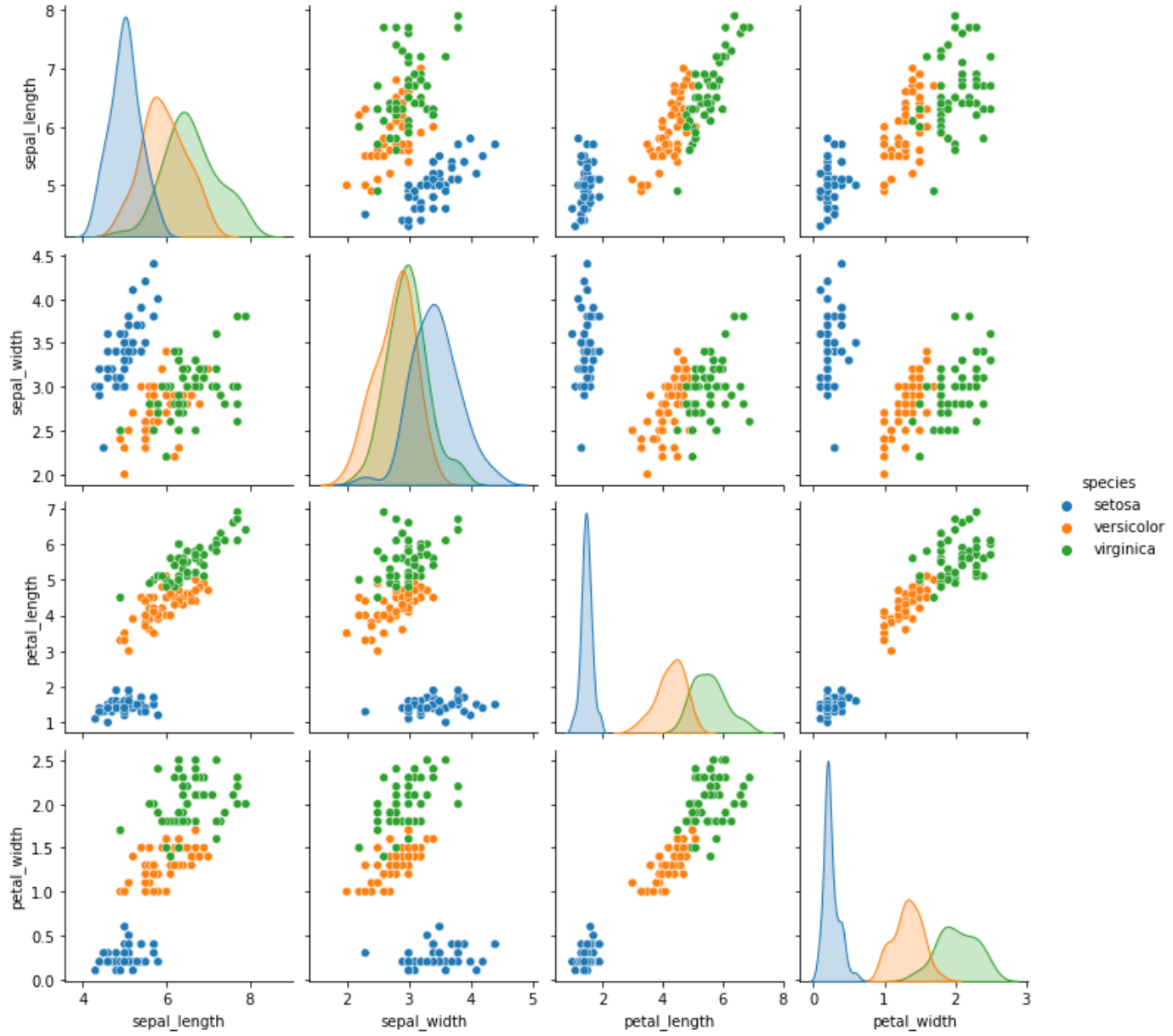
Exploratory Data Analysis

Which flower species seems to be the most separable?

Answer is: A) Setosa

In [6]:
```
sns.pairplot(data=iris, hue='species')
```

Out[6]:
```
<seaborn.axisgrid.PairGrid at 0x14806b38280>
```



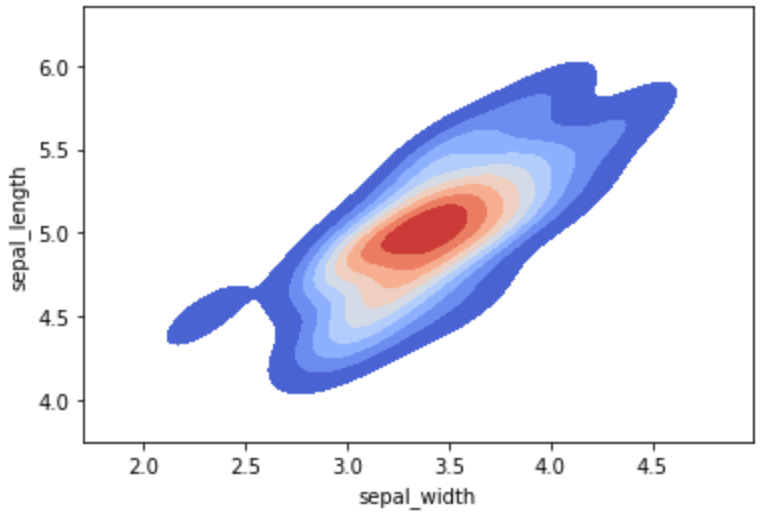
In [7]:
```
# let's create a kde plot of sepal_length versus sepal width for setosa species of flowe

setosa = iris[iris['species']== 'setosa']
setosa.head()
```

Out[7]:

	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 [9]:
```
sns.kdeplot(x = setosa['sepal_width'], y= setosa['sepal_length'], cmap = 'coolwarm',shad
```

Out[9]:
```
<AxesSubplot:xlabel='sepal_width', ylabel='sepal_length'>
```



Train Test Split

In [11]:
```
X = iris.drop('species', axis = 1)
y = iris['species']
```

In [12]:
```
X.head()
```

Out[12]:

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

In [13]:
```
from sklearn.model_selection import train_test_split
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10
```

In [15]:
```
# instantiate a SV classifier
from sklearn.svm import SVC
```

In [16]:
```
classifier = SVC()
```

In [17]:
```
# train/fit the classifier to the dataset
classifier.fit(X_train, y_train)
```

Out[17]:
```
▾ SVC
SVC()
```

In [21]:
```
# test the classifer and see the predictions
predictions = classifier.predict(X_test)
```

In [22]:
```
# let's see the performance of this classifier using confusion atrix and classification
from sklearn.metrics import classification_report, confusion_matrix
```

In [23]:
```
print('**************Classification Report*************************')
print(classification_report(y_test, predictions))
print('****************Confusion Matrix*****************************')
print(confusion_matrix(y_test, predictions))
```

```
**************Classification Report*************************
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        13
  versicolor       1.00      0.95      0.97        20
   virginica       0.92      1.00      0.96        12

    accuracy                           0.98        45
   macro avg       0.97      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45

****************Confusion Matrix*****************************
[[13  0  0]
 [ 0 19  1]
 [ 0  0 12]]
```

In [24]:
```
cm = confusion_matrix(y_test, predictions)
sns.heatmap(cm, annot = True, fmt = "d", cmap ='Blues')
```

Out[24]:
```
<AxesSubplot:>
```