

importing all necessary libraries for our project


```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
%matplotlib inline

from warnings import filterwarnings
filterwarnings('ignore')
```

now we have to load dataset into the notebook

```
# Importing pandas to the cell
import pandas as pd

iris =pd.read_csv('/content/iris.csv')
iris.head(151)
```



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

150 rows × 5 columns

visuaization of our dataset

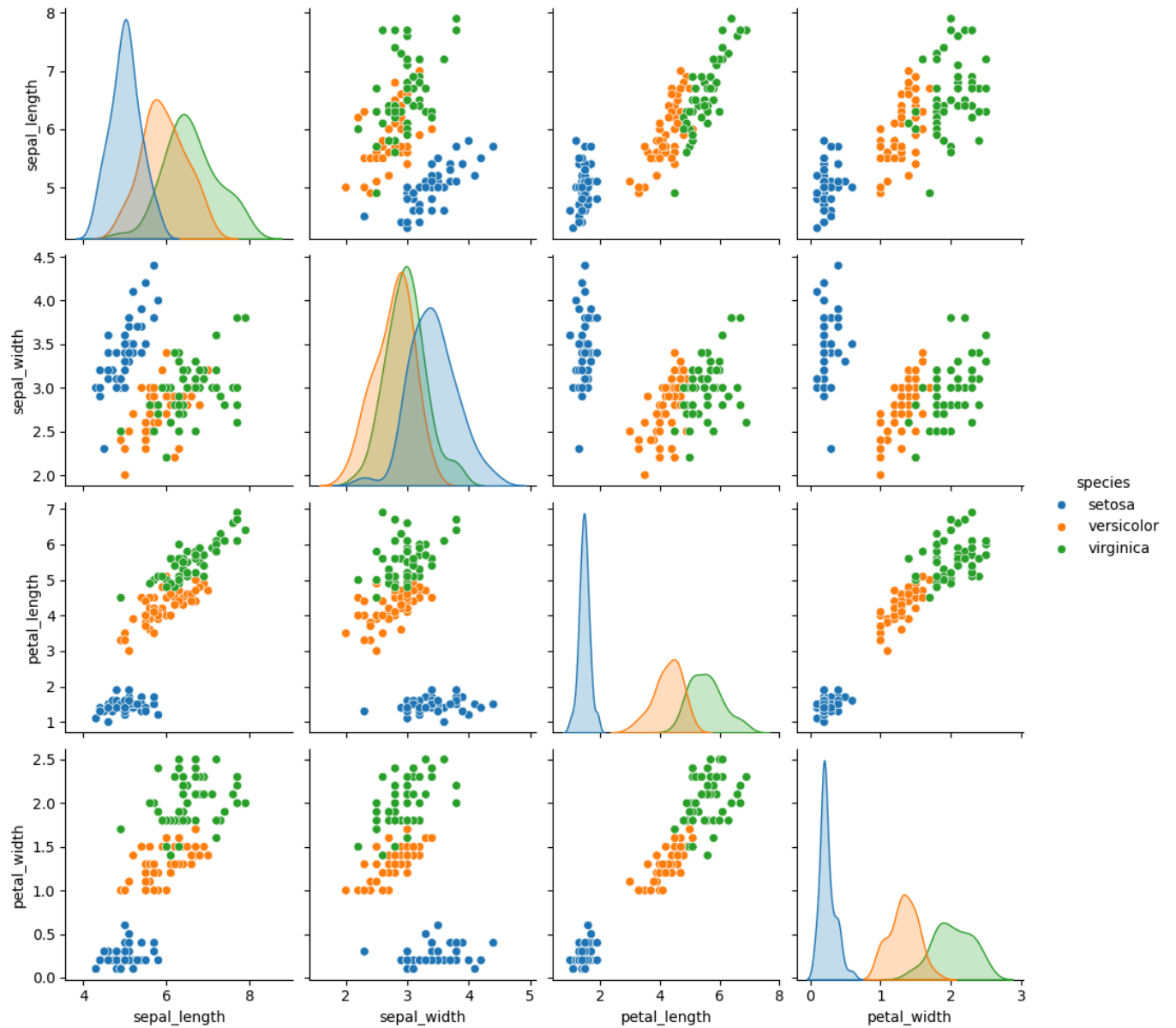
```
# detailed descriptions if dataset
iris.describe()
```



	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

```
#graphical visualization of dataset  
sns.pairplot(iris, hue='species')
```

```
>> <seaborn.axisgrid.PairGrid at 0x7ae988503c10>
```



```
# bar graph representation
iris.hist()
plt.show
```



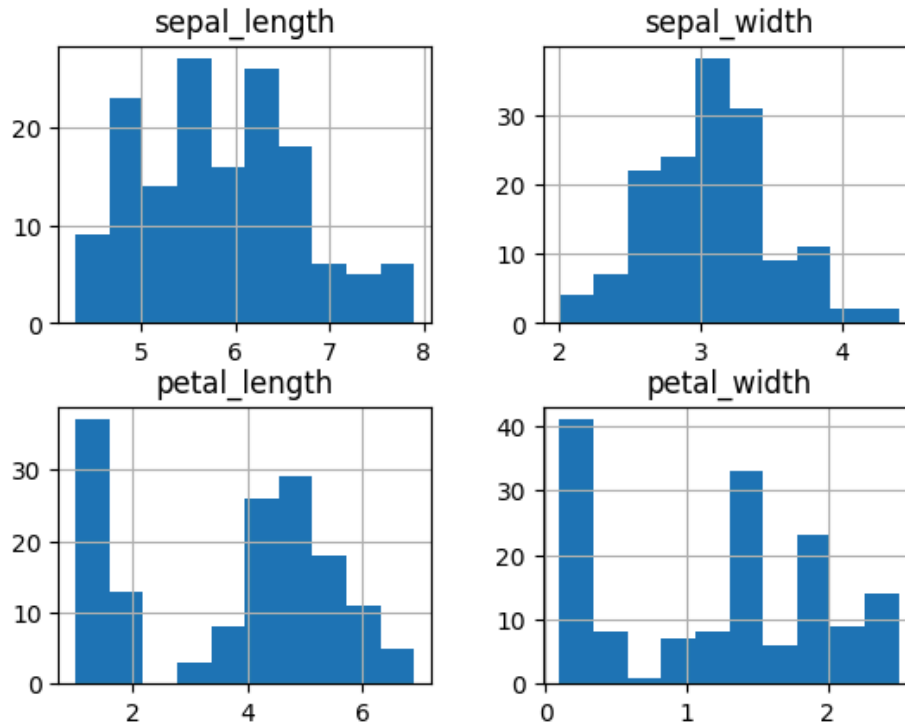
```
matplotlib.pyplot.show
def show(*args, **kwargs) -> None
```

Display all open figures.

Parameters

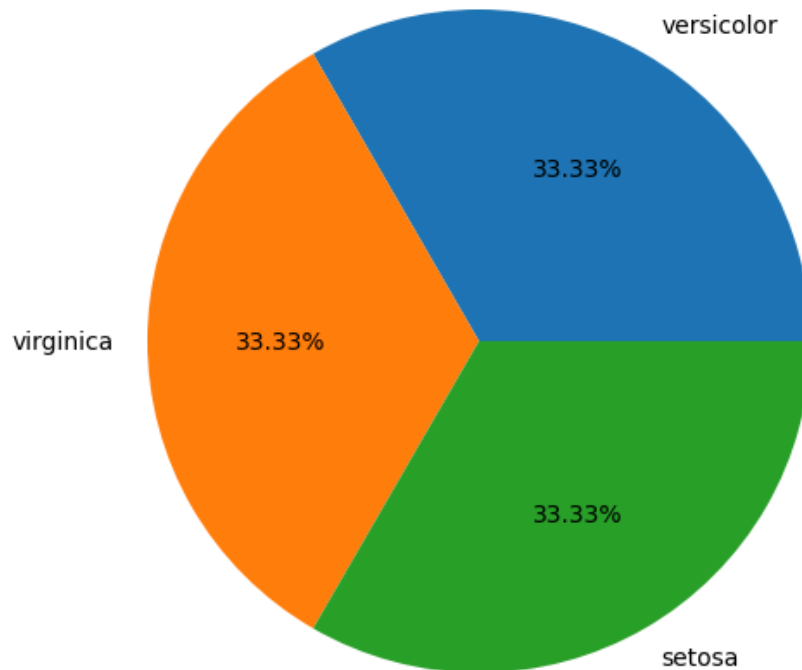
block : bool, optional

Whether to wait for all figures to be closed before returning.



```
# pie chart representation
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.axis('equal')
l = ['versicolor','virginica','setosa']
s = [50,50,50]
ax.pie(s,labels=l,autopct='%1.2f%%')
plt.show()
ax.axis('equal')
l = ['versicolor','virginica','setosa']
```

⚡ WARNING:matplotlib.axes._base:Ignoring fixed x limits to fulfill fixed data aspect with adjust.



seperating input columns and output columns

```
data = iris.values
x = data[:,0:5] # This is correct to extract the features
y = data[:,4]   # Changed this line: Access the species column using index 4 (the 5th column)
# check input column and output columns
```

splitting the data for training/testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

support vactor machine algorithm

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

⚡ [Show hidden output](#)

```
prediction_1 = model_SVC.predict(x_test)
prediction_1 = model_SVC.predict(x_test)
# to calculate the accuracy of model
from sklearn.metrics import accuracy_score
print("accuracy score:",accuracy_score(y_test,prediction_1)*100)
```

⇒ accuracy score: 100.0

MODEL 2:logistic regression

```
from sklearn.linear_model import LogisticRegression
model_LR = LogisticRegression() # Create an instance of LogisticRegression
model_LR.fit(x_train, y_train)
```

⇒

▼ LogisticRegression ⓘ ?
LogisticRegression()

```
prediction2=model_LR.predict(x_test)
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report # Corrected the import statement
print("Accuracy score:",accuracy_score(y_test,prediction2)*100)
```

⇒ Accuracy score: 100.0

MODEL 2: decision tree classifier