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


iris=pd.read_csv("iris.csv")
print(iris)
```

```
     Unnamed: 0  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width  \
0             1           5.1          3.5           1.4          0.2
1             2           4.9          3.0           1.4          0.2
2             3           4.7          3.2           1.3          0.2
3             4           4.6          3.1           1.5          0.2
4             5           5.0          3.6           1.4          0.2
..          ...           ...          ...           ...          ...
145         146           6.7          3.0           5.2          2.3
146         147           6.3          2.5           5.0          1.9
147         148           6.5          3.0           5.2          2.0
148         149           6.2          3.4           5.4          2.3
149         150           5.9          3.0           5.1          1.8

        Species
0        setosa
1        setosa
2        setosa
3        setosa
4        setosa
..          ...
145  virginica
146  virginica
147  virginica
148  virginica
149  virginica

[150 rows x 6 columns]
```

```
print(iris.shape)
```

```
(150, 6)
```

```
print(iris.describe())
```

```
        Unnamed: 0  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
count  150.000000    150.000000   150.000000    150.000000   150.000000
mean    75.500000      5.843333     3.057333      3.758000     1.199333
std     43.445368      0.828066     0.435866      1.765298     0.762238
min      1.000000      4.300000     2.000000      1.000000     0.100000
25%     38.250000      5.100000     2.800000      1.600000     0.300000
50%     75.500000      5.800000     3.000000      4.350000     1.300000
75%    112.750000      6.400000     3.300000      5.100000     1.800000
max    150.000000      7.900000     4.400000      6.900000     2.500000
```

```
#Checking for null values
print(iris.isna().sum())
print(iris.describe())
```

```
Unnamed: 0      0
Sepal.Length    0
Sepal.Width     0
Petal.Length    0
Petal.Width     0
Species         0
dtype: int64
       Unnamed: 0  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
count  150.000000    150.000000   150.000000    150.000000   150.000000
mean    75.500000      5.843333     3.057333      3.758000     1.199333
std     43.445368      0.828066     0.435866      1.765298     0.762238
min      1.000000      4.300000     2.000000      1.000000     0.100000
25%     38.250000      5.100000     2.800000      1.600000     0.300000
50%     75.500000      5.800000     3.000000      4.350000     1.300000
75%    112.750000      6.400000     3.300000      5.100000     1.800000
max    150.000000      7.900000     4.400000      6.900000     2.500000
```

```
iris.head()
```

	Unnamed: 0	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	1	5.1	3.5	1.4	0.2	setosa
1	2	4.9	3.0	1.4	0.2	setosa
2	3	4.7	3.2	1.3	0.2	setosa
3	4	4.6	3.1	1.5	0.2	setosa

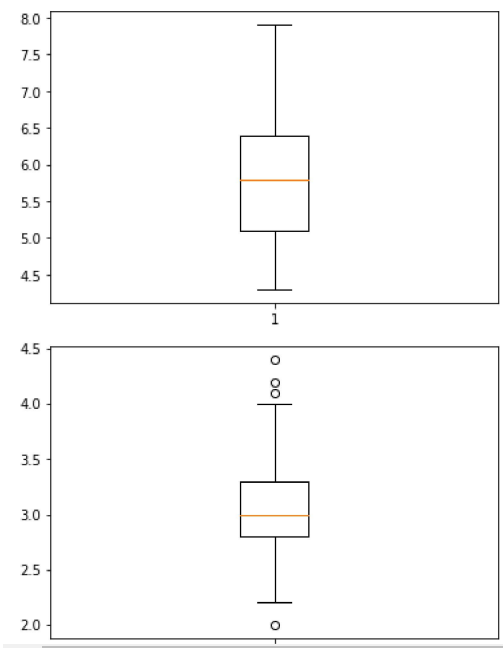
```
iris.head(150)
```

	Unnamed: 0	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	1	5.1	3.5	1.4	0.2	setosa
1	2	4.9	3.0	1.4	0.2	setosa
2	3	4.7	3.2	1.3	0.2	setosa
3	4	4.6	3.1	1.5	0.2	setosa
4	5	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	virginica
146	147	6.3	2.5	5.0	1.9	virginica
147	148	6.5	3.0	5.2	2.0	virginica
148	149	6.2	3.4	5.4	2.3	virginica
149	150	5.9	3.0	5.1	1.8	virginica

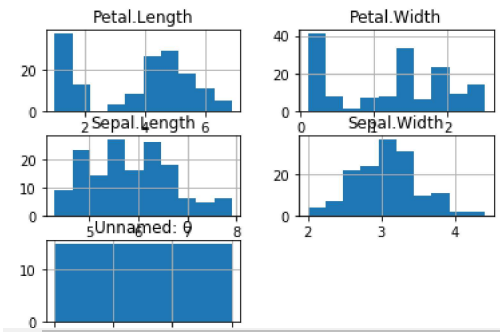
```
iris.tail(100)
```

	Unnamed: 0	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
50	51	7.0	3.2	4.7	1.4	versicolor
51	52	6.4	3.2	4.5	1.5	versicolor
52	53	6.9	3.1	4.9	1.5	versicolor
53	54	5.5	2.3	4.0	1.3	versicolor
54	55	6.5	2.8	4.6	1.5	versicolor
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	virginica
146	147	6.3	2.5	5.0	1.9	virginica
147	148	6.5	3.0	5.2	2.0	virginica
148	149	6.2	3.4	5.4	2.3	virginica
149	150	5.9	3.0	5.1	1.8	virginica

```
#Checking for outliars
import matplotlib.pyplot as plt
plt.figure(1)
plt.boxplot([iris['Sepal.Length']])
plt.figure(2)
plt.boxplot([iris['Sepal.Width']])
plt.show()
```

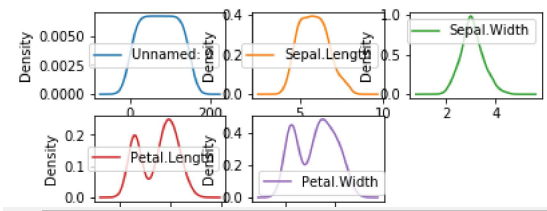


```
iris.hist()
plt.show()
```



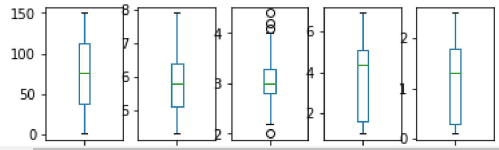

```
iris.plot(kind ='density',subplots = True, layout =(3,3),sharex = False)
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000002900CD2EA48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CDA7048>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CDD9E88>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000002900CE0F0C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CE43A88>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CE7D488>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000002900CEB6448>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CEF4E48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002900CEFC248>]],
      dtype=object)
```




```
iris.plot(kind ='box',subplots = True, layout =(2,5),sharex = False)
```

```
Unnamed: 0         AxesSubplot(0.125,0.536818;0.133621x0.343182)
Sepal.Length    AxesSubplot(0.285345,0.536818;0.133621x0.343182)
Sepal.Width      AxesSubplot(0.44569,0.536818;0.133621x0.343182)
Petal.Length    AxesSubplot(0.606034,0.536818;0.133621x0.343182)
Petal.Width     AxesSubplot(0.766379,0.536818;0.133621x0.343182)
dtype: object
```



```
X = iris['Sepal.Length'].values.reshape(-1,1)
print(X)
```

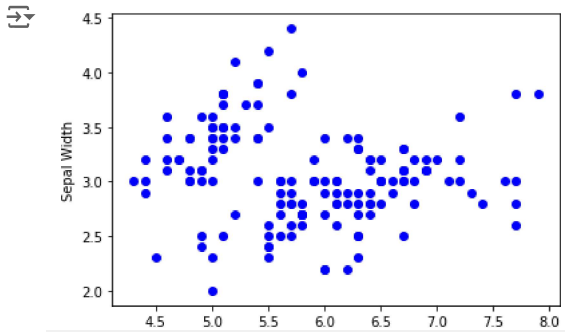
```
[[5.8]
 [5. ]
 [5.6]
 [5.7]
 [5.7]
 [6.2]
 [5.1]
 [5.7]
 [6.3]
 [5.8]
 [7.1]
 [6.3]
 [6.5]
 [7.6]
 [4.9]
 [7.3]
 [6.7]
 [7.2]
 [6.5]
 [6.4]
 [6.8]
 [5.7]
 [5.8]
 [6.4]
 [6.5]
 [7.7]
 [7.7]
 [6. ]
 [6.9]
 [5.6]
 [7.7]
 [6.3]
 [6.7]
 [7.2]
 [6.2]
 [6.1]
 [6.4]
 [7.2]
 [7.4]
 [7.9]
 [6.4]
 [6.3]
 [6.1]
 [7.7]
 [6.3]
 [6.4]
 [6. ]
 [6.9]
 [6.7]
 [6.9]
 [5.8]
 [6.8]
 [6.7]
 [6.7]
 [6.3]
 [6.5]
 [6.2]
 [5.9]]
```

```
Y = iris['Sepal.Width'].values.reshape(-1,1)
print(Y)
```

```
[[3.5]
 [3. ]
```

```
[3.2]
[3.1]
[3.6]
[3.9]
[3.4]
[3.4]
[2.9]
[3.1]
[3.7]
[3.4]
[3. ]
[3. ]
[4. ]
[4.4]
[3.9]
[3.5]
[3.8]
[3.8]
[3.4]
[3.7]
[3.6]
[3.3]
[3.4]
[3. ]
[3.4]
[3.5]
[3.4]
[3.2]
[3.1]
[3.4]
[4.1]
[4.2]
[3.1]
[3.2]
[3.5]
[3.6]
[3. ]
[3.4]
[3.5]
[2.3]
[3.2]
[3.5]
[3.8]
[3. ]
[3.8]
[3.2]
[3.7]
[3.3]
[3.2]
[3.2]
[3.1]
[2.3]
[2.8]
[2.8]
[3.3]
```

```
plt.xlabel("Sepal Length")
plt.ylabel("Sepal Width")
plt.scatter(X,Y,color='b')
plt.show()
```




```
#Correlation
corr_mat = iris.corr()
print(corr_mat)
```

```
              Unnamed: 0  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
Unnamed: 0      1.000000      0.716676    -0.402301      0.882637     0.900027
Sepal.Length    0.716676      1.000000    -0.117570      0.871754     0.817941
Sepal.Width    -0.402301     -0.117570     1.000000     -0.428440    -0.366126
Petal.Length    0.882637      0.871754    -0.428440      1.000000     0.962865
Petal.Width     0.900027      0.817941    -0.366126      0.962865     1.000000
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn import metrics
from sklearn.tree import DecisionTreeClassifier
```

```
train, test = train_test_split(iris, test_size = 0.25)
print(train.shape)
print(test.shape)
```

```
(112, 6)
(38, 6)
```

```
train_X = train[['Sepal.Length', 'Sepal.Width', 'Petal.Length',
                 'Petal.Width']]
train_y = train.Species

test_X = test[['Sepal.Length', 'Sepal.Width', 'Petal.Length',
                 'Petal.Width']]
test_y = test.Species
```

```
train_X.head()
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
99	5.7	2.8	4.1	1.3
22	4.6	3.6	1.0	0.2
86	6.7	3.1	4.7	1.5
50	7.0	3.2	4.7	1.4

```
test_y.head()
```

```
90     versicolor
92     versicolor
147     virginica
16         setosa
82     versicolor
Name: Species, dtype: object
```

```
test_y.head()
```

```
90     versicolor
92     versicolor
147     virginica
16         setosa
82     versicolor
Name: Species, dtype: object
```

```
model = LogisticRegression()
model.fit(train_X, train_y)
prediction = model.predict(test_X)
print('Accuracy:',metrics.accuracy_score(prediction,test_y))
```

```
Accuracy: 0.9210526315789473
```

```
#Confusion matrix
from sklearn.metrics import confusion_matrix
confusion_mat = confusion_matrix(test_y,prediction)
print("Confusion matrix: \n",confusion_mat)
```

```
Confusion matrix:
 [[11  0  0]
 [ 0 15  3]
 [ 0  0  9]]
```

Start coding or generate with AI.