```
# 8 Bayesian classification on any dataset.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
from sklearn import datasets
iris=datasets.load_iris()
```

```
x=iris.data[:,]
#four features value of iris flower
```

```
y=iris.target
y
#corrsponding three flower classes 0, 1,2
```

```
array([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, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

```
print("Features=",iris['feature_names'])
```

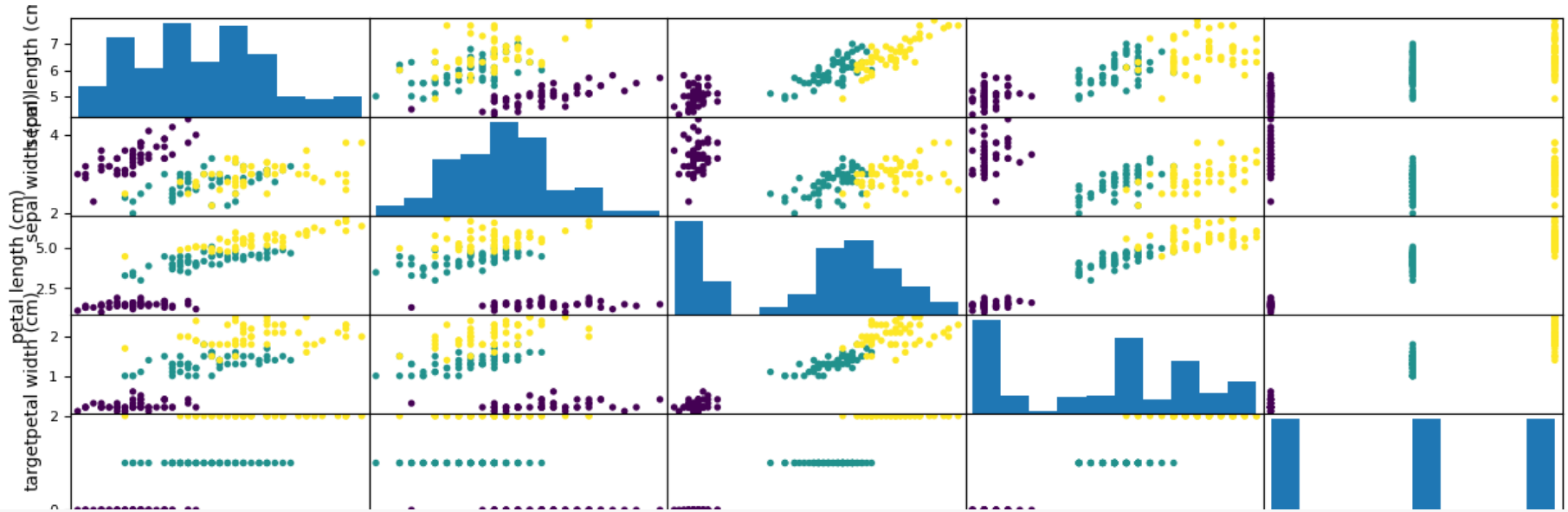
```
Features= ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

```
iris_dataframe = pd.DataFrame(data=np.c_[iris['data'],iris['target']], columns=iris['feature_names']+['target'])
```

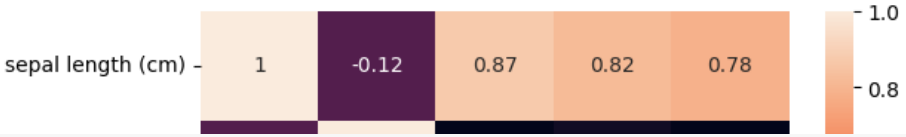
```
plt.figure()
```

```
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
```

```
import pandas as pd
import matplotlib.pyplot as plt
plt = pd.plotting.scatter_matrix(iris_dataframe, c=iris['target'],
                                 figsize=(15,5),
                                 s=60,alpha=1)
plt.show()
```




```
#step-3 visualising corelation &checking assumptions of Naive bayes
import matplotlib.pyplot as plt
import seaborn as sns
dataplot = sns.heatmap(iris_dataframe.corr(), annot=True)
plt.show()
```



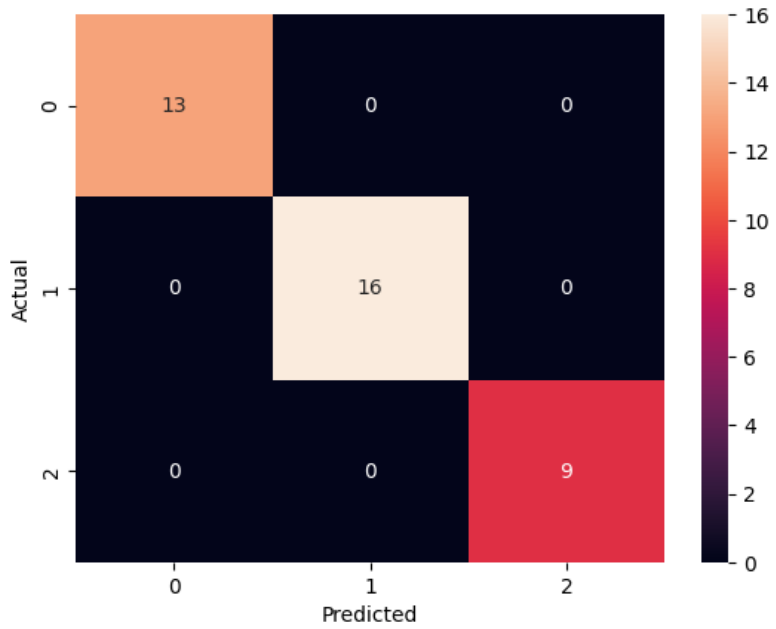

```
#step-4  split dataset

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x,y,test_size=0.25 ,random_state=0 )
```

```
#step-5  fit the model
from sklearn.naive_bayes import GaussianNB
NB = GaussianNB()
NB.fit(X_train, y_train)
```

GaussianNB

GaussianNB()

```
#step-6  Evaluate the model
import matplotlib.pyplot as plt
Y_pred = NB.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, Y_pred)
df_cm = pd.DataFrame(cm, columns=np.unique(y_test), index = np.unique(y_test))
df_cm.index.name = 'Actual'
df_cm.columns.name = 'Predicted'
sns.heatmap(df_cm, annot=True) #font size
plt.show()
```


13
0
0
0
16
0
0
0
9
Actual
Predicted
0
1
2
16
14
12
10
8
6
4
2
0