LETSGROWMORE

TASK 4 - Iris Flowers Classification ML Project

NITIN SINGH RATHORE

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

In [2]:
```
df = pd.read_csv("D:\\stock\\Iris.csv")
```

In [3]:
```
df_columns = ["sepal length", "sepal width", "petal length", "petal width", "Class"]
df.head()
```

Out[3]:

	sepal length	sepal width	petal length	petal width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [4]:
```
df.describe()
```

Out[4]:

	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

In [5]:
```
df.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   Class         150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

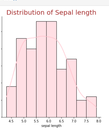
In [6]:
```
df.isnull().sum()
```

Out[6]:
```
sepal length    0
sepal width     0
petal length    0
petal width     0
Class           0
dtype: int64
```

In [7]:
```
df.value_counts()
```

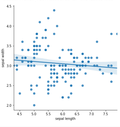
Out[7]:
```
sepal length  sepal width  petal length  petal width  Class
4.3           3.0          1.1           0.1          Iris-setosa       1
5.8           2.7          5.1           1.9          Iris-virginica    1
              4.0          1.2           0.2          Iris-setosa       1
5.9           3.0          4.2           1.5          Iris-versicolor   1
6.2           2.8          4.8           1.8          Iris-virginica    1
                                                                        ..
5.5           2.3          4.0           1.3          Iris-versicolor   1
              2.4          3.7           1.0          Iris-versicolor   1
              2.5          4.0           1.3          Iris-versicolor   1
7.0           3.2          4.7           1.4          Iris-versicolor   1
Length: 147, dtype: int64
```

In [8]:
```
sns.displot(df['sepal length'],kde=True, color='pink")
plt.title("Distribution of Sepal length", fontsize=20, color = 'brown')
plt.show()
```



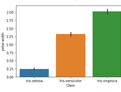
In [9]:
```
sns.lmplot(x='sepal length',y='sepal width',data=df)
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x1412b924e80>
```



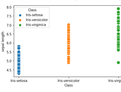
In [10]:
```
sns.barplot(x='Class',y='petal width',data=df)
```

Out[10]:
```
<AxesSubplot:xlabel='Class', ylabel='petal width'>
```



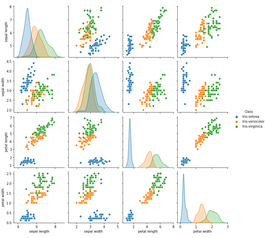
In [11]:
```
sns.scatterplot(x='Class', y='sepal length',s = 100, data=df , hue="Class")
```

Out[11]:
```
<AxesSubplot:xlabel='Class', ylabel='sepal length'>
```



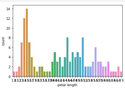
In [12]:
```
sns.pairplot(df, hue ='Class')
```

Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x1412c127b80>
```



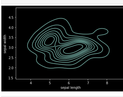
In [13]:
```
sns.countplot(x='petal length',data=df)
plt.style.use("dark_background")
```

Out[13]:



In [14]:
```
sns.kdeplot(x='sepal length',y='sepal width',data=df)
plt.style.use("grayscale")
```

Out[14]:



In [15]:
```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'Class'.
df['Class']= label_encoder.fit_transform(df['Class'])

df['Class'].unique()
```

Out[15]:
```
array([0, 1, 2])
```

In [16]:
```
x = df.iloc[:, 0:4]
x.head()
```

Out[16]:

	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 [17]:
```
y = df[['Class']]
y
```

Out[17]:

	Class
0	0
1	0
2	0
3	0
4	0
...	...
145	2
146	2
147	2
148	2
149	2

150 rows × 1 columns

In [18]:
```
from sklearn.model_selection import train_test_split
```

In [19]:
```
x_train,x_test,y_train,y_test=train_test_split (x, y, test_size = .20,random_state=0)
print("Size of x_train is",x_train.shape)
print("Size of y_train is",y_train.shape)
print("Size of x_test is",x_test.shape)
print("Size of y_test is",y_test.shape)
```

```
Size of x_train is (120, 4)
Size of y_train is (120, 1)
Size of x_test is (30, 4)
Size of y_test is (30, 1)
```

In [20]:
```
x_train
```

Out[20]:

	sepal length	sepal width	petal length	petal width
137	6.4	3.1	5.5	1.8
84	5.4	3.0	4.5	1.5
27	5.2	3.5	1.5	0.2
127	6.1	3.0	4.9	1.8
132	6.4	2.8	5.6	2.2
...	...	...	...	...
9	4.9	3.1	1.5	0.1
103	6.3	2.9	5.6	1.8
67	5.8	2.7	4.1	1.0
117	7.7	3.8	6.7	2.2
47	4.6	3.2	1.4	0.2

120 rows × 4 columns

In [21]:
```
from sklearn.linear_model import LogisticRegression
lgr = LogisticRegression(max_iter=1000)
lgr.fit (x_train, y_train.values.ravel())

prediction = lgr.predict (x_test)
print(prediction)
```

```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0]
```

In [22]:
```
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix (y_test, prediction)
confusionMatrix
```

Out[22]:
```
array([[11,  0,  0],
       [ 0, 13,  0],
       [ 0,  0,  6]], dtype=int64)
```

In [23]:
```
from sklearn.metrics import accuracy_score

print("Accuracy is",accuracy_score(y_test, prediction))
```

```
Accuracy is 1.0
```

In [24]:
```
from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(x_train,y_train.values.ravel())

y_pred = knn.predict(x_test)

from sklearn.metrics import accuracy_score
print("Accuracy is",accuracy_score(y_test, y_pred))
```

```
Accuracy is 0.9666666666666667
```

In [25]:
```
from sklearn.metrics import confusion_matrix
confusionMatrix = confusion_matrix (y_test, y_pred)
cm = confusion_matrix(y_test, y_pred, labels=["Iris-virginica"])
confusionMatrix_eval = pd.DataFrame(confusionMatrix, columns = values, index = values)
confusionMatrix_eval
```

Out[25]:

	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	11	0	0
Iris-versicolor	0	12	1
Iris-virginica	0	0	6

In [26]:
```
from sklearn import tree
D_tree = tree.DecisionTreeClassifier()
D_tree.fit(x_train,y_train)
```

Out[26]:
```
DecisionTreeClassifier()
```

In [27]:
```
pred_tree = D_tree.predict(x_test)
```

In [28]:
```
accuracy = accuracy_score(y_test, pred_tree)*100
```

In [29]:
```
print("Accuracy is", accuracy)
```

```
Accuracy is 100.0
```

In [ ]: