LETS GROW MORE(LGM) INTERNSHIP AUGUST-2021

IRIS FLOWERS CLASSIFICATION MACHINE LEARNING

Step1 Defining objectives

In [54]:
```
import numpy as np
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt
```

In [55]:
```
from sklearn.metrics import confusion_matrix
```

Step2 importing dataset

In [56]:
```
df = pd.read_csv('Iris.csv')
df.head()
```

Out[56]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Step 3 Data Preprocessing

In [40]:

```
# to basic info about datatype
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [41]:

```
# to display no. of samples on each class
df['Species'].value_counts()
```

Out[41]:

```
Iris-versicolor    50
Iris-virginica     50
Iris-setosa        50
Name: Species, dtype: int64
```

In [42]:

```
# check for null values
df.isnull().sum()
```

Out[42]:

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [43]:

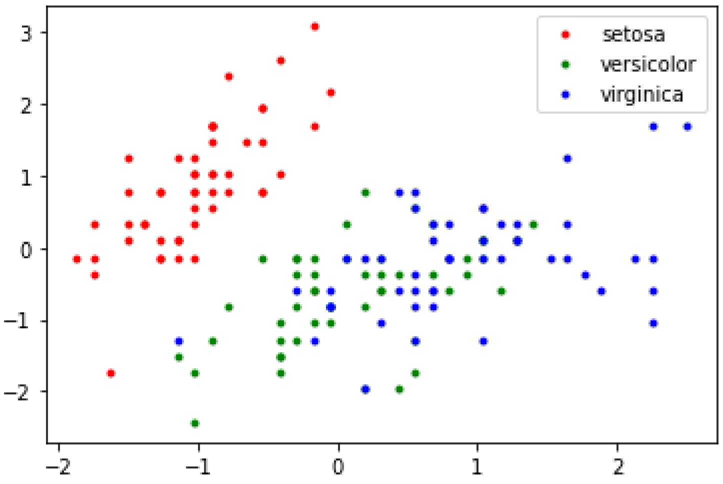
```
df.describe()
```

Out[43]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
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

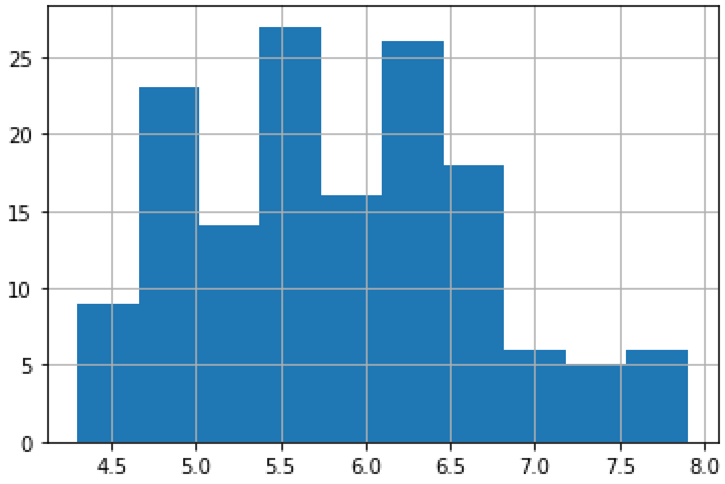
Step4 Data Visualization

In [44]:
```
plt.plot(x[:,0][y==0],x[:,1][y==0],'r.',label="setosa")
plt.plot(x[:,0][y==1],x[:,1][y==1],'g.',label="versicolor")
plt.plot(x[:,0][y==2],x[:,1][y==2],'b.',label="virginica")
plt.legend()
plt.show()
```



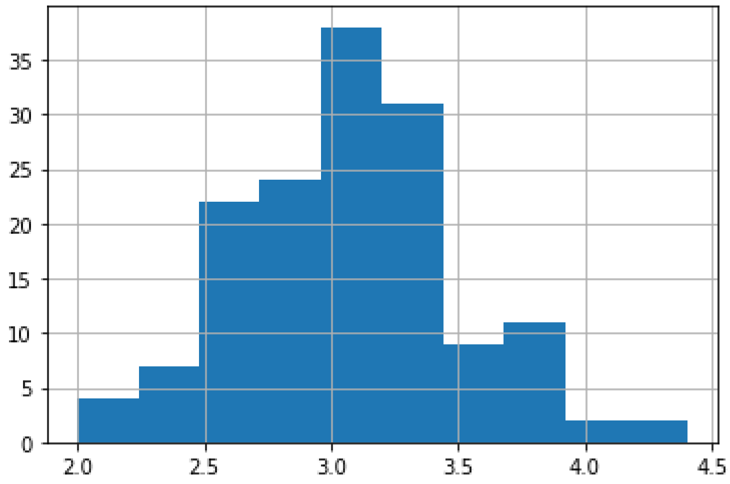
In [45]:
```
df['SepalLengthCm'].hist()
```

Out[45]:
```
<AxesSubplot:>
```



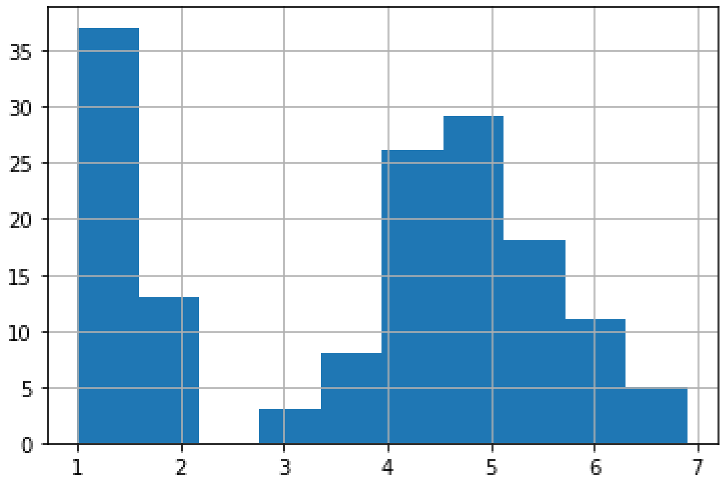
In [46]:
```
df['SepalWidthCm'].hist()
```

Out[46]:
```
<AxesSubplot:>
```



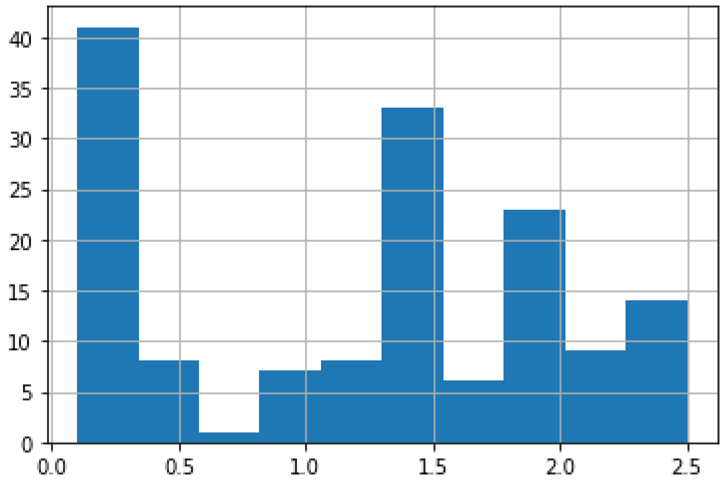
In [47]:
```
df['PetalLengthCm'].hist()
```

Out[47]:
```
<AxesSubplot:>
```



In [48]:
```
df['PetalWidthCm'].hist()
```

Out[48]:
```
<AxesSubplot:>
```



Step 5 Finding Co-relation

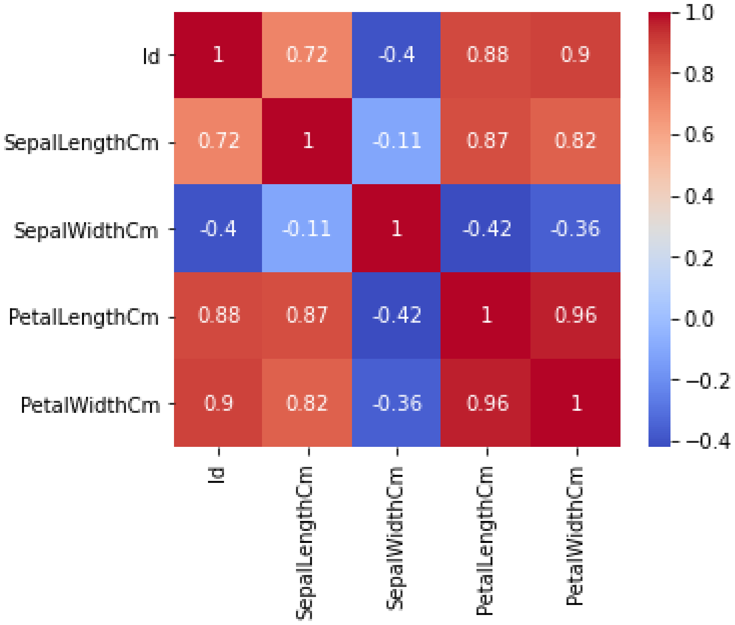
In [49]:
```
df.corr()
```

Out[49]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Id	1.000000	0.716676	-0.397729	0.882747	0.899759
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000

In [50]:
```
corr = df.corr()
fig, ax = plt.subplots(figsize=(5,4))
sns.heatmap(corr, annot=True, ax=ax, cmap = 'coolwarm')
```

Out[50]:
```
<AxesSubplot:>
```



Step 6 Model Building

In [51]:
```
from sklearn.preprocessing import StandardScaler
x=StandardScaler().fit_transform(x)
```

In [52]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state=1)
```

In [53]:
```
from sklearn.linear_model import LogisticRegression
```

```
In [29]: log_reg = LogisticRegression()
         log_reg.fit(x_train,y_train)

Out[29]: LogisticRegression()
```

```
In [30]: y_pred=log_reg.predict(x_test)
         y_pred

Out[30]: array([0, 1, 1, 0, 2, 1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 0, 1, 1, 0, 0, 1, 1,
                2, 0, 2, 1, 0, 0, 1, 2, 1, 2, 1, 2, 2, 0, 1, 0])
```

```
In [31]: log_reg.score(x_test,y_test)

Out[31]: 0.9736842105263158
```

```
In [32]: log_reg.score(x,y)

Out[32]: 0.9733333333333334
```

Confusion Matrix

```
In [33]: confusion_matrix(y_test,y_pred)

Out[33]: array([[13,  0,  0],
                [ 0, 15,  1],
                [ 0,  0,  9]], dtype=int64)
```

Thank You

```
In [ ]:
```