Name: Priyanka Koradkar

LGMVIP-TASK-1-DATASET-Iris

In [47]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
from sklearn.model_selection import train_test_split
```

Loading the data

In [48]:
```
df=pd.read_csv('iris.csv')
```

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

Out[49]:

	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

In [50]:
```
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 [51]:
```
df['Species'].value_counts()
```

Out[51]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [52]:
```
df.isnull().sum()
```

Out[52]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [53]:
```
df.describe()
```

Out[53]:

	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

In [54]:
```
df
```

Out[54]:

	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
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

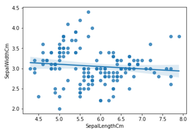
In [55]:
```
df.count()
```

Out[55]:
```
Id               150
SepalLengthCm    150
SepalWidthCm     150
PetalLengthCm    150
PetalWidthCm     150
Species          150
dtype: int64
```

Visualising the data

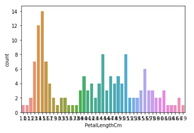
In [56]:
```
sns.regplot(x='SepalLengthCm',y='SepalWidthCm',data=df)
```

Out[56]:
```
<AxesSubplot:xlabel='SepalLengthCm', ylabel='SepalWidthCm'>
```



In [57]:
```
sns.countplot(x='PetalLengthCm',data=df)
```

Out[57]:
```
<AxesSubplot:xlabel='PetalLengthCm', ylabel='count'>
```



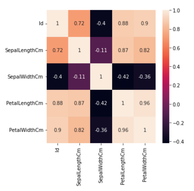
In [58]:
```
df.corr()
```

Out[58]:

	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 [59]:
```
corr=df.corr()
fig,ax=plt.subplots(figsize=(5,5))
sns.heatmap(corr, annot=True, ax=ax)
```

Out[59]:
```
<AxesSubplot:>
```



In [60]:
```
lbec=LabelEncoder()
```

In [61]:
```
df['Species']=lbec.fit_transform(df['Species'])
```

In [62]:
```
df.head()
```

Out[62]:

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

Splitting the data into testing and training datasets

In [63]:
```
X=df.drop(columns=['Species'])
Y=df['Species']
x_train,x_test,y_train,y_test=train_test_split(X,Y)
```

In [64]:
```
# Linear Regression
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

In [65]:
```
model.fit(x_train,y_train)
```

Out[65]:
```
LinearRegression()
```

Checking the accuracy using KNN

In [66]:
```
print(model.score(x_test,y_test))
```

```
0.9564812524400 6
```

In [67]:
```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
```

In [68]:
```
model.fit(x_train,y_train)
```

Out[68]:
```
DecisionTreeClassifier()
```

Checking the accuracy using Decision tree

In [46]:
```
print(model.score(x_test,y_test))
```

```
1.0
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: