In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.linear_model import LinearRegression
import statsmodels.formula.api as smf
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
```

In [2]:
```
iris=pd.read_csv("C:\\Users\\Pranav\\Desktop\\DATA SCIENCE DATA\\CVC file\\Iris.csv")
iris.head()
```

Out[2]:

	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 [3]:
```
iris.tail()
```

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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

In [4]:
```
#Shape of dataset
iris.shape
```

Out[4]:
```
(150, 6)
```

In [5]:
```
#information about datast
iris.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 [6]:
```
# mathamathical information about data
iris.describe()
```

Out[6]:

	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 [7]:
```
# Find out null value in dataset
iris.isnull().sum()
```

Out[7]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [8]:
```
iris['Species'].value_counts()
```

Out[8]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [9]:
```
# find the value in iris dataset the sepalwidth is greater than 1
iris[iris['SepalWidthCm']>4]
```

Out[9]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
15	16	5.7	4.4	1.5	0.4	Iris-setosa
32	33	5.2	4.1	1.5	0.1	Iris-setosa
33	34	5.5	4.2	1.4	0.2	Iris-setosa

In [10]:
```
# find the value in iris dataset the petalwidth is greater than 1
iris[iris['PetalWidthCm']>1]
```

Out[10]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
50	51	7.0	3.2	4.7	1.4	Iris-versicolor
51	52	6.4	3.2	4.5	1.5	Iris-versicolor
52	53	6.9	3.1	4.9	1.5	Iris-versicolor
53	54	5.5	2.3	4.0	1.3	Iris-versicolor
54	55	6.5	2.8	4.6	1.5	Iris-versicolor
...	...	...	...	...	...	...
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

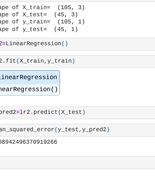
93 rows × 6 columns

In [11]:
```
# find the value in iris dataset the petalwidth is greater than 2
iris[iris['PetalWidthCm']>2]
```

Out[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
100	101	6.3	3.3	6.0	2.5	Iris-virginica
102	103	7.1	3.0	5.9	2.1	Iris-virginica
104	105	6.5	3.0	5.8	2.2	Iris-virginica
105	106	7.6	3.0	6.6	2.1	Iris-virginica
109	110	7.2	3.6	6.1	2.5	Iris-virginica
112	113	6.8	3.0	5.5	2.1	Iris-virginica
114	115	5.8	2.8	5.1	2.4	Iris-virginica
115	116	6.4	3.2	5.3	2.3	Iris-virginica
117	118	7.7	3.8	6.7	2.2	Iris-virginica
118	119	7.7	2.6	6.9	2.3	Iris-virginica
120	121	6.9	3.2	5.7	2.3	Iris-virginica
124	125	6.7	3.3	5.7	2.1	Iris-virginica
128	129	6.4	2.8	5.6	2.1	Iris-virginica
132	133	6.4	2.8	5.6	2.2	Iris-virginica
135	136	7.7	3.0	6.1	2.3	Iris-virginica
136	137	6.3	3.4	5.6	2.4	Iris-virginica
139	140	6.9	3.1	5.4	2.1	Iris-virginica
140	141	6.7	3.1	5.6	2.4	Iris-virginica
141	142	6.9	3.1	5.1	2.3	Iris-virginica
143	144	6.8	3.2	5.9	2.3	Iris-virginica
144	145	6.7	3.3	5.7	2.5	Iris-virginica
145	146	6.7	3.0	5.2	2.3	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica

In [12]:
```
# Visualization
sns.scatterplot(x='SepalLengthCm',y='PetalLengthCm',data=iris,hue='Species')
plt.grid(True)
plt.show()
```



In [13]:
```
X=iris[['SepalWidthCm']]
y=iris[['SepalLengthCm']]
```

In [14]:
```
print(X)
```

```
     SepalWidthCm
0             3.5
1             3.0
2             3.2
3             3.1
4             3.6
..            ...
145           3.0
146           2.5
147           3.0
148           3.4
149           3.0

[150 rows x 1 columns]
```

In [15]:
```
print(y)
```

```
     SepalLengthCm
0              5.1
1              4.9
2              4.7
3              4.6
4              5.0
..             ...
145            6.7
146            6.3
147            6.5
148            6.2
149            5.9

[150 rows x 1 columns]
```

In [16]:
```
#training and test data
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=2)
```

In [17]:
```
print("shape of X_train= ",X_train.shape)
print("shape of X_test= ",X_test.shape)
print("shape of y_train= ",y_train.shape)
print("shape of y_test= ",y_test.shape)
```

```
shape of X_train=  (105, 1)
shape of X_test=  (45, 1)
shape of y_train=  (105, 1)
shape of y_test=  (45, 1)
```

In [18]:
```
X_test.head()
```

Out[18]:

	SepalWidthCm
6	3.4
3	3.1
113	2.5
12	3.0
24	3.4

In [19]:
```
y_test.head()
```

Out[19]:

	SepalLengthCm
6	4.6
3	4.6
113	5.7
12	4.8
24	4.8

In [20]:
```
# creating the model
lr=LinearRegression()
```

In [21]:
```
# fitthe model
lr.fit(X_train,y_train)
```

Out[21]:
```
LinearRegression()
```

In [22]:
```
y_pred=lr.predict(X_test)
```

In [23]:
```
y_test.head()
```

Out[23]:

	SepalLengthCm
6	4.6
3	4.6
113	5.7
12	4.8
24	4.8

In [24]:
```
y_pred[0:5]
```

Out[24]:
```
array([[5.79624971],
       [5.87619788],
       [6.03609422],
       [5.99624727],
       [5.79624971]])
```

In [25]:
```
# Error in prediction findout
from sklearn.metrics import mean_squared_error
```

In [26]:
```
mean_squared_error(y_test,y_pred)
```

Out[26]:
```
0.7284298865085145
```

In [27]:
```
# Creatinf the model 2
y=iris[['SepalLengthCm']]
```

In [28]:
```
X=iris[['SepalWidthCm','PetalWidthCm','PetalLengthCm']]
```

In [29]:
```
#training and test data
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=2)
```

In [30]:
```
print("shape of X_train= ",X_train.shape)
print("shape of X_test= ",X_test.shape)
print("shape of y_train= ",y_train.shape)
print("shape of y_test= ",y_test.shape)

shape of X_train=  (105, 3)
shape of X_test=  (45, 3)
shape of y_train=  (105, 1)
shape of y_test=  (45, 1)
```

In [31]:
```
lr2=LinearRegression()
```

In [32]:
```
lr2.fit(X_train,y_train)
```

Out[32]:
```
LinearRegression()
```

In [35]:
```
y_pred2=lr2.predict(X_test)
```

In [36]:
```
mean_squared_error(y_test,y_pred2)
```

Out[36]:
```
0.08042496370919266
```

In [ ]:
```

```