In [13]:

```
import pandas as pd
```

In [14]:

```
from pandas import DataFrame
```

In [15]:

```
import numpy as np
```

In [16]:

```
import seaborn as sns
```

In [17]:

```
import matplotlib.pyplot as plt

%matplotlib inline
```

In [18]:

```
IrisDF =pd.read_csv("iris-data.csv")
```

In [19]:

```
IrisDF.head()
```

Out[19]:

	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 [20]:

```
IrisDF.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 [21]:

```
IrisDF
```

Out[21]:

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

150 rows × 5 columns

In [22]:

```
IrisDF.describe()
```

Out[22]:

	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 [23]:

```
IrisDF.columns
```

Out[23]:

```
Index(['sepal-length', 'sepal-width', 'petal-length', ' petal-width', 'cla
ss'], dtype='object')
```

In [27]:

```
x=IrisDF.drop(columns=["class"],axis=1)
x
```

Out[27]:

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

150 rows × 4 columns

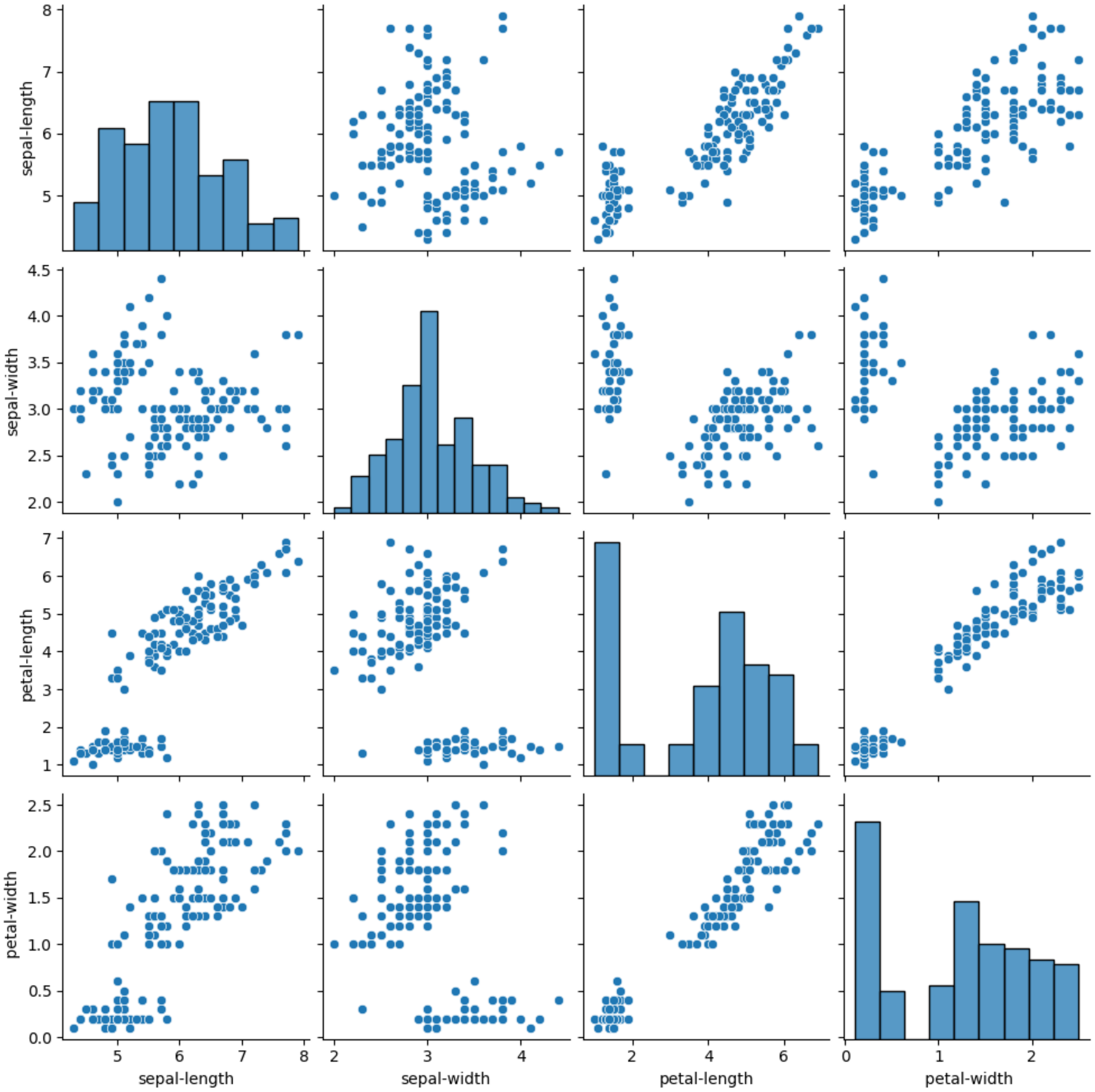
In [28]:

```
sns.pairplot(IrisDF)
```

```
C:\Users\DIVYA\AppData\Local\Programs\Python\Python39\lib\site-packages\se
aborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[28]:

```
<seaborn.axisgrid.PairGrid at 0x1e5ec307790>
```

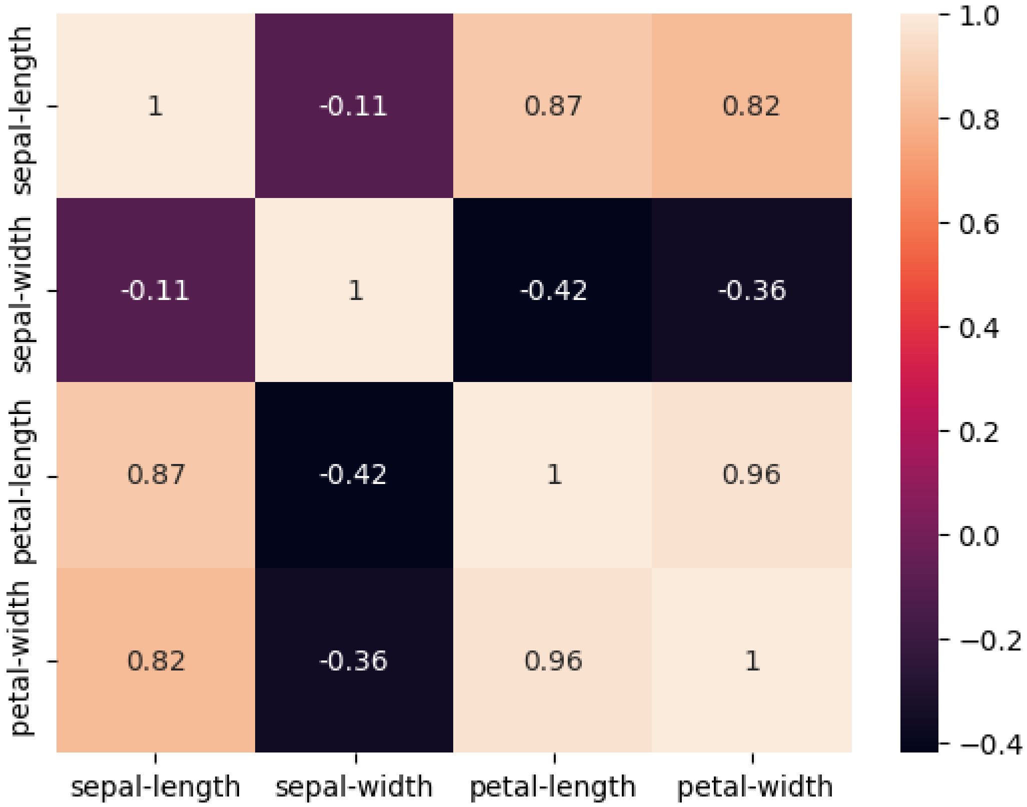


In [29]:

```
sns.heatmap(x.corr(),annot=True)
```

Out[29]:

```
<Axes: >
```



In [55]:

```
X=IrisDF[['sepal-length', 'sepal-width', ' petal-width']]

y=IrisDF['petal-length']
```

In [56]:

```
 from sklearn.model_selection import train_test_split
```

In [57]:

```
X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.40, random_state=101)
```

In [58]:

```
from sklearn.linear_model import LinearRegression
```

In [59]:

```
lm = LinearRegression()
```

In [60]:

```
lm.fit (X_train, y_train)
```

Out[60]:

```
▾ LinearRegression
LinearRegression()
```

In [61]:

```
coeff_df = pd.DataFrame (lm.coef_, X.columns, columns=['Coefficient'])
```

In [62]:

```
coeff_df
```

Out[62]:

	Coefficient
sepal-length	0.712393
sepal-width	-0.654431
petal-width	1.461103

In [63]:

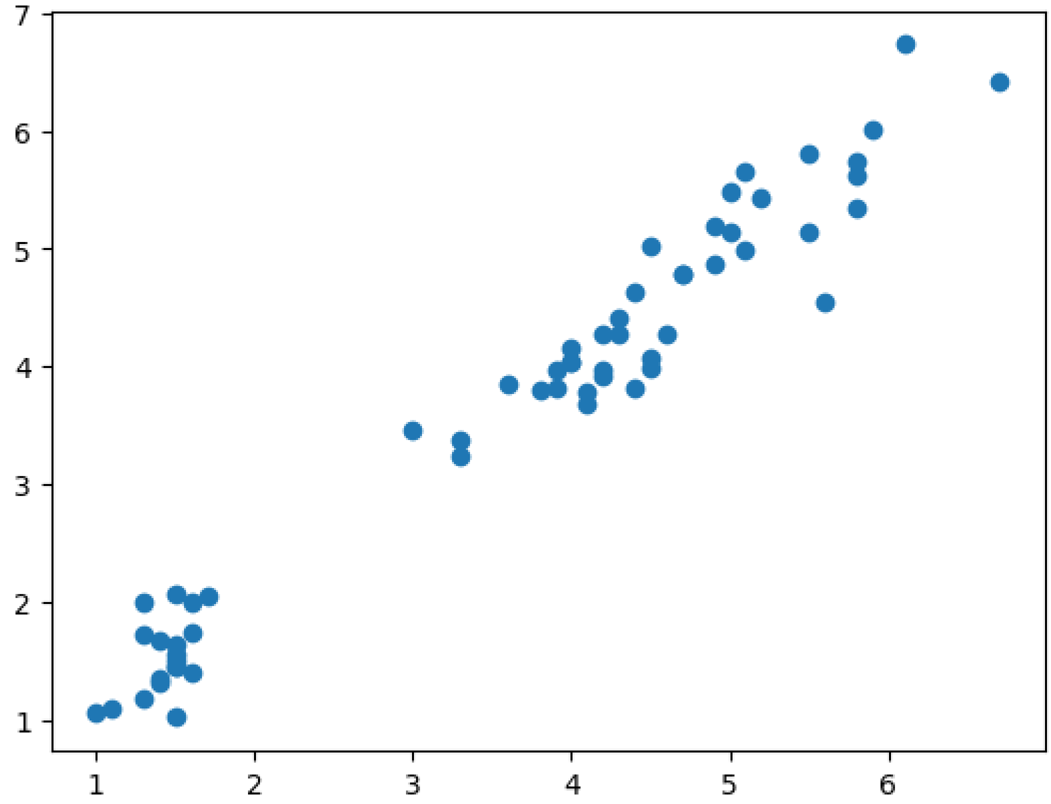
```
predictions = lm.predict (X_test)
```

In [64]:

```
plt.scatter (y_test, predictions)
```

Out[64]:

```
<matplotlib.collections.PathCollection at 0x1e5f0aab9a0>
```



In [65]:

```
sns.distplot((y_test-predictions),bins=50);
```

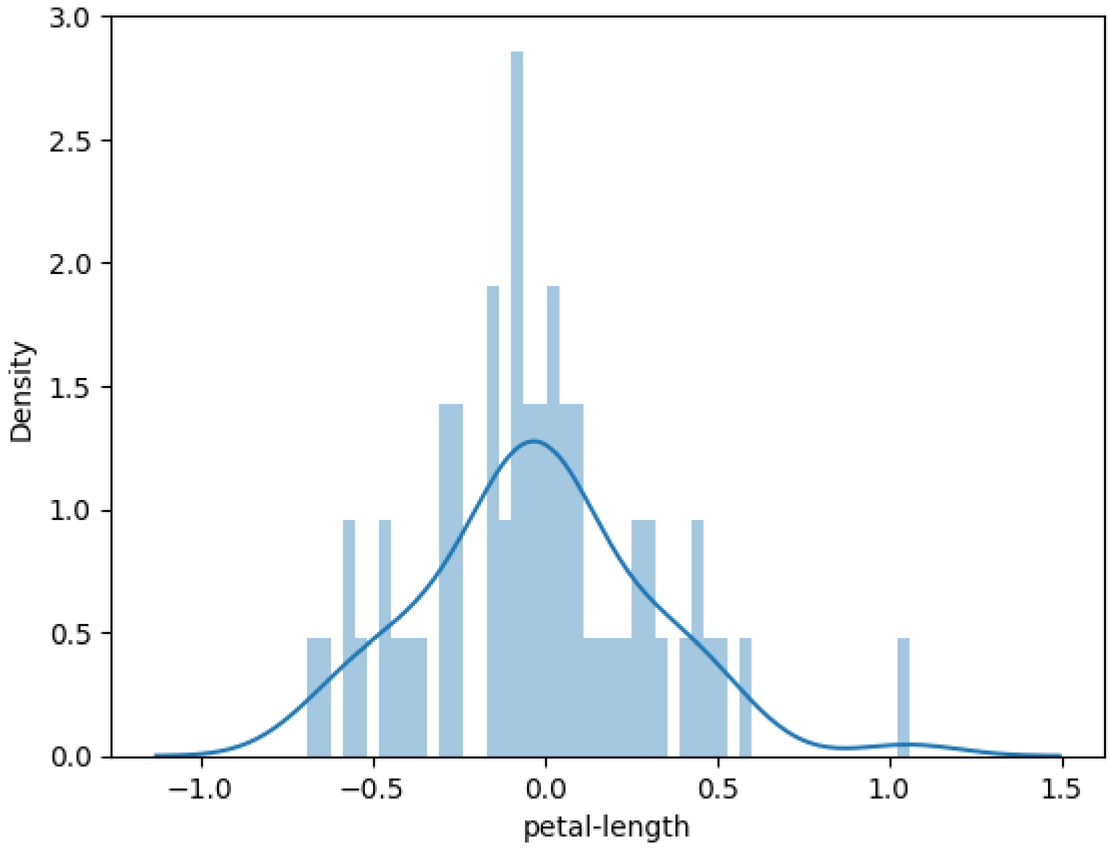
```
C:\Users\DIVYA\AppData\Local\Temp\ipykernel_8336\1326397652.py:1: UserWarn
ing:

`distplot` is a deprecated function and will be removed in seaborn v0.14.
0.

Please adapt your code to use either `displot` (a figure-level function wi
th
similar flexibility) or `histplot` (an axes-level function for histogram
s).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://
gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot((y_test-predictions),bins=50);
```



In [ ]: