jupyter hard-challenge Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

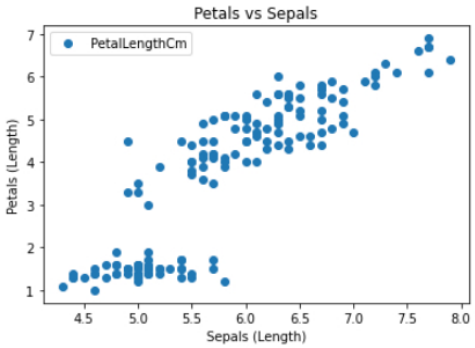
In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

In [2]:
```
dataset = pd.read_csv('sample-iris.csv')
dataset.shape
dataset.head()
dataset.describe()
```

Out[2]:

	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 [9]:
```
dataset.plot(x='SepalLengthCm', y='PetalLengthCm', style='o')
plt.title('Petals vs Sepals')
plt.xlabel('Sepals (Length)')
plt.ylabel('Petals (Length)')
plt.show()
```



In [14]:
```
trainX, testX, trainY, testY = train_test_split(X, Y, test_size=0.3, random_state=0)
```

In [15]:
```
regressor = LinearRegression()
regressor.fit(trainX, trainY)
```

Out[15]:
```
LinearRegression()
```

In [17]:
```
print(regressor.intercept_)
```

```
-1.0036416142611415e-13
```

In [20]:
```
print(regressor.coef_)
```

```
[ 1.26492984e-15  1.00000000e+00 -6.93889390e-17  4.44089210e-16
  3.33066907e-16]
```

In [21]:
```
predY = regressor.predict(testX)
df = pd.DataFrame({'Actual': testY, 'Predicted': predY})
df
```

Out[21]:

	Actual	Predicted
0	5.8	5.8

1	6.0	6.0
2	5.5	5.5
3	7.3	7.3
4	5.0	5.0
5	6.3	6.3
6	5.0	5.0
7	6.7	6.7
8	6.8	6.8
9	6.1	6.1
10	6.1	6.1
11	6.4	6.4
12	6.1	6.1
13	6.5	6.5
14	6.1	6.1
15	4.9	4.9
16	6.0	6.0
17	5.5	5.5
18	4.8	4.8
19	5.4	5.4
20	5.6	5.6
21	5.6	5.6
22	4.8	4.8
23	4.4	4.4
24	6.2	6.2
25	4.6	4.6
26	5.1	5.1
27	6.2	6.2
28	5.0	5.0
29	5.0	5.0
30	6.4	6.4
31	5.4	5.4
32	5.2	5.2
33	6.1	6.1
34	6.4	6.4
35	5.2	5.2
36	5.7	5.7
37	6.0	6.0
38	5.9	5.9
39	5.8	5.8
40	6.8	6.8
41	4.7	4.7
42	6.9	6.9
43	5.0	5.0
44	5.4	5.4

In [22]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(testY, predY))
print('Mean Squared Error:', metrics.mean_squared_error(testY, predY))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(testY, predY)))
```

```
Mean Absolute Error: 4.407338691534177e-14
Mean Squared Error: 2.695502652655521e-27
Root Mean Squared Error: 5.191823044611133e-14
```

In [ ]: