servo Prediction using Linear Regression

In control engineering a servomechanism, usally shortend to servo, is an automatic device that uses error- sensing negative feedback to correct the action of a mechanism.

from previous projects data

In [1]:
```
import pandas as pd
import numpy as np
```

In [131]:
```
df= pd.read_csv("/home/student/Downloads/finance.csv")
```

In [132]:
```
df.head()
```

Out[132]:

	motor	screw	Pgain	Vgain	class
0	1	2	5	4	4
1	3	5	6	5	11
2	2	3	4	3	6
3	2	4	3	2	48
4	5	3	6	5	6

In [133]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   motor   5 non-null      int64
 1   screw   5 non-null      int64
 2   Pgain   5 non-null      int64
 3   Vgain   5 non-null      int64
 4   class   5 non-null      int64
dtypes: int64(5)
memory usage: 328.0 bytes
```

In [134]:
```
df.describe()
```

Out[134]:

	motor	screw	Pgain	Vgain	class
count	5.000000	5.000000	5.00000	5.00000	5.000000
mean	2.600000	3.400000	4.80000	3.80000	15.000000
std	1.516575	1.140175	1.30384	1.30384	18.627936
min	1.000000	2.000000	3.00000	2.00000	4.000000
25%	2.000000	3.000000	4.00000	3.00000	6.000000
50%	2.000000	3.000000	5.00000	4.00000	6.000000
75%	3.000000	4.000000	6.00000	5.00000	11.000000

motor screw Pgain Vgain class

In [135]:
```
df[['motor']].value_counts()
```

Out[135]:
```
motor
2        2
1        1
3        1
5        1
dtype: int64
```

In [136]:
```
df[['screw']].value_counts()
```

Out[136]:
```
screw
3        2
2        1
4        1
5        1
dtype: int64
```

In [137]:
```
y =df['class']
```

In [139]:
```
y.shape
```

Out[139]:
```
(5,)
```

In [138]:
```
y
```

Out[138]:
```
0     4
1    11
2     6
3    48
4     6
Name: class, dtype: int64
```

In [140]:
```
X=df[['Pgain','Vgain']]
```

In [141]:
```
X= df.drop('class',axis=1)
```

In [142]:
```
X.shape
```

Out[142]:
```
(5, 4)
```

In [143]:
```
X
```

Out[143]:

	motor	screw	Pgain	Vgain
0	1	2	5	4
1	3	5	6	5
2	2	3	4	3
3	2	4	3	2
4	5	3	6	5

In [110]:
```
from sklearn.model_selection import train_test_split
```

In [111]:
```
x_train,x_test,y_train,y_test =train_test_split(x,y,test_size =0.3,r
```

In [112]:
```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[112]: ((3, 2), (2, 2), (3,), (2,))

In [115]:
```
from sklearn.linear_model import LinearRegression
```

In [114]:
```
lr =LinearRegression()
```

In [151]:
```
lr.fit(x_train,y_train)
```

Out[151]: LinearRegression()

In [118]:
```
y_pred =lr.predict(x_test)
```

In [79]:
```
y_pred.shape
```

Out[79]: (2,)

In [119]:
```
y_pred
```

Out[119]: array([5.33333333, 5.33333333])

In [81]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r
```

In [120]:
```
mean_squared_error(y_test,y_pred)
```

Out[120]: 926.2777777777777

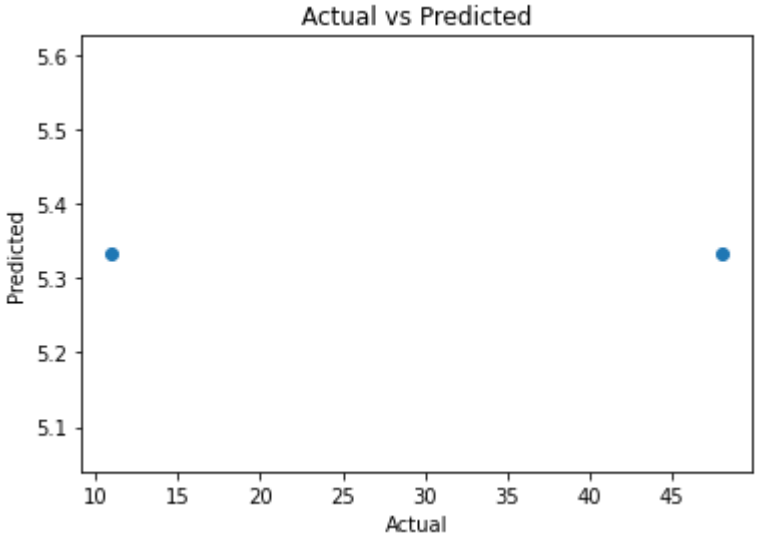
In [121]:
```
mean_absolute_error(y_test,y_pred)
```

Out[121]: 24.166666666666664

In [122]:
```
r2_score(y_test,y_pred)
```

Out[122]: -1.7064361658956253

In [144]:
```
import matplotlib.pyplot as plt
plt.scatter(y_test,y_pred)
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.title("Actual vs Predicted")
plt.show()
```



In [145]:
```
x_new = df.sample(1)
```

In [146]:
```
x_new
```

Out[146]:

	motor	screw	Pgain	Vgain	class
3	2	4	3	2	48

In [147]:
```
x_new.shape
```

Out[147]: (1, 5)

In [ ]:
```
y_pred_new =lr.predict(x_new)
```

In [ ]:
```
y_pred_new
```

In [ ]:
```
A data frame with some observations on variables,4 nominals and 1 as
of a servo system involving a servo amplifier,a motor, a lead screw/

1. Motor A,B,C,D,E
2. Screw A,B,C,D,E
3. Pgain 3,4,5,6
4. Vgain 1,2,3,4,5
5. Class 0.13 to 7.10
```

In [ ]:
```

```