Get understanding about Data Set

Data Set Characteristics:

Multivariate

Number of Instances:

167

Area:

Computer

Attribute Characteristics:

Categorical, Integer

Number of Attributes:

4

Date Donated

1993-05-01

Associated Tasks:

Regression

Missing Values?

No

Number of Web Hits:

123354

Source:

Creator:

Karl Ulrich (MIT)

Donor:

Ross Quinlan

Data Set Information:

Ross Quinlan:

This data was given to me by Karl Ulrich at MIT in 1986. I didn't record his description at the time, but here's his subsequent (1992) recollection:

"I seem to remember that the data was from a simulation of a servo system involving a servo amplifier, a motor, a lead screw/nut, and a sliding carriage of some sort. It may have been on of the translational axes of a robot on the 9th floor of the AI lab. In any case, the output value is almost certainly a rise time, or the time required for the system to respond to a step change in a position set point."

(Quinlan, ML'93)

"This is an interesting collection of data provided by Karl Ulrich. It covers an extremely non-linear phenomenon - predicting the rise time of a servomechanism in terms of two (continuous) gain settings and two (discrete) choices of mechanical linkages."

Attribute Information:

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

Import Library

```
import pandas as pd

import numpy as np
```

Import CSV as DataFrame

```
df = pd.read_csv(r'https://github.com/YBI-Foundation/Dataset/raw/main/Servo%20Mechan
```

GET THE fIRST FIVE ROWS OF THE DATA FRAME

```
df.head()
```

	Motor	Screw	Pgain	Vgain	Class
0	E	E	5	4	4
1	B	D	6	5	11
2	D	D	4	3	6
3	B	A	3	2	48
4	D	B	6	5	6

Get information about the data frame

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Motor   167 non-null    object
 1   Screw   167 non-null    object
 2   Pgain   167 non-null    int64 
 3   Vgain   167 non-null    int64 
 4   Class   167 non-null    int64 
dtypes: int64(3), object(2)
memory usage: 6.6+ KB
```

Get the Summary statistics

```
df.describe()
```

	Pgain	Vgain	Class
count	167.000000	167.000000	167.000000
mean	4.155689	2.538922	21.173653
std	1.017770	1.369850	13.908038
min	3.000000	1.000000	1.000000
25%	3.000000	1.000000	10.500000
50%	4.000000	2.000000	18.000000
75%	5.000000	4.000000	33.500000
max	6.000000	5.000000	51.000000

Get column names

```
df.columns
```

```
Index(['Motor', 'Screw', 'Pgain', 'Vgain', 'Class'], dtype='object')
```

Get the shape of the DataFrame

```
df.shape
```

```
(167, 5)
```

Get catagories and counts of catagorical Variables

```
df[['Motor']].value_counts()
```

```
Motor
C        40
A        36
B        36
E        33
D        22
dtype: int64
```

```
df[['Screw']].value_counts()
```

```
Screw
A        42
B        35
C        31
D        30
E        29
dtype: int64
```

Get Encoding of Catagorical Features

```
df.replace({'Motor':{'A':0,'B':1,'C':2,'D':3,'E':4}},inplace = True)
```

```
df.replace({'Screw':{'A':0,'B':1,'C':2,'D':3,'E':4}},inplace = True)
```

Define y (dependant or label or target variable) and X (independent ,feature and attribute Variable)

```
y = df['Class']
```

```
y.shape
```

```
(167,)
```

```
X = df.drop('Class',axis = 1)
```

```
X.shape
```

```
(167, 4)
```

Get train_test_split

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

```
X_train,X_test,y_train,y_test = train_test_split(X,y ,test_size=0.3,random_state=252
```

```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

```
((116, 4), (51, 4), (116,), (51,))
```

Get Model Train

```
#from sklearn.linear_model import LinearRegression
#from sklearn.neighbors import KNeighborsRegressor
#from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
```

```
model =  DecisionTreeRegressor()
```

Get Model train

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

```
DecisionTreeRegressor()
```

Get model prediction

```
y_pred = model.predict(X_test)
```

```
y_pred.shape
```

```
(51,)
```

Gel Model Evaluation

```
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error,mean_
```

```
mean_squared_error(y_test,y_pred)
```

```
28.058823529411764
```

```
mean_absolute_error(y_test,y_pred)
```

```
3.823529411764706
```

```
r2_score(y_test,y_pred)
```

```
0.8643390102720245
```

mean_square_error for

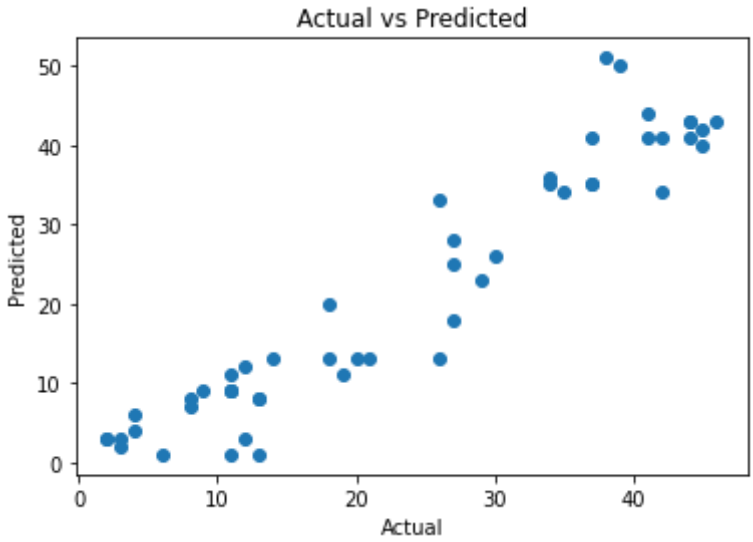
1. LinearRegression = 66
2. KNeighborsRegressor = 85
3. SVR = 181
4. DecisionTreeRegressor = 28

Double-click (or enter) to edit

Get visualisation of the actual vs predicted Results

```
import matplotlib.pyplot as plt
plt.scatter(y_test,y_pred)
```

```
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs Predicted')
plt.show()
```



Get Future Prediction

Lets select a random sample from existing dataset as new value

1. Extract a randomrow using sample function
2. Separate X and y
3. Standardize X
4. Predict

```
df_new = df.sample(1)
```

```
df_new
```

	Motor	Screw	Pgain	Vgain	Class
142	0	4	3	1	41

```
df_new.shape
```

```
(1, 5)
```

```
X_new = df_new.drop('Class',axis = 1)
```

```
X_new
```

	Motor	Screw	Pgain	Vgain
142	0	4	3	1

```
X_new.shape
```

```
(1, 4)
```

```
y_pred_new = model.predict(X_new)
```

```
y_pred_new
```

```
array([41.])
```

**CONCLUSION **

- I tried all the regression model for the servo prediction among them **DecisionTreeRegressor** gives the higher accurecy of the Servo prediction.