** Calcuate MSE **

In [ ]:
```
import numpy as np
# MSE for Training set Model 1
Y_true = [1.8,2.4,2.23,3.8,5.3]  # Y_true = Y (original values)

# Calculated values
Y_pred = [1.38,2.24,3.35,4.21,5.078] # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

0.3347568

In [ ]:
```
#  MSE for Training set Model 2
Y_true = [1.8,2.4,2.23,3.8,5.3] # Y_true = Y (original values)

# Calculated values
Y_pred = [1.75,2.15,3.08,4.10,5.39]

  # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

0.17712

In [ ]:
```
#  MSE for validation set Model 1
Y_true = [1.7,2.7,2.5,2.8,5.5]
  # Y_true = Y (original values)

# Calculated values
Y_pred = [1.81,3.014,3.702,4.562,4.902]
 # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

1.0035496000000002

In [ ]:
```
# MSE for validation set Model 2
Y_true = [1.7,2.7,2.5,2.8,5.5]  # Y_true = Y (original values)

# Calculated values
Y_pred = [1.91,2.74,3.45,4.59,5.11]
  # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

0.8608800000000001

In [ ]:
```
# max(Training_Set_MSE, Validation_Set_MSE) / min(Training_Set_MSE, Validation_Set_MSE
# Compare Model 1 and Model 2
```

```
# Mode1
Training_Set_MSE = 0.3347568
Validation_Set_MSE =1.0035496000000002
ModelOne =Validation_Set_MSE/Training_Set_MSE
print("Model 1 MSE :", round(ModelOne,2))


# Model 2
Training_Set_MSE =0.17712
Validation_Set_MSE = 0.8608800000000001
ModelTwo =Validation_Set_MSE/Training_Set_MSE
print("Model 2 MSE :", round(ModelTwo,2))

# Compare to find better model
if ( ModelTwo < ModelOne ):
   print("Model 2 is a better model")
else:
   print("Model 1 is a better model")
```

```
Model 1 MSE : 3.0
Model 2 MSE : 4.86
Model 1 is a better model
```

** Test phase **

In [ ]:

```
# After getting better model based on MSE, we then find the values of y
# In our case MODEL 1 is better so we choose linear Regression Equation(y):
# ŷ=a1 + b1 * x
# we will take the value of slope(b) ad intercept(a) same what we calculated before fr
# slope(b) is:  0.86
# intercept(a):  0.52
x = np.array([1.4,2.5,3.6,4.5,5.4])
a= 0.52
b= 0.86
y=a+b*x
print(y)
```

```
[1.724 2.67  3.616 4.39  5.164]
```