**Building a Model:**

**Methods of Building a Model**

1. All – in (prior knowledge or guess)
2. Backward Elimination (Stepwise Regression, Fastest method)
3. Forward Selection (Stepwise Regression)
4. Bidirectional Elimination (Stepwise Regression)
5. Score Comparison

**Backward Elimination Model**

STEP 1: Select a Significance level to stay in the model(SL). (Suppose 0.05)

STEP 2: Fit the full model with all possible predictors. (Using all the independent variable)

STEP 3: Consider the predictor with the highest P-value.

If P > SL

Go to STEP 4

Else

FIN

STEP 4: Remove the predictor.

STEP 5: Fit model without this variable. Go to STEP 3

**Forward Selection Model**

STEP 1: Select a Significance level to stay in the model (SL). (Suppose 0.05)

STEP 2: Fit all possible simple regression models y ~ xn. Select the one with the lowest P-value.

STEP 3: Keep this variable and fit all possible models with one extra predictor added to the one(s)

STEP 4: Consider the predictor with the lowest P-value.

If P < SL

Go to STEP 3

Else

FIN (Keep the previous model)

**Bidirectional Elimination Model**

STEP 1: Select a significance level to enter and to stay in model. (SLENTRY = 0.05 and SLSTAY = 0.05)

STEP 2: Perform the next step of Forward Selection. (New variables must have P < SLENTRY to enter)

STEP 3: Perform all steps of Backward Elimination (old variables must have P < SLSTAY to stay)

STEP 4: No new variables can enter and no old variables can exit.

**All Possible Models**

STEP 1: Select a criterion of goodness of fit ( e.g Akaike criterion)

STEP 2: Construct all possible regression models: 2N -1 total combinations.

STEP 3: Select the one with the best criterion.

**Code for Multiple linear regression**

# Fitting Multiple Linear Regression to the Training set

from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred = regressor.predict(X\_test)

**Backward Elimination with p-values code**

import statsmodels.formula.api as sm

X=np.append(arr=np.ones((50,1)).astype(int),values=X,axis=1)

X\_opt=X[:,[0,1,2,3,4,5]]

regressor\_OLS=sm.OLS(endog=y,exog=X\_opt).fit()

regressor\_OLS.summary()

***Autometed code:***

import statsmodels.formula.api as sm

def backwardElimination(x, sl):

numVars = len(x[0])

for i in range(0, numVars):

regressor\_OLS = sm.OLS(y, x).fit()

maxVar = max(regressor\_OLS.pvalues).astype(float)

if maxVar > sl:

for j in range(0, numVars - i):

if (regressor\_OLS.pvalues[j].astype(float) == maxVar):

x = np.delete(x, j, 1)

regressor\_OLS.summary()

return x

SL = 0.05

X\_opt = X[:, [0, 1, 2, 3, 4, 5]]

X\_Modeled = backwardElimination(X\_opt, SL)

**Backward Elimination with p-values and Adjusted R Squared code**

import statsmodels.formula.api as sm

def backwardElimination(x, SL):

numVars = len(x[0])

temp = np.zeros((50,6)).astype(int)

for i in range(0, numVars):

regressor\_OLS = sm.OLS(y, x).fit()

maxVar = max(regressor\_OLS.pvalues).astype(float)

adjR\_before = regressor\_OLS.rsquared\_adj.astype(float)

if maxVar > SL:

for j in range(0, numVars - i):

if (regressor\_OLS.pvalues[j].astype(float) == maxVar):

temp[:,j] = x[:, j]

x = np.delete(x, j, 1)

tmp\_regressor = sm.OLS(y, x).fit()

adjR\_after = tmp\_regressor.rsquared\_adj.astype(float)

if (adjR\_before >= adjR\_after):

x\_rollback = np.hstack((x, temp[:,[0,j]]))

x\_rollback = np.delete(x\_rollback, j, 1)

print (regressor\_OLS.summary())

return x\_rollback

else:

continue

regressor\_OLS.summary()

return x

SL = 0.05

X\_opt = X[:, [0, 1, 2, 3, 4, 5]]

X\_Modeled = backwardElimination(X\_opt, SL)