# Automated BackwardElimination

## #Backward Elimination with p-values only:

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:

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)