In [5]:

```
# Creating automated backward elimination function with p-values
import statsmodels.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)
        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()
    regressor_OLS.summary()
    return x

# Applying the backward elimination

X = np.append(arr = np.ones((50, 1)).astype(int), values = X, axis = 1)
SL = 0.05
X_opt = X[:,[0,1,2,3,4,5]]
X_opt = np.array(X_opt, dtype=float)

X_Res = backwardElimination(X_opt, SL)
regressor_OLS = sm.OLS(endog = Y, exog = X_Res).fit()
regressor_OLS.summary()
```

Out[5]:

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.947
Model:	OLS	**Adj. R-squared:**	0.945
Method:	Least Squares	**F-statistic:**	849.8
Date:	Tue, 19 May 2020	**Prob (F-statistic):**	3.50e-32
Time:	16:42:53	**Log-Likelihood:**	-527.44
No. Observations:	50	**AIC:**	1059.
Df Residuals:	48	**BIC:**	1063.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4.903e+04	2537.897	19.320	0.000	4.39e+04	5.41e+04
x1	0.8543	0.029	29.151	0.000	0.795	0.913

Omnibus:	13.727	**Durbin-Watson:**	1.116
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	18.536
Skew:	-0.911	**Prob(JB):**	9.44e-05
Kurtosis:	5.361	**Cond. No.**	1.65e+05