Module 3 Homework

ISE-529 Predictive Analytics

In [1]:

```
#importing libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score as r2, mean_squared_error as mse, mean_absolute_error
```

1a) Read the file "HW Problem 1 Dataset.csv" into a dataframe and convert the category column X6 into binary dummary variables. Display the first three rows of the resulting dataset.

In [2]:

```
data = pd.read_csv(r'C:\Users\Chinmay\Downloads\HW Problem 1 Dataset.csv')
data.head()
```

Out[2]:

	X1	X2	X3	X4	X5	X6	Y
0	11	47	18	3	56	Yellow	153.157223
1	19	91	11	93	1	Red	809.384179
2	48	33	36	31	22	Red	395.466944
3	4	86	43	68	98	Yellow	892.610788
4	82	52	37	65	100	Blue	476.573108

In [34]:

```
#making the first entry of color category (Yellow) into default value for color category

data['X6_Blue'] = pd.get_dummies(data['X6'])['Blue']
data['X6_Red'] = pd.get_dummies(data['X6'])['Red']
data.head(3)
```

Out[34]:

	X1	X2	X3	X4	X5	X6	Y	X6_Blue	X6_Red	X3_squared
0	11	47	18	3	56	Yellow	153.157223	0	0	324
1	19	91	11	93	1	Red	809.384179	0	1	121
2	48	33	36	31	22	Red	395.466944	0	1	1296

In [ ]:

1b) Using statsodels, perform a regression for Y using X1 through X5 and your dummy variables display the fit summary below.

In [4]:

```
X = data.drop(['Y','X6'],axis=1)
y = data['Y']

X = sm.add_constant(X)
mlr1 = sm.OLS(y,X).fit().summary()
mlr1
```

Out[4]:

OLS Regression Results

Dep. Variable:	Y	**R-squared:**	0.892
Model:	OLS	**Adj. R-squared:**	0.891
Method:	Least Squares	**F-statistic:**	1170.
Date:	Mon, 18 Jul 2022	**Prob (F-statistic):**	0.00
Time:	22:47:12	**Log-Likelihood:**	-6515.8
No. Observations:	1000	**AIC:**	1.305e+04
Df Residuals:	992	**BIC:**	1.309e+04
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	19.4186	22.324	0.870	0.385	-24.389	63.226
X1	4.0399	0.181	22.331	0.000	3.685	4.395
X2	0.0312	0.179	0.174	0.862	-0.321	0.383
X3	13.0721	0.181	72.131	0.000	12.717	13.428
X4	4.8075	0.180	26.651	0.000	4.454	5.162
X5	0.0114	0.182	0.063	0.950	-0.346	0.369
X6_Blue	-473.6667	11.629	-40.732	0.000	-496.487	-450.847
X6_Red	-90.8354	14.185	-6.404	0.000	-118.671	-63.000

Omnibus:	3.577	**Durbin-Watson:**	1.920
Prob(Omnibus):	0.167	**Jarque-Bera (JB):**	3.649
Skew:	0.139	**Prob(JB):**	0.161
Kurtosis:	2.899	**Cond. No.**	514.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

1c) Investigating the resulting coefficient p-values, Which predictors appear to not have a statistically significant relationship to the response variable?

Columns X2 (p-value: 0.862) and X5 (p-value: 0.950) appear to not have a statistically significant relationship to response variable as p-value > 0.05

1d) Drop any predictors that you found not to have a relationship with the response and display the first 10 rows of the resulting dataframe.

In [35]:

```
df = data.drop(['X2','X5','X6'],axis=1)
df.head(10)
```

Out[35]:

	X1	X3	X4	Y	X6_Blue	X6_Red	X3_squared
0	11	18	3	153.157223	0	0	324
1	19	11	93	809.384179	0	1	121
2	48	36	31	395.466944	0	1	1296
3	4	43	68	892.610788	0	0	1849
4	82	37	65	476.573108	1	0	1369
5	41	6	88	797.891711	0	0	36
6	29	83	12	871.984975	1	0	6889
7	22	44	89	952.367041	0	0	1936
8	12	39	67	343.993916	1	0	1521
9	2	68	96	1297.651894	0	0	4624

1e) Re-run the regression without the irrelevant variables and display the fit summary

In [6]:

```
X = df.drop('Y',axis=1)
y = df['Y']

X = sm.add_constant(X)
mlr2 = sm.OLS(y,X).fit().summary()
mlr2
```

Out[6]:

OLS Regression Results

Dep. Variable:	Y	**R-squared:**	0.892
Model:	OLS	**Adj. R-squared:**	0.891
Method:	Least Squares	**F-statistic:**	1641.
Date:	Mon, 18 Jul 2022	**Prob (F-statistic):**	0.00
Time:	22:47:12	**Log-Likelihood:**	-6515.9
No. Observations:	1000	**AIC:**	1.304e+04
Df Residuals:	994	**BIC:**	1.307e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	21.5910	18.091	1.193	0.233	-13.911	57.093
X1	4.0403	0.181	22.375	0.000	3.686	4.395
X3	13.0704	0.181	72.310	0.000	12.716	13.425
X4	4.8084	0.180	26.693	0.000	4.455	5.162
X6_Blue	-473.6877	11.608	-40.806	0.000	-496.468	-450.908
X6_Red	-90.8587	14.169	-6.412	0.000	-118.664	-63.053

Omnibus:	3.549	**Durbin-Watson:**	1.919
Prob(Omnibus):	0.170	**Jarque-Bera (JB):**	3.621
Skew:	0.138	**Prob(JB):**	0.164
Kurtosis:	2.898	**Cond. No.**	346.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [7]:

```
### This cell is just a trial
### Since the constant term has a p-value > 0.05 I tried removing it also and the r2 fit ju

X = df.drop(['Y'],axis=1)
y = df['Y']


mlr2_1 = sm.OLS(y,X).fit().summary()
mlr2_1
```

Out[7]:

OLS Regression Results

Dep. Variable:	Y	**R-squared (uncentered):**	0.976
Model:	OLS	**Adj. R-squared (uncentered):**	0.976
Method:	Least Squares	**F-statistic:**	8163.
Date:	Mon, 18 Jul 2022	**Prob (F-statistic):**	0.00
Time:	22:47:12	**Log-Likelihood:**	-6516.6
No. Observations:	1000	**AIC:**	1.304e+04
Df Residuals:	995	**BIC:**	1.307e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
X1	4.1518	0.155	26.858	0.000	3.848	4.455
X3	13.1876	0.152	86.872	0.000	12.890	13.485
X4	4.9221	0.153	32.191	0.000	4.622	5.222
X6_Blue	-469.2810	11.008	-42.632	0.000	-490.882	-447.680
X6_Red	-86.7950	13.757	-6.309	0.000	-113.791	-59.799

Omnibus:	3.724	**Durbin-Watson:**	1.923
Prob(Omnibus):	0.155	**Jarque-Bera (JB):**	3.798
Skew:	0.143	**Prob(JB):**	0.150
Kurtosis:	2.900	**Cond. No.**	262.

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

1f) Write the full regression equation

This equation is with constant term:

y = 21.5910 + 4.0403(X1) + 13.0704(X3) +4.8084(X4) -473.6877(X6_Blue) -90.8587(X6_Red)

1g) Write the equation for the observations where the "color" category is yellow:

Because this dataframe considers yellow to be default; hence, that is the case when blue and red are 0

y = 21.5910 + 4.0403(X1) + 13.0704(X3) +4.8084(X4)

1h) Write the equation for the observations where the "color" category is blue:

y = 21.5910 + 4.0403(X1) + 13.0704(X3) +4.8084(X4) - 473.6877(X6_Blue)

Write the equation for the observations where the "color" category is red:

y = 21.5910 + 4.0403(X1) + 13.0704(X3) +4.8084(X4) - 90.8587(X6_Red)

1i) Now, use the sklearn library to run the same regression and display the resulting model coefficients

In [8]:

```
lr = LinearRegression(fit_intercept=True)
pd.DataFrame(lr.fit(X,y).coef_,columns=(['Coefficients']) ,index=X.columns)
```

Out[8]:

	Coefficients
X1	4.040325
X3	13.070444
X4	4.808389
X6_Blue	-473.687750
X6_Red	-90.858682

1j) Calculate and display the following fit assessment statistics: R^2, Mean Squared Error, Mean Absolute Error, and Max Error

In [9]:

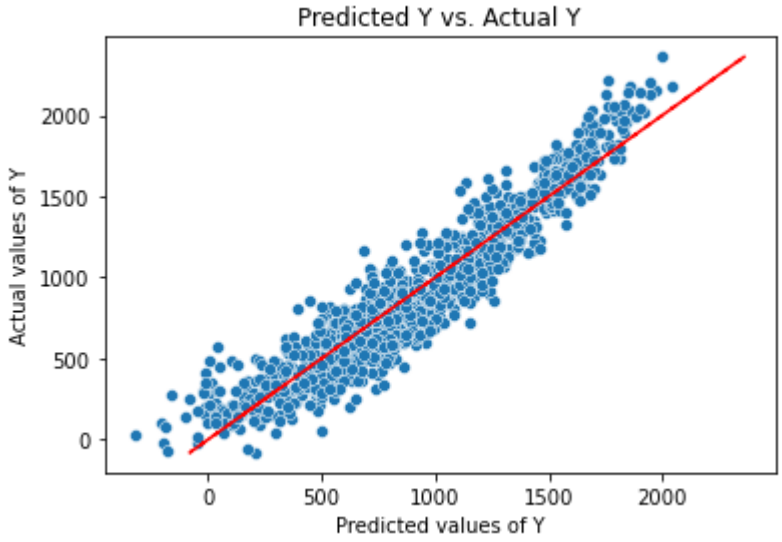
```
lr = LinearRegression(fit_intercept=True).fit(X,y)

y_hat = lr.predict(X)

r2 = print(f'R2_error is :{r2(y_hat,y)}')
mse = print(f'MSE is :{mse(y_hat,y)}')
mae = print(f'MAE is :{mae(y_hat,y)}')
me = print(f'MAE is :{me(y_hat,y)}')
```

```
R2_error is :0.8788911841790388
MSE is :26738.19374639029
MAE is :130.86268562302578
MAE is :540.8391996665022
```

1k) Using Seaborn, create a scatterplot of the actual values of Y vs the predicted values of Y

In [10]:

```
sns.scatterplot(data=df, x=y_hat, y=y)
plt.plot(y, y, linestyle="--",color='red')
plt.ylabel("Actual values of Y")
plt.xlabel("Predicted values of Y")
plt.title("Predicted Y vs. Actual Y")
plt.show()
```



Investigate adding nonlinear terms

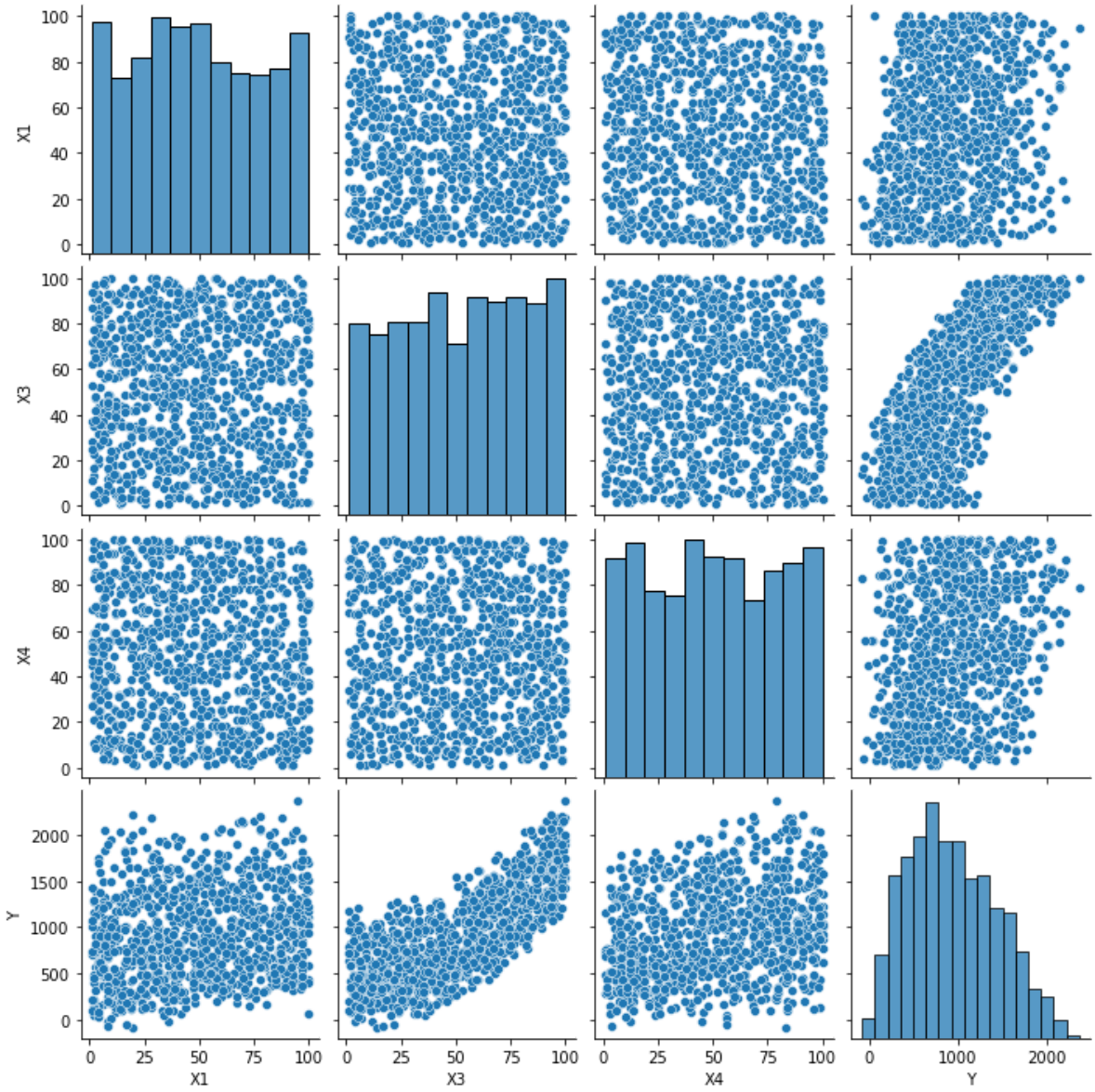
1L) Now, create one scatterplot for each numeric predictor (not including dummy variables) against the response variables:

In [11]:

```
sns.pairplot(data.drop(['X2','X5','X6_Red','X6_Blue'], axis=1))
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x19175cd9490>
```



1M) Which predictor or predictors appear to have a nonlinear relationship with the response variable?

X3 seems to have a non-linear relationship with the response variable.

1n) Try adding a squared term of any predictors that appear to have a nonlinear relationship. Re-run the regression and display the resulting coefficients and assessment statistics (R^2, Mean Squared Error, Mean

Absolute Error, and Max Error)

In [12]:

```
df['X3_squared'] = df['X3']**2
X = df.drop('Y',axis=1)
y = df['Y']

X = sm.add_constant(X)
mlr3 = sm.OLS(y,X).fit().summary()
mlr3
```

Out[12]:

OLS Regression Results

Dep. Variable:	Y	**R-squared:**	0.939
Model:	OLS	**Adj. R-squared:**	0.939
Method:	Least Squares	**F-statistic:**	2561.
Date:	Mon, 18 Jul 2022	**Prob (F-statistic):**	0.00
Time:	22:47:14	**Log-Likelihood:**	-6227.6
No. Observations:	1000	**AIC:**	1.247e+04
Df Residuals:	993	**BIC:**	1.250e+04
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	268.1458	16.205	16.547	0.000	236.345	299.946
X1	4.0083	0.135	29.597	0.000	3.743	4.274
X3	-1.8530	0.553	-3.350	0.001	-2.939	-0.767
X4	4.8822	0.135	36.132	0.000	4.617	5.147
X6_Blue	-460.3862	8.719	-52.802	0.000	-477.496	-443.276
X6_Red	-72.5791	10.647	-6.817	0.000	-93.472	-51.686
X3_squared	0.1469	0.005	27.825	0.000	0.137	0.157

Omnibus:	1.501	**Durbin-Watson:**	2.042
Prob(Omnibus):	0.472	**Jarque-Bera (JB):**	1.568
Skew:	-0.072	**Prob(JB):**	0.457
Kurtosis:	2.870	**Cond. No.**	1.98e+04

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.98e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

In [15]:

```
lr2 = LinearRegression(fit_intercept=True).fit(X,y)
pd.DataFrame(data=lr2.coef_, columns=(['Coefficients']), index=X.columns)
```

Out[15]:

	Coefficients
const	0.000000
X1	4.008261
X3	-1.852961
X4	4.882227
X6_Blue	-460.386174
X6_Red	-72.579109
X3_squared	0.146948

In [16]:

```
y_hat_2 = lr2.predict(X)

r2 = print(f'R2_error is :{r2(y_hat_2,y)}')
mse = print(f'MSE is :{mse(y_hat_2,y)}')
mae = print(f'MAE is :{mae(y_hat_2,y)}')
me = print(f'MAE is :{me(y_hat_2,y)}')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-16-933a55afa1df> in <module>
      1 y_hat_2 = lr2.predict(X)
      2
----> 3 r2 = print(f'R2_error is :{r2(y_hat_2,y)}')
      4 mse = print(f'MSE is :{mse(y_hat_2,y)}')
      5 mae = print(f'MAE is :{mae(y_hat_2,y)}')

TypeError: 'NoneType' object is not callable
```

In [17]:

```
### For some reason it is not printing now. It was being printed unitl now. I have it even
```

Investigate adding interaction effects

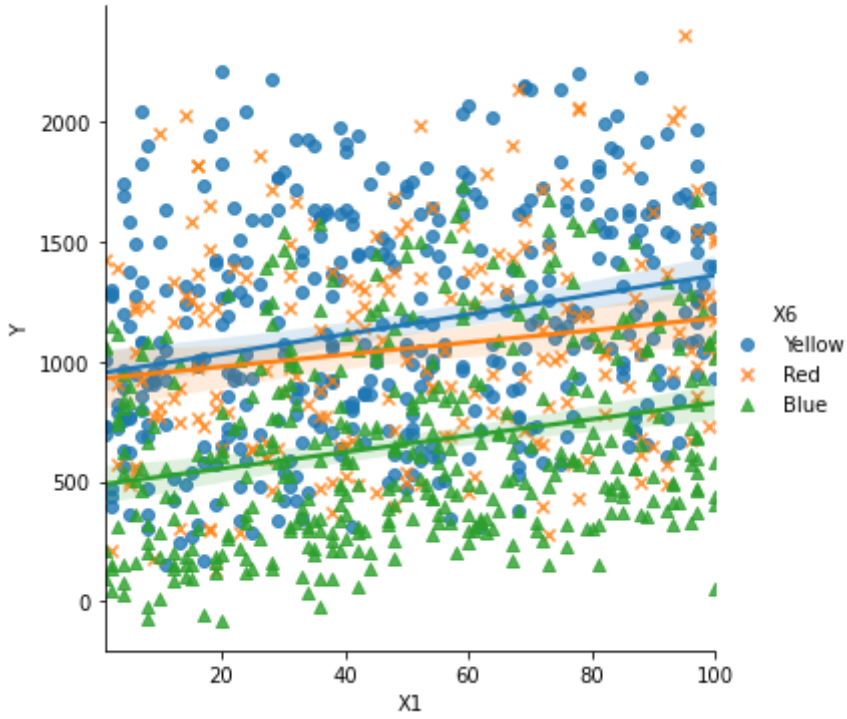
1o) For each numeric predictor, plot a scatterplot against the response variable color coding and the points according to their category values and include regresison lines

In [18]:

```
sns.lmplot(data=data, x='X1', y='Y', hue='X6', markers=['o','x','^'])
```

Out[18]:

```
<seaborn.axisgrid.FacetGrid at 0x191768eafd0>
```



In [19]:

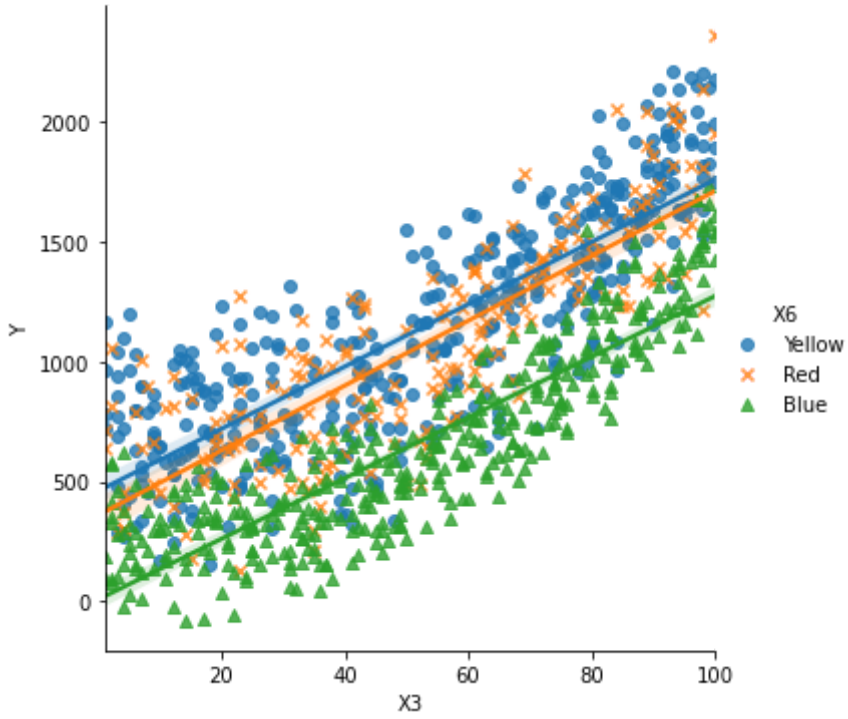
```
## X1 seems to have some interaction with Red; but here is a judgement call, that the lines
## seem parallel to each other. So hence my judgement is that I won't be considering an int
## And also I asked this to Prof. Bruce as well ,who said it's my personal judgement call.
```

In [20]:

```
sns.lmplot(data=data, x='X3', y='Y', hue='X6', markers=['o','x','^'])
```

Out[20]:

```
<seaborn.axisgrid.FacetGrid at 0x19177d7c310>
```



In [21]:

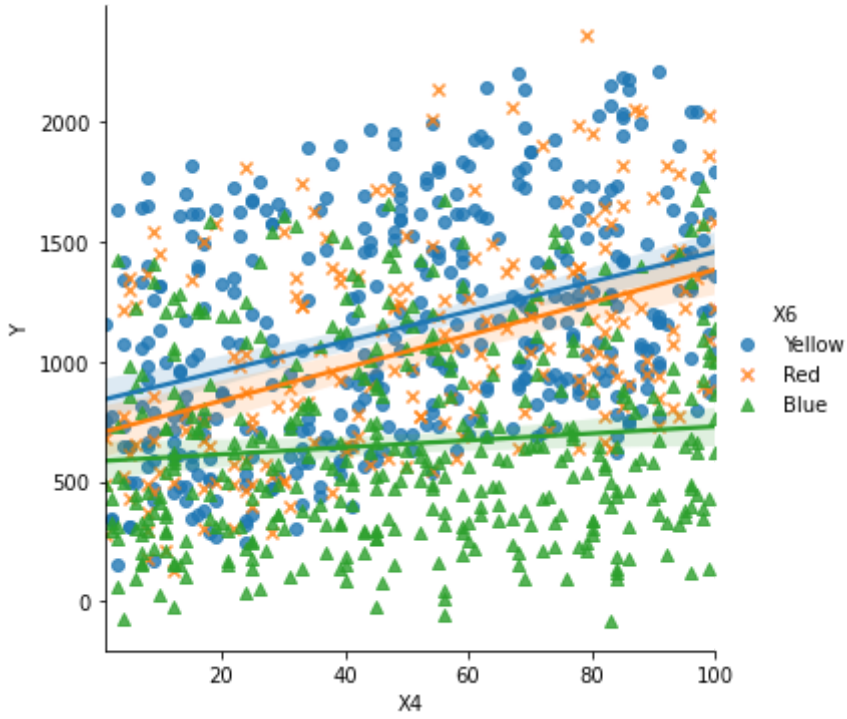
```
## No effect
```

In [22]:

```
sns.lmplot(data=data, x='X4', y='Y', hue='X6', markers=['o','x','^'])
```

Out[22]:

```
<seaborn.axisgrid.FacetGrid at 0x19177e2b100>
```



In [23]:

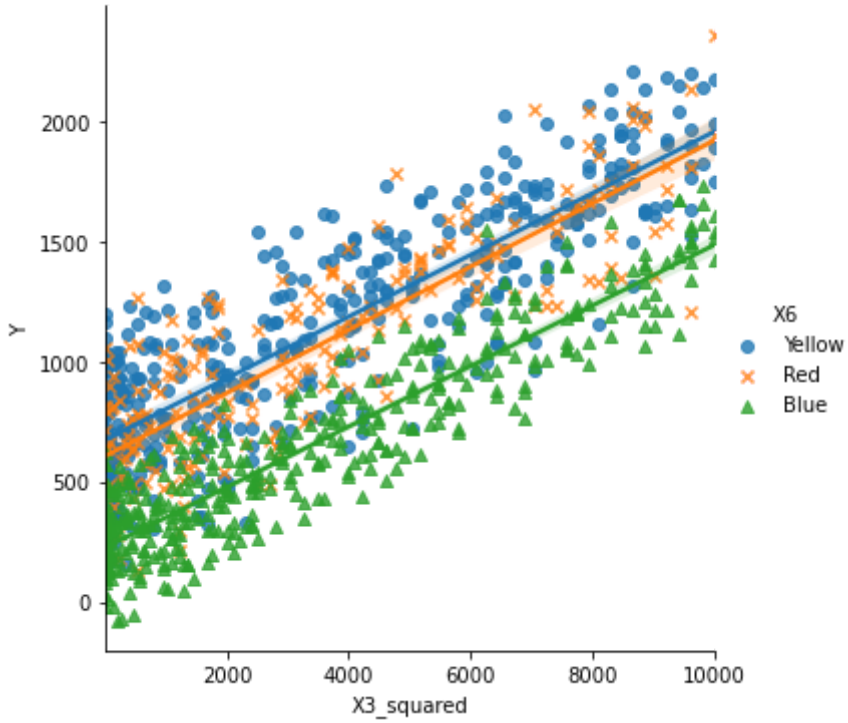
```
## Yellow and Red somehow seem to be somewhat parallel; but blue is showing a clear interac
## be constructed
```

In [24]:

```
data['X3_squared'] = data['X3']**2
sns.lmplot(data=data, x='X3_squared', y='Y', hue='X6', markers=['o','x','^'])
```

Out[24]:

```
<seaborn.axisgrid.FacetGrid at 0x19177e991f0>
```



In [25]:

```
## No effect seen whatsoever
```

1p) Which predictor appears to have interactions with the color category?

X4 seems to have an interaction with Blue category

1q) Add an interaction effect to the model for this predictor, run the new regression, and display the coefficients and fit statistics

In [32]:

```
df['X4*Blue'] = df['X4']*df['X6_Blue']
df['X1*Red'] = df['X4']*df['X6_Red']

x1 = df.drop('Y',axis=1)
y = df['Y']

lr3 = LinearRegression(fit_intercept=True).fit(x1,y)
pd.DataFrame(lr3.coef_, columns=(['Coefficients']), index=x1.columns)
```

Out[32]:

	Coefficients
X1	4.115607
X3	-1.876092
X4	6.967302
X6_Blue	-202.419518
X6_Red	-63.006278
X3_squared	0.147005
X4*Blue	-5.196123
X1*Red	-0.265455

In [30]:

```
y_hat_3 = lr3.predict(x1)
r2 = print(f'R2_error is :{r2(y_hat_3,y)}')
mse = print(f'MSE is :{mse(y_hat_3,y)}')
mae = print(f'MAE is :{mae(y_hat_3,y)}')
me = print(f'MAE is :{me(y_hat_3,y)}')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-30-8995c94ab4e9> in <module>
      1 y_hat_3 = lr3.predict(x1)
----> 2 r2 = print(f'R2_error is :{r2(y_hat_3,y)}')
      3 mse = print(f'MSE is :{mse(y_hat_3,y)}')
      4 mae = print(f'MAE is :{mae(y_hat_3,y)}')
      5 me = print(f'MAE is :{me(y_hat_3,y)}')

TypeError: 'NoneType' object is not callable
```

1r) Using statsmodels, run the same regression and assess the p-values of the coefficients. Which interaction affects appear to be statistically significant?

In [33]:

```
x1 = sm.add_constant(x1)
sm.OLS(y,x1).fit().summary()
```

Out[33]:

OLS Regression Results

Dep. Variable:	Y	**R-squared:**	0.960
Model:	OLS	**Adj. R-squared:**	0.960
Method:	Least Squares	**F-statistic:**	2981.
Date:	Mon, 18 Jul 2022	**Prob (F-statistic):**	0.00
Time:	22:52:44	**Log-Likelihood:**	-6017.8
No. Observations:	1000	**AIC:**	1.205e+04
Df Residuals:	991	**BIC:**	1.210e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	159.6626	14.894	10.720	0.000	130.435	188.891
X1	4.1156	0.110	37.400	0.000	3.900	4.332
X3	-1.8761	0.449	-4.178	0.000	-2.757	-0.995
X4	6.9673	0.173	40.188	0.000	6.627	7.308
X6_Blue	-202.4195	14.156	-14.299	0.000	-230.199	-174.640
X6_Red	-63.0063	17.426	-3.616	0.000	-97.202	-28.810
X3_squared	0.1470	0.004	34.295	0.000	0.139	0.155
X4*Blue	-5.1961	0.247	-21.069	0.000	-5.680	-4.712
X1*Red	-0.2655	0.295	-0.900	0.368	-0.844	0.313

Omnibus:	2.115	**Durbin-Watson:**	1.985
Prob(Omnibus):	0.347	**Jarque-Bera (JB):**	2.130
Skew:	-0.030	**Prob(JB):**	0.345
Kurtosis:	3.218	**Cond. No.**	3.11e+04

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.11e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

Since p-value for X4(Blue) is <0.05 it can be considered statistically significant

But p-value for X1(Red) is >>0.05; hence insignificant and must be removed.

R2 value in both cases is same; hence better to remove in final model.

In [ ]: