Name : Shubham Sapkal

Roll No. : 2012118

Subject : KR AI ML Dl

Class : SYMCA

Assignment : Assignment 03

```
# Dummy variables
## Import the relevant libraries
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()

## Load the data
raw_data = pd.read_csv("1.03.+Dummies.csv")
raw_data
```

	SAT	GPA	Attendance
0	1714	2.40	No
1	1664	2.52	No
2	1760	2.54	No
3	1685	2.74	No
4	1693	2.83	No
...	...	...	...
79	1936	3.71	Yes
80	1810	3.71	Yes
81	1987	3.73	No
82	1962	3.76	Yes

83 2050 3.81 Yes

```
## Map the data
data = raw_data.copy()
data['Attendance'] = data['Attendance'].map({'Yes':1,'No':0})
data
```

	SAT	GPA	Attendance
0	1714	2.40	0
1	1664	2.52	0
2	1760	2.54	0
3	1685	2.74	0
4	1693	2.83	0
...	...	...	...
79	1936	3.71	1
80	1810	3.71	1
81	1987	3.73	0
82	1962	3.76	1
83	2050	3.81	1

```
data.describe()
```

	SAT	GPA	Attendance
count	84.000000	84.000000	84.000000
mean	1845.273810	3.330238	0.464286
std	104.530661	0.271617	0.501718
min	1634.000000	2.400000	0.000000
25%	1772.000000	3.190000	0.000000
50%	1846.000000	3.380000	0.000000
75%	1934.000000	3.502500	1.000000
max	2050.000000	3.810000	1.000000

```
## Regression
y = data['GPA']
x1 = data[['SAT','Attendance']]
x = sm.add_constant(x1)
results = sm.OLS(y,x).fit()
results.summary()
```

OLS Regression Results

Dep. Variable:	GPA	R-squared:	0.565
Model:	OLS	Adj. R-squared:	0.555
Method:	Least Squares	F-statistic:	52.70
Date:	Sat, 29 Jan 2022	Prob (F-statistic):	2.19e-15
Time:	21:35:03	Log-Likelihood:	25.798
No. Observations:	84	AIC:	-45.60
Df Residuals:	81	BIC:	-38.30
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.6439	0.358	1.797	0.076	-0.069	1.357
SAT	0.0014	0.000	7.141	0.000	0.001	0.002
Attendance	0.2226	0.041	5.451	0.000	0.141	0.304

Omnibus:	19.560	Durbin-Watson:	1.009
Prob(Omnibus):	0.000	Jarque-Bera (JB):	27.189
Skew:	-1.028	Prob(JB):	1.25e-06
Kurtosis:	4.881	Cond. No.	3.35e+04

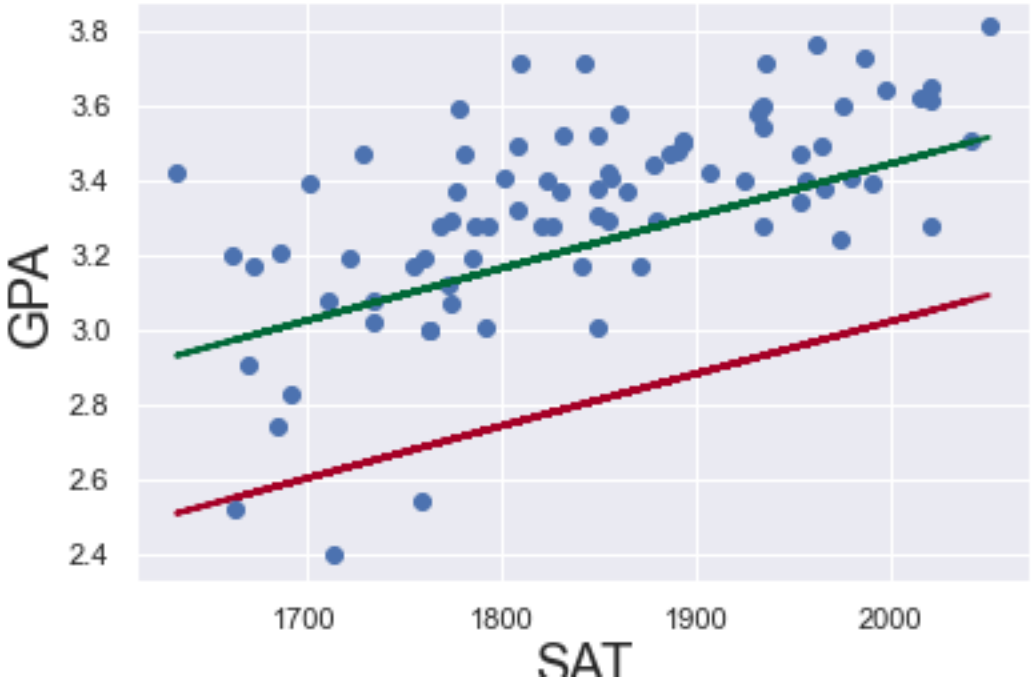
Notes:

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

[2] The condition number is large, 3.35e+04. This might indicate that there are strong multicollinearity or other numerical problems.

Plot the regression line(s) on the scatter plot

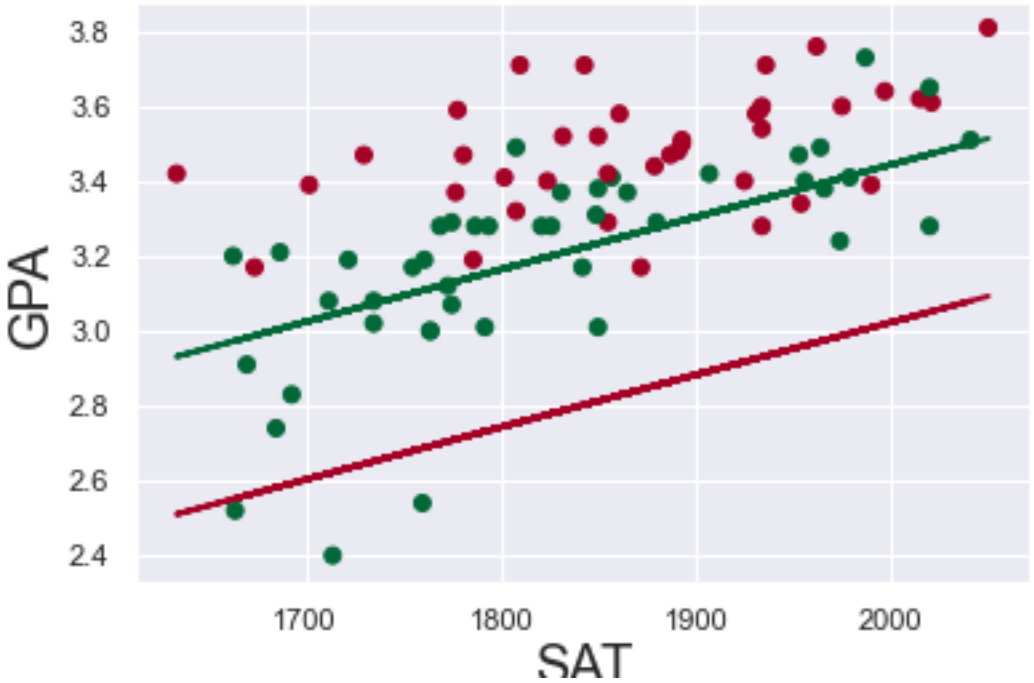
```
plt.scatter(data['SAT'],y)
yhat_no = 0.6439 + 0.0014*data['SAT']
yhat_yes = 0.2226 + 0.0014*data['SAT']
fig = plt.plot(data['SAT'],yhat_no, lw=2, c='#006837')
fig = plt.plot(data['SAT'],yhat_yes, lw=2, c='#a50026')
plt.xlabel('SAT', fontsize = 20)
plt.ylabel('GPA', fontsize = 20)
plt.show()
```



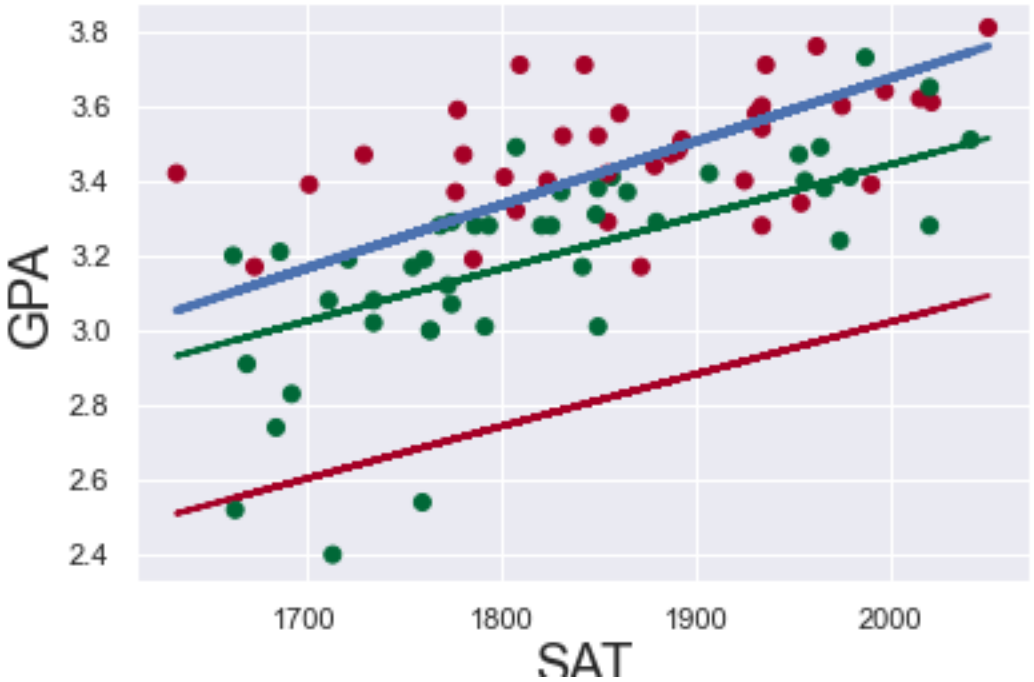
Plot the regression line(s) on the scatter plot and colour the data points

```
plt.scatter(data['SAT'],y,c=data['Attendance'],cmap='RdYlGn_r')
yhat_no = 0.6439 + 0.0014*data['SAT']
yhat_yes = 0.2226 + 0.0014*data['SAT']
fig = plt.plot(data['SAT'],yhat_no, lw=2, c='#006837')
fig = plt.plot(data['SAT'],yhat_yes, lw=2, c='#a50026')
```

```
plt.xlabel('SAT', fontsize = 20)
plt.ylabel('GPA', fontsize = 20)
plt.show()
```




```
## Add the original regression line for comparison
plt.scatter(data['SAT'],data['GPA'], c=data['Attendance'],cmap='RdYlGn_r')
yhat_no = 0.6439 + 0.0014*data['SAT']
yhat_yes =0.2226  + 0.0014*data['SAT']
yhat = 0.0017*data['SAT'] + 0.275
fig = plt.plot(data['SAT'],yhat_no, lw=2, c='#006837', label ='regression line1')
fig = plt.plot(data['SAT'],yhat_yes, lw=2, c='#a50026', label ='regression line2')
fig = plt.plot(data['SAT'],yhat, lw=3, c='#4C72B0', label ='regression line')
plt.xlabel('SAT', fontsize = 20)
plt.ylabel('GPA', fontsize = 20)
plt.show()
```



How to make predictions based on the regressions we create

X

	const	SAT	Attendance
0	1.0	1714	0
1	1.0	1664	0
2	1.0	1760	0
3	1.0	1685	0
4	1.0	1693	0
...	...	...	...
79	1.0	1936	1
80	1.0	1810	1
81	1.0	1987	0
82	1.0	1962	1
83	1.0	2050	1

84 rows × 3 columns

```
new_data = pd.DataFrame({'const': 1,'SAT': [1540, 1840], 'Attendance': [0, 1]})
new_data = new_data[['const','SAT','Attendance']]
new_data
```

	const	SAT	Attendance
0	1	1540	0
1	1	1840	1

```
new_data.rename(index={0: 'Ram',1:'Balram'})
```

	const	SAT	Attendance
Ram	1	1540	0
Balram	1	1840	1

```
predictions = results.predict(new_data)
predictions
0    2.799545
1    3.442129
dtype: float64
predictionsdf = pd.DataFrame({'Predictions':predictions})
joined = new_data.join(predictionsdf)
joined.rename(index={0: 'Ram',1:'Balram'})
```

	const	SAT	Attendance	Predictions
Ram	1	1540	0	2.799545
Balram	1	1840	1	3.442129