Multilinear Regression&Prediction_GPA

November 13, 2020

[1]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

[10]:
```
raw_data = pd.read_csv('GPA&SAT_Scores.csv')
```

[11]:
```
raw_data
```

[11]:

	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

[84 rows x 3 columns]

[12]:
```
data = raw_data
data['Attendance']= data['Attendance'].map({'Yes':1,'No':0})
```

[13]:
```
data.describe()
```

[13]:

	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
```

[14]:
```
y = data["GPA"]
x1= data[['SAT','Attendance']]
```

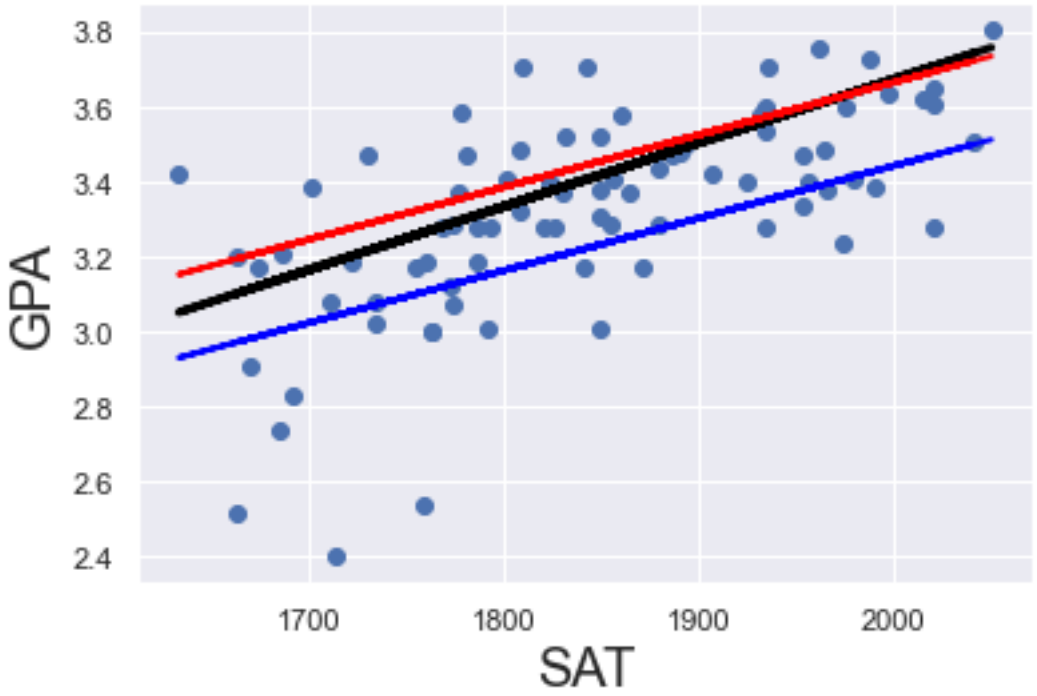
[15]:
```
x = sm.add_constant(x1)
results = sm.OLS(y,x).fit()
results.summary()
```

[15]:
```
<class 'statsmodels.iolib.summary.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:                Fri, 13 Nov 2020   Prob (F-statistic):           2.19e-15
Time:                        21:31:33   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
==============================================================================

Warnings:
[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.
"""
```

[24]:
```
plt.scatter(data['SAT'],y)
yhat= 0.0017*data['SAT']+0.275
yhatno= 0.6439+0.0014*data['SAT']
```

```
yhatyes=0.8665+0.0014*data['SAT']
fig = plt.plot(data['SAT'], yhat, lw=3, color='black', label= 'Original␣
↪Regression Line')
fig = plt.plot(data['SAT'],yhatno, lw=2, color='blue', label= 'No Attendance␣
↪Reg Line')
fig = plt.plot(data['SAT'],yhatyes, lw=2, color='red', label= 'Yes Attendance␣
↪Reg Line')
plt.xlabel('SAT',fontsize=20)
plt.ylabel('GPA',fontsize=20)
plt.show()
```



[26]:
```
new_data = pd.DataFrame({'const':1,'SAT':[1700,1670],'Attendance':[0,1]})
new_data = new_data[['const','SAT','Attendance']]
new_data
```

[26]:

	const	SAT	Attendance
0	1	1700	0
1	1	1670	1

[27]:
```
new_data.rename(index={0:'Bob',1:'Alice'})
```

[27]:

	const	SAT	Attendance
Bob	1	1700	0

```
Alice      1  1670           1
```

[29]:
```
prediction= results.predict(new_data)
prediction
```

[29]:
```
0    3.023513
1    3.204163
dtype: float64
```

[33]:
```
predictiondf=pd.DataFrame({'Predictions': prediction})
joined = new_data.join(predictiondf)
joined.rename(index={0:'Bob',1:'Alice'})
```

[33]:

	const	SAT	Attendance	Predictions
Bob	1	1700	0	3.023513
Alice	1	1670	1	3.204163

[ ]: