In [8]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sbs
sbs.set()

from sklearn.linear_model import LinearRegression
```

In [9]:
```
data = pd.read_csv('1.02. Multiple linear regression.csv')
data.head()
```

Out[9]:

	SAT	GPA	Rand 1,2,3
0	1714	2.40	1
1	1664	2.52	3
2	1760	2.54	3
3	1685	2.74	3
4	1693	2.83	2

In [10]:
```
data.describe()
```

Out[10]:

	SAT	GPA	Rand 1,2,3
count	84.000000	84.000000	84.000000
mean	1845.273810	3.330238	2.059524
std	104.530661	0.271617	0.855192
min	1634.000000	2.400000	1.000000
25%	1772.000000	3.190000	1.000000
50%	1846.000000	3.380000	2.000000
75%	1934.000000	3.502500	3.000000
max	2050.000000	3.810000	3.000000

In [11]:
```
x = data[['SAT', 'Rand 1,2,3']]
y = data['GPA']
```

In [12]:
```
reg = LinearRegression()
```

In [13]:
```
reg.fit(x,y)
```

Out[13]:
```
LinearRegression()
```

In [14]:
```
reg.coef_
```

Out[14]:
```
array([ 0.00165354, -0.00826982])
```

In [15]:
```
reg.intercept_
```

Out[15]:
```
0.29603261264909486
```

In [16]:
```
# R-squared
reg.score(x,y)
```

Out[16]:
```
0.40668119528142843
```

Formula for Adjusted R-squared

$R^2_{adj.} = 1 - (1 - R^2) * \frac{n-1}{n-p-1}$

In [18]:
```
x.shape
```

Out[18]:
```
(84, 2)
```

In [19]:
```
r2 = reg.score(x,y)
n = x.shape[0]
p = x.shape[1]
adjusted_r2 = 1 - (1-r2) * (n-1)/(n-p-1)
adjusted_r2
```

Out[19]:
```
0.39203134825134023
```

In [ ]: