In [ ]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
length = 99.11/1000 #in mm
width = 11.40/1000 #in mm
thick = 5.31/1000 #in mm
area = width*thick
```

Aluminium

In [ ]:
```
aluminium = pd.read_excel("aluminium.xlsx").fillna(0)
```

In [ ]:
```
X=np.array(aluminium.loc[:,"extension"])/1000
col=np.ones(100)
arr = np.column_stack((X,col))
```

In [ ]:
```
y = (np.array(aluminium.loc[:,"load"])/area).reshape(100)
```

In [ ]:
```
import statsmodels.api as sm
model = sm.OLS(y, arr).fit()
```

In [ ]:
```
model.summary()
```

Out[ ]:

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.531
Model:	OLS	**Adj. R-squared:**	0.526
Method:	Least Squares	**F-statistic:**	111.0
Date:	Mon, 10 Oct 2022	**Prob (F-statistic):**	8.44e-18
Time:	23:05:30	**Log-Likelihood:**	-1899.7
No. Observations:	100	**AIC:**	3803.
Df Residuals:	98	**BIC:**	3809.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
x1	6.817e+09	6.47e+08	10.534	0.000	5.53e+09	8.1e+09
const	8.2e+07	8.51e+06	9.634	0.000	6.51e+07	9.89e+07

Omnibus:	73.023	**Durbin-Watson:**	0.221
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	411.789
Skew:	-2.428	**Prob(JB):**	3.81e-90
Kurtosis:	11.674	**Cond. No.**	149.

Notes:

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

In [ ]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
```

In [ ]:
```
regressor.fit(arr, y)
```

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

In [ ]:
```
print("young's modulus = "+str(regressor.coef_[0]*1e-9)+"GPa")
```

```
young's modulus = 6.817209409795241GPa
```

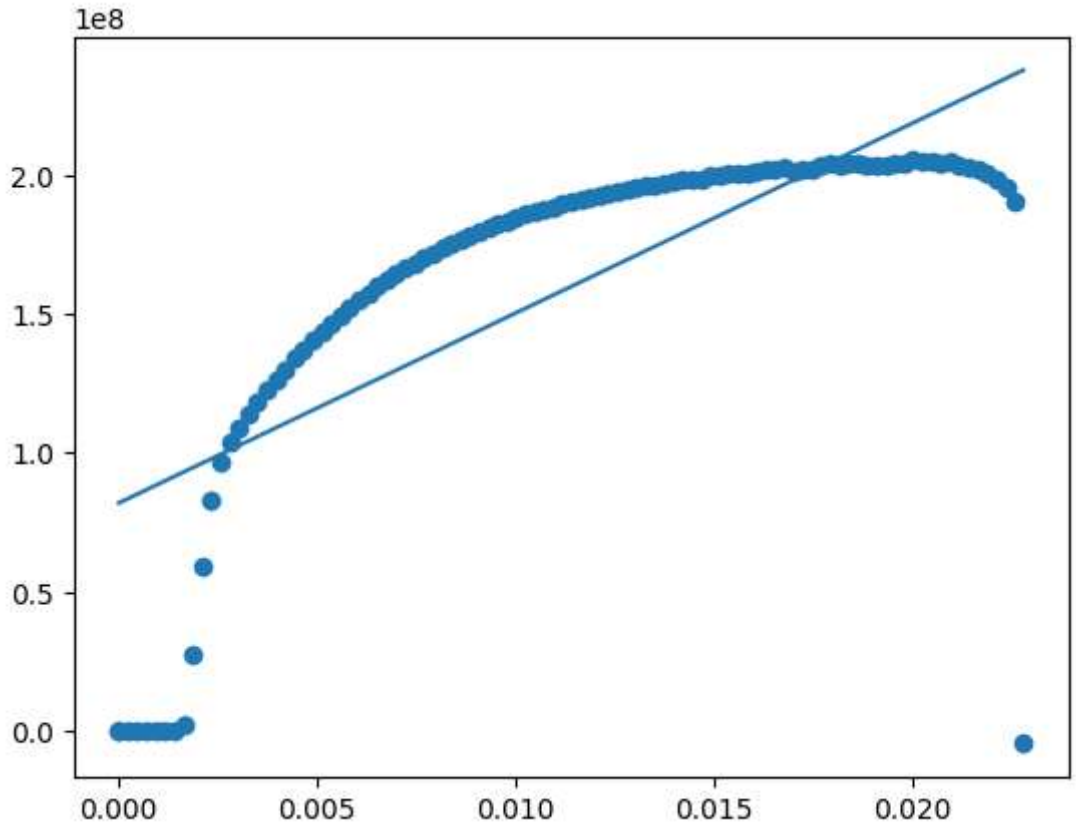
In [ ]:
```
regressor
```

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

plot

In [ ]:
```
ypredict = regressor.predict(arr)
```

In [ ]:
```
import matplotlib.pyplot as plt
plt.scatter(X,y)
plt.plot(X,ypredict)
plt.xticks()
plt.yticks()
plt.show()
```



Mild Steel

In [ ]:
```
mildSteel = pd.read_excel("mildSheet.xlsx").fillna(0)
```

In [ ]:
```
X1=np.array(mildSteel["extension"])/1000
col1=np.ones(100)
arr1 = np.column_stack((X1,col1))
```

In [ ]:
```
X11=np.array(mildSteel.loc[12:21,"extension"])/1000
col11=np.ones(10)
arr11 = np.column_stack((X11,col11))
```

In [ ]:
```
y1 = (np.array(mildSteel.loc[:,"load"])/area).reshape(100)
```

In [ ]:
```
y11 = (np.array(mildSteel.loc[12:21,"load"])/area).reshape(10)
```

In [ ]:
```
regressor1 = LinearRegression()
```

In [ ]:
```
from sklearn.linear_model import LinearRegression
regressor11 = LinearRegression()
```

In [ ]:
```
regressor1.fit(arr1, y1)
ypredict1=regressor1.predict(arr1)
```

In [ ]:
```
regressor11.fit(arr11, y11)
ypredict11=regressor11.predict(arr11)
```

In [ ]:
```
print("young's modulus = "+str(regressor1.coef_[0]*1e-9)+"GPa")
```

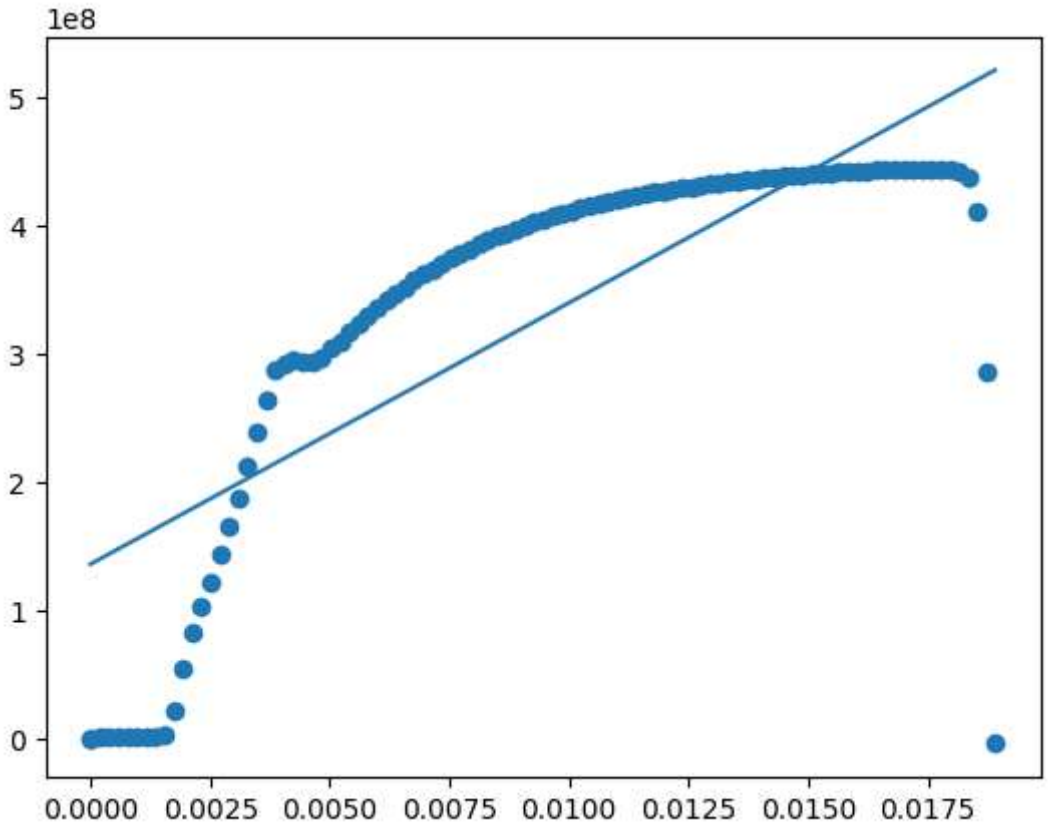
```
young's modulus = 20.3699933358117GPa
```

In [ ]:
```
print("young's modulus = "+str(regressor11.coef_[0]*1e-9)+"GPa")
```

```
young's modulus = 119.00873836841923GPa
```

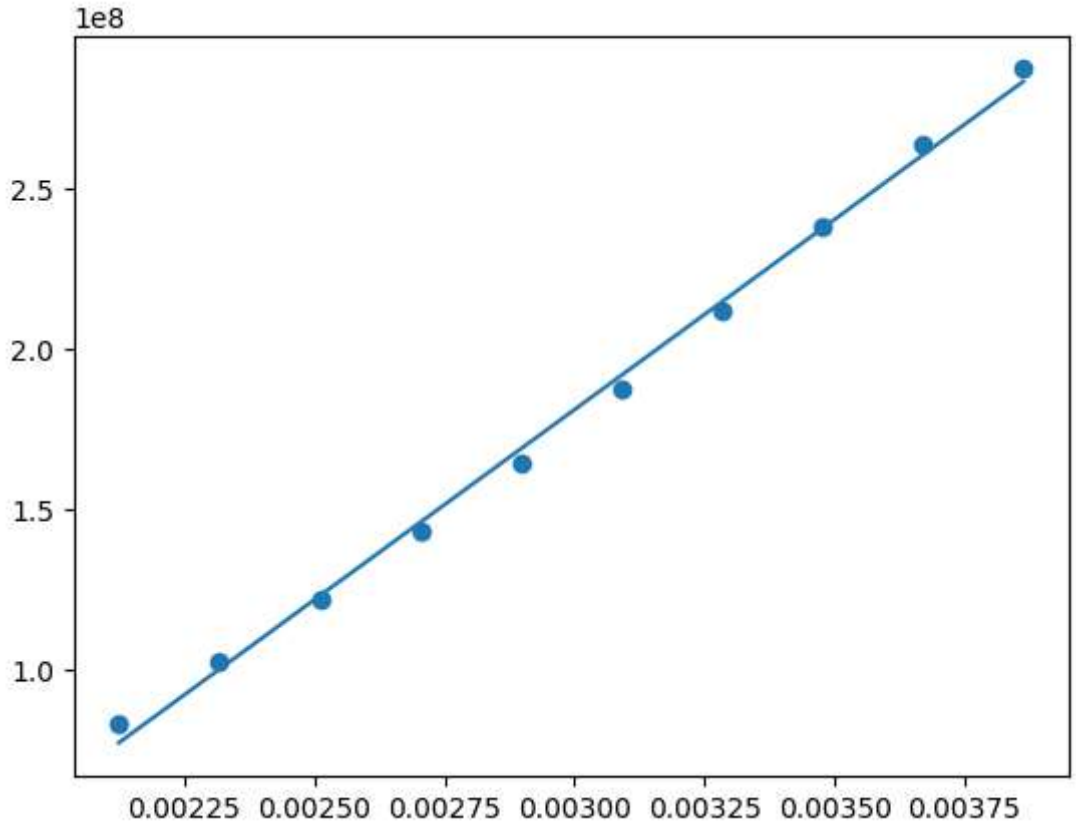
plot

In [ ]:
```
plt.scatter(X1,y1)
plt.plot(X1,ypredict1)
plt.xticks()
plt.yticks()
plt.show()
```



In [ ]:
```
import matplotlib.pyplot as plt
plt.scatter(X11,y11)
```

```
plt.plot(X11,ypredict11)
plt.xticks()
plt.yticks()
plt.show()
```



In [ ]:
```
model11 = sm.OLS(y11, arr11).fit()
model11.summary()
```

```
c:\Program Files\Python310\lib\site-packages\scipy\stats\_stats_py.py:1769: UserWarni
ng: kurtosistest only valid for n>=20 ... continuing anyway, n=10
  warnings.warn("kurtosistest only valid for n>=20 ... continuing "
```

Out[ ]:

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.997
Model:	OLS	**Adj. R-squared:**	0.997
Method:	Least Squares	**F-statistic:**	2881.
Date:	Mon, 10 Oct 2022	**Prob (F-statistic):**	1.61e-11
Time:	23:05:32	**Log-Likelihood:**	-164.81
No. Observations:	10	**AIC:**	333.6
Df Residuals:	8	**BIC:**	334.2
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
x1	1.19e+11	2.22e+09	53.673	0.000	1.14e+11	1.24e+11
const	-1.759e+08	6.75e+06	-26.061	0.000	-1.91e+08	-1.6e+08

Omnibus:	2.328	**Durbin-Watson:**	0.421
Prob(Omnibus):	0.312	**Jarque-Bera (JB):**	0.915
Skew:	0.189	**Prob(JB):**	0.633
Kurtosis:	1.567	**Cond. No.**	1.80e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.8e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

In [ ]: