```python
(you can use, for example, a loop). In addition, do not use the scaled data set for prediction
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from scipy import stats
df=pd.read_csv('materials.csv')
print(df)
y=np.array(df['Strength'])
x=np.array(df.loc[:,'Time':'Temperature'])
#print(x)
xScaled = StandardScaler().fit_transform(x)
#print(xScaled)
li=[]
for i in range(3):
    slope, intercept, r, p, std_error = stats.linregress(xScaled[:,i], y)
    #print(xScaled[:,i])
    li.append(r)
li=np.array(li)
li=abs(li)
print(li)
max=li.argmax()
print(f'most important feature is: {df.columns[max+1]}') #+1 because index values include the response variable
reg=LinearRegression()
reg.fit(x,y)
c=np.array(reg.coef_)
print(f'coefficients: {c}')
yIntercept=reg.intercept_
print(f'Intercept: {yIntercept}')
p1=np.array([32.1, 37.5, 128.95])
p2=np.array([36.9, 35.37, 130.03])
predict1=0
predict2=0
for i in range(len(p1)):
    predict1+=c[i]*p1[i]
    predict2+=c[i]*p2[i]
predict1+=yIntercept
predict2+=yIntercept
print(f'predictions with given data are\n{predict1}\n{predict2}')
```

```
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/
[0.10235205 0.11209137 0.84883697]
most important feature is: Temperature
coefficients: [ 2.12474546  5.31846906 -3.01654815]
Intercept: 389.1659157434116
predictions with given data are
267.82895127641154
263.441518393183
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython %
```

```
the same graph) create a 3D scatter plot between the same ind
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from scipy import stats
import matplotlib.pyplot as plt

df=pd.read_csv('materials.csv')
#print(df)
y=np.array(df['Strength'])
x=np.array(df.loc[:,'Pressure':'Temperature'])

reg=LinearRegression()
reg.fit(x,y)
c=np.array(reg.coef_)
yIntercept=reg.intercept_
#print(x[:,0])
X1, X2 = np.meshgrid(x[:,0], x[:,1])
print(yIntercept)
Z = yIntercept + c[0]*X1 + c[1]*X2
#print(Z)
#3D plot
fig = plt.figure()
ax = plt.axes(projection = '3d')
ax.plot_wireframe(X1, X2, Z, color = 'blue')
#3D scattet plot (data points)
ax.scatter3D(x[:,0], x[:,1], y, c=y, cmap='Greens')
ax.set_title('3D Graph')
plt.show()
```

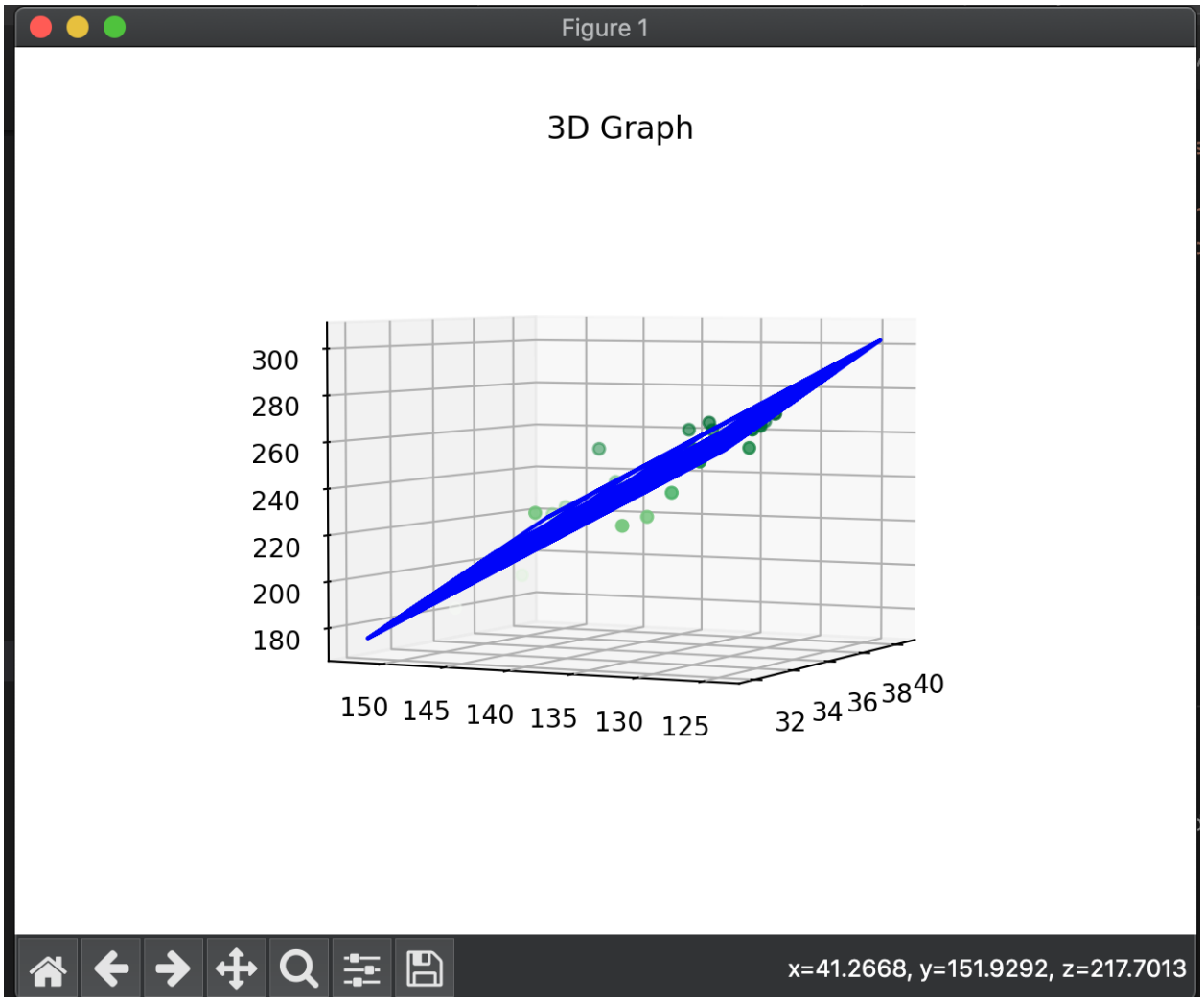
Figure 1
3D Graph
300
280
260
240
220
200
180
150 145 140 135 130 125
32 34 36 38 40
x=41.2668, y=151.9292, z=217.7013

```python
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from functools import reduce
df=pd.read_csv('materialsOutliers.csv')
x=np.array(df['Strength'])
x=x.reshape(-1,1)
y1=np.array(df['Time'])
y2=np.array(df['Pressure'])
y3=np.array(df['Temperature'])
# Robustly fit linear model with RANSAC algorithm
ransac = linear_model.RANSACRegressor(residual_threshold=15,stop_probability=1.00)
ransac.fit(x, y1)
inlier_mask1 = ransac.inlier_mask_
outlier_mask1 = np.logical_not(inlier_mask1)
#print(outlier_mask1)
res=[]
res.append(list(filter(lambda i: outlier_mask1[i],range(len(outlier_mask1)))))
ransac.fit(x, y2)
inlier_mask2 = ransac.inlier_mask_
outlier_mask2 = np.logical_not(inlier_mask2)
res.append(list(filter(lambda i: outlier_mask2[i],range(len(outlier_mask2)))))
ransac.fit(x, y3)
inlier_mask3 = ransac.inlier_mask_
outlier_mask3 = np.logical_not(inlier_mask3)
res.append(list(filter(lambda i: outlier_mask3[i],range(len(outlier_mask3)))))
resFlat= reduce(lambda a,b: a+b, res)
df=df.drop(df.index[resFlat])
print(df)
y=np.array(df['Strength'])
x=np.array(df.loc[:,'Time':'Temperature'])
reg=LinearRegression()
reg.fit(x,y)
c=np.array(reg.coef_)
print(f'coefficients: {c}')
yIntercept=reg.intercept_
print(f'Intercept: {yIntercept}')
```

inlier_mask_: Any

```
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/HW5_3ML.py
s/raulrodriguez/Documents/WorkSpaceVSPython/HW5_3ML.py
coefficients: [ 2.12474546  5.31846906 -3.01654815]
Intercept: 389.1659157434116
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython %
```