In [1]:
```python
# importing all neccessary packages
import numpy as np
import pandas as pd
from pandas import DataFrame, Series
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
import statsmodels.api as sm
from sklearn.ensemble import RandomForestRegressor
from sklearn.cluster import KMeans
import scipy.cluster.hierarchy as sch
```

Question 1:

(a) Load the neurons group 1.csv dataset into Python as a pandas DataFrame. (b) Inspect the data. How many neurons are included in this dataset? How many different measurements are included? Does this dataset contain any missing values? (c) Perform an exploratory data analysis, creating both numerical and graphical summaries of the data. Discuss and interpret your results

In [2]:
```python
# Loading the first neurons data into pandas
neurons_1 = pd.read_csv("neurons_group_1.csv")
# Printing the first few rows
neurons_1.head()
```

Out[2]:

	id	average_diameter	overall_depth	overall_height	overall_width	soma_surface	total_
0	484775243	0.195628	90.3529	548.798070	257.109717	128.269219	3658.
1	485996843	0.457635	87.0383	717.408343	199.214267	430.635072	4158.
2	486041253	0.295455	75.3286	584.083922	386.076695	502.033948	2667.
3	491119181	0.414033	89.0718	284.641670	239.492610	383.828302	1543.
4	491119245	0.201323	44.5237	302.038542	323.493562	120.229052	1621.

In [3]:
```python
# Checking the number of neurons included in the first dataset
neurons_1.shape
```

Out[3]:
```
(311, 9)
```

> This dataset includes 311 neurons and 8 different measurements.

In [4]:
```python
# Checking if there are missing values in the first nueron dataset
neurons_1.isnull().sum()
```

Out[4]:
```
id                  0
average_diameter    0
overall_depth       0
overall_height      0
overall_width       0
soma_surface        0
total_length        0
total_surface       0
total_volume        0
dtype: int64
```

> There are no missing values in the dataset.

In [5]:
```
# Computing the summary statistics
neurons_1.describe()
```

Out[5]:

	id	average_diameter	overall_depth	overall_height	overall_width	soma_surface
count	3.110000e+02	311.000000	311.000000	311.000000	311.000000	311.000000
mean	5.885866e+08	0.421175	91.967024	523.516774	320.548089	361.849689
std	8.490396e+07	0.159435	35.658760	299.256684	124.039953	253.912293
min	4.847752e+08	0.053899	22.680000	82.836871	49.173247	2.895610
25%	4.961239e+08	0.322128	64.574450	328.964661	233.480547	176.194233
50%	5.912744e+08	0.415613	86.212200	469.151885	291.532148	311.568275
75%	6.568502e+08	0.527572	115.566350	651.885452	392.534334	478.796933
max	8.460831e+08	1.156730	183.960000	1928.118350	827.752239	1283.720986

- The average diameter has a mean of 0.421 and a standard deviation of 0.16. The median(0.415) is approximately equal to the mean, which implies the distribution for average diameter is symmetric.
- The overall depth of the neurons has a mean of 91.96 and a standard deviation of 35.65. The median(86.21) is lower than its mean inidicates that the distribution is right skewed.
- The mean of the overall height is 523.52 and it has a standard deviation of 299.26. The median(469.15) is lower than the mean indicates that the distribution is also right skewed.
- The overall width has a mean of 320.55 and a standard deviation of 124.04. The median(291.53) is lower than the mean and this implies that the distribution is right skewed.
- The soma surface has a mean of 361.85 and a standard deviation of 253.91. The median(311.57) is significantly lower than its mean indicates that the distribution for soma surface is right skewed.
- The mean for the total lenght of neurons is 3792.4 and it has a standard deviation of 2775.993. The median (2946.85) is significantly lower than its mean indicates that the

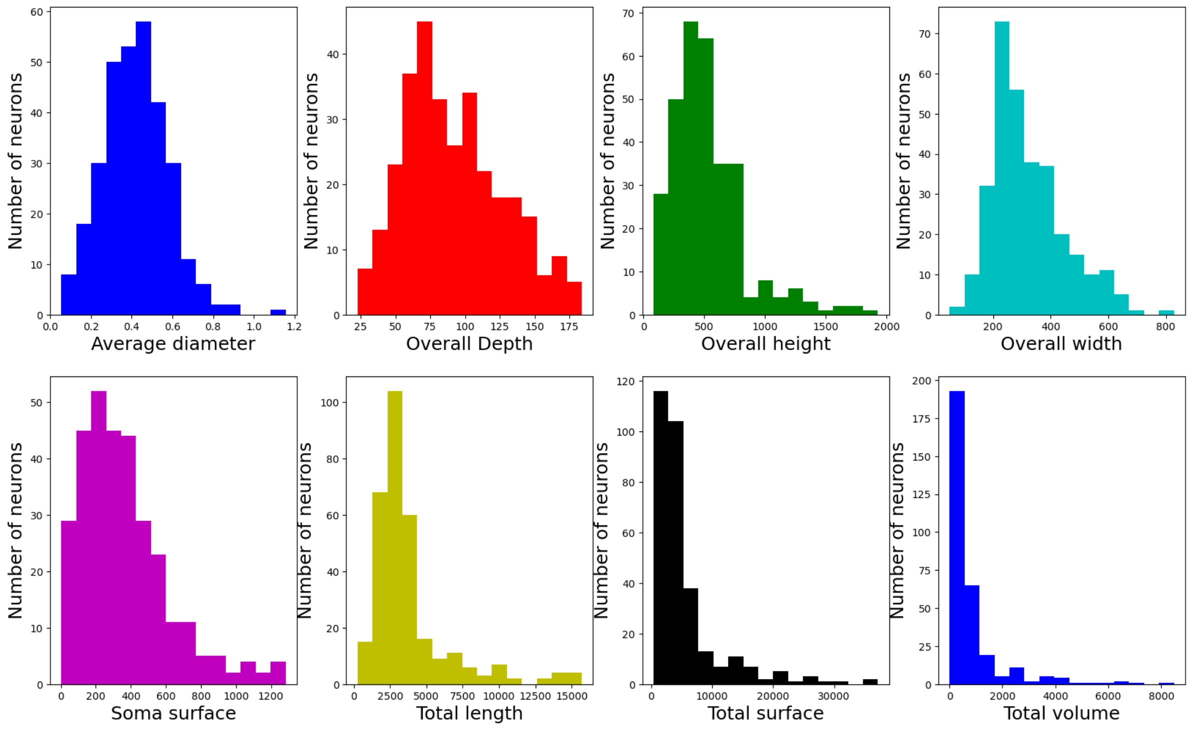
distribution for total lenght is strong right skewed.

- The total_surface has a mean of 5492.74 and a standard deviation of 5719.71. The meadian(3636.75) is lower than the mean implies that the distribution is right skewed and it is a very strong positive skewed distribution because the standard deviation is higher than the mean.

- The total volume has a mean of 808.48 and a standard deviation of 1190.46. The median (407.18) is lower than the mean implies that the distribution is right skewed and it is also has a very strong positive skewed distribution.because the standard deviation is higher than the mean.

In [6]:
```
# Creating a numerical summary for each of the morphological measurements.
fig = plt.figure(figsize=(20,12))

plt.subplot(2,4,1)
plt.hist(neurons_1.average_diameter, color='b', bins=15)
plt.xlabel('Average diameter',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,2)
plt.hist(neurons_1.overall_depth, color='r', bins=15)
plt.xlabel('Overall Depth',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,3)
plt.hist(neurons_1.overall_height, color='g', bins=15)
plt.xlabel('Overall height',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,4)
plt.hist(neurons_1.overall_width, color='c', bins=15)
plt.xlabel('Overall width',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,5)
plt.hist(neurons_1.soma_surface, color='m', bins=15)
plt.xlabel('Soma surface',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,6)
plt.hist(neurons_1.total_length, color='y', bins=15)
plt.xlabel('Total length',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,7)
plt.hist(neurons_1.total_surface, color='k', bins=15)
plt.xlabel('Total surface',size=18)
plt.ylabel('Number of neurons',size=18)

plt.subplot(2,4,8)
plt.hist(neurons_1.total_volume, color='b', bins=15)
plt.xlabel('Total volume',size=18)
plt.ylabel('Number of neurons',size=18)
```

Out[6]:
```
Text(0, 0.5, 'Number of neurons')
```



- As expected from the numerical summaries, the average diameter is relatively a symmetric distribution.The graphs comfirmed that the overall depth, overall height, overall width, soma surface , total length, total surface and total volume has a right skewed distribution with total length, total surface and total volume having a strong positive skewed distribution.

- Also, The centre's of each of these graphes for each measurement matches the mean and median, and the range matches the min and max values.

We could derive the following from their positive skewed distribution.

- more than 40 of the neurons has their overall depth approximately between 70 and 85
- more than 60 neurons has their overall height approximately between 450 and 500
- more than 70 neurons has thier overall widhth approximately between 250 and 300
- more than 50 neurons has their soma surface approximately between 200 and 250
- more than 100 neurons has their total length approximately between 2500 and 3000
- more than 100 neurons has their total surface approximately between 0 and 7500
- more than 175 neurons has their total volume appproximately between 0 and 500.

Question 2

(a) Load the neurons group 2.csv dataset into Python as a pandas DataFrame. (b) Inspect the data. How many neurons are included in this dataset? Are the measurements the same as those in neurons group 1.csv? (c) Perform a t-test, for each of the measurements, to test whether any of the neuron properties differ between the group 1 and group 2. Use a significance level of α =0.01. Display the t-score and p-value for each measurement. Clearly state the conclusion of your tests and explain your reasonining

In [7]:
```
# Loading the neurons group 2 into pandas
neurons_2 = pd.read_csv("neurons_group_2.csv")
```

```
# Printing the first few rows
neurons_2.head()
```

Out[7]:

	id	average_diameter	overall_depth	overall_height	overall_width	soma_surface	total_
0	397905347	0.316091	117.5429	585.602322	287.122628	268.777679	3498.(
1	491119234	0.331268	81.9012	461.280515	275.146120	551.788645	2008.3
2	491119269	0.139015	57.5697	324.422347	280.851229	50.092109	1774.2
3	491119394	0.230412	76.0357	368.298267	251.377567	244.457685	1650.1
4	491119419	0.321163	98.8344	417.890620	193.590563	252.423672	2066.3

In [8]:
```
# Inspecting the data
neurons_2.shape
```

Out[8]:
```
(390, 9)
```

> This dataset includes 390 neurons and 8 different measurements. It includes same measurements as the group 1 neuron measurements

In [9]:
```
# Performing a t-test for the measurements in group 1 and group 2
stats.ttest_ind(neurons_1.drop("id", axis = 1), neurons_2.drop("id", axis = 1))
```

Out[9]:
```
Ttest_indResult(statistic=array([-1.19700758, -1.18572309,  0.58603484, -0.9231920
4, -0.54779764,
       -0.33259965, -0.81523978, -0.93137355]), pvalue=array([0.2317094 , 0.236134
58, 0.55804127, 0.35622569, 0.58400573,
       0.73953623, 0.41521275, 0.35198193]))
```

For each of the measurement: Null hpothesis is there is no difference between the measurement in group 1 and group 2. While the alternative hypothesis is that there is a diffrence between the measurement in neuron group 1 and neuron group 2.

- At significance level of α =0.01, there is no difference between the average diameter for group 1 and group 2. This is because the p-value(0.2317094) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the overall depthf for group 1 and group 2. This is because the p-value(0.23613458) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the overall height for group 1 and group 2. This is because the p-value(0.55804127) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the overall width for group 1 and group 2. This is because the p-value(0.35622569) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the soma surface for group 1 and group 2. This is because the p-value(0.58400573) is greater than α =0.01, so I refuse to reject the null hypothesis.

- At significance level of α =0.01, there is no difference between the total length for group 1 and group 2. This is because the p-value(0.73953623) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the total surface for group 1 and group 2. This is because the p-value(0.41521275) is greater than α =0.01, so I refuse to reject the null hypothesis.
- At significance level of α =0.01, there is no difference between the total volume for group 1 and group 2. This is because the p-value(0.35198193) is greater than α =0.01, so I refuse to reject the null hypothesis.

In conclusion, none of the neuron properties differ between the group 1 and group 2

Question 3: (a) Load the neurons additional measurements.csv into Python and combine all three datasets into a single DataFrame. (b) Comment on the dimensions of the combined dataset. Are all of the neurons from group 1 and 2 included in the dataset neurons additional measurements.csv? (c) Compute the Pearson correlation coefficient between each of the measurements and identify which morphological features are strongly correlated. List the four most strongly correlated pairs. (d) Create scatter plots for the each of the strongly correlated pairs identified in (c). Are the relationships as expected from the correlation coefficients?

In [10]:
```
# Loading the neurons additional measurements
neurons_add = pd.read_csv("neurons_additional_measurements.csv")
# Appending group 1 and group 2 dataset
neurons_df1 = neurons_1.append(neurons_2)
# combining all three datasets into a single dataframe.
neurons_df2 = pd.merge(neurons_add,neurons_df1,left_on="id", right_on="id" )
# Printing out few line from the new dataframe
neurons_df2.head()
```

```
C:\Users\sofiy\AppData\Local\Temp\ipykernel_22692\2540933181.py:4: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a future
version. Use pandas.concat instead.
  neurons_df1 = neurons_1.append(neurons_2)
```

Out[10]:

	id	average_bifurcation_angle_local	average_contraction	average_fragmentation	averag
0	491119743	82.727781	0.864267	20.723077	
1	546781359	82.506680	0.903890	105.277778	
2	537042261	77.536678	0.863104	73.666667	
3	689123605	76.583222	0.900537	95.979167	
4	657879305	72.019250	0.873518	47.535714	

5 rows × 21 columns

In [11]:
```
# Getting the dimension of the combined data
neurons_df2.shape
```

Out[11]:
```
(694, 21)
```

Not all th nuerons from group 1 and 2 are included in the additional dataset (neurons additional measurements.csv)

In [12]:
```
# Computing the Pearson correlation coefficient between each of the measurements
neurons_df2.drop('id', axis = 1).corr()
```

Out[12]:

	average_bifurcation_angle_local	average_contraction	average_fragm
average_bifurcation_angle_local	1.000000	-0.256651	-
average_contraction	-0.256651	1.000000	-
average_fragmentation	-0.033428	-0.232868	
average_parent_daughter_ratio	0.144248	-0.196380	
max_branch_order	-0.057797	-0.055551	
max_euclidean_distance	-0.160161	-0.038668	
max_path_distance	-0.138639	-0.095109	
number_bifurcations	-0.104305	0.054079	-
number_branches	-0.107519	0.059078	-
number_nodes	-0.139077	-0.067872	
number_stems	-0.088756	0.093060	-
number_tips	-0.110187	0.063447	-
average_diameter	0.039447	-0.262180	
overall_depth	-0.067234	-0.168192	
overall_height	-0.170636	-0.011630	
overall_width	-0.088734	-0.070939	
soma_surface	-0.062920	-0.049990	
total_length	-0.143934	-0.064172	
total_surface	-0.124021	-0.136860	
total_volume	-0.113610	-0.168131	

In [13]:
```
# Identifing the morphological features are strongly correlated
corr_matrix = neurons_df2.drop('id', axis = 1).corr()
sol = (corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(bool))
                  .stack()
                  .sort_values(ascending=False))
sol.head(10)
```

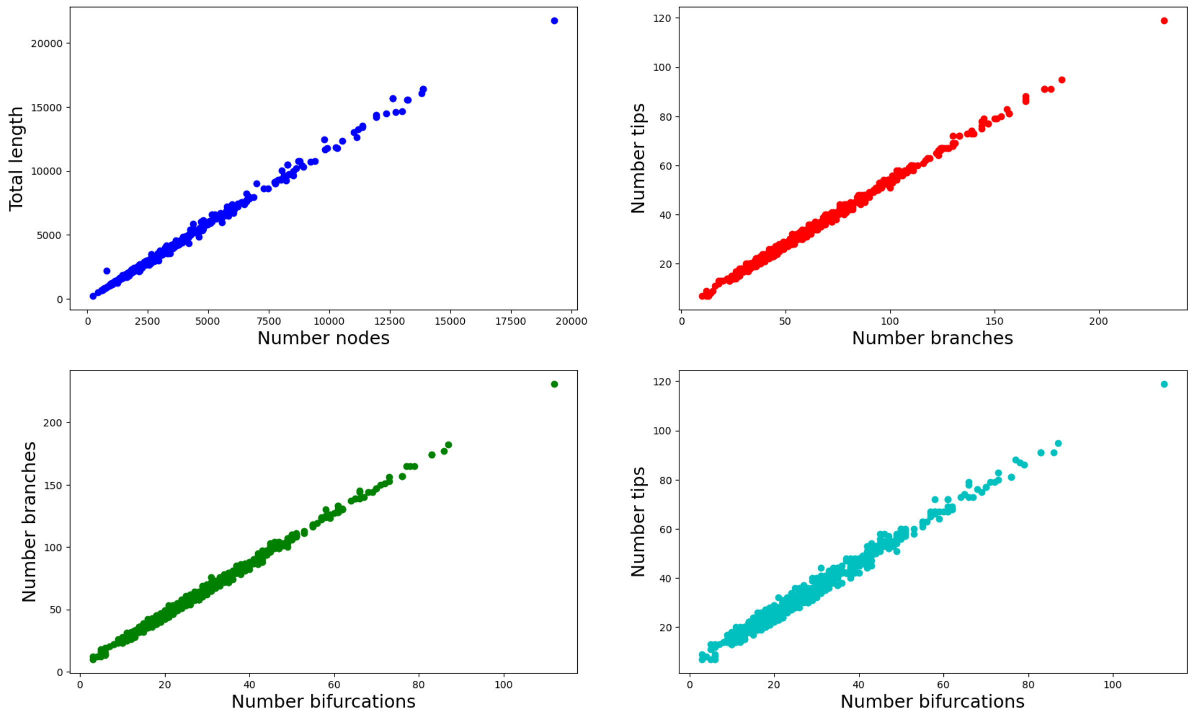
Out[13]:
```
number_nodes           total_length         0.998333
number_branches        number_tips          0.998230
number_bifurcations    number_branches      0.997993
                       number_tips          0.992465
max_euclidean_distance max_path_distance    0.989598
total_surface          total_volume         0.958704
max_euclidean_distance overall_height       0.954598
max_path_distance      overall_height       0.941420
total_length           total_surface        0.913416
number_nodes           total_surface        0.909903
dtype: float64
```

The four most strongly correlated pairs are:

- number_nodes and total_length
- number_branches and number_tips
- number_bifurcations and number_branches
- number_bifurcations and number_tips

In [14]:
```
# Creating scatter plots for the each of the strongly correlated pairs identified 

fig = plt.figure(figsize=(20,12))

plt.subplot(2,2,1)
plt.scatter(neurons_df2.number_nodes, neurons_df2.total_length, color='b')
plt.xlabel('Number nodes',size=18)
plt.ylabel('Total length',size=18)

plt.subplot(2,2,2)
plt.scatter(neurons_df2.number_branches, neurons_df2.number_tips, color='r')
plt.xlabel('Number branches',size=18)
plt.ylabel('Number tips',size=18)

plt.subplot(2,2,3)
plt.scatter(neurons_df2.number_bifurcations, neurons_df2.number_branches, color='g
plt.xlabel('Number bifurcations',size=18)
plt.ylabel('Number branches',size=18)

plt.subplot(2,2,4)
plt.scatter(neurons_df2.number_bifurcations, neurons_df2.number_tips, color='c')
plt.xlabel('Number bifurcations',size=18)
plt.ylabel('Number tips',size=18)
```

Out[14]:
```
Text(0, 0.5, 'Number tips')
```



Yes, All the relationships as expected from the correlation coefficients.

Question 4 Linear regression to predict the total surface area of a neuron (total surface). (Remaining morphological measurements to be used as predictor variables.) (a) Separate the data into response and predictor variables and standardise the predictor variables. (b) Fit a linear regression model and interpret the fitted model. (c) Perform a forward selection Akaike Information Criterion (AIC) regression. Examine the selected model and discuss your findings in relation to the model fitted in part (b). (d) Perform a forward selection Bayes Information Criterion (BIC) regression. Examine the selected model and discuss your findings in relation to the models fitted in part (b) and (c). (e) Explain how using BIC for model selection differs from using AIC.

In [15]:
```
# Separating the data into response and predictor variables
y = neurons_df2.total_surface
neuron_numeric = neurons_df2.drop(['id','total_surface'],axis = 1)
# Standardising the predictor variables
neuron_std = (neuron_numeric - neuron_numeric.mean())/neuron_numeric.std()
```

In [16]:
```
# Fitting a linear regression model
neuron_std.insert(0,'intercept',1)
X = neuron_std
mod = sm.OLS(y,X).fit()
#res = mod.fit()
print(mod.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:          total_surface   R-squared:                       0.992
Model:                            OLS   Adj. R-squared:                  0.991
Method:                 Least Squares   F-statistic:                     4465.
Date:                Mon, 26 Jun 2023   Prob (F-statistic):               0.00
Time:                        15:38:56   Log-Likelihood:                -5362.3
No. Observations:                 694   AIC:                         1.076e+04
Df Residuals:                     675   BIC:                         1.085e+04
Df Model:                          18                                         
Covariance Type:            nonrobust                                         
==================================================================================
=================
                                      coef    std err          t      P>|t|      
[0.025      0.975]
----------------------------------------------------------------------------------
-----------------
intercept                        5705.7931     21.123    270.127      0.000    566
4.319    5747.267
average_bifurcation_angle_local    -5.6851     22.755     -0.250      0.803     -5
0.365      38.994
average_contraction                32.2640     26.690      1.209      0.227     -2
0.141      84.669
average_fragmentation             -95.7414     49.252     -1.944      0.052    -19
2.446       0.963
average_parent_daughter_ratio      54.8073     24.030      2.281      0.023      
7.624     101.990
max_branch_order                  -26.1737     42.612     -0.614      0.539    -10
9.841      57.494
max_euclidean_distance            636.0267    188.029      3.383      0.001     26
6.835    1005.218
max_path_distance                -346.6786    181.393     -1.911      0.056    -70
2.841       9.484
number_bifurcations             -3895.7825   6022.607     -0.647      0.518   -1.5
7e+04    7929.514
number_branches                  1985.8946   3204.822      0.620      0.536   -430
6.724    8278.513
number_nodes                     -813.9945    408.079     -1.995      0.046   -161
5.251     -12.738
number_stems                     -458.2155    620.789     -0.738      0.461   -167
7.125     760.694
number_tips                      1897.7216   3070.338      0.618      0.537   -413
0.840    7926.283
average_diameter                  693.1540     36.058     19.223      0.000     62
2.354     763.953
overall_depth                    -130.8024     31.030     -4.215      0.000    -19
1.730     -69.875
overall_height                   -296.8748     90.839     -3.268      0.001    -47
5.235    -118.514
overall_width                      25.8528     38.994      0.663      0.508     -5
0.712     102.418
soma_surface                       56.9857     28.512      1.999      0.046      
1.004     112.968
total_length                     3956.4533    411.041      9.625      0.000    314
9.380    4763.526
total_volume                     2994.2462     48.152     62.183      0.000    289
9.701    3088.792
==============================================================================
Omnibus:                      611.210   Durbin-Watson:                   1.998
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            32896.241
Skew:                          -3.626   Prob(JB):                         0.00
Kurtosis:                      35.940   Cond. No.                     1.53e+16
==============================================================================
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
[2] The smallest eigenvalue is 2.51e-29. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

- R-squared obtained(0.992) implies that the all the predictive varaibles explains 99% of the varaibility in total surface area of the neurons, and the since the value is close to 1 then the regressions explains the entire variability of the total surface.
- The regression coefficient for the intercept is equal to 5705.7931, which the total surface for any neuron on average is 5705.7931, without any of the predictor varaibles in consideration.
- Using α =0.05 as the significance level, average_bifurcation_angle_local,average_parent_daughter_ratio, max_euclidean_distance, number_nodes, average_diameter, overall_depth, overall_height, soma_surface,total_length, total_volume has a statistically significant relationship with the response variable(total_surface)

In [17]:
```python
def forwardAIC(X,y):
    # The formular below fits a linear model with just the intercept with no varail
    mod = sm.OLS(y, X.iloc[:,0]).fit()
    # recording the aic for the current model
    best_aic = mod.aic

    # Setting a boolean for the bad model
    bad_model = True
    chosen_vars = [0]
    # This equation includes the remaining varaible in the model
    remaining_vars = range(1,X.shape[1])
    while(bad_model):
        curr_aic = np.empty(len(remaining_vars))
        curr_aic_diff = np.empty(len(remaining_vars))
        for count, i in enumerate(remaining_vars):
            curr_vars = np.append(chosen_vars,i)
            curr_mod = sm.OLS(y, X.iloc[:,curr_vars]).fit()
            curr_aic[count] = curr_mod.aic
            curr_aic_diff[count] = curr_mod.aic - best_aic
            # makes all aic equls to zero a good model
        if len(remaining_vars)==0:
            bad_model=False
            # makes all aic greater than zero a good model
        elif np.min(curr_aic_diff)>0:
            bad_model = False
        else:
            best_var = remaining_vars[np.argmin(curr_aic_diff)]
            best_aic = curr_aic[np.argmin(curr_aic_diff)]
            chosen_vars = np.append(chosen_vars,best_var)
            remaining_vars = [x for x in remaining_vars if x != best_var]
    return chosen_vars
```

In [18]:
```python
# Printing out the final chosen variables accordig the lowest AIC value
ans = forwardAIC(X,y)
mod = sm.OLS(y, X.iloc[:,ans])
res = mod.fit()
print(res.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          total_surface   R-squared:                       0.991
Model:                            OLS   Adj. R-squared:                  0.991
Method:                 Least Squares   F-statistic:                     6583.
Date:                Mon, 26 Jun 2023   Prob (F-statistic):               0.00
Time:                        15:38:59   Log-Likelihood:                -5371.2
No. Observations:                 694   AIC:                         1.077e+04
Df Residuals:                     681   BIC:                         1.083e+04
Df Model:                          12
Covariance Type:            nonrobust
==================================================================================
===============
                                   coef    std err          t      P>|t|      [0.
025      0.975]
----------------------------------------------------------------------------------
---------------
intercept                     5705.7931     21.302    267.851      0.000    5663.
967    5747.619
total_volume                  3006.6073     48.140     62.455      0.000    2912.
087    3101.128
total_length                  3738.2780    406.017      9.207      0.000    2941.
083    4535.473
average_diameter               691.1032     35.595     19.416      0.000     621.
215     760.992
overall_depth                 -135.2543     29.395     -4.601      0.000    -192.
970     -77.538
number_stems                   -56.7562     24.052     -2.360      0.019    -103.
980      -9.532
number_nodes                  -598.5320    405.474     -1.476      0.140   -1394.
662     197.598
soma_surface                    48.4343     28.267      1.713      0.087      -7.
067     103.935
average_parent_daughter_ratio   47.3583     23.019      2.057      0.040       2.
161      92.556
average_contraction             38.2642     23.536      1.626      0.104      -7.
948      84.476
number_bifurcations           -212.3105     64.774     -3.278      0.001    -339.
492     -85.129
average_fragmentation         -148.1038     44.649     -3.317      0.001    -235.
770     -60.437
overall_width                   64.7112     35.276      1.834      0.067      -4.
551     133.973
==============================================================================
Omnibus:                      603.992   Durbin-Watson:                   1.989
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            32666.481
Skew:                          -3.553   Prob(JB):                         0.00
Kurtosis:                      35.851   Cond. No.                         60.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
```

- The R-squared obtained(0.991) is similar to the linear model fitted above, it implies that the all the predictive varaibles explains 99% of the varaibility in total surface area of the neurons, and the since the value is close to 1 then the regressions explains the entire variability of the total surface.
- The matrix above returns the a model that the predictive varaible that offers the fit for the model. But, according to this model, the following predictive varaiables are

satistically significant to the model: total_volume, total_length, average_diameter, overall_depth,number_stems, average_parent_daughter_ratio,number_bifurcations, average_fragmentation. For the model fitted before, number_bifurcations and average_fragmentation are not significant to the model.

In [22]:
```
# d)  Minor edits to function given in lecture material
def forwardBIC(X,y):
    # X here must be a matrix with the first column just a constant.
    # The remaining columns should be the explanatory variables
    # y should be a series containing the response variable
    # First, fit a model with just a constant:
    mod = sm.OLS(y, X.iloc[:,0]).fit()
    best_bic = mod.bic
    # Create a while loop to run through the model
    bad_model = True
    # Get the column indices of the chosen vars so far
    chosen_vars = [0]
    # Get the column indices of the remaining vars so far
    remaining_vars = range(1,X.shape[1])
    while(bad_model):
        # Loop through all the remaining vars
        curr_bic = np.empty(len(remaining_vars))
        curr_bic_diff = np.empty(len(remaining_vars))
        for count, i in enumerate(remaining_vars):
            curr_vars = np.append(chosen_vars,i)
            curr_mod = sm.OLS(y, X.iloc[:,curr_vars]).fit()
            curr_bic[count] = curr_mod.bic
            curr_bic_diff[count] = curr_mod.bic - best_bic
        # If the models are better at least one of these should be negative
        if len(remaining_vars)==0:
            bad_model=False
        elif np.min(curr_bic_diff)>0:
            bad_model = False
        else:
            best_var = remaining_vars[np.argmin(curr_bic_diff)]
            best_bic = curr_bic[np.argmin(curr_bic_diff)]
            chosen_vars = np.append(chosen_vars,best_var)
            remaining_vars = [x for x in remaining_vars if x != best_var]
    return chosen_vars
BIC_vars = forwardBIC(neuron_std,y)
print('The variables included in forward selection BIC regression were:')
for j in range(len(BIC_vars)):
    print(neuron_std.columns[BIC_vars][j])
```

```
The variables included in forward selection BIC regression were:
intercept
total_volume
total_length
average_diameter
overall_depth
number_stems
number_nodes
```

In [23]:
```
# Fitting a model with BIC model
mod = sm.OLS(y,neuron_std.iloc[:,BIC_vars])
res = mod.fit()
print(res.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:          total_surface   R-squared:                       0.991
Model:                            OLS   Adj. R-squared:                  0.991
Method:                 Least Squares   F-statistic:                 1.279e+04
Date:                Mon, 26 Jun 2023   Prob (F-statistic):               0.00
Time:                        15:51:35   Log-Likelihood:                -5384.2
No. Observations:                 694   AIC:                         1.078e+04
Df Residuals:                     687   BIC:                         1.081e+04
Df Model:                           6                                         
Covariance Type:            nonrobust                                         
==================================================================================
==
                       coef    std err          t      P>|t|      [0.025      0.97
5]
----------------------------------------------------------------------------------
--
intercept         5705.7931     21.609    264.047      0.000    5663.366    5748.2
21
total_volume      3030.2338     47.786     63.412      0.000    2936.409    3124.0
59
total_length      4017.9089    397.912     10.097      0.000    3236.639    4799.1
79
average_diameter   709.4829     31.482     22.536      0.000     647.670     771.2
96
overall_depth     -168.0593     27.537     -6.103      0.000    -222.126    -113.9
92
number_stems       -67.3998     23.011     -2.929      0.004    -112.580     -22.2
20
number_nodes     -1057.0739    391.404     -2.701      0.007   -1825.565    -288.5
83
==============================================================================
Omnibus:                      577.310   Durbin-Watson:                   1.978
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            28055.491
Skew:                          -3.339   Prob(JB):                         0.00
Kurtosis:                      33.424   Cond. No.                         47.5
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
```

- Standard Errors assume that the covariance matrix of the errors is correctly specified.
- At a significance level of 0.05, all of the variables are significant, which indicates that we have a good model with strong predictive power. The R2 and adjusted R2 value is the same as for the forward selection AIC regression. This model only uses 7 of the 20 variables (6 forward selection AIC model) and still provides as good a fit as the other model. With 7 significant variables we can be confident that the level of over fitting is minimal. As such, we conclude that this is the preferred model.

Explaination on how using BIC for model selection differs from using AIC

> AIC and BIC for model selecion are effective ways for selecting models, BIC estimates a function of the posterior probability of a model being true under some certain bayesian set up, however AIC estimates a constant plus the relative distance between the unknown true likelihood function of the data and the fitted likelihood function of the model. AIC produces result in

> complex traits but BIC produces more finite dimensions and consistent attributes. AIC is better for negative findings while BIC is better for positive findings.

Question 5 Random forest regression to predict the total surface area of a neuron (total surface). (Remaining morphological measurements to be used as predictor variables.) (a) Split the data into appropriate training and test sets. (b) Fit a random forest regression model with 10 trees using the training data. Include the argument random state=101 in the random forest regression function to ensure reproducible results. Determine which variables are most important in predicting the total surface area of a neuron. Discuss your findings in relation to the linear models fit in question 4. (c) Use the random forest regression model to predict the total surface area of a neuron for the test set. Create a scatter plot of the true surface area of a neuron versus the predicted surface area. Interpret your plot. (d) Assess the performance of a random forest regression model with 5, 10, 20, 50, 100, 200, 500 and 1000 trees in predicting the total surface area of a neuron. You should repeat the model fit and prediction 30 times for each number of trees, using a different random state for each repeat. Create a plot of the model performance as a function of the number of trees (use a log axis for the number of trees). The plot should show the mean and standard error of the performance metric for each number of trees. Discuss your findings. (e) Explain the rationale for fitting the model multiple time with different random states.

In [26]:
```
# a)  Splitting the data into appropriate training and test sets
X = neurons_df2.drop(['id','total_surface'], axis = 1)
y = neurons_df2.total_surface
train_size = 521
np.random.seed(425)
train_select = np.random.permutation(range(len(y)))
X_train = X.iloc[train_select[:train_size],:].reset_index(drop=True)
X_test = X.iloc[train_select[train_size:],:].reset_index(drop=True)
y_train = y[train_select[:train_size]].reset_index(drop=True)
y_test = y[train_select[train_size:]].reset_index(drop=True)
```

In [27]:
```
# b)  Fitting a random forest regression model with 10 trees
rf = RandomForestRegressor(n_estimators=10,random_state = 101, max_depth = 5).fit(
```

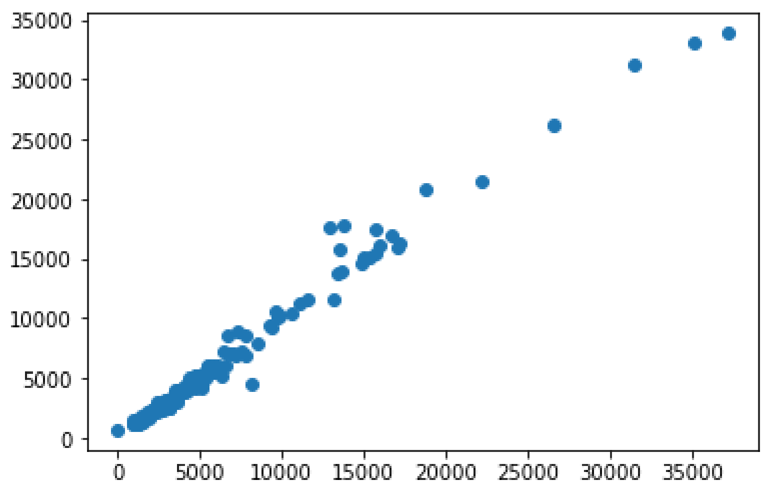
In [21]:
```
rf_test_pred = rf.predict(X_test)
MSE_rf = np.mean(pow((rf_test_pred- y_test),2))
print(MSE_rf)
```

```
601631.4340617411
```

In [133…
```
# c)  Checking which feature are important
rf.feature_importances_
```

Out[133]:
```
array([6.60042103e-04, 4.22304260e-04, 4.51111750e-04, 3.76344007e-03,
       2.55149255e-03, 1.92790494e-04, 1.58828815e-04, 9.69753273e-04,
       1.36863773e-02, 1.98387123e-02, 3.08174505e-05, 3.19269961e-02,
       2.29616210e-03, 1.77552307e-04, 1.61451777e-04, 7.48250650e-04,
       7.82627417e-04, 3.89723063e-02, 8.82208983e-01])
```

In [22]:
```
# Creating a scatter plot of the true surface area and predicted surface area.
plt.scatter(y_test,rf_test_pred)
plt.show()
```



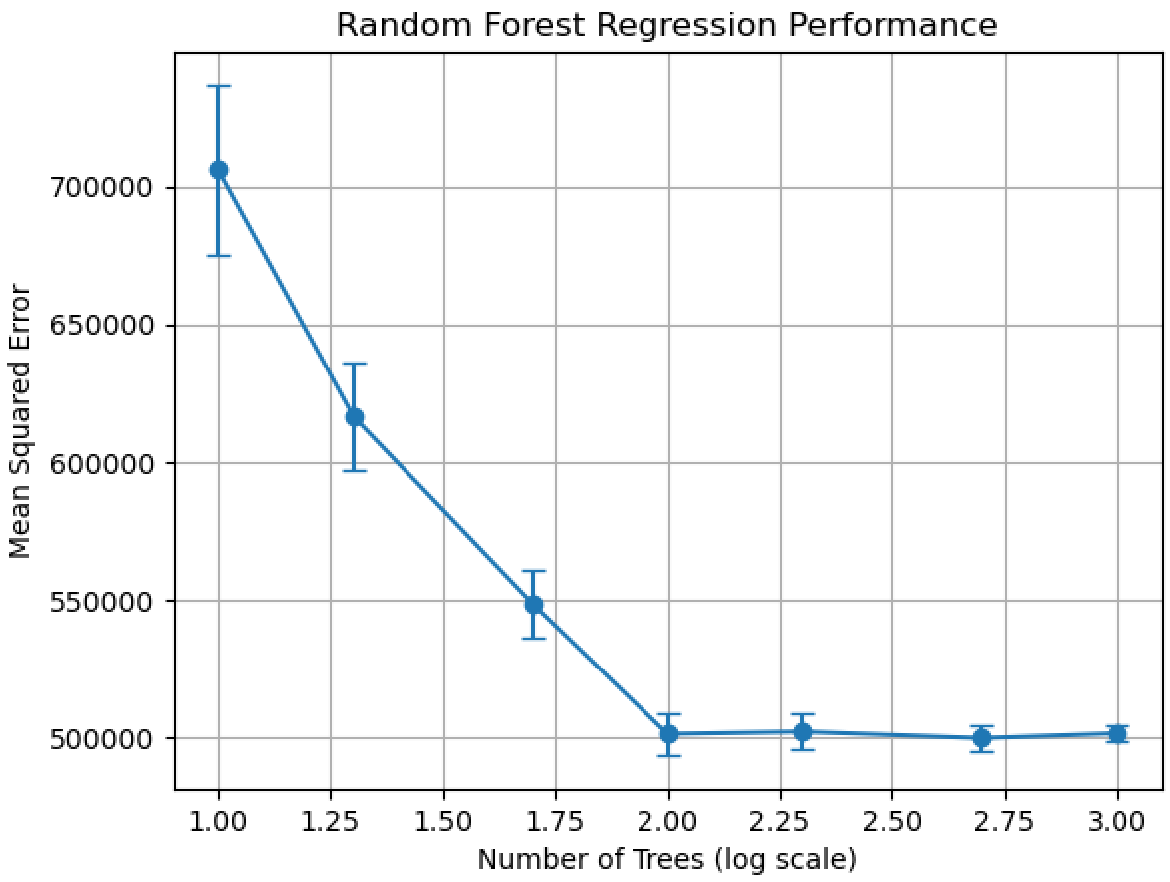
There seem to be a strong positive correlation between the true surface area of a neuron and the predicted surface area.

In [33]:
```
 # Load in packages
import numpy as np
import pandas as pd
from pandas import DataFrame, Series
import numpy.random as npr
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
import statsmodels.api as sm
from sklearn import model_selection
from sklearn.ensemble import RandomForestRegressor
from sklearn.cluster import KMeans
#from yellowbrick.cluster.elbow import kelbow_visualizer
```

In [28]:
```
# d)
trees = [10,20,50,100,200,500,1000]
seeds = np.random.randint(1,999,size=30)
MSE_rf = np.zeros((len(trees),len(seeds)))
for i,j in enumerate(trees):
    for n,m in enumerate(seeds):
        rf = RandomForestRegressor(n_estimators=j, random_state=m)
        rf.fit(X_train, y_train)
        rf_test_pred = rf.predict(X_test)
        MSE_rf[i,n] = np.mean(pow((rf_test_pred- y_test),2))
```

In [29]:
```
# Getting the standard error
standard_error_rf = MSE_rf.std(axis=1)/np.sqrt(len(seeds))
```

In [35]:
```
# Plot
mean_performance = MSE_rf.mean(axis=1)
trees_log = np.log10(trees)

plt.errorbar(trees_log, mean_performance, yerr=standard_error_rf, fmt='o-', capsiz
plt.xlabel('Number of Trees (log scale)')
plt.ylabel('Mean Squared Error')
plt.title('Random Forest Regression Performance')
plt.grid(True)
plt.show()
```



From the plot we can conclude that 100-200 trees is sufficient to predict the surface area of the neuron. We do not get a substantially lower mean square error by including additional trees