```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

# however you like, as long as it is all suitable for the assignment.
def groupScatter2(groups):
   ax = plt.gca()
   for key, group in groups:
      ax.scatter(group.PC1, group.PC2, s=10, label=key, 
color=colors[key], marker=markers[key], alpha=0.5)
   ax.legend()
   plt.xlabel('PC1')
   plt.ylabel('PC2')
   ax.set_xlim(a1,a2)
   ax.set_ylim(a1,a2)

def groupScatter3(groups):
   ax = plt.gca()
   for key, group in groups:
      ax.scatter(group.PC1, group.PC2, group.PC3, s=10, label=key, 
color=colors[key], marker=markers[key], alpha=0.5)
   ax.legend()
   ax.set_xlabel('PC1')
   ax.set_ylabel('PC2')
   ax.set_zlabel('PC3')
   ax.set_xlim3d(a1,a2)
   ax.set_ylim3d(a1,a2)
   ax.set_zlim3d(a1,a2)

colors  = {'Taseko BC':'r', 'Grassy Island BC':'g', 'E. Russia':'b', 
'Greenland':'k',}
markers = {'Taseko BC':'.', 'Grassy Island BC':'.', 'E. Russia':'.', 
'Greenland':'.',}
# Define plot axis limits (don't change these!):
a1 = -10 # DON'T CHANGE THIS
a2 =  10 # DON'T CHANGE THIS

buchia = pd.read_csv('buchia.csv')

# Print some information about the loaded data
print(buchia.head())  # Check the first few rows of the DataFrame
print(buchia.info())
```

```
    Location         J         I       Ld        Lv        Wa   
Wp  \
0  Taseko BC  17.40690 -11.35860  2.30469  2.003470  1.649000  
1.317150
```

```
1  Taseko BC  13.72180  -8.14460  1.39880  0.992295  0.835080
0.723899
2  Taseko BC  10.31080 -12.20650  2.60316  2.248300  1.709080
1.046780
3  Taseko BC  12.07540 -10.83030  2.42738  1.763230  1.236510
0.784577
4  Taseko BC   4.66833  -5.20979  1.54111  1.121260  0.809059
0.559734

         Dd       Dv        In     Delta
0  1.203080  3.24611  0.938409  0.145662
1  0.975138  1.49029  0.588475  0.166572
2  1.446370  3.33082  0.709000  0.195045
3  1.507450  2.80939  0.687856  0.160851
4  0.910708  1.89792  0.487325  0.110510
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1523 entries, 0 to 1522
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Location  1523 non-null   object
 1   J         1523 non-null   float64
 2   I         1523 non-null   float64
 3   Ld        1523 non-null   float64
 4   Lv        1523 non-null   float64
 5   Wa        1523 non-null   float64
 6   Wp        1523 non-null   float64
 7   Dd        1523 non-null   float64
 8   Dv        1523 non-null   float64
 9   In        1523 non-null   float64
 10  Delta     1523 non-null   float64
dtypes: float64(10), object(1)
memory usage: 131.0+ KB
None

A = buchia.drop('Location', axis=1)

scaler = StandardScaler()
X = scaler.fit_transform(A)

pca = PCA()
X_pca = pca.fit_transform(X)

eigenvalues = pca.explained_variance_
print(eigenvalues)

[6.65380637 1.82837722 0.61367128 0.23519443 0.1841071  0.17200264
 0.12283048 0.11132335 0.05095089 0.03430653]

eigenvalue_ratios = eigenvalues / np.sum(eigenvalues)
print(eigenvalue_ratios)
```

```
[0.66494375 0.18271767 0.06132683 0.023504   0.01839862 0.01718897
 0.01227498 0.01112503 0.00509174 0.0034284 ]
```

```
sum_eigenvalue_ratios = np.sum(eigenvalue_ratios)
print(sum_eigenvalue_ratios)
```

```
0.9999999999999998
```

```
print("Eigenvalues")
print(eigenvalues)
print("eigenvalue ratios")
print(eigenvalue_ratios)
print("sum of eigenvalue ratios")
print(sum_eigenvalue_ratios)
# Calculate the percent of total variation explained by the first
three principal components
percent_variance_explained = np.sum(eigenvalue_ratios[:3]) * 100
print("\nPercentage of Total Variation Explained by the First Three
PCs:", percent_variance_explained)
```

```
Eigenvalues
[6.65380637 1.82837722 0.61367128 0.23519443 0.1841071  0.17200264
 0.12283048 0.11132335 0.05095089 0.03430653]
eigenvalue ratios
[0.66494375 0.18271767 0.06132683 0.023504   0.01839862 0.01718897
 0.01227498 0.01112503 0.00509174 0.0034284 ]
sum of eigenvalue ratios
0.9999999999999998

Percentage of Total Variation Explained by the First Three PCs:
90.89882543048904
```

1)The total sum of the eigenvalue ratios is almost 1, but not exactly. This slight difference is because of rounding during the calculation. It's okay to say that the sum is about 1, as the small variation is likely due to how numbers are handled in the calculations.

2)Approximately 90.90% of the data's variability is explained by the first three principal components. This suggests that these three components capture a large portion of the overall patterns in the data. Typically, in analyses like this, people often set a threshold, like 90% or 95%, to decide how many principal components to keep. In this case, the first three components already cover a significant part of the total variation.