```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import normalize
import scipy.cluster.hierarchy as shc
```

```
# write csv into datafile
medical_clean_df = pd.read_csv('medical_clean.csv')
print(medical_clean_df.head())
```

```
   CaseOrder Customer_id  ... Item7 Item8
0          1     C412403  ...     3     4
1          2     Z919181  ...     3     3
2          3     F995323  ...     3     3
3          4     A879973  ...     5     5
4          5     C544523  ...     4     3

[5 rows x 50 columns]
```

```
#creating new datafile with variables to measure
newdf = medical_clean_df[['Income', 'VitD_levels', 'Doc_visits', 'Initial_days', 'TotalCharge
print(newdf.head())
```

```
     Income  VitD_levels  ...  TotalCharge  Additional_charges
0  86575.93    19.141466  ...  3726.702860        17939.403420
1  46805.99    18.940352  ...  4193.190458        17612.998120
2  14370.14    18.057507  ...  2434.234222        17505.192460
3  39741.49    16.576858  ...  2127.830423        12993.437350
4   1209.56    17.439069  ...  2113.073274         3716.525786

[5 rows x 6 columns]
```

```
# checking for duplicated and null values
print(newdf.loc[newdf.duplicated()])
print(newdf.isnull().sum())
```

```
Empty DataFrame
Columns: [Income, VitD_levels, Doc_visits, Initial_days, TotalCharge, Additional_charge
Index: []
Income                0
VitD_levels           0
Doc_visits            0
Initial_days          0
TotalCharge           0
Additional_charges    0
dtype: int64
```

```
# since null values were found, here we are deleting them and writing the new clean data to c
newdf = newdf.dropna()
newdf.to_csv('newdf.csv')

# scaling the data for analysis (hierarchical)
# normalizing will allow for our analysis to not be overly bias towards one varia
scaled_newdf = normalize(newdf)
scaled_newdf = pd.DataFrame(scaled_newdf, columns = newdf.columns)
print(scaled_newdf.head())
```

```
     Income  VitD_levels  ...  TotalCharge  Additional_charges
0  0.978331     0.000216  ...     0.042113            0.202720
1  0.932656     0.000377  ...     0.083554            0.350957
2  0.630865     0.000793  ...     0.106866            0.768497
3  0.949260     0.000396  ...     0.050825            0.310359
4  0.272234     0.003925  ...     0.475587            0.836474

[5 rows x 6 columns]
```

```
# creating a dendrogram to see the clusters that are formed from the normalized c
plt.figure(figsize=(15,12))
plt.title('Hierarchy table')
dend = shc.dendrogram(shc.linkage(scaled_newdf, method='ward'))
plt.axhline(y=10, color= 'r', linestyle='--')
plt.savefig('dendrogram.jpg')
plt.close()
```

```
# importing agglomerative clustering to allow us to create our clusters for easie
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=6, affinity='euclidean', linkage='wa
cluster.fit_predict(scaled_newdf)
```

```
array([3, 1, 4, ..., 3, 1, 3])
```

```
plt.figure(figsize=(15,12))
plt.scatter(scaled_newdf.iloc[:,0], scaled_newdf.iloc[:,1], c=cluster.labels_, cm
plt.savefig('all_factors.jpg')
plt.close()
```

```
ax = plt.subplots(figsize=(12,12))
ax = sns.heatmap(scaled_newdf.corr(), annot=True)
plt.savefig('heatmap.jpg')
plt.close()
```

```
# importing agglomerative clustering to allow us to create our clusters for easier visualizat
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
cluster.fit_predict(scaled_newdf)
```

```
array([1, 1, 0, ..., 1, 1, 1])
```

Note: closer to 0 in normalization shows that the data is closer related, instead of being closer to 1, which means less related.

```
plt.figure(figsize=(15,12))
plt.title('Income v VitD_Levels') #income is x, vitd levels are y
plt.xlabel('Income')
plt.ylabel('VitD_Levels')
plt.scatter(scaled_newdf['Income'], scaled_newdf['VitD_levels'], c=cluster.labels_)
plt.savefig('income_v_vitd.jpg')
plt.close()


plt.figure(figsize=(15,12))
plt.title('Income v Additional_charges') #income is x, addn charges are y
plt.xlabel('Income')
plt.ylabel('Additional Charges')
plt.scatter(scaled_newdf['Income'], scaled_newdf['Additional_charges'], c=cluster.labels_)
plt.savefig('income_v_addn.jpg')
plt.close()


plt.figure(figsize=(15,12))
plt.title('Doc visits v Initial days') #doc visits are x, initial days are y
plt.xlabel('Doc Visits')
plt.ylabel('Initial Days')
plt.scatter(scaled_newdf['Doc_visits'], scaled_newdf['Initial_days'], c=cluster.labels_)
plt.savefig('doc_v_days.jpg')
plt.close()


'''
plt.figure(figsize=(15,12))
plt.title('Income v HighBlood') #income are x, high blood are y
plt.xlabel('Income')
plt.ylabel('High Blood')
plt.scatter(scaled_newdf['Income'], scaled_newdf['HighBlood'], c=cluster.labels_)
plt.savefig('Income_v_HighBlood.jpg')
plt.close()
'''
```

```
'\nplt.figure(figsize=(15,12))\nplt.title('Income v HighBlood') #income are x, high bl
ood are y\nplt.xlabel('Income')\nplt.ylabel('High Blood')\nplt.scatter(scaled_newdf['I
ncome'], scaled_newdf['HighBlood'], c=cluster.labels_)\nplt.savefig('Income_v_HighBloo
d.jpg')\nplt.close()\n'
```

Can we say, that since income is closely clustered around 1.0 in this analysis, that it doesn't have much relation to each other datapoint?

0s completed at 8:30 AM