In [1]:
```python
import numpy as np
import pandas as pd
pd.options.display.max_columns=100
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler as SSc
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
%matplotlib inline

#set width of window to preference
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:90% !important; }</style>"))
```

In [2]:
```python
data = pd.read_csv("Data-Prepped.csv",index_col=0)
data = data.astype(np.float32)
data.head()
```

Out[2]:

	Bronze	Silver	Gold	Platinum	Diamond	Master	GrandMaster	LeagueIndex	Age	HoursPerWe
0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	5.0	27.0	1
1	0.0	0.0	0.0	0.0	1.0	0.0	0.0	5.0	23.0	1
2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	4.0	30.0	1
3	0.0	0.0	1.0	0.0	0.0	0.0	0.0	3.0	19.0	2
4	0.0	0.0	1.0	0.0	0.0	0.0	0.0	3.0	32.0	1

In [3]:
```python
data.describe()
```

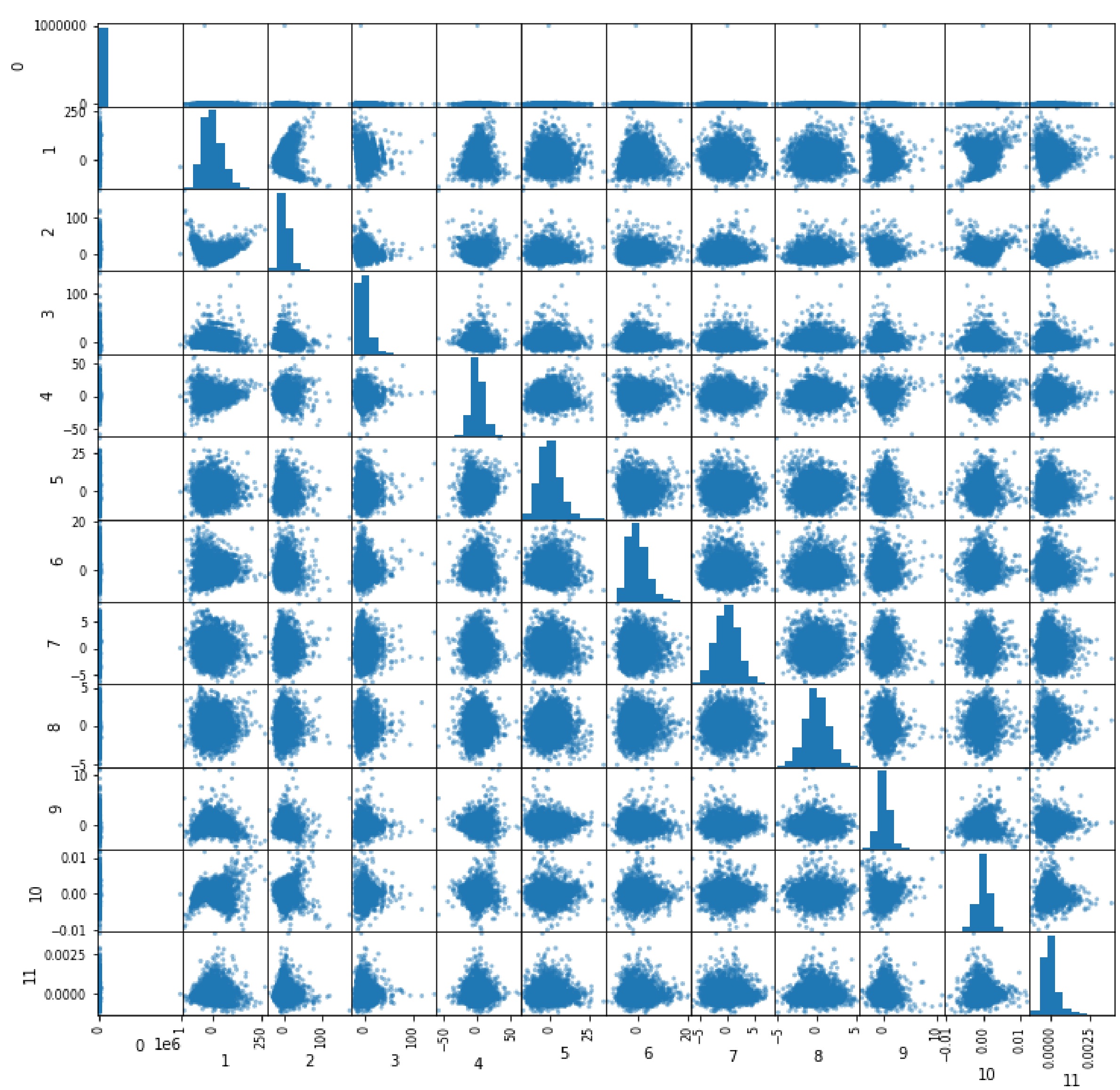
Out[3]:

	Bronze	Silver	Gold	Platinum	Diamond	Master	GrandMaste
count	3338.000000	3338.000000	3338.000000	3338.000000	3338.000000	3338.000000	3338.00000
mean	0.050030	0.103954	0.165668	0.242960	0.240863	0.186040	0.01048
std	0.218039	0.305247	0.371838	0.428935	0.427671	0.389197	0.10187
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
75%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00000

PCA without data standardization

```
In [4]:
X = data.iloc[:,8:]
y = data.iloc[:,7]
pca = PCA(n_components=12)
pca.fit(X)
#print("PCA components:                    "+str(pca.components_))
print("\nPCA explained variance ratio: "+str(pca.explained_variance_ratio_))
print("\nPCA singular values:           "+str(pca.singular_values_))

x = pd.DataFrame(pca.transform(X))
pd.plotting.scatter_matrix(x,figsize=(12,12));
```

```
PCA explained variance ratio: [9.9998921e-01 8.7728649e-06 8.1147618e-07 4.4801
843e-07 3.5246873e-07
 1.4564591e-07 5.3040186e-08 1.5384209e-08 7.3575586e-09 4.9137507e-09
 1.2940993e-14 7.0367040e-16]

PCA singular values:          [1.0004127e+06 2.9631406e+03 9.0119635e+02 6.6962
164e+02 5.9393896e+02
 3.8179538e+02 2.3040083e+02 1.2408496e+02 8.5812073e+01 7.0127457e+01
 1.1380605e-01 2.6537877e-02]
```

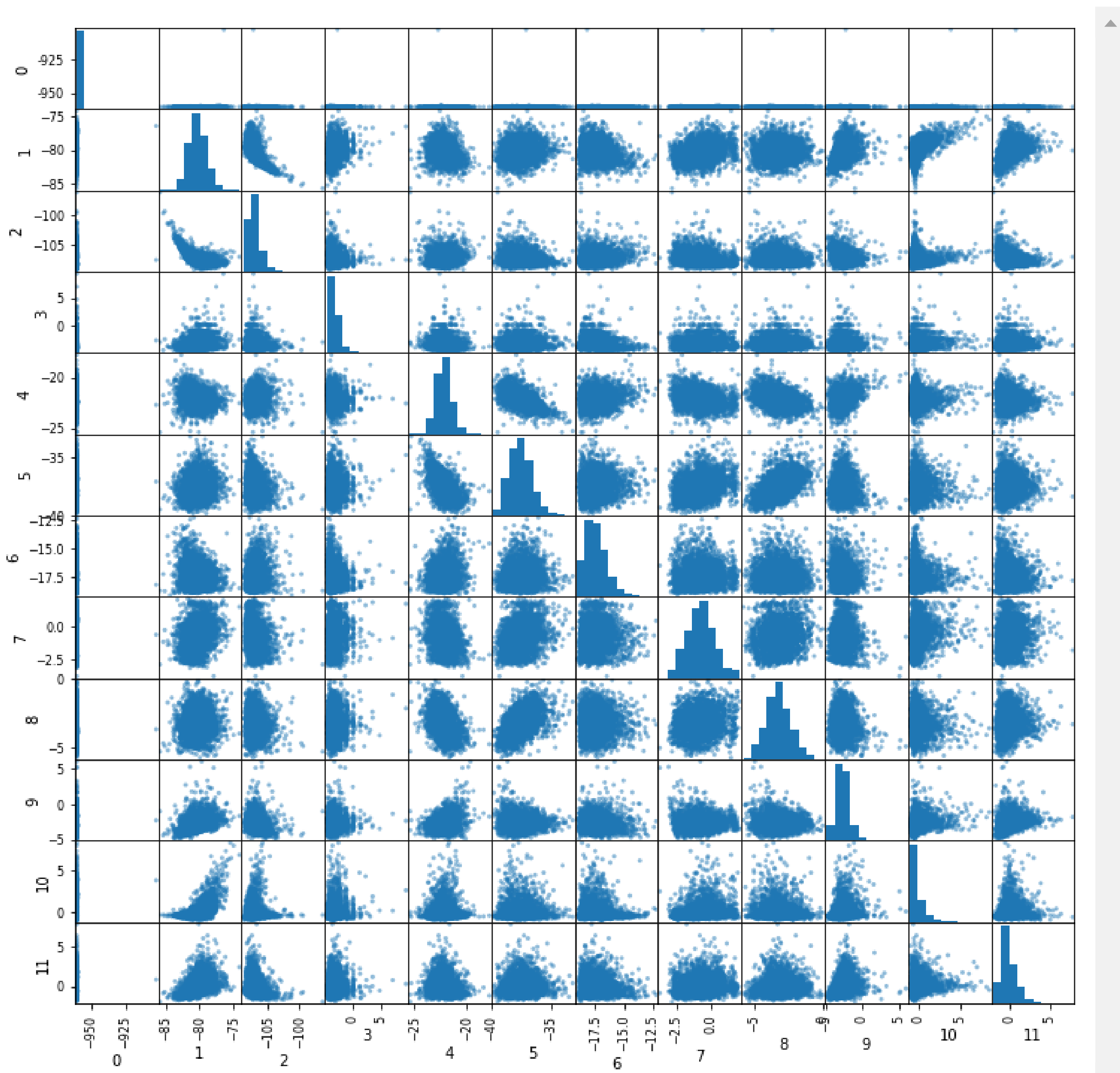


PCA with data standardization

In [5]:

```python
pca = PCA(n_components=12)
pca.fit(X)
#print("PCA components:                "+str(pca.components_))
print("\nPCA explained variance ratio: "+str(pca.explained_variance_ratio_))
print("\nPCA singular values:          "+str(pca.singular_values_))

ssc = SSc()
x = pd.DataFrame(ssc.fit_transform(X))

x = pd.DataFrame(pca.transform(x))
pd.plotting.scatter_matrix(x,figsize=(12,12));
```

```
PCA explained variance ratio: [9.9998921e-01 8.7728895e-06 8.1147579e-07 4.4799
535e-07 3.5246850e-07
 1.4564579e-07 5.3040122e-08 1.5384233e-08 7.3575546e-09 4.9137490e-09
 1.2940999e-14 7.0366955e-16]

PCA singular values:          [1.00041269e+06 2.96314453e+03 9.01196106e+02 6.6
9604370e+02
 5.93938782e+02 3.81795227e+02 2.30400696e+02 1.24085045e+02
 8.58120499e+01 7.01274490e+01 1.13806076e-01 2.65378617e-02]
```



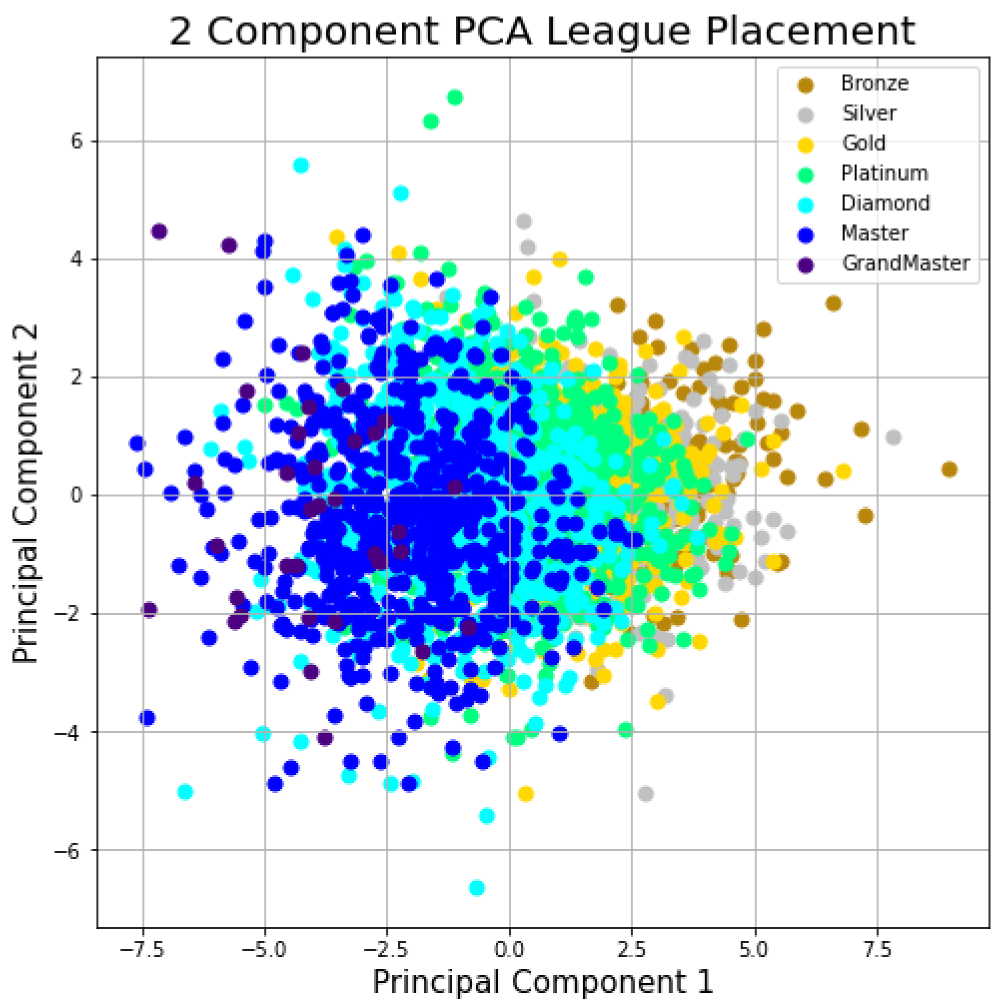
PCA analysis of League Placement (2 components)

In [6]:
```
def pcaout(pca, n_ftrs, col_nms):
    print("Principal components:")
    idx = ['PC-1','PC-2','PC-3','PC-4','PC-5','PC-6']
    return pd.DataFrame(pca.components_, columns=col_nms, index = idx[:n_ftr
```

In [7]:

```
X = data.iloc[:,8:]
Xcol = X.columns
Y = data.iloc[:,7]
#transform input data (normalize)
ssc = SSc()
Xft = ssc.fit_transform(X)
X = pd.DataFrame(Xft)

pca = PCA(n_components=2)
components = pca.fit_transform(X)
componentDf = pd.DataFrame(data=components, columns=['principal component 1'

pltDF = pd.concat([componentDf, Y], axis = 1)
print("PCA explained variance ratio:  {}".format(pca.explained_variance_rati
print("Portion of variance explained: {}".format(pca.explained_variance_rati


#plot
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 Component PCA League Placement', fontsize = 20)


results = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0]
colors = ['darkgoldenrod','silver','gold','springgreen','aqua','blue','indig
for result, color in zip(results,colors):
    indicesToKeep = (pltDF['LeagueIndex'] == result)
    ax.scatter(pltDF.loc[indicesToKeep, 'principal component 1']
               , pltDF.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(["Bronze","Silver","Gold","Platinum","Diamond","Master","GrandMast
ax.grid()
```

```
PCA explained variance ratio:  [0.26761666 0.11463843]
Portion of variance explained: [0.7000997  0.29990032]
```



In [8]:

```
df = pcaout(pca, 2, Xcol)
df
```

```
Principal components:
```

Out[8]:

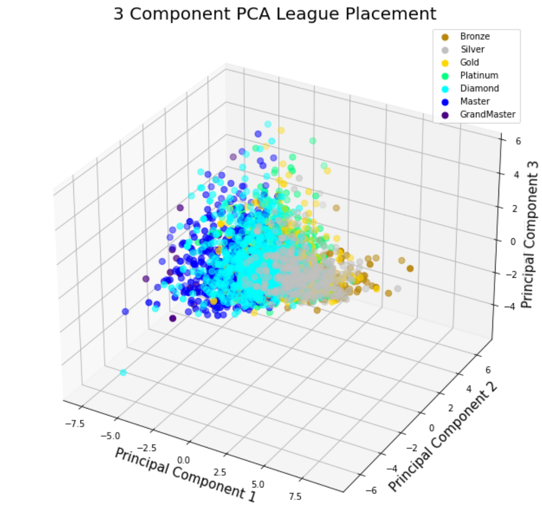
	Age	HoursPerWeek	TotalHours	APM	SelectByHotkeys	AssignToHotkeys	UniqueHc
PC-1	0.108518	-0.130121	-0.029533	-0.396779	-0.276801	-0.299392	-0.2
PC-2	0.085461	-0.092836	-0.044787	-0.243875	-0.240381	-0.063163	0.0

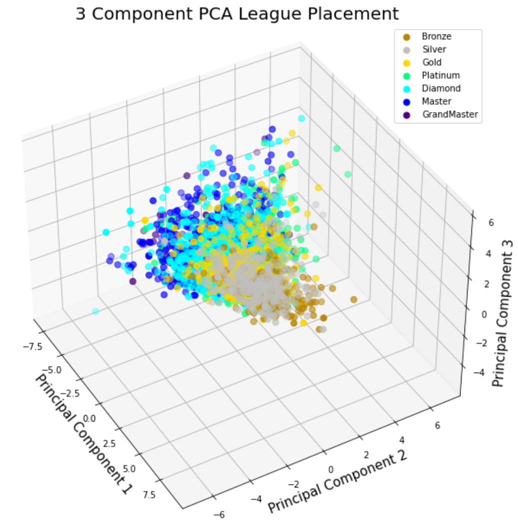
PCA analysis of League Placement (3 components)

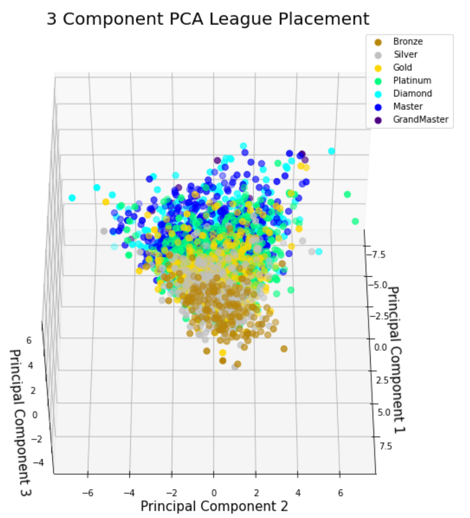
In [11]:

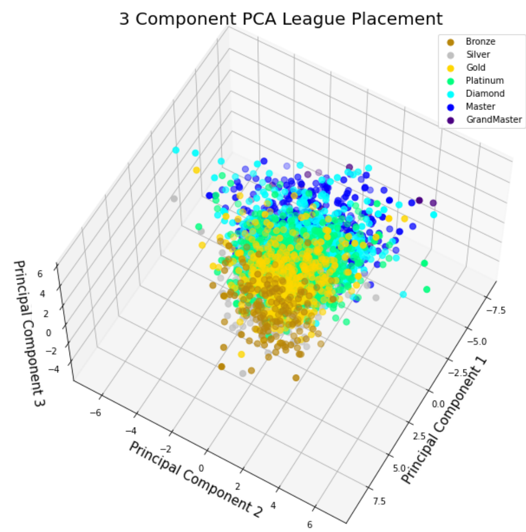
```
X = data.iloc[:,8:]
Y = data.iloc[:,7]
#transform input data (normalize)
ssc = SSc()
Xft = ssc.fit_transform(X)
X = pd.DataFrame(Xft)

pca = PCA(n_components=3)
components = pca.fit_transform(X)
componentDf = pd.DataFrame(data=components, columns=['principal component 1'

pltDF = pd.concat([componentDf, Y], axis = 1)
print("PCA explained variance ratio:  {}".format(pca.explained_variance_rati
print("Portion of variance explained: {}".format(pca.explained_variance_rati


#plot
fig = plt.figure(figsize = (24,36))


results = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0]
colors = ['darkgoldenrod','silver','gold','springgreen','aqua','blue','indig

for i in range(6):
    ax = fig.add_subplot(3,2,i+1,projection='3d')
    ax.view_init(30+(10*i),300+(30*i))
    ax.set_xlabel('Principal Component 1', fontsize = 15)
    ax.set_ylabel('Principal Component 2', fontsize = 15)
    ax.set_zlabel('Principal Component 3', fontsize = 15)
    ax.set_title('3 Component PCA League Placement', fontsize = 20)

    for result, color in zip(results,colors):
        indicesToKeep = (pltDF['LeagueIndex'] == result)
        ax.scatter(pltDF.loc[indicesToKeep, 'principal component 1']
                   , pltDF.loc[indicesToKeep, 'principal component 2']
                   , pltDF.loc[indicesToKeep, 'principal component 3']
                   , c = color
                   , s = 50)
    ax.legend(["Bronze","Silver","Gold","Platinum","Diamond","Master","Grand
    ax.grid()

```

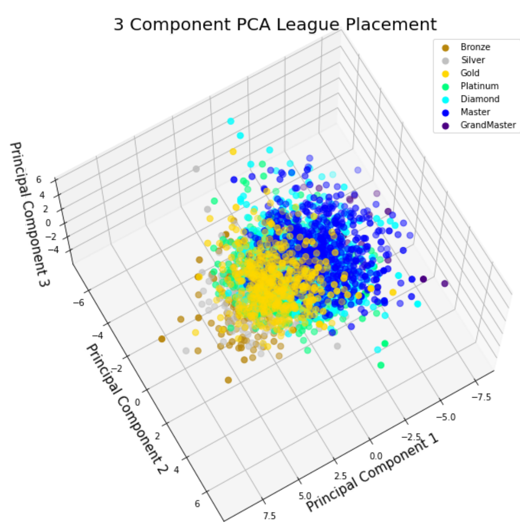
```
PCA explained variance ratio:  [0.26761657 0.11463729 0.08399942]
Portion of variance explained: [0.5739725  0.24586913 0.18015835]
```

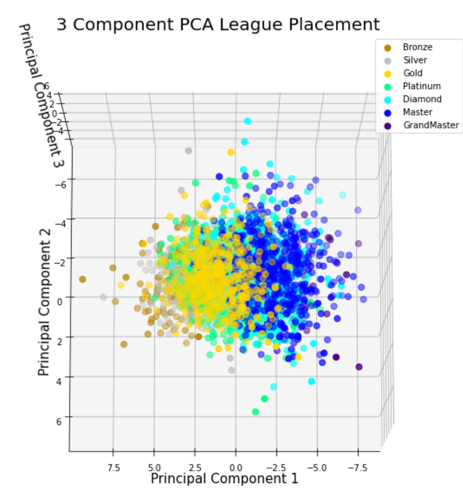












from this, it's pretty clear that the PCA variables (particularly component 1) differentiates the league of the player very well despite the player's league not being input.

In [ ]:

```
```