Tokyo olympic games medals 2021 visualization and prediction

In [34]:
```
# Importing required packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [35]:
```
# Invoke data from CSV file
olympic_dataset = pd.read_csv('Tokyo Medals 2021.csv')
print(olympic_dataset)
```

```
                  Country  Gold Medal  Silver Medal  Bronze Medal  Total  \
0    United States of America          39            41            33    113
1    People's Republic of China          38            32            18     88
2                   Japan          27            14            17     58
3           Great Britain          22            21            22     65
4                    ROC          20            28            23     71
..                   ...         ...           ...           ...    ...
88                 Ghana           0             0             1      1
89               Grenada           0             0             1      1
90                Kuwait           0             0             1      1
91    Republic of Moldova           0             0             1      1
92        Syrian Arab Republic          0             0             1      1

   Rank By Total
0              1
1              2
2              5
3              4
4              3
..           ...
88            77
89            77
90            77
91            77
92            77

[93 rows x 6 columns]
```

In [36]:
```
# Printing the first 5 rows from data set
olympic_dataset.head()
```

Out[36]:

	Country	Gold Medal	Silver Medal	Bronze Medal	Total	Rank By Total
0	United States of America	39	41	33	113	1
1	People's Republic of China	38	32	18	88	2
2	Japan	27	14	17	58	5
3	Great Britain	22	21	22	65	4
4	ROC	20	28	23	71	3

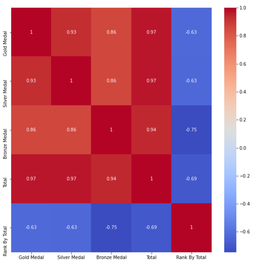
In [37]:
```
# Checking the shape of data
olympic_dataset.shape
```

Out[37]:
```
(93, 6)
```

Data Visualization

In [38]:
```
# Constructing a heatmap
olympic = olympic_dataset.corr()
plt.figure(figsize=(10,10))
sns.heatmap(olympic,cbar=True,annot=True,cmap='coolwarm')
```

Out[38]:
```
<AxesSubplot:>
```



Preprocessing

In [39]:
```
# Features selection
X = olympic_dataset.iloc[:,1:4].values
y = olympic_dataset.iloc[:,3].values
```

In [40]:
```
print(X)
```

```
[[39 41 33]
 [38 32 18]
 [27 14 17]
 [22 21 22]
 [20 28 23]
 [17  2 22]
 [10 12 14]
 [10 12 11]
 [10 11 16]
 [10 10 20]
 [ 7  6 11]
 [ 7  6  8]
 [ 7  5  7]
 [ 7  3  5]
 [ 6  4 10]
 [ 6  4 10]
 [ 5  5  9]
 [ 4  4  3]
 [ 4  4  2]
 [ 4  2  2]
 [ 4  1  4]
 [ 3  8  6]
 [ 3  6  9]
 [ 3  4  6]
 [ 3  4  4]
 [ 3  3  2]
 [ 3  2  2]
 [ 3  1  5]
 [ 3  1  3]
 [ 3  1  2]
 [ 3  1  1]
 [ 3  0  2]
 [ 2  5  1]
 [ 2  4  6]
 [ 2  2  9]
 [ 2  1  1]
 [ 2  1  1]
 [ 2  1  0]
 [ 2  0  2]
 [ 2  0  0]
 [ 2  0  0]
 [ 2  0  0]
 [ 1 12]
 [ 1  3  3]
 [ 1  3  0]
 [ 1  3  0]
 [ 1  2  4]
 [ 1  2  3]
 [ 1  2  1]
 [ 1  2  1]
 [ 1  2  0]
 [ 1  1  5]
 [ 1  1  4]
 [ 1  1  3]
 [ 1  1  2]
 [ 1  1  2]
 [ 1  1  0]
 [ 1  0  1]
 [ 1  0  1]
 [ 1  0  1]
 [ 1  0  1]
 [ 1  0  0]
 [ 1  0  0]
 [ 0  4  1]
 [ 0  3  4]
 [ 0  3  2]
 [ 0  2  2]
 [ 0  2  1]
 [ 0  1  3]
 [ 0  1  2]
 [ 0  1  1]
 [ 0  1  1]
 [ 0  1  0]
 [ 0  1  0]
 [ 0  1  0]
 [ 0  1  0]
 [ 0  1  0]
 [ 0  0  8]
 [ 0  0  4]
 [ 0  0  2]
 [ 0  0  1]
 [ 0  0  1]
 [ 0  0  1]
 [ 0  0  1]
 [ 0  0  1]
 [ 0  0  1]]
```

In [41]:
```
print(y)
```

```
[33 18 17 22 23 22 14 11 16 20 11  8  7  5  7 10  5  3  2  2  4  6  0  6
  4  2  2  5  3  2  1  2  1  6  9  1  1  0  2  2  1  0  0 12  3  0  0  4
  3  1  1  0  5  4  3  2  2  0  1  1  1  1  0  0  0  1  4  2  2  1  3  2
  2  1  1  1  0  0  0  0  0  0  0  4  2  1  1  1  1  1  1  1]
```

In [42]:
```
# Splitting the data into training and testing data
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=10)
```

Using naive bayes

In [43]:
```
# Loading Algorithm module
from sklearn.naive_bayes import GaussianNB
clfr = GaussianNB()
```

In [44]:
```
# Fitting training data
clfr.fit(X_train,y_train)
```

Out[44]:
```
GaussianNB()
```

In [45]:
```
# Prediction on test data
y_pre = clfr.predict(X_test)
```

Accuracy score for naive bayes

In [47]:
```
from sklearn.metrics import accuracy_score
result=accuracy_score(y_test,y_pre)
print(result)
```

```
0.8
```

Using support vector classifier

In [48]:
```
# Loading Algorithm module
from sklearn.svm import SVC
clfr=SVC()
```

In [49]:
```
# Fitting training data
clfr.fit(X_train,y_train)
```

Out[49]:
```
SVC()
```

In [50]:
```
# Prediction on test data
y_pre = clfr.predict(X_test)
```

Accuracy score for support vector classifier

In [52]:
```
from sklearn.metrics import accuracy_score
result=accuracy_score(y_test,y_pre)
print(result)
```

```
0.1
```

Using K Neighbors classsifier

In [78]:
```
# Loading Algorithm module
from sklearn.neighbors import KNeighborsClassifier
clfr=KNeighborsClassifier(n_neighbors=3)
```

In [79]:
```
# Fitting training data
clfr.fit(X_train,y_train)
```

Out[79]:
```
KNeighborsClassifier(n_neighbors=3)
```

In [80]:
```
# Prediction on test data
y_pre = clfr.predict(X_test)
```

Accuracy score for K Neighbors classifier

In [82]:
```
from sklearn.metrics import accuracy_score
result=accuracy_score(y_test,y_pre)
print(result*100)
```

```
50.0
```

CONCLUSION:

Naive bayes = 80%

Support vector classifier = 10%

K Neighbors classifier = 50%

In [ ]:
```
```