In [70]:
```
#Clustering Teams by Offensive and Defensive Rating during the 2022-2023 season, How man
#or are there more
```

In [71]:
```
## Import Library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="darkgrid")
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler
from yellowbrick.cluster import KElbowVisualizer
from sklearn.metrics import silhouette_samples, silhouette_score
import matplotlib.cm as cm

## Load Data
dataset = pd.read_excel(r"C:\Users\djbro\OneDrive\Desktop\Clustering\sportsref_download
dataset.head()
```

Out[71]:

	Rk	Season	Team	W	G	W.1	L	W/L%	MOV	SOS	...	eFG%	TS%	TOV%	ORB%	FTr	eFG%.1	TS'
0	1	2022-23	BOS	23	33	23	10	0.697	5.85	-0.20	...	0.566	0.604	12.4	21.3	0.255	0.529	0.
1	2	2022-23	MIL	22	32	22	10	0.688	3.25	0.07	...	0.534	0.566	13.3	26.5	0.265	0.515	0.
2	3	2022-23	CLE	22	34	22	12	0.647	6.00	-0.20	...	0.549	0.587	13.4	23.3	0.278	0.523	0.
3	4	2022-23	BRK	21	33	21	12	0.636	3.00	0.13	...	0.579	0.613	13.6	19.6	0.251	0.525	0.
4	5	2022-23	NOP	20	32	20	12	0.625	4.91	-0.38	...	0.547	0.586	13.2	26.7	0.282	0.536	0.

5 rows × 24 columns

In [72]:
```
dataset.columns
```

Out[72]:
```
Index(['Rk', 'Season', 'Team', 'W', 'G', 'W.1', 'L', 'W/L%', 'MOV', 'SOS',
       'SRS', 'Pace', 'ORtg', 'DRtg', 'eFG%', 'TS%', 'TOV%', 'ORB%', 'FTr',
       'eFG%.1', 'TS%.1', 'TOV%.1', 'ORB%.1', 'FTr.1'],
      dtype='object')
```

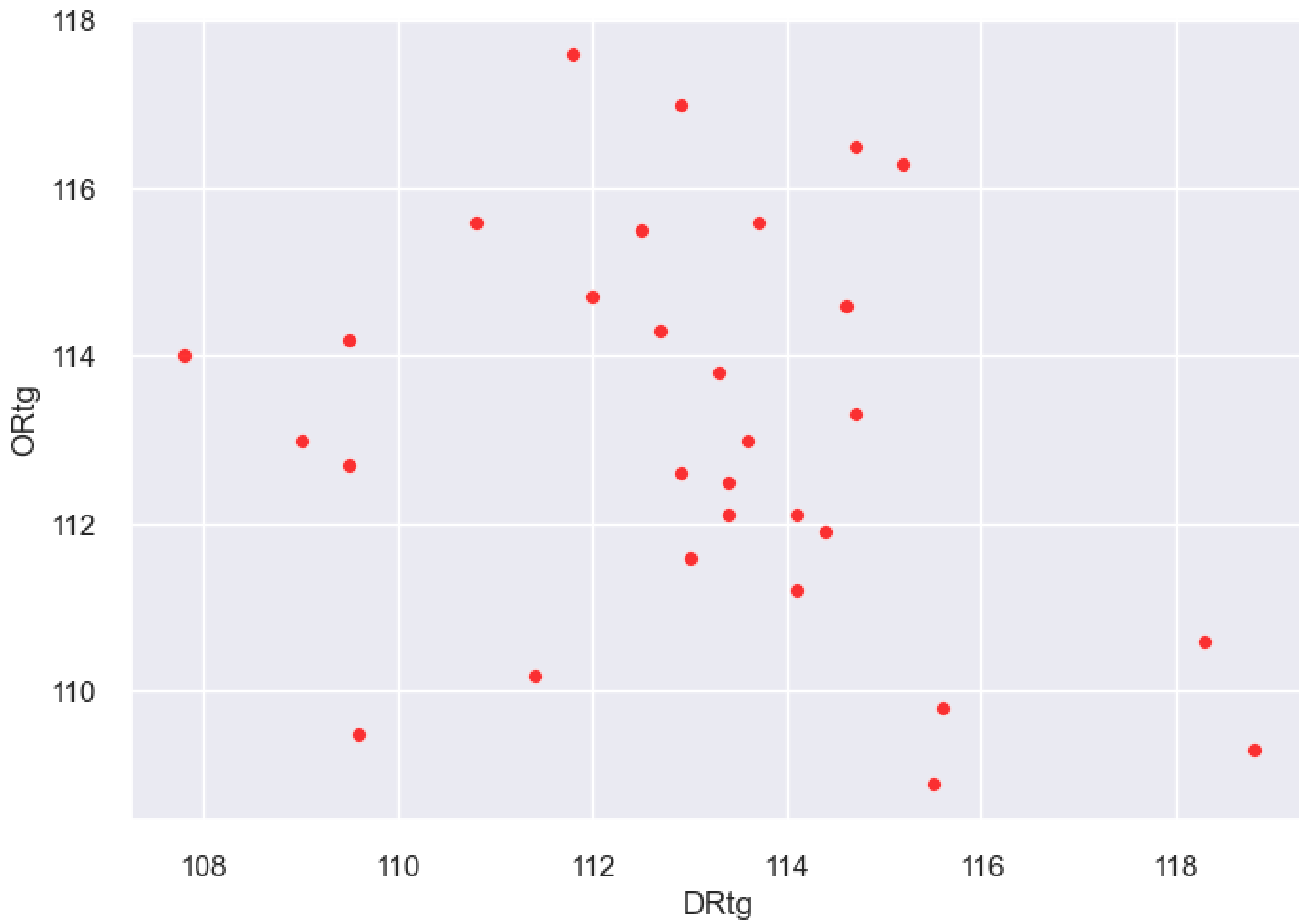
In [73]:
```
dataset = dataset[['Team','ORtg','DRtg','W/L%']]
data = dataset.drop(['Team','W/L%'],axis=1)
data.head()
```

Out[73]:

	ORtg	DRtg
0	117.6	111.8
1	112.7	109.5
2	114.0	107.8
3	115.5	112.5
4	115.6	110.8

In [74]:
```
#Plot
sns.scatterplot(x="DRtg", y="ORtg", data=data, s=30, color="red", alpha = 0.8)
```

Out[74]: <AxesSubplot:xlabel='DRtg', ylabel='ORtg'>



In [75]:

```
#specify our cluster features
data_x = dataset.iloc[:, 1:3]
data_x.head()

x_array =  np.array(data_x)
print(x_array)
```

```
[[117.6 111.8]
 [112.7 109.5]
 [114.  107.8]
 [115.5 112.5]
 [115.6 110.8]
 [116.5 114.7]
 [114.2 109.5]
 [117.  112.9]
 [113.  109. ]
 [109.5 109.6]
 [116.3 115.2]
 [114.7 112. ]
 [113.  113.6]
 [115.6 113.7]
 [112.6 112.9]
 [114.3 112.7]
 [114.6 114.6]
 [110.2 111.4]
 [112.5 113.4]
 [113.8 113.3]
 [113.3 114.7]
 [111.6 113. ]
 [112.1 113.4]
 [111.9 114.4]
 [111.2 114.1]
```

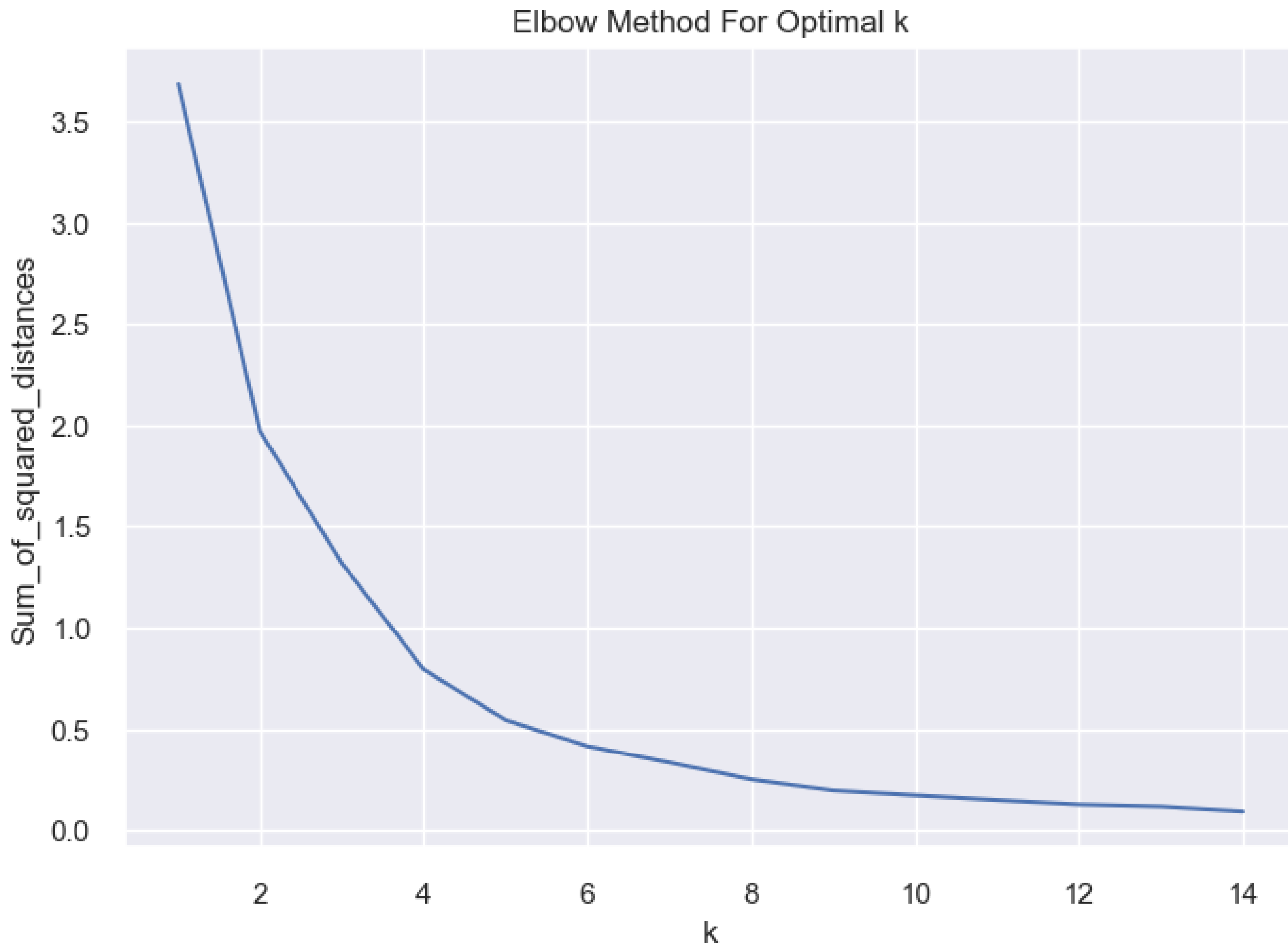
```
 [112.1 114.1]
 [109.3 118.8]
 [109.8 115.6]
 [108.9 115.5]
 [110.6 118.3]]
```

In [76]:
```
# Scale features
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x_array)
x_scaled
```

Out[76]:
```
array([[1.        , 0.36363636],
       [0.43678161, 0.15454545],
       [0.5862069 , 0.        ],
       [0.75862069, 0.42727273],
       [0.77011494, 0.27272727],
       [0.87356322, 0.62727273],
       [0.6091954 , 0.15454545],
       [0.93103448, 0.46363636],
       [0.47126437, 0.10909091],
       [0.06896552, 0.16363636],
       [0.85057471, 0.67272727],
       [0.66666667, 0.38181818],
       [0.47126437, 0.52727273],
       [0.77011494, 0.53636364],
       [0.42528736, 0.46363636],
       [0.62068966, 0.44545455],
       [0.65517241, 0.61818182],
       [0.14942529, 0.32727273],
       [0.4137931 , 0.50909091],
       [0.56321839, 0.5       ],
       [0.50574713, 0.62727273],
       [0.31034483, 0.47272727],
       [0.36781609, 0.50909091],
       [0.34482759, 0.6       ],
       [0.26436782, 0.57272727],
       [0.36781609, 0.57272727],
       [0.04597701, 1.        ],
       [0.10344828, 0.70909091],
       [0.        , 0.7       ],
       [0.1954023 , 0.95454545]])
```

In [77]:
```
#Elbow method to minimize WSS (within-cluster Sum of Square)
Sum_of_squared_distances =[]
K = range(1,15)
for k in K:
    km =KMeans(n_clusters =k)
    km =km.fit(x_scaled)
    Sum_of_squared_distances.append(km.inertia_)
###plotting Elbow
plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distances')
plt.title('Elbow Method For Optimal k')
plt.show()
```

```
C:\Users\djbro\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1036: UserWarning:
KMeans is known to have a memory leak on Windows with MKL, when there are less chunks th
an available threads. You can avoid it by setting the environment variable OMP_NUM_THREA
DS=1.
  warnings.warn(
```

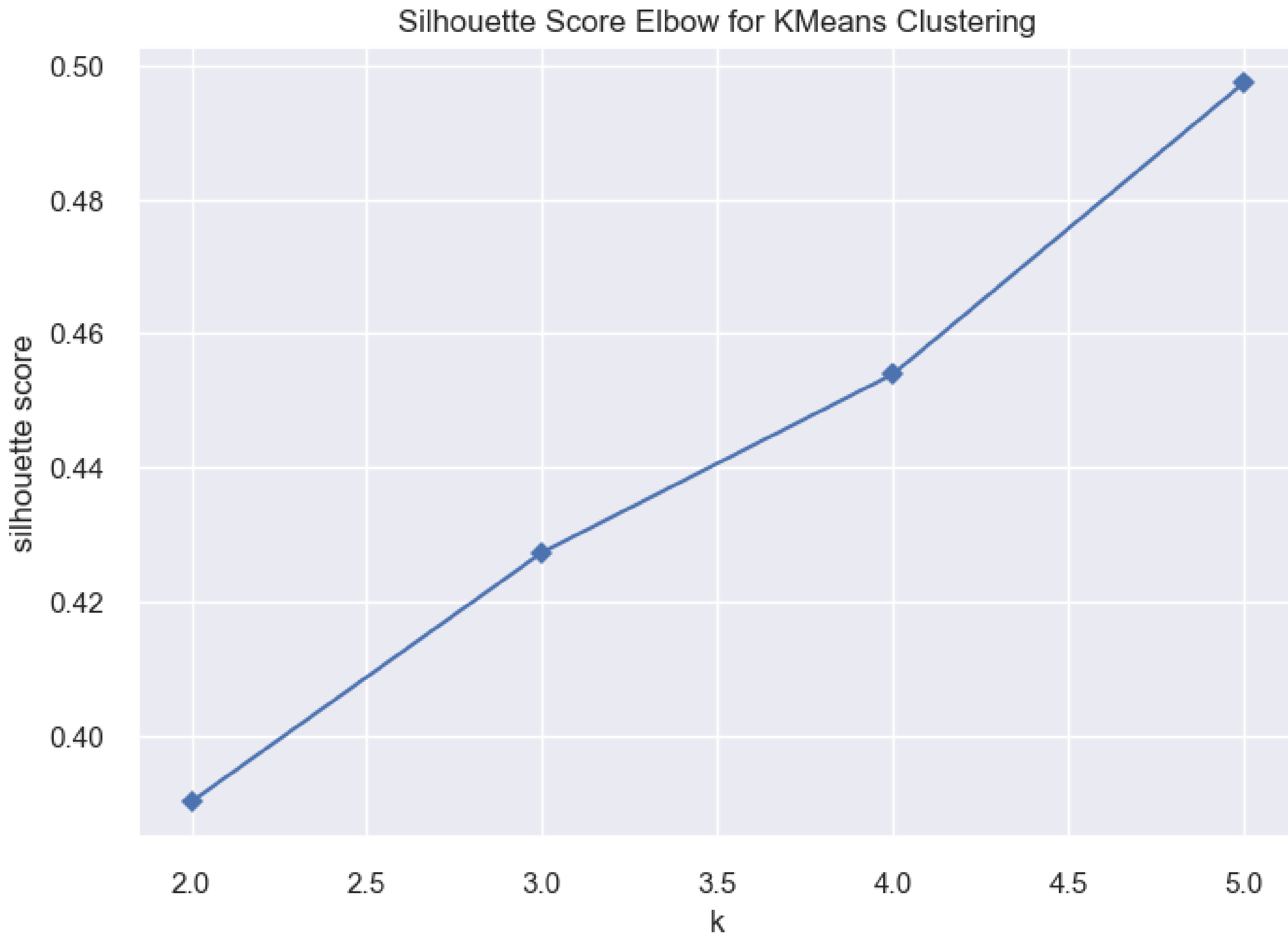


In [78]:
```
#Now if we observe the point after which there isn’t a sudden change in WCSS in K=5.
#So we will choose K=5 as an appropriate number of clusters.
```

In [79]:
```
#Silhouette Coefficient method, the silhouette coefficient of a data
#measures how well data are assigned to its own cluster and how far they
#are from other clusters. A silhouette close to 1 means the data points
#are in an appropriate cluster and a silhouette
#coefficient close to −1 implies out data is in the wrong cluster.
```

In [80]:
```
model = KMeans(random_state=123)
# Instantiate the KElbowVisualizer with the number of clusters and the metric
visualizer = KElbowVisualizer(model, k=(2,6), metric='silhouette', timings=False)
# Fit the data and visualize
visualizer.fit(x_scaled)
visualizer.poof()
```

```
C:\Users\djbro\anaconda3\lib\site-packages\yellowbrick\utils\kneed.py:156: YellowbrickWa
rning: No 'knee' or 'elbow point' detected This could be due to bad clustering, no actua
l clusters being formed etc.
  warnings.warn(warning_message, YellowbrickWarning)
C:\Users\djbro\anaconda3\lib\site-packages\yellowbrick\cluster\elbow.py:374: Yellowbrick
Warning: No 'knee' or 'elbow' point detected, pass `locate_elbow=False` to remove the wa
rning
  warnings.warn(warning_message, YellowbrickWarning)
```



Out[80]:

```
<AxesSubplot:title={'center':'Silhouette Score Elbow for KMeans Clustering'}, xlabel
='k', ylabel='silhouette score'>
```

In [81]:

```
# Menentukan kluster dari data
model.fit(x_scaled)
# Menampilkan pusat cluster
print(model.cluster_centers_)
```

```
[[0.78965517 0.48090909]
 [0.40344828 0.53545455]
 [0.52586207 0.10454545]
 [0.0862069  0.84090909]
 [0.1091954  0.24545455]]
```

In [82]:

```
# Menampilkan hasil kluster
print(model.labels_)
```

```
[0 2 2 0 0 0 2 0 2 4 0 0 1 0 1 0 0 4 1 1 1 1 1 1 1 1 3 3 3 3]
```

In [83]:

```
# Menambahkan kolom "kluster" dalam data frame dataset
dataset["kluster"] = model.labels_
dataset.sort_values(by='W/L%',ascending=False)
```

Out[83]:

	Team	ORtg	DRtg	W/L%	kluster
0	BOS	117.6	111.8	0.697	0
1	MIL	112.7	109.5	0.688	2
2	CLE	114.0	107.8	0.647	2
5	DEN	116.5	114.7	0.645	0
6	MEM	114.2	109.5	0.645	2
3	BRK	115.5	112.5	0.636	0

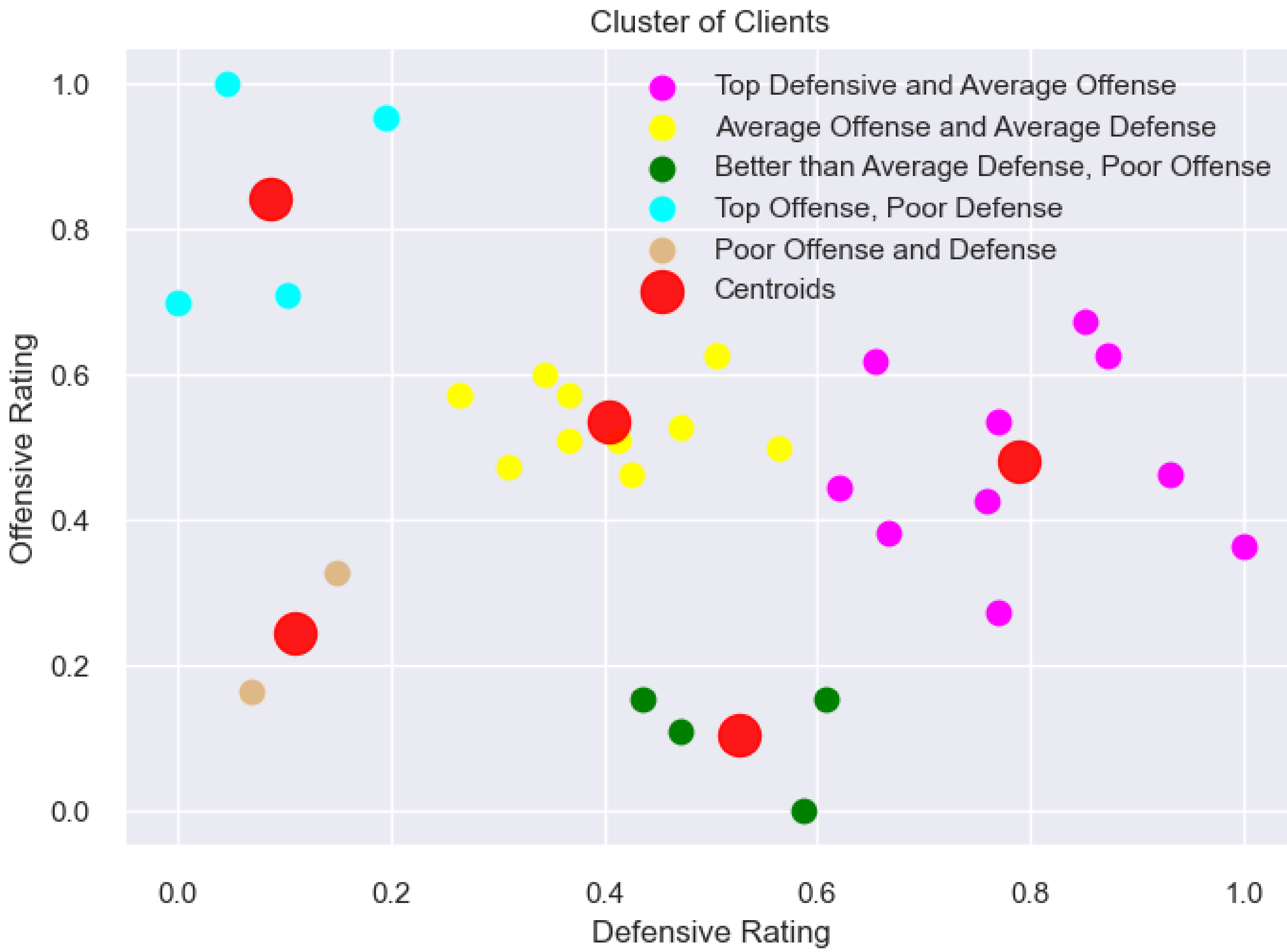
4	NOP	115.6	110.8	0.625	0
8	PHI	113.0	109.0	0.613	2
7	PHO	117.0	112.9	0.576	0
9	LAC	109.5	109.6	0.559	4
13	SAC	115.6	113.7	0.548	0
11	NYK	114.7	112.0	0.545	0
10	UTA	116.3	115.2	0.543	0
16	POR	114.6	114.6	0.515	0
15	DAL	114.3	112.7	0.515	0
14	ATL	112.6	112.9	0.515	1
12	IND	113.0	113.6	0.515	1
17	MIA	110.2	111.4	0.485	4
18	MIN	112.5	113.4	0.485	1
19	TOR	113.8	113.3	0.455	1
20	GSW	113.3	114.7	0.455	1
22	CHI	112.1	113.4	0.438	1
21	OKC	111.6	113.0	0.424	1
25	LAL	112.1	114.1	0.406	1
23	WAS	111.9	114.4	0.382	1
24	ORL	111.2	114.1	0.382	1
26	SAS	109.3	118.8	0.313	3
27	HOU	109.8	115.6	0.281	3
28	CHO	108.9	115.5	0.273	3
29	DET	110.6	118.3	0.229	3

In [84]:

```
# Memvisualkan hasil kluster
plt.scatter(x_scaled[model.labels_==0,0],x_scaled[model.labels_==0,1],s=80,c='magenta',l
plt.scatter(x_scaled[model.labels_==1,0],x_scaled[model.labels_==1,1],s=80,c='yellow',la
plt.scatter(x_scaled[model.labels_==2,0],x_scaled[model.labels_==2,1],s=80,c='green',lab
plt.scatter(x_scaled[model.labels_==3,0],x_scaled[model.labels_==3,1],s=80,c='cyan',labe
plt.scatter(x_scaled[model.labels_==4,0],x_scaled[model.labels_==4,1],s=80,c='burlywood'
plt.scatter(model.cluster_centers_[:,0],model.cluster_centers_[:,1],marker = "o", alpha
plt.title('Cluster of Clients')
plt.xlabel('Defensive Rating')
plt.ylabel('Offensive Rating')
plt.legend()
plt.show
```

Out[84]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [85]:
```
clusters = dataset.groupby(['kluster'])['W/L%'].mean()
clusters
```

Out[85]:
```
kluster
0    0.58450
1    0.44570
2    0.64825
3    0.27400
4    0.52200
Name: W/L%, dtype: float64
```

In [86]:
```
clusters = dataset.groupby(['kluster'])['W/L%'].var()
clusters
```

Out[86]:
```
kluster
0    0.003884
1    0.002394
2    0.000945
3    0.001199
4    0.002738
Name: W/L%, dtype: float64
```

In [ ]:
```
#It  looks like the top defensive teams have the best win percentage in the league, desp
#to poor offenses.
```