In [26]:
```
## 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_csv(r"C:\Users\djbro\OneDrive\Desktop\Clustering\Mall_Customers.csv")
dataset.head()
```

Out[26]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [27]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

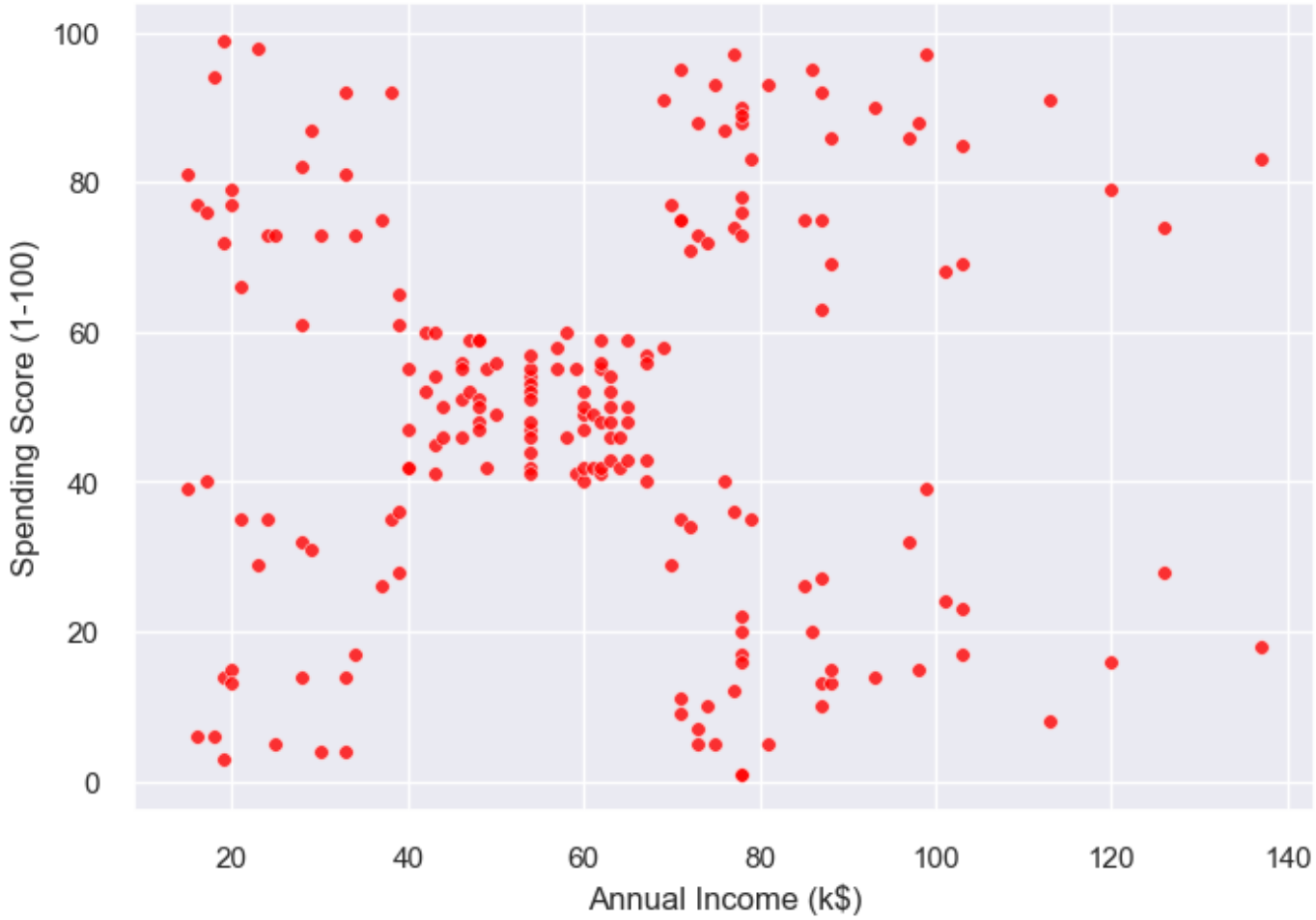
In [28]:
```
#We are only performing 2 feature Clustering
data = dataset.drop(["CustomerID","Gender","Age"], axis = 1)
data.head()
```

Out[28]:

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40

In [29]:
```
#Plot
sns.scatterplot(x="Annual Income (k$)", y="Spending Score (1-100)", data=data, s=30, col
```

Out[29]:
```
<AxesSubplot:xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



In [30]:
```
#specify our cluster features
data_x = dataset.iloc[:, 3:5]
data_x.head()

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

```
[[ 15  39]
 [ 15  81]
 [ 16   6]
 [ 16  77]
 [ 17  40]
 [ 17  76]
 [ 18   6]
 [ 18  94]
 [ 19   3]
 [ 19  72]
 [ 19  14]
 [ 19  99]
 [ 20  15]
 [ 20  77]
 [ 20  13]
 [ 20  79]
 [ 21  35]
 [ 21  66]
 [ 23  29]
 [ 23  98]
 [ 24  35]
 [ 24  73]
 [ 25   5]
 [ 25  73]
 [ 28  14]
 [ 28  82]
 [ 28  32]
```

```
[ 28  61]
[ 29  31]
[ 29  87]
[ 30   4]
[ 30  73]
[ 33   4]
[ 33  92]
[ 33  14]
[ 33  81]
[ 34  17]
[ 34  73]
[ 37  26]
[ 37  75]
[ 38  35]
[ 38  92]
[ 39  36]
[ 39  61]
[ 39  28]
[ 39  65]
[ 40  55]
[ 40  47]
[ 40  42]
[ 40  42]
[ 42  52]
[ 42  60]
[ 43  54]
[ 43  60]
[ 43  45]
[ 43  41]
[ 44  50]
[ 44  46]
[ 46  51]
[ 46  46]
[ 46  56]
[ 46  55]
[ 47  52]
[ 47  59]
[ 48  51]
[ 48  59]
[ 48  50]
[ 48  48]
[ 48  59]
[ 48  47]
[ 49  55]
[ 49  42]
[ 50  49]
[ 50  56]
[ 54  47]
[ 54  54]
[ 54  53]
[ 54  48]
[ 54  52]
[ 54  42]
[ 54  51]
[ 54  55]
[ 54  41]
[ 54  44]
[ 54  57]
[ 54  46]
[ 57  58]
[ 57  55]
[ 58  60]
[ 58  46]
[ 59  55]
[ 59  41]
[ 60  49]
```

```
[ 60  40]
[ 60  42]
[ 60  52]
[ 60  47]
[ 60  50]
[ 61  42]
[ 61  49]
[ 62  41]
[ 62  48]
[ 62  59]
[ 62  55]
[ 62  56]
[ 62  42]
[ 63  50]
[ 63  46]
[ 63  43]
[ 63  48]
[ 63  52]
[ 63  54]
[ 64  42]
[ 64  46]
[ 65  48]
[ 65  50]
[ 65  43]
[ 65  59]
[ 67  43]
[ 67  57]
[ 67  56]
[ 67  40]
[ 69  58]
[ 69  91]
[ 70  29]
[ 70  77]
[ 71  35]
[ 71  95]
[ 71  11]
[ 71  75]
[ 71   9]
[ 71  75]
[ 72  34]
[ 72  71]
[ 73   5]
[ 73  88]
[ 73   7]
[ 73  73]
[ 74  10]
[ 74  72]
[ 75   5]
[ 75  93]
[ 76  40]
[ 76  87]
[ 77  12]
[ 77  97]
[ 77  36]
[ 77  74]
[ 78  22]
[ 78  90]
[ 78  17]
[ 78  88]
[ 78  20]
[ 78  76]
[ 78  16]
[ 78  89]
[ 78   1]
[ 78  78]
[ 78   1]
```

```
 [ 78  73]
 [ 79  35]
 [ 79  83]
 [ 81   5]
 [ 81  93]
 [ 85  26]
 [ 85  75]
 [ 86  20]
 [ 86  95]
 [ 87  27]
 [ 87  63]
 [ 87  13]
 [ 87  75]
 [ 87  10]
 [ 87  92]
 [ 88  13]
 [ 88  86]
 [ 88  15]
 [ 88  69]
 [ 93  14]
 [ 93  90]
 [ 97  32]
 [ 97  86]
 [ 98  15]
 [ 98  88]
 [ 99  39]
 [ 99  97]
 [101  24]
 [101  68]
 [103  17]
 [103  85]
 [103  23]
 [103  69]
 [113   8]
 [113  91]
 [120  16]
 [120  79]
 [126  28]
 [126  74]
 [137  18]
 [137  83]]
```

In [31]:

```
# Scale features
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x_array)
x_scaled
```

Out[31]:

```
array([[0.        , 0.3877551 ],
       [0.        , 0.81632653],
       [0.00819672, 0.05102041],
       [0.00819672, 0.7755102 ],
       [0.01639344, 0.39795918],
       [0.01639344, 0.76530612],
       [0.02459016, 0.05102041],
       [0.02459016, 0.94897959],
       [0.03278689, 0.02040816],
       [0.03278689, 0.7244898 ],
       [0.03278689, 0.13265306],
       [0.03278689, 1.        ],
       [0.04098361, 0.14285714],
       [0.04098361, 0.7755102 ],
       [0.04098361, 0.12244898],
       [0.04098361, 0.79591837],
       [0.04918033, 0.34693878],
       [0.04918033, 0.66326531],
       [0.06557377, 0.28571429],
```

```
[0.06557377, 0.98979592],
[0.07377049, 0.34693878],
[0.07377049, 0.73469388],
[0.08196721, 0.04081633],
[0.08196721, 0.73469388],
[0.10655738, 0.13265306],
[0.10655738, 0.82653061],
[0.10655738, 0.31632653],
[0.10655738, 0.6122449 ],
[0.1147541 , 0.30612245],
[0.1147541 , 0.87755102],
[0.12295082, 0.03061224],
[0.12295082, 0.73469388],
[0.14754098, 0.03061224],
[0.14754098, 0.92857143],
[0.14754098, 0.13265306],
[0.14754098, 0.81632653],
[0.1557377 , 0.16326531],
[0.1557377 , 0.73469388],
[0.18032787, 0.25510204],
[0.18032787, 0.75510204],
[0.18852459, 0.34693878],
[0.18852459, 0.92857143],
[0.19672131, 0.35714286],
[0.19672131, 0.6122449 ],
[0.19672131, 0.2755102 ],
[0.19672131, 0.65306122],
[0.20491803, 0.55102041],
[0.20491803, 0.46938776],
[0.20491803, 0.41836735],
[0.20491803, 0.41836735],
[0.22131148, 0.52040816],
[0.22131148, 0.60204082],
[0.2295082 , 0.54081633],
[0.2295082 , 0.60204082],
[0.2295082 , 0.44897959],
[0.2295082 , 0.40816327],
[0.23770492, 0.5       ],
[0.23770492, 0.45918367],
[0.25409836, 0.51020408],
[0.25409836, 0.45918367],
[0.25409836, 0.56122449],
[0.25409836, 0.55102041],
[0.26229508, 0.52040816],
[0.26229508, 0.59183673],
[0.2704918 , 0.51020408],
[0.2704918 , 0.59183673],
[0.2704918 , 0.5       ],
[0.2704918 , 0.47959184],
[0.2704918 , 0.59183673],
[0.2704918 , 0.46938776],
[0.27868852, 0.55102041],
[0.27868852, 0.41836735],
[0.28688525, 0.48979592],
[0.28688525, 0.56122449],
[0.31967213, 0.46938776],
[0.31967213, 0.54081633],
[0.31967213, 0.53061224],
[0.31967213, 0.47959184],
[0.31967213, 0.52040816],
[0.31967213, 0.41836735],
[0.31967213, 0.51020408],
[0.31967213, 0.55102041],
[0.31967213, 0.40816327],
[0.31967213, 0.43877551],
[0.31967213, 0.57142857],
```

```
[0.31967213, 0.45918367],
[0.3442623 , 0.58163265],
[0.3442623 , 0.55102041],
[0.35245902, 0.60204082],
[0.35245902, 0.45918367],
[0.36065574, 0.55102041],
[0.36065574, 0.40816327],
[0.36885246, 0.48979592],
[0.36885246, 0.39795918],
[0.36885246, 0.41836735],
[0.36885246, 0.52040816],
[0.36885246, 0.46938776],
[0.36885246, 0.5       ],
[0.37704918, 0.41836735],
[0.37704918, 0.48979592],
[0.3852459 , 0.40816327],
[0.3852459 , 0.47959184],
[0.3852459 , 0.59183673],
[0.3852459 , 0.55102041],
[0.3852459 , 0.56122449],
[0.3852459 , 0.41836735],
[0.39344262, 0.5       ],
[0.39344262, 0.45918367],
[0.39344262, 0.42857143],
[0.39344262, 0.47959184],
[0.39344262, 0.52040816],
[0.39344262, 0.54081633],
[0.40163934, 0.41836735],
[0.40163934, 0.45918367],
[0.40983607, 0.47959184],
[0.40983607, 0.5       ],
[0.40983607, 0.42857143],
[0.40983607, 0.59183673],
[0.42622951, 0.42857143],
[0.42622951, 0.57142857],
[0.42622951, 0.56122449],
[0.42622951, 0.39795918],
[0.44262295, 0.58163265],
[0.44262295, 0.91836735],
[0.45081967, 0.28571429],
[0.45081967, 0.7755102 ],
[0.45901639, 0.34693878],
[0.45901639, 0.95918367],
[0.45901639, 0.10204082],
[0.45901639, 0.75510204],
[0.45901639, 0.08163265],
[0.45901639, 0.75510204],
[0.46721311, 0.33673469],
[0.46721311, 0.71428571],
[0.47540984, 0.04081633],
[0.47540984, 0.8877551 ],
[0.47540984, 0.06122449],
[0.47540984, 0.73469388],
[0.48360656, 0.09183673],
[0.48360656, 0.7244898 ],
[0.49180328, 0.04081633],
[0.49180328, 0.93877551],
[0.5       , 0.39795918],
[0.5       , 0.87755102],
[0.50819672, 0.1122449 ],
[0.50819672, 0.97959184],
[0.50819672, 0.35714286],
[0.50819672, 0.74489796],
[0.51639344, 0.21428571],
[0.51639344, 0.90816327],
[0.51639344, 0.16326531],
```

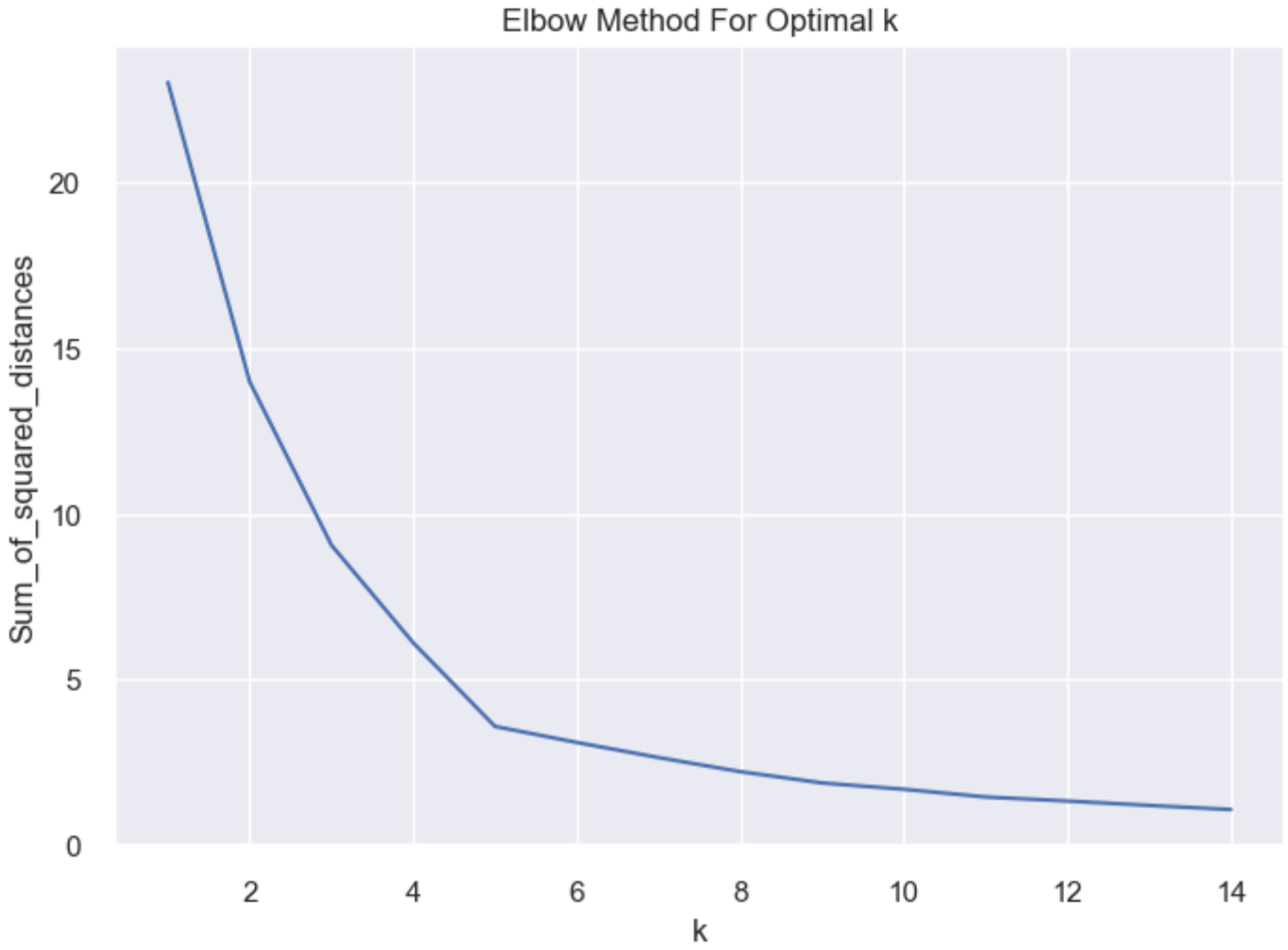
```
       [0.51639344, 0.8877551 ],
       [0.51639344, 0.19387755],
       [0.51639344, 0.76530612],
       [0.51639344, 0.15306122],
       [0.51639344, 0.89795918],
       [0.51639344, 0.        ],
       [0.51639344, 0.78571429],
       [0.51639344, 0.        ],
       [0.51639344, 0.73469388],
       [0.52459016, 0.34693878],
       [0.52459016, 0.83673469],
       [0.54098361, 0.04081633],
       [0.54098361, 0.93877551],
       [0.57377049, 0.25510204],
       [0.57377049, 0.75510204],
       [0.58196721, 0.19387755],
       [0.58196721, 0.95918367],
       [0.59016393, 0.26530612],
       [0.59016393, 0.63265306],
       [0.59016393, 0.12244898],
       [0.59016393, 0.75510204],
       [0.59016393, 0.09183673],
       [0.59016393, 0.92857143],
       [0.59836066, 0.12244898],
       [0.59836066, 0.86734694],
       [0.59836066, 0.14285714],
       [0.59836066, 0.69387755],
       [0.63934426, 0.13265306],
       [0.63934426, 0.90816327],
       [0.67213115, 0.31632653],
       [0.67213115, 0.86734694],
       [0.68032787, 0.14285714],
       [0.68032787, 0.8877551 ],
       [0.68852459, 0.3877551 ],
       [0.68852459, 0.97959184],
       [0.70491803, 0.23469388],
       [0.70491803, 0.68367347],
       [0.72131148, 0.16326531],
       [0.72131148, 0.85714286],
       [0.72131148, 0.2244898 ],
       [0.72131148, 0.69387755],
       [0.80327869, 0.07142857],
       [0.80327869, 0.91836735],
       [0.86065574, 0.15306122],
       [0.86065574, 0.79591837],
       [0.90983607, 0.2755102 ],
       [0.90983607, 0.74489796],
       [1.        , 0.17346939],
       [1.        , 0.83673469]])
```

In [32]:

```
#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(
```

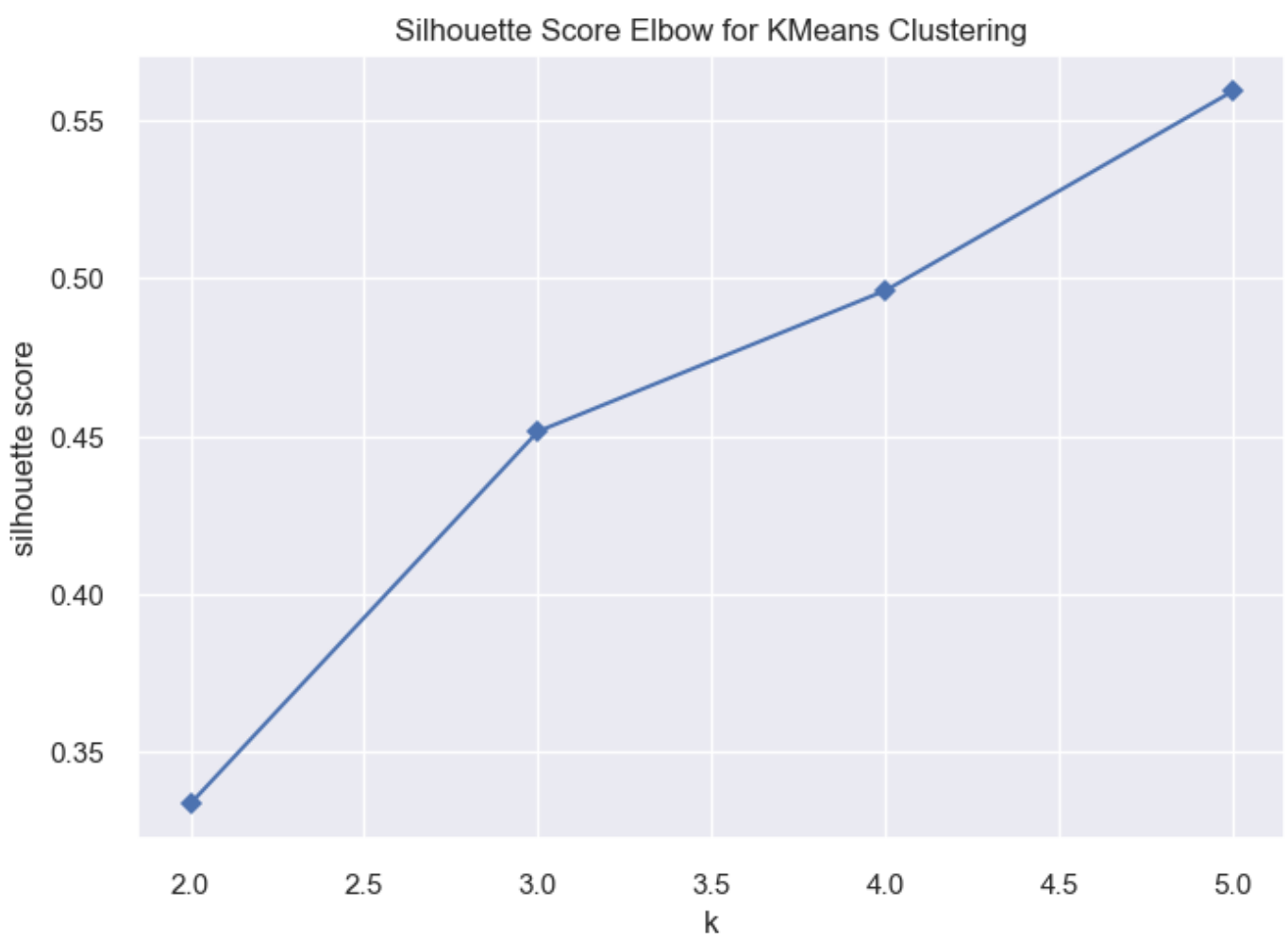


In [33]:
```
#Now if we observe the point after which there isn't a sudden change in WCSS in K=5.
#So we will choose K=5 as an appropriate number of clusters.
```

In [34]:
```
#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 [35]:
```
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: YellowbrickW
arning: No 'knee' or 'elbow' point detected, pass `locate_elbow=False` to remove the wa
rning
  warnings.warn(warning_message, YellowbrickWarning)
```



Out[35]:
```
<AxesSubplot:title={'center':'Silhouette Score Elbow for KMeans Clustering'}, xlabel
='k', ylabel='silhouette score'>
```

In [37]:
```
# Menentukan kluster dari data
model.fit(x_scaled)
# Menampilkan pusat cluster
print(model.cluster_centers_)
```

```
[[0.33029751 0.49508692]
 [0.58638083 0.82783883]
 [0.6        0.16443149]
 [0.09265859 0.20319432]
 [0.08792846 0.79962894]]
```

In [38]:
```
# Menampilkan hasil kluster
print(model.labels_)
```

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

In [39]:
```
# Menambahkan kolom "kluster" dalam data frame dataset
dataset["kluster"] = model.labels_
dataset.head()
```

Out[39]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	kluster
0	1	Male	19	15	39	3
1	2	Male	21	15	81	4
2	3	Female	20	16	6	3

3	4	Female	23	16	77	4
4	5	Female	31	17	40	3

In [41]:
```
# Memvisualkan hasil klusterplt.scatter(x_scaled[kmeans.labels_==0,0],x_scaled[kmeans.la
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('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show
```

Out[41]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

In [ ]:

In [ ]: