In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
mall_customers=pd.read_csv(r"C:\Users\s323\Desktop\Gatherings\Data Science\ML\Amit
```

In [3]:
```
mall_customers.head()
```

Out[3]:

	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 [4]:
```
mall_customers.shape
```

Out[4]:
```
(200, 5)
```

Data Wrangling

In [5]:
```
mall_customers.isnull().sum()
```

Out[5]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

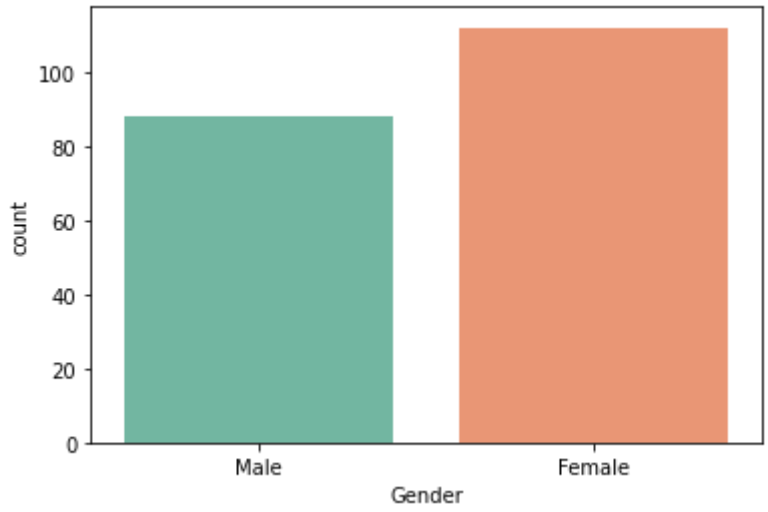
EDA

In [6]:
```
mall_customers["Gender"].value_counts()
```

Out[6]:
```
Female    112
Male       88
Name: Gender, dtype: int64
```

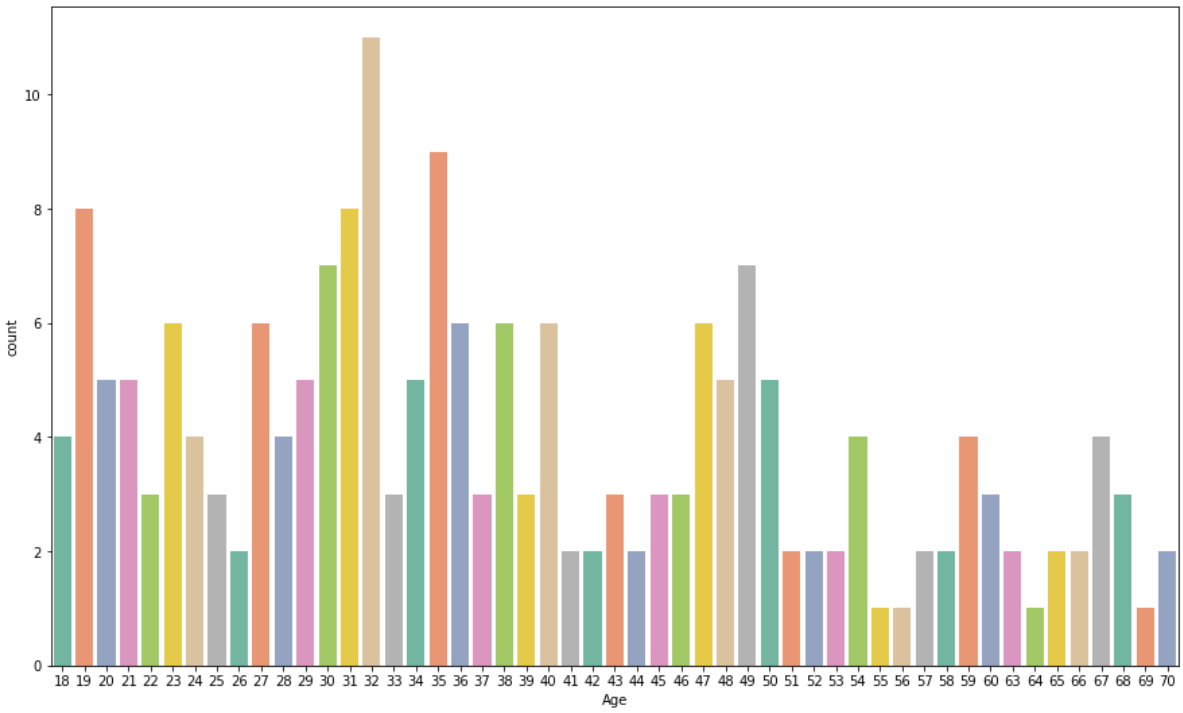
In [7]:
```
# count plot is necessary in clustering
sns.countplot(x="Gender",data = mall_customers, palette = 'Set2')
```

Out[7]:
```
<AxesSubplot:xlabel='Gender', ylabel='count'>
```



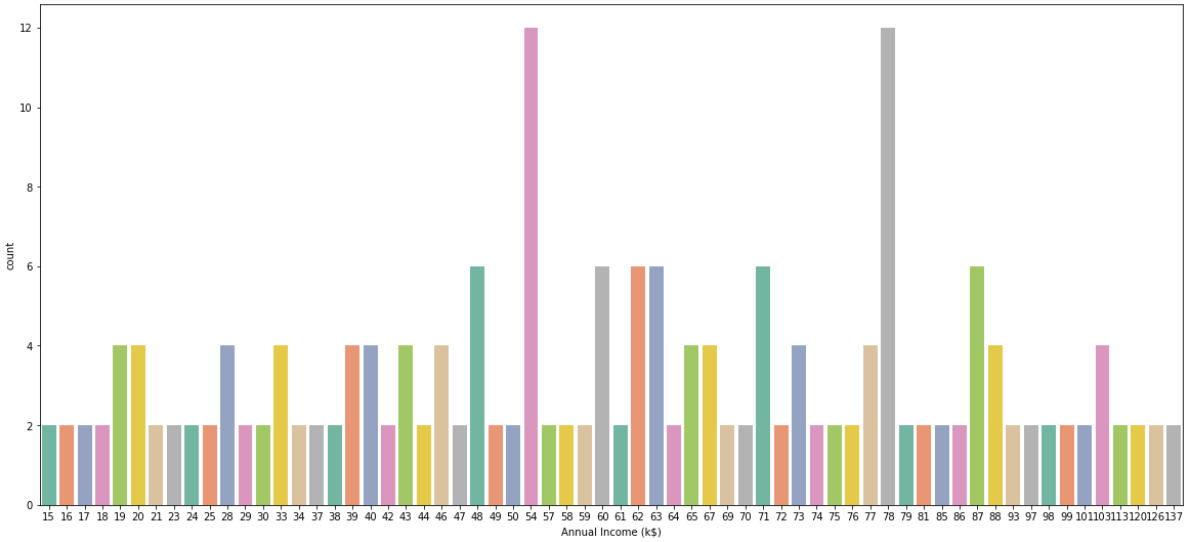
In [8]:
```
plt.figure(figsize=(15,9))
sns.countplot(x="Age",data = mall_customers, palette = 'Set2')
```

Out[8]:
```
<AxesSubplot:xlabel='Age', ylabel='count'>
```



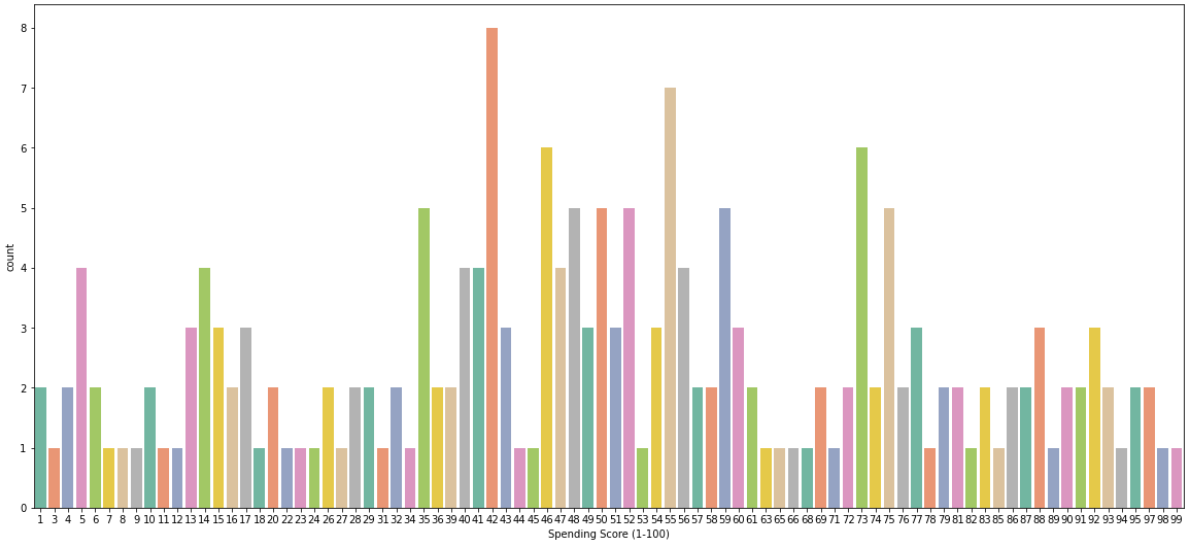
In [9]:
```
plt.figure(figsize=(20,9))
sns.countplot(x="Annual Income (k$)",data = mall_customers, palette = 'Set2')
```

Out[9]:
```
<AxesSubplot:xlabel='Annual Income (k$)', ylabel='count'>
```



In [10]:
```
plt.figure(figsize=(20,9))
sns.countplot(x="Spending Score (1-100)",data = mall_customers, palette = 'Set2')
```

Out[10]:
```
<AxesSubplot:xlabel='Spending Score (1-100)', ylabel='count'>
```



Features

In [11]:
```
# first one is all rows and 2nd one is for columns
X=mall_customers.iloc[:,[3,4]]
```

In [12]:
```
X
```

Out[12]:

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40
...	...	...
195	120	79
196	126	28
197	126	74
198	137	18
199	137	83

200 rows × 2 columns

Elbow Method

- We will try to create a plot which starts from 1-15

In [13]:
```
from sklearn.cluster import KMeans
```

In [14]:
```
# 1st is wcss-  which will be a list
# diffrent values of cluster
wcss = []
# wcss will be a list,range of K 1-16, each value will be passing within the cluste
# kmeans ++- algorithm for selecting the intial data point such that it would be al
# most imp= With the help of kmeans ++, random sample will be choosen, kmeans++ is
# n_init= (no of iteration ) no of times kmeans algorithm will run with diffrent ce
# i - value of clusters, in other words value of k
for i in range(1,16):
    kmeans=KMeans(n_clusters=i, init='k-means++',max_iter = 300, n_init = 10, rando
    kmeans.fit(X)
    # .inertia_gives wcss (within cluster sum of squares)
    wcss.append(kmeans.inertia_)
```

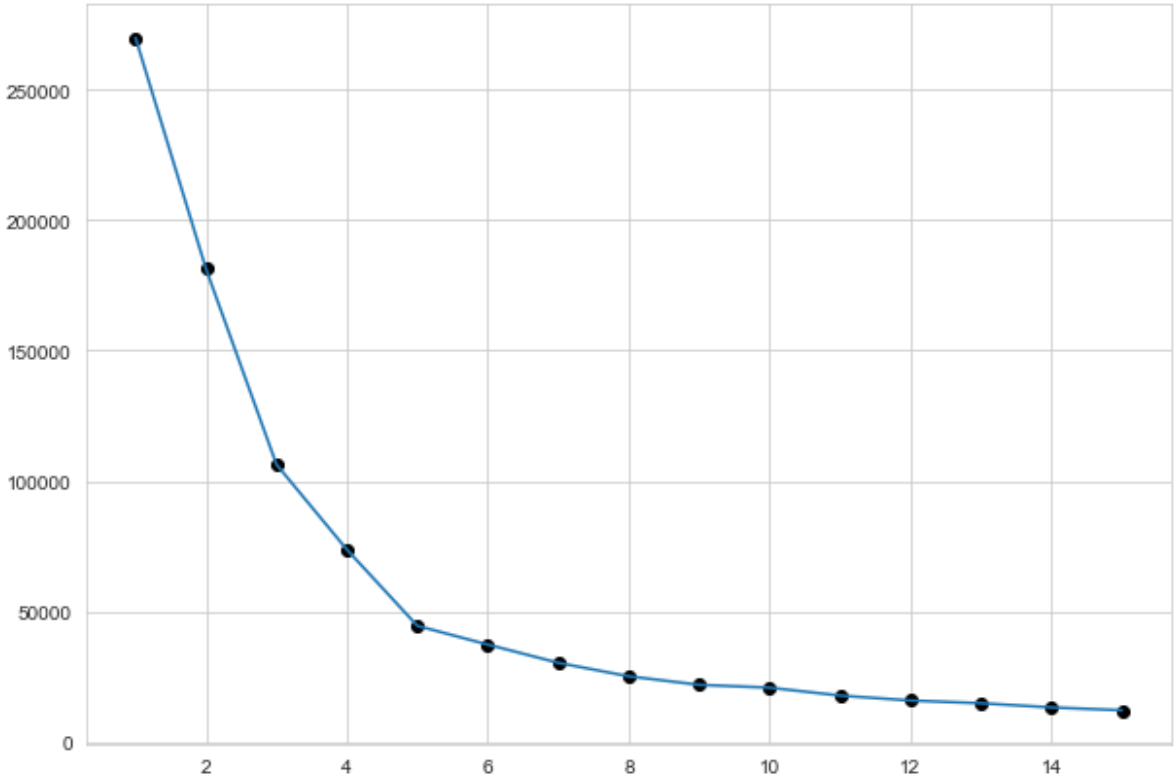
In [15]:
```
wcss
```

Out[15]:
```
[269981.28,
 181363.59595959593,
 106348.37306211118,
 73679.78903948836,
 44448.45544793371,
 37265.86520484347,
 30259.65720728547,
 25095.703209997548,
 21830.041978049438,
 20736.679938924124,
 17702.595932296277,
 15810.838613705502,
 14763.330402558204,
 13165.329070181626,
 12064.939000692291]
```

Value of Elbow Point

In [16]:
```
# Plot visualization b/w WCSS and Number of Clusters (K)
# Next find the value of elbow point
sns.set_style("whitegrid")
plt.figure(figsize=(10,7))
plt.plot(range(1,16), wcss)
plt.scatter(range(1,16),wcss, marker="o",color="k")
```

Out[16]:
```
<matplotlib.collections.PathCollection at 0x19f1a928250>
```



From the above graph - Optimum Value of K = 5

In [17]:
```
# k-means++ : selects initial cluster centres (centroid values/cluster means) for k
# kmeans++ algorithm selects intial data points as clusters in smart way to speed-u
# max_iter = Number of Iterations- iterating kmeans
# n_init = Number of time k-means algorithm will run with different centroids- iter
kmeans = KMeans(n_clusters = 5, init = 'k-means++', max_iter = 300, n_init = 10, r
```

In [18]:
```
kmeans.fit(X)
```

Out[18]:
```
KMeans(n_clusters=5, random_state=0)
```

In [19]:
```
y_clusters = kmeans.predict(X)
y_clusters
# classyfiying the unlabeled data- K=5(0,1,2,3,4)
```

Out[19]:
```
array([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, 1,
       4, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 0, 2, 1, 2, 0, 2, 0, 2,
       1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2])
```

In [20]:
```
# Centroidal value for all 5 clusters
kmeans.cluster_centers_
```

Out[20]:
```
array([[88.2       , 17.11428571],
       [55.2962963 , 49.51851852],
       [86.53846154, 82.12820513],
       [25.72727273, 79.36363636],
       [26.30434783, 20.91304348]])
```

Cluster Visualization

In [21]:
```
# Either we can do simple way or making a dummy cluster
x_cluster= mall_customers.iloc[:,[3,4]]
```

In [22]:
```
x_cluster["Cluster"] =y_clusters
```

In [23]:
```
x_cluster
```

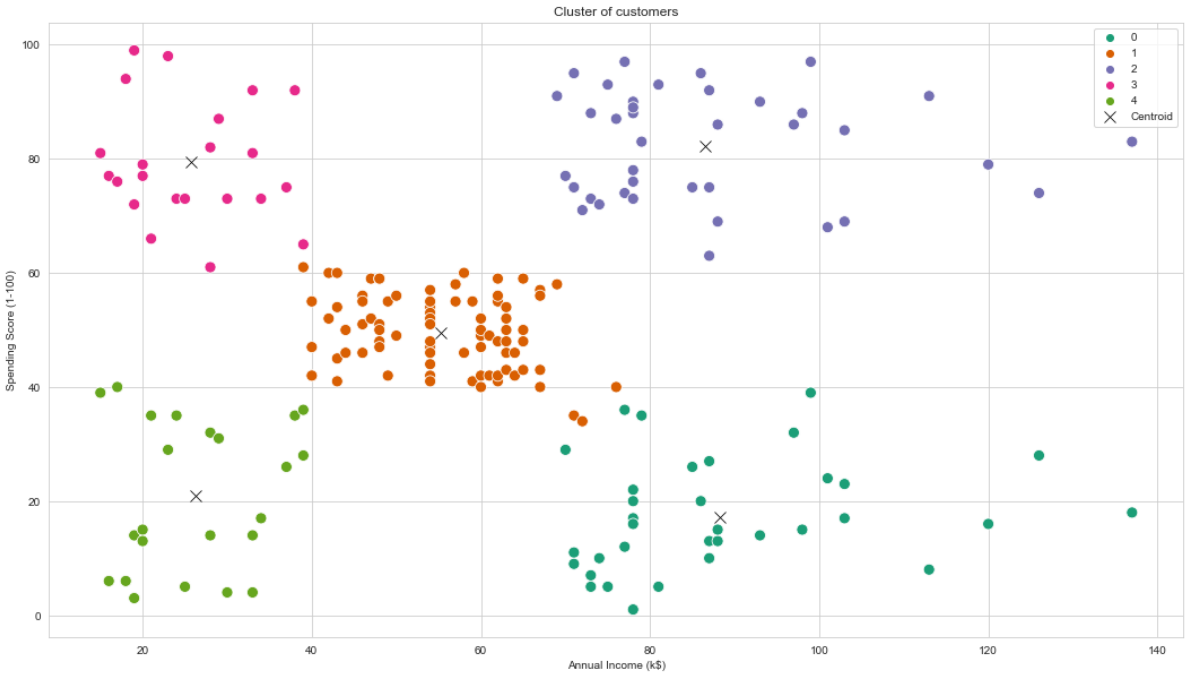
Out[23]:

	Annual Income (k$)	Spending Score (1-100)	Cluster
0	15	39	4
1	15	81	3
2	16	6	4
3	16	77	3
4	17	40	4
...	...	...	...
195	120	79	2
196	126	28	0
197	126	74	2
198	137	18	0
199	137	83	2

200 rows × 3 columns

In [27]:
```
# scatter plot
# s- size of your bubble
```

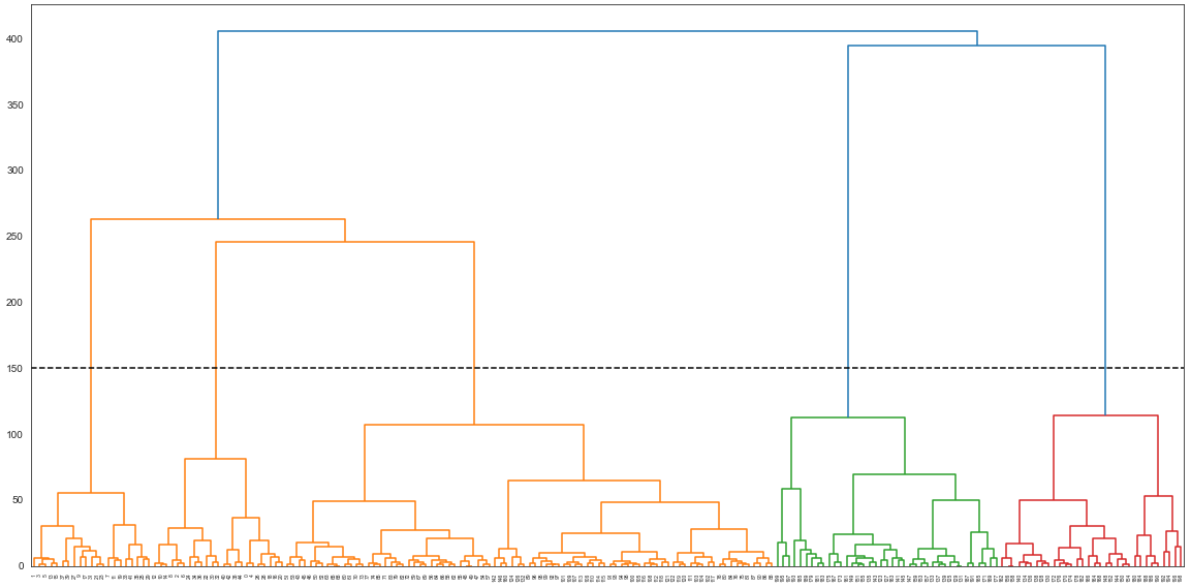
```
plt.figure(figsize=(18,10))
sns.scatterplot(x="Annual Income (k$)",y="Spending Score (1-100)",hue="Cluster",da
# now we have to have centroid value, x=0 and y=1
sns.scatterplot(x=kmeans.cluster_centers_[:,0],y=kmeans.cluster_centers_[:,1],s=10
plt.title("Cluster of customers",size=12)
plt.show()
```



Hierarchical Clustering

In [30]:
```
from seaborn.matrix import dendrogram
import scipy.cluster.hierarchy as sch
```

In [34]:
```
sns.set_style("white")
plt.figure(figsize=(20,10))
dendrogram= sch.dendrogram(sch.linkage(X, method="ward"))
# at the top - all samples are taken as one single clusters
# at the bottom- all the individual samples are clusters in itself
plt.axhline(150, linestyle="--",color="black")
plt.show()
# we can decrease further to 120, 130 we have to create horizontal line just above
```



- In order to find the no of clusters from the above Hierchial structure, will see the longest vertical line and after it we will make a horizontal line

This is an alternative method to Elbow method as sometimes it may be confusing, to find optimimum no of clusters, though Dendogram is used for Hierchial cluster

- No of cluster =5, since horizontal line is cutting longest vertical lines at 5 points

Creating model for prediction

In [35]:
```
from sklearn.cluster import AgglomerativeClustering
# linkage = ward, avg, min,complete(max)
hc = AgglomerativeClustering(n_clusters = 5, affinity = 'euclidean', linkage = 'wa
```

In [37]:
```
y_hc=hc.fit_predict(X)
```

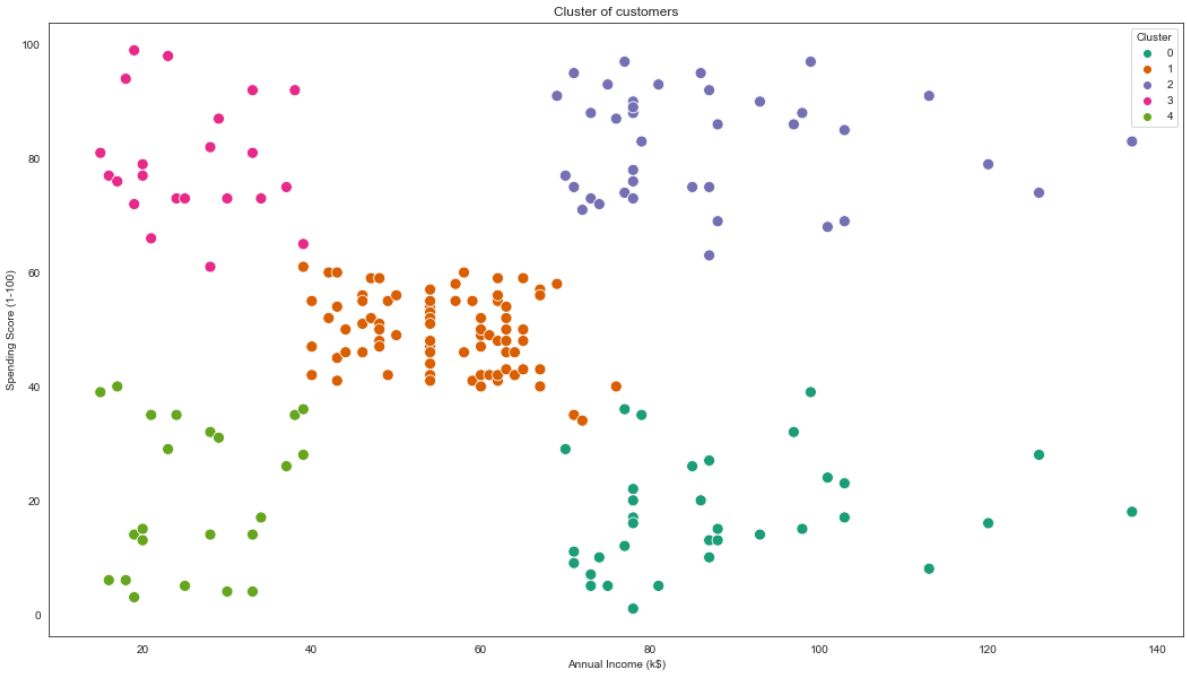
In [38]:
```
y_hc
```

Out[38]:
```
array([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, 1,
       4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 0, 2, 0, 2,
       1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
       0, 2], dtype=int64)
```

In [41]:
```
X=mall_customers.iloc[:,[3,4]]
```

In [42]:
```
y_clusters = y_hc
# replace with our predicted cluster
```

In [43]:
```
plt.figure(figsize=(18,10))
sns.scatterplot(x="Annual Income (k$)",y="Spending Score (1-100)",hue="Cluster",da
# now we have to have centroid value, x=0 and y=1
plt.title("Cluster of customers",size=12)
plt.show()
```


Cluster of customers
Cluster
0
1
2
3
4
Spending Score (1-100)
Annual Income (k$)
0
20
40
60
80
100
20
40
60
80
100
120
140

In [ ]: