In [2]:

*#Name:Pawar ved balasaheb(T512037)*

**import** numpy **as** np

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** warnings

warnings**.**filterwarnings('ignore')

In [4]:

df **=** pd**.**read\_csv("Mall\_Customers.csv")

In [6]:

df**.**head()

Out[6]:

**CustomerID Genre Age Annual Income (k$) Spending Score (1-100) 0** 1 Male 19 15 39

**2** 3 Female 20 16 6

**1**

2 Male 21

15

81

**3**

4 Female 23

16

77

**4** 5 Female 31 17 40

In [8]:

df**.**tail()

Out[8]:

**CustomerID Genre Age Annual Income (k$) Spending Score (1-100) 195** 196 Female 35 120 79

**197** 198 Male 32 126 74

**196**

197 Female 45

126

28

**198**

199 Male 32

137

18

**199** 200 Male 30 137 83

In [10]:

df**.**shape

Out[10]:

In [12]:

Out[12]:

In [14]:

In [16]:

df

(200, 5)

df**.**columns

Index(['CustomerID', 'Genre', 'Age', 'Annual Income (k$)', 'Spending Score (1-100)'],

dtype='object')

df**.**drop("CustomerID",axis**=**1,inplace**=True**)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Out[16]: |  | **Genre** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
|  | **0** | Male | 19 | 15 | 39 |
|  | **1** | Male | 21 | 15 | 81 |
|  | **2** | Female | 20 | 16 | 6 |
|  | **3** | Female | 23 | 16 | 77 |
|  | **4** | Female | 31 | 17 | 40 |
|  | **...** | ... | ... | ... | ... |
|  | **195** | Female | 35 | 120 | 79 |
|  | **196** | Female | 45 | 126 | 28 |
|  | **197** | Male | 32 | 126 | 74 |
|  | **198** | Male | 32 | 137 | 18 |
|  | **199** | Male | 30 | 137 | 83 |

# 200 rows × 4 columns

In [18]:

print("Missing values:") df**.**isnull()**.**sum()

Out[18]:

In [20]:

Missing values:

Genre 0

Age 0

Annual Income (k$) 0

Spending Score (1-100) 0

dtype: int64

df**.**describe()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Out[20]: |  | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
|  | **count** | 200.000000 | 200.000000 | 200.000000 |
|  | **mean** | 38.850000 | 60.560000 | 50.200000 |
|  | **std** | 13.969007 | 26.264721 | 25.823522 |
|  | **min** | 18.000000 | 15.000000 | 1.000000 |
|  | **25%** | 28.750000 | 41.500000 | 34.750000 |
|  | **50%** | 36.000000 | 61.500000 | 50.000000 |
|  | **75%** | 49.000000 | 78.000000 | 73.000000 |
|  | **max** | 70.000000 | 137.000000 | 99.000000 |

In [22]:

df**.**info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 200 entries, 0 to 199

Data columns (total 4 columns):

# Column Non-Null Count Dtype

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  | Genre |  | 200 | non-null |  | object |
| 1 |  | Age |  | 200 | non-null |  | int64 |
| 2 |  | Annual | Income (k$) | 200 | non-null |  | int64 |

3 Spending Score (1-100) 200 non-null int64 dtypes: int64(3), object(1)

memory usage: 6.4+ KB

In [24]:

df**.**nunique()

Out[24]:

In [34]:

In [36]:

df['Genre']**.**value\_counts()**.**plot(kind**=**'pie',figsize**=**(5,5),autopct**=**'%1.1f%%') plt**.**title("Total Gender Count")

plt**.**show()

Genre 2

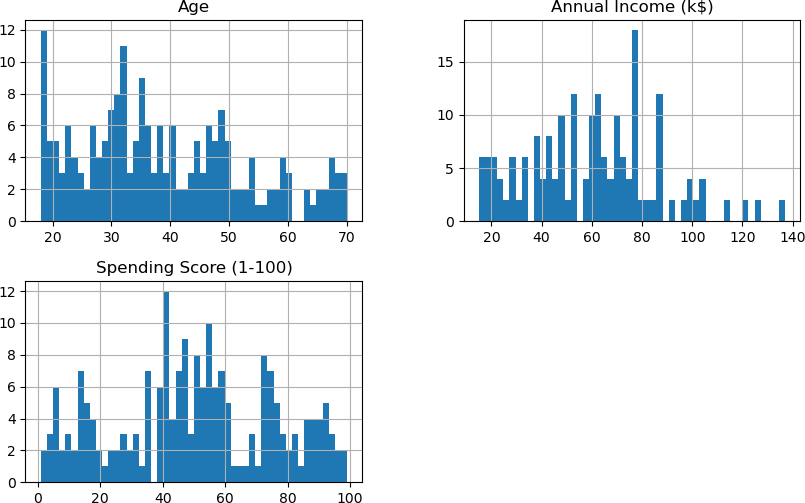
Age 51

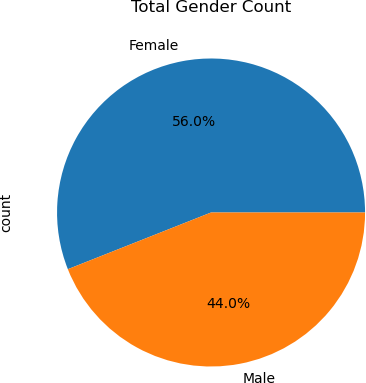
Annual Income (k$) 64

Spending Score (1-100) 84

dtype: int64

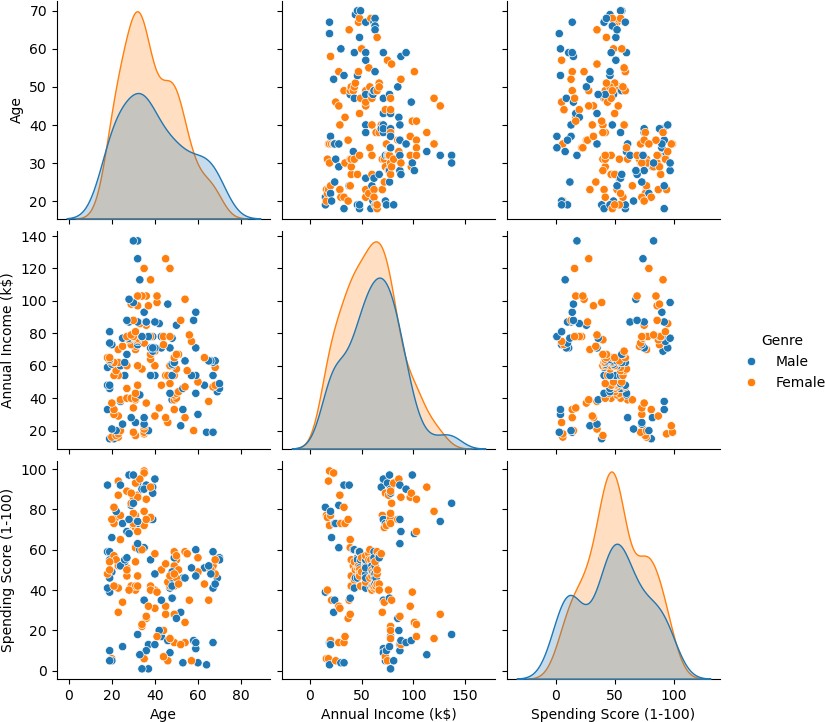
df**.**hist(bins **=** 50,figsize **=** (10,6));





In [38]:

sns**.**pairplot(df,hue**=**"Genre");

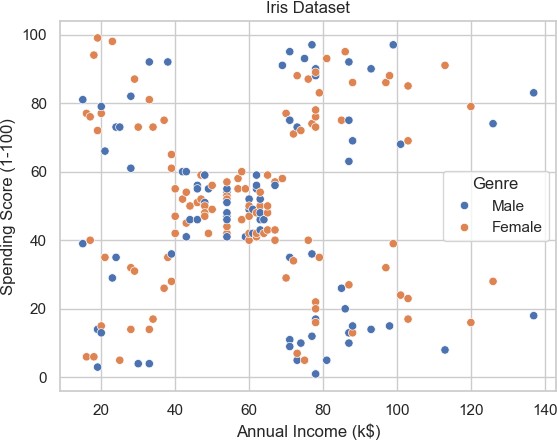


In [40]:

sns**.**set(style **=** 'whitegrid')

sns**.**scatterplot(y **=** 'Spending Score (1-100)',x **=**'Annual Income (k$)',data **=** df,h plt**.**title('Iris Dataset')

plt**.**show()



In [42]:

*# LabelEncoder for encoding binary categories in a column*

**from** sklearn.preprocessing **import** LabelEncoder

**from** sklearn **import** metrics le **=** LabelEncoder()

*# One single vector so it is ovbious what we want to encode*

df["Genre"] **=** le**.**fit\_transform(df["Genre"])

In [44]:

df

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Out[44]: |  | **Genre** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
|  | **0** | 1 | 19 | 15 | 39 |
|  | **1** | 1 | 21 | 15 | 81 |
|  | **2** | 0 | 20 | 16 | 6 |
|  | **3** | 0 | 23 | 16 | 77 |
|  | **4** | 0 | 31 | 17 | 40 |
|  | **...** | ... | ... | ... | ... |
|  | **195** | 0 | 35 | 120 | 79 |
|  | **196** | 0 | 45 | 126 | 28 |
|  | **197** | 1 | 32 | 126 | 74 |
|  | **198** | 1 | 32 | 137 | 18 |
|  | **199** | 1 | 30 | 137 | 83 |

# 200 rows × 4 columns

In [46]:

*# Finding the optimum number of clusters using k-means*

data **=** df**.**copy()

x **=** data**.**iloc[:,[2,3]]

*#importing Kmean model*

**from** sklearn.cluster **import** KMeans wcss **=** []

**for** i **in** range(1,11):

kmeans **=** KMeans(n\_clusters**=**i, init**=**'k-means++', random\_state**=**42) kmeans**.**fit(x)

*# appending the WCSS to the list*

*#(kmeans.inertia\_ returns the WCSS value for an initialized cluster)*

wcss**.**append(kmeans**.**inertia\_)

print('k:',i ,"-> wcss:",kmeans**.**inertia\_)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k: | 1 | -> | wcss: | 269981.28 |
| k: | 2 | -> | wcss: | 183653.32894736837 |
| k: | 3 | -> | wcss: | 106348.37306211118 |
| k: | 4 | -> | wcss: | 73880.64496247197 |
| k: | 5 | -> | wcss: | 44448.45544793371 |
| k: | 6 | -> | wcss: | 40825.16946386946 |
| k: | 7 | -> | wcss: | 33642.579220779226 |
| k: | 8 | -> | wcss: | 26686.83778518779 |
| k: | 9 | -> | wcss: | 24766.47160979344 |

k: 10 -> wcss: 23103.122085983916

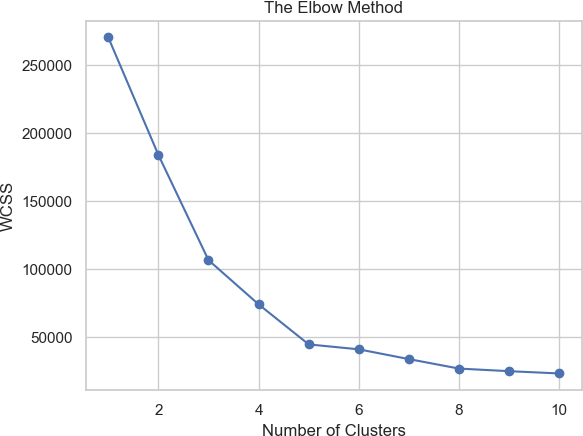
In [48]:

*# Plotting the results onto a line graph, allowing us to observe 'The elbow'*

plt**.**plot(range(1,11),wcss,marker**=**'o') plt**.**title('The Elbow Method')

plt**.**xlabel('Number of Clusters') plt**.**ylabel('WCSS')

plt**.**show()



In [50]:

*#Taking 5 clusters*

km1**=**KMeans(n\_clusters**=**5) *#Fitting the input data* km1**.**fit(data)

*#predicting the labels of the input data*

y**=**km1**.**predict(data)

*#adding the labels to a column named label*

data["label"] **=** y

*#The new dataframe with the clustering done*

data**.**head()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Out[50]: | **Genre** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** | **label** |
|  | **0** 1 | 19 | 15 | 39 | 2 |
|  | **1** 1 | 21 | 15 | 81 | 4 |
|  | **2** 0 | 20 | 16 | 6 | 2 |
|  | **3** 0 | 23 | 16 | 77 | 4 |
|  | **4** 0 | 31 | 17 | 40 | 2 |

In [52]:

*#Scatterplot of the clusters*

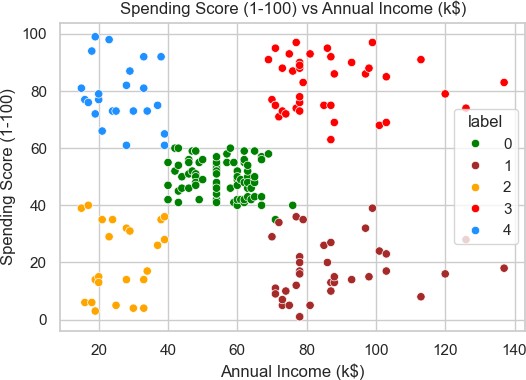
plt**.**figure(figsize**=**(6,4))

sns**.**scatterplot(x **=** 'Annual Income (k$)',y **=** 'Spending Score (1-100)',hue**=**"label palette**=**['green','brown','orange','red','dodgerblue'],data **=** da

plt**.**xlabel('Annual Income (k$)')

plt**.**ylabel('Spending Score (1-100)')

plt**.**title('Spending Score (1-100) vs Annual Income (k$)') plt**.**show()



|  |  |  |
| --- | --- | --- |
| In | [54]: | X**=**data**.**iloc[:,:4]  y**=**data**.**iloc[:,**-**1] |
|  |  |  |
| In | [56]: | **from** sklearn.model\_selection **import** train\_test\_split  X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.2, random\_ |
|  |  | *# Shape of train Test Split*  print(X\_train**.**shape,y\_train**.**shape) print(X\_test**.**shape,y\_test**.**shape) |

(160, 4) (160,)

(40, 4) (40,)

In [58]:

**from** sklearn.cluster **import** KMeans km**=**KMeans(n\_clusters**=**5)

km**.**fit(X\_train)

*#predicting the target value from the model for the samples*

y\_train\_km **=** km**.**predict(X\_train) y\_test\_km **=** km**.**predict(X\_test)

In [60]:

**from** sklearn.metrics.cluster **import** adjusted\_rand\_score

acc\_train\_gmm **=** adjusted\_rand\_score(y\_train,y\_train\_km) acc\_test\_gmm **=** adjusted\_rand\_score(y\_test,y\_test\_km)

print("K mean : Accuracy on training Data: {:.3f}"**.**format(acc\_train\_gmm)) print("K mean : Accuracy on test Data: {:.3f}"**.**format(acc\_test\_gmm))

K mean : Accuracy on training Data: 0.965 K mean : Accuracy on test Data: 0.912

In [62]:

data **=** df**.**copy()

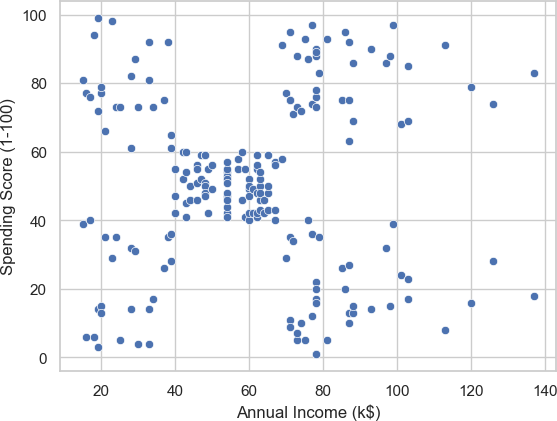
data **=** data**.**iloc[:,[2,3]] data

|  |  |  |  |
| --- | --- | --- | --- |
| Out[62]: |  | **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

In [64]:

sns**.**scatterplot(x**=**"Annual Income (k$)",y**=**"Spending Score (1-100)",data **=** data );



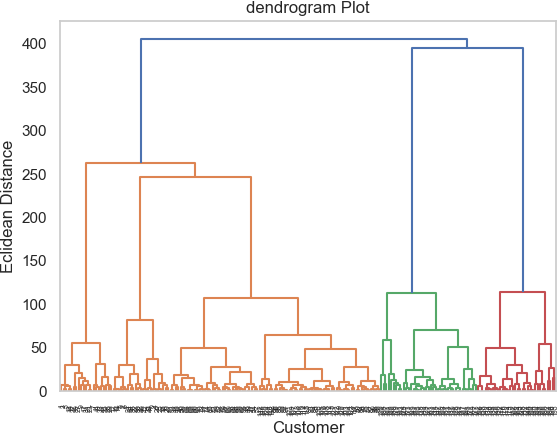
In [66]:

**import** scipy.cluster.hierarchy **as** shc

dendrogram **=** shc**.**dendrogram(shc**.**linkage(data,method**=**"ward")) plt**.**title("dendrogram Plot")

plt**.**xlabel("Customer")

plt**.**ylabel("Eclidean Distance") plt**.**grid(**False**)



In [68]:

**from** sklearn.cluster **import** AgglomerativeClustering agc **=** AgglomerativeClustering(n\_clusters**=**5)

data["label"] **=** agc**.**fit\_predict(data) data

|  |  |  |  |
| --- | --- | --- | --- |
| Out[68]: | **Annual Income (k$)** | **Spending Score (1-100)** | **label** |
|  | **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 [70]:

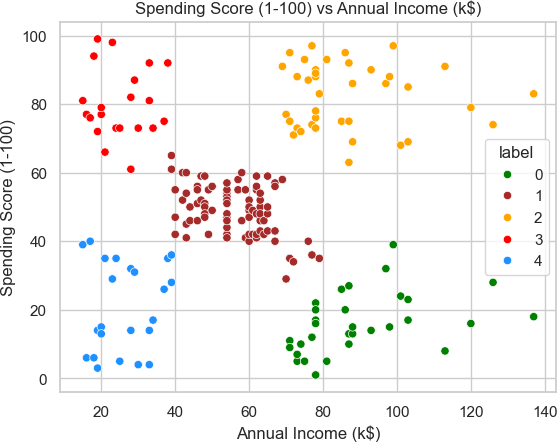
*#Scatterplot of the clusters*

sns**.**scatterplot(x **=** 'Annual Income (k$)',y **=** 'Spending Score (1-100)',hue**=**"label

palette**=**['green','brown','orange','red','dodgerblue'],data **=** da plt**.**xlabel('Annual Income (k$)')

plt**.**ylabel('Spending Score (1-100)')

plt**.**title('Spending Score (1-100) vs Annual Income (k$)') plt**.**show()



In [ ]: