Customer Segmentation is the subdivision of a market into discrete customer groups that share similar characteristics. Customer Segmentation can be a powerful means to identify unsatisfied customer needs. Using the above data companies can then outperform the competition by developing uniquely appealing products and services.

The most common ways in which businesses segment their customer base are:

1. **Demographic information**, such as gender, age, familial and marital status, income, education, and occupation.
2. **Geographical information**, which differs depending on the scope of the company. For localized businesses, this info might pertain to specific towns or counties. For larger companies, it might mean a customer’s city, state, or even country of residence.
3. **Psychographics**, such as social class, lifestyle, and personality traits.
4. **Behavioral data**, such as spending and consumption habits, product/service usage, and desired benefits.

**Advantages of Customer Segmentation**

1. Determine appropriate product pricing.
2. Develop customized marketing campaigns.
3. Design an optimal distribution strategy.
4. Choose specific product features for deployment.
5. Prioritize new product development efforts.

**K Means Clustering Algorithm**

1. Specify number of clusters *K*.
2. Initialize centroids by first shuffling the dataset and then randomly selecting *K*data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn’t changing.



**The Challenge**

You own a supermarket mall and through membership cards, you have some basic data about your customers like Customer ID, age, gender, annual income and spending score. You want to understand the customers like who are the target customers so that the sense can be given to marketing team and plan the strategy accordingly.

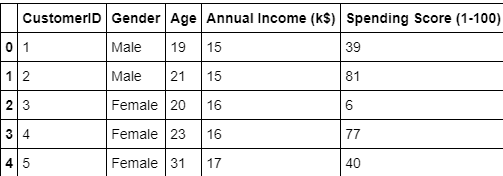
**Environment and tools**

1. scikit-learn
2. seaborn
3. numpy
4. pandas
5. matplotlib

Initialization:

Load needed libraries and analyze data

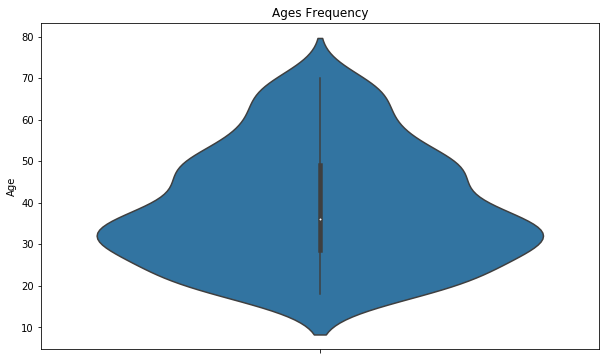
|  |  |
| --- | --- |
|  | import numpy as np |
|  | import pandas as pd |
|  | import matplotlib.pyplot as plt |
|  | import seaborn as sns |
|  |  |
|  | df = pd.read\_csv("../input/customer-segmentation-tutorial-in-python/Mall\_Customers.csv") |
|  | df.head() |



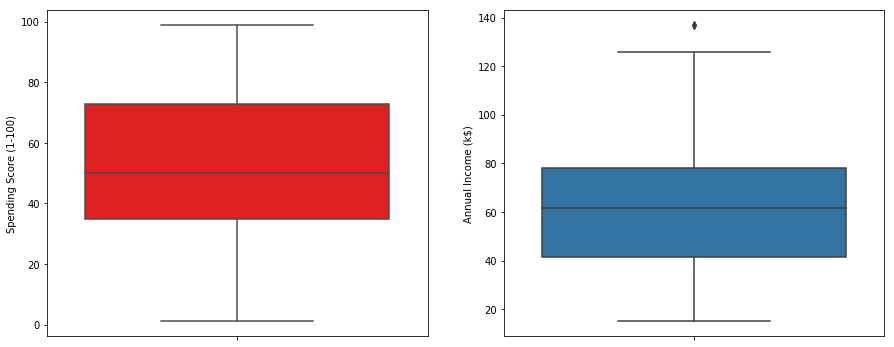
Visualization:

Remove irrelevant data and make a graph for better visualized understanding

|  |  |
| --- | --- |
|  | df.drop(["CustomerID"], axis = 1, inplace=True) |
|  |  |
|  | plt.figure(figsize=(10,6)) |
|  | plt.title("Ages Frequency") |
|  | sns.axes\_style("dark") |
|  | sns.violinplot(y=df["Age"]) |
|  | plt.show() |

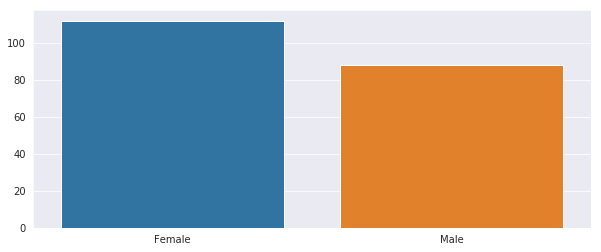
 A box plot of spending score and annual income

|  |  |
| --- | --- |
|  | *plt.figure(figsize=(15,6))* |
|  | *plt.subplot(1,2,1)* |
|  | *sns.boxplot(y=df["Spending Score (1-100)"], color="red")* |
|  | *plt.subplot(1,2,2)* |
|  | *sns.boxplot(y=df["Annual Income (k$)"])* |
|  | *plt.show()* |



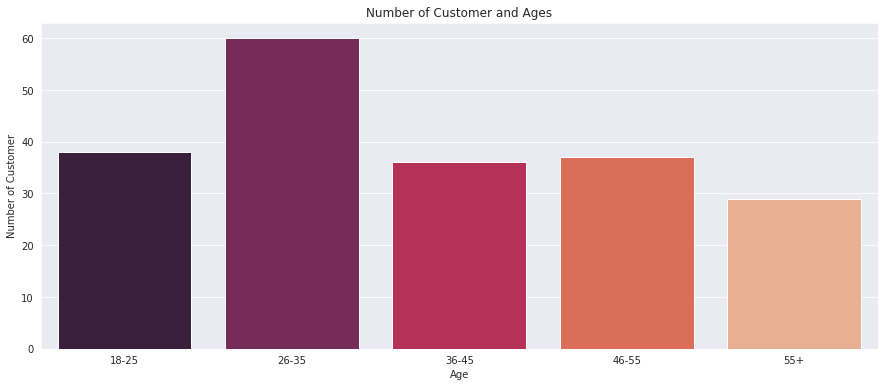
a bar plot to check the distribution of male and female population in the dataset.

|  |  |
| --- | --- |
|  | genders = df.Gender.value\_counts() |
|  | sns.set\_style("darkgrid") |
|  | plt.figure(figsize=(10,4)) |
|  | sns.barplot(x=genders.index, y=genders.values) |
|  | plt.show() |



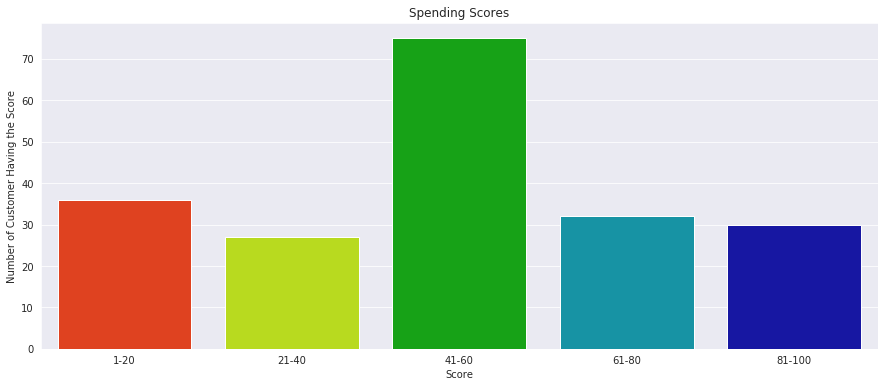
bar plot to check the distribution of number of customers in each age group.

|  |  |
| --- | --- |
|  | age18\_25 = df.Age[(df.Age <= 25) & (df.Age >= 18)] |
|  | age26\_35 = df.Age[(df.Age <= 35) & (df.Age >= 26)] |
|  | age36\_45 = df.Age[(df.Age <= 45) & (df.Age >= 36)] |
|  | age46\_55 = df.Age[(df.Age <= 55) & (df.Age >= 46)] |
|  | age55above = df.Age[df.Age >= 56] |
|  |  |
|  | x = ["18-25","26-35","36-45","46-55","55+"] |
|  | y = [len(age18\_25.values),len(age26\_35.values),len(age36\_45.values),len(age46\_55.values),len(age55above.values)] |
|  |  |
|  | plt.figure(figsize=(15,6)) |
|  | sns.barplot(x=x, y=y, palette="rocket") |
|  | plt.title("Number of Customer and Ages") |
|  | plt.xlabel("Age") |
|  | plt.ylabel("Number of Customer") |
|  | plt.show() |



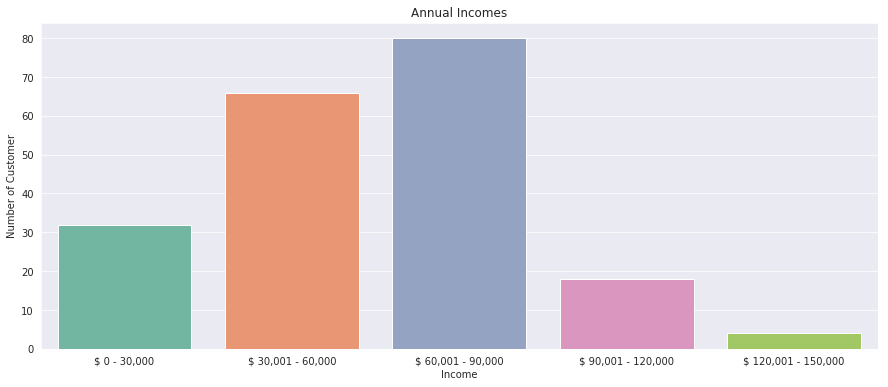
a bar plot to visualize the number of customers according to their spending scores

|  |  |
| --- | --- |
|  | ss1\_20 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 1) & (df["Spending Score (1-100)"] <= 20)] |
|  | ss21\_40 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 21) & (df["Spending Score (1-100)"] <= 40)] |
|  | ss41\_60 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 41) & (df["Spending Score (1-100)"] <= 60)] |
|  | ss61\_80 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 61) & (df["Spending Score (1-100)"] <= 80)] |
|  | ss81\_100 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 81) & (df["Spending Score (1-100)"] <= 100)] |
|  |  |
|  | ssx = ["1-20", "21-40", "41-60", "61-80", "81-100"] |
|  | ssy = [len(ss1\_20.values), len(ss21\_40.values), len(ss41\_60.values), len(ss61\_80.values), len(ss81\_100.values)] |
|  |  |
|  | plt.figure(figsize=(15,6)) |
|  | sns.barplot(x=ssx, y=ssy, palette="nipy\_spectral\_r") |
|  | plt.title("Spending Scores") |
|  | plt.xlabel("Score") |
|  | plt.ylabel("Number of Customer Having the Score") |
|  | plt.show() |

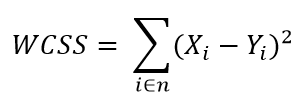


bar plot to visualize the number of customers according to their annual income.

|  |  |
| --- | --- |
|  | ai0\_30 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 0) & (df["Annual Income (k$)"] <= 30)] |
|  | ai31\_60 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 31) & (df["Annual Income (k$)"] <= 60)] |
|  | ai61\_90 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 61) & (df["Annual Income (k$)"] <= 90)] |
|  | ai91\_120 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 91) & (df["Annual Income (k$)"] <= 120)] |
|  | ai121\_150 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 121) & (df["Annual Income (k$)"] <= 150)] |
|  |  |
|  | aix = ["$ 0 - 30,000", "$ 30,001 - 60,000", "$ 60,001 - 90,000", "$ 90,001 - 120,000", "$ 120,001 - 150,000"] |
|  | aiy = [len(ai0\_30.values), len(ai31\_60.values), len(ai61\_90.values), len(ai91\_120.values), len(ai121\_150.values)] |
|  |  |
|  | plt.figure(figsize=(15,6)) |
|  | sns.barplot(x=aix, y=aiy, palette="Set2") |
|  | plt.title("Annual Incomes") |
|  | plt.xlabel("Income") |
|  | plt.ylabel("Number of Customer") |
|  | plt.show() |

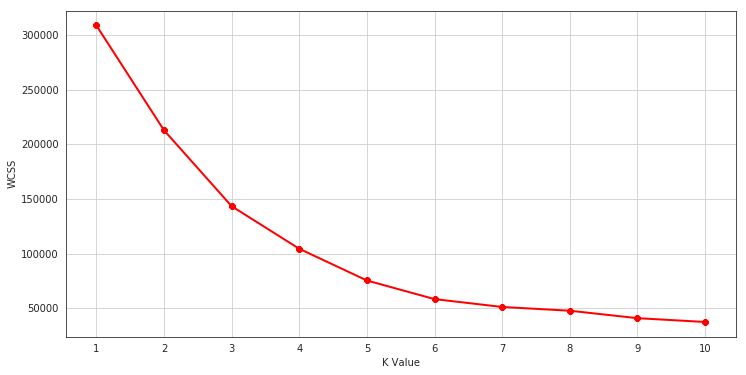


WCSS measures sum of distances of observations from their cluster centroids which is given by the below formula.



where *Yi* is centroid for observation *Xi*. The main goal is to maximize number of clusters and in limiting case each data point becomes its own cluster centroid.

|  |  |
| --- | --- |
|  | from sklearn.cluster import KMeans |
|  | wcss = [] |
|  | for k in range(1,11): |
|  | kmeans = KMeans(n\_clusters=k, init="k-means++") |
|  | kmeans.fit(df.iloc[:,1:]) |
|  | wcss.append(kmeans.inertia\_) |
|  | plt.figure(figsize=(12,6)) |
|  | plt.grid() |
|  | plt.plot(range(1,11),wcss, linewidth=2, color="red", marker ="8") |
|  | plt.xlabel("K Value") |
|  | plt.xticks(np.arange(1,11,1)) |
|  | plt.ylabel("WCSS") |
|  | plt.show() |



**The Elbow Method**

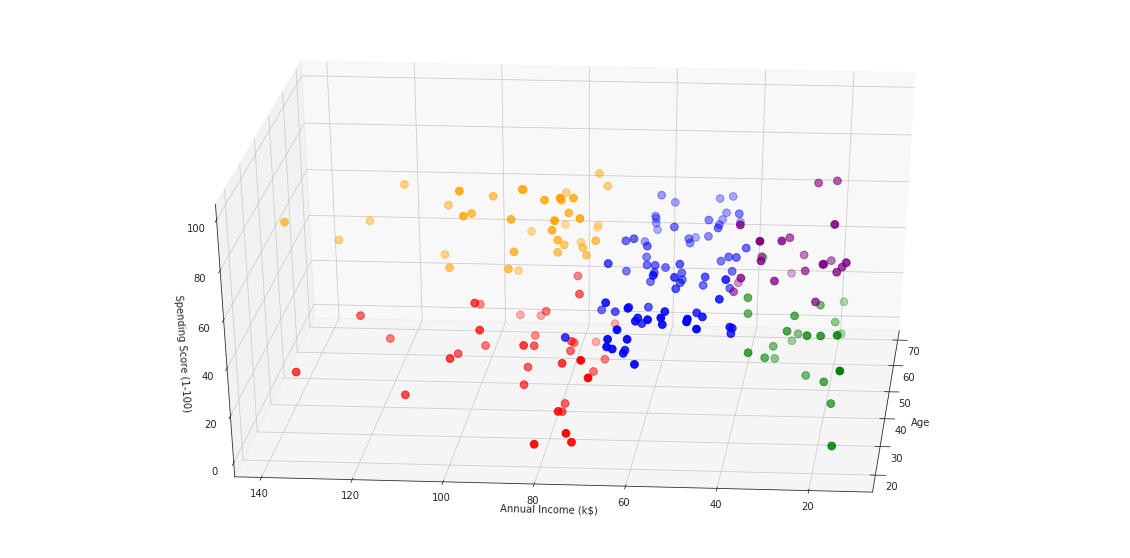
Calculate the Within Cluster Sum of Squared Errors (WSS) for different values of k, and choose the k for which WSS first starts to diminish. In the plot of WSS-versus k, it is visible as an elbow.

The optimal K value is found to be 5 using the elbow method.

3D plot to visualize the spending score of the customers with their annual income. The data points are separated into 5 classes which are represented in different colours as shown in the 3D plot.

|  |  |
| --- | --- |
|  |  |
|  | km = KMeans(n\_clusters=5) |
|  | clusters = km.fit\_predict(df.iloc[:,1:]) |
|  | df["label"] = clusters |
|  |  |
|  | from mpl\_toolkits.mplot3d import Axes3D |
|  | import matplotlib.pyplot as plt |
|  | import numpy as np |
|  | import pandas as pd |
|  |  |
|  | fig = plt.figure(figsize=(20,10)) |
|  | ax = fig.add\_subplot(111, projection='3d') |
|  | ax.scatter(df.Age[df.label == 0], df["Annual Income (k$)"][df.label == 0], df["Spending Score (1-100)"][df.label == 0], c='blue', s=60) |
|  | ax.scatter(df.Age[df.label == 1], df["Annual Income (k$)"][df.label == 1], df["Spending Score (1-100)"][df.label == 1], c='red', s=60) |
|  | ax.scatter(df.Age[df.label == 2], df["Annual Income (k$)"][df.label == 2], df["Spending Score (1-100)"][df.label == 2], c='green', s=60) |
|  | ax.scatter(df.Age[df.label == 3], df["Annual Income (k$)"][df.label == 3], df["Spending Score (1-100)"][df.label == 3], c='orange', s=60) |
|  | ax.scatter(df.Age[df.label == 4], df["Annual Income (k$)"][df.label == 4], df["Spending Score (1-100)"][df.label == 4], c='purple', s=60) |
|  | ax.view\_init(30, 185) |
|  | plt.xlabel("Age") |
|  | plt.ylabel("Annual Income (k$)") |
|  | ax.set\_zlabel('Spending Score (1-100)') |
|  | plt.show() |

**Results**



**Conclusions**

K means clustering is one of the most popular clustering algorithms and usually the first thing practitioners apply when solving clustering tasks to get an idea of the structure of the dataset. The goal of K means is to group data points into distinct non-overlapping subgroups. One of the major application of K means clustering is segmentation of customers to get a better understanding of them which in turn could be used to increase the revenue of the company.