PROJECT REPORT ON
“MALL CUSTOMERS DATASET”

Submitted by:
Nikita Agarwal-TLS21A097

INDEX

INTRODUCTION

Machine Learning is the science of getting computers to learn without being explicitly programmed. It is closely related to computational statistics, which focuses on making prediction using computer. In its application across business problems, machine learning is also referred as predictive analysis. Machine Learning is closely related to computational statistics. Machine Learning focuses on the development of computer programs that can access data and use it to learn themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

The use of machine learning can be seen almost everywhere around us, be it Facebook recognizing you or your friends, or YouTube recommending you a video or two based on your history — Machine Learning is everywhere!

Machine Learning is broadly categorized as **Supervised** and **Unsupervised Learning**.

- **Supervised Learning** is one in which we teach the machine by providing both independent and dependent variables, for example, Classifying or predicting values.
- **Unsupervised Learning** mainly deals with identifying the structure or pattern of the data. In this type of algorithms, we do not have labeled data(or the dependent variable is absent), for example, clustering data, recommendation systems, etc. It provides amazing results as one can deduce many hidden relations between different attributes or features.

Under Supervised learning we come across-

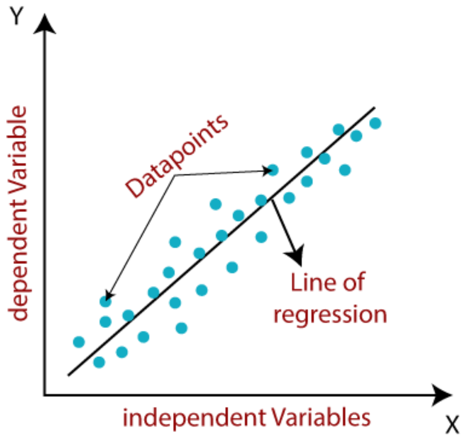
1.1 Linear Regression in Machine Learning

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:

Mall Customer Dataset



Under unsupervised learning we come across:

1.2 Clustering and k means

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The algorithm takes the unlabelled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

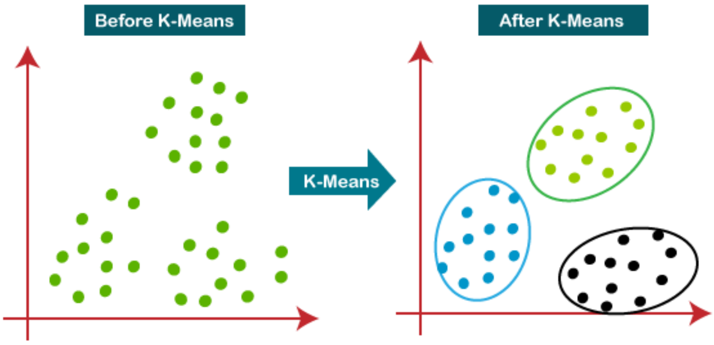
The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

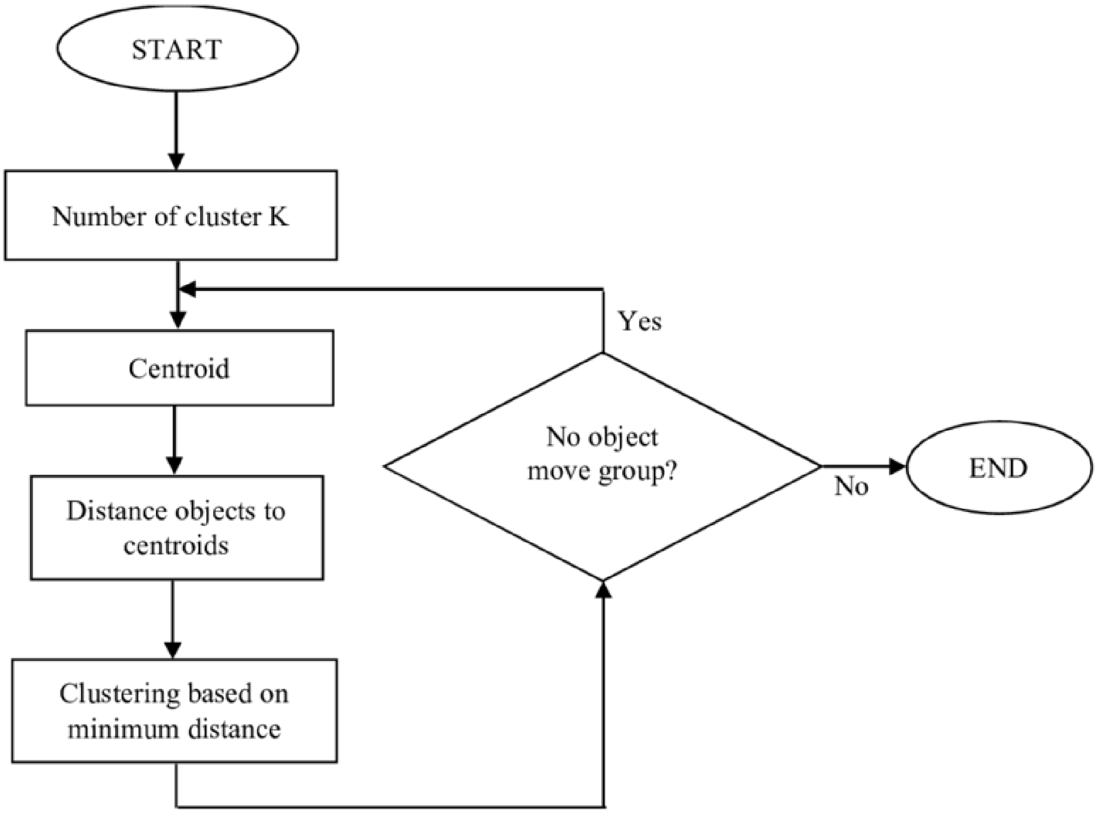
Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:

Mall Customer Dataset



The working of the K-Means algorithm:


START
Number of cluster K
Centroid
Distance objects to centroids
Clustering based on minimum distance
No object move group?
Yes
No
END

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

Libraries used:

pandas - pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python

numpy- numPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

matplotlib- matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

seaborn - Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with dataframes and the Pandas library. The graphs created can also be customized easily

sklearn - Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Mall Customer Dataset

Problem statement:

The Mall customers dataset contains information about people visiting the mall. The dataset has gender, customer id, age, annual income, and spending score. It collects insights from the data and group customers based on their behaviors. Segment the customers based on the age, gender, interest. Customer segmentation is an important practise of dividing customers base into individual groups that are similar. It is useful in customised marketing.

ABOUT THE TOPIC-Mall_Customer Segmentation

Malls or shopping complexes are often indulged in the race to increase their customers and hence making huge profits. To achieve this task machine learning is being applied by many stores already.
It is amazing to realize the fact that how machine learning can aid in such ambitions. The shopping complexes make use of their customers' data and develop ML models to target the right ones. This not only increases sales but also makes the complexes efficient.

Management and maintain of customer relationship have always played a vital role to provide business intelligence to organizations to build, manage and develop valuable long term customer relationships. The importance of treating customers as an organizations main asset is increasing in value in present day and era. Organizations have an interest to invest in the development of customer acquisition, maintenance and development strategies. The business intelligence has a vital role to play in allowing companies to use technical expertise to gain better customer knowledge and Programs for outreach.

By using clustering techniques like k-means, customers with similar means are clustered together.

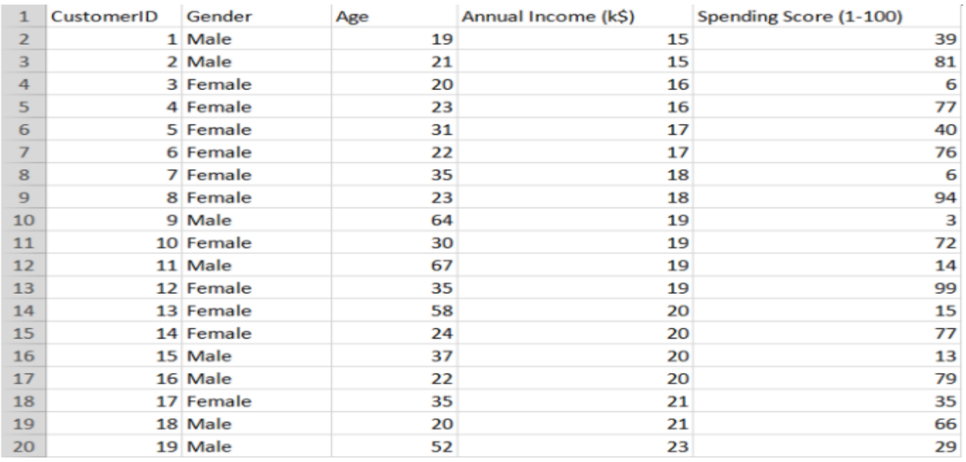
Customer segmentation helps the marketing team to recognize and expose different customer segments that think differently and follow different purchasing strategies.

Customer segmentation helps in figuring out the customers who vary in terms of preferences, expectations, desires and attributes. The main purpose of performing customer segmentation is to group people, who have similar interest so that the marketing team can converge in an effective marketing plan.

Mall Customer Dataset

Clustering is an iterative process of knowledge discovery from vast amounts of raw and unorganized data. Clustering is a type of exploratory data mining that is used in many applications, such as machine learning, classification and pattern recognition.

DataSet:



1	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
2	1	Male	19	15	39
3	2	Male	21	15	81
4	3	Female	20	16	6
5	4	Female	23	16	77
6	5	Female	31	17	40
7	6	Female	22	17	76
8	7	Female	35	18	6
9	8	Female	23	18	94
10	9	Male	64	19	3
11	10	Female	30	19	72
12	11	Male	67	19	14
13	12	Female	35	19	99
14	13	Female	58	20	15
15	14	Female	24	20	77
16	15	Male	37	20	13
17	16	Male	22	20	79
18	17	Female	35	21	35
19	18	Male	20	21	66
20	19	Male	52	23	29

Fig 2.1:mall customers dataset downloaded from kaggle

Here we have the following features :
1. CustomerID: It is the unique ID given to a customer
2. Gender: Gender of the customer
3. Age: The age of the customer
4. Annual Income(k$): It is the annual income of the customer
5. Spending Score: It is the score(out of 100) given to a customer by the mall authorities, based on the money spent and the behaviour of the customer.

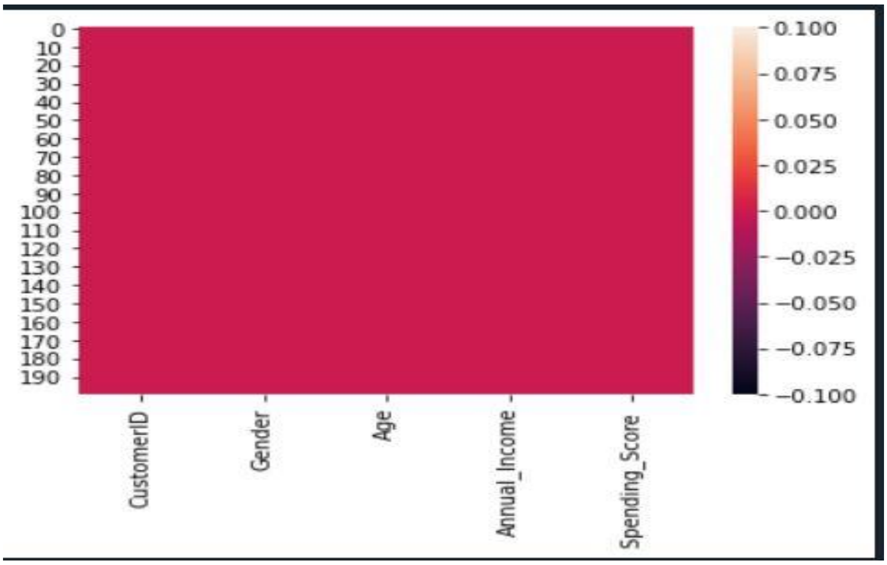
1.DATA CLEANING

Data Cleaning means the process of identifying the incorrect, incomplete, inaccurate, irrelevant or missing part of the data and then modifying, replacing or deleting them according to the necessity. Data cleaning is considered a foundational element of the basic data science.

Data is the most valuable thing for Analytics and Machine learning. In computing or Business data is needed everywhere. When it comes to the real world data, it is not improbable that data may contain incomplete, inconsistent or missing values. If the data is corrupted then it may hinder the process or provide inaccurate results.

1.1 Checking whether there is any empty cells in dataset

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
data=pd.read_csv("Mallcustomers.csv")
sns.heatmap(data.isnull())
plt.show()
```

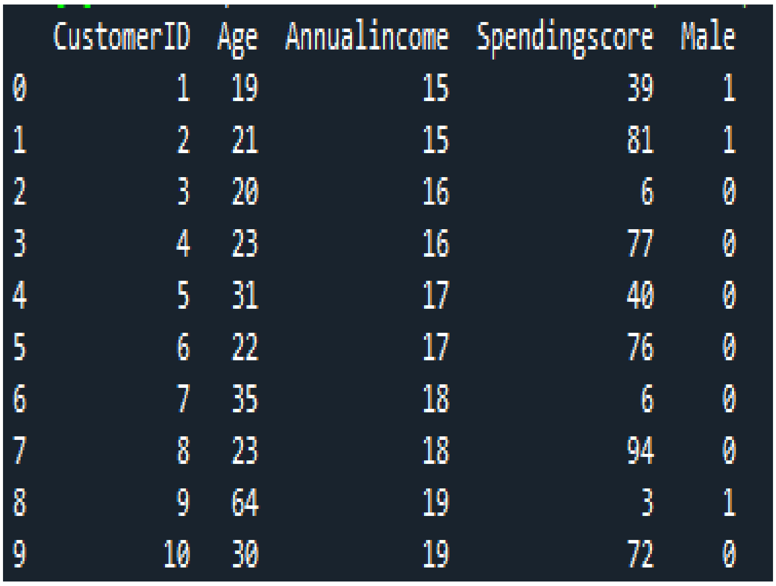


From this graph we got to know that there are no empty cells in data set

1.2.Converting string column(Gender) to integer

get_dummies converts categorical data into dummy or indicator variables. This categorical data encoding method transforms the categorical variable into a set of binary variables (also known as dummy variables). In the case of one-hot encoding, for N categories in a variable, it uses N binary variables. The dummy encoding is a small improvement over one-hot-encoding.

```
gender=pd.get_dummies(data['Gender'],drop_first=True)
data.drop(['Gender'],axis=1,inplace=True)
data=pd.concat([data,gender],axis=1)
print(data[:10])
```



```
   CustomerID  Age  Annualincome  Spendingscore  Male
0           1   19            15             39     1
1           2   21            15             81     1
2           3   20            16              6     0
3           4   23            16             77     0
4           5   31            17             40     0
5           6   22            17             76     0
6           7   35            18              6     0
7           8   23            18             94     0
8           9   64            19              3     1
9          10   30            19             72     0
```

In the above code we have converted the string column(gender) to integer in order to plot the graphs efficiently. Here 1 is assigned for male and 0 for female.

2.DATA VISUALIZATION

Data visualization is the representation of data or information in a graph, chart, or other visual format. It communicates relationships of the data with images. This is important because it allows trends and patterns to be more easily seen. With the rise of big data upon us, we need to be able to interpret increasingly larger batches of data. Machine learning makes it easier to conduct analyses such as predictive analysis, which can then serve as helpful visualizations to present.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
data=pd.read_csv("mallCustomer.csv")
```

2. 1.Visualization of Gender column

```
gen=['Male','Female']
size=data['Gender'].value_counts()
plt.pie(size,labels=gen,autopct="%1.1f%%",explode=(0,0.1),shadow=True)
plt.title('Gender Count')
plt.show()
```

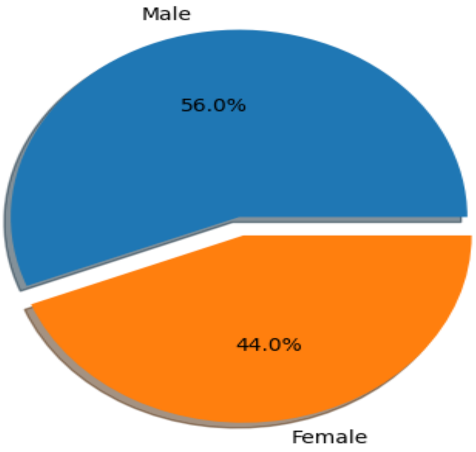


Fig 4.1:piechart for gender count

From this pie-chart we see that male percentage is more when compared to female percentage

2.2 Visualization of Age

```
sns.countplot(data['Age'],palette='hsv',order=[18,22,25,28,30,32,35,38,42,46,50,52,56,60,65,
70])
plt.title("Visualization of Age")
plt.show()
```

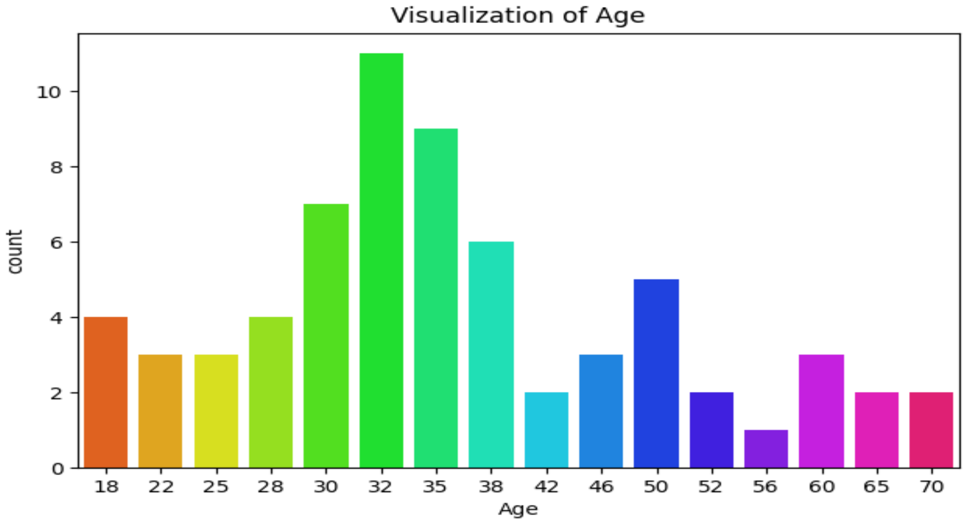


Fig 4.2:countplot for age column

From this countplot graph it’s obvious that the count of people having age group 32-35 are more compared to other age groups

2.3.Visualization of Annual_Income

```
ann=data['Annual Income (k$)']
range1=[15,25,35,45,55,65,75,85,95,105,120,130,140]
plt.hist(ann,range1,histtype='bar',rwidth=0.8,color='r')
plt.title('Visualization of Annual income')
plt.xlabel('Annual income')
plt.ylabel('count')
plt.show()
```

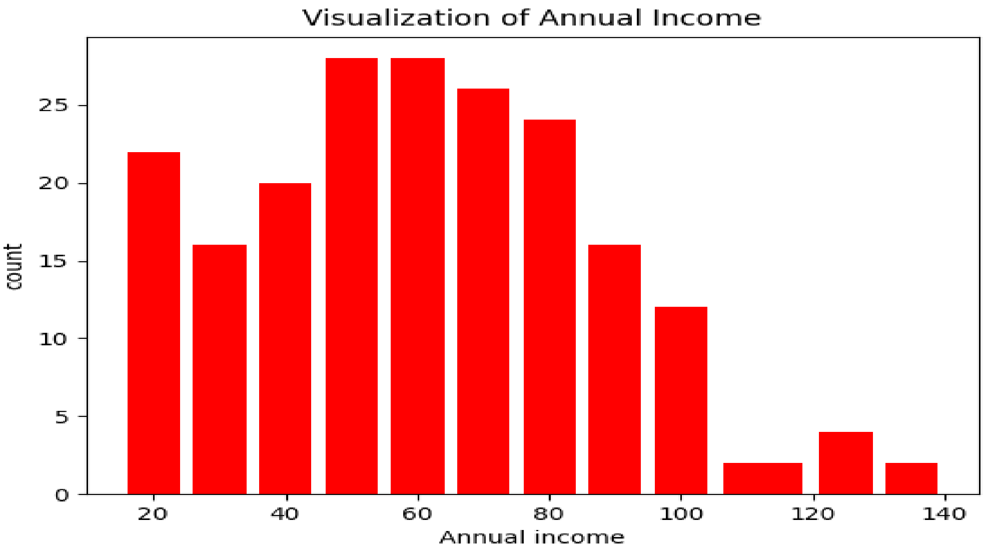


Fig 4.3:histogram graph for annual income visualization

From this histogram plot we can conclude that maximum customers have their annual income around 50 to 60(k$) and very few have more than 100 k$

2.4.Visualization of Spending Score

```
sns.countplot(data['SpendingScore'],palette='copper',order=[3,6,9,12,15,18,24,27,36,39,42,45
,48,51,54,57,60,63,66,69,72,75,78,81,87,90,93,99])
plt.title("Visualization of Spendingscore")
plt.show()
```

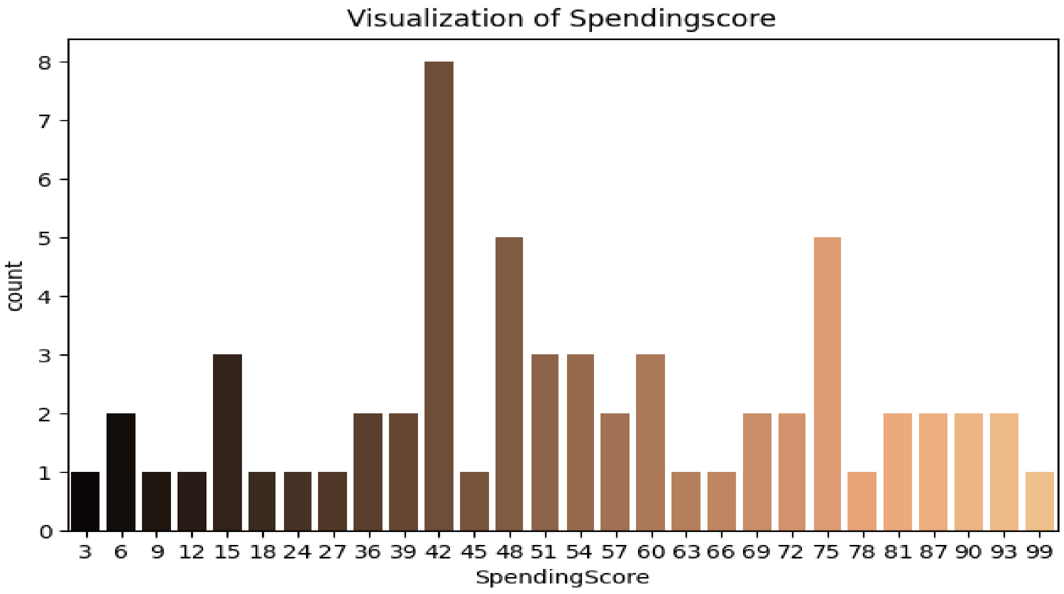


Fig 4.4:countplot for spending score column

From this countplot graph we can say that majority of people’s spending score is around 40 to 42 irrespective of their annual income

2.5.Subplots between income ,spending and age

```
income=data[' Annual Income (k$)']
spending=data[' Spending Score (1-100) ']
x1=income[0:10]
y1=spending[0:10]
plt.subplot(331)
plt.scatter(y1,x1,s=10)
plt.xlabel('income')
plt.ylabel('spending')
plt.title('customer mall')
age=data['Age']
spending=data[' Spending Score (1-100) ']
x=age[0:10]
y=spending[0:10]
plt.subplot(336)
plt.scatter(y,x,s=10)
plt.xlabel('age')
plt.ylabel('spending')
plt.show()
```

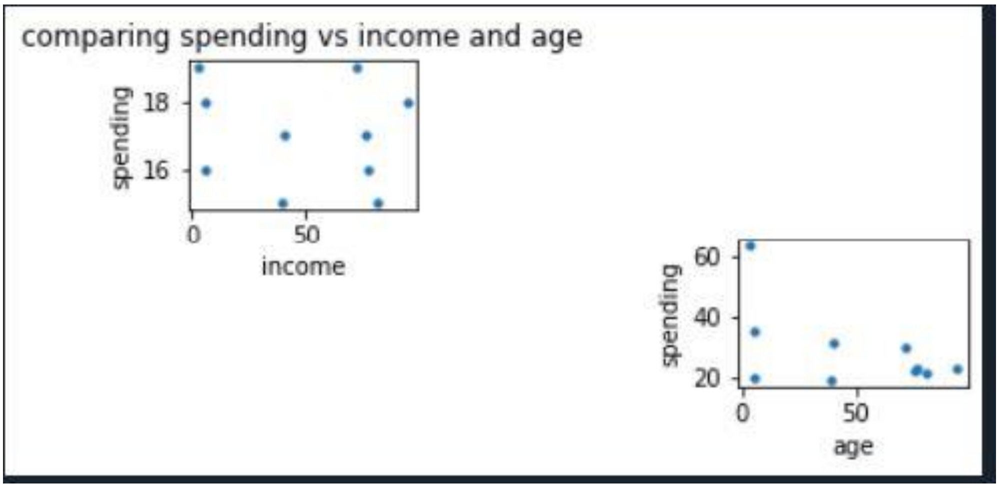


Fig 4.5:multiplots graph for spending score vs annual income and age

From the above subplot we can analyse that early age group spends more than the annual income and as the age increases the spending score decreases slowly though the income remains slightly constant

2.6. Gender vs SpendingScore

```
sns.stripplot(data['Gender'],data['SpendingScore'],palette='Blues',size=5)
plt.title("Gender VS SpendingScore")
plt.show()
```

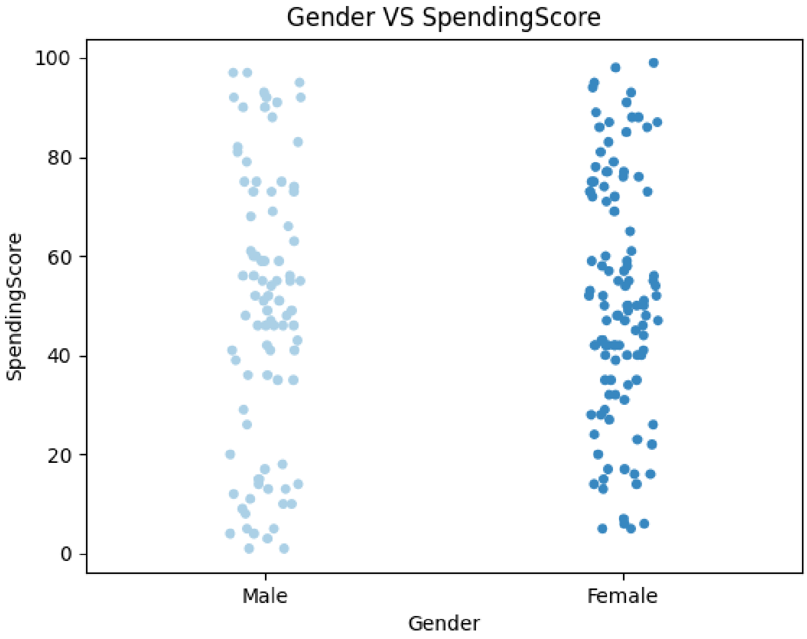


Fig 4.6 :strip plot between gender and spending score

From this plot we can say that female spends more in the range of 40-60 compared to male

2.7.Customers classified based on the range of age

```
age_18_25 = data.Age[(data.Age >= 18) & (data.Age <= 25)]
age_26_35 = data.Age[(data.Age >= 26) & (data.Age <= 35)]
age_36_45 = data.Age[(data.Age >= 36) & (data.Age <= 45)]
age_46_55 = data.Age[(data.Age >= 46) & (data.Age <= 55)]
age_55above = data.Age[data.Age >= 56]
agex = ["18-25","26-35","36-45","46-55","55+"]
agey =
[len(age_18_25.values),len(age_26_35.values),len(age_36_45.values),len(age_46_55.values)
,len(age_55above.values)]
plt.figure(figsize=(15,6))
sns.barplot(x=agex, y=agey, palette="mako")
plt.title("Number of Customer and Ages")
plt.xlabel("Age")
plt.ylabel("Number of Customer")
plt.show()
```

Mall Customer Dataset

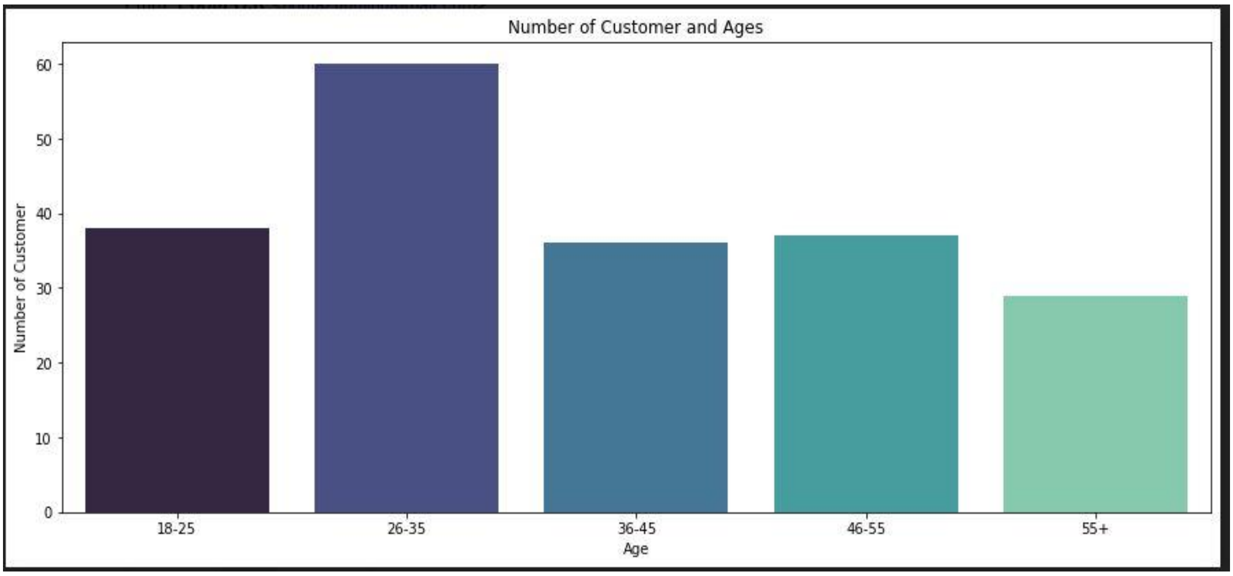


Fig 4.7:visualization of customer classified based on range of age

From this graph it is clear that count of customers of age group 26-35 is higher compared to other age group,hence we can conclude that more youth goes for shopping in a mall .

2.8. Age vs SpendingScore

```
d=sns.distplot(data['Age'], label='SpendingScore & Age')
d=sns.distplot(data['Spendingscore'])
plt.legend(labels=['Age','SpendingScore'])
d.set(xlabel=None)
plt.show()
```

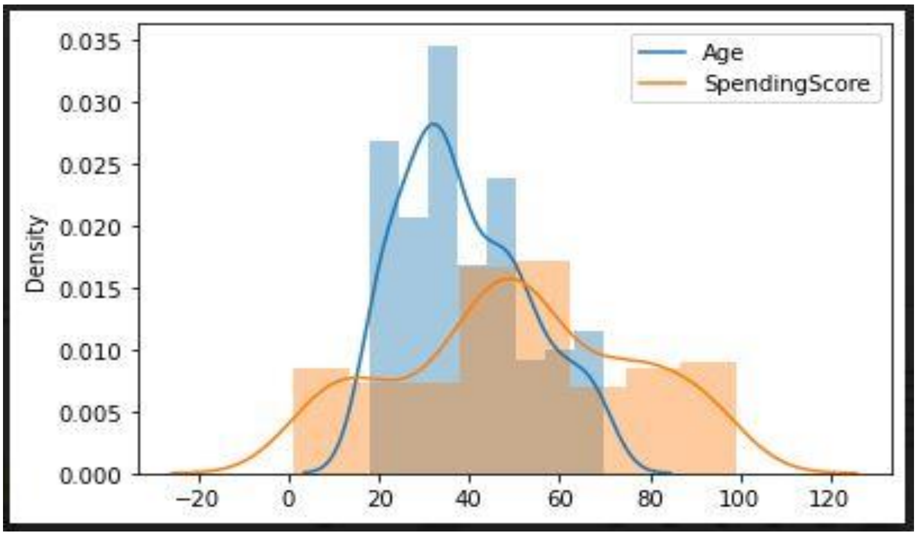


Fig 4.8:visualization of age and spending score

Here we can see that as the age increases the spending score decreases and it is higher at the mid age group.

2.9.Line plot graph for comparison between annual income,age,spending score

```
x = data['Annual Income (k$)']
y = data['Age']
z = data['Spending Score (1-100)']
sns.lineplot(x, y, color = 'blue')
sns.lineplot(x, z, color = 'pink')
plt.title('Annual Income vs Age and Spending Score', fontsize = 20)
plt.show()
plt.figure(figsize=(10,8), dpi= 80)
sns.pairplot(data, kind="scatter", plot_kws=dict(s=80, edgecolor="white",
linewidth=2.5))
    plt.show()
```

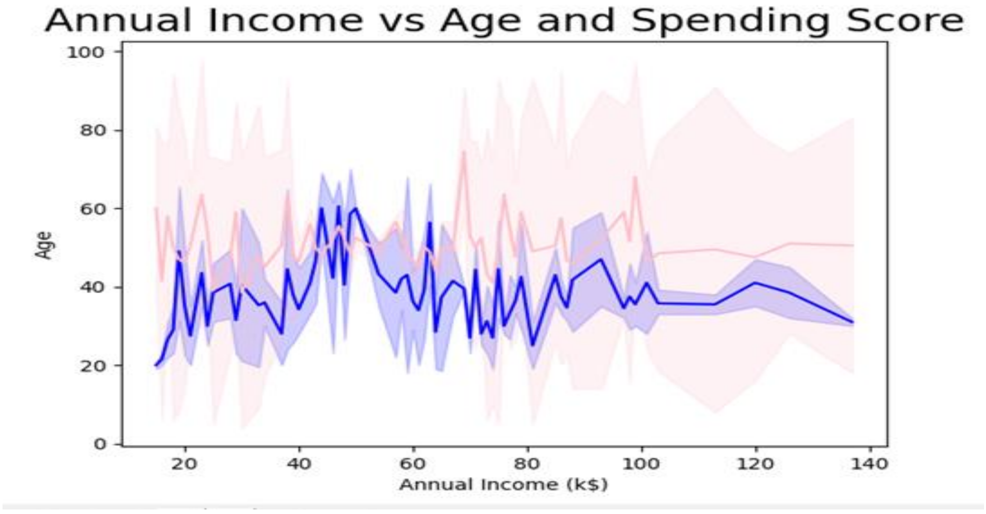


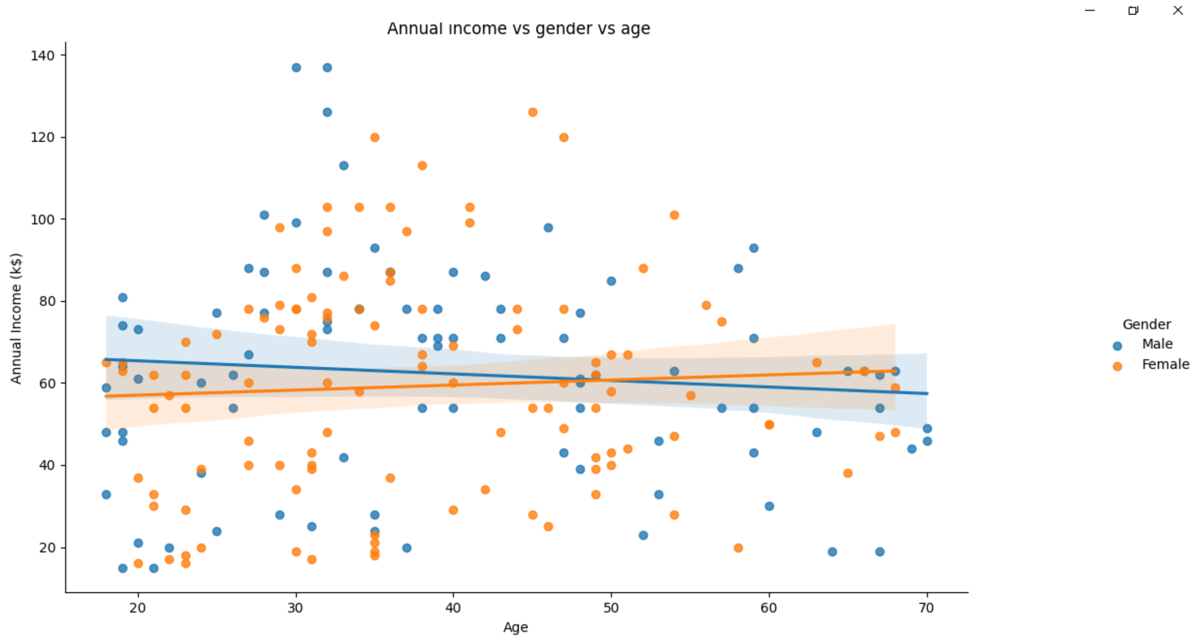
Fig 4.9:Line plot graph

The above Plot Between Annual Income and Age represented by a blue color line, and a plot between Annual Income and the Spending Score represented by a pink color. shows how Age and Spending Varies with Annual Income.

2.10. FacetGrid to plot annual income for genders and different age group

```
#annual income of  genders of different age groups
df=pd.read_csv("Mallcustomers.csv")
sns.lmplot(x = "Age", y = "Annual Income (k$)", data = df, hue = "Gender")
plt.title(" Annual income vs spending score vs age")
plt.show()
```

Mall Customer Dataset



In this graph we get to see something we could predict, young people tend to spend way more than older people. That can be due to many reasons: young people usually have more free time than old people, shopping malls tend to have shops that target young people such as videogames and tech stores, etc.

3. MODELLING

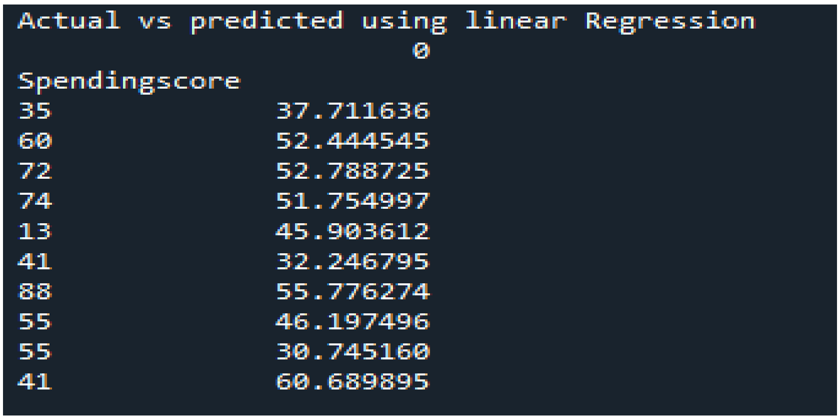
3.1 Linear Regression

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

CODE-

```
x=data.iloc[:,:-1]
y=data.iloc[:,4]

from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
A=make_column_transformer((OneHotEncoder(categories="auto"),[1]),remainder="passthrough")
x=A.fit_transform(x)
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=3)
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
```

```
y_pred=reg.predict(x_test)
df=pd.DataFrame(y_pred,y_test)
print('Actual vs predicted using linear Regression')
print(df[:10])
```



```
Actual vs predicted using linear Regression
                        0
Spendingscore
35             37.711636
60             52.444545
72             52.788725
74             51.754997
13             45.903612
41             32.246795
88             55.776274
55             46.197496
55             30.745160
41             60.689895
```

Fig 4.10:displaying predicted values

```
col=['Annual_Income']
col1=['SpendingScore']

x=data[col]
y=data[col1]
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=3)
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
y_pred=reg.predict(x_test)
#print(y_pred)
plt.style.use('dark_background')
plt.scatter(x_train,y_train,c='yellow')
plt.plot(x_train,reg.predict(x_train),c='r')
plt.title("Actual vs Predicted using linear Regression")
plt.xlabel("Annual_Income")
plt.ylabel("Spending Score")
plt.show()
```

Mall Customer Dataset

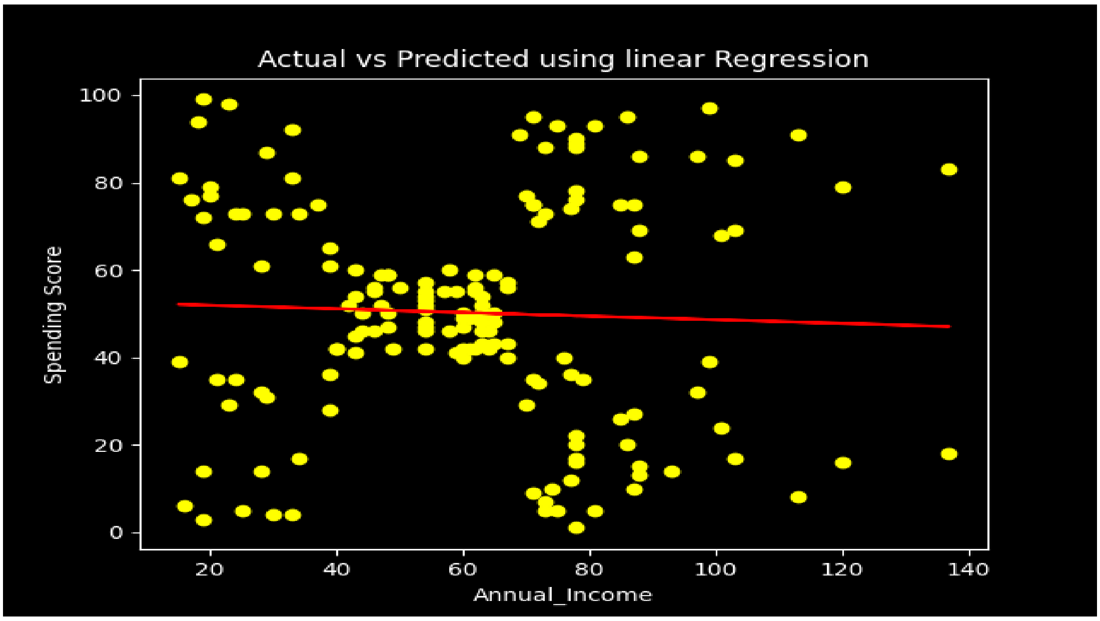
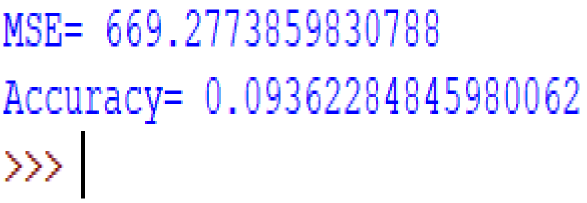


Fig 4.11:Linear regressing graph between annual income and spending score

```
from sklearn import metrics
print("MSE=",metrics.mean_squared_error(y_test,y_pred))
print("Accuracy=",reg.score(x_test,y_test))
```



```
MSE= 669.2773859830788
Accuracy= 0.09362284845980062
>>> 
```

3.2 K-Means Clustering

K-Means Clustering is an Unsupervised Learning algorithm which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

CODE-

```
#to find k using elbow method
x=data.iloc[:,[3,4]].values
from sklearn.cluster import KMeans
wcs=[]#variance list
for i in range(1,11):
        kmeans=KMeans(n_clusters=i,init='k-means++', max_iter=100,n_init=10,
                                                random_state=0)             #random points choosed
        kmeans.fit(x)
        wcs.append(kmeans.inertia_)#variance data is added

plt.plot(range(1,11),wcs,’-o’)
plt.xlabel('k value')
plt.ylabel('variance')
plt.title('elbow method ')
plt.show()
```

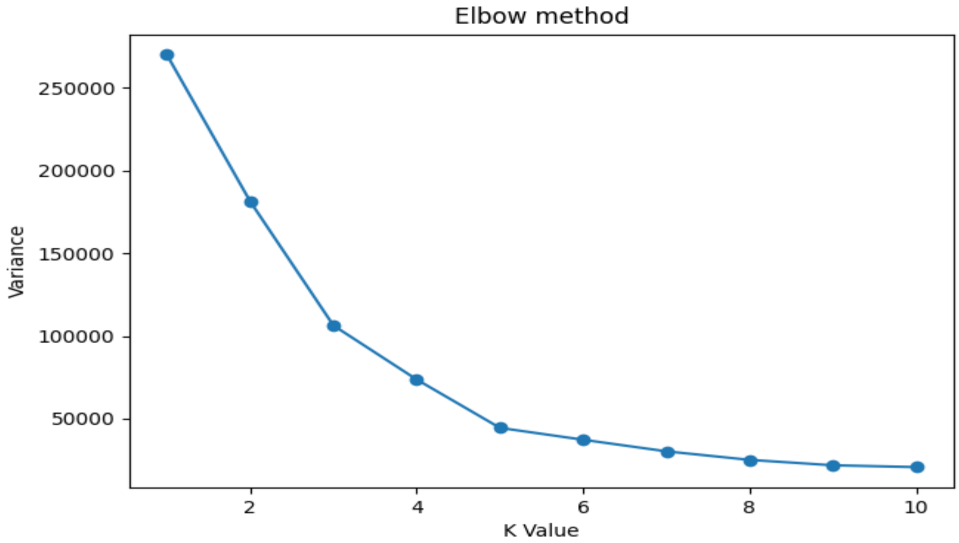


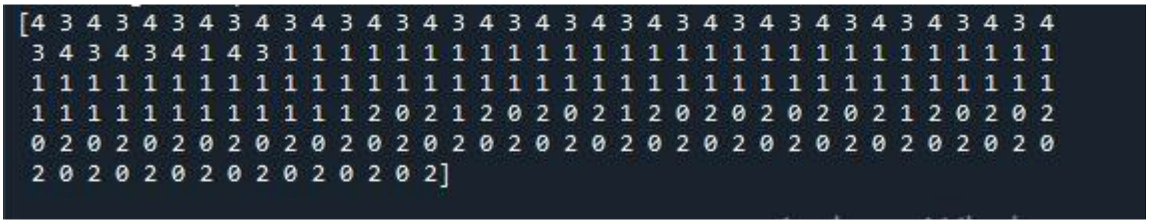
Fig 4.13:elbow method to find the k value

The elbow method is used to determine the optimal number of clusters in k-means clustering. The elbow method plots the value of the cost function produced by different values of k and one should choose a number of clusters so that adding another cluster doesn't give much better modelling of the data. In this problem, we are using the inertia as cost function in order to identify the sum of squared distances of samples to the nearest cluster centre.

Looking at the graph above, if we imagine the line in the graphic is an arm, the elbow can be found, approximately, where the number of clusters is equal to 5. Therefore we are selecting **5** as the number of clusters to divide our data in.

```
kmeans=KMeans(n_clusters=5,init='k-means++',max_iter=100,n_init=10,random_state=0)
y_kmeans=kmeans.fit_predict(x) #which cluster x belonged to is predicted
print(y_kmeans)
```

Mall Customer Dataset



```
[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 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 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]
```

Fig 4.14: displaying kmeans value

```
plt.scatter(x[y_kmeans==0,0],x[y_kmeans==0,1],s=10,c='green',label='cluster 1')
#x,y cluster is mentioned
plt.scatter(x[y_kmeans==1,0],x[y_kmeans==1,1],s=10,c='blue',label='cluster 2')
plt.scatter(x[y_kmeans==2,0],x[y_kmeans==2,1],s=10,c='red',label='cluster 3')
plt.scatter(x[y_kmeans==3,0],x[y_kmeans==3,1],s=10,c='yellow',label='cluster 4')
plt.scatter(x[y_kmeans==4,0],x[y_kmeans==4,1],s=10,c='black',label='cluster 5')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=30,c='cyan',label='
centroid')
 plt.legend()
plt.title('clusters of customers')
plt.xlabel('annual income')
plt.ylabel('spending score')
plt.show()
```

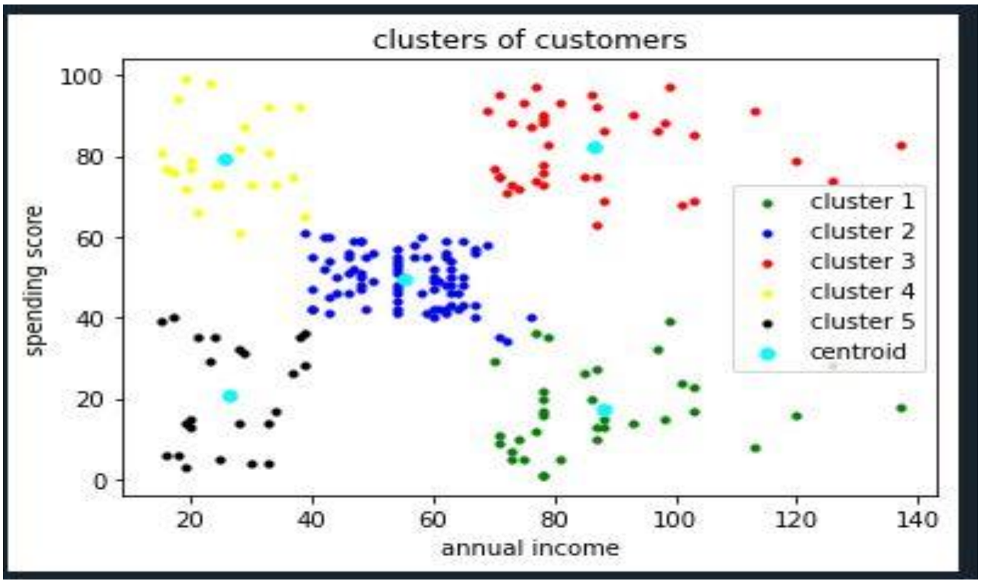


Fig 4.15: clusters of customers displayed comparing the annual income and spending score

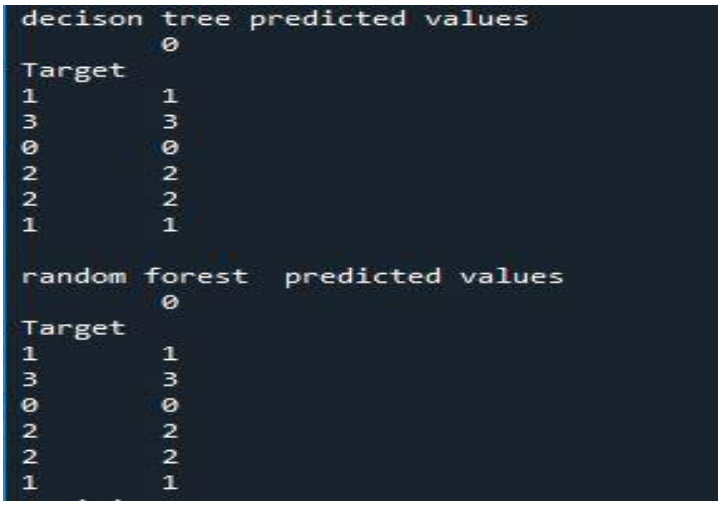
3.3 Random Forest and Decision Tree

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split

A random forest is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems. A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

CODE-

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

target=pd.DataFrame({'Target':kmeans.labels_})
new_data=pd.concat([data,target],axis=1,sort=False)
#print(new_data.head())

x_new=new_data.drop(['Target'],axis=1)
y_new=new_data['Target']

gen=pd.get_dummies(x_new['Gender'])
x_new=x_new.drop(['Gender'],axis=1)
x_new=pd.concat([x_new,gen],axis=1,sort=False)

x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.20,random_state=2)

dt=DecisionTreeClassifier()
rf=RandomForestClassifier()

model_dt=dt.fit(x_train,y_train)
model_rf=rf.fit(x_train,y_train)

y_pred=model_dt.predict(x_test)
df=pd.DataFrame(y_pred,y_test)
print('decison tree predicted values')
print(df[:6])
print()
y_pred1=model_rf.predict(x_test)
rf1=pd.DataFrame(y_pred1,y_test)
print('random forest  predicted values')
print(rf1[:6])
```

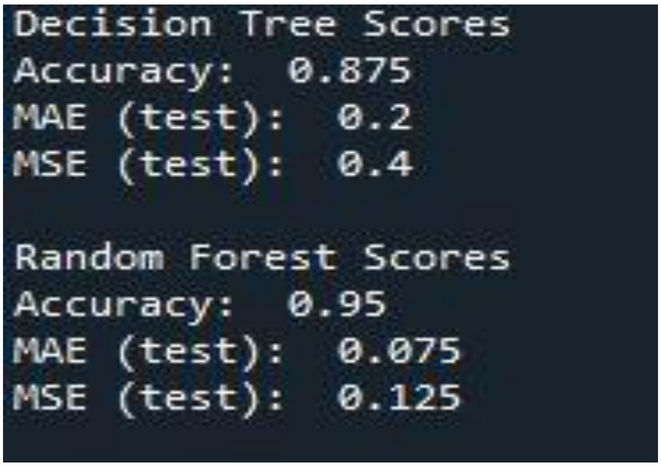


```
decison tree predicted values
        0
Target
1       1
3       3
0       0
2       2
2       2
1       1

random forest  predicted values
        0
Target
1       1
3       3
0       0
2       2
2       2
1       1
```

These are the actual and predicted clusters for both decision tree and random forest

3.4 MEASUREMENT AND COMPARISON BETWEEN DECISION TREE AND RANDOM FOREST CLASSIFIER

```
from sklearn import metrics
print("Decision Tree Scores")
print("Accuracy: ",metrics.accuracy_score(y_test,y_pred))
print("MAE (test): ",metrics.mean_absolute_error(y_test, y_pred))
print("MSE (test): ",metrics.mean_squared_error(y_test, y_pred))
print()
print("Random Forest Scores")
print("Accuracy: ",metrics.accuracy_score(y_test,y_pred1))
print("MAE (test): ",metrics.mean_absolute_error(y_test, y_pred1))
print("MSE (test): ",metrics.mean_squared_error(y_test, y_pred1))
```



```
Decision Tree Scores
Accuracy:  0.875
MAE (test):  0.2
MSE (test):  0.4

Random Forest Scores
Accuracy:  0.95
MAE (test):  0.075
MSE (test):  0.125
```

4.TESTING

4.1.TESTING LINEAR REGRESSION

```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
own_value=[[60],[50],[140],[144],[148],[150],[155],[160],[164],[168],[170],[175],[178],[180]
,[184],[188],[192],[195],[198],[250]]
y_pred=reg.predict(own_value)
plt.scatter(x_train,y_train,c='cyan')
plt.plot(own_value,reg.predict(own_value),c='black')
plt.title("Actual vs Predicted using linear Regression")
plt.xlabel("Annual_Income")
plt.ylabel("Spending Score")
plt.show()
```

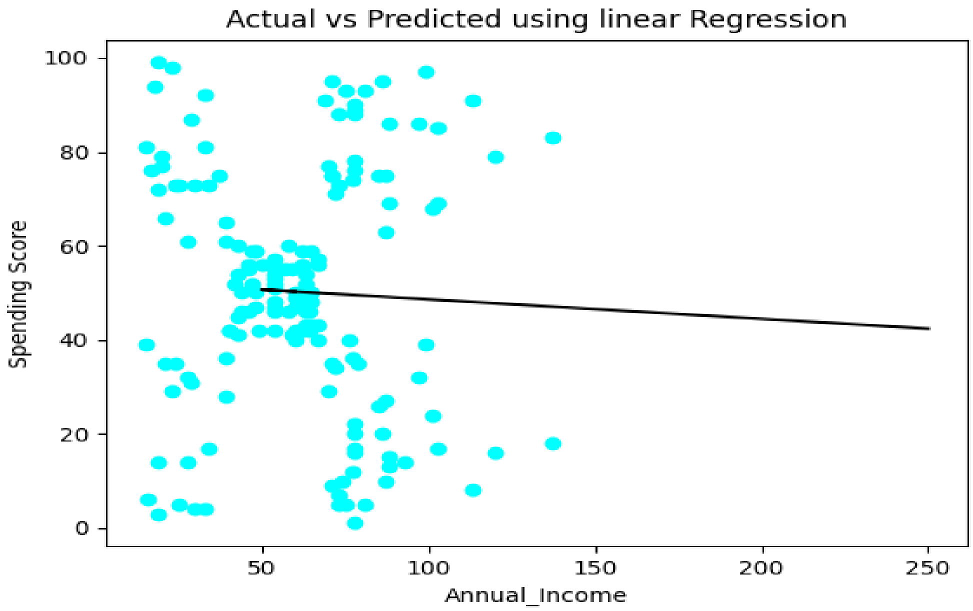


Fig 4.12:Linear regressing graph between annual income and spending score for user defined values

4.2 TESTING K-MEANS

```
pred_data = np.array([[30,10],[70,50],[20,80],[100,80],[100,20],[20,20],[60,60]])
predictions = kmeans.predict(pred_data)
print(predictions)
plt.figure(figsize=(7,7))
plt.scatter(X[:,0], X[:,1], s=20, c=y_kmeans, cmap='gist_rainbow')
```

Mall Customer Dataset

```
plt.scatter(pred_data[:,0], pred_data[:,1], s=250, c=predictions, cmap='gist_rainbow',
marker='+')
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s=75, c='black')
plt.title('Annual income vs spending score distribution')
plt.xlabel('Annual income (k$)')
plt.ylabel('Spending score (1-100)')
plt.show()
```

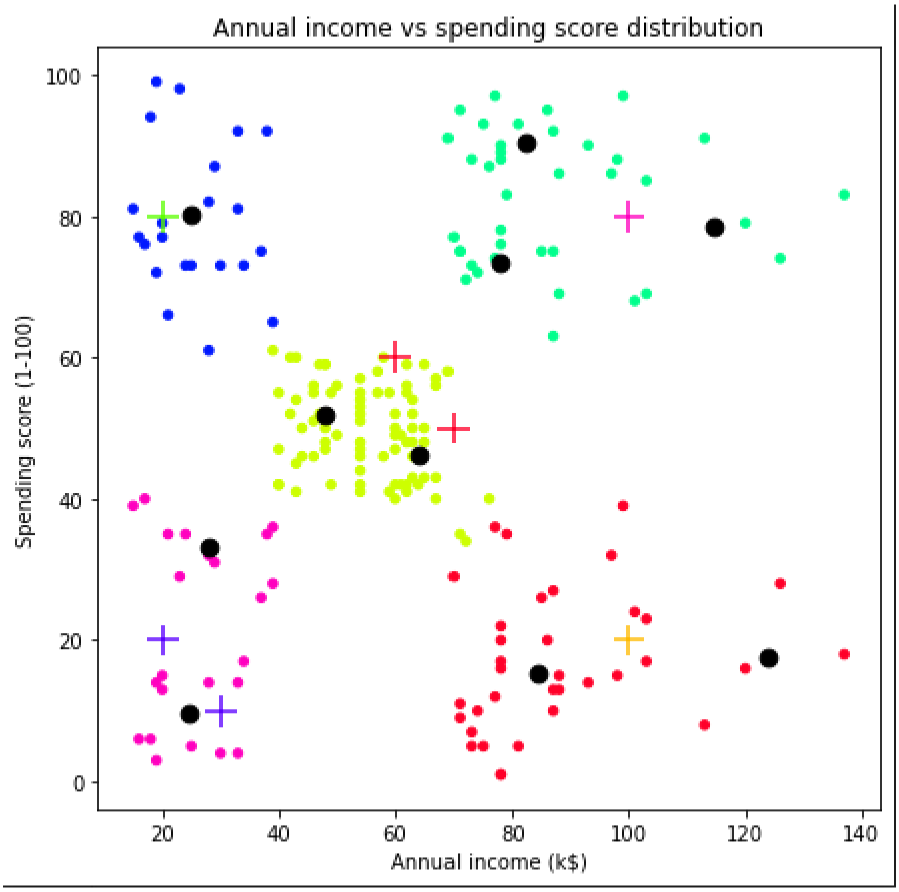


Fig 4.16: clusters of customers displayed comparing the annual income and spending score for user defined values

TASK PERFORMED

Have done few graphs in visualization such as piechart to find out percentage of male and female in the dataset choosed.Histogram graph for annual income column and we found that most of the annual income were in the range of 50 to 60(k$) and very few have more than 100 k$.Countplot graph to find out spending score of the customers and we found out most of the customers were scored 40-42.In the multiplot graph we did comparison with age,spending score and annual income we do a comparison with the help of scatter plot .

In Modelling I performed kmeans cluster algorithm where in after finding out the k value by using the elbow method 5 clusters were made.After that we train the kmeans model dataset and displayed the kmeans value based on which further clusters were to be formed.

This line of code :

```
    plt.scatter(x[y_kmeans==0,0],x[y_kmeans==0,1],s=10,c='green',label='cluster 1')
```

it acts as a filter.Here x[y_kmeans==0,0] selects elements of x where the corresponding y_kmeans value is 0 and second dimension is 0.

This line of code :

```
    plt.scatter(x[y_kmeans==1,0],x[y_kmeans==1,1],s=10,c='blue',label='cluster 2')
```

here x[y_kmeans==1,0] means model is in x plain and x[y_kmeans==0,1] means model is in y plain.where as 1 refers to the value of [i] or the cluster value and the same is followed for the rest of the code.

By using random forest method we predicted the clusters between annual income and spending score .We could also analyse the accuracy of descision tree and random forest method and we found out that random forest method shows more accuracy when compared to decision tree.

CONCLUSION

After developing a solution for this problem, we have come to the following conclusions:

- KMeans Clustering is a powerful technique in order to achieve a decent customer segmentation.

- Customer segmentation is a good way to understand the behaviour of different customers and plan a good marketing strategy accordingly.

- There is not much difference between the spending score of women and men, which leads us to think that our behaviour when it comes to shopping is pretty similar.

- Observing the clustering graphic, it can be clearly observed that the ones who spend more money in malls are young people. That is to say they are the main target when it comes to marketing, so doing deeper studies about what they are interested in may lead to higher profits.

- Although younglings seem to be the ones spending the most, we can not forget there are more people we have to consider, like people who belong to the pink cluster, they are what we would commonly name middle class and it seems to be the biggest cluster.

- Promoting discounts on some shops can be something of interest to those who don’t actually spend a lot and they may end up spending more!