```
import numpy as np
import pandas as pd
import seaborn as sns
```

load dataset

```
df=pd.read_csv("/content/Mall_Customers.csv")
```

```
df.head()
```

	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

chech missing values

```
df.isna().sum()
```

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

```
df.isna().sum().sum()
```

```
0
```

check catogrical values

```
df._get_numeric_data()
```

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
0	1	19	15	39
1	2	21	15	81
2	3	20	16	6
3	4	23	16	77
4	5	31	17	40
...	...	...	...	...
195	196	35	120	79
196	197	45	126	28
197	198	32	126	74
198	199	32	137	18
199	200	30	137	83

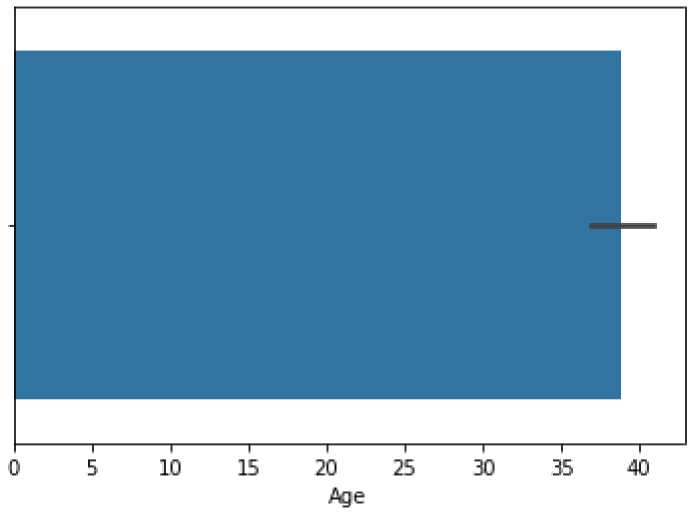
```
df.shape
```

```
(200, 5)
```

univariant analysis

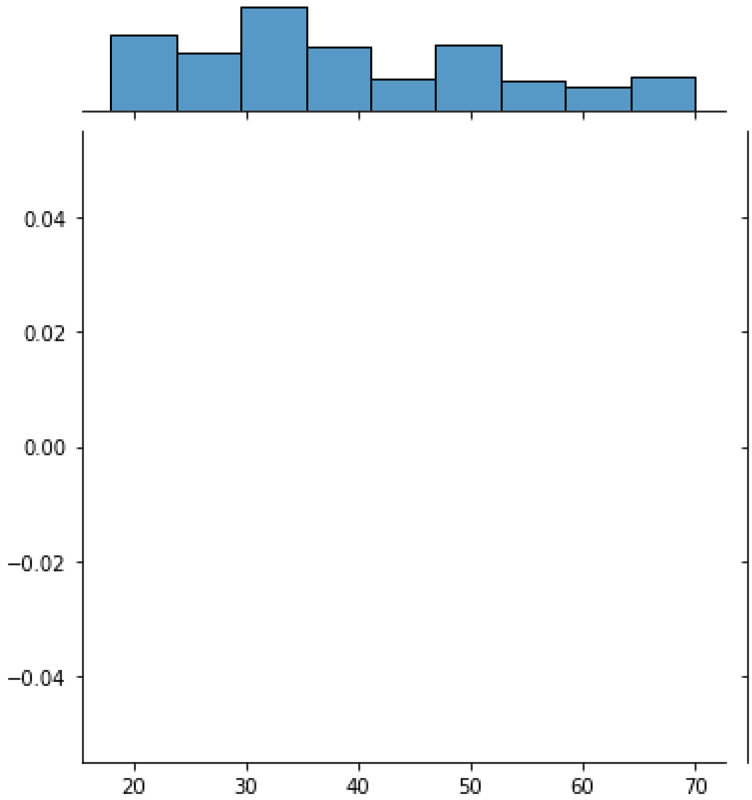
```
sns.barplot(df.Age)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f916e84a490>
```




```
sns.jointplot(df.Age)
```

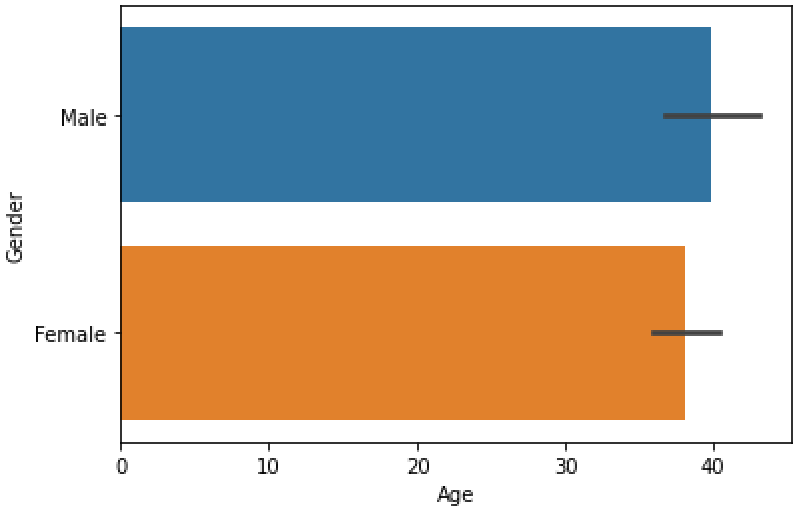
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<seaborn.axisgrid.JointGrid at 0x7f916dddf250>
```



bivariant analysis

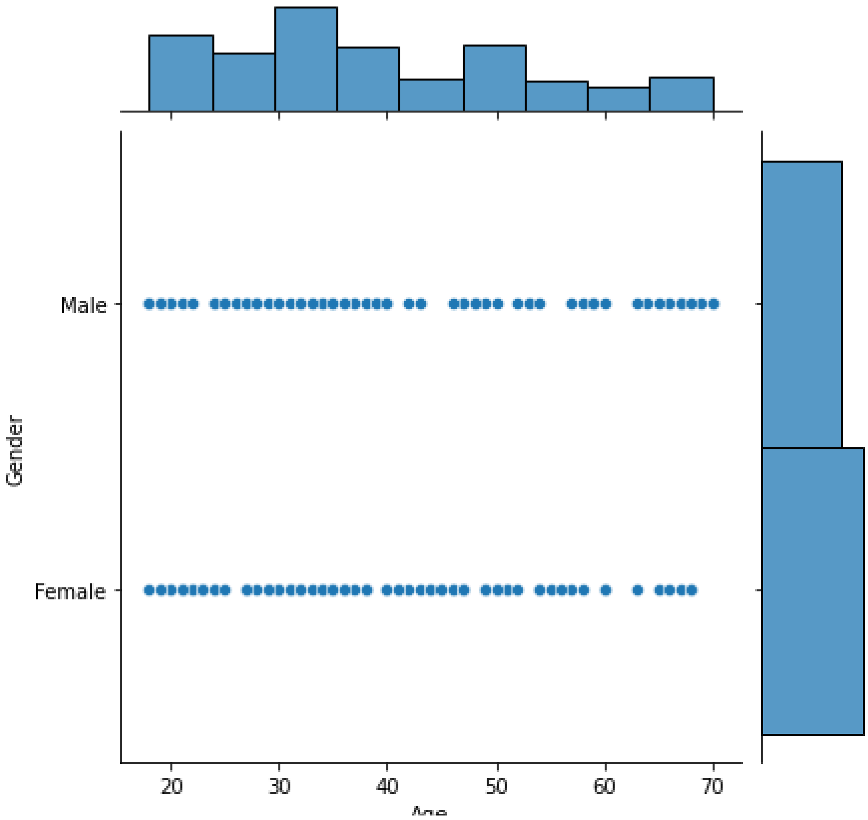
```
sns.barplot(df.Age,df.Gender)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f916b4737d0>
```




```
sns.jointplot(df.Age,df.Gender)
```

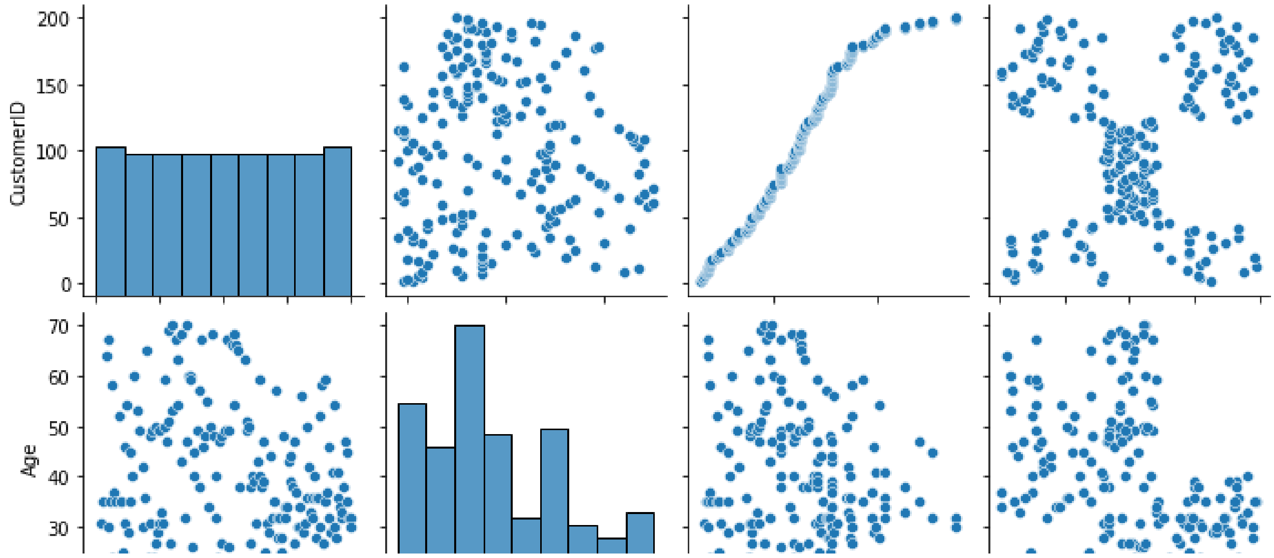
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<seaborn.axisgrid.JointGrid at 0x7f916b487810>
```



multi varient analysis

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f916b469750>
```



statistics values

```
df.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
```

CustomerID Age Annual Income (k$) Spending Score (1-100)

scale the data

```
from sklearn.preprocessing import MinMaxScaler
scalar=MinMaxScaler()
df_new1=df.iloc[:, :-1]
```

```
df_new1
```

	CustomerID	Gender	Age	Annual Income (k$)
0	1	Male	19	15
1	2	Male	21	15
2	3	Female	20	16
3	4	Female	23	16
4	5	Female	31	17
...	...	...	...	...
195	196	Female	35	120

split dependant and independant variable

```
x=df_new1
y=df['Spending Score (1-100)']
```

split test and train data

```
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)
```

build clustering algorithm model

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn=KNeighborsClassifier
```

predict the data

```
knn.fit(x_train,y_train)
```

```
pred=knn.predict(x_test)
```

evaluate our model

```
from sklearn.metrics import accuracy_score,confusion_matrix
```

```
accuracy_score(y_test,pred)
```

```
confusion_matrix(y_test,pred)
```