ASSIGNMENT 4

1. Download the dataset: Dataset

2. Load the dataset into the tool.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import warnings
```

```
data=pd.read_csv("Mall_Customers.csv",encoding='ISO-8859-1')
data.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
				17	40

Saved successfully!

```
data.describe()
```

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000

```
data.dtypes
```

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

3. Perform Below Visualizations.

· Univariate Analysis

· Bi- Variate Analysis

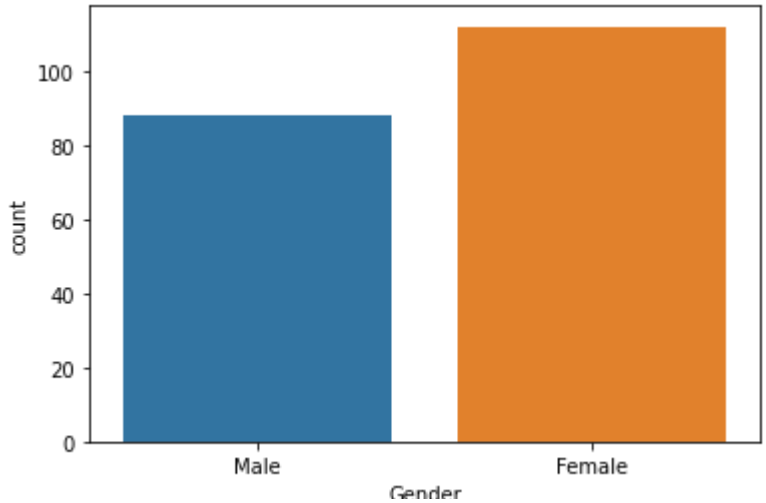
· Multi-Variate Analysis

```
#univariate analysis "Histogram"
sns.histplot(data["Gender"],color='darkorange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe16d00fa10>
```

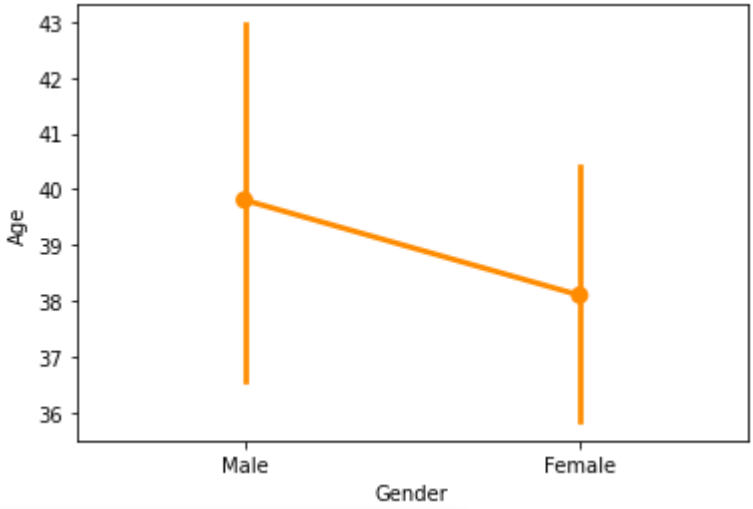

```
#univariate analysis "Countlot"
sns.countplot(data['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fe16cf59150>
```




```
#bivariate analysis"Pointplot"
sns.pointplot(x='Gender',y='Age',data=data,color='darkorange')
```

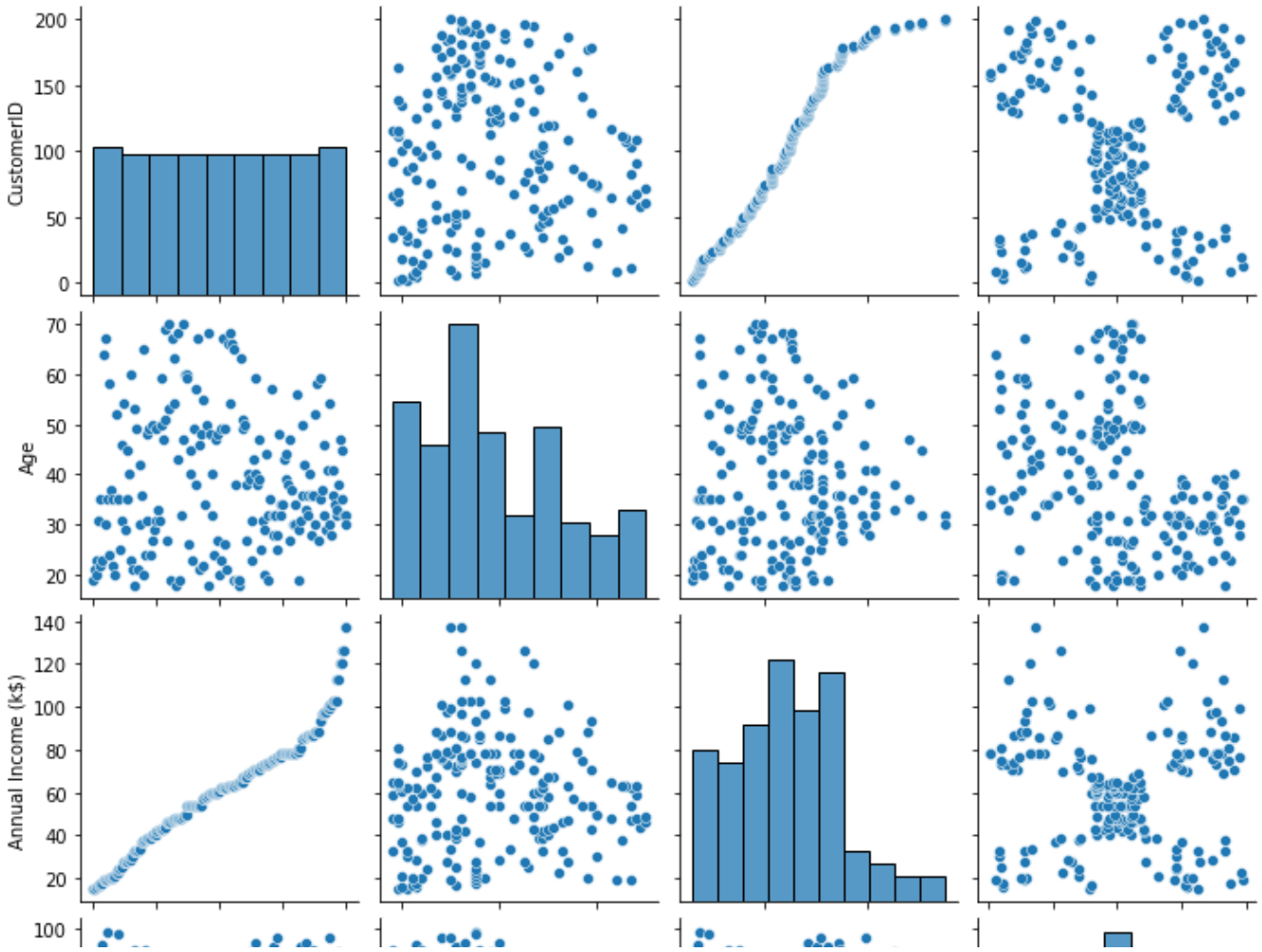
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe16cf14910>
```



Saved successfully!

```
#multivariate analysis "pairplot"
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x7fe16cf0f5d0>
```



4. Perform descriptive statistics on the dataset.

```
# Descriptive statistics of the data set accessed.
data.describe().T
```

Saved successfully!

	count	mean	std	min	25%	50%	75%	max
CustomerID	200.0	100.50	57.879185	1.0	50.75	100.5	150.25	200.0
Age	200.0	38.85	13.969007	18.0	28.75	36.0	49.00	70.0
Annual Income (k$)	200.0	60.56	26.264721	15.0	41.50	61.5	78.00	137.0
Spending Score (1-100)	200.0	50.20	25.823522	1.0	34.75	50.0	73.00	99.0

```
data.isnull().any().any()
```

```
False
```

```
data.isnull().any()
```

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

```
df2=data.dropna(how='all')
```

5. Check for Missing values and deal with them.

```
df2.isnull().sum().sum()
```

```
0
```

This dataset does not contain any missing value.

```
missing_values=data.isnull().sum()
missing_values[missing_values>0]/len(data)*100
```

```
Series([], dtype: float64)
```

6. Find the outliers and replace them outliers

```
sns.boxplot(data['Age'],data=data)
```

Saved successfully!

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```

```
for x in ['Age']:
    q75,q25 = np.percentile(data.loc[:,x],[75,25])
    intr_qr = q75-q25

    max = q75+(1.5*intr_qr)
    min = q25-(1.5*intr_qr)

    data.loc[data[x] < min,x] = np.nan
    data.loc[data[x] > max,x] = np.nan
```

```
data.isnull().sum()
```

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

7. Check for Categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
data['Gender']=encoder.fit_transform(data['Gender'])
```

```
data.head()
```

Saved successfully!

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	1	19.0	15	39
1	2	1	21.0	15	81
2	3	0	20.0	16	6
3	4	0	23.0	16	77
4	5	0	31.0	17	40

8. Scaling the data

```
from sklearn.preprocessing import StandardScaler
df=StandardScaler()
data1=df.fit_transform(data)
```

```
data1
```

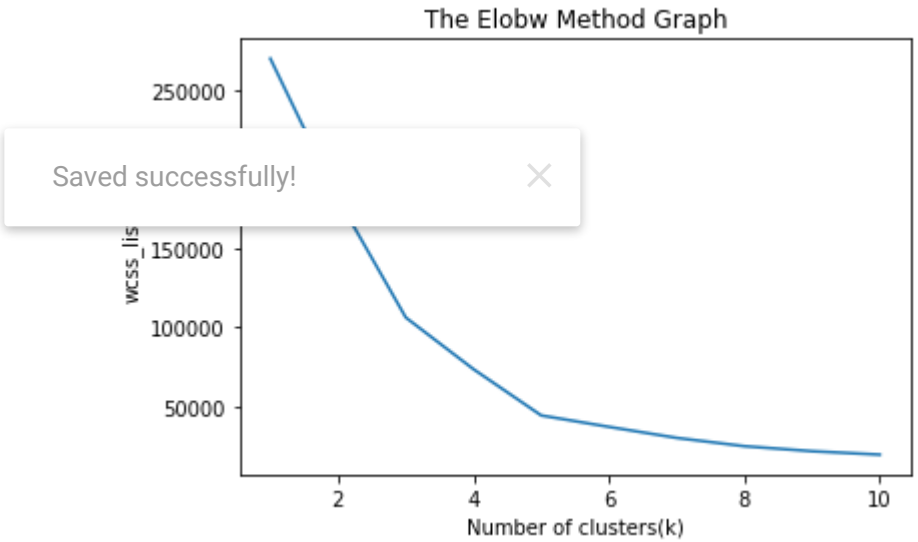
```
array([[ 0.73613079, -0.88640526, -0.77866858,  0.58933599, -0.39597992],
       [ 0.75345152, -0.88640526, -0.49160182,  0.58933599,  1.42863343],
       [ 0.77077224,  1.12815215, -0.99396865,  0.62750542, -1.48298362],
       [ 0.78809297,  1.12815215, -0.77866858,  0.62750542,  1.81684904],
       [ 0.80541369,  1.12815215,  0.65666521,  0.62750542, -0.55126616],
       [ 0.82273442, -0.88640526, -0.49160182,  0.62750542,  0.92395314],
       [ 0.84005514, -0.88640526, -0.34806844,  0.66567484, -1.09476801],
       [ 0.85737587,  1.12815215, -0.34806844,  0.66567484,  1.54509812],
       [ 0.87469659,  1.12815215,  0.29783176,  0.66567484, -1.28887582],
       [ 0.89201732,  1.12815215,  0.010765  ,  0.66567484,  1.46745499],
       [ 0.90933804, -0.88640526,  0.36959845,  0.66567484, -1.17241113],
       [ 0.92665877, -0.88640526, -0.06100169,  0.66567484,  1.00159627],
       [ 0.94397949, -0.88640526,  0.58489852,  0.66567484, -1.32769738],
       [ 0.96130021, -0.88640526, -0.85043527,  0.66567484,  1.50627656],
       [ 0.97862094,  1.12815215, -0.13276838,  0.66567484, -1.91002079],
       [ 0.99594166, -0.88640526, -0.6351352 ,  0.66567484,  1.07923939],
       [ 1.01326239,  1.12815215, -0.34806844,  0.66567484, -1.91002079],
       [ 1.03058311, -0.88640526, -0.6351352 ,  0.66567484,  0.88513158],
       [ 1.04790384, -0.88640526,  1.23079873,  0.70384427, -0.59008772],
       [ 1.06522456, -0.88640526, -0.70690189,  0.70384427,  1.27334719],
       [ 1.08254529,  1.12815215, -1.42456879,  0.78018313, -1.75473454],
       [ 1.09986601, -0.88640526, -0.56336851,  0.78018313,  1.6615628 ],
       [ 1.11718674,  1.12815215,  0.80019859,  0.93286085, -0.93948177],
       [ 1.13450746, -0.88640526, -0.20453507,  0.93286085,  0.96277471],
       [ 1.15182818,  1.12815215,  0.22606507,  0.97103028, -1.17241113],
       [ 1.16914891, -0.88640526, -0.41983513,  0.97103028,  1.73920592],
       [ 1.18646963, -0.88640526, -0.20453507,  1.00919971, -0.90066021],
       [ 1.20379036,  1.12815215, -0.49160182,  1.00919971,  0.49691598],
       [ 1.22111108,  1.12815215,  0.08253169,  1.00919971, -1.44416206],
       [ 1.23843181,  1.12815215, -0.77866858,  1.00919971,  0.96277471],
       [ 1.25575253,  1.12815215, -0.20453507,  1.00919971, -1.56062674],
       [ 1.27307326,  1.12815215, -0.20453507,  1.00919971,  1.62274124],
       [ 1.29039398, -0.88640526,  0.94373197,  1.04736914, -1.44416206],
       [ 1.30771471, -0.88640526, -0.6351352 ,  1.04736914,  1.38981187],
       [ 1.32503543,  1.12815215,  1.37433211,  1.04736914, -1.36651894],
       [ 1.34235616,  1.12815215, -0.85043527,  1.04736914,  0.72984534],
       [ 1.35967688,  1.12815215,  1.4460988 ,  1.23821628, -1.4053405 ],
       [ 1.3769976 ,  1.12815215, -0.27630176,  1.23821628,  1.54509812],
       [ 1.39431833, -0.88640526, -0.13276838,  1.390894  , -0.7065524 ],
       [ 1.41163905, -0.88640526, -0.49160182,  1.390894  ,  1.38981187],
       [ 1.42895978,  1.12815215,  0.51313183,  1.42906343, -1.36651894],
       [ 1.4462805 , -0.88640526, -0.70690189,  1.42906343,  1.46745499],
       [ 1.46360123, -0.88640526,  0.15429838,  1.46723286, -0.43480148],
       [ 1.48092195,  1.12815215, -0.6351352 ,  1.46723286,  1.81684904],
       [ 1.49824268, -0.88640526,  1.08726535,  1.54357172, -1.01712489],
       [ 1.5155634 ,  1.12815215, -0.77866858,  1.54357172,  0.69102378],
       [ 1.53288413, -0.88640526,  0.15429838,  1.61991057, -1.28887582],
       [ 1.55020485, -0.88640526, -0.20453507,  1.61991057,  1.35099031],
       [ 1.56752558, -0.88640526, -0.34806844,  1.61991057, -1.05594645],
```

Saved successfully!

```
[ 1.5848463 , -0.88640526, -0.49160182,  1.61991057,  0.72984534],
[ 1.60216702,  1.12815215, -0.41983513,  2.00160487, -1.63826986],
[ 1.61948775, -0.88640526, -0.06100169,  2.00160487,  1.58391968],
[ 1.63680847, -0.88640526,  0.58489852,  2.26879087, -1.32769738],
[ 1.6541292 , -0.88640526, -0.27630176,  2.26879087,  1.11806095],
[ 1.67144992, -0.88640526,  0.44136514,  2.49780745, -0.86183865],
[ 1.68877065,  1.12815215, -0.49160182,  2.49780745,  0.92395314],
[ 1.70609137,  1.12815215, -0.49160182,  2.91767117, -1.25005425],
[ 1.7234121 ,  1.12815215, -0.6351352 ,  2.91767117,  1.27334719]])
```

9. Perform any of the clustering algorithms 10. Add the cluster data with the primary dataset

```
x = data.iloc[:, [3, 4]].values
from sklearn.cluster import·KMeans··
wcss_list=·[]···
for·i·in·range(1,·11):··
····kmeans·=·KMeans(n_clusters=i,·init='k-means++',·random_state=·42)··
····kmeans.fit(x)··
····wcss_list.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss_list)
plt.title('The Elobw Method Graph')
plt.xlabel('Number of clusters(k)')
plt.ylabel('wcss_list')
plt.show()
```



Saved successfully!

```
kmeans = KMeans(n_clusters=5, init='k-means++', random_state= 42)
y_predict= kmeans.fit_predict(x)
plt.scatter(x[y_predict == 0, 0], x[y_predict == 0, 1], s = 100, c = 'blue',
```

```
plt.scatter(x[y_predict == 1, 0], x[y_predict == 1, 1], s = 100, c = 'green',
plt.scatter(x[y_predict== 2, 0], x[y_predict == 2, 1], s = 100, c = 'red', la
plt.scatter(x[y_predict == 3, 0], x[y_predict == 3, 1], s = 100, c = 'cyan',
plt.scatter(x[y_predict == 4, 0], x[y_predict == 4, 1], s = 100, c = 'magenta
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s =
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

11. Split the data into dependent and independent variables

Saved successfully!

```
y=data['Age']
y.head()
```

```
0    19.0
1    21.0
2    20.0
3    23.0
4    31.0
Name: Age, dtype: float64
```

```
#independent
x=data.drop(columns=['Age'],axis=1)
x.head()
```

	CustomerID	Gender	Annual Income (k$)	Spending Score (1-100)
0	1	1	15	39
1	2	1	15	81
2	3	0	16	6
3	4	0	16	77
4	5	0	17	40

```
data=pd.get_dummies(data,columns=['Age'])
data.head()
```

	CustomerID	Gender	Annual Income (k$)	Spending Score (1-100)	Age_18.0	Age_19.0	Age_20.0	Age_21.0	Age_22.
0	1	1	15	39	0	1	0	0	
1	2	1	15	81	0	0	0	1	
2	3	0	16	6	0	0	1	0	
3	4	0	16	77	0	0	0	0	
4	5	0	17	40	0	0	0	0	

5 rows × 55 columns

12. Split the data into training and testing

Saved successfully!

```
import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_sta
x_train.shape
```

```
(160, 4)
```

```
x_test.shape
```

```
(40, 4)
```

```
y_train.shape
```

```
(160,)
```

```
y_test.shape
```

```
(40,)
```

13. Build the Model

14. Train the Model

15. Test the Model

16. Measure the performance using Evaluation Metrics.

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
x_train.shape
```

```
(160, 4)
```

```
x_test.shape
```

```
(40, 4)
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```



```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: Convergenc
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
train_pred = model.predict(x_train)
train_pred
```

```
array([57., 35., 60., 32., 34., 35., 53., 32., 32., 32., 32., 32., 32.,
       32., 32., 53., 53., 36., 60., 35., 32., 32., 32., 21., 32., 32.,
       35., 39., 35., 32., 32., 32., 32., 32., 34., 32., 32., 53., 35.,
       29., 32., 35., 34., 32., 32., 35., 53., 20., 41., 32., 35., 53.,
       47., 35., 47., 34., 32., 32., 32., 32., 57., 32., 35., 35., 32.,
       32., 32., 35., 32., 35., 32., 20., 20., 32., 21., 39., 21., 32.,
       32., 20., 21., 32., 32., 35., 32., 32., 21., 32., 34., 45., 53.,
       35., 32., 32., 39., 32., 57., 32., 32., 21., 32., 32., 35., 35.,
       35., 21., 32., 52., 60., 32., 32., 32., 21., 53., 32., 35., 32.,
       59., 53., 60., 32., 32., 35., 34., 29., 34., 53., 32., 32., 32.,
       21., 32., 32., 32., 32., 34., 32., 32., 32., 21., 32., 32., 32.,
       34., 32., 35., 35., 57., 32., 35., 32., 53., 21., 21., 32., 32.,
       37., 32., 35., 34.])
```

```
test_pred= model.predict(x_test)
test_pred
```

```
array([53., 34., 32., 32., 28., 34., 21., 32., 53., 34., 35., 32., 32.,
       34., 32., 21., 21., 34., 35., 32., 32., 29., 35., 28., 35., 29.,
       32., 32., 32., 53., 35., 20., 34., 32., 53., 32., 32., 35., 32.,
       32.])
```

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

```
accuracy_score(y_test,test_pred)
```

```
0.025
```

```
accuracy_score(y_train,train_pred)
```

```
0.1
```

Saved successfully!

```
ored)
```

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

```
pd.crosstab(y_test,test_pred)
```

col_0	20.0	21.0	28.0	29.0	32.0	34.0	35.0	53.0
Age								
18.0	0	1	0	0	0	0	0	0
19.0	0	0	0	0	1	0	1	0
22.0	0	1	0	0	0	0	0	0
23.0	0	1	0	0	0	0	0	0
24.0	0	0	0	0	0	0	1	0
27.0	0	0	1	0	0	0	0	0
28.0	0	0	0	1	0	0	0	0
29.0	0	0	0	0	1	0	0	0
30.0	0	0	1	0	1	0	1	0
31.0	0	0	0	0	1	0	1	0
35.0	0	0	0	0	1	0	1	0
36.0	0	0	0	0	1	0	0	0
37.0	0	0	0	0	1	0	0	0
38.0	0	0	0	0	1	0	0	0
39.0	0	0	0	1	0	0	0	0
40.0	0	0	0	0	1	1	0	0
43.0	0	0	0	0	0	1	0	0
44.0	0	0	0	0	0	1	0	0
46.0	0	0	0	0	1	1	0	1
					0	2	0	0
					2	0	0	0
49.0	0	0	0	0	0	0	0	1
52.0	0	0	0	0	0	0	0	1
54.0	0	0	0	0	1	0	0	0
57.0	0	0	0	0	1	0	0	0
58.0	0	0	0	0	0	0	0	1
59.0	0	0	0	0	1	0	0	0

Saved successfully!

```
print(classification_report(y_test,test_pred))
```

```
              precision    recall  f1-score   support

        18.0       0.00      0.00      0.00         1
        19.0       0.00      0.00      0.00         2
        20.0       0.00      0.00      0.00         0
        21.0       0.00      0.00      0.00         0
        22.0       0.00      0.00      0.00         1
        23.0       0.00      0.00      0.00         1
        24.0       0.00      0.00      0.00         1
        27.0       0.00      0.00      0.00         1
        28.0       0.00      0.00      0.00         1
        29.0       0.00      0.00      0.00         1
        30.0       0.00      0.00      0.00         3
        31.0       0.00      0.00      0.00         2
        32.0       0.00      0.00      0.00         0
        34.0       0.00      0.00      0.00         0
        35.0       0.17      0.50      0.25         2
        36.0       0.00      0.00      0.00         1
        37.0       0.00      0.00      0.00         1
        38.0       0.00      0.00      0.00         1
        39.0       0.00      0.00      0.00         1
        40.0       0.00      0.00      0.00         2
        43.0       0.00      0.00      0.00         1
        44.0       0.00      0.00      0.00         1
        46.0       0.00      0.00      0.00         3
        47.0       0.00      0.00      0.00         3
        48.0       0.00      0.00      0.00         2
        49.0       0.00      0.00      0.00         1
        52.0       0.00      0.00      0.00         1
        53.0       0.00      0.00      0.00         0
        54.0       0.00      0.00      0.00         1
        57.0       0.00      0.00      0.00         1
        58.0       0.00      0.00      0.00         1
        59.0       0.00      0.00      0.00         1
        66.0       0.00      0.00      0.00         1
        70.0       0.00      0.00      0.00         1

    accuracy                           0.03        40
                                01      0.01        40
                                03      0.01        40
```

Saved successfully!

```
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefine
  _warn_prf(average, modifier, msg_start, len(result))
```

Colab paid products - Cancel contracts here

Saved successfully!