Group members:-

208)Samruddhi Krishna Bhujbal

209)Vidya Ganesh Bingi

216)Vaibhav Rajendra Dingalwar

Problem Satement :Mumbai University Result

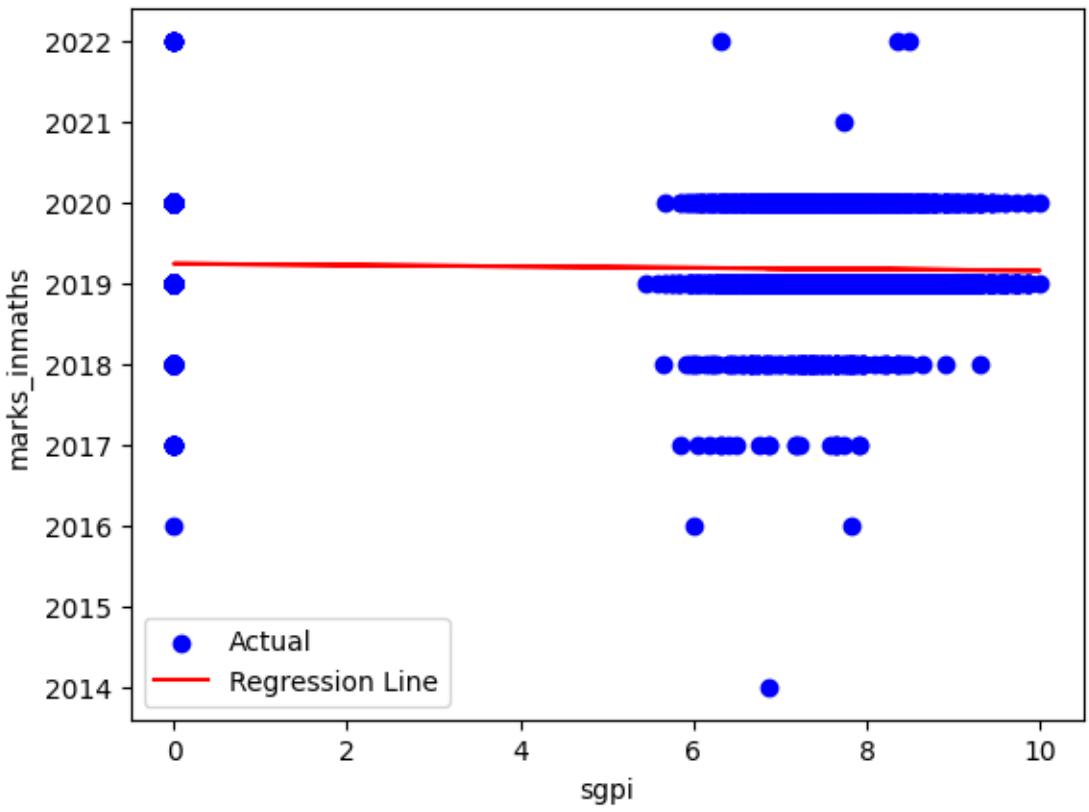
Linear regression

```
#Linear regration
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.linear_model import LinearRegression
df1=pd.read_csv("/content/sample_data/ass4_dataset.csv")
data = df1.dropna()
print(data)
# Extract the columns for linear regression
X = data['sgpi'].values.reshape(-1, 1)  # Input feature
y = data['year_of_admission'].values  # Target variable
# Create and fit the linear regression model
model = LinearRegression()
model.fit(X, y)

# Predict the target variable
y_pred = model.predict(X)

# Plot the data points and the regression line
plt.scatter(X, y, color='blue', label='Actual')
plt.plot(X, y_pred, color='red', label='Regression Line')
plt.xlabel('sgpi')
plt.ylabel('marks_inmaths')
plt.legend()
```

```
plt.show()
```

OUTPUT

```
5207  5018235  2.019020e+15   Thane                193  8.77  Successful
5208  5018236  2.019020e+15   Thane                170  7.73  Successful
5209  5018237  2.019020e+15   Thane                175  7.95  Successful
5210  5018238  2.018020e+15   Thane                151  6.86  Successful

      year_of_admission  clg_id gender  cgpa  marks_in_phy  marks_in_chem
\
0                  2020      10      M  9.14            49             45
1                  2020      10      M  8.09            39             48
2                  2018      10      M  7.95            39             45
3                  2019      10      M  7.55            47             46
4                  2019      10      M  8.05            49             49
...                 ...     ...    ...   ...           ...            ...
5206               2019     996      M  8.50            40             45
5207               2019     996      M  8.77            41             49
5208               2019     996      F  7.73            48             49
5209               2019     996      M  7.95            37             48
5210               2018     996      M  6.86            47             50

      marks_inmaths
0                48
1                39
2                45
3                48
4                44
...             ...
5206             44
5207             50
5208             49
5209             49
5210             35
```

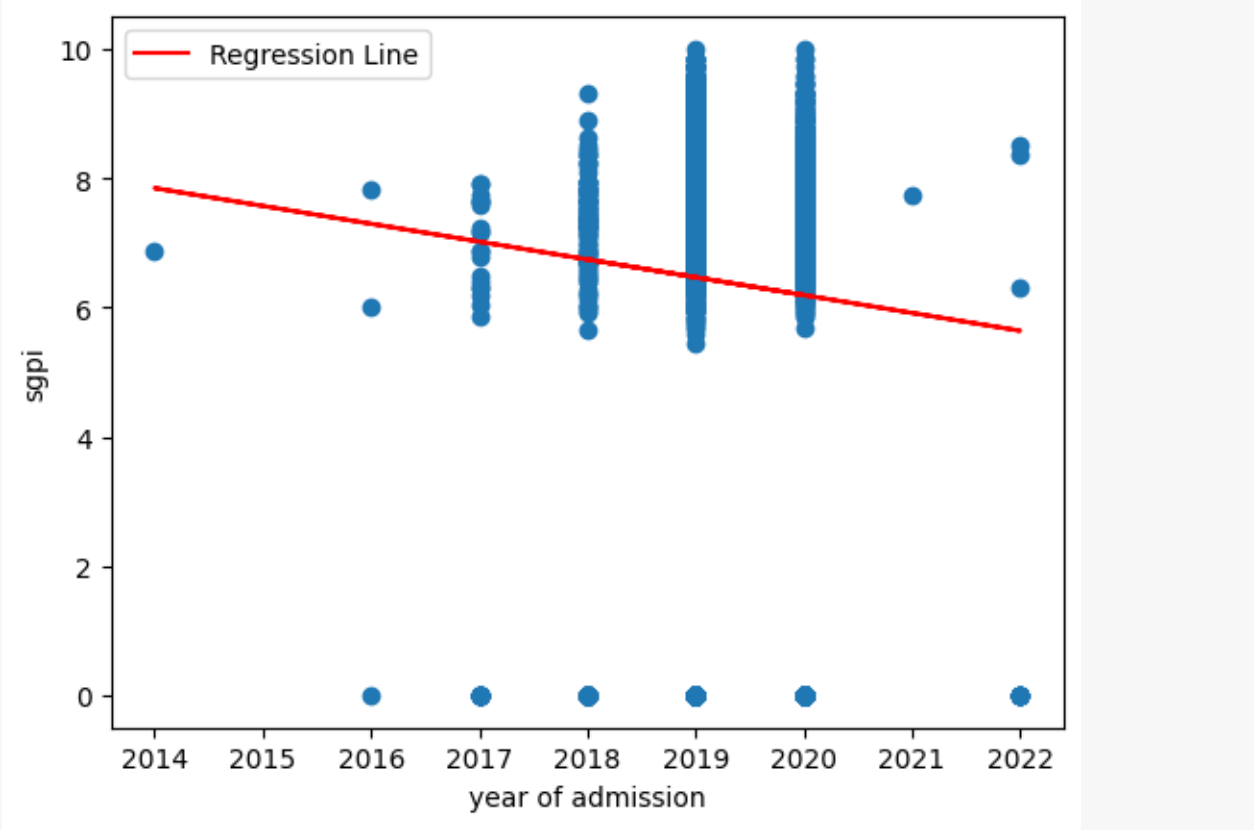



```
#Linear regration
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.linear_model import LinearRegression
df1=pd.read_csv("/content/sample_data/ass4_dataset.csv")
data = df1.dropna()
print(data)
# Extract the columns for linear regression
X = data['year_of_admission'].values.reshape(-1, 1)  # Input feature
y = data['sgpi'].values  # Target variable
# Create and fit the linear regression model
model = LinearRegression()
model.fit(X, y)

# Predict the target variable
y_pred = model.predict(X)
regr.fit(X_train,y_train)
print(regr.score(X_test,y_test))
# Plot the data points and the regression line
plt.scatter(X, y)
```

```
plt.plot(X, y_pred, color='red', label='Regression Line')
plt.xlabel('year of admission')
plt.ylabel('sgpi')
plt.legend()
plt.show()
```

OUTPUT:

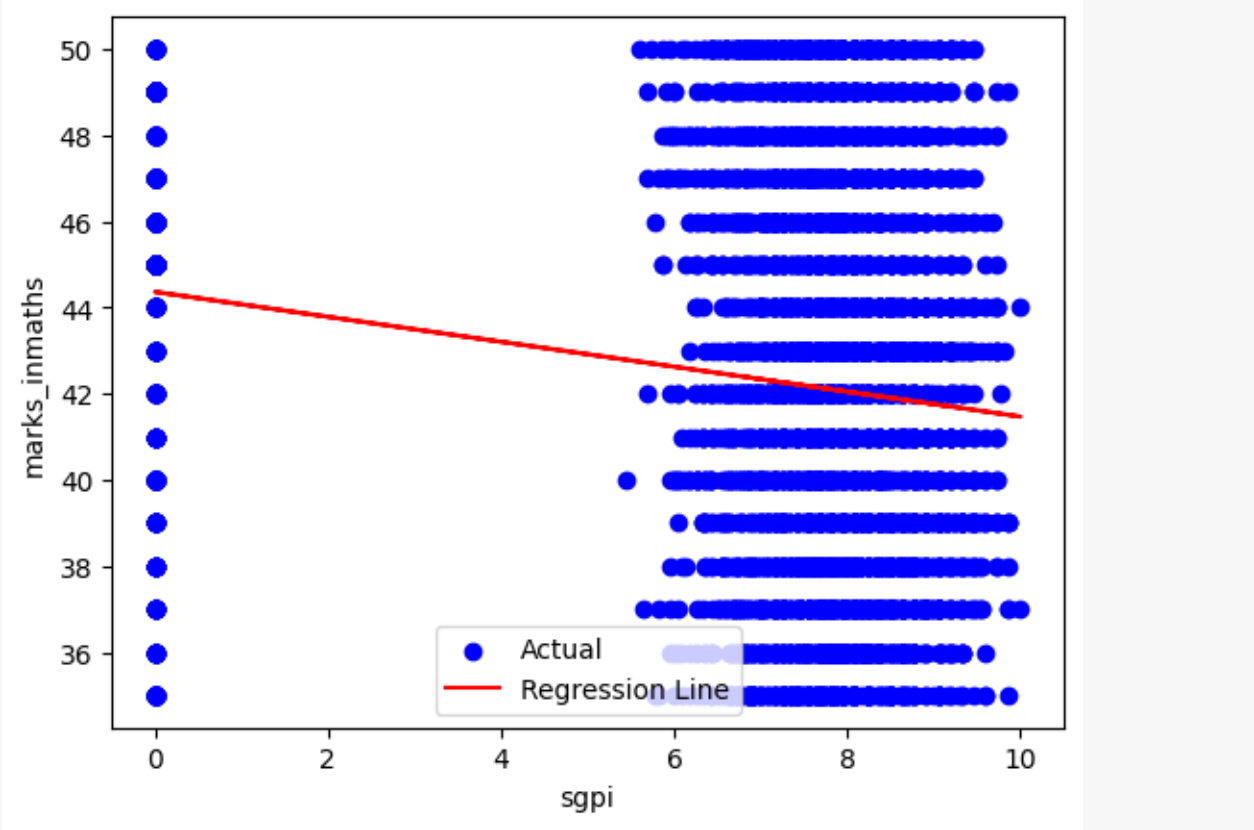
```
5208  5018236  2.019020e+15   Thane                170  7.73  Successful
5209  5018237  2.019020e+15   Thane                175  7.95  Successful
5210  5018238  2.018020e+15   Thane                151  6.86  Successful

      year_of_admission  clg_id gender  cgpa  marks_in_phy  marks_in_chem
\
0                  2020      10      M  9.14            49             45
1                  2020      10      M  8.09            39             48
2                  2018      10      M  7.95            39             45
3                  2019      10      M  7.55            47             46
4                  2019      10      M  8.05            49             49
...                 ...     ...    ...   ...           ...            ...
5206               2019     996      M  8.50            40             45
5207               2019     996      M  8.77            41             49
5208               2019     996      F  7.73            48             49
5209               2019     996      M  7.95            37             48
5210               2018     996      M  6.86            47             50

      marks_inmaths
0                48
1                39
2                45
3                48
4                44
...             ...
5206             44
5207             50
5208             49
5209             49
5210             35

[5190 rows x 13 columns]
0.00039381084648448805
```




```
data = df1.dropna()
print(data)
# Extract the columns for linear regression
X = data['sgpi'].values.reshape(-1, 1)  # Input feature
y = data['marks_inmaths'].values  # Target variable
X.sort()
y.sort()
# Create and fit the linear regression model
model = LinearRegression()
model.fit(X, y)

# Predict the target variable
y_pred = model.predict(X)

# Plot the data points and the regression line
plt.scatter(X, y, color='blue', label='Actual')
plt.plot(X, y_pred, color='red', label='Regression Line')
plt.xlabel('sgpi')
plt.ylabel('marks_inmaths')
plt.legend()
plt.show()
```

OUTPUT:

```
      seat_no           prn  centre  total_gradepoints  sgpi       status
\
0     5201541  2.020020e+15  Mumbai                201  9.14  Successful
1     5201542  2.020020e+15  Mumbai                178  8.09  Successful
2     5201543  2.018020e+15  Mumbai                175  7.95  Successful
3     5201544  2.019020e+15  Mumbai                166  7.55  Successful
4     5201545  2.019020e+15  Mumbai                177  8.05  Successful
...       ...           ...     ...                ...   ...         ...
5206  5018234  2.019020e+15   Thane                187  8.50  Successful
5207  5018235  2.019020e+15   Thane                193  8.77  Successful
5208  5018236  2.019020e+15   Thane                170  7.73  Successful
5209  5018237  2.019020e+15   Thane                175  7.95  Successful
5210  5018238  2.018020e+15   Thane                151  6.86  Successful

      year_of_admission  clg_id gender  cgpa  marks_in_phy  marks_in_chem
\
0                  2020      10      M  9.14            49             45
1                  2020      10      M  8.09            39             48
2                  2018      10      M  7.95            39             45
3                  2019      10      M  7.55            47             46
4                  2019      10      M  8.05            49             49
...                 ...     ...    ...   ...           ...            ...
5206               2019     996      M  8.50            40             45
5207               2019     996      M  8.77            41             49
5208               2019     996      F  7.73            48             49
5209               2019     996      M  7.95            37             48
5210               2018     996      M  6.86            47             50

      marks_inmaths
0                48
1                39
2                45
3                48
4                44
...             ...
5206             44
5207             50
5208             49
5209             49
5210             35
```



KNN

```
# Drop the missing values
df = df.dropna()
X=df['marks_in_phy']
df=df.dropna()
Y=df['marks_inmaths']
X=np.array(df['marks_in_phy']).reshape(-1,1)
Y=np.array(df['marks_inmaths']).reshape(-1,1)
X_train, X_test,y_train, y_test = train_test_split(X,Y,test_size=0.30)

from sklearn.metrics import classification_report,\
    confusion_matrix

knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)
pred = knn.predict(X_test)

# Predictions and Evaluations
# Let's evaluate our KNN model !
print(confusion_matrix(y_test, pred))
```

```
print(classification_report(y_test, pred))
```

Output

```
[[ 0  0  8  0  0 20  7  7 15  5 11 12  6  0  8  0]
 [ 0  0  7  0  0 19  6  6 11  6 15  9  9  0  9  0]
 [ 0  0  8  0  0 18  4 11 16  4 10 15 11  0 17  0]
 [ 0  0  8  0  0 19 11  3 11 10 11 15  6  0 10  0]
 [ 0  0  9  0  0 14  3  9  6  9  9  7  4  0  7  0]
 [ 0  0  6  0  0 21  3  4 12  7 13  8  1  0  9  0]
 [ 0  0  4  0  0 14  5  9 14  3 17  5  5  0 16  0]
 [ 0  0  5  0  0 19  7  4 20  9 17  9  7  0 14  0]
 [ 0  0  7  0  0 12  4  4 17 10 15 11  8  0 18  0]
 [ 0  0  5  0  0 19  7  8  9  9 20 12  7  0 11  0]
 [ 0  0  5  0  0 21  3  4 19  3 12  8  9  0 11  0]
 [ 0  0  9  0  0 20  6 10  4  4 13 15 11  0 10  0]
 [ 0  0  7  0  0 17  8  2 14  5 11 17  4  0 11  0]
 [ 0  0  7  0  0 15  6  2 10  5 10 14  5  0 10  0]
 [ 0  0 12  0  0 25  6  7 12  3 10 10  9  0 13  0]
 [ 0  0  5  0  0 16  4  6 11  5 12  7  8  0  8  0]]
              precision    recall  f1-score   support

          35       0.00      0.00      0.00        99
          36       0.00      0.00      0.00        97
          37       0.07      0.07      0.07       114
          38       0.00      0.00      0.00       104
          39       0.00      0.00      0.00        77
          40       0.07      0.25      0.11        84
          41       0.06      0.05      0.05        92
          42       0.04      0.04      0.04       111
          43       0.08      0.16      0.11       106
          44       0.09      0.08      0.09       107
          45       0.06      0.13      0.08        95
          46       0.09      0.15      0.11       102
          47       0.04      0.04      0.04        96
          48       0.00      0.00      0.00        84
          49       0.07      0.12      0.09       107
          50       0.00      0.00      0.00        82

    accuracy                           0.07      1557
   macro avg       0.04      0.07      0.05      1557
weighted avg       0.04      0.07      0.05      1557
```

```
# Drop the missing values
df = df.dropna()
X=df['sgpi']
df=df.dropna()
Y=df['year_of_admission']
X=np.array(df['sgpi']).reshape(-1,1)
Y=np.array(df['year_of_admission']).reshape(-1,1)
X_train, X_test,y_train, y_test = train_test_split(X,Y,test_size=0.30)

from sklearn.metrics import classification_report,\
```

```
    confusion_matrix

knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)
pred = knn.predict(X_test)

# Predictions and Evaluations
# Let's evaluate our KNN model !
print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred))
```

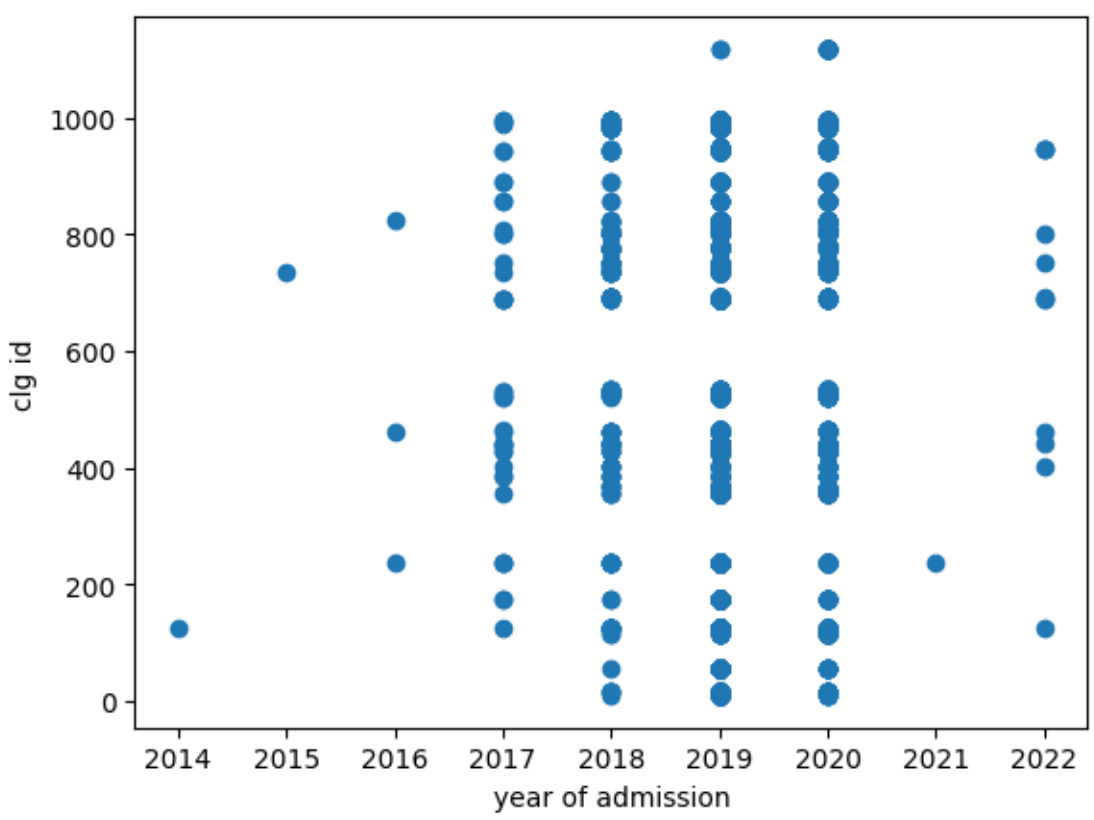
OUTPUT:

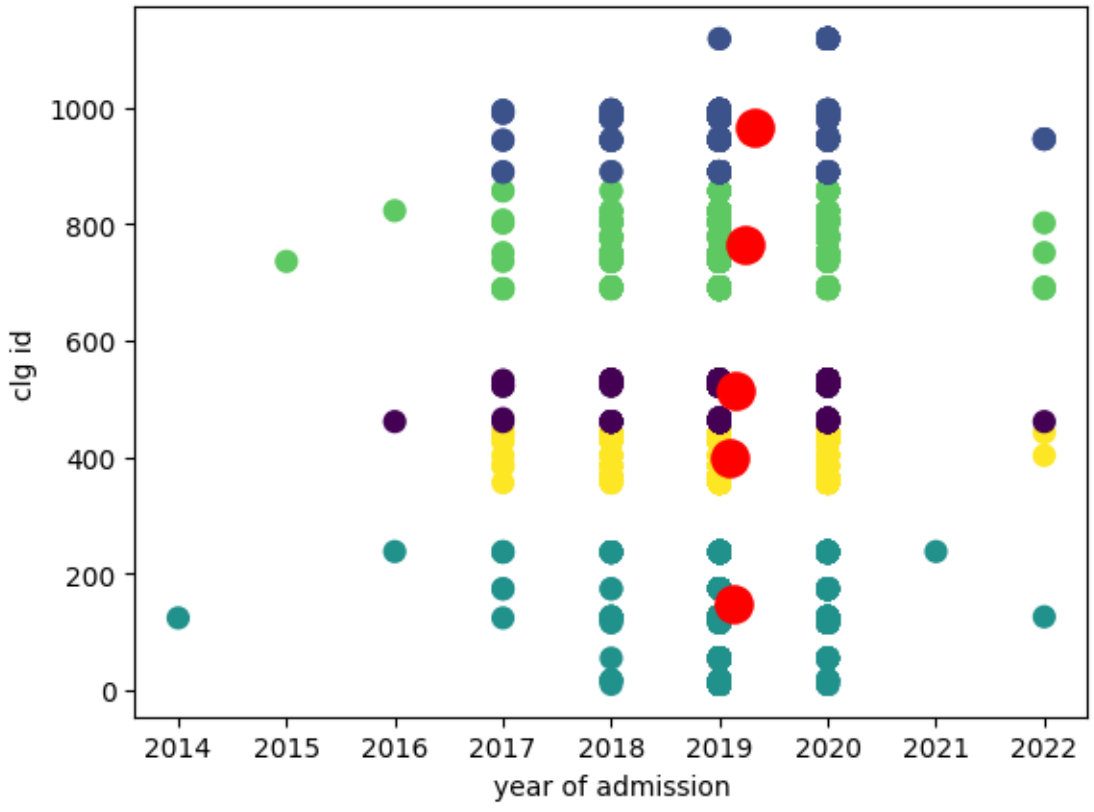
```
[[  0   0   0   0   1   0   0   0]
 [  0   0   0   0   0   0   0   0]
 [  0   0   0   0   5   8   0   0]
 [  0   0   3   2  29  28   0   0]
 [  0   0  14   9 809 247   0   0]
 [  0   1   2  10 229 157   0   0]
 [  0   0   0   0   0   1   0   0]
 [  0   0   0   0   0   2   0   0]]
              precision    recall  f1-score   support

        2014       0.00      0.00      0.00         1
        2016       0.00      0.00      0.00         0
        2017       0.00      0.00      0.00        13
        2018       0.10      0.03      0.05        62
        2019       0.75      0.75      0.75      1079
        2020       0.35      0.39      0.37       399
        2021       0.00      0.00      0.00         1
        2022       0.00      0.00      0.00         2

    accuracy                           0.62      1557
   macro avg       0.15      0.15      0.15      1557
weighted avg       0.62      0.62      0.62      1557
```

K MEANS CLUSTERING

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

df = pd.read_csv("/content/sample_data/ass4_dataset.csv")
Data = {'x': df["year_of_admission"], 'y': df["clg_id"]}
df=pd.DataFrame(Data, columns=['x', 'y'])

plt.xlabel("year of admission")
```

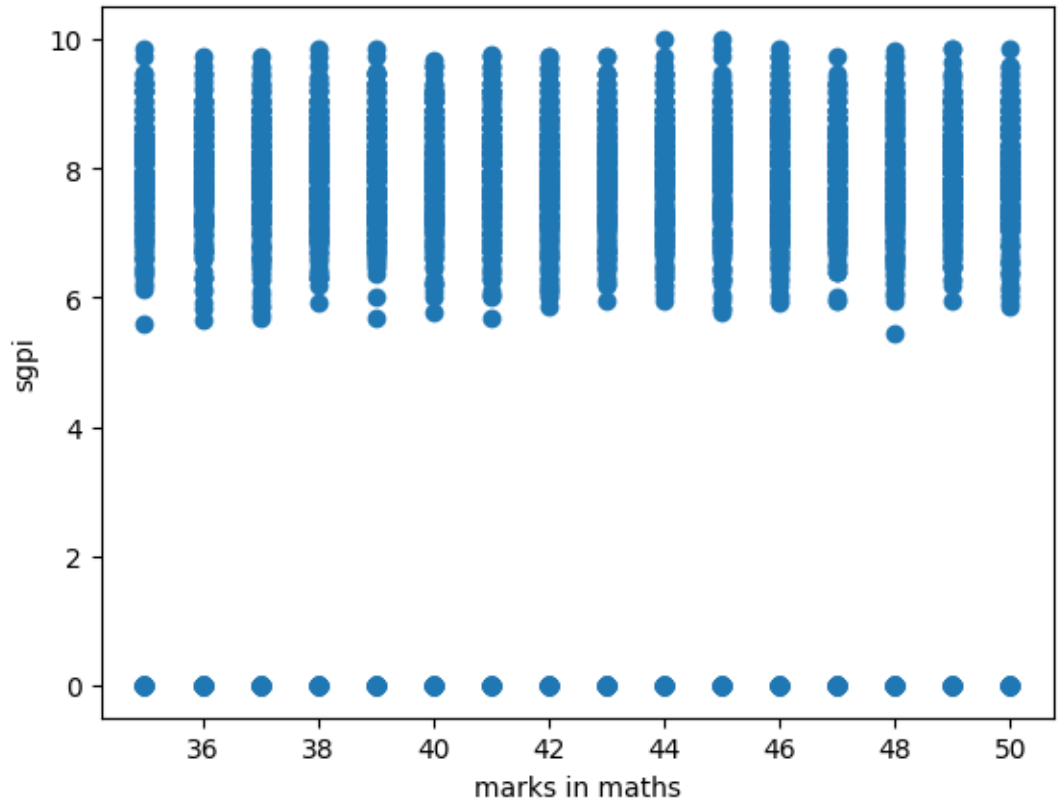
```
plt.ylabel("clg id")

plt.scatter(df['x'], df['y'])
plt.show()
df.dropna(inplace=True)

km = KMeans(n_clusters=5).fit(df)
centroids = km.cluster_centers_

plt.xlabel("year of admission")
plt.ylabel("clg id")
plt.scatter(df['x'], df['y'], c=km.labels_.astype(float), s=60, alpha=1)
plt.scatter(centroids[:, 0], centroids[:, 1 c='red', s=190)
plt.show()
```

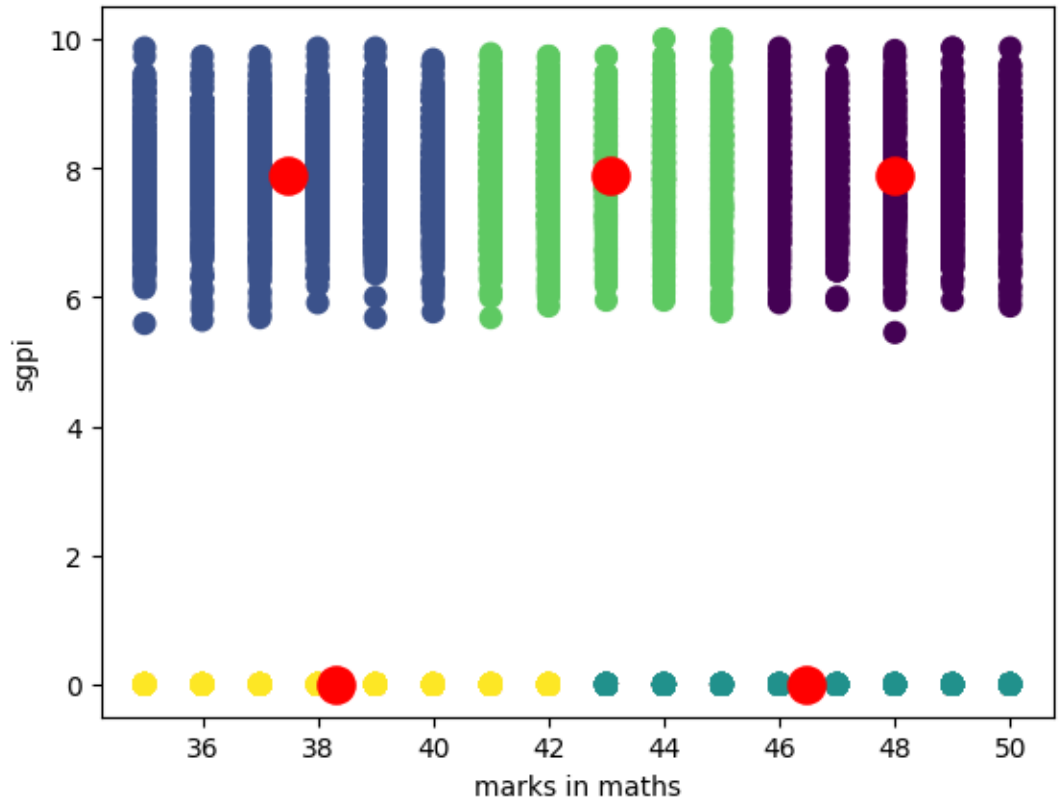

clg id
year of admission
0
200
400
600
800
1000
2014
2015
2016
2017
2018
2019
2020
2021
2022


clg id
year of admission
0
200
400
600
800
1000
2014
2015
2016
2017
2018
2019
2020
2021
2022

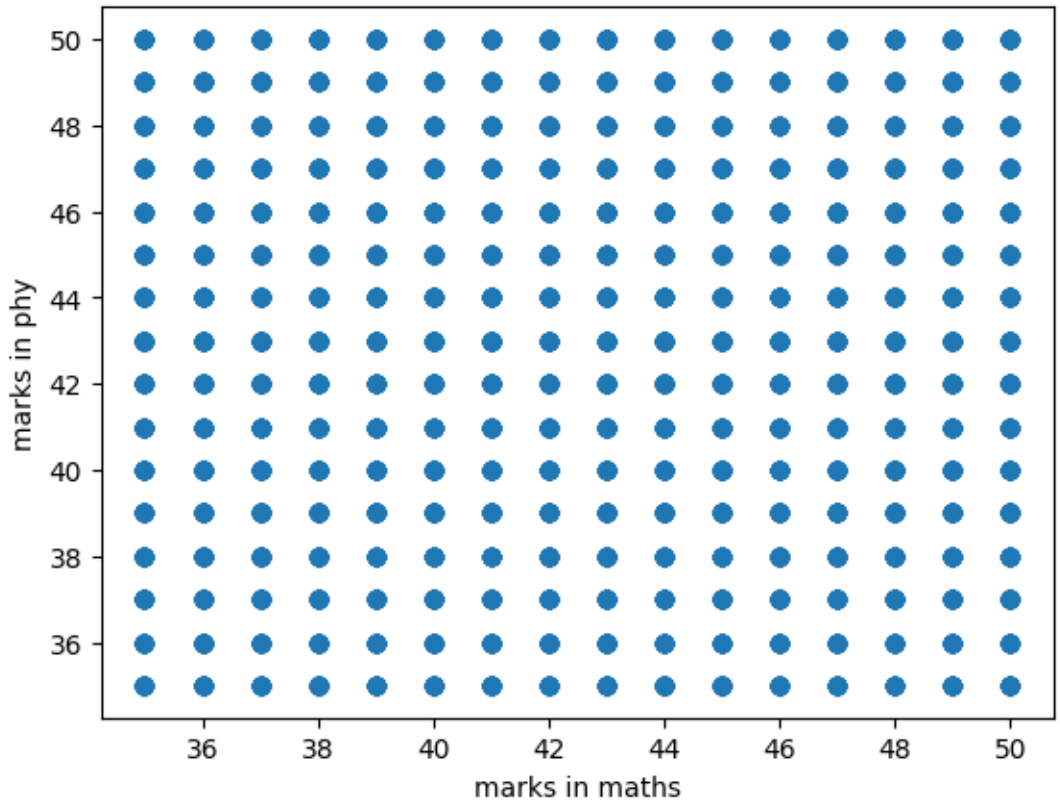
],

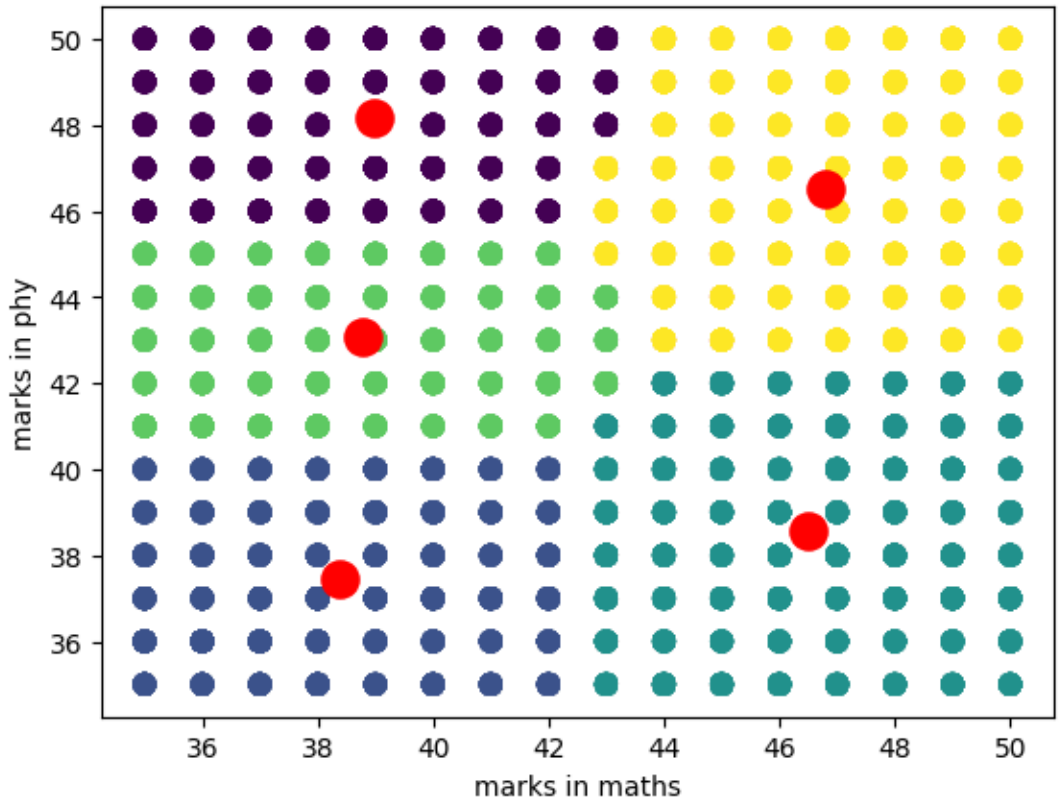
```
# K MEANS CLUSTERING
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
df = pd.read_csv("/content/sample_data/ass4_dataset.csv")
Data = {'x': df["marks_inmaths"], 'y': df["sgpi"]}
df=pd.DataFrame(Data, columns=['x', 'y'])
plt.xlabel("marks in maths")
plt.ylabel("sgpi")
plt.scatter(df['x'], df['y'])
plt.show()
df.dropna(inplace=True)
km = KMeans(n_clusters=5).fit(df)
centroids = km.cluster_centers_
plt.xlabel("marks in maths")
plt.ylabel("sgpi")
plt.scatter(df['x'], df['y'], c=km.labels_.astype(float), s=60, alpha=1)
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=190)
plt.show()
```


sgpi
marks in maths
0
2
4
6
8
10
36
38
40
42
44
46
48
50


sgpi
marks in maths
0
2
4
6
8
10
36
38
40
42
44
46
48
50

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

df = pd.read_csv("/content/sample_data/ass4_dataset.csv")
Data = {'x': df["marks_inmaths"], 'y': df["marks_in_phy"]}
df=pd.DataFrame(Data, columns=['x', 'y'])
plt.xlabel("marks in maths")
plt.ylabel("marks in phy")
plt.scatter(df['x'], df['y'])
plt.show()
df.dropna(inplace=True)
km = KMeans(n_clusters=5).fit(df)
centroids = km.cluster_centers_
plt.xlabel("marks in maths")
plt.ylabel("marks in phy")
plt.scatter(df['x'], df['y'], c=km.labels_.astype(float), s=60, alpha=1)
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=190)
plt.show()
```


marks in phy
marks in maths
50
48
46
44
42
40
38
36
36
38
40
42
44
46
48
50


marks in phy
marks in maths
50
48
46
44
42
40
38
36
36
38
40
42
44
46
48
50