pandas

```
#inporting
import pandas as pd
```

serise

```
mydata=["ananya","aishwitha","nayana","navya","gowthami"]
Ser1=pd.Series(mydata)                                              #series
are always 1D
print(Ser1)
```

```
0       ananya
1    aishwitha
2       nayana
3        navya
4     gowthami
dtype: object
```

```
mydata=["ananya","aishwitha","nayana","navya","gowthami"]
roll=[2,14,21,20,12]
Ser2=pd.Series(mydata,index=roll)
print(Ser2)
```

```
2        ananya
14    aishwitha
21       nayana
20        navya
12     gowthami
dtype: object
```

```
Ser1[3]
```

```
'navya'
```

```
Ser2[14]
```

```
'aishwitha'
```

```
mydata=["ananya","aishwitha","nayana","navya","gowthami"]      #Series
always S is captial
roll=['A','B','C','D','E']
Ser2=pd.Series(mydata,index=roll)
print(Ser2)
```

```
A       ananya
B    aishwitha
C       nayana
```

```
D         navya
E      gowthami
dtype: object
```

```
Ser2['B']
```

```
'aishwitha'
```

```
Ser2.to_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\mydata.csv")
#unicode error in that time we put r ,it will make uniqe ,it will
create a new file in that location.
```

DataFrames

```
mydirct={"Names":["raj","sumith","rakesh"],
         "age":[19,19,20],
         "city":["raichur","shivmoga","bidar"]
         }
print(mydirct)
```

```
{'Names': ['raj', 'sumith', 'rakesh'], 'age': [19, 19, 20], 'city':
['raichur', 'shivmoga', 'bidar']}
```

```
dict_df=pd.DataFrame(mydirct)
print(dict_df)
```

```
    Names  age      city
0     raj   19   raichur
1  sumith   19  shivmoga
2  rakesh   20     bidar
```

```
dict_df.to_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\mydirct.csv")
# to.csv ->in this we are aplode the the data
```

load data

```
df1=pd.read_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\
bhoomika.csv")    # .read_csv->is used to already existing data can be
shown
```

```
df1.head()
```

```
       name Dept  Sem1  Sem2  Sem3
0       Sam  ISE   7.2   8.9   9.0
1  Bhoomika  ISE   9.2   8.9   9.0
2     megha  ISE   8.8   NaN   9.0
3    sindhu  ISE   8.9   7.8   8.8
```

```
debt_df=pd.read_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\
diabetcsvsmall.csv")

debt_df.head() #head will getting only first 5 rows
```

```
   preg  plas  pres  skin  insu  mass   pedi  age            class
0   6.0   148  72.0  35.0     0  33.6  0.627   50  tested_positive
1   1.0    85  66.0  29.0     0  26.6  0.351   31  tested_negative
2   8.0   183  64.0   0.0     0  23.3  0.672   32  tested_positive
3   1.0    89  66.0  23.0    94  28.1  0.167   21  tested_negative
4   0.0   137  40.0  35.0   168  43.1  2.288   33  tested_positive
```

```
debt_df.tail()   #appers last five rows
```

```
     preg  plas  pres  skin  insu  mass   pedi  age            class
97    1.0    71  48.0   NaN    76  20.4  0.323   22  tested_negative
98    6.0    93  50.0  30.0    64  28.7  0.356   23  tested_negative
99    NaN   122  90.0  51.0   220  49.7  0.325   31  tested_positive
100   1.0   163  72.0   0.0     0  39.0  1.222   33  tested_positive
101   1.0   151  60.0   0.0     0  26.1  0.179   22  tested_negative
```

access

```
loc->location
iloc->integer location,it accept only the integer number

debt_df.loc[12:19]
```

```
    preg  plas  pres  skin  insu  mass   pedi  age            class
12  10.0   139  80.0   0.0     0  27.1  1.441   57  tested_negative
13   1.0   189  60.0  23.0   846  30.1  0.398   59  tested_positive
14   5.0   166  72.0  19.0   175  25.8  0.587   51  tested_positive
15   7.0   100   0.0   0.0     0  30.0  0.484   32  tested_positive
16   0.0   118  84.0  47.0   230  45.8  0.551   31  tested_positive
17   7.0   107  74.0   0.0     0  29.6  0.254   31  tested_positive
18   1.0   103  30.0  38.0    83  43.3  0.183   33  tested_negative
19   1.0   115  70.0  30.0    96  34.6  0.529   32  tested_positive
```

```
debt_df.loc[12:19,"age"]
```

```
12    57
13    59
14    51
15    32
16    31
17    31
18    33
19    32
Name: age, dtype: int64
```

```
debt_df.iloc[12:19,3:8]   #only index wise 3:8 is staring from the 
3[column range] up to 8[column range]

    skin  insu  mass   pedi  age
12   0.0     0  27.1  1.441   57
13  23.0   846  30.1  0.398   59
14  19.0   175  25.8  0.587   51
15   0.0     0  30.0  0.484   32
16  47.0   230  45.8  0.551   31
17   0.0     0  29.6  0.254   31
18  38.0    83  43.3  0.183   33
```

feature engineering

```
preg plas pres skin insu mass pedi age  ==>independent(Feature)
class is the dependent(target)
eg=time[independent]
weight[dependent]
inplace=true is change the paramenently database
rename will change the only in that time not paramanently

debt_df.rename(columns ={"plas":"glucose"})     # not paramently 
changed

     preg  glucose  pres  skin  insu  mass   pedi  age
class
0     6.0      148  72.0  35.0     0  33.6  0.627   50
tested_positive
1     1.0       85  66.0  29.0     0  26.6  0.351   31
tested_negative
2     8.0      183  64.0   0.0     0  23.3  0.672   32
tested_positive
3     1.0       89  66.0  23.0    94  28.1  0.167   21
tested_negative
4     0.0      137  40.0  35.0   168  43.1  2.288   33
tested_positive
..    ...      ...   ...   ...   ...   ...    ...  ...               ..
.
97    1.0       71  48.0   NaN    76  20.4  0.323   22
tested_negative
98    6.0       93  50.0  30.0    64  28.7  0.356   23
tested_negative
99    NaN      122  90.0  51.0   220  49.7  0.325   31
tested_positive
100   1.0      163  72.0   0.0     0  39.0  1.222   33
tested_positive
101   1.0      151  60.0   0.0     0  26.1  0.179   22
tested_negative
```

```
[102 rows x 9 columns]

debt_df.rename(columns ={"plas":"glucose"},inplace = True)
#paramently changed [inplace = True]
debt_df.head()

   preg  glucose  pres  skin  insu  mass   pedi  age            class
0   6.0      148  72.0  35.0     0  33.6  0.627   50  tested_positive
1   1.0       85  66.0  29.0     0  26.6  0.351   31  tested_negative
2   8.0      183  64.0   0.0     0  23.3  0.672   32  tested_positive
3   1.0       89  66.0  23.0    94  28.1  0.167   21  tested_negative
4   0.0      137  40.0  35.0   168  43.1  2.288   33  tested_positive

debt_df.head()

   preg  glucose  pres  skin  insu  mass   pedi  age            class
0   6.0      148  72.0  35.0     0  33.6  0.627   50  tested_positive
1   1.0       85  66.0  29.0     0  26.6  0.351   31  tested_negative
2   8.0      183  64.0   0.0     0  23.3  0.672   32  tested_positive
3   1.0       89  66.0  23.0    94  28.1  0.167   21  tested_negative
4   0.0      137  40.0  35.0   168  43.1  2.288   33  tested_positive

debt_df['glucose_in_mmol'] =debt_df['glucose']/18.018
#dataframe ['new clo name']=content(formula)
#converting glucose from mg to mmol and creating new col

debt_df.head()

   preg  glucose  pres  skin  insu  mass   pedi  age            class
\
0   6.0      148  72.0  35.0     0  33.6  0.627   50  tested_positive

1   1.0       85  66.0  29.0     0  26.6  0.351   31  tested_negative

2   8.0      183  64.0   0.0     0  23.3  0.672   32  tested_positive

3   1.0       89  66.0  23.0    94  28.1  0.167   21  tested_negative

4   0.0      137  40.0  35.0   168  43.1  2.288   33  tested_positive


   glucose_in_mmol
0         8.214008
1         4.717505
2        10.156510
3         4.939505
4         7.603508
```

filter and groups

```
fil_age_30less =debt_df[debt_df['age']<30]
fil_age_30less.head()
```

```
    preg  glucose  pres  skin  insu  mass   pedi  age            class  \
3    1.0       89  66.0  23.0    94  28.1  0.167   21  tested_negative   
6    3.0       78  50.0  32.0    88  31.0  0.248   26  tested_positive   
7   10.0      115   0.0   0.0     0  35.3  0.134   29  tested_negative   
20   3.0      126  88.0  41.0   235  39.3  0.704   27  tested_negative   
23   9.0      119  80.0  35.0     0  29.0  0.263   29  tested_positive   

    glucose_in_mmol  
3          4.939505  
6          4.329004  
7          6.382506  
20         6.993007  
23         6.604507  
```

```
glu_above100=debt_df[debt_df['glucose']>100]
glu_above100.head(7)
```

```
   preg  glucose  pres  skin  insu  mass   pedi  age            class  \
0   6.0      148  72.0  35.0     0  33.6  0.627   50  tested_positive   
2   8.0      183  64.0   0.0     0  23.3  0.672   32  tested_positive   
4   0.0      137  40.0  35.0   168  43.1  2.288   33  tested_positive   
5   5.0      116  74.0   0.0     0  25.6  0.201   30  tested_negative   
7  10.0      115   0.0   0.0     0  35.3  0.134   29  tested_negative   
8   2.0      197  70.0  45.0   543  30.5  0.158   53  tested_positive   
9   8.0      125  96.0   0.0     0   0.0  0.232   54  tested_positive   

   glucose_in_mmol  
0         8.214008  
2        10.156510  
4         7.603508  
5         6.438006  
7         6.382506  
```

```
8        10.933511
9         6.937507

glu_below100=debt_df[debt_df['glucose']<100]
glu_below100.head(7)

    preg  glucose  pres  skin  insu  mass   pedi  age            class
\
1    1.0       85  66.0  29.0     0  26.6  0.351   31  tested_negative

3    1.0       89  66.0  23.0    94  28.1  0.167   21  tested_negative

6    3.0       78  50.0  32.0    88  31.0  0.248   26  tested_positive

21   8.0       99  84.0   0.0     0  35.4  0.388   50  tested_negative

27   1.0       97  66.0  15.0   140  23.2  0.487   22  tested_negative

32   3.0       88  58.0  11.0    54  24.8  0.267   22  tested_negative

33   6.0       92  92.0   0.0     0  19.9  0.188   28  tested_negative


    glucose_in_mmol
1          4.717505
3          4.939505
6          4.329004
21         5.494505
27         5.383505
32         4.884005
33         5.106005
```

create a filter data set which as only the rows with age b/w 20 and 30

```
age_above20and30=debt_df[(debt_df['age']>20) & (debt_df['age']<30)]
age_above20and30.head(7)

    preg  plas  pres  skin  insu  mass   pedi  age            class
3    1.0    89  66.0  23.0    94  28.1  0.167   21  tested_negative
6    3.0    78  50.0  32.0    88  31.0  0.248   26  tested_positive
7   10.0   115   0.0   0.0     0  35.3  0.134   29  tested_negative
20   3.0   126  88.0  41.0   235  39.3  0.704   27  tested_negative
23   9.0   119  80.0  35.0     0  29.0  0.263   29  tested_positive
27   1.0    97  66.0  15.0   140  23.2  0.487   22  tested_negative
31   3.0   158  76.0  36.0   245  31.6  0.851   28  tested_positive

group_by_class_age = debt_df.groupby('class')['age'].mean()
group_by_class_age
#grouped by class and calculate average age
#grouped age if adabitices people is 40.5
#grouped age if non- adabitices people is 31.2
```

```
class
tested_negative    31.238095
tested_positive    40.589744
Name: age, dtype: float64
```

```
group_by_class_age = debt_df.groupby('class')['insu'].mean()
group_by_class_age
#grouped by class and calculate average insu
#average insulin if non- dabitices people is 114.69
#average insulin if non- dabitices people is 52.5714
```

```
class
tested_negative     52.571429
tested_positive    114.692308
Name: insu, dtype: float64
```

```
group_by_class_age = debt_df.groupby('class')['age'].max()
group_by_class_age
#grouped by class and calculate max age
#the least age of diabites is 60
#the least age of non-diabites is 60
```

```
class
tested_negative    60
tested_positive    60
Name: age, dtype: int64
```

```
group_by_class_age = debt_df.groupby('class')['insu'].max()
group_by_class_age
#grouped by class and calculate min insu
#the least age of diabites is 846
#the least age of non-diabites is 342
```

```
class
tested_negative    342
tested_positive    846
Name: insu, dtype: int64
```

```
group_by_class_age = debt_df.groupby('class')['age'].min()
group_by_class_age
#grouped by class and calculate min age
#the least age of diabites is 25
#the least age of non-diabites is 21
```

```
class
tested_negative    21
tested_positive    25
Name: age, dtype: int64
```

cleaning data

handling null

```
debt_df.isnull()
```

```
      preg   plas   pres   skin   insu   mass   pedi    age  class
0    False  False  False  False  False  False  False  False  False
1    False  False  False  False  False  False  False  False  False
2    False  False  False  False  False  False  False  False  False
3    False  False  False  False  False  False  False  False  False
4    False  False  False  False  False  False  False  False  False
..     ...    ...    ...    ...    ...    ...    ...    ...    ...
97   False  False  False   True  False  False  False  False  False
98   False  False  False  False  False  False  False  False  False
99    True  False  False  False  False  False  False  False  False
100  False  False  False  False  False  False  False  False  False
101  False  False  False  False  False  False  False  False  False

[102 rows x 9 columns]
```

```
debt_df.isnull().sum()
```

```
preg     1
plas     0
pres     1
skin     1
insu     0
mass     1
pedi     1
age      0
class    0
dtype: int64
```

```
debt_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102 entries, 0 to 101
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   preg    101 non-null    float64
 1   plas    102 non-null    int64  
 2   pres    101 non-null    float64
 3   skin    101 non-null    float64
 4   insu    102 non-null    int64  
 5   mass    101 non-null    float64
 6   pedi    101 non-null    float64
 7   age     102 non-null    int64  
 8   class   102 non-null    object 
dtypes: float64(5), int64(3), object(1)
memory usage: 7.3+ KB
```

```
debt_df.dropna    #it will show thw null value by 0
```

```
     preg  plas  pres  skin  insu  mass   pedi  age            class
0     6.0   148  72.0  35.0     0  33.6  0.627   50  tested_positive
1     1.0    85  66.0  29.0     0  26.6  0.351   31  tested_negative
2     8.0   183  64.0   0.0     0  23.3  0.672   32  tested_positive
3     1.0    89  66.0  23.0    94  28.1  0.167   21  tested_negative
4     0.0   137  40.0  35.0   168  43.1  2.288   33  tested_positive
..    ...   ...   ...   ...   ...   ...    ...  ...              ...
95    6.0   144  72.0  27.0   228  33.9  0.255   40  tested_negative
96    2.0    92  62.0  28.0     0  31.6  0.130   24  tested_negative
98    6.0    93  50.0  30.0    64  28.7  0.356   23  tested_negative
100   1.0   163  72.0   0.0     0  39.0  1.222   33  tested_positive
101   1.0   151  60.0   0.0     0  26.1  0.179   22  tested_negative

[98 rows x 9 columns]
```

```
debt_df.dropna(inplace=True)  #it will remove the all null values by using inplace original values are changed
```

```
debt_df.isnull().sum()
```

```
preg     0
plas     0
pres     0
skin     0
insu     0
mass     0
pedi     0
age      0
class    0
dtype: int64
```

handeling duplictes

```
debt_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 98 entries, 0 to 101
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   preg    98 non-null     float64
 1   plas    98 non-null     int64
 2   pres    98 non-null     float64
 3   skin    98 non-null     float64
 4   insu    98 non-null     int64
 5   mass    98 non-null     float64
 6   pedi    98 non-null     float64
 7   age     98 non-null     int64
 8   class   98 non-null     object
```

```
dtypes: float64(5), int64(3), object(1)
memory usage: 7.7+ KB

debt_df.drop_duplicates(inplace=True)   #it will remove the duplicates

debt_df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 96 entries, 0 to 101
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   preg    96 non-null     float64
 1   plas    96 non-null     int64
 2   pres    96 non-null     float64
 3   skin    96 non-null     float64
 4   insu    96 non-null     int64
 5   mass    96 non-null     float64
 6   pedi    96 non-null     float64
 7   age     96 non-null     int64
 8   class   96 non-null     object
dtypes: float64(5), int64(3), object(1)
memory usage: 7.5+ KB
```

lodading taxt file

```
diab_ex=pd.read_excel(r"C:\Users\Bhoomika.G\Downloads\
diabetes.xlsx",sheet_name="Hello")
diab_ex

Empty DataFrame
Columns: [hello, guys, how, are ]
Index: []

diab_ex=pd.read_excel(r"C:\Users\Bhoomika.G\Downloads\
diabetes.xlsx",sheet_name="dora")
diab_ex

  Dead Alive
0  yes    no
1  yes    no
2  yes    no
3  yes    no
4  yes    no

df_txt=pd.read_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\
grades.txt")   # in this all the values are in one box
df_txt.head()

  Names Initials SEM1 SEM2 SEM3 Grade
0                 Joe K 9.8 10 9.9 A+
1              Rajesh M 8.9 9.1 9.3 A
2              Kissan V 9.9 9.3 9.2 A
3                  Mary N 7.7 8 7.1 B
4               Jeen K 9.8 9.1 9.9 A+
```

```
df_txt=pd.read_csv(r"C:\Users\Bhoomika.G\OneDrive\Documents\
grades.txt",sep=" ")   #in all the values are saparate by the sep
df_txt.head()
```

```
    Names Initials  SEM1  SEM2  SEM3 Grade
0     Joe        K   9.8  10.0   9.9    A+
1  Rajesh        M   8.9   9.1   9.3     A
2  Kissan        V   9.9   9.3   9.2     A
3    Mary        N   7.7   8.0   7.1     B
4    Jeen        K   9.8   9.1   9.9    A+
```

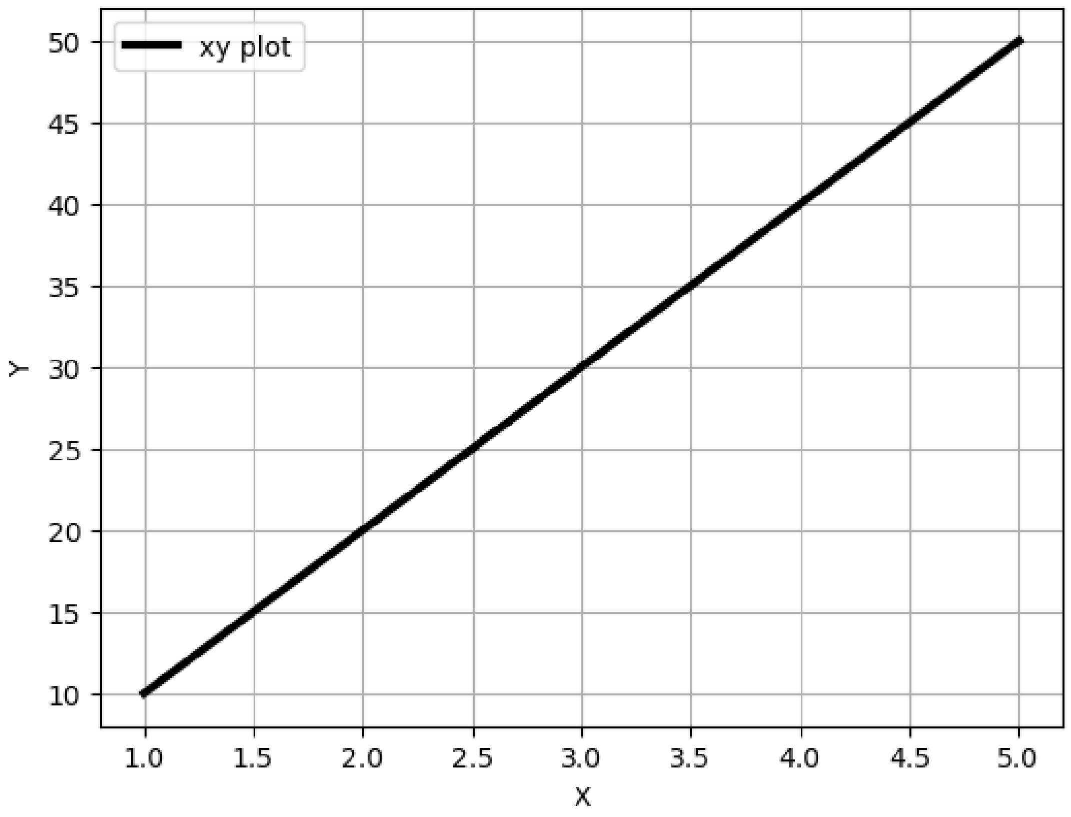
```
df_txt['SEM1_int']=df_txt['SEM1'].astype(int)  #another column is
created it will remove flot by int [modify the data type]
df_txt.head()
```

```
    Names Initials  SEM1  SEM2  SEM3 Grade  SEM1_int
0     Joe        K   9.8  10.0   9.9    A+         9
1  Rajesh        M   8.9   9.1   9.3     A         8
2  Kissan        V   9.9   9.3   9.2     A         9
3    Mary        N   7.7   8.0   7.1     B         7
4    Jeen        K   9.8   9.1   9.9    A+         9
```

mathplotlib

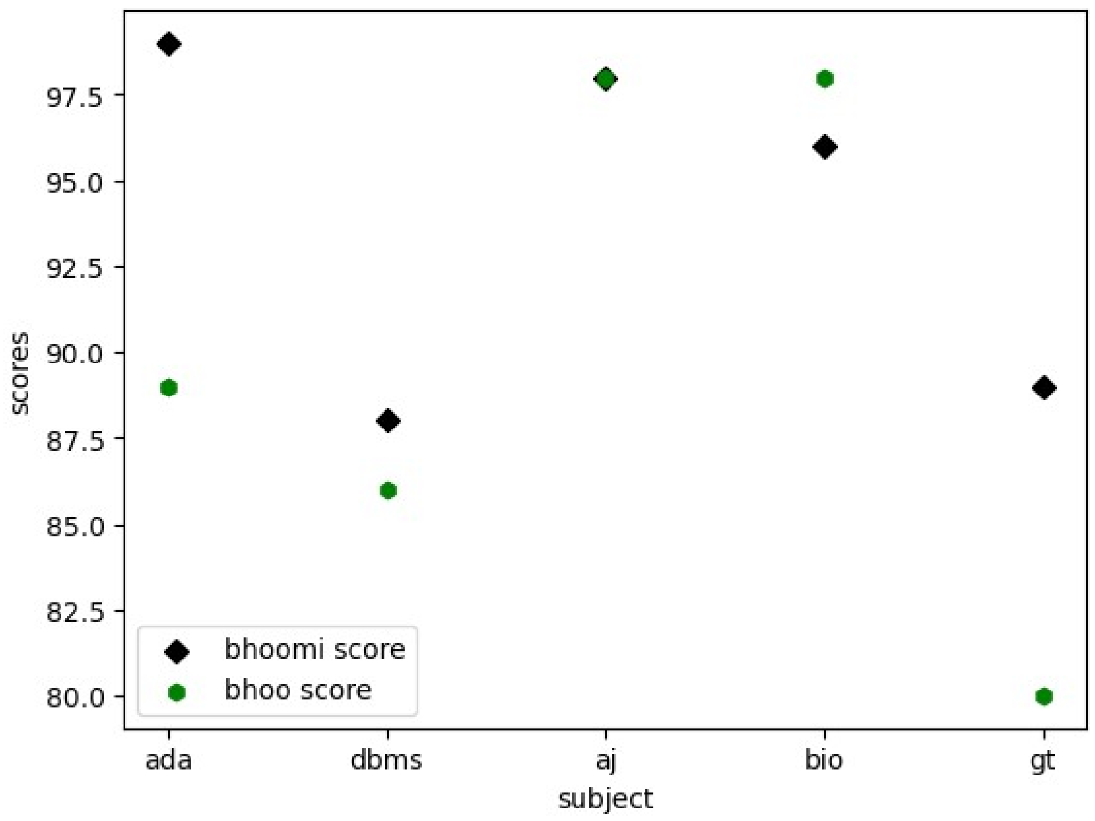
```
x=[1,2,3,4,5]
y=[10,20,30,40,50]

import matplotlib.pyplot as plt
plt.plot(x,y,color='k',label="xy plot",linestyle="-",linewidth=3)
plt.xlabel('X')
plt.ylabel('Y')
plt.grid()
plt.legend()
```

```
<matplotlib.legend.Legend at 0x1f4628633e0>
```

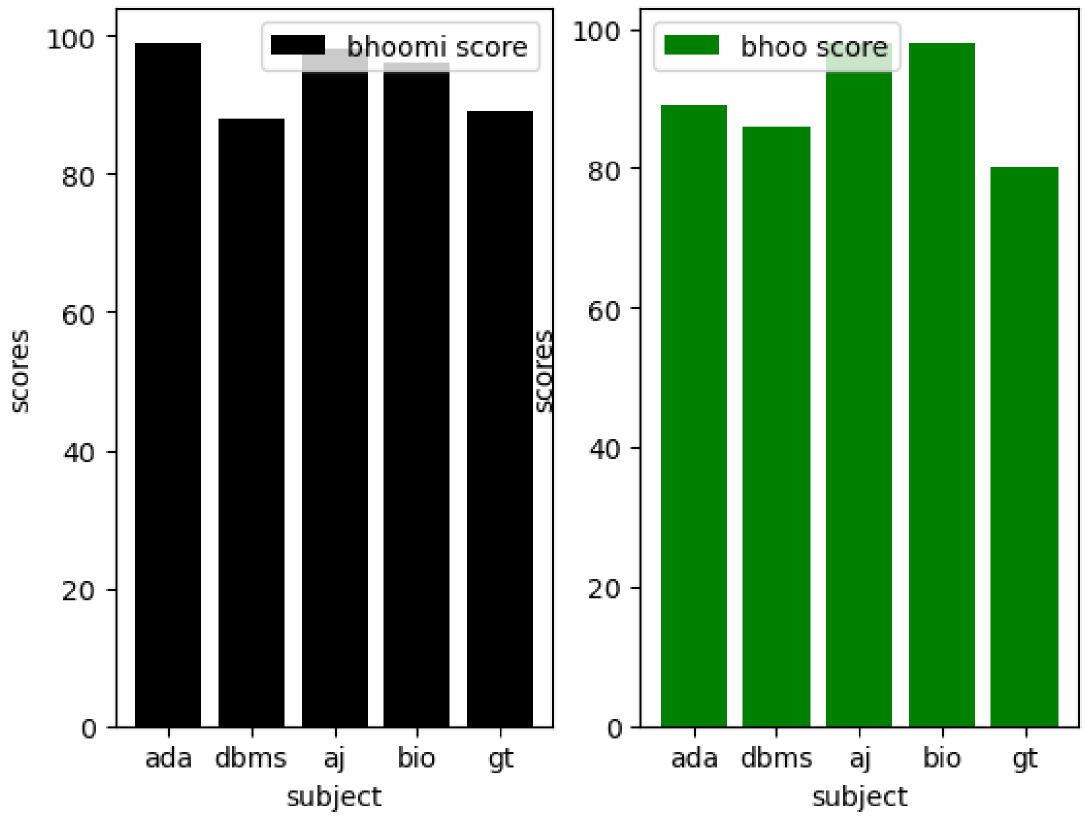


```
import matplotlib.pyplot as plt
sub=["ada","dbms","aj","bio","gt"]
bhoomi=[99,88,98,96,89]
bhoo=[89,86,98,98,80]
plt.scatter(sub,bhoomi,color='k',label='bhoomi score',marker='D')
plt.scatter(sub,bhoo,color='green',label='bhoo score',marker='h')
plt.xlabel('subject')
plt.ylabel('scores')
plt.legend()
```

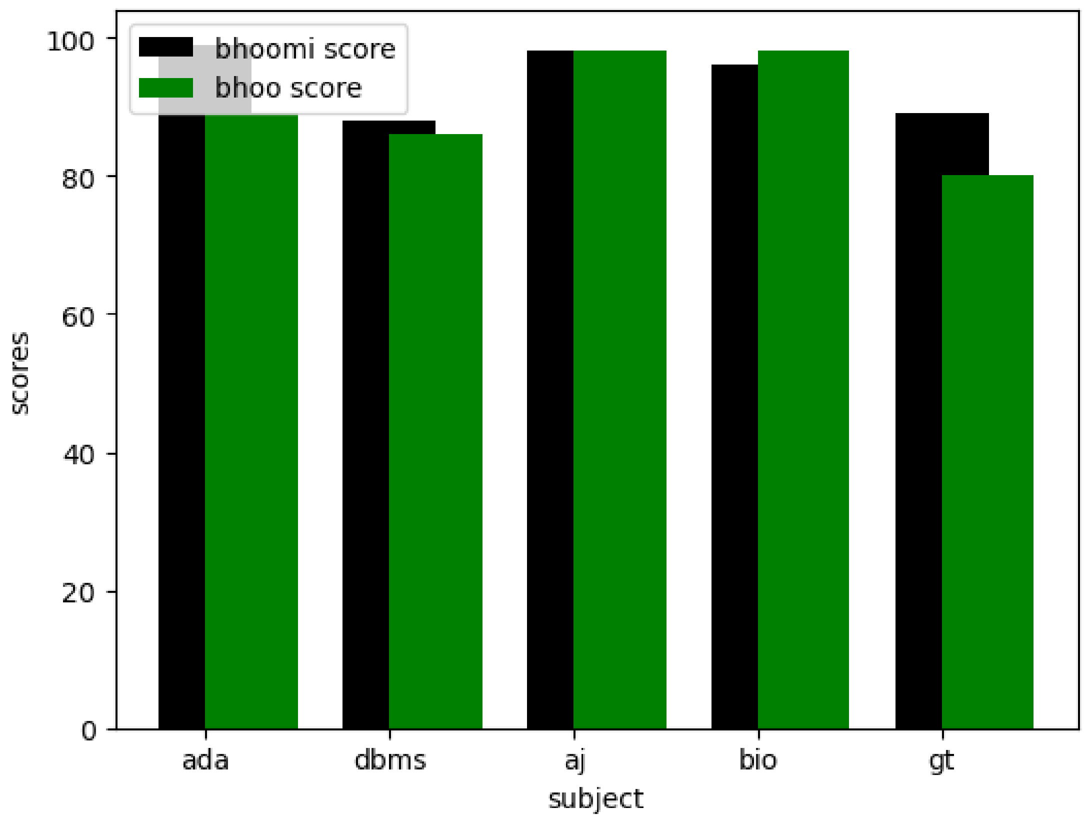
```
<matplotlib.legend.Legend at 0x1f462861760>
```



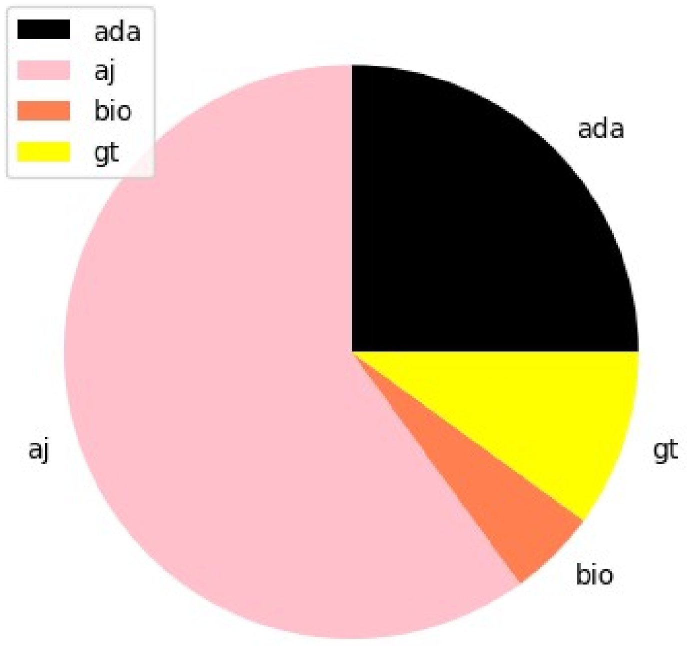
```
import matplotlib.pyplot as plt
sub=["ada","dbms","aj","bio","gt"]
bhoomi=[99,88,98,96,89]
bhoo=[89,86,98,98,80]
width=0.4
#first plot
plt.subplot(1,2,1)
plt.bar(sub,bhoomi,color='k',label='bhoomi score')
plt.xlabel('subject')
plt.ylabel('scores')
plt.legend()
#second plot
plt.subplot(1,2,2)
plt.bar(sub,bhoo,color='green',label='bhoo score')
plt.xlabel('subject')
plt.ylabel('scores')
plt.legend()

<matplotlib.legend.Legend at 0x1f4635967b0>
```



```
import matplotlib.pyplot as plt
sub=["ada","dbms","aj","bio","gt"]
bhoomi=[99,88,98,96,89]
bhoo=[89,86,98,98,80]
plt.bar(sub,bhoomi,color='k',label='bhoomi
score',width=0.5,align="center")
plt.bar(sub,bhoo,color='green',label='bhoo
score',width=0.5,align="edge")
plt.xlabel('subject')
plt.ylabel('scores')
plt.legend()

<matplotlib.legend.Legend at 0x1f463a5e720>
```



```
import numpy as np
a=np.array([25,60,5,10])
labe=["ada","aj","bio","gt"]
color=['black','pink','coral','yellow']
plt.pie(a,labels=labe,colors=color)
plt.legend()
plt.show()
```


ada
aj
bio
gt
ada
aj
gt
bio