Expt. No. 7 Page No. 13

Expt. Name. Compute : $S = (f(x_1) + f(x_2) + \cdots + f(x_m))$ Modulo 2 Date :

Aim:

To write a python program by considering a function $f(x) = x^3$. Input is 'N' list. Each list contains 'M' elements. From the list, find the maximum element. Compute : $S = (f(x_1) + f(x_2) + \cdots + f(x_m))$ Modulo 2

Algorithm:

step 1: start
step 2: Get the input value of m & n from the user
step 3: initialize a empty list l and mx
step 4: Get the input & append it in mx
step 4: using double for loop for N and M
step 5: initialize a empty list l = []
step 6: Get the input value of z from the user
step 7: initialize s is equal to zero
step 8: for loop in mx
step 8.1: increment s by increasing the power of 3
step 9: print s/2
step 10: stop

Program:

```
def f(x):
    return x**3
N = int(input("Enter N: "))
M = int(input("Enter M: "))
l = []
mx = []
for i in range(N):
    for j in range(M):
```

Expt. No. _____

Page No. 14

Expt. Name. _____

Date : _____

```
l.append(int(input("Enter elements:")))
mx.append(max(l))
l=[]
z=int(input("Enter z"))
s=0
for i in mx:
    3
    st=f(i)
print(s/2)
```

Result:

The above program is executed successfully and the output is verified.

22/3/22

```
# Validate the credit numbers used on the following
import re
import itertools
text='5133-3367-8912-3456'
print(len(text))
l=[(k,sum(1 for i in g)) for k,g in itertools.groupby(text)]
if re.search(r'^[456]\d{3}-\d{4}-\d{4}-\d{4}$',text) and len(text)==19 and re.search(r'[4d]',text) and all(next for k,v in l) and bool(re.search(r'\s',text)) is False and bool(re.search(r'[a-z]',text))is False or(bool(re.search(r'-',text))is True and len(text)==19):
    print("It passed")
else:
    print("False")

19
It passed
```

Expt. No. 8 Page No. 15

Expt. Name. Validate the credit Numbers based on the following Date :

Aim:

To write a python program by validating the credit numbers based on the following condition: Begin with 4, 5 or 6, contain exactly 16 digits, contains only number (0 to 9). For every 4 digits a hyphen (-) may be included (not mandatory). No other special character permitted. Must not have 4 or more executive same digits.

Algorithm:

step 1: start
step 2: import the library re and itertools
step 3: initialize the text
step 4: print the length of the text
step 5: initialize the d to k, sum (i for i in g) for k, g in itertools.
groupby(text)
step 6: check the condition, if yes print "passed" else "False"
step 7: stop

Program:

```
import re
import itertools
text = "5133-3387-8912-3456"
print(len(text))
d = [(k, sum(1 for i in g)) for k,g in itertools.groupby(text)]
if re.search(r"^[456]", text) and len(text)==16 and re.search
(r'(\d)', text) and all [v<=3 for k,v in d) and bool(re.search
(r"\s", text)) is False and bool (re.search (r'[a-z]', text)) is
False or (bool(re.search(r'-', text)) is True and len(text)==19):
    print("it passed")
```

```
else:
    print("False")
```

Result:

The above program is executed successfully and the output is verified

Jeya 22/3/22

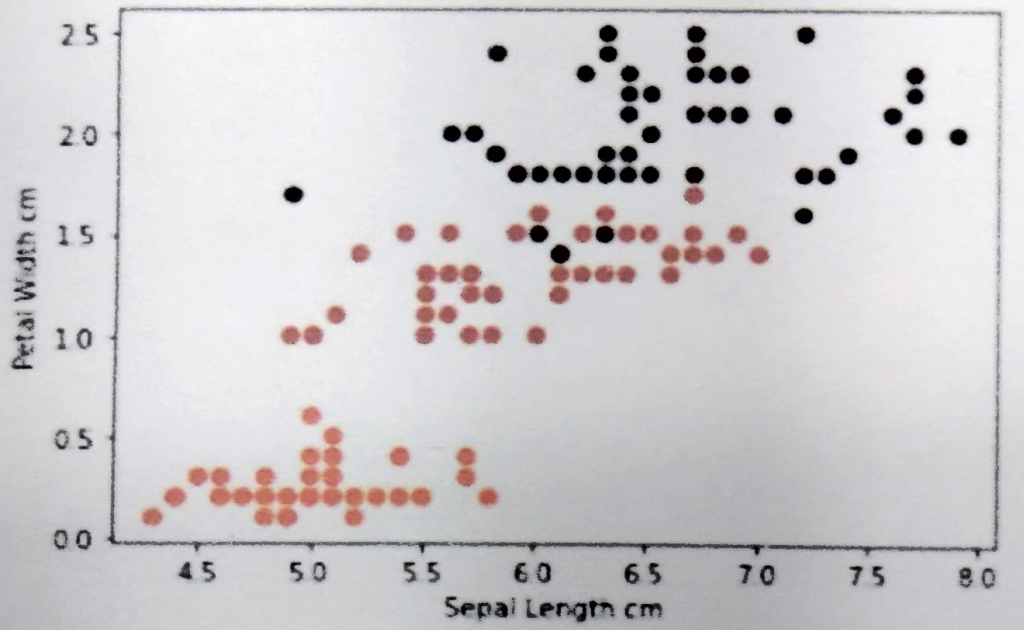
```
[ ]  x_test_std = sc.transform(x_test)
     x_test[0:5]

     array([[6.1, 2.8, 4.7, 1.2],
            [5.7, 3.8, 1.7, 0.3],
            [7.7, 2.6, 6.9, 2.3],
            [6. , 2.9, 4.5, 1.5],
            [6.8, 2.8, 4.8, 1.4]])

[ ]  x_test_std[0:5]

     array([[ 0.3100623 , -0.50256349,  0.484213  , -0.05282593],
            [-0.17225683,  1.89603497, -1.26695916, -1.27039917],
            [ 2.23933883, -0.98228318,  1.76840592,  1.43531914],
            [ 0.18948252, -0.26270364,  0.36746819,  0.35303182],
            [ 1.15412078, -0.50256349,  0.54258541,  0.2177459 ]])

     #x = iris.iloc[:,:-1].values
     #y = iris.iloc[:,4].values
     plt.scatter(x[:,0], x[:,3], c = y, cmap='flag')
     plt.xlabel('Sepal Length cm')
     plt.ylabel('Petal Width cm')
     plt.show()
```



Expt. No. 1 CYCLE-II Page No. 17

Expt. Name. Data Preprocessing: Building Good Training Sets Date :

Aim:

To write a python program for Data Preprocessing, Building Good Training set.

Algorithm:

step 1: start
step 2: import the necessary libraries
step 3: import the iris file from sklearn
step 4: initialize iris.data to x and iris.target to y
step 5: Split the data into train and test, test to 30%.
step 6: Using standard scaler, fit the x train
step 7: x train std to standard scaler transform x test
step 8: print the x test first five columns
step 9: print the x test std first five columns
step 10: plot the scatter plot of x-axis as sepal length and y-axis on petal width
step 11: stop

Program:

```
impot pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import standard scaler
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
iris = load_iris()
x = iris.data
y = iris.target
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3,
random_state = 42)
sc = standard scaler()
sc.fit(x_train)
x_train_std = sc.transform(x_test)
print(x_test_std[0:5])
print(x_test[0:5])
plt.scatter(x[:,0], x[:,3], c=y, cmap='flag')
plt.xlabel("sepal length cm")
plt.ylabel("Petal width cm")
plt.show()
```

Result:

The above program of Data pre-processing is successfully completed.

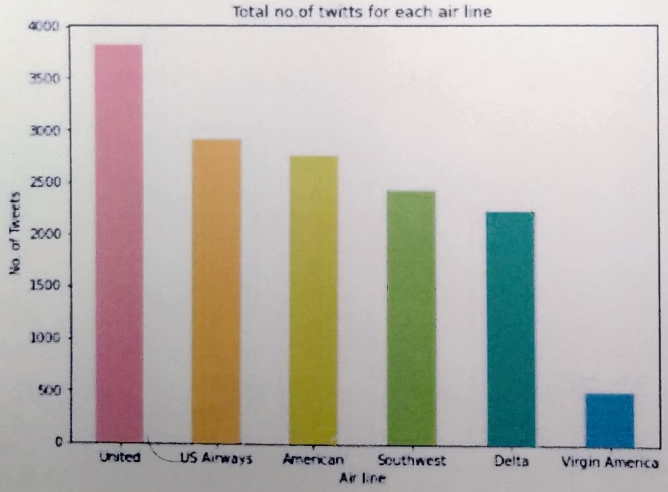
22/3/22

[5]
```
tweet.shape
tweet.airline.value_counts()
```

```
United            3822
US Airways        2913
American          2759
Southwest         2420
Delta             2222
Virgin America     504
Name: airline, dtype: int64
```

```
pd.Series(tweet["airline"].value_counts().plot(kind="bar", color=colors, figsize=(8,6), fontsize=10, rot=0, title="Total no.of twitts for each air line"))
plt.xlabel("Air line", fontsize=10)
plt.ylabel("No. of Tweets", fontsize=10)
tweet.airline_sentiment.value_counts()
colors = sns.color_palette("husl",10)
pd.Series(colors)
```

```
0    (0.9677975592919913, 0.44127456009157356, 0.53...
1    (0.8616090647292522, 0.536495730113334, 0.1954...
2    (0.6804189127793346, 0.6151497514677574, 0.194...
3    (0.46810256823426105, 0.6699492535792404, 0.19...
4    (0.20125317221201128, 0.6907920815379025, 0.47...
5    (0.21044753832183283, 0.6773105080456748, 0.64...
6    (0.2197995660828324, 0.6625157876850336, 0.773...
7    (0.433280341176423, 0.6065273407962815, 0.9585...
8    (0.8004936186423958, 0.47703363533737203, 0.95...
9    (0.962272393509669, 0.3976451968965351, 0.8008...
dtype: object
```



Expt. No. 2 Page No. 19

Expt. Name. Manipulate the twitter Dataset Date :

Aim:

To write a python program manipulate the twitter Dataset.

Algorithm:

step1: start
step2: import necessary libraries
step3: import the twitter dataset
step4: initialize the twitter dataset
step5: print the twitter dataset to tweet
step6: print the bar graph using airline value counts with x-axis as airline and y-axis as No. of tweets
step7: stop

Program:

```
import sklearn
import matplotlib.pyplot as plt
% matplotlib inline
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
import seaborn as sns
from nltk.corpus import stopwords
from wordcloud import wordcloud, STOPWORDS
from google.colab import files
uploaded = files.upload()
tweet = pd.read_csv("Tweets.csv")
tweet.tail()
tweet.shape()
```

Expt. No. _____ Page No. 20

Expt. Name. _____ Date : _____

```
tweet.airline.value_counts()

pd.Series(tweet['airline'].value_counts().plot(kind='bar', color = colors,
figsize=(8,6), fontsize=10, rot=0, title="Total no. of tweets for each airline")
plt.xlabel("Airline", fontsize=10)
plt.ylabel("No. of tweets", fontsize=10)
tweet.airline.sentiment.value_counts()
colors = sns.color_palette('hsl', 10)
pd.Series(colors)
```

Result :

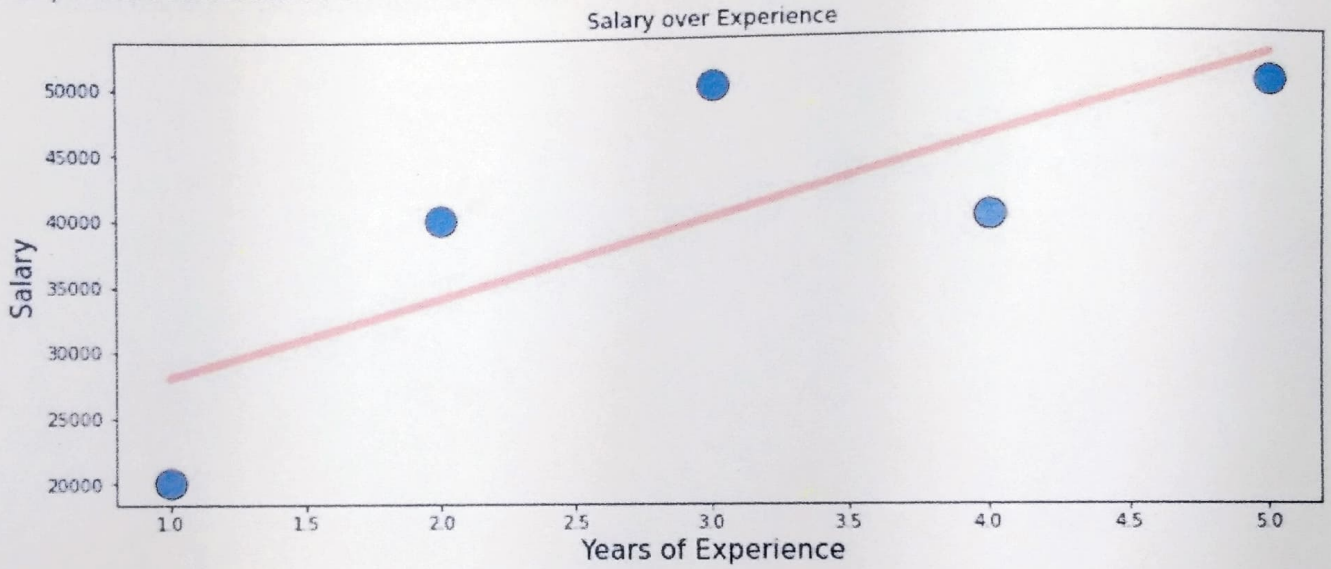
The above program of manipulating twitter data is succesfully executed.

22/3/22

+ Code + Text

```
plt.text(x=1, y=10, s=text, fontsize=12, bbox={'facecolor': 'grey', 'alpha':0.2, 'pad':10})
plt.title("Salary over Experience")
plt.xlabel('Years of Experience', fontsize=15)
plt.ylabel('Salary', fontsize=15)
plt.plot(x,B0 + B1*x, c='r',linewidth=5, alpha=.5, solid_capstyle='round')
plt.scatter(x=x.mean(), y=y.mean(), marker ='', s=10*2.5, c='r')
```

```
<matplotlib.collections.PathCollection at 0x7fee67aa7210>
```



X : 3.0 Years
Y : $40000.0
y = 22000.0 + 6000.0X

Expt. No. 3 a Page No. 21

Expt. Name. Implementing Regression and Correlation Technique Date :

Aim:

To write a python program Implementing Regression and Correlation Technique.

Algorithm:

step 1: start
step 2: import the necessary libraries
step 3: Using dictionary input the job title
step 4: Calculate the frame of the dictionary
step 5: define the class linear regression and find the x mean and y mean
step 6: define the class predict and find the prediction, reg-line, B0, B1
step 7: Using the library matplotlib, plot the graph with attributes x, B0 + B1 * x.
step 8: Stop

Program:

```
import pandas as pd
import numpy as np
dict = {"Experience": [1,2,3,4,5], "Salary": [20000, 40000, 50000,
        40000, 50000"]}
df = pd.DataFrame(dict)
df
x = df.iloc[:, 0].values
y = df.iloc[:, 1].values
```

```
def linear Regression (x, y):
    N = len(x)
    x_mean = x.mean()
    y_mean = y.mean()
    B1_num = ((x - x_mean)*(y-y_mean)).sum()
    B1_den = ((x -x_mean)**2).sum()
    B1 = B1_num / B1_den
    B0 = y_mean - (B1* x_mean)
    reg_line = 'y = {} + {} β'.format(B0, round(B1, 3))
    return (B0, B1, reg_line)

def Predict (B0, B1, new_x):
    y = B0 + B1 * new_x
    return y
    pred = redict (B0, B1, 8)
    Pred
B0, B1, regline = linear Regression (x,y)

import matplotlib.pyplot as plt
plt.figure (figsize = (12, 5))
plt.scatter (x, y, s = 300, linewidth =1, edgecolor = 'black')
test = "'x: {} Years Y: ${}  y = {} + {} x'''.format (round(x.mean(),2),
    round(y.mean(), 2), round (B0, 3), round (B1, 3))
plt.text (x=1, y=10, s= text, fontsize =12, bbox = {'facecolor': 'grey', 'alpha':
 0.2, 'pad': 10})
plt.title ("Salary over Experience")
plt.xlabel ("Years of Expence", fontsize = 15)
plt.ylabel ("Salary", fontsize = 15)
plt.plot (x, B0 + B1 * x, c='r', linewidth = 5, alpha =.5, solid_capstyle = "round")
```

Expt. No. _____ Page No. 23

Expt. Name. _____ Date : _____

```
plt.scatter(x=x.mean(), y=y.mean(), marker='*', s=s-10*2.5, c='r')
```

Result:

The above program of evaluating the results of machine learning algorithm is executed successfully.

22/3/22

```
['I1', 'I2', 'I3', 'I4', 'I5']
C1:
['I1']: 6
['I2']: 7
['I3']: 6
['I4']: 2
['I5']: 2

L1:
['I1']: 6
['I2']: 7
['I3']: 6
['I4']: 2
['I5']: 2

C2:
['I4', 'I3']: 0
['I5', 'I1']: 2
['I1', 'I3']: 4
['I2', 'I4']: 2
['I2', 'I1']: 4
['I5', 'I4']: 0
['I2', 'I3']: 4
['I2', 'I5']: 2
['I4', 'I1']: 1
['I5', 'I3']: 1

L2:
['I5', 'I1']: 2
['I1', 'I3']: 4
['I2', 'I4']: 2
['I2', 'I1']: 4
['I2', 'I3']: 4
['I2', 'I5']: 2

C3:
['I2', 'I5', 'I1']: 2
['I2', 'I1', 'I3']: 2
['I2', 'I4', 'I3']: 0
['I2', 'I5', 'I3']: 1
['I5', 'I1', 'I3']: 1
['I2', 'I4', 'I1']: 1
['I2', 'I5', 'I4']: 0

L3:
['I2', 'I5', 'I1']: 2
['I2', 'I1', 'I3']: 2

C4:
['I5', 'I1', 'I3', 'I2']: 1

L4:

Result:
L3:
['I2', 'I5', 'I1']: 2
['I2', 'I1', 'I3']: 2
```

Expt. No. 3b

Page No. 24

Expt. Name. Implementing Regression and Correlation Technique (Apriori)

Date :

Aim:

To write a python program Implementing Regression and Correlation Technique (Apriori).

Algorithm:

step 1: start
step 2: import necessary libraries
step 3: initialize the data by giving the values
step 4: sort the data
step 5: Calculate the len and initialize it to 5
step 6: print the C1 & L1 with respect to the item
step 7: Similarly, using loop combine the items and display C2 & L2
step 8: Continue, the loop until maximum items match
step 9: Calculate the support & Confidence
step 10: Display support & Confidence
step 11: stop

Program:

```
data = [
        ['T100', ['I1', 'I2', 'I5']],
        ['T200', ['I2', 'I4']],
        ['T300', ['I2', 'I3']],
        ['T400', ['I1', 'I2', 'I4']],
        ['T500', ['I1', 'I3']],
        ['T600', ['I2', 'I3']],
        ['T700', ['I1', 'I3']],
        ['T800', ['I1', 'I2', 'I3', 'I5']],
        ['T900', ['I1', 'I2', 'I3']] ]
```

```
['I2', 'I5'] -> ['I1'] = 100.0%
['I1'] -> ['I2', 'I5'] = 33.333333333333%
['I2', 'I1'] -> ['I5'] = 50.0%
['I5'] -> ['I2', 'I1'] = 100.0%
['I5', 'I1'] -> ['I2'] = 100.0%
['I2'] -> ['I5', 'I1'] = 28.571428571428%
choosing: 1 4 5

['I2', 'I1'] -> ['I3'] = 50.0%
['I3'] -> ['I2', 'I1'] = 33.333333333333%
['I2', 'I3'] -> ['I1'] = 50.0%
['I1'] -> ['I2', 'I3'] = 33.333333333333%
['I1', 'I3'] -> ['I2'] = 50.0%
['I2'] -> ['I1', 'I3'] = 28.571428571428%
choosing: 1 3 5
```

```
init = []
for i in data:
    for q in i[i]:
        if (q not in init):
            init.append(q)
init = sorted(init)
print(init)
sp = 0.4
s = int(sp * len(init))
s
from Collection import Counter
c = Counter()
for i in init:
    for d in data:
        if (i in d[i]):
            c[i] += 1
print("C1:")
for i in C:
    print(str[i]) + "! " + str(c[i]))
print()
l = Counter()
for i in C:
    if (c[i] >= s):
        l[frozenset(C[i])] += c[i]
print("L1:")
for i in l:
    print(str(list(i)) + ": " + str(l[i]))
print()
pl = l
```

```
pos = 1
for count in range(2, 1000):
    nc = set()
    temp = list(l)
    for i in range(0, len(temp)):
        for j in range(i+1, len(temp)):
            t = temp[i].union(temp[j])
            if (len(t) == count):
                nc.add(temp[i].union(temp[j]))
    nc = list(nc)
    c = Counter()
    for i in nc:
        c[i] = 0
        for q in data:
            temp = sd(q[i])
            if [i.issubset(temp)]:
                c[i] += 1
    print("C" + str(count) + ":")
    for i in c:
        print(str(list(i)) + ":" + str(c[i]))
print()
l = Counter()
for i in c:
    if (c[i] >= s):
        l[i] = c[i]
print("L" + str(count) + ":")
for i in l:
    print(str(list(i)) + ":" + str(l[i]))
print()
```

```
        if(len(l)==0):
            break
        pl=l
        pos=count
    print("Result:")
    print("L"+str(pos)+":")
    for i in pl:
        print(str(list(i))+":"+str(pl[i]))
    print()
    from itertools import combination
    for l in pl:
        c=[frozenset(q) for q in combination(l,len(l)-1)]
        mmax=0
        for a in c:
            b=l-a
            ab=l
            sab=0
            sa=0
            sb=0
            for q in data:
                temp=set(q[I])
                if(a.issubset(temp)):
                    sa+=1
                if(b.issubset(temp)):
                    sb+=1
                if(ab.issubset(temp)):
                    sab+=1
            temp=sab/sa*100
            if(temp>mmax)
                mmax=temp
```

```
        temp = sab/sb*100
        if (temp > mmax)
            mmax = temp
        print(str(list(a)) + " --> " + str(list(b)) + " = " + str(sab/sa*100)+'%')
        print(str(list(b)) + " --> " + str(list(a)) + " = " + str(sab/sa*100)+'%')
    curr=1
    print("choosing:",end=' ')
    for a in c:
        b = l-a
        ab=l
        sab=0
        sa=0
        sb=0
        for q in data:
            temp = set(q[1])
            if(a.issubset(temp)):
                sa+=1
            if(b.issubset(temp)):
                sb+=1
            if(ab.issubset(temp)):
                sab+=1
        temp = sab/sa*100
        if (temp == mmax)
            mmax=temp print(curr,end=' ')
        curr+=1
        temp= sab/sb*100
        if(temp==mmax):
            print(curr,end=" ")
        curr+=1
```

Expt. No. _____ Page No. 29

Expt. Name. _____ Date : _____

print()
print()

Result:

The above program to evaluating the results of machine Learning algorithm is executed successfully.

22/3/22

```
Confusion Matrix:  [6, 7, 5, 2]
Accuracy:  0.65
Precision:  0.5454545454545454
Recall:  0.75
Sensitivity:  0.75
Specificity:  0.5833333333333334
Missclassification Error:  0.35
```

Expt. No. 4 Page No. 30

Expt. Name. Evaluating the results of machine Learning algorithm Date :

Aim:

To Evaluate the results of machine Learning algorithm

Algorithm:

step 1: start
step 2: Input the file_Y into program
step 3: predict the Y
step 4: Initialize the TP, TN, FP, FN as '0'.
step 5: Using for loop Calculate the confusion_matrix
step 6: Calculate the Accuracy, precision, Recall Misclassification error and specificity
step 7: print the result ACC, PRE, REC, SN, SP & MCB.
step 8: stop

Program:

```
y= ['0','1','0','1','1','1','1','0','1','1','0','1','0','0','0','1','1','1','0',
    '1','1','0']
y_pred = ['0','0','0','0','1','0','1','1','1','1','0','0','0','0',
          '0','1','0','1','1','0']
j=0
TP,TN,FP,FN=0,0,0,0
for i in y:
    if (i=='1' and y_pred[j]=='1'):
        TP+=1
    elif (i=='0' and y_pred[j]=='0'):
        TN+=1
    elif(i=='1' and y_pred[j]=='0'):
        FP+=1
```