**PATTERNS AND TRENDS IN CAMPUS PLACEMENTS**

***INTRODUCTION:***

***1.1 Overview***

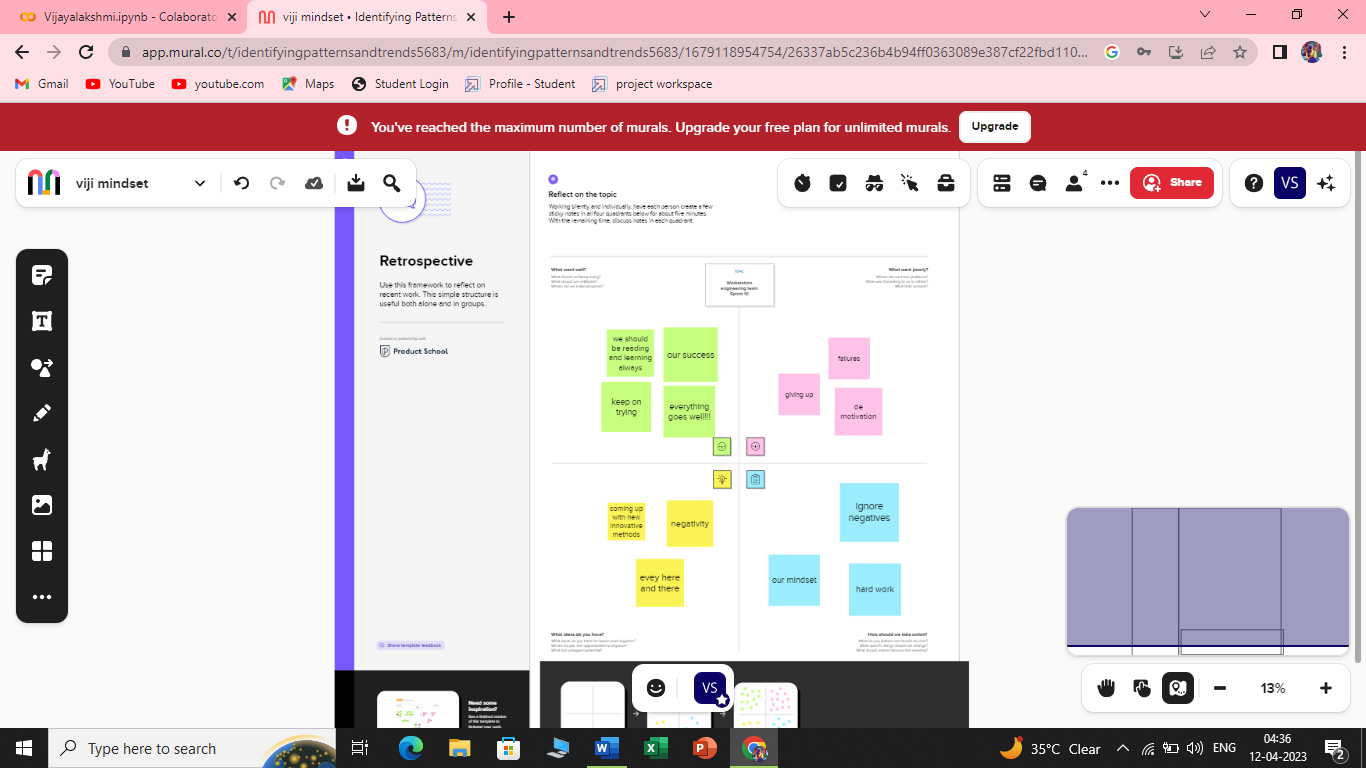
This dataset consist of Placement data of students in our campus. It includes age, gender, stream, internship and CGPA . Where it has various factors candidates getting hired such has work work experience, exam percentage etc..,

We will be using algorithms such as KNN, SVM and ANN. We will train and test the data with these algorithms. From these the best model is selected and saved in.pkl format. We will be doing flask integration and IBM deployment.

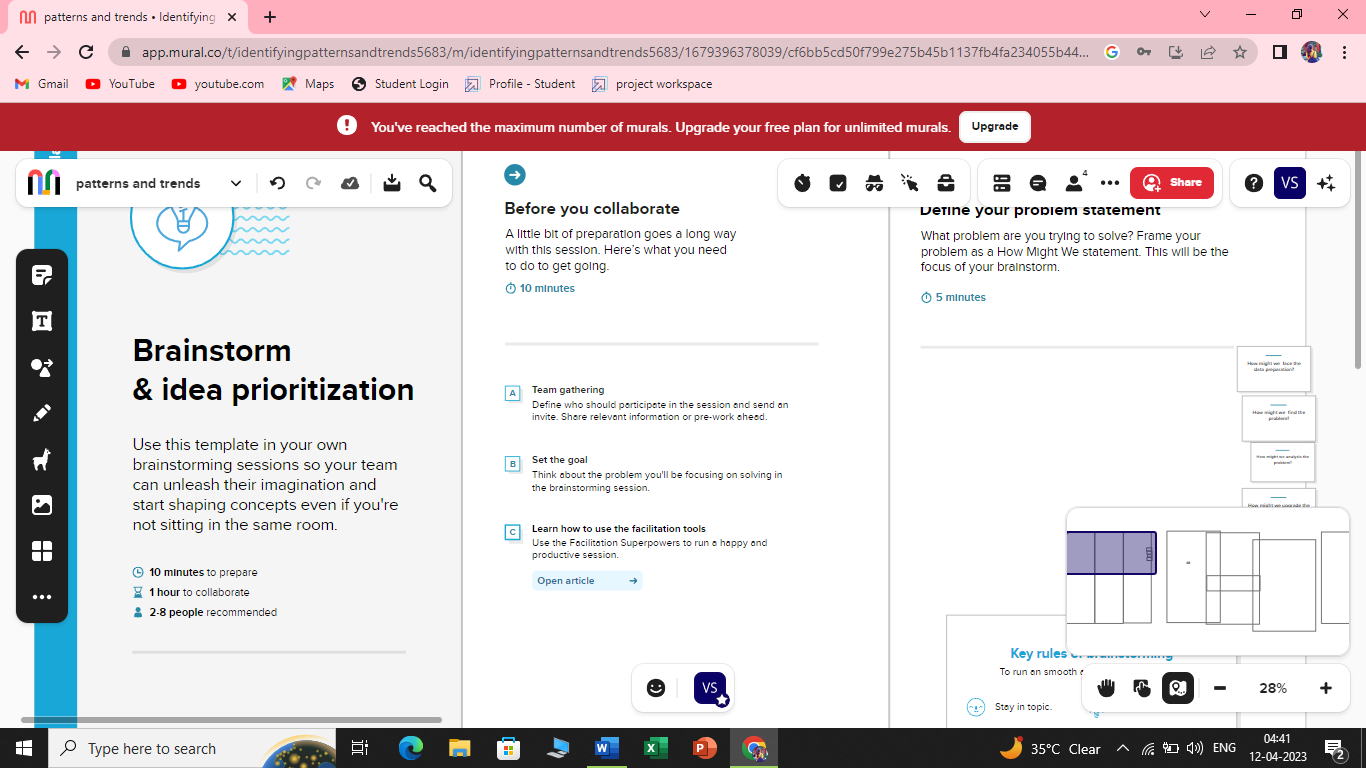
1.2 purpose

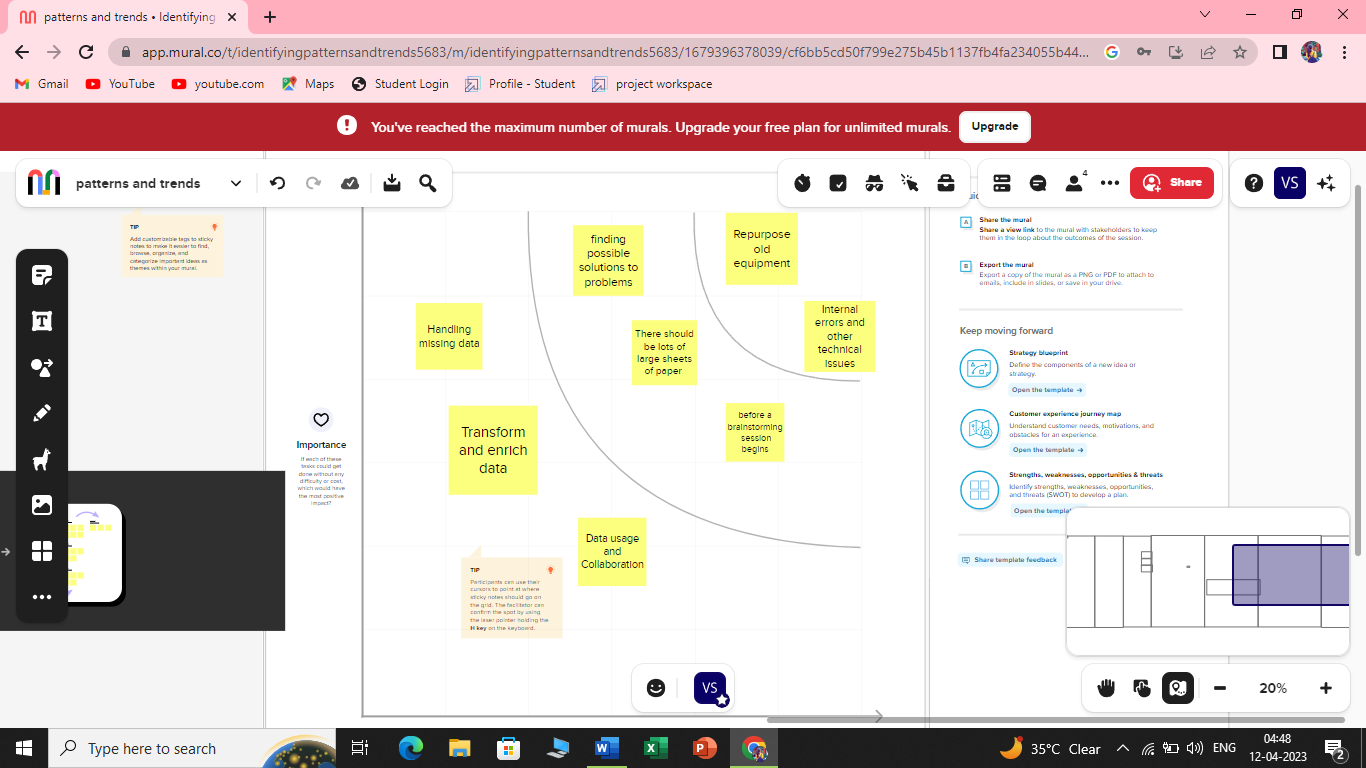
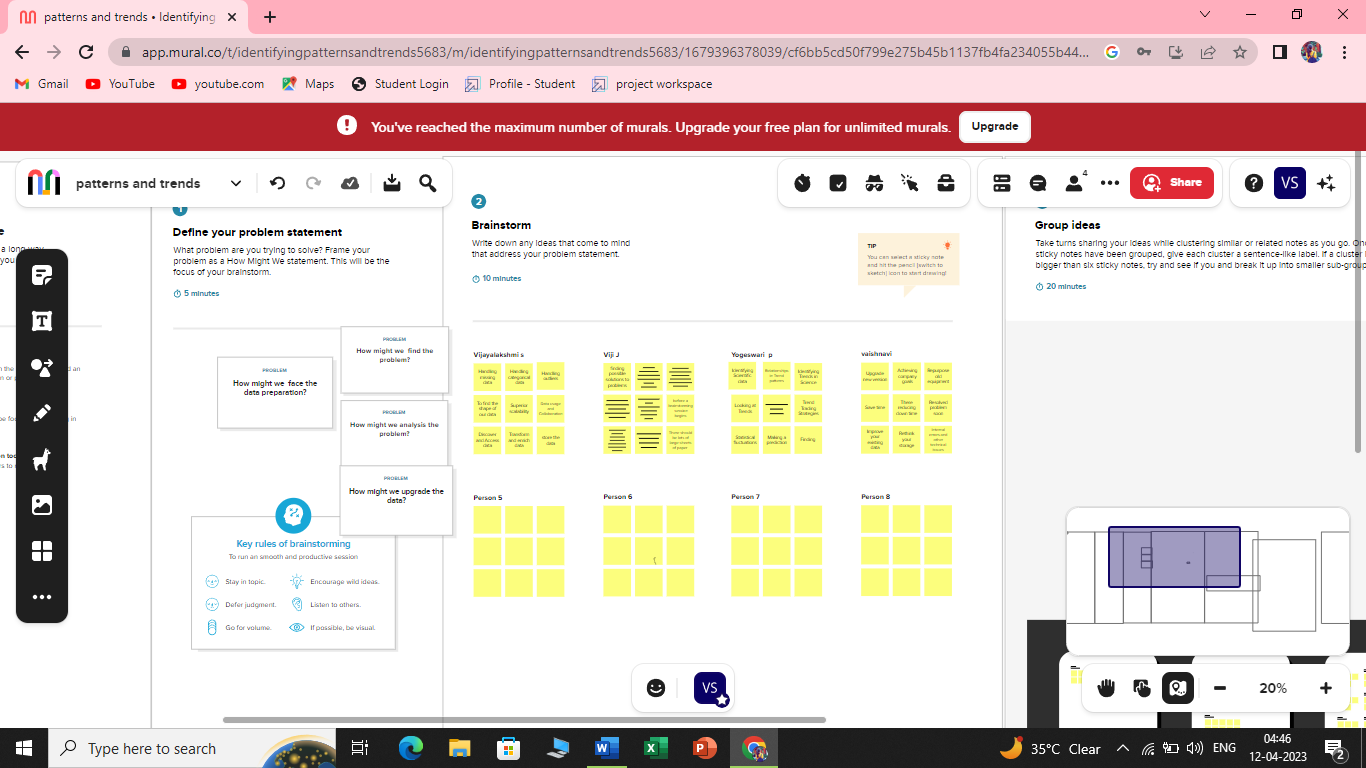
* In on-Campus placement drives, requirement companies are officially invited on the college campus to conduct interviews to gauge their potential as future employees.
* The placement process is centralized. Before the interviews they make a selection on criteria like student’s knowledge technical abilities ,and zeal to work.

**2. PROBLEM DEFINITION & DESIGN THINKING**

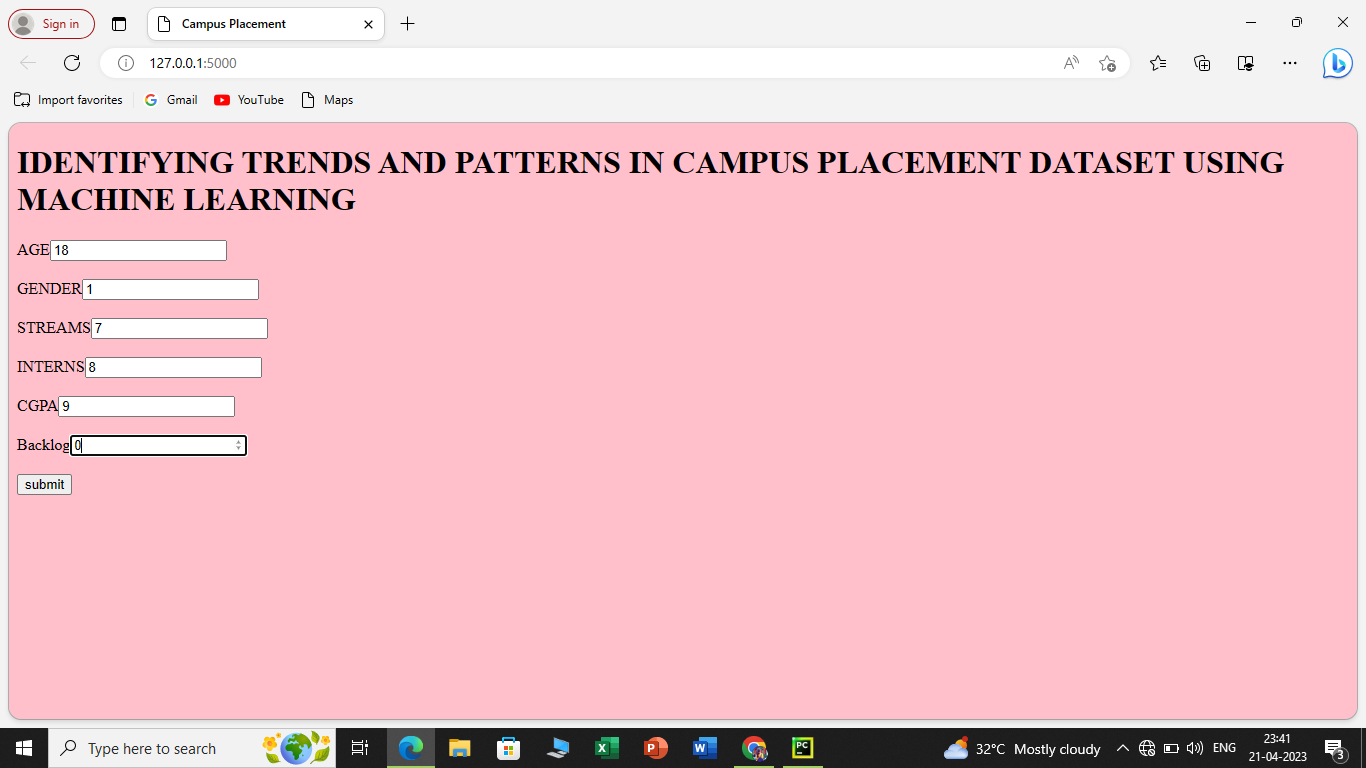
2.1.Empathy map:

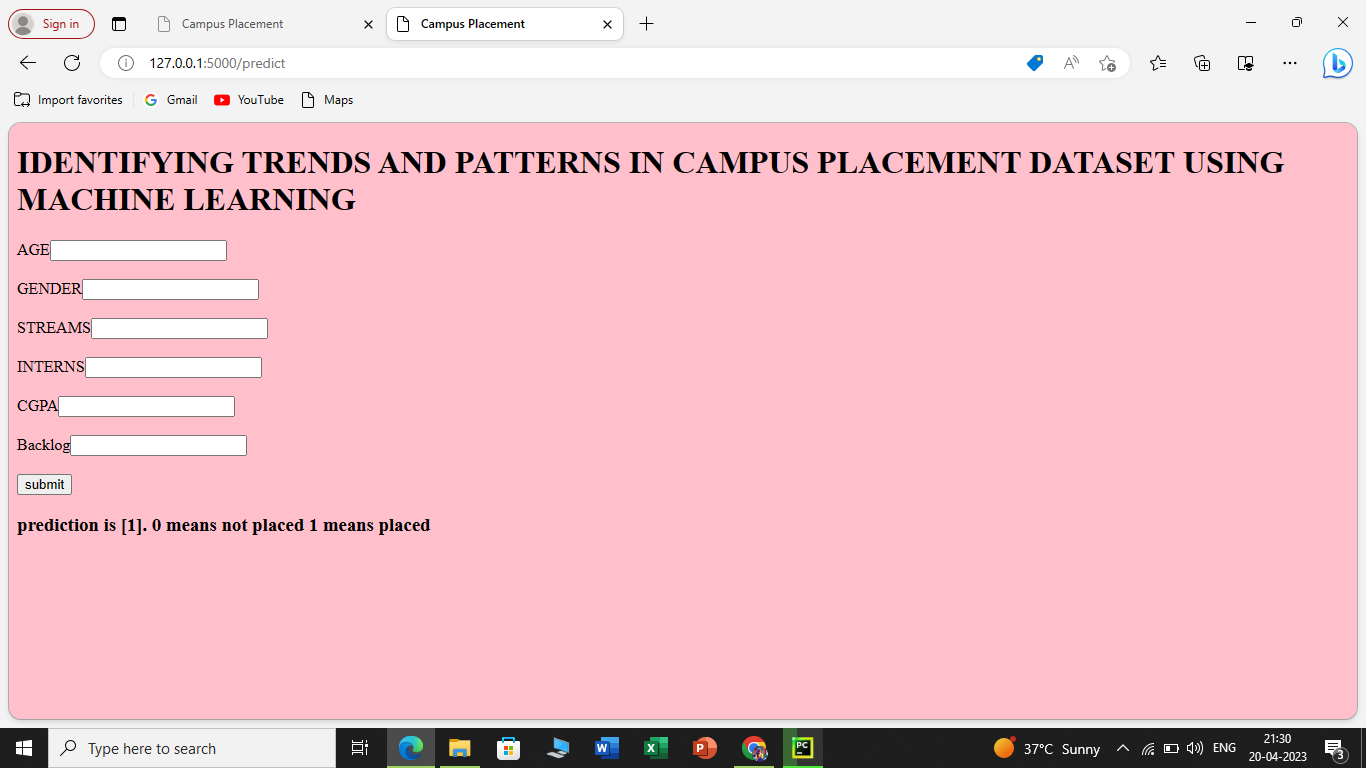
2.2.Ideation and brainstorming map:





**3.RESULT:**





**4. ADVANTAGES & DISADVANTAGES**

* Predict the result and we then compare the efficiency of the algorithms, which is based on the dataset.
* This model helps the position cell at intervals a corporation to spot the potential students and concentrate and improve their technical and social skills.
* They used normal conditional statements of getting information
* They don’t have proper algorithms for analysing for new placements.

**5.APPLICATIONs:**

* Trend Analysis
* Computer view

**6.CONCLUSION:**

* The algorithms of machine learning we have discussed are can used to find the trend of placement, which will be helpful for university to get more admission in future.
* We compared the algorithm and find out the accuracy by considering some of attributes of students.
* Here we used deep neural network classifier with the 1000,2000,5000 iteration with 71%,77% and 91% of accuracy.

**7.FUTURE SCOPE**

* A machine learning algorithms are play a very important role while predicting something, in future student performance will tell the institute and student weather he/she will clear the subject or not
* The students can evaluate themselves about their suitable job role.
* The students can analyze about their strengths and weakness.
* They can improve the weak points and get success for their goals.
* The screening of students by recruiters often take less time instead of spending lots of months to give a role to the students.

**APPENDIX**

1. Source code:
2. **APPENDIX**

from sklearn import metrics

import numpy as np

import pandas as pd

import os

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn import svm

import sklearn.metrics as metrics

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import cross\_val\_score

from sklearn import preprocessing

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

import joblib

from sklearn.metrics import accuracy\_score

import warnings

warnings.filterwarnings('ignore')

df = pd.read\_csv(r"/content/collegePlace.csv")

1. df.head()

df.shape

df.info()

df.isnull().sum()

sns.pairplot(df)

corr = df.corr()

ax=sns.heatmap(corr,vmin=-1,vmax=1,annot=True)

bottom,top=ax.get\_ylim()

ax.set\_ylim(bottom+0.5, top-0.5)

plt.show()

corr

def transformationplot(feature):

plt.figure(figsize=(12,5))

plt.subplot(1,2,1)

sns.distplot(feature)

transformationplot(np.log(df['Age']))

df = df.replace(['Male'],[0])

df = df.replace(['Female'],[1])

df = df.replace(['computer science'],[0])

df = df.replace(['Computer Science'],[0])

df = df.replace(['Information Technology'],[1])

df = df.replace(['Electronics And Communication'],[2])

df = df.replace(['Mechanical'],[3])

df = df.replace(['Electrical'],[4])

df = df.replace(['Civil'],[5])

df

plt.figure(figsize=(12,5))

plt.subplot(121)

sns.distplot(df['CGPA'],color='r')

plt.figure(figsize=(12,5))

plt.subplot(121)

sns.distplot(df['PlacedOrNot'],color='g')

df.info

1. plt.figure(figsize=(12,5))

plt.subplot(121)

sns.distplot(df['CGPA'],color='r')

1. from matplotlib.offsetbox import martist

plt.figure(figsize=(30,5))

plt.subplot(1,4,1)

sns.countplot(x="PlacedOrNot",data=df, ec='black')

plt.subplot(1,4,2)

sns.countplot(y="Stream",data=df, ec='black')

1. plt.show()

plt.figure(figsize=(20,5))

plt.subplot(131)

sns.countplot(x="PlacedOrNot", data=df, hue='CGPA' ec='black')

#sns.swarmplot(x='PlacedOrNot',y='CGPA', hue='Stream',data=df)

df.describe()

df.info()

df['Gender'].value\_counts()

df=df.drop(['Hostel'], axis=1)

x=df.drop('PlacedOrNot',axis=1)

y=df['PlacedOrNot']

import joblib

joblib.dump(x,"placement")

print(x)

print(y)

sc=StandardScaler()

print(sc)

sc.fit(x)

sd=sc.transform(x)

#x=sc.fit\_transform()

#x = pd.DataFrame(x)

print(sd)

x=sd

y=df['PlacedOrNot']

y

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size= 0.2, stratify=y, random\_state=2)

print(x.shape,x\_train.shape,x\_test.shape)

classifier =svm.SVC(kernel='linear')

classifier.fit(x\_train,y\_train)

svm.SVC(kernel='linear')

X\_train\_prediction=classifier.predict(x\_train)

training\_data\_accuracy=accuracy\_score(X\_train\_predictin,y\_train)

print('Accuracy score of the training data:',training\_data\_accuracy)

x\_train

best\_k={"Regular":0}

best\_score={"Regular":0}

for k in range(3, 50, 2):

knn\_temp=KNeighborsClassifier(n\_neighbors=k)

knn\_temp.fit(x\_train, y\_train)

knn\_temp\_pred=knn\_temp.predict(x\_test)

score=metrics.accuracy\_score(y\_test, knn\_temp\_pred)\*100

if score >=best\_score["Regular"]and score<100:

best\_score["Regular"]=score

best\_k["Regular"] = k



print("---Results---\nK: {}\n Score:{}".format(best\_k, best\_score))

knn = KNeighborsClassifier(n\_neighbors=best\_k["Regular"])

knn.fit(x\_train, y\_train)

knn\_pred = knn.predict(x\_test)

testd = accuracy\_score(knn\_pred, y\_test)

knn\_pred

import tensorflow as tf

from tensorflow import keras

from keras.models import Sequential

from tensorflow.keras import layers

classifier = Sequential()

#add input layer and first hidden layer

classifier.add(keras.layers.Dense(6,activation ='relu',input\_dim = 6))

classifier.add(keras.layers.Dropout(0.50))

#add second hidden layer

classifier.add(keras.layers.Dense(6,activation = 'relu'))

classifier.add(keras.layers.Dropout(0.50))

#final or output layer

classifier.add(keras.layers.Dense(1,activation = 'sigmoid'))

#compiling the moddel

loss\_1 = tf. keras.losses.BinaryCrossentropy()

classifier.compile(optimizer='Adam',loss=loss\_1,metrics=['accuracy'])

classifier.fit(x\_train, y\_train, batch\_size = 20, epochs = 100)

pred=classifier.predict(x\_test)

pred= (pred>0.5)

pred

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_test, pred)

cm

import pickle

pickle.dump(knn,open("placement.pkl",'wb'))

model = pickle.load(open('placement.pkl','rb'))

input\_data=[[22,0,2,1,8,1]]

prediction=knn.predict(input\_data)

print(prediction)

if(prediction[0]==0):

print('Not placed')

else:

print('placed')

input\_data=[[23,0,1,0,7,0]]

prediction=knn.predict(input\_data)

print(prediction)

if(prediction[0]==0):

print('Not placed')

else:

print('placed')