#### **IDENTIFYING PATTERNS AND TRENDS IN CAMPUSPLACEMENT DATA USING MACHINE LEARNING**

INDRODUCTION:

In recent years, there has been an increasing interest in using machine learning algorithms to analyze and identify patterns and trends in campus placement data. This type of data includes various factors such as academic performance, skill sets, internships, and recruiters' feedback, among others. By leveraging machine learning techniques, universities and colleges can gain insights into the hiring patterns of recruiters, the success of their placement programs, and the factors that are most likely to impact students' job prospects.

Some of the commonly used machine learning algorithms for analyzing campus placement data include decision trees, logistic regression, clustering, and neural networks. These algorithms can help universities predict which students are most likely to get placed, what factors can help them improve their job prospects, and which recruiters are most likely to hire from their campus.

In conclusion, machine learning can enable universities and colleges to make better decisions regarding their placement programs by identifying patterns and trends in campus placement data. With the help of such insights, institutions can tailor their programs to better meet the needs of their students and improve their overall success rates in terms of placing students into desirable jobs.

OverView:

Identifying patterns and trends in campus placement data using machine learning is a process of analyzing historical data to gain insights into the factors that influence successful placements. Machine learning algorithms can be trained to recognize patterns and relationships in large datasets, enabling recruiters to make informed decisions about how to best allocate their resources.

We will be using algorithms such as KNN, SVM and ANN. We will train and test

the data with these algorithms

The first step in this process is to collect data on campus placements, including information such as the number of candidates who applied, the number of candidates who were selected, the positions they were selected for, their educational background, and their performance in interviews. Once this data has been collected, machine learning algorithms can be used to identify patterns and trends in the data, such as which educational backgrounds are most successful in securing placements, which positions are most in demand, and which interview questions are most effective in identifying qualified candidates.

Purpose :

The purpose of identifying patterns and trends in campus placement data using machine learning is to gain insights into factors that affect the placement process and to improve the accuracy of predicting future placements. Campus placement data typically includes information such as student profiles, academic performance, skills, and placement outcomes. Machine learning algorithms can analyze this data to identify patterns and trends that can be used to make informed decisions about the placement process.

identifying patterns and trends in campus placement data using machine learning can help universities and recruiters make better decisions and improve the chances of successful placements for both students and employers.

Business problem

The business problem is to analyze campus placement data and identify patterns and trends using machine learning techniques. This will help in understanding the factors that affect campus placements and provide insights to improve placement rates. The data may include information about student profiles, job offers, companies, and other relevant variables. The ultimate goal is to increase the success rate of campus placements and improve the overall job market for students***.***

Business Requirement:

The business requirements for a project aimed at "Identifying Patterns and

Trends in Campus Placement Data using Machine Learning" would likely

include the following

❖ Access to campus placement data: The project would require

access to data on student performance, qualifications, and job

placement outcomes. This data would need to be collected,

cleaned, and prepared for analysis.

❖ Machine learning expertise: The project would require

individuals with expertise in machine learning, data science and

statistical analysis to develop and implement the algorithms and

models needed to analyze the data.

❖ Data storage and management: The project would require a

robust and secure data storage and management system to store

and organize the large amounts of data used in the analysis.

❖ Infrastructure for model deployment: The project would require

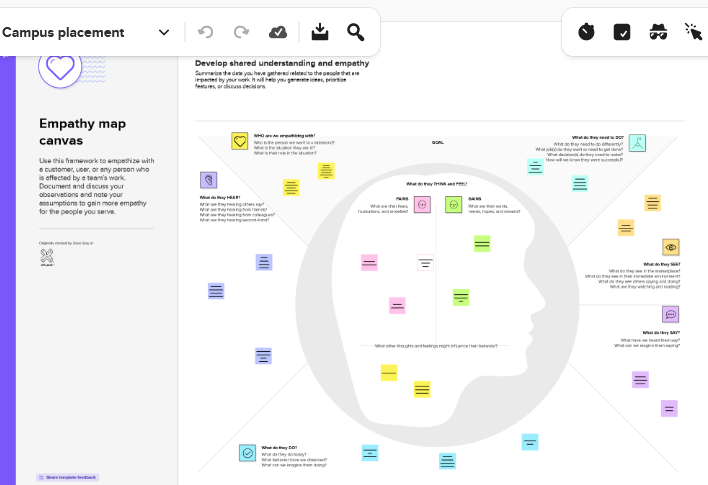
infrastructure for deploying the models and algorithms

developed, including hardware, software, and cloud-based

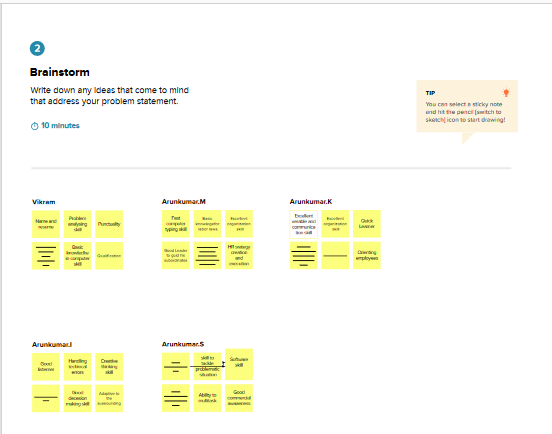
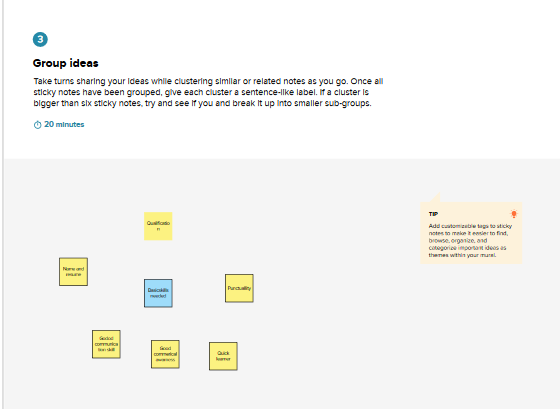
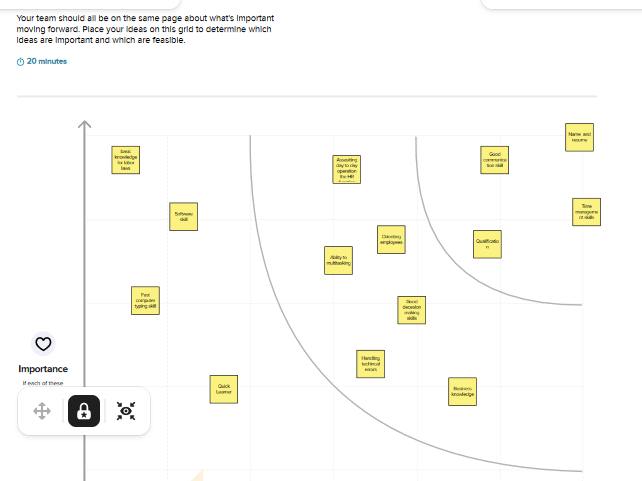
resources.

2) PROBLEM DEFINITION AND DESING THINKING

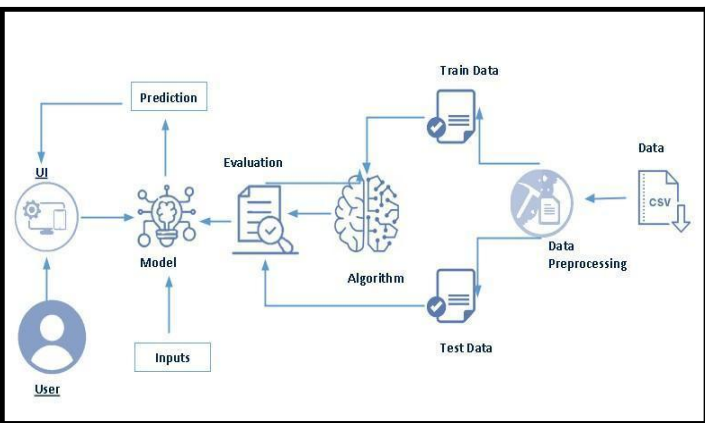
* Empathy Map:



* Ideation and Brainstorming Map:



Technical Architecture:



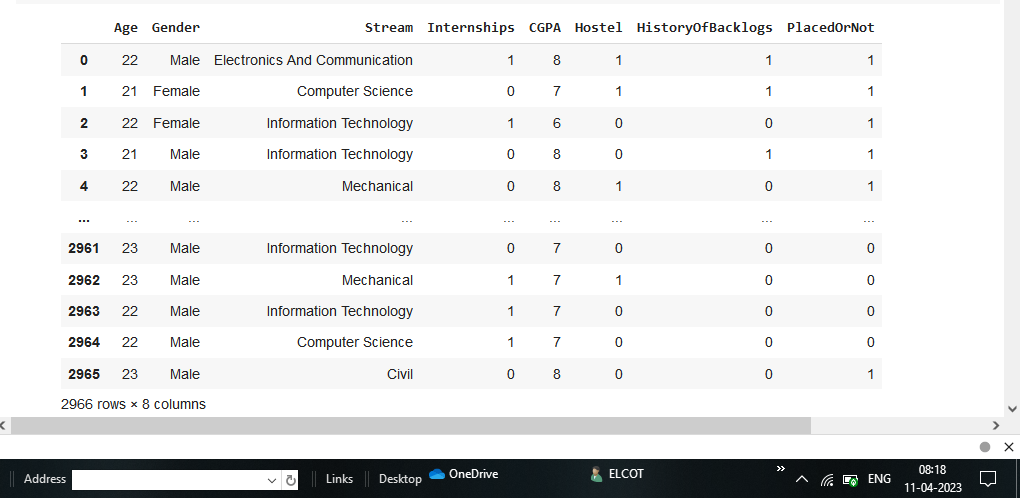
1. RESULT:

Result:1

Result 2:

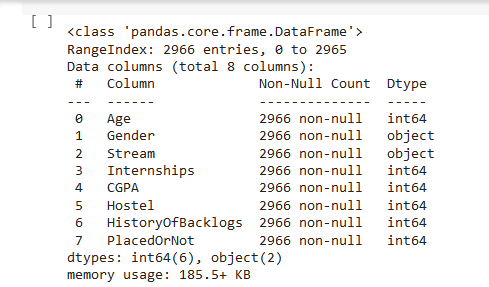
df = pd.read\_csv(r”/content/collegePlace.csv”)

Df.head()



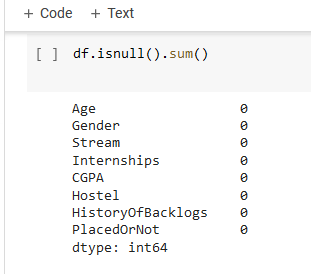
Resule 3:

Df.info()



Result 4:

Df.isnull().sum()



Result 5:

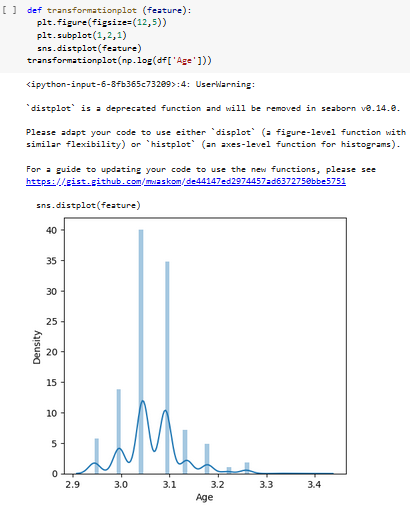
Def transformationplot(feature):

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

Plt.subplot(1,2,1)

Sns.displot(feature)

Transformationplot(np.log(df[‘Age’]))



Result 6:

df = df.replace([‘Male’],[0])

df = df.replace([‘Female’],[1])

Result 7:

df = df.replace([‘Computer science’.’Information Techonology’,’Electronices and

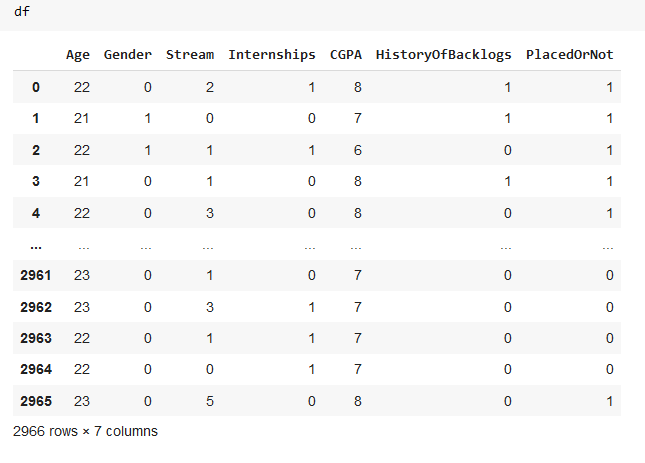
Communication’,’Mechinical’,’Electrical’,’Civil’],[0,1,2,3,4,5])

Result 8:

Df = df.drop([‘Hostal’], axis=1)

Result 9:

Df

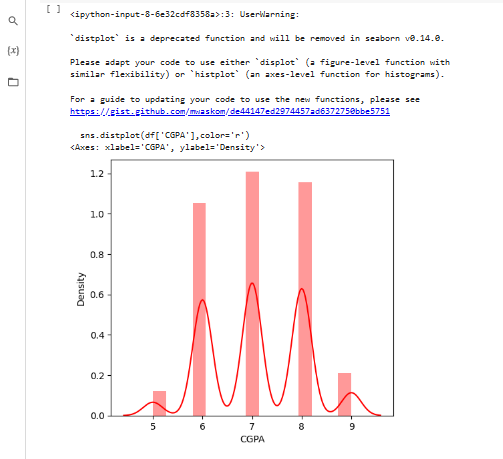


Result 10:

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

Plt.subplot(121)

Sns.displot(df[‘CGPA’],Color =’r’)

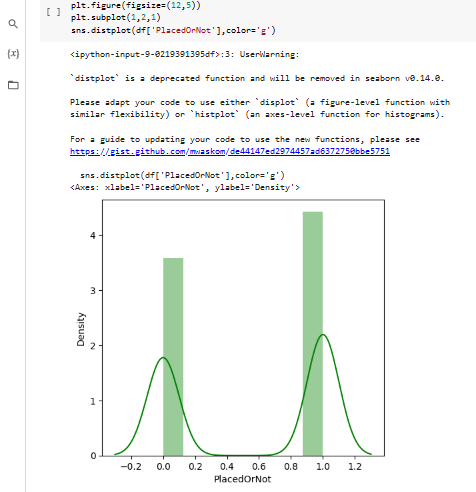


Result 11:

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

Plt.subplot(121)

Sns.displot(df[‘placedOrNot’],Color =’r’)



Result 12:

Plt.figure(figize=(18,4))

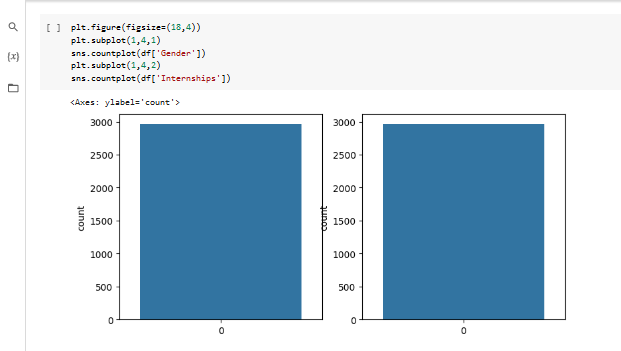
Plt.subplot(1,4,1)

Sns.ccountplot(data[‘Gender’])

Plt.subplot(1,4,2)

Sns.countplot(data[‘Eucation’])

Plt.show()

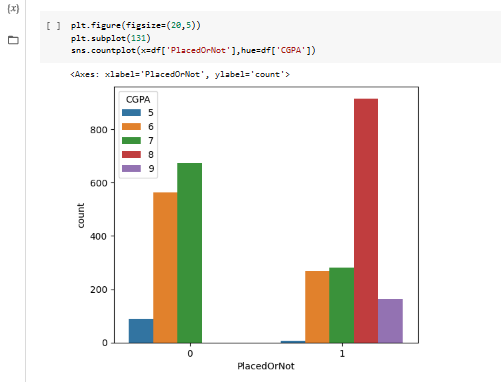


Result 13:

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

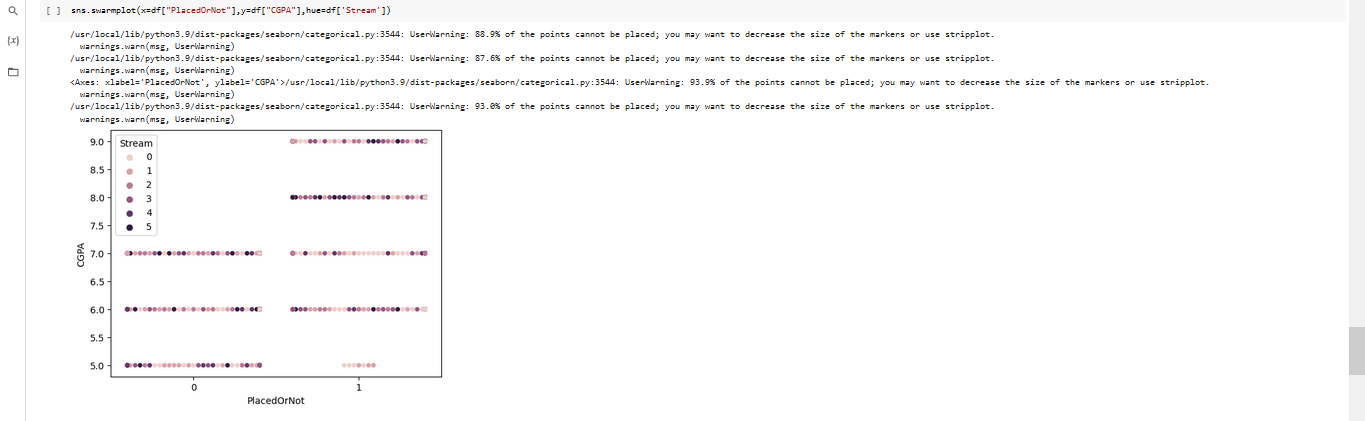
Plt.subplot(131)

Sns.countplot(df[‘PlacedOrNot’],hue=df[‘CGPA’] )



Result 14:

Sns.swarmplot(df[‘PlacedOrNot’],df[‘CGPA’],hue=df[‘team’])



Result 15:

Sc=StandaredScaler()

X\_bal=sc.fit\_transform(x\_bal)

Result 16:

X\_bal = pd.DataFrame(x\_bal,coloumns=nams)

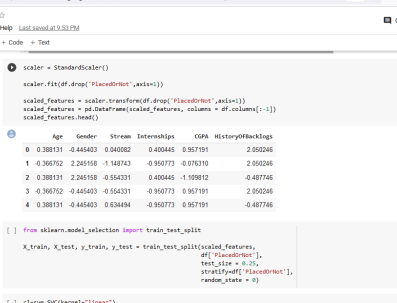
Result 17:

X=standardized\_data

Y=df[‘PlacedOrNot’]

Result 18:

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



Result 19:

Classifier = svm.svc(kernal=’linear)

Result 20:

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

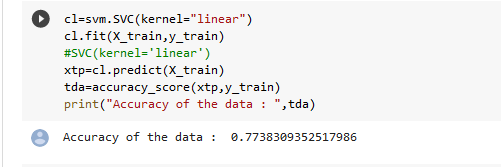
Result 21:

X\_train\_prediction = classifier.predictct(X\_train)

Training\_data\_accuracy = accuarcy\_score(X\_train\_prediction, y\_train)

Result 22:

Print(‘Accuracy score of the training data :’, training\_data\_accuracy)



Result 23:

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"] =100

best\_k["Regular"]=k

Result 24:

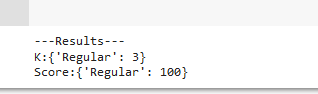
print("---Results---\nK:{}\nScore:{}".format(best\_k,best\_score))

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

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

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

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



Result 25:

import tensorflow as tf

from tensorflow import keras

from keras.models import Sequential

from tensorflow.keras import layers

Result 26:

classifier=Sequential()

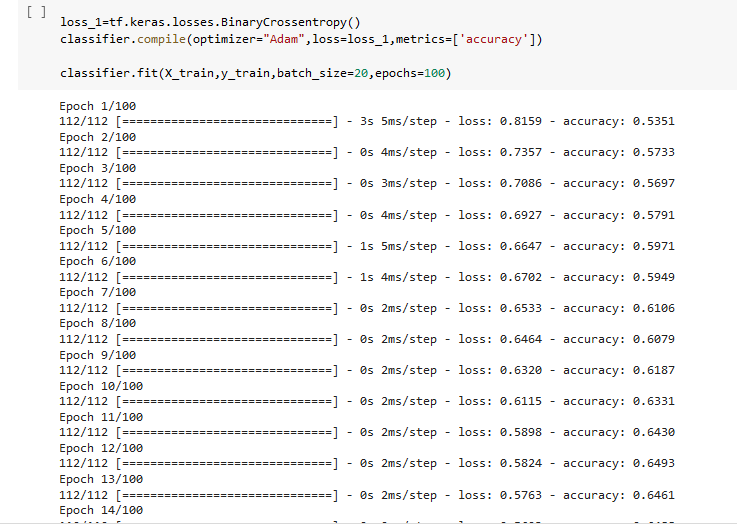
classifier.add(keras.layers.Dense(6,activation='relu'))

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

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

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

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



Result 27:

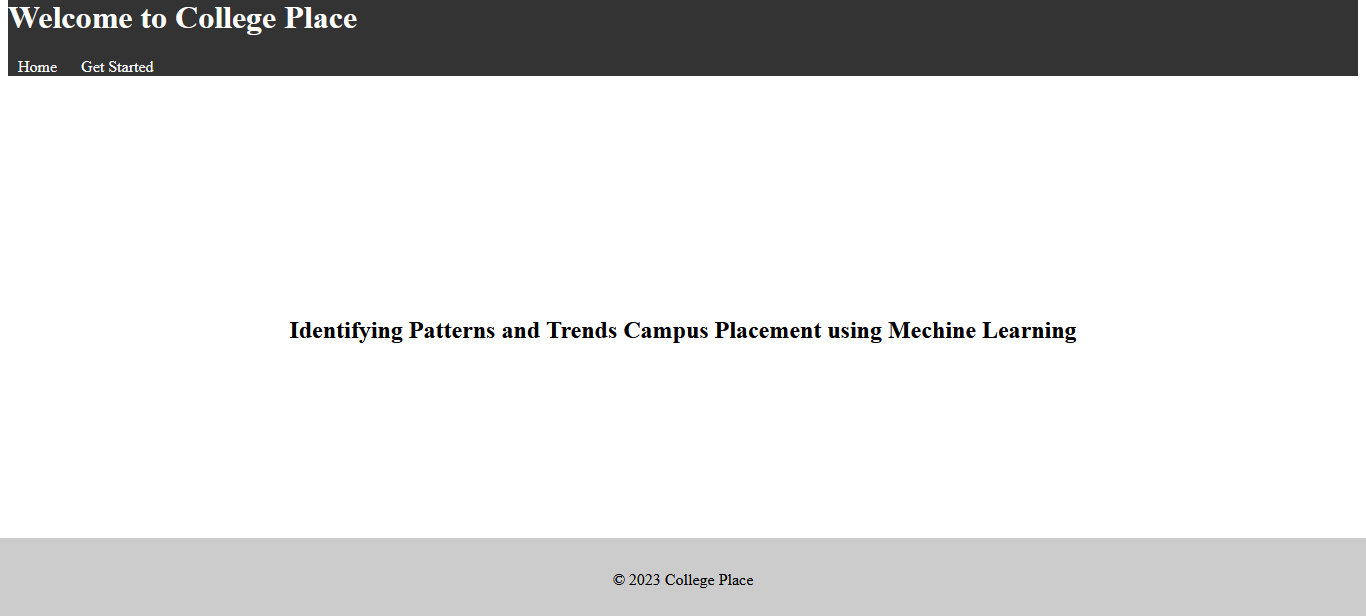
loss\_1=tf.keras.losses.BinaryCrossentropy()

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

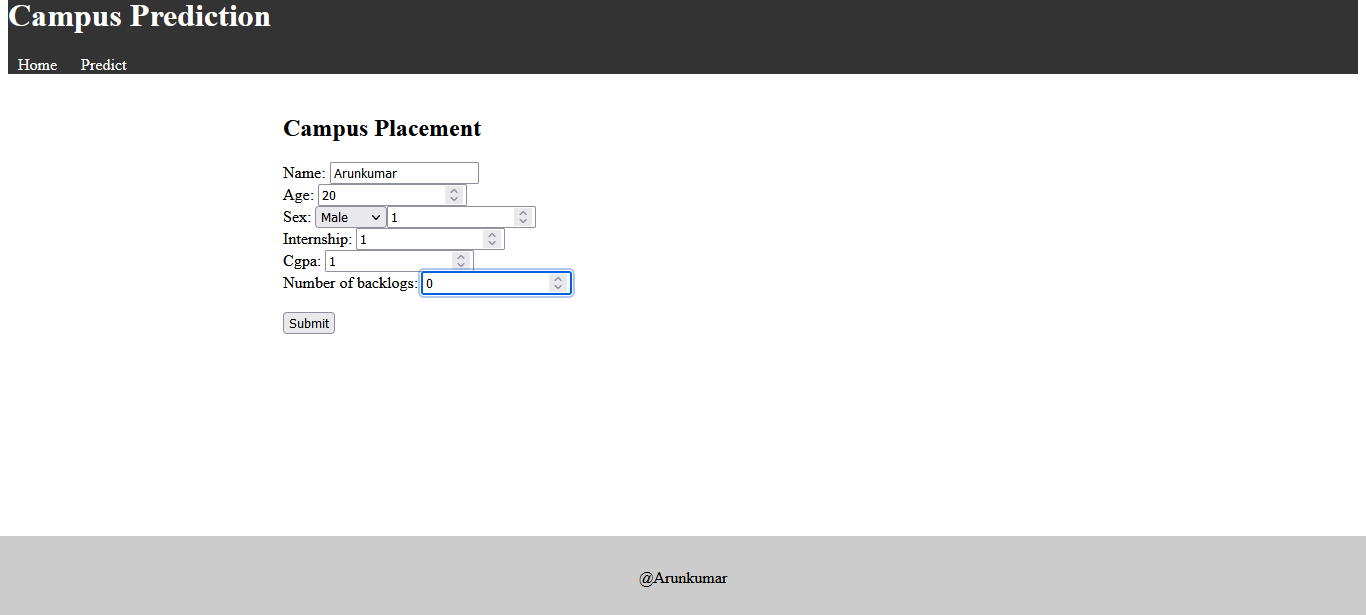
Result 28:

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

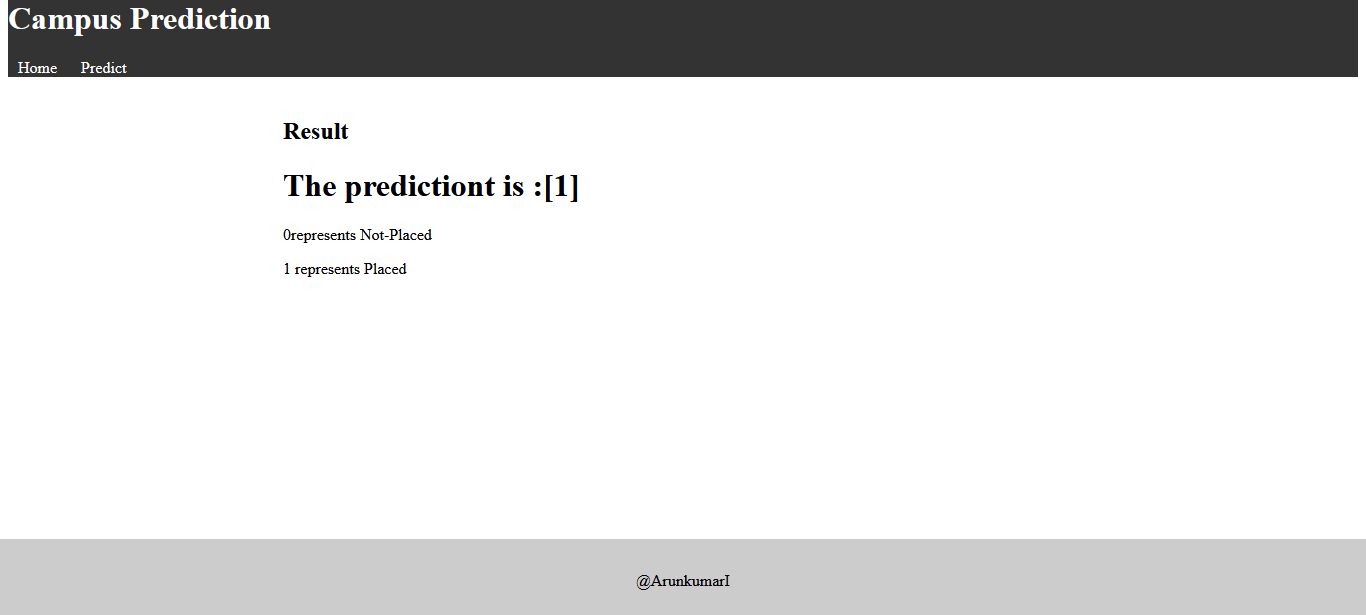
Result 29



Result 30



Result 31



4) ADVANTAGES AND DISADVANTAGE

Advantage :

Machine learning can be used to identify patterns and trends in campus placement data, which can help in making informed decisions about hiring processes. With machine learning algorithms, it is possible to analyze large amounts of data and identify correlations and patterns that may not be visible to the human eye. This can be useful for predicting which candidates are most likely to be successful in a particular role, identifying factors that contribute to successful placements, and optimizing recruitment processes to improve overall success rates. By leveraging machine learning in this way, organizations can gain valuable insights that can help them make data-driven decisions and improve their campus placement outcomes.

Disadvantage :

One potential disadvantage of using machine learning to identify patterns and trends in campus placement data is the risk of producing biased or inaccurate results. This can occur if the data used to train the model is not representative of the overall population or if the model is not designed to account for certain variables or factors that may impact placement outcomes. Additionally, relying solely on machine learning algorithms may overlook important contextual information that could be valuable in interpreting the results. It is important to approach machine learning as a tool to augment human decision-making, rather than a replacement for it, and to continually evaluate and refine the model to ensure its accuracy and fairness.

5) APPLICATIONS

Using machine learning algorithms to analyze campus placement data can help identify patterns and trends that may be difficult to identify with traditional statistical methods. By analyzing data on factors such as student qualifications, industry sectors, and job roles, machine learning algorithms can identify correlations and predict future trends. This information can be useful to educational institutions in adapting their curricula to meet the demands of the job market and to employers in making informed decisions about their hiring strategies. Additionally, machine learning can help reduce bias in the recruitment process by identifying and addressing potential sources of bias in job postings and candidate selection.

1. APPENDIX

import numpy as np

import pandas as pd

from  google.colab import files

import io

import os

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn import svm

from sklearn.metrics import accuracy\_score

from sklearn.neighbors import KNeighborsClassifier

from sklearn import metrics

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

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

df

df.info()

df.isnull().sum()

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','Information Technology','Electronics And Communication','Mechanical','Electrical','Civil'],[0,1,2,3,4,5])

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

Df

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

plt.subplot(1,2,1)

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

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

plt.subplot(1,2,1)

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

plt.figure(figsize=(18,4))

plt.subplot(1,4,1)

sns.countplot(df['Gender'])

plt.subplot(1,4,2)

sns.countplot(df['Internships'])

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

plt.subplot(131)

sns.countplot(x=df['PlacedOrNot'],hue=df['CGPA'])

sns.swarmplot(x=df["PlacedOrNot"],y=df["CGPA"],hue=df['Stream'])

scaler = StandardScaler()

scaler.fit(df.drop('PlacedOrNot',axis=1))

scaled\_features = scaler.transform(df.drop('PlacedOrNot',axis=1))

scaled\_features = pd.DataFrame(scaled\_features, columns = df.columns[:-1])

scaled\_features.head()

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(scaled\_features,

                                                    df['PlacedOrNot'],

                                                    test\_size = 0.25,

                                                    stratify=df['PlacedOrNot'],

                                                    random\_state = 0)

cl=svm.SVC(kernel="linear")

cl.fit(X\_train,y\_train)

#SVC(kernel='linear')

xtp=cl.predict(X\_train)

tda=accuracy\_score(xtp,y\_train)

print("Accuracy of the data : ",tda)

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"] =100

      best\_k["Regular"]=k

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

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

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

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

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

import tensorflow as tf

from tensorflow import keras

from keras.models import Sequential

from tensorflow.keras import layers

classifier=Sequential()

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

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

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

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

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

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)

import pickle

pickle.dump(knn,open("placement.pk1",'wb'))

model = pickle.load(open('placement.pk1','rb'))

1. FUTURE SCOPE

Campus placement prediction is an important field that can benefit both students and recruiters. With the help of machine learning algorithms, it is possible to analyze past data and make predictions about the future campus placements. Here are some potential areas of future scope in campus placement prediction:

Predictive models: Developing predictive models that can accurately predict campus placements based on various factors such as academic performance, skill sets, and job requirements.

Personalized recommendations: Providing personalized recommendations to students based on their strengths, weaknesses, and interests. This can help students identify the right job opportunities and prepare for interviews accordingly.

Real-time analytics: Using real-time analytics to track the progress of campus placements and provide insights to recruiters and students. This can help recruiters identify potential candidates and help students make informed decisions about their job search.

Natural language processing: Using natural language processing (NLP) to analyze job descriptions and candidate resumes. This can help recruiters identify the right candidates for a particular job role and help students tailor their resumes to match job requirements.

Automation: Automating various aspects of campus placement, such as resume screening and interview scheduling. This can help recruiters save time and focus on more important tasks such as candidate engagement.

Overall, there is a lot of potential for innovation and growth in the field of campus placement prediction. As the job market becomes more competitive, students and recruiters alike will benefit from the insights and recommendations provided by these technologies.

1. CONCLUSION

Identification of factors that affect campus placement: By analyzing the data, we can identify the factors that have a significant impact on the campus placement process. These factors may include academic performance, skills, work experience, and communication skills.

Prediction of placement probability: By using machine learning algorithms, we can predict the probability of a candidate getting placed in a company based on the given parameters. This prediction can help candidates and recruiters make informed decisions.

Recommendations for improvement: Based on the analysis, we can provide recommendations to candidates and colleges to improve their chances of getting placed. These recommendations may include suggestions to improve academic performance, acquire new skills, or enhance communication skills.