**IDENTIFYING PATTERNS AND TRENDS IN CAMPUS PLACEMENT DATA USING MACHINE LEARNING**

**PROJECT REPORT TEMPLATE**

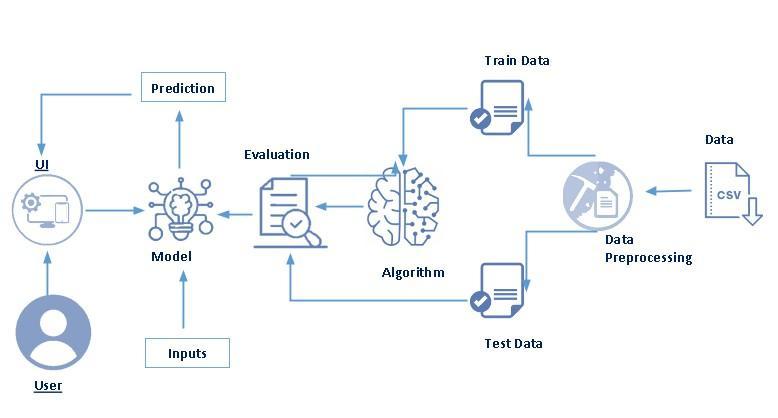
**1.INTRODUCTION:**

* 1. **OVERVIEW**

**Project Description:**

* Campus recruitment is a strategy for sourcing, engaging and hiring young talent for internship and entry-level positions. College recruiting is typically a tactic for medium- to large-sized companies with high-volume recruiting needs, but can range from small efforts (like working with university career centers to source potential candidates) to large-scale operations (like visiting a wide array of colleges and attending recruiting events throughout the spring and fall semester).Campus recruitment often involves working with university career services centers and attending career fairs to meet in-person with college students and recent graduates.Our solution revolves around the placement season of a Business School in India. Where it has various factors on candidates getting hired such as work experience,exam percentage etc., Finally it contains the status of recruitment and remuneration details.
* We will be using algorithms such as KNN, SVM and ANN. We will train and test the data with these algorithms. From this the best model is selected and saved in .pkl format. We will be doing flask integration and IBM deployment.

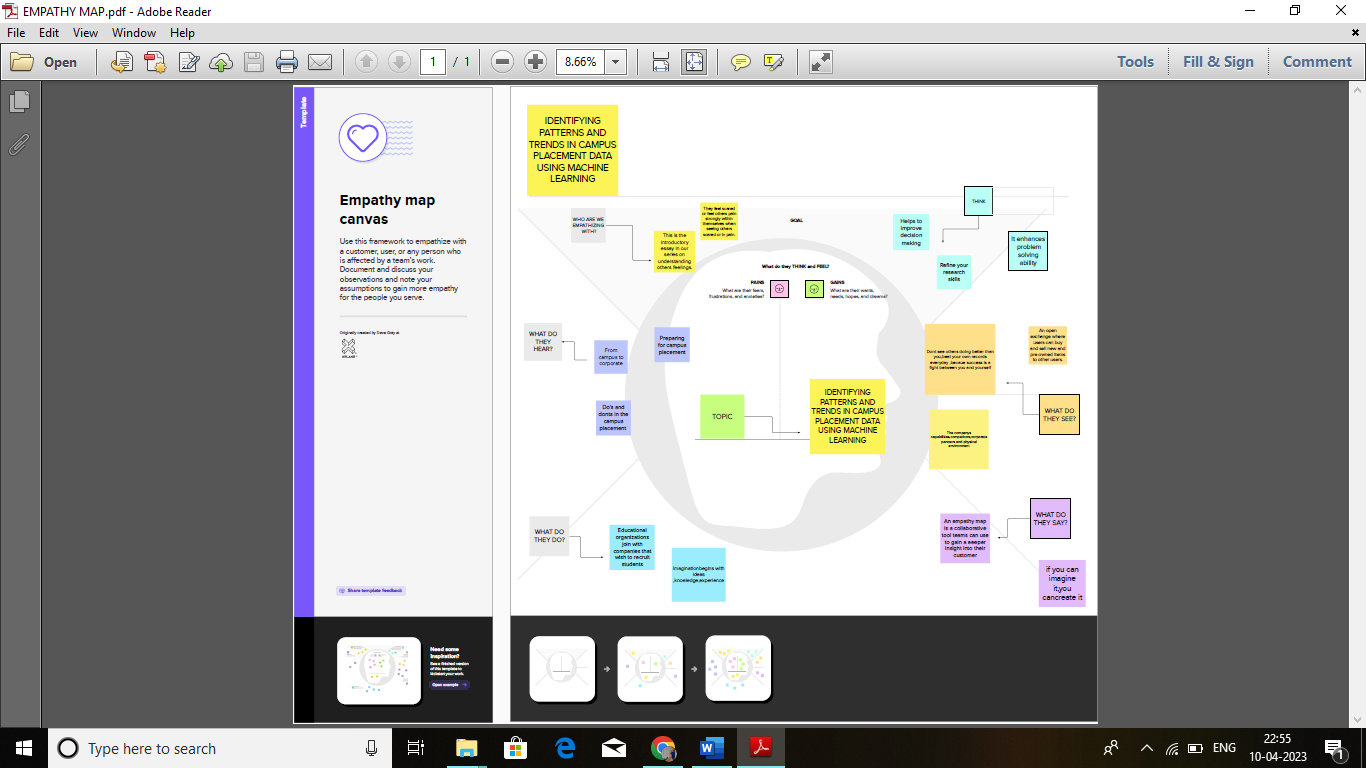
**Technical Architecture:**



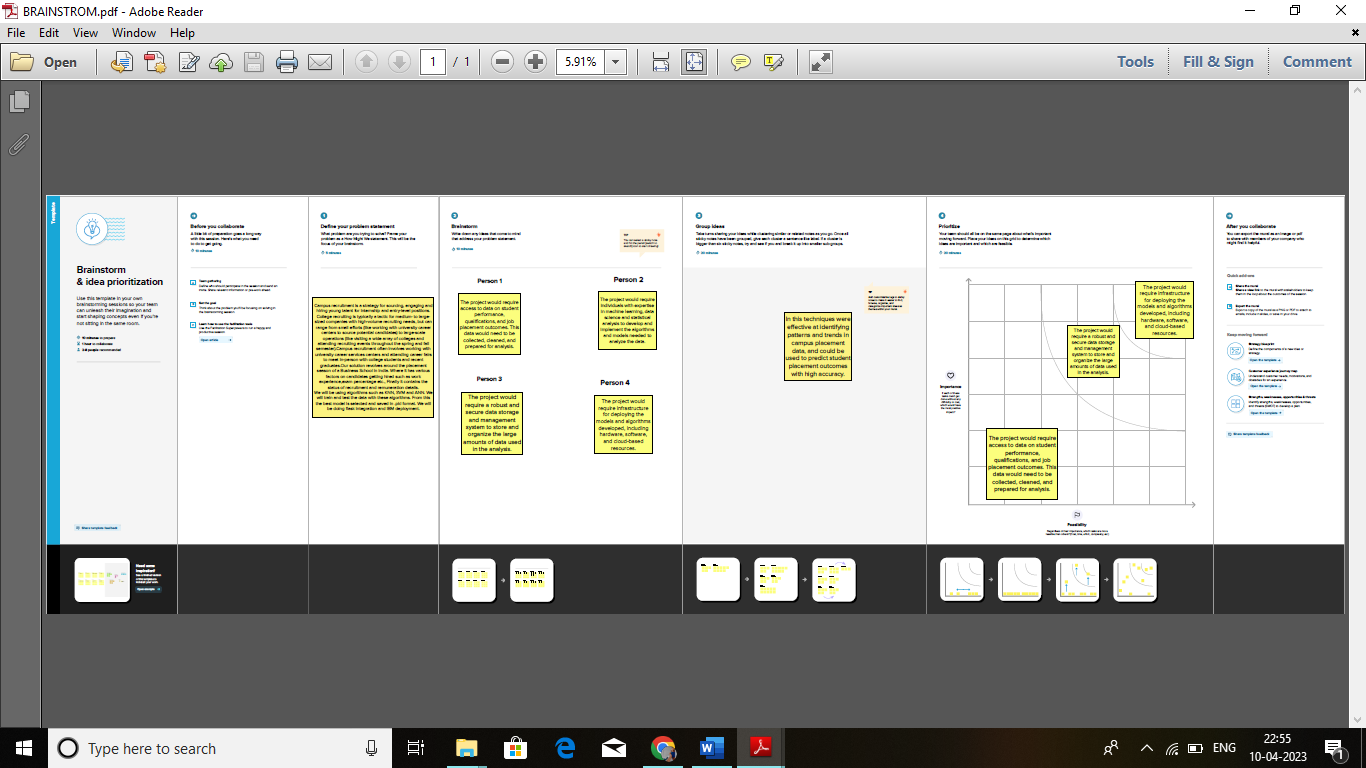
* 1. **PURPOSE**
* Campus placement or campus recruiting is a program conducted within universities or other educational institutions to provide jobs to students nearing completion of their studies. In this type of program, the educational institutions partner with corporations who wish to recruit from the student population.

**2. PROBLEM DEFINITION & DESIGN THINKING:**

**2.1 EMPATHY MAP**



**2.2 IDEATION & BRAINSTORMING MAP**

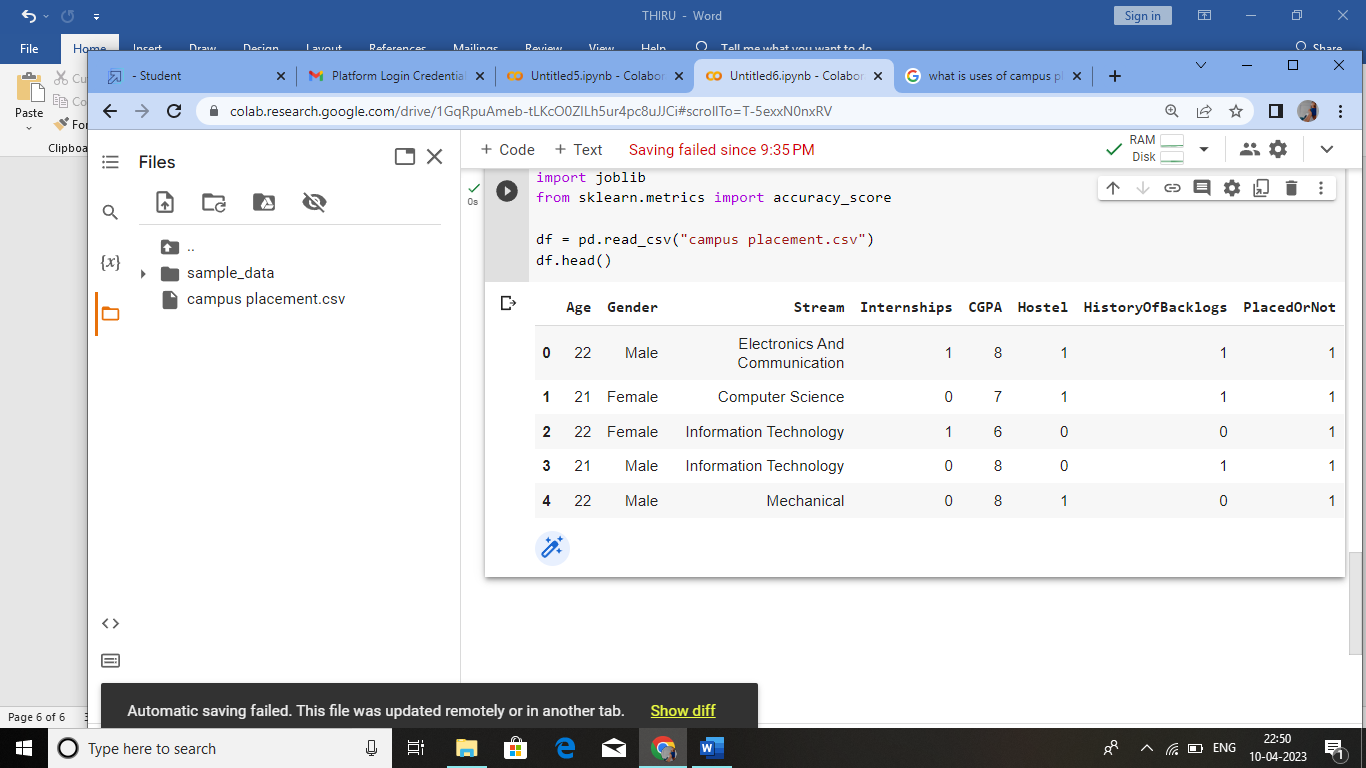


**3.RESULT:**

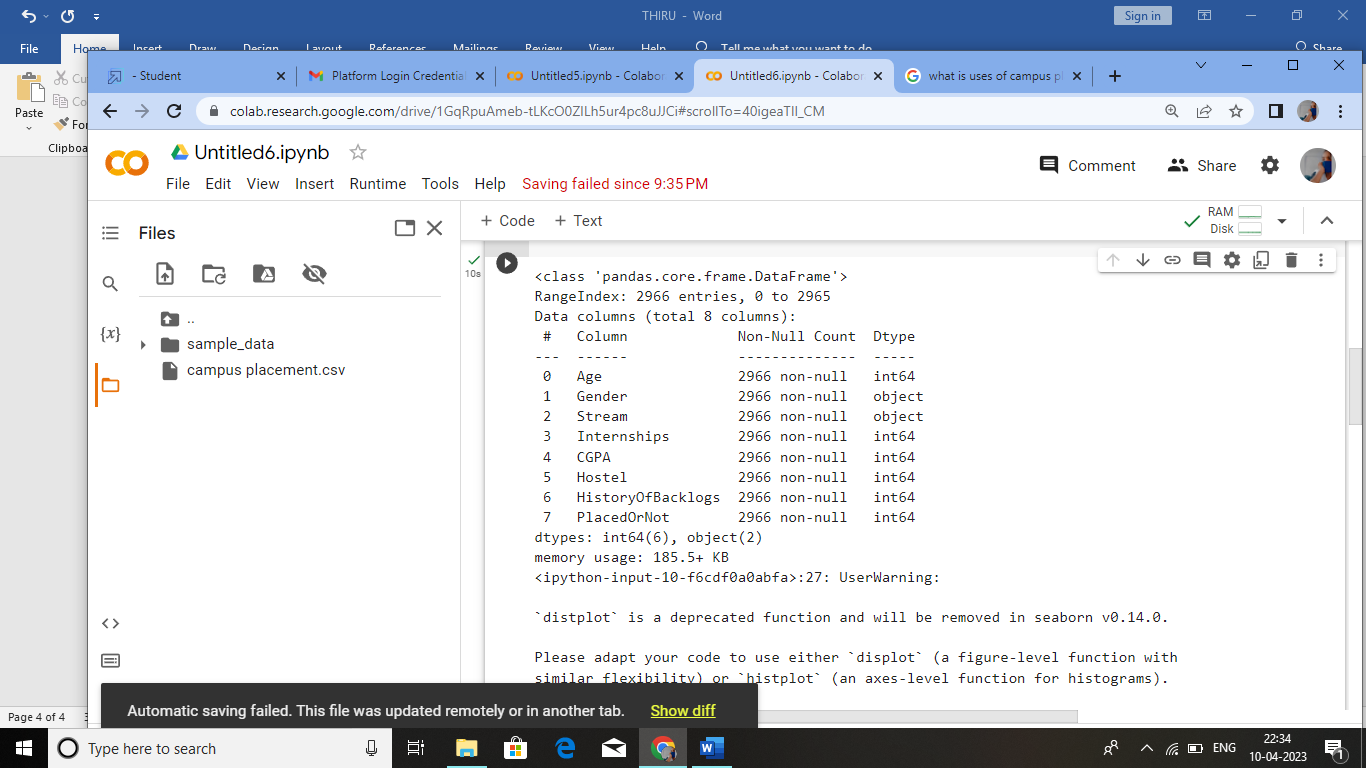
**(DATACOLLECTION AND PREPARATION)**

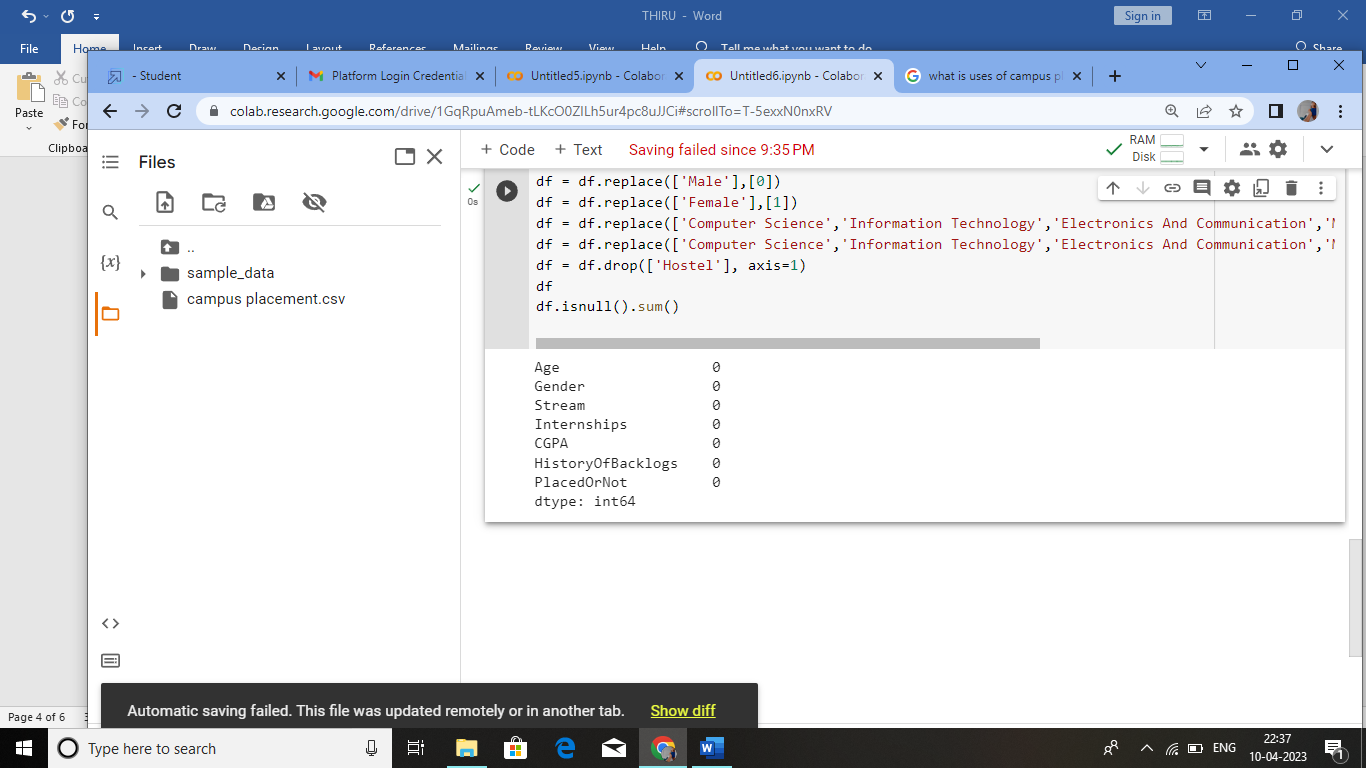
**IMPORTING THE LIBRARIES:**

READ THE DATASET:

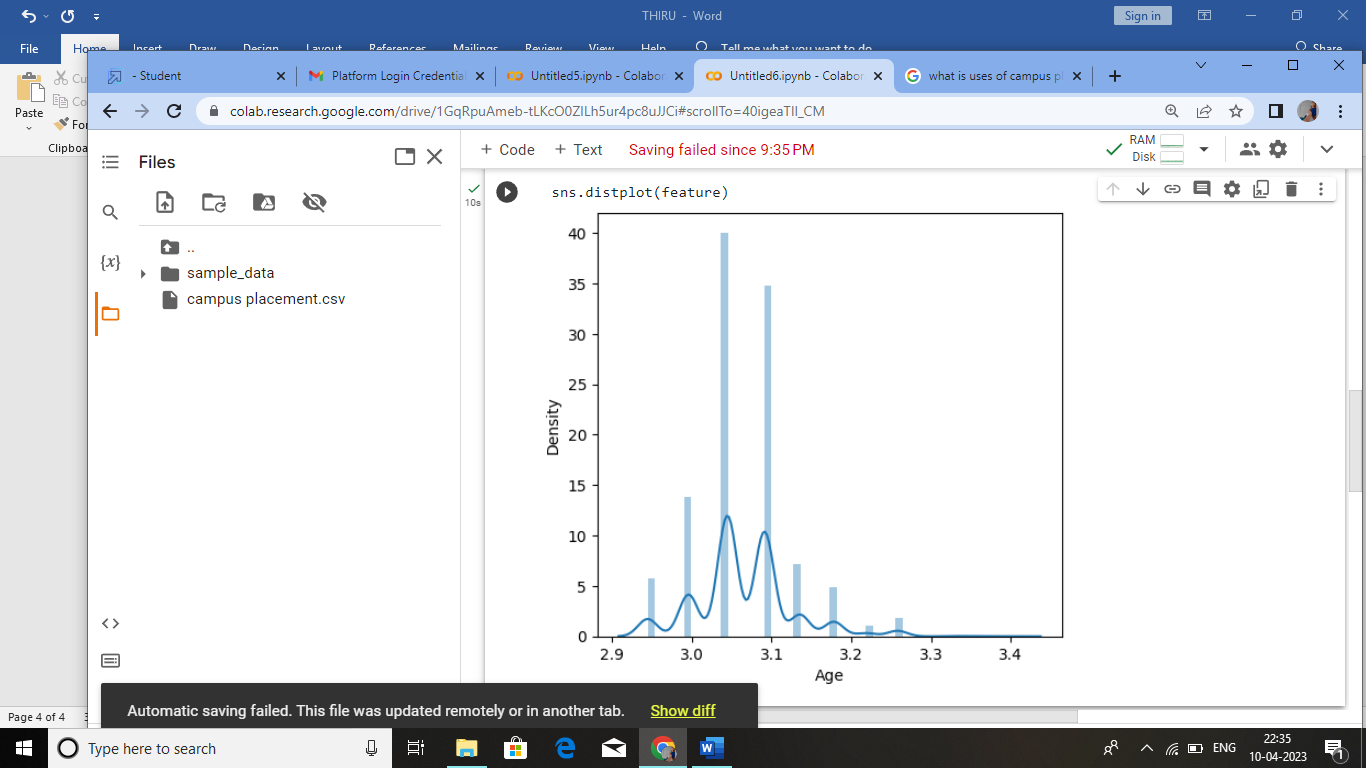


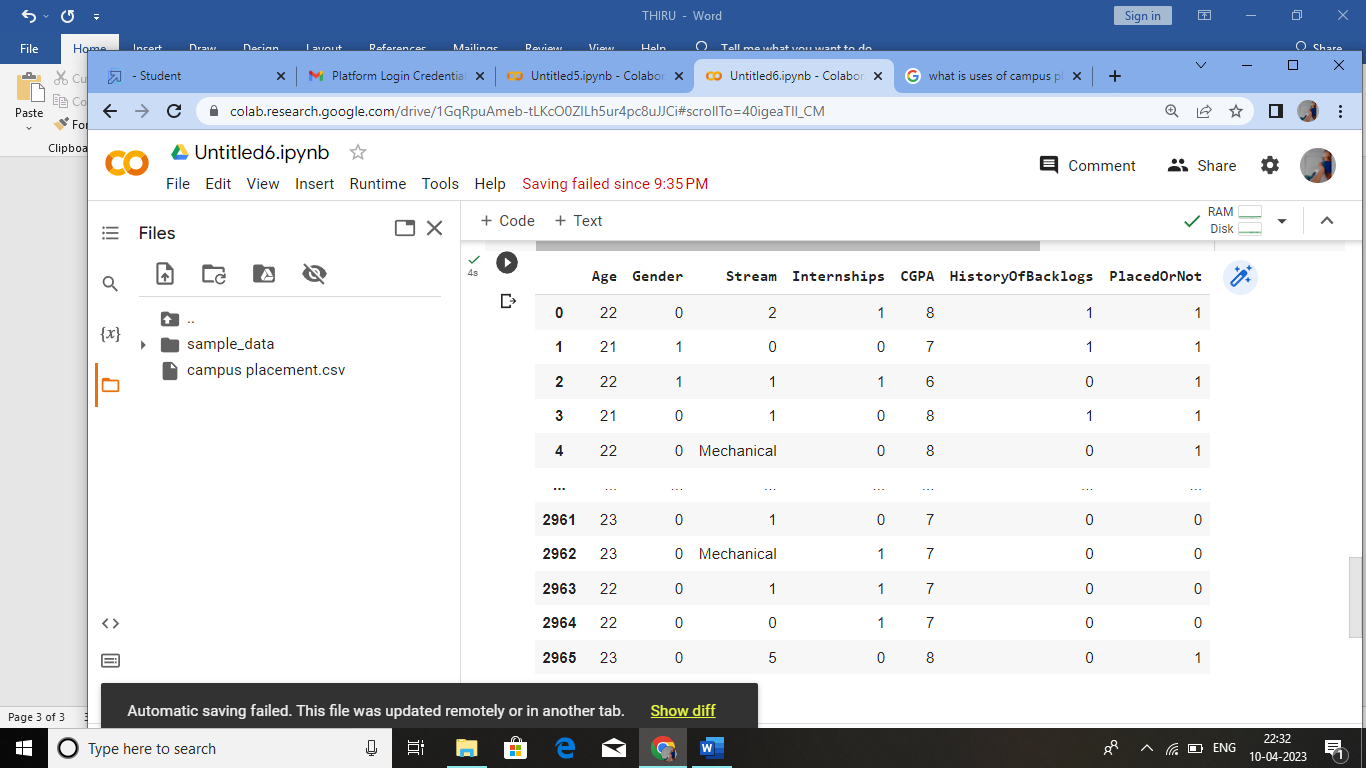
HANDLING MISSING VALUES:





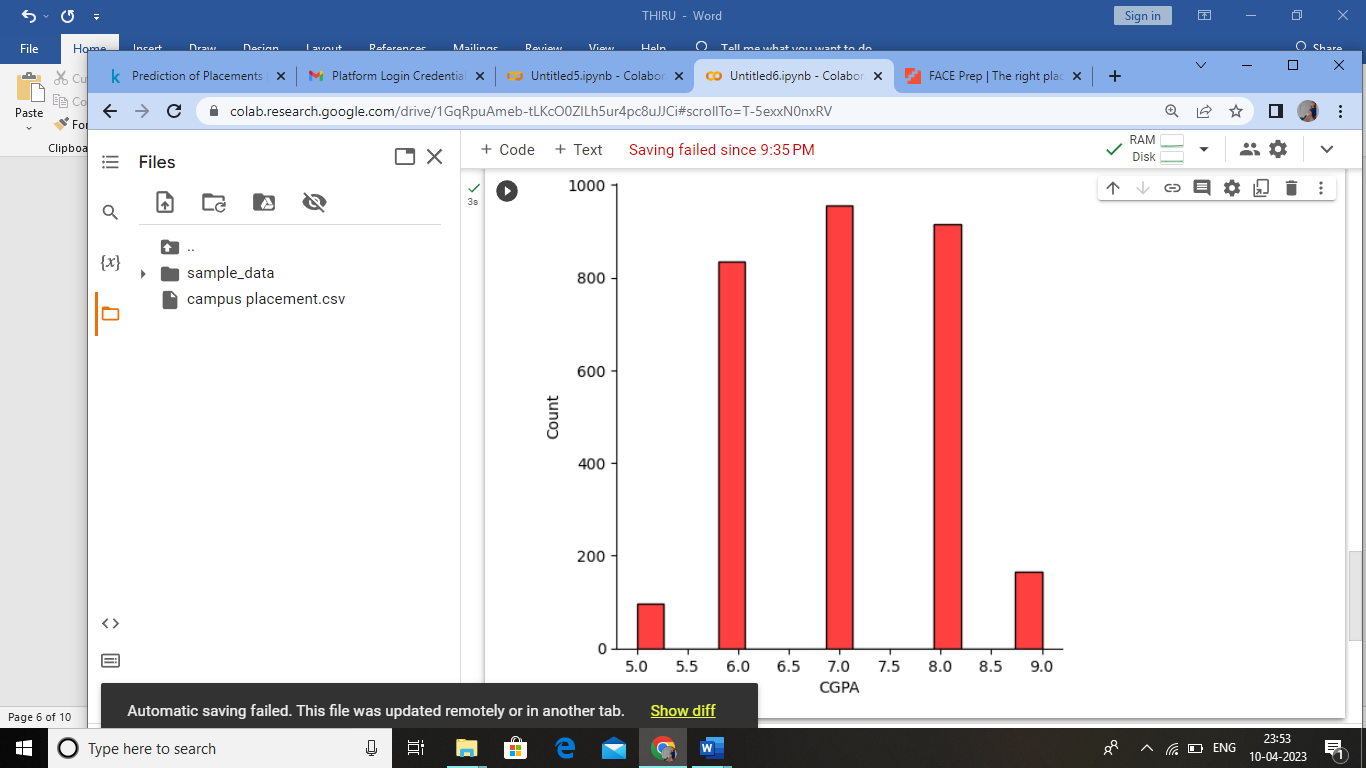
HANDLING OUTLIERS:

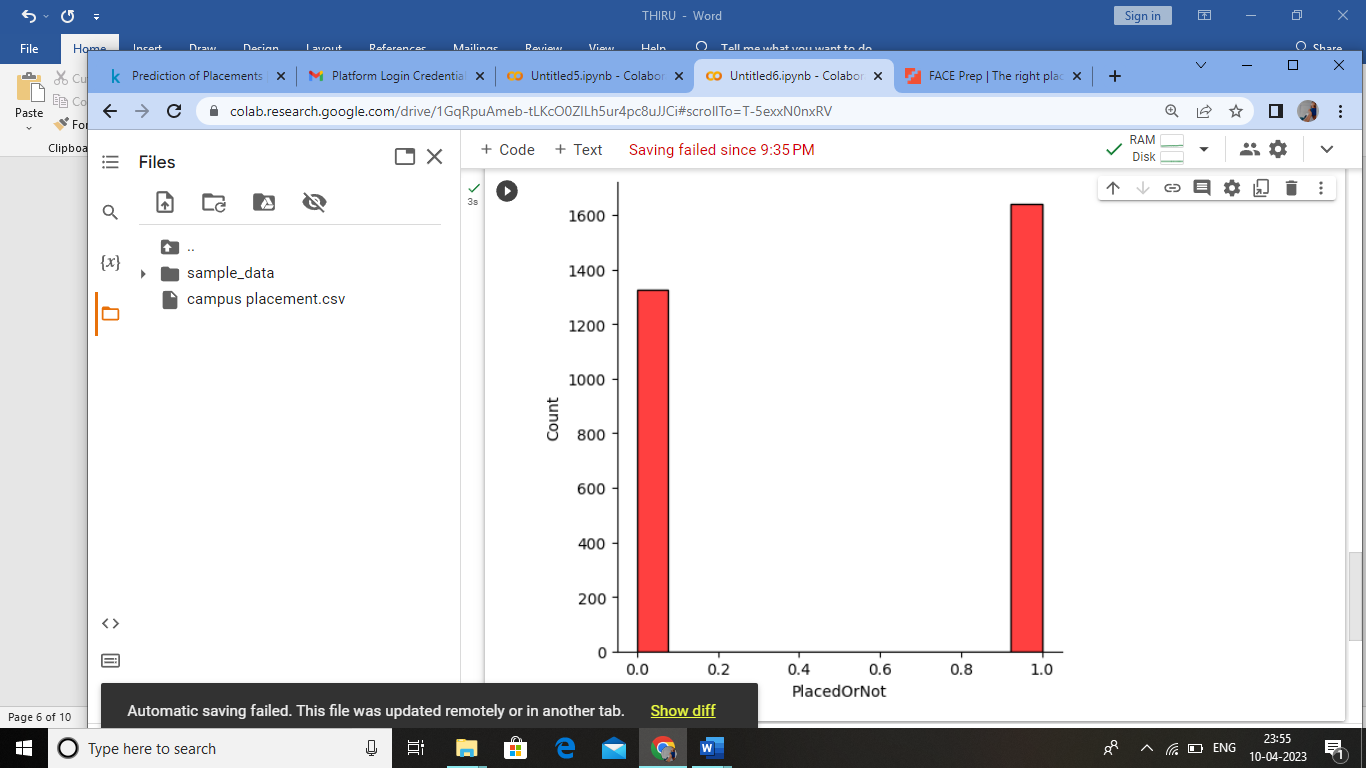


HANDLING CATEGORICAL VALUES:

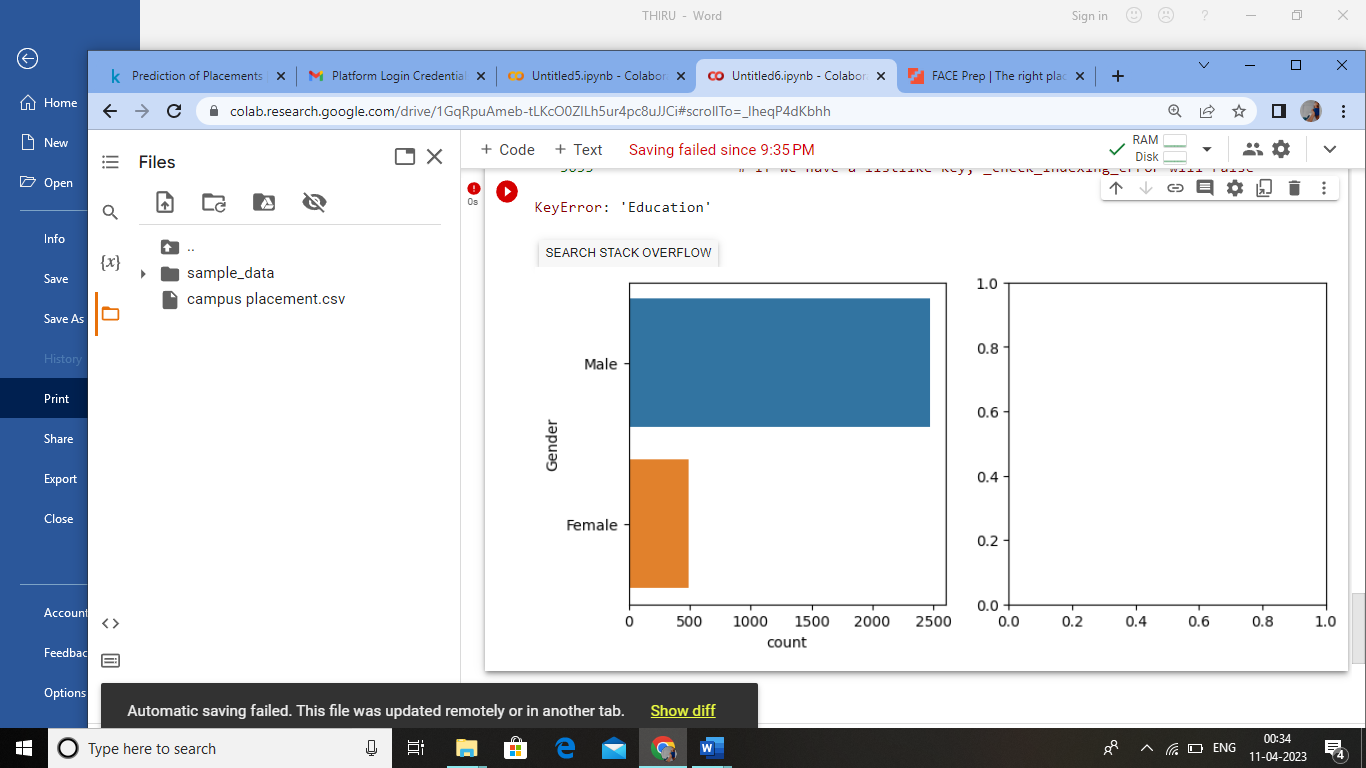
**EXPLORATORY DATA ANALYSIS:**

**UNIVARIATE ANALYSIS:**





**BIBARIATE ANALYSIS:**



**4.ADVANTAGES AND DISADVANTAGES:**

**Advantages:**  
  
1. A student who get’s recruited in Campus would have saved time, Even before completing their degree they getting recruited that is lot’s of Time saved.

2. A Students finishes Degree in a happy mood as they have bagged a Job. They do not have to think about the job search strategies.

3. A student gets Top Branded Organizations and attracts good offers, Top MNC’s recruit students from collage and offer Good Salary Package.

4. A student who gets recruited during campus selection enjoy a better social status with the collages, class mates & family.

**Disadvantages:**

1. Limited opportunities, assume if your collage invited 10 companies for campus selection then you are exposed to those 10 opportunities and if you are selected in campus then you had limited opportunities in front of you. but If you are not selected then you will search job in the open market then you would be having all companies in the domain as your opportunity.

2. If you are selected in campus then there are least chances for you to choose your special area of your interest. Assume if you are an Engineer and have dreams of making Career as IoT Developer then there are least chances you getting placed in special area of your interest in Campus.

3. Most of the Campus recruitments are conducted by big organizations like MNC’s, but small sector and startups companies offer an excellent growth and learning curve, if you want to get expertise in a specific area then startup helps building your career better than an MNC. MNC’s may provide a better salary and facilities but small companies and startup provide better projects & work experience and better foundation for your career. You may not get social status but you will be satisfied with your project & Work Experiences.

**5.APPLICATIONS:**

* University
* College
* Organization

**6.CONCLUSION:**

* At the completion of placement, students and supervisors should complete the end of placement evaluation form. It is the student's responsibility to arrange for a time for feedback no later than the last day of placement. Students or supervisors may contact the Placement Coordinator or Course Convenor and request the attendance of one of the University staff at the meeting. This evaluation form should be forwarded to the Placement Coordinator, with the supervisor and the student also keeping copies.

**7.FUTURE SCOPE:**

* In addition to completion of a final evaluation form, the supervisor and the student are requested to complete a form identifying goals for future placement. These goals will identify the competency areas that are seen as requiring further development. This form must be taken by the student to his/her next placement for the subsequent supervisor to review and sign, and should also be forwarded to the Placement. This particular measure enables some continuity between placements.

**8.APPENDIX:**

**A.SOURCE CODE:**

**DATA COLLECTION & PREPARATION:**

* **COLLECT THE DATASET:**

<https://www.kaggle.com/code/neesham/prediction-of-placements/data>

* **IMPORTING THE LIBRARIES:**

import numpy as np

import pandas as pd

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

from sklearn.metrics import accuracy\_score

* **READ THE DATASET:**

df = pd.read\_csv("campus placement.csv")

df.head()

* **HANDLING MISSING VALUES:**

df.info()

df.isnull().sum()

* **HANDLING OUTLIERS:**

def transformationplot(feature):

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

plt.subplot(1,2,1)

sns.distplot(feature)

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

* **HANDLING CATEGORICAL VALUES:**

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

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

df = df.replace(['Computer Science','Information Technology','Electronics And Communication','Mechancial','Electrical','Civil'],[0,1,2,3,4,5])

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

df

**EXPLORATORY DATA ANALYSIS:**

* **UNIVARIATE ANALYSIS:**

plt.figure (figsize=(12,5))

plt.subplot(121)

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

plt.figure (figsize=(12,5))

plt.subplot(121)

sns.displot(df['PlacedOrNot'],color='r')

* **BIBARIATE ANALYSIS:**

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

plt.subplot(1,4,1)

df=pd.read\_csv('campus placement.csv')

data=df

sns.countplot(y=data['Gender'],data=df)

plt.subplot(1,4,2)

sns.countplot(data['Education'])

plt.show()

**Multivariate Analysis:**

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

Plt.subplot(131)

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

2.sns.swarmplot(df[‘PlacedOrNot’],df[’CGPA’],hue=df[‘Stream’])

Scaling The Data

1.sc=stantardScaler()

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

X\_bal = pd.DataFrame(x\_bal,columns=names)

**Splitting The Data Into Train And Test**

1.x = standardized\_data

Y=df[‘PlacedOrNot’]

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

**Model Building**

SVM Model:

1.classifier = svm.SVC(kernel=’linear’)

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

SVC(kernel =’linear’)

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

Training\_data\_accuracy = accuracy\_score(x\_train\_prediction, y\_train)

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

Accuracy score of the training data : 0.768497470489039

**KNN Model**

1.best\_k = {“Regular”:0}

Best\_score = {“Regular”:0}

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

## Using Regular training set

Knn\_temp = kNeighborsClassification(n\_neighbors=k)

Knn\_temp.(X\_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

2.print(“------Regulars---\nk: {}”.format(best\_k, best\_score))

## Instantiate the models

Knn = KNeighborsClassification(n\_neighbors=best\_k[“Regular”])

## Fit the model to the training set

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

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

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

**Artificial Neural Network Model:**

1.import tensorflow as tf

from tensorflow import keras

from keras.model 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\_dia = 6))

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

#add 2nd 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 model

Loss\_1 = tf . keras.losses.binarycrossentropy( )

Classifier.compile(optimizer = ‘Adam’, loss = loss\_1 , metrics = [‘accuracy’])

[ ] #fitting the model

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

**Model Deployment**

**Save the Best Model**

1.[ ] import pickle

Pickle.dump(knn,open(“placement.pk1” , ‘wb’))

Model = pickle.load(open(‘placement.pk1’ , ‘rb’))

**Building Html Pages**

<section id=”hero” class=’d-flex flex- column justify-content-center”>

<div Class=”container”>

<div class=”row justify-content-center”>

<div class=”col-xl-8”>

<h1>Identifying patterns and trends in campus placement Data using Machine Learning</h1>

</div>

</div>

</div>

</section><!—End Hero -->

**Building Html Pages(part-2)**

1.<section id=”about” class=”about”>

<div class=”container”>

<div class=”section-title”>

<h2>fill the details</h2>

</div>

<div class=”row content”>

<div class=”first”>

<form action=”{{ url\_for(‘y\_predict’)}}” method=”POST”>

<input type=”number” id=”sen1” name=”sen1” placeholder=”Age”>

<input type=”number” id=”sen2” name=”sen2” placeholder=”Gender M(0),F(0)”>

<input type=”number” id=”sen3” name=”sen3” placeholder=”Stream CS(0),IT(1),ECE(2),Mech(3),EEE(4),Civil(5)”>

<input type=”number” id=”sen4” name=”sen4” placeholder=”Interships”>

<input type=”number” id=”sen5” name=”sen5” placeholder=”CGPA” >

<input type=”number” id=”sen6” name=”sen6” Placeholder=”Number of backlogs”>

</form>

</div>

</div>

</div>

</section><! --End About Us Section -->

2.<section id=”hero” class=”d-flex flex-column justify-cintent-center”>

<div class=”container”>

<div class=”row justify-content-center”>

<div class=”col-x-8”>

<h1>The Prediction is : {{y}}</h1>

<h3> 0 represents Not-Placed </h3>

<h3> 1 represents Placed<h2>

</div>

</div>

</div>

</section><! - - End Hero - ->

**Building Html Pages(part-2)**

1.<section id=”about” class=”about”>

<div class=”container”>

<div class=”section-title”>

<h2>fill the details</h2>

</div>

<div class=”row content”>

<div class=”first”>

<form action=”{{ url\_for(‘y\_predict’)}}” method=”POST”>

<input type=”number” id=”sen1” name=”sen1” placeholder=”Age”>

<input type=”number” id=”sen2” name=”sen2” placeholder=”Gender M(0),F(0)”>

<input type=”number” id=”sen3” name=”sen3” placeholder=”Stream CS(0),IT(1),ECE(2),Mech(3),EEE(4),Civil(5)”>

<input type=”number” id=”sen4” name=”sen4” placeholder=”Interships”>

<input type=”number” id=”sen5” name=”sen5” placeholder=”CGPA” >

<input type=”number” id=”sen6” name=”sen6” Placeholder=”Number of backlogs”>

</form>

</div>

</div>

</div>

</section><! --End About Us Section -->

2.<section id=”hero” class=”d-flex flex-column justify-cintent-center”>

<div class=”container”>

<div class=”row justify-content-center”>

<div class=”col-x-8”>

<h1>The Prediction is : {{y}}</h1>

<h3> 0 represents Not-Placed </h3>

<h3> 1 represents Placed<h2>

</div>

</div>

</div>

</section><! - - End Hero - ->

**Import The Libraries**

From flask import flask, render\_template , request

App=Flask(\_\_name\_\_)

Import pickle

Import joblib

Model=pickle.load(open(“placement123.pkl”,’rb’))

Ct=joblib.load(‘placement’)

**Render HTML Page**

1.@app.route(‘/’)

Def hello():

Return render\_template(“index.html”)

[2.@app.route(‘/guest](mailto:2.@app.route(‘/guest)’ , methods = [“POST”])

Def Guest():

Se1=request.form[“sen1”]

Se2=request.form[“sen2”]

Se3=request.form[“sen3”]

Se4=request.form[“sen4”]

Se5=request.form[“sen5”]

Se6=request.form[“sen6”]

@app.route(‘/y\_predict’ , methods = [“POST”)

def y\_predict():

]x\_test = [[(yo) for yo in request.form.values()]]

Prediction =model.predict(x\_test)

Prediction = prediction[0]

Return render\_template(“secondpage.html”,y=prediction)

**Main Function**

1.app.run(debug=True)