CAMPUS PLACEMENT DATA USING MACHINE LEARNING

INTRODUCTION:

OVERVIEW:

The placement both for final jobs and summer internships is an integral part of any institute's annual calendar of activities.

Campus placement is hiring young talent for internships and entry level positions.

PURPOSE:

The companies will be benefited from getting wide choice of

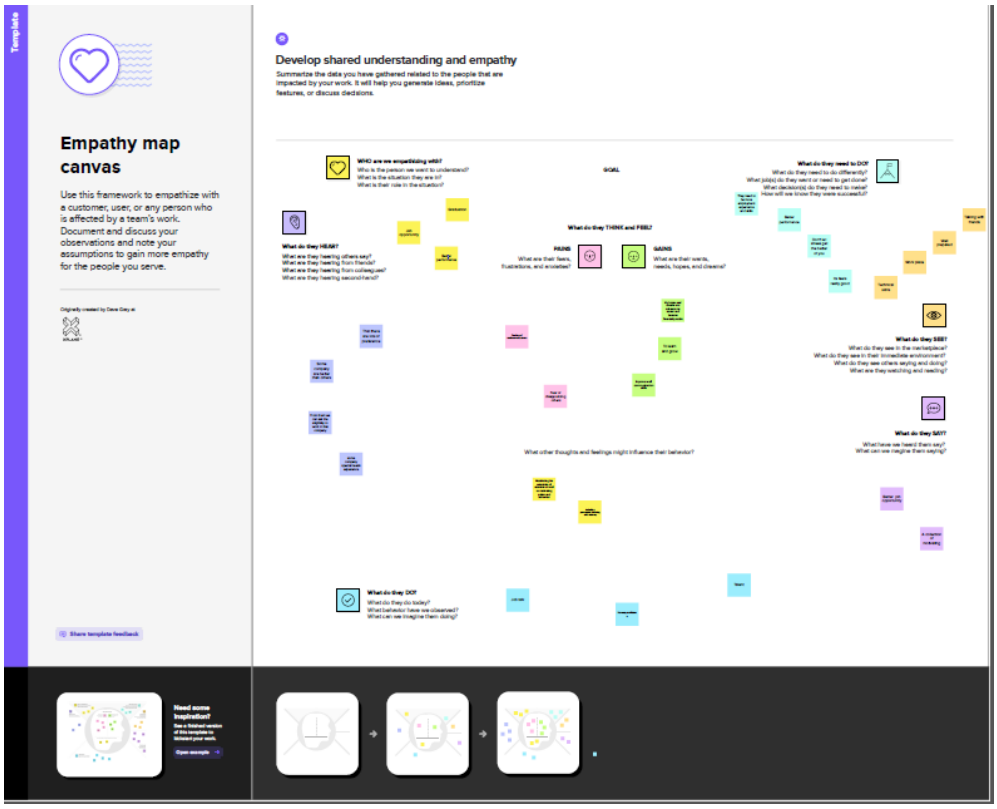
candidates to select for different job posts. Companies can select the right and talented candidate from a vast pool of young applicants within a limited time. On the other hand, students have the advantage of getting a good job according to their qualification level even before the completion of their academic course in college.

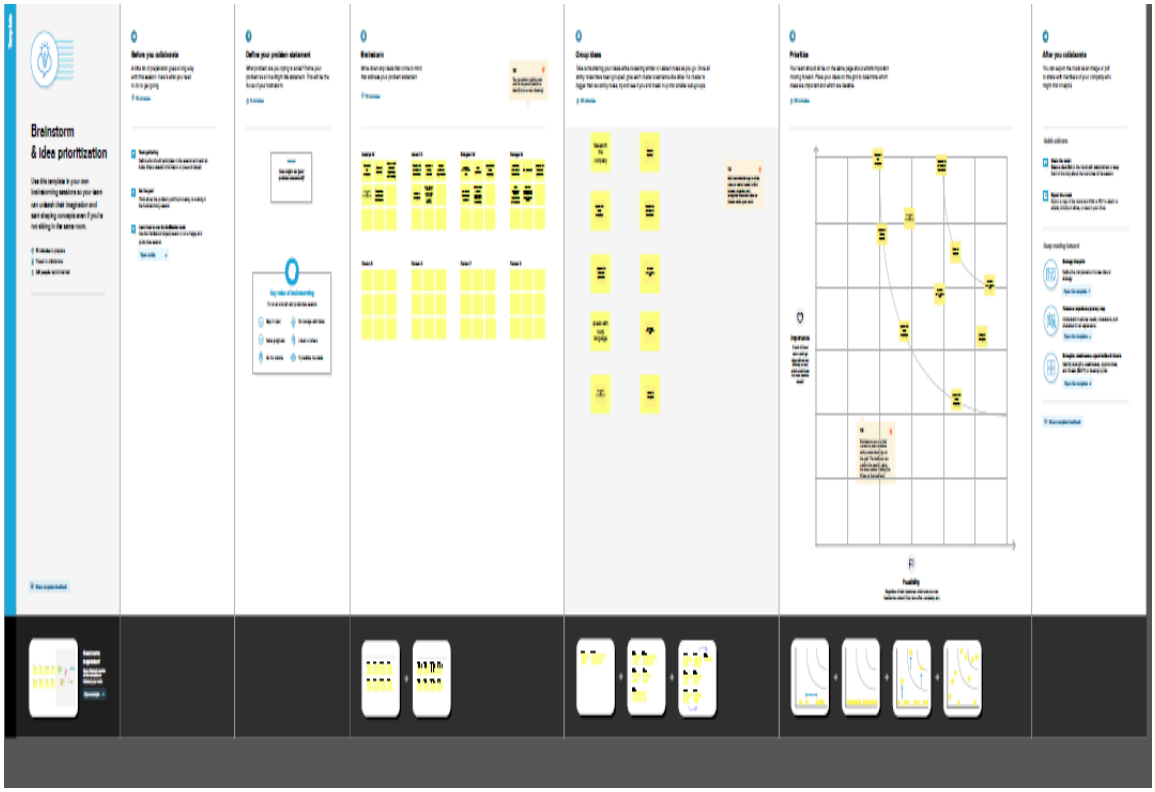
Campus placement or campus recruiting is a program conducted within universities or other educational institutions to provide jobs to students nearing completion of their studies.

In this type of program the educational institutions partner with corporations who wish to recruit from the student population.

PROBLEM DEFINITION AND DESIGN THINKING

EMPATHY MAP



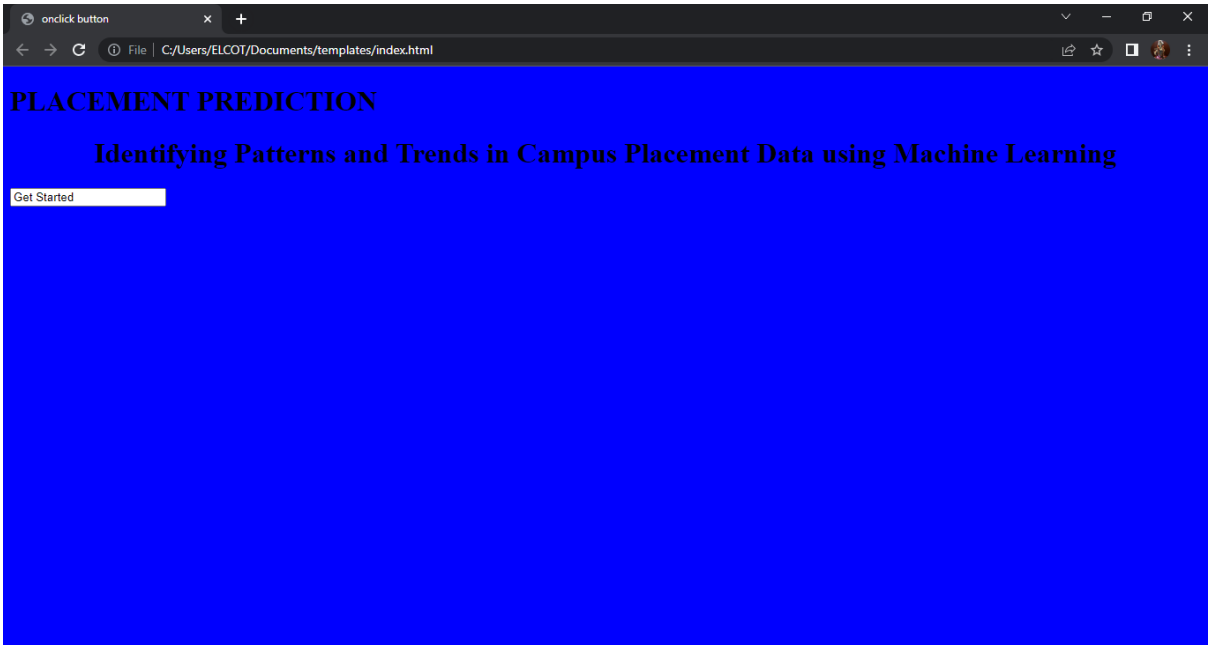


IDEATION & BRAINSTROMING MAP

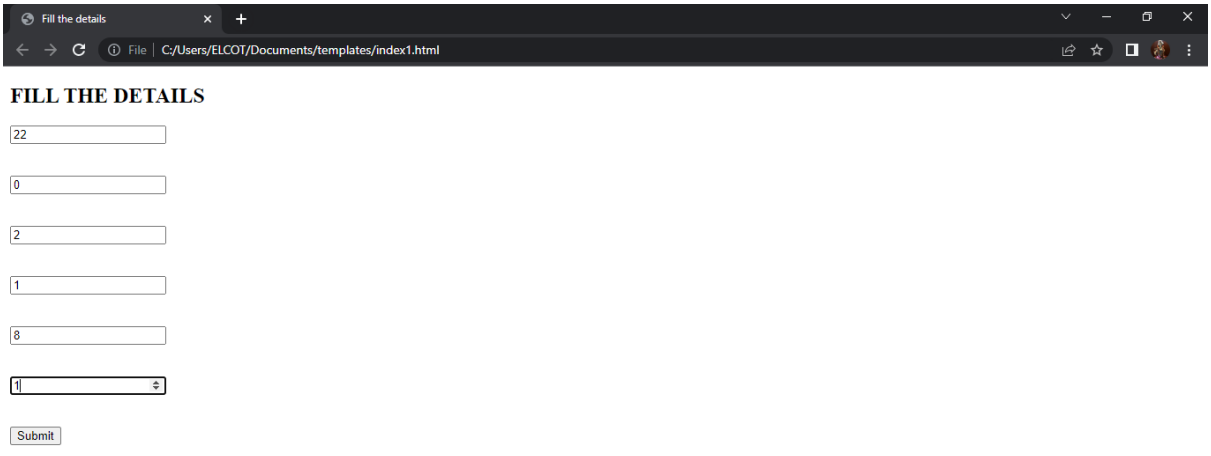
RESULT

Final finding output of the project

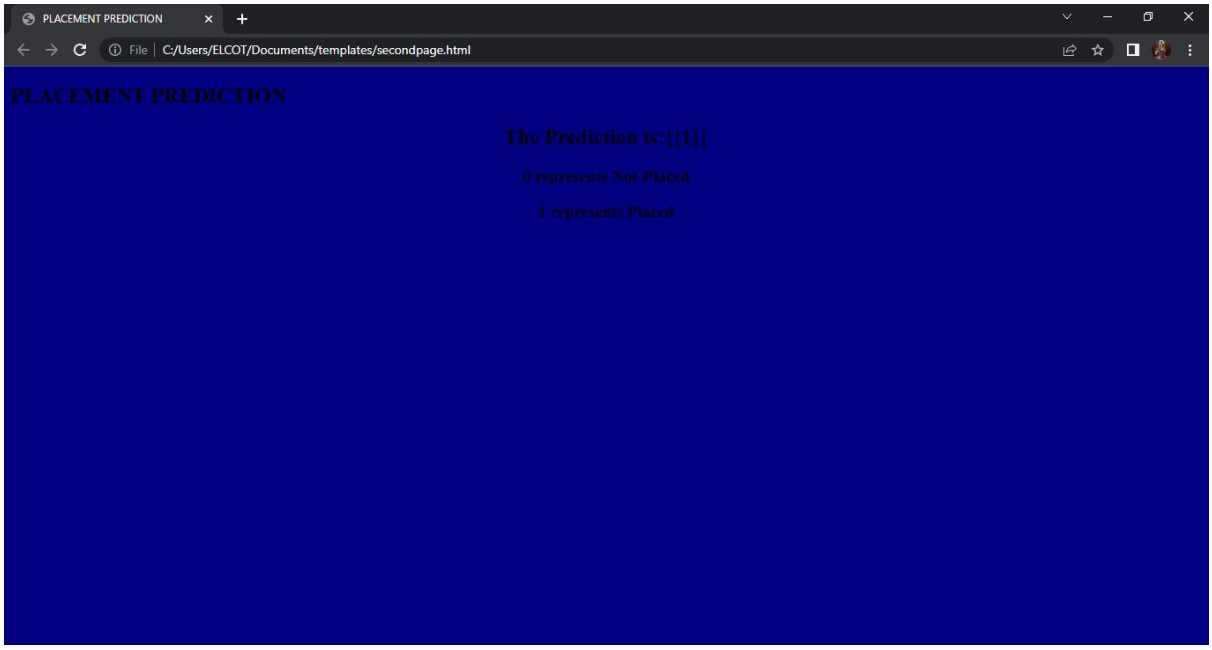
- **Index.html**



➢ Index1.html



➢ Second page.html



ADVANTAGE OF CAMPUS PLACEMENT:

- The companies will be benefited from getting wide choice of candidates to select for different jobs.
- The companies will be benefited from getting wide choice of candidates to select for different job posts.
- Companies can select the right and talented candidate

from a vast pool of young applicants within a limited time.

- On the other hand, students have the advantage of getting a good job according to their qualification level even before the completion of their academic course in college.

DISADVANTAGES OF CAMPUS PLACEMENT:

- Campus recruitment is an expensive affair for majority of the companies as it adds up costs to the bottom line.
- Companies incur different expenses related to travel, boarding, training etc while conducting campus selection process.

- The experienced and skilled candidates having practical job exposures cannot be recruited through campus placements.
- Fresh candidates selected through campus placements require adequate training for work.
- This is an additional expense for the company. Also, students can't work with their dream company and will have to remain satisfied with the company that recruits them during campus selection.

APPLICATION

- Companies hold on campus recruitment drives for students in their final year, and sever large.

- You can expect questions related to coding, algorithm and machine learning.

CONCLUSION

- Goal for future placement
- Performance of student
  - At the completion of placement, student and supervisors should complete the end of placement evaluation form.
  - To determine what merits satisfactory or unsatisfactory performance on placement.

FUTURE SCOPE

- Given the boom in the market, college grads or undergrads have access to a wide pool of employers promising attractive roles and benefits.

- We need to use technology and creative communication strategies to stay top of mind in our target demographic.

APPENDIX

Source code

identify_patterns_campus_placement.ipynb

```
# -*- coding: utf-8 -*-
"""identify_patterns_campus_placement.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1pTQAxMsKojtcn_aaPPpUUBibtRMz-U9Y
"""
```

```
import pandas as pd
import numpy as np
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

from google.colab import drive
drive.mount('/content/drive')

with open('/content/drive/My Drive/Colab Notebooks/Dataset/collegePlace.csv', 'r') as dataset:
  df = pd.read_csv(dataset)

df.head()
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','Civi
l'],[0,1,2,3,4,5])

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

3

```
df.describe()


df.isnull().any()


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='r')


from pandas.core.indexes.multi import
names_compat
#perfroming feature Scaling op[eration using
standard scaller on X part of the dataset
because
```

```
#there different type of values in the columns
x_bal = df.iloc[:,0:6]
x_bal.head()

sc = StandardScaler()
x_bal = sc.fit_transform(x_bal)

X = x_bal
Y = df['PlacedOrNot'].values

X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)

4

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

```
classifier.fit(X_train, Y_train)

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)

best_k = {"Regular":0}
best_score = {"Regular":0}
for k in range (3,50,2):

    ## Using Regular training set
    knn_temp = KNeighborsClassifier(n_neighbors = k) #Instantiate the model
```

```
    knn_temp.fit(X_train, Y_train)
#Fit the model to the training set
    knn_temp_pred =
knn_temp.predict(X_test)
#Predict on the test set
    score = metrics.accuracy_score(Y_test,
knn_temp_pred)*100       #Get accuracy
    if score >=best_score["Regular"] and
score < 100:              #store best params
        best_score["Regular"] = score
        best_k["Regular"] = k


print (" --- Results---\nK: {}\nScore:
{}".format(best_k, best_score))
##Instantiate the models
knn =
KNeighborsClassifier(n_neighbors=best_k["R
egular"])
##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)

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 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)
```

 # Commented out IPython magic to ensure Python compatibility.

import pickle

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

 # %cp '/content/placement.pkl' '/content/drive/MyDrive/Colab Notebooks/'

APP.PY

```
# -*- coding: utf-8 -*-
"""app.py

Automatically generated by Colaboratory.
```

Original file is located at

https://colab.research.google.com/drive/1jqE58myCD81b_eM0ol-ulcictLe9GtFU
"""

# Commented out IPython magic to ensure Python compatibility.
from flask import Flask, render_template, request
app=Flask(__name__)

from google.colab import drive
drive.mount('/content/drive')

%cp '/content/drive/MyDrive/Campus

```
placement/Traiining
/placement.pkl' '/content/'
# %cp -r
'/content/drive/MyDrive/Campus
placement/Flask/templates/'
'/content/'

import pickle
import joblib
model =
pickle.load(open("placement.pkl",'
rb'))
#ct=joblib.load('placement')

#install pyngrok
!pip install pyngrok

#install ngrok
from pyngrok import ngrok
```

```
app= Flask(__name__)

ngrok.set_auth_token("2OEHHtKu
tPcQBe8AQKVWfgXGfB6_3cBJA7m
cM1KPU76VBhuBy")
public_url = ngrok.connect(5000)
print("URL : ",public_url)

@app.route('/', methods=['GET',
'POST'])
def hello():
  return
render_template("index.html")

@app.route('/guest',
methods=["POST"])
def Guest():
    return
render_template("index1.html")
```

```
@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.ht
ml",y=prediction)

app.run(debug=True)
```