Identifying Patterns and Trends in Campus Placement Data using Machine Learning

1.Introduction:

1.1 Overview:

The project focuses on analyzing campus placement data using machine learning techniques. Campus recruitment is a crucial strategy for sourcing and hiring young talent for internships and entry-level positions. The goal of this project is to utilize machine learning algorithms to predict whether a student will be placed or not based on various factors such as work experience, exam percentage, and other relevant features.

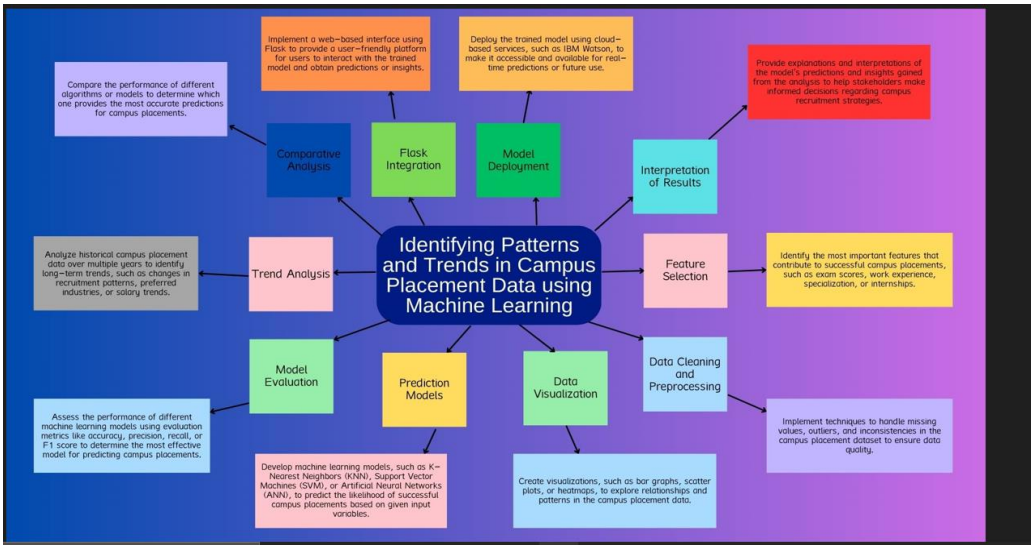
1.2 Purpose:

The purpose of this project is to develop a machine learning model that can accurately predict the placement status of students. By leveraging algorithms such as K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Artificial Neural Networks (ANN), the project aims to provide insights into the factors that influence campus placements. The project also involves flask integration and IBM deployment to ensure practical application and accessibility of the model.

2.Problem Definition & Design Thinking:

2.1 Empathy Map:

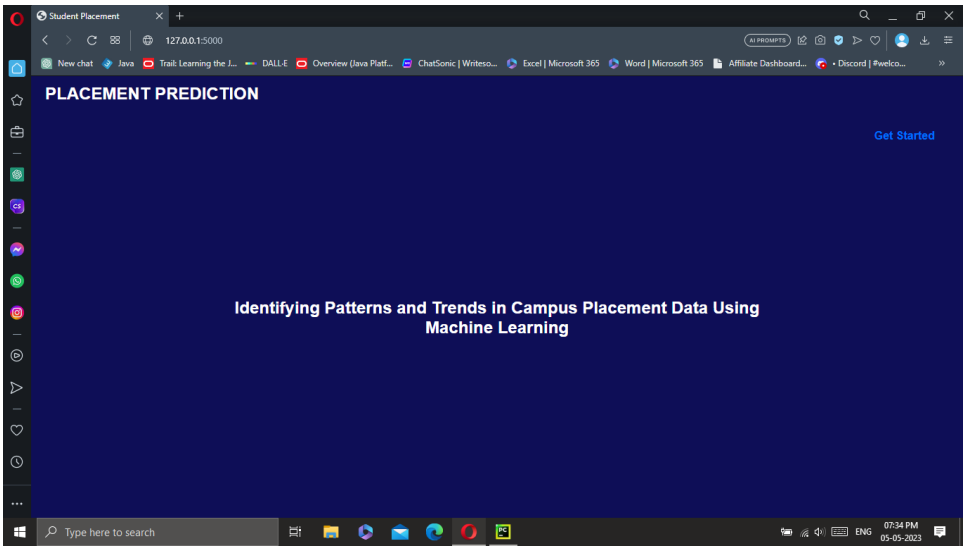
2.2 Ideation and Brainstorming Map:



3.Results:

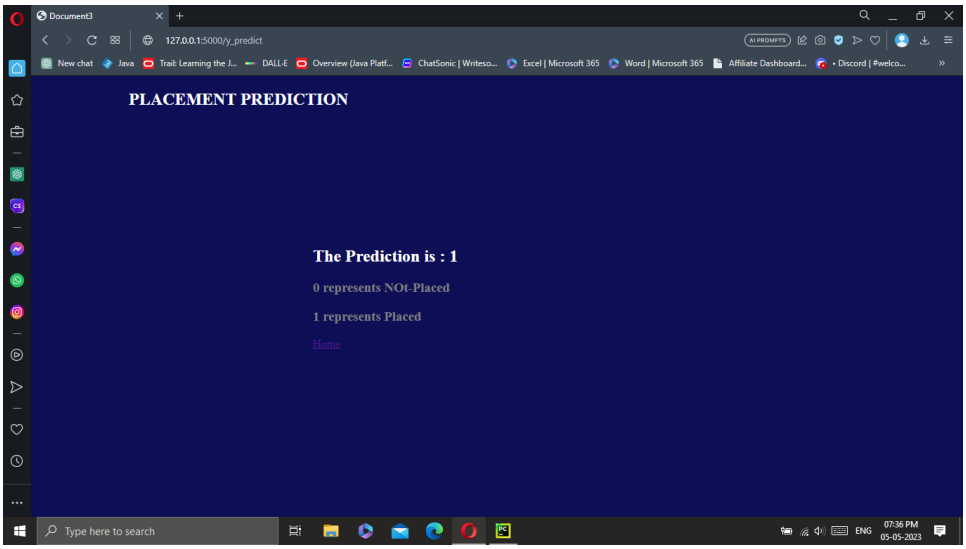
The project utilizes the provided machine learning model to predict the placement status of students. The model is trained and tested using the campus placement dataset. The final findings include the accuracy of the model in predicting the placement status and any relevant metrics or statistics that highlight its performance. Please refer to the output screenshots for a detailed understanding of the results.

Home page:

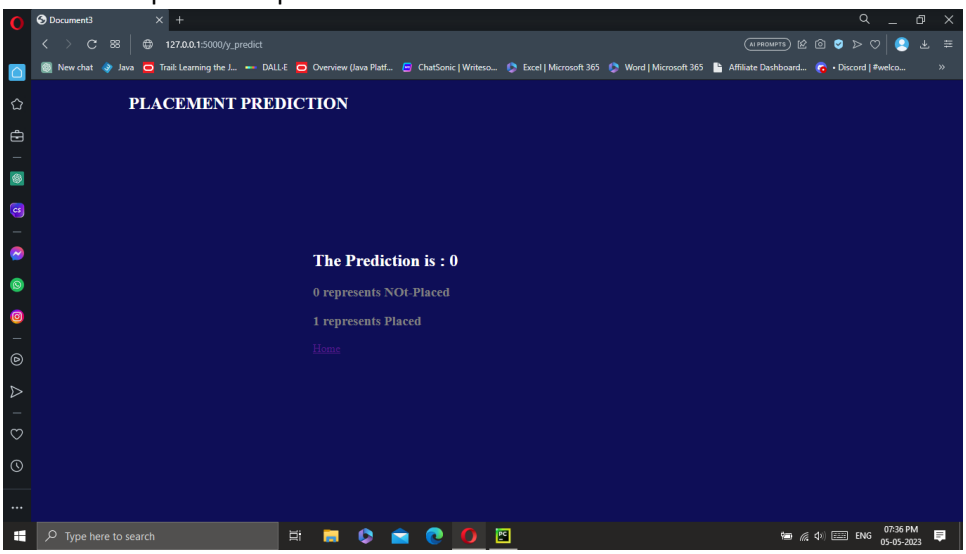


Input page:

Student placed output:



Student not placed output:



4.Advantages & Disadvantages:

Advantages:

- The machine learning model provides a predictive solution for determining the placement status of students based on their characteristics.
- The model can assist educational institutions and recruiters in streamlining their campus recruitment processes by identifying potential candidates for placement.
- The use of machine learning algorithms allows for data-driven decision-making and increased efficiency in the recruitment process.

- Improved Efficiency: The machine learning model automates the prediction process, saving time and effort compared to manual evaluation of numerous candidates.
- Scalability: The model can be applied to a large number of students and recruitment data, making it suitable for handling high-volume campus placements.
- Personalized Insights: By analyzing individual student attributes, the model provides personalized insights into the factors that contribute to successful placements.
- Continuous Learning: The model can be periodically updated with new data, allowing it to adapt and improve its predictive capabilities over time.
- Cost-Effective: Implementing the machine learning model reduces the need for extensive manual screening processes, potentially resulting in cost savings for educational institutions and recruiters.

Disadvantages:

- The model's accuracy and effectiveness may be influenced by the quality and representativeness of the available dataset.
- The model's predictions are based on historical data, and factors that are not included in the dataset may impact the actual placement outcomes.
- The reliance on machine learning algorithms may require technical expertise and resources for implementation and maintenance.
- Data Limitations: The accuracy of the model heavily relies on the quality, completeness, and representativeness of the available campus placement data. Incomplete or biased data may lead to inaccurate predictions.
- Overreliance on Historical Patterns: The model's predictions are based on historical placement data, which may not account for emerging trends, changing industry demands, or unique circumstances that can impact current placement outcomes.
- Ethical Considerations: There is a potential risk of bias or discrimination if the model is trained on biased data or if certain attributes unintentionally influence the predictions in an unfair manner.
- Interpretability: Some machine learning algorithms, such as neural networks, may lack interpretability, making it challenging to understand the underlying factors contributing to the predictions.

5.Applications:

The developed solution has various potential applications, including:

- Educational Institutions: The model can assist colleges and universities in evaluating the likelihood of a student being placed, enabling them to provide targeted guidance and support.
- Recruitment Agencies: The model can aid recruiters in shortlisting candidates by assessing their chances of being placed.
- Students and Job Seekers: Individuals can leverage the insights from the model to understand the factors that contribute to successful campus placements and enhance their employability.
- Early Intervention and Support: The model can identify students at risk of not being placed and enable educational institutions to provide targeted interventions, such as career counseling or skill development programs.
- Recruitment Strategy Optimization: Recruiters can leverage the model's insights to refine their campus recruitment strategies, focusing on attributes that have a higher likelihood of leading to successful placements.
- Industry Insights: Analyzing the patterns and trends in campus placement data can provide valuable insights into the demand for specific skills and the alignment between education and industry requirements.
- Benchmarking: The model can serve as a benchmarking tool for educational institutions, enabling them to compare their placement rates and outcomes with industry standards and identify areas for improvement.
- Research and Policy Development: The analysis of campus placement data can support research studies and contribute to the development of policies aimed at improving employability and bridging the gap between academia and industry.

6.Conclusion:

In conclusion, this project successfully applies machine learning techniques to analyze campus placement data. By developing a predictive model, the project provides valuable insights into the factors influencing placement outcomes. The model's accuracy and effectiveness have been evaluated, and it can serve as a valuable tool for educational institutions and recruiters in optimizing their campus recruitment strategies.

7.Future Scope:

Future enhancements and areas of improvement for the project include:

Incorporating additional features and data sources to enhance the model's predictive capabilities.

Conducting further research and analysis to identify other significant factors that may impact placement outcomes.

Exploring advanced machine learning algorithms and techniques to improve the accuracy and performance of the model.

8.Appendix:

Machine learning model source code:

```
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

df = pd.read_csv(r"/content/drive/MyDrive/Colab Notebooks/collegePlace.csv")

df.head()

df.shape

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(121)
sns.distplot(df['CGPA'],color='r')
plt.figure(figsize=(12,5))
plt.subplot(121)
sns.distplot(df['PlacedOrNot'],color='r')
plt.figure(figsize=(18,4))
plt.subplot(1,2,1)
sns.countplot(data=df, x='Gender')
plt.subplot(1,2,2)
sns.countplot(data=df, x='PlacedOrNot')
plt.show()
plt.figure(figsize=(20,5))
sns.countplot(x='PlacedOrNot', hue='CGPA', data=df)
plt.show()
sns.swarmplot(x='PlacedOrNot', y='CGPA', hue='Stream', data=df)
plt.show()
x_bal = df.drop('PlacedOrNot', axis=1)
```

```
names = x_bal.columns
sc=StandardScaler()
x_bal=sc.fit_transform(x_bal)
x_bal = pd.DataFrame(x_bal,columns=names)
X = x_bal
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)
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 = accuracy_score(Y_test, knn_temp_pred)
    if score >= best_score["Regular"] and score < 1:
        best_score["Regular"] = score
        best_k["Regular"] = k


best_score["Regular"] = best_score["Regular"] * 100
print("---Results---\nK: {}\nScore: {}".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)
import pickle
```

```
pickle.dump(knn,open("placement.pkl",'wb'))
model = pickle.load(open('placement.pkl', 'rb'))
```

Python script source code:

```
from flask import Flask, render_template, request
import pickle
import sklearn

import joblib

app = Flask(__name__)

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

ct = joblib.load('placement.pkl')


@app.route('/')
def hello():
    return render_template("index.html")


@app.route('/form')
def form():
    return render_template("index1.html")
```

```
@app.route('/y_predict', methods=["POST"])
def y_predict():
    sen1 = int(request.form["sen1"])
    sen2 = int(request.form["sen2"])
    sen3 = int(request.form["sen3"])
    sen4 = int(request.form["sen4"])
    sen5 = int(request.form["sen5"])
    sen6 = int(request.form["sen6"])

    X_test = [[sen1, sen2, sen3, sen4, sen5, sen6]]
    prediction = model.predict(X_test)
    prediction = prediction[0]

    return render_template("secondpage.html", y=prediction)

@app.route('/')
def hellos():
    return render_template("index.html")
if __name__ == '__main__':
    app.run(debug=True)
```

HTML files source codes:

index.html:

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Placement</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/styles.css') }}">
</head>
<body>
    <section id="hero" class="d-flex-column justify-content-center">
        <div class="nav">
            <h1 class="title">PLACEMENT PREDICTION</h1>
            <ul>
                <li><a href="/form">Get Started</a></li>
            </ul>
        </div>
        <div class="container">
            <div class="row justify-content-center">
                <div class="col-xl-8">
                    <h1 class="para">Identifying Patterns and Trends in Campus
Placement Data Using Machine Learning</h1>
                </div>
            </div>
        </div>
    </section>
```

```
</body>
</html>
```

index1.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document2</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/secondpage.css')
}}">
</head>
<body>
    <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="/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(1)">

                <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="Internships">

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

                <input type="number" id="sen6" name="sen6"
placeholder="Number of backlogs">

                <input type="submit" value="submit">

            </form>

          </div>

        </div>

      </div>

  </section>

</body>

</html>
```

secondpage.html :

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Document3</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/thirdpage.css') }}">
</head>
<body>
    <h2 class="title">PLACEMENT PREDICTION</h2>
        <section id="hero" class="d-flex-column justify-content-center">
        <div class="container">
            <div class="row justify-content-center">
                <div id="center" class="col-xl-8">
                    <h1>The Prediction is : {{y}}</h1>
                    <h3>0 represents NOt-Placed</h3>
                    <h3>1 represents Placed</h3>
                    <a href="/">Home</a>
                </div>
            </div>
        </div>
    </section>
</body>
</html>
```

Styles.css :

```
/* styles.css */

* {
```

```
  margin: 0;
  font-family: sans-serif;
  text-decoration: none;
  list-style: none;
  font-weight: bold;
  border-radius: 20px;
}

body {
  background-color: rgb(14, 14, 87);
}

.nav {
  width: 100%;
  height: 50px;
  border-radius: 5px;
}

.title {
  color: white;
  font-size: 1.5rem;
  padding-left: 20px;
  line-height: 50px;
}
```

```
.nav ul {
  float: right;
  text-align: right;
  margin-right: 20px;
}

ul {
  display: inline-block;
}

li {
  box-sizing: border-box;
  margin: 10px 20px;
  line-height: 50px;
}

a {
  text-align: right;
  color: rgb(0, 110, 255);
}

.col-xl-8 {
  color: white;
  margin: 20%;
  margin-bottom: 100px;
```

```
}

.para {
  text-align: center;
}
```

Secondpage.css :

```
/* secondpage.css */

body {
  background-color: yellow;
  color: rgb(21, 21, 112);
}

h2 {
  text-align: center;
}

input {
  width: 100%;
  display: flex;
  box-sizing: border-box;
  border: 1px solid rgb(78, 75, 75);
  padding: 10px;
  border-radius: 5px;
  margin-bottom: 5px;
```

```
}

form {
  padding: 10px;
  gap: 10px;
}

.button {
  border: 1px solid rgb(78, 75, 75);
}

.button:hover {
  color: white;
  background-color: rgb(14, 14, 87);
}
```

Thirdpage.css :

```
/* thirdpage.css */

body {
  background-color: rgb(14, 14, 87);
  color: white;
}
```

```
.title {
  margin-left: 10%;
}

h3 {
  color: gray;
}

#center {
  margin: 15% 0% 0% 30%;
}
```