Identifying Pattens and Trends in Campus Placement Data Using Machine Learning

Project Record Template

CHAPTER 1

1. INTRODUCTION

1.1 OVERVIEW

This project aims to use machine learning techniques to analyse data related to campus placements and identify patterns and trends that can provide insights into student employability and the job market. The project involves collecting and pre-processing data on various parameters such as student academic performance, skills, internships, job preferences, campus placement drives, and company recruitment patterns. The next step is to apply machine learning algorithms such as classification, regression, clustering, and association rule mining to identify correlations and patterns among the various parameters. Finally, the project aims to provide actionable insights that can help students, colleges, and recruiters make informed decisions related to campus placements. Overall, the project has the potential to improve the efficiency and effectiveness of campus placements by leveraging the power of machine learning.

Campus placements are a crucial part of a student's academic journey as they pave the way for future employment opportunities. However, the process can be challenging for both students and recruiters, and it requires careful consideration of various factors such as academic performance, skills, job preferences, and company requirements. In recent years, machine learning techniques have emerged as a powerful tool to analyse data and identify patterns, making them ideal for addressing complex problems such as campus placements.

1.2 PURPOSE

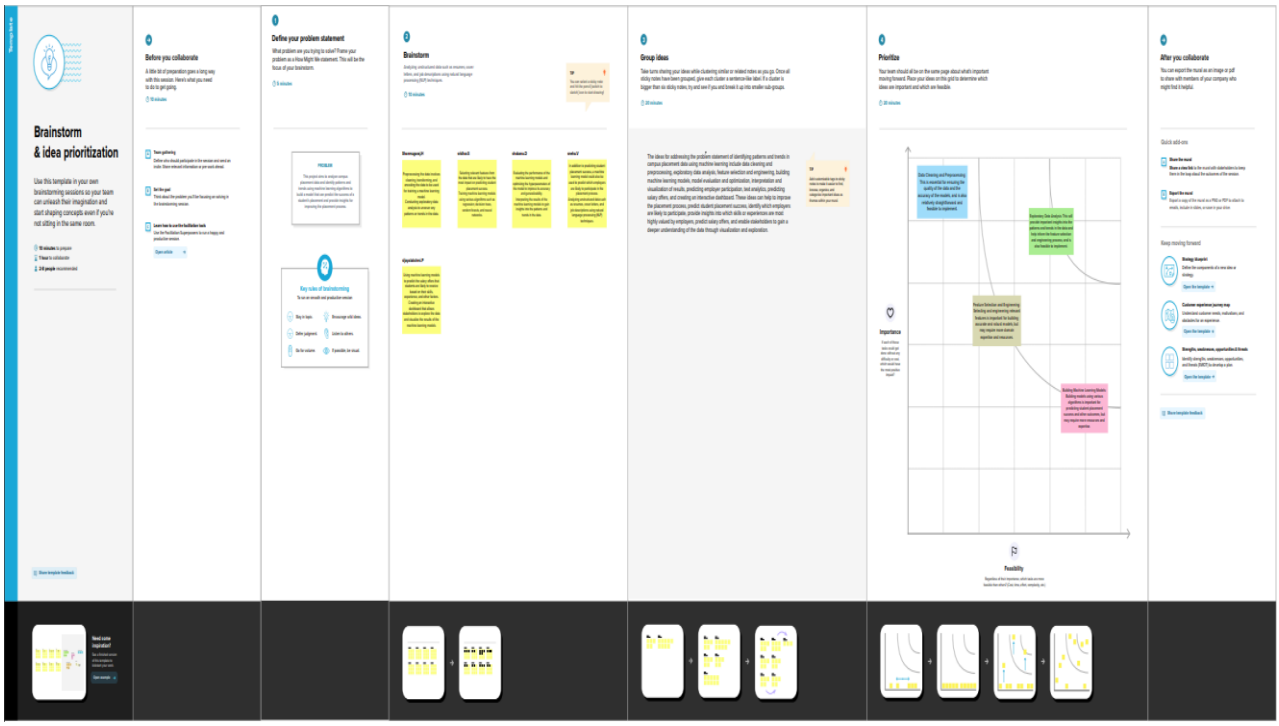
The purpose of this project is to leverage the power of machine learning techniques to identify patterns and trends in campus placement data. The primary objective is to provide insights that can help colleges, recruiters, and students make informed decisions related to campus placements. By analysing data on various parameters such as academic performance, skills, job preferences, campus placement drives, and company recruitment patterns, the project aims to identify correlations and patterns that can enhance the efficiency and effectiveness of the placement process. The goal is to improve student employability and the job market by providing valuable insights that can inform policy decisions and improve placement strategies.

Colleges and universities can use the insights generated from the project to improve their placement strategies and policies, such as modifying their curriculum to meet the industry's requirements and providing more relevant training to students. Recruiters can use the insights to identify high-potential candidates and make more informed decisions about the hiring process, such as targeting specific departments or student profiles. Students can benefit from the project by using the insights to identify areas they need to improve to enhance their employability and make informed decisions about their career paths. Policymakers and government agencies can use the insights generated to formulate policies that support the growth of the job market and improve the employability of graduates.

CHAPTER 2

2.PROBLEM DEFINITION & DESIGN THINKING

2.2 IDEATION & BRAINSTROMING MAP



CHAPTER 3

3.RESULT

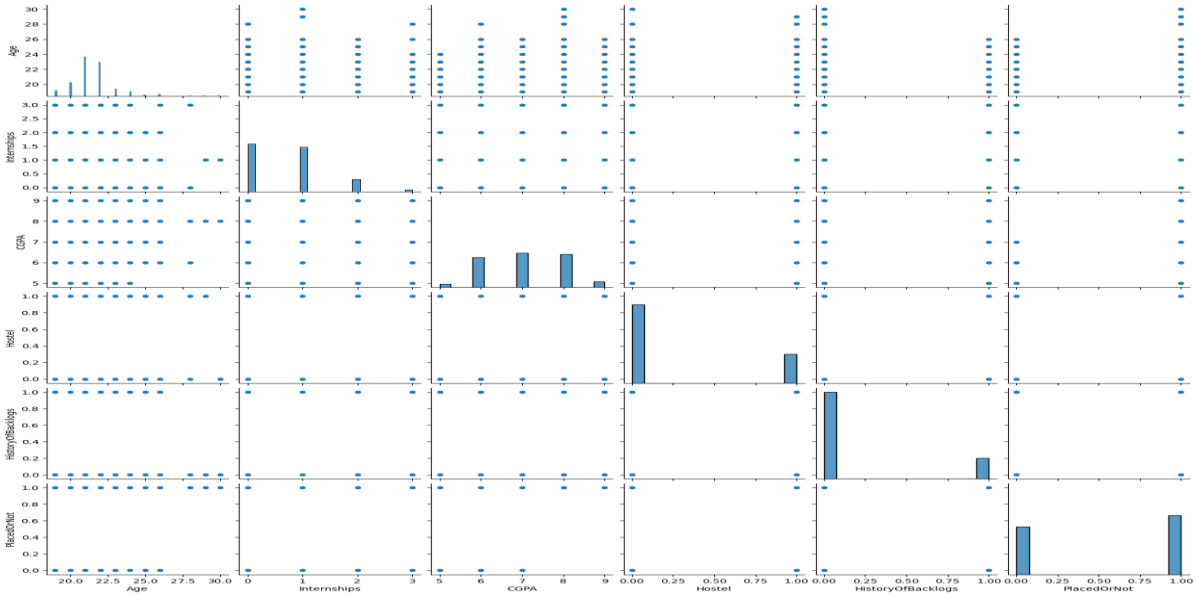
Result 1:

- Import all the tools we need.
- All needed tools import successful.

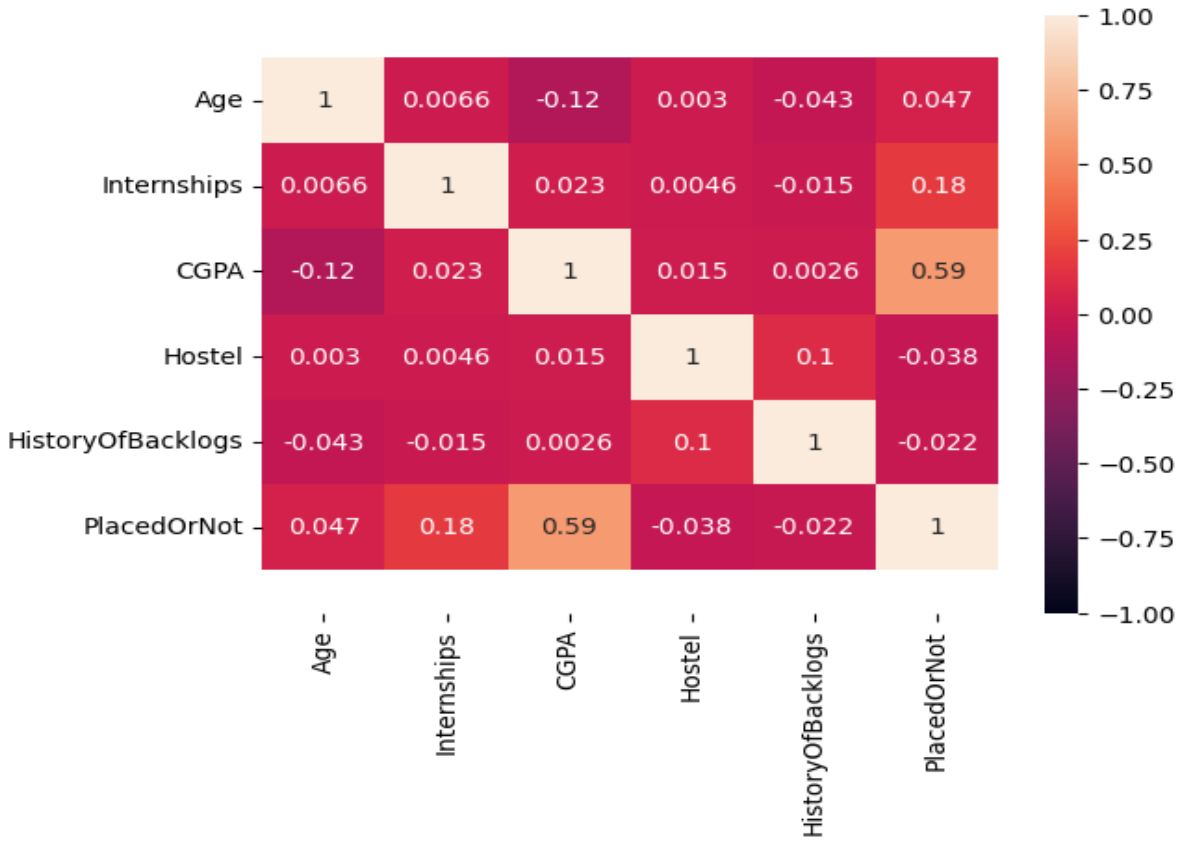
Result 2:

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	Male	Electronics And Communication	1	8	1	1	1
1	21	Female	Computer Science	0	7	1	1	1
2	22	Female	Information Technology	1	6	0	0	1
3	21	Male	Information Technology	0	8	0	1	1
4	22	Male	Mechanical	0	8	1	0	1

Result 3:



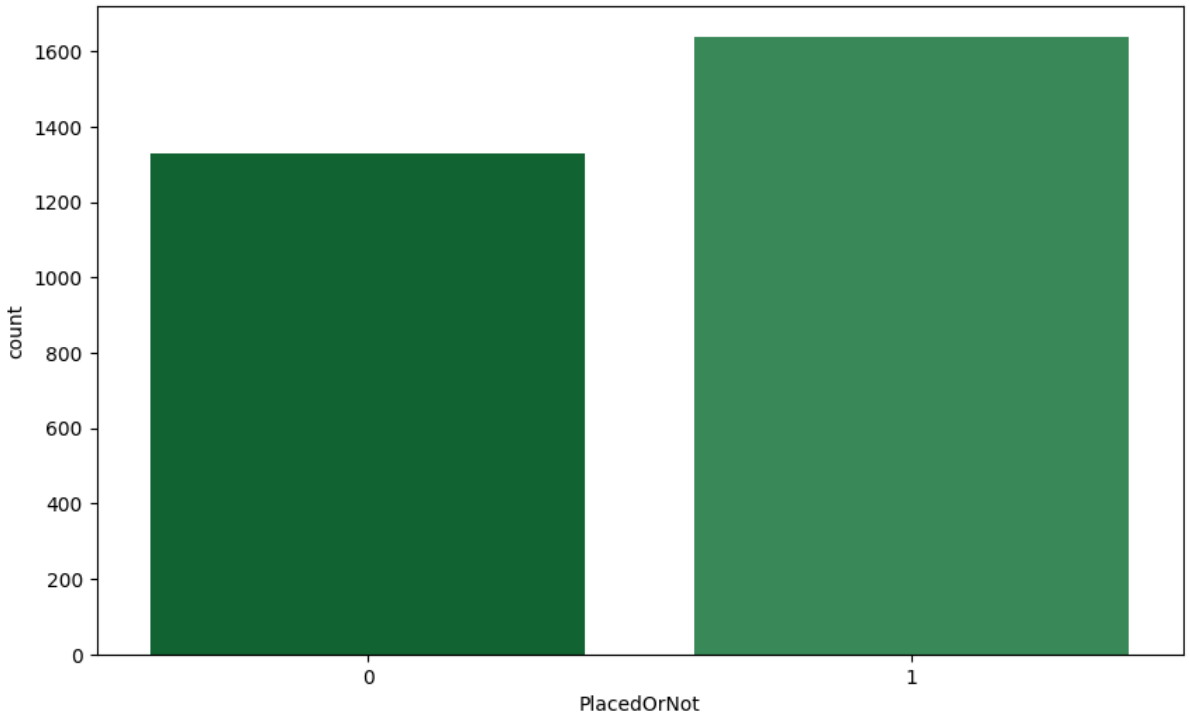
Result 4:



Result 5:

	Age	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
Age	1.000000	0.006552	-0.119787	0.003042	-0.042586	0.046943
Internships	0.006552	1.000000	0.023496	0.004617	-0.015118	0.179334
CGPA	-0.119787	0.023496	1.000000	0.014991	0.002576	0.588648
Hostel	0.003042	0.004617	0.014991	1.000000	0.103506	-0.038182
HistoryOfBacklogs	-0.042586	-0.015118	0.002576	0.103506	1.000000	-0.022337
PlacedOrNot	0.046943	0.179334	0.588648	-0.038182	-0.022337	1.000000

Result 6:



Result 7:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2966 entries, 0 to 2965
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Age                2966 non-null   int64
 1   Gender             2966 non-null   object
 2   Stream             2966 non-null   object
 3   Internships        2966 non-null   int64
 4   CGPA               2966 non-null   int64
 5   Hostel             2966 non-null   int64
 6   HistoryOfBacklogs  2966 non-null   int64
 7   PlacedOrNot        2966 non-null   int64
dtypes: int64(6), object(2)
memory usage: 185.5+ KB
```

Result 8:

```
Age                 0
Gender              0
Stream              0
Internships         0
CGPA                0
Hostel              0
HistoryOfBacklogs   0
PlacedOrNot         0
dtype: int64
```

Result 9:

	Age	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
count	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000
mean	21.485840	0.703641	7.073837	0.269049	0.192178	0.552596
std	1.324933	0.740197	0.967748	0.443540	0.394079	0.497310
min	19.000000	0.000000	5.000000	0.000000	0.000000	0.000000
25%	21.000000	0.000000	6.000000	0.000000	0.000000	0.000000
50%	21.000000	1.000000	7.000000	0.000000	0.000000	1.000000
75%	22.000000	1.000000	8.000000	1.000000	0.000000	1.000000
max	30.000000	3.000000	9.000000	1.000000	1.000000	1.000000

Result 10:

```
Male      2475
Female     491
Name: Gender, dtype: int64
```

```
[ ] df['Stream'].value_counts()
```

```
Computer Science                776
Information Technology          691
Electronics And Communication   424
Mechanical                      424
Electrical                      334
Civil                           317
Name: Stream, dtype: int64
```

Result 11:

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	0	2	1	8	1	1	1
1	21	0	0	0	7	1	1	1
2	22	0	1	1	6	0	0	1
3	21	0	1	0	8	0	1	1
4	22	0	3	0	8	1	0	1
...	...	...	...	...	...	...	...	...
2961	23	0	1	0	7	0	0	0
2962	23	0	3	1	7	1	0	0
2963	22	0	1	1	7	0	0	0
2964	22	0	0	1	7	0	0	0
2965	23	0	5	0	8	0	0	1

2966 rows × 8 columns

Result 12:

```
[ ] <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 2966 entries, 0 to 2965
    Data columns (total 8 columns):
     #   Column             Non-Null Count  Dtype
    ---  ------             --------------  -----
     0   Age                2966 non-null   int64
     1   Gender             2966 non-null   int64
     2   Stream             2966 non-null   int64
     3   Internships        2966 non-null   int64
     4   CGPA               2966 non-null   int64
     5   Hostel             2966 non-null   int64
     6   HistoryOfBacklogs  2966 non-null   int64
     7   PlacedOrNot        2966 non-null   int64
    dtypes: int64(8)
    memory usage: 185.5 KB
```

Result 13:

	Age	Gender	Stream	Internships	CGPA	HistoryOfBacklogs	PlacedOrNot
0	22	0	2	1	8	1	1
1	21	0	0	0	7	1	1
2	22	0	1	1	6	0	1
3	21	0	1	0	8	1	1
4	22	0	3	0	8	0	1
...	...	...	...	...	...	...	...
2961	23	0	1	0	7	0	0
2962	23	0	3	1	7	0	0
2963	22	0	1	1	7	0	0
2964	22	0	0	1	7	0	0
2965	23	0	5	0	8	0	1

2966 rows × 7 columns

Result 14:

```
      Age  Gender  Stream  Internships  CGPA  HistoryOfBacklogs
0      22       0       2            1     8                  1
1      21       0       0            0     7                  1
2      22       0       1            1     6                  0
3      21       0       1            0     8                  1
4      22       0       3            0     8                  0
...   ...     ...     ...          ...   ...                ...
2961   23       0       1            0     7                  0
2962   23       0       3            1     7                  0
2963   22       0       1            1     7                  0
2964   22       0       0            1     7                  0
2965   23       0       5            0     8                  0

[2966 rows x 6 columns]
```

Result 15:

```
0       1
1       1
2       1
3       1
4       1
       ..
2961    0
2962    0
2963    0
2964    0
2965    1
Name: PlacedOrNot, Length: 2966, dtype: int64
```

Result 16:

```
[[ 0.38813058  0.          0.04008175  0.40044544  0.95719068  2.05024603]
 [-0.36675158  0.         -1.14874288 -0.95077319 -0.07631043  2.05024603]
 [ 0.38813058  0.         -0.55433057  0.40044544 -1.10981154 -0.48774634]
 ...
 [ 0.38813058  0.         -0.55433057  0.40044544 -0.07631043 -0.48774634]
 [ 0.38813058  0.         -1.14874288  0.40044544 -0.07631043 -0.48774634]
 [ 1.14301273  0.          1.82331869 -0.95077319  0.95719068 -0.48774634]]
```

Result 17:

```
[ ] array([[-0.36675158,  0.        ,  0.04008175,  1.75166407, -0.07631043,
            -0.48774634],
           [ 0.38813058,  0.        ,  0.63449406, -0.95077319, -0.07631043,
            -0.48774634],
           [ 1.89789488,  0.        , -0.55433057,  0.40044544, -0.07631043,
            -0.48774634],
           ...,
           [-1.12163373,  0.        ,  0.63449406,  0.40044544,  1.99069179,
            -0.48774634],
           [ 0.38813058,  0.        , -1.14874288,  0.40044544, -1.10981154,
            -0.48774634],
           [ 0.38813058,  0.        , -0.55433057,  0.40044544,  0.95719068,
            -0.48774634]])
```

Result 18:

```
array([1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0,
       1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1,
       1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
       0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1,
       0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1,
       0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1])
```

Result 19:

```
Epoch 1/100
119/119 [==============================] - 1s 2ms/step - loss: 0.7145 - accuracy: 0.5502
Epoch 2/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6897 - accuracy: 0.5729
Epoch 3/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6736 - accuracy: 0.5700
Epoch 4/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6652 - accuracy: 0.5890
Epoch 5/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6576 - accuracy: 0.6096
Epoch 6/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6518 - accuracy: 0.6067
Epoch 7/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6289 - accuracy: 0.6172
Epoch 8/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6256 - accuracy: 0.6391
Epoch 9/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6223 - accuracy: 0.6383
Epoch 10/100
119/119 [==============================] - 0s 2ms/step - loss: 0.6034 - accuracy: 0.6606
Epoch 11/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5996 - accuracy: 0.6661
Epoch 12/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5920 - accuracy: 0.6644
Epoch 13/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5771 - accuracy: 0.6889
Epoch 14/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5798 - accuracy: 0.6889
Epoch 15/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5797 - accuracy: 0.6762
Epoch 16/100
119/119 [==============================] - 0s 2ms/step - loss: 0.5613 - accuracy: 0.7049
```

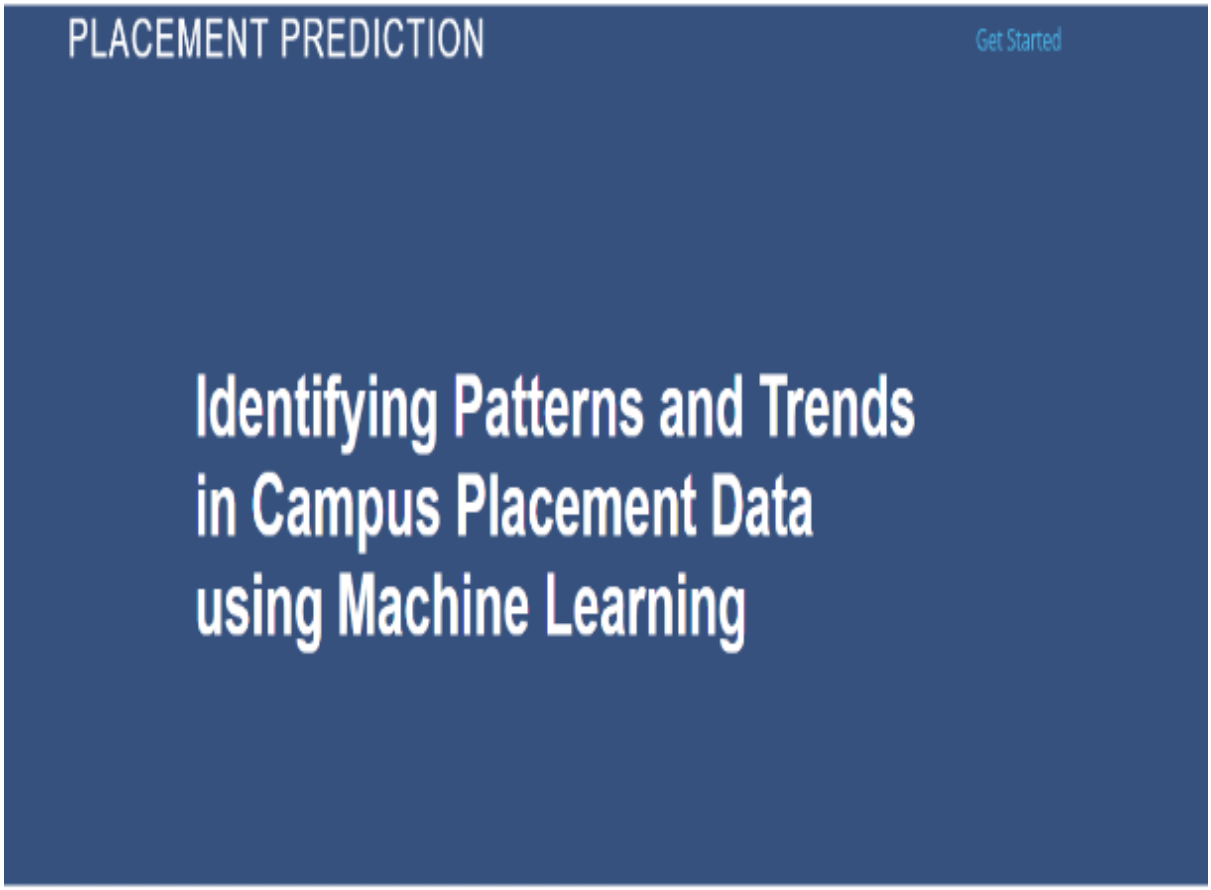
Result 20:

```
19/19 [==============================] - 0s 2ms/step
array([[ True],
       [False],
       [False],
       [ True],
       [False],
       [ True],
       [ True],
       [False],
       [ True],
       [ True],
       [False],
       [False],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [False],
       [False],
       [ True],
       [ True],
       [ True],
       [ True],
       [False],
       [False],
       [ True],
       [False],
       [False],
       [ True],
```

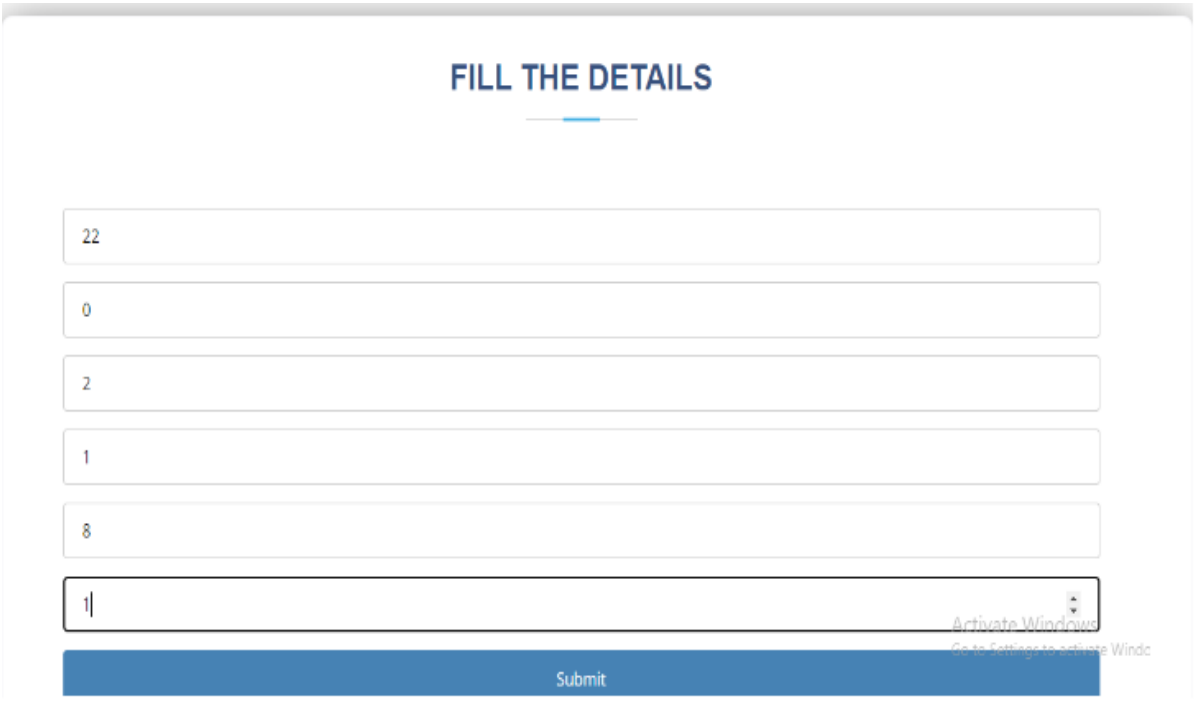
Result 21:

```
array([[257,   9],
       [ 92, 236]])
```

Result 22:



Result 23:



Result 24:

CHAPTER 4

4. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Identifying patterns and trends in campus placements using machine learning can offer several advantages, such as

- Improved decision-making: By analysing historical data of campus placements, machine learning algorithms can identify patterns and trends that can help decision-makers make informed decisions. For example, they can identify which departments or courses have the highest placement rates or which companies are more likely to hire from a particular campus.

- Better resource allocation: By understanding the patterns and trends in campus placements, institutions can allocate their resources more effectively. For example, they can focus on improving the skills of students in departments or courses that have low placement rates, or they can tailor their recruitment efforts to target companies that have historically hired more graduates from their institution.

- Personalized career counselling: Machine learning algorithms can analyse individual student data and provide personalized career counselling based on their strengths, weaknesses, and interests. This can help students make informed decisions about their future careers and improve their chances of getting placed in their desired companies.

- Early identification of trends: By analysing historical data, machine learning algorithms can identify emerging trends in the job market and adjust their placement strategies accordingly. For example, if the data indicates that a particular industry is growing rapidly, institutions can focus on developing programs that prepare students for careers in that industry.

- Increased competitiveness: Institutions that use machine learning to analyse campus placement data can gain a competitive advantage over

those that do not. They can use the insights gained from their analysis to improve their placement rates and attract more students and companies to their campus.

DISADVANTAGES:

- While identifying patterns and trends in campus placement using machine learning has several advantages, there are also some potential disadvantages to consider, including

- Biased data: Machine learning algorithms can only identify patterns and trends based on the data they are trained on. If the data is biased or incomplete, it can lead to inaccurate or incomplete insights. For example, if the historical placement data does not represent the diversity of the student population, the algorithm may overlook opportunities for underrepresented groups.

- Privacy concerns: Machine learning algorithms require access to large amounts of data, including sensitive information about students and companies. If this data is not properly protected, it could lead to privacy violations and potential legal issues.

- Lack of transparency: Machine learning algorithms are often complex and difficult to interpret, making it challenging to understand how they arrive at their conclusions. This lack of transparency can make it difficult to identify errors or biases in the analysis.

- Limited scope: Machine learning algorithms can only analyse the data they are trained on, which may not capture all of the relevant factors that influence campus placement. For example, the algorithm may overlook the impact of individual student performance or the current state of the job market.

- Implementation costs: Implementing machine learning algorithms can be costly and require specialized expertise. Institutions may need to invest in

new infrastructure, software, and training to properly use and maintain these systems.

CHAPTER 5

5.APPLICATIONS

The application of identifying patterns and trends in campus placement using machine learning can be done in several ways, such as

1. Predictive analysis: Machine learning algorithms can be used to analyse historical campus placement data and predict the likelihood of a student getting placed in a particular company or industry. This can help institutions tailor their placement strategies to better match the interests and skills of their students.

2. Personalized career counselling: By analysing individual student data, machine learning algorithms can provide personalized career counselling and guidance to students. This can help students make informed decisions about their future careers and improve their chances of getting placed in their desired companies.

3. Resource allocation: By understanding the patterns and trends in campus placements, institutions can allocate their resources more effectively. For example, they can focus on improving the skills of students in departments or courses that have low placement rates, or they can tailor their recruitment efforts to target companies that have historically hired more graduates from their institution.

4. Early identification of trends: Machine learning algorithms can analyse job market trends and adjust placement strategies accordingly. For example, if the data indicates that a particular industry is growing rapidly, institutions can focus on developing programs that prepare students for careers in that industry.

5. Competitive advantage: By using machine learning to analyse campus placement data, institutions can gain a competitive advantage over others. They can use the insights gained from their analysis to improve their placement rates and attract more students and companies to their campus. Overall, the application of identifying patterns and trends in campus placement using machine learning can help institutions make data-driven decisions, improve student outcomes, and stay competitive in a rapidly changing job market.

CHAPTER 6

6.CONCLUSION

In conclusion, identifying patterns and trends in campus placement using machine learning can offer several advantages for institutions, students, and employers. By analyzing historical data, machine learning algorithms can provide insights that can help institutions make informed decisions about resource allocation, recruitment strategies, and program development. This can lead to improved student outcomes, increased placement rates, and a more competitive institution.

However, there are also potential disadvantages to consider, such as the risk of biased data, privacy concerns, and implementation costs. Institutions must carefully evaluate the benefits and risks of using machine learning in campus placement and take steps to mitigate potential issues.

Overall, the use of machine learning in campus placement has the potential to revolutionize how institutions approach career counseling, recruitment, and program development. As technology continues to advance, it will be important for institutions to stay up-to-date with the latest developments in order to remain competitive and meet the needs of their students and employers.

CHAPTER 7

7. FUTURE SCOPE

The future scope for identifying patterns and trends in campus placement using machine learning is vast, and it is expected to play an increasingly important role in higher education and recruitment. Some potential future applications include:

1. Improved job matching: Machine learning algorithms can be used to match students with job opportunities based on their skills, interests, and career goals. This can help students find jobs that are a better fit for their individual needs and preferences.

2. Real-time placement tracking: Institutions can use machine learning algorithms to track the progress of their students in real-time, identifying areas where they may need additional support or guidance. This can help institutions improve their placement rates and better meet the needs of their students.

3. Greater personalization: As machine learning algorithms become more sophisticated, they will be able to provide even more personalized career counseling and guidance to students. This can help students make more informed decisions about their future careers and improve their chances of success.

4. Better data integration: Institutions can use machine learning algorithms to integrate data from a variety of sources, such as student records, employer data, and job market trends. This can provide a more comprehensive view of campus placement and help institutions make more informed decisions about resource allocation and program development.

5. Greater efficiency: By automating many of the tasks associated with campus placement, machine learning can help institutions operate more efficiently and effectively. This can free up resources to focus on other important areas, such as student support and program development.

Overall, the future scope for identifying patterns and trends in campus placement using machine learning is vast, and it has the potential to transform the way that institutions approach career counseling, recruitment, and program development. As technology continues to evolve, it will be important for institutions to stay up-to-date with the latest developments in order to remain competitive and meet the needs of their students and employers.

CHAPTER 8

8.APPENDIX

8.1 SOURCE CODE

```
from sklearn import svm
from sklearn.metrics import accuracy_import numpy as np
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt
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('/content/collegePlace.csv')
```

```
df.head()

df.shape

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

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')

from seaborn.widgets import color_palette
plt.figure(figsize=(10,6),dpi=100)

color_palette = sns.color_palette("BuGn_r")
sns.set_palette(color_palette)
```

```
sns.countplot(x = "PlacedOrNot",data=df)

plt.show

df.info()

df.isnull().sum()

df.describe()

df['Gender'].value_counts()

df['Stream'].value_counts()

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

df.info()

def transformationplot(feature):
  plt.figure(figsize=(12,5))
  plt.subplot(1,2,1)
  sns.displot(feature)
```

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

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

X = df.drop(columns = 'PlacedOrNot',axis=1)
Y = df['PlacedOrNot']

import joblib
joblib.dump(X,'Placement')

print(X)

print(Y)

scalar = StandardScaler()
scalar.fit(X)

standardized_data = scalar.transform(X)
print(standardized_data)

X = standardized_data
Y = df['PlacedOrNot']

"""# **Splitting the data**"""
```

```
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)


"""# **Model 1**"""


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

classifier.fit(X_train,Y_train)


#testing accurancy

X_test_prediction = classifier.predict(X_test)

Y_pred = accuracy_score(X_test_prediction,Y_test)

Y_pred


X_test


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)


"""# **Model2 (KNN)**"""


from sklearn.neighbors import KNeighborsClassifier

from sklearn import metrics

from sklearn.model_selection import cross_val_score


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)
   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: {}\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)

knn_pred

"""# **Model3(ANN)**"""

X_train.shape

Y_train.shape

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', input_dim= 6))
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'))

#Compiling the model

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

"""# **saving and downloading**"""

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):
  print ('not placed')
else:
  print('placed')
```