ARIGNAR ANNA GOVERNMENT ARTS COLLEGE

VILLUPURAM – 605 402.

DEPARTMENT OF COMPUTER SCIENCE

PROJECT REPORT

Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Team ID	–	NM2023TMID16927
Team Leader	–	MURUGESAN S
Team members	–	MADHANKUMAR S MANIBALAN S MOHAMED RIYAS R

1. Introduction

1.1 Overview

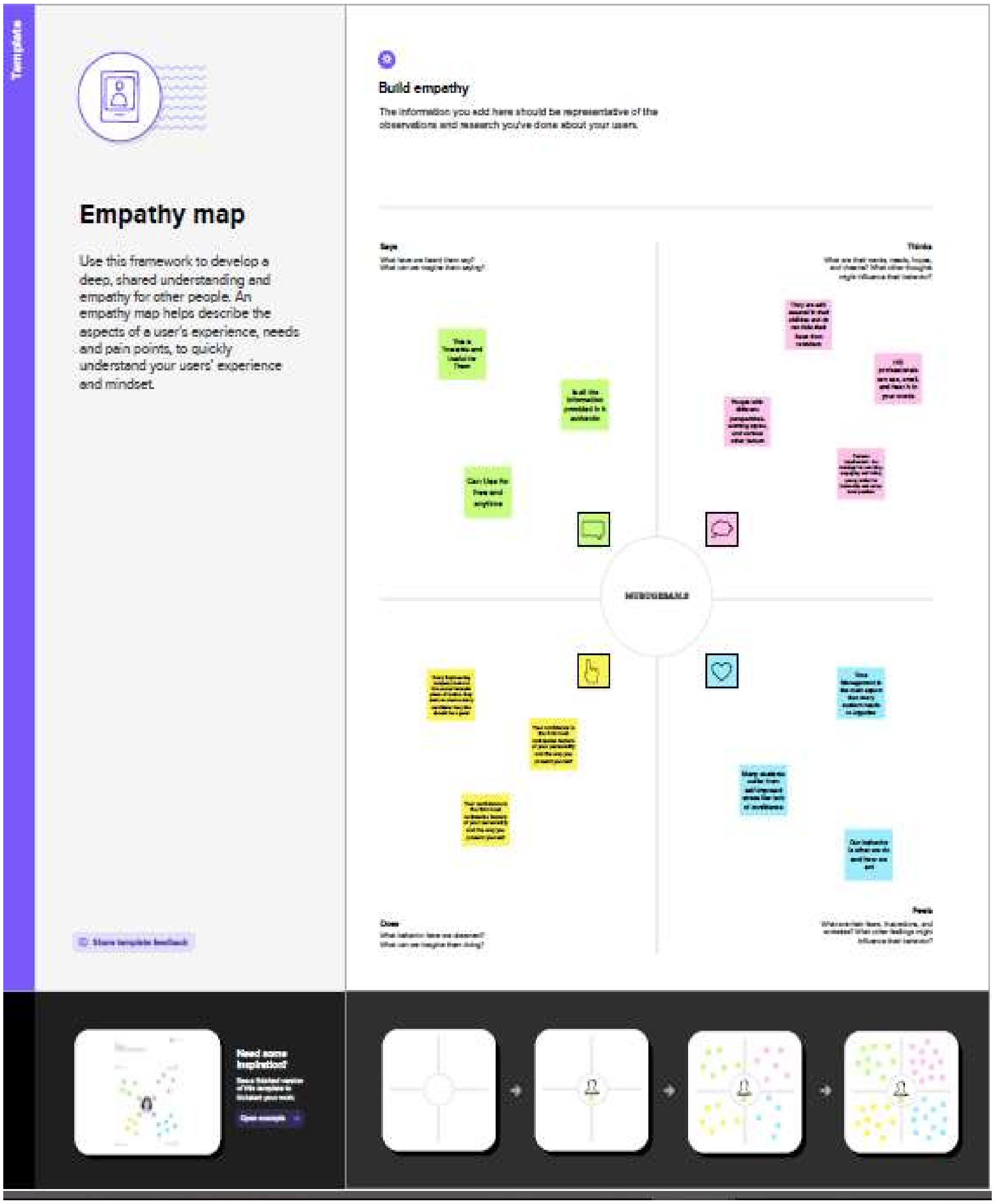
The project aims to analyse and identify patterns and trends in campus placement data using machine learning techniques. The project analyses data collected from campus placement drives conducted by various colleges and universities. The project uses machine learning algorithms to identify patterns and trends in the data that can help in predicting the placement status of future candidates.

1.2 Purpose

The purpose of the project is to provide a tool for colleges and universities to analyse and understand the placement data of their students. The project helps in identifying the factors that affect the placement status of candidates and provides insights into the placement process. The project also helps in predicting the placement status of future candidates, which can be useful for students, colleges, and recruiters.

2. Problem Definition & Design Thinking

2.1 Empathy Map



Template

Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Share template feedback

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says

What have we heard them say? What can we imagine them saying?

This is Traceable and Useful for Them

Is all the information provided in a website

Can Use for free and anytime

Thinks

What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

[illegible]

Does

What behavior have we observed? What can we imagine them doing?

[illegible]

Feels

What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

[illegible]

Need some inspiration?

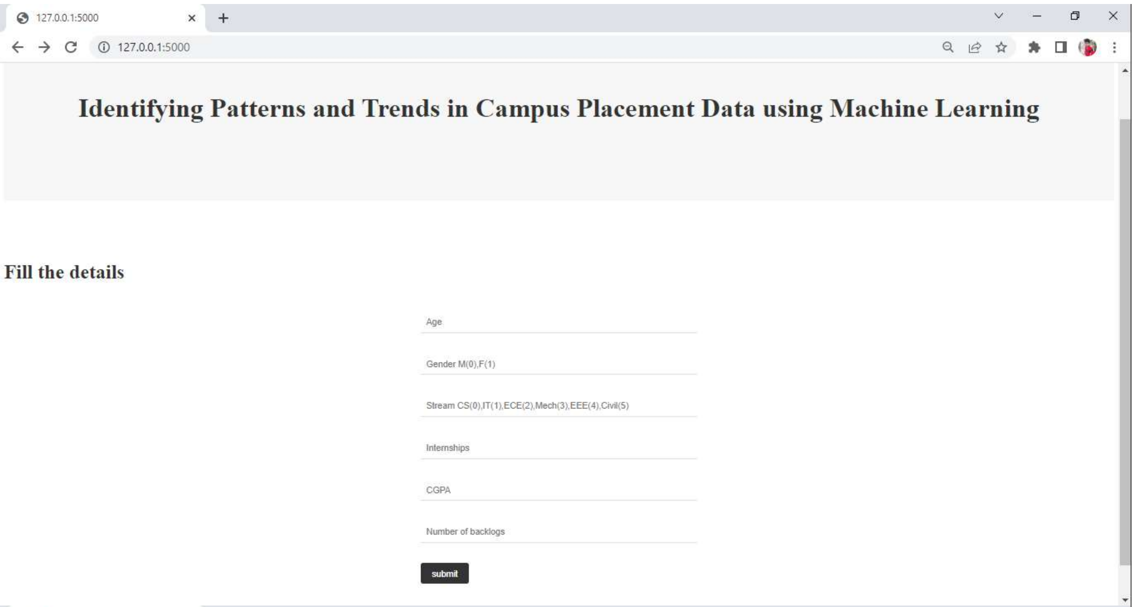
See a finished version of this template to kickstart your work.

Open example

2.2 Ideation & Brainstorming Map

3. Result

The final output of the project includes a machine learning model that can predict the placement status of future candidates based on the analysis of the campus placement data. The project also provides various visualizations and insights into the placement data that can help in understanding the placement process better.



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

Age

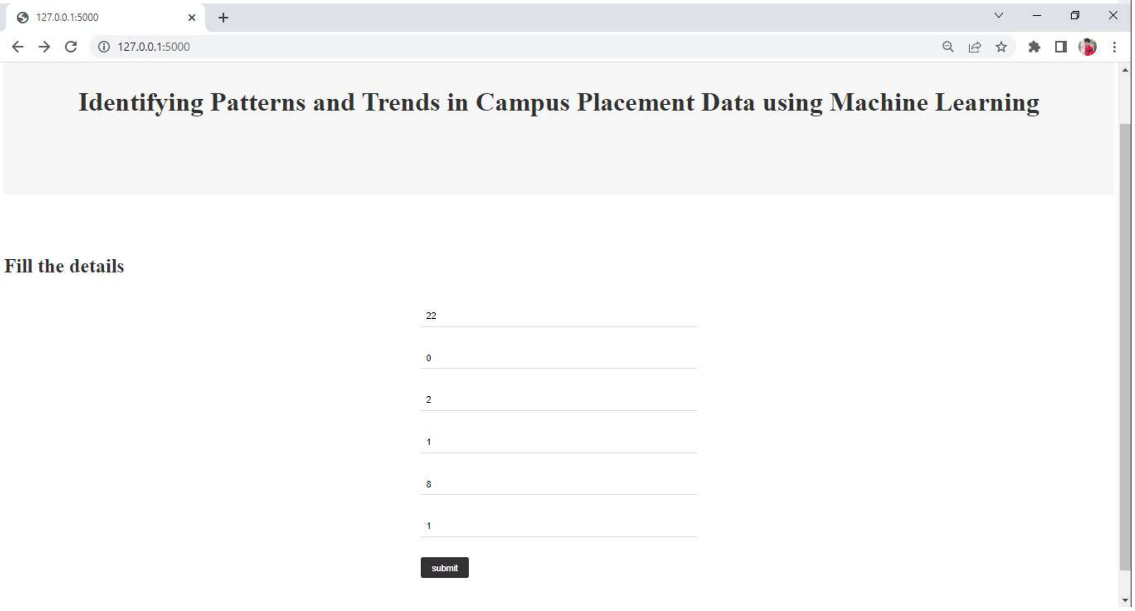
Gender M(0),F(1)

Stream CS(0),IT(1),ECE(2),Mech(3),EEE(4),Civil(5)

Internships

CGPA

Number of backlogs



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

22

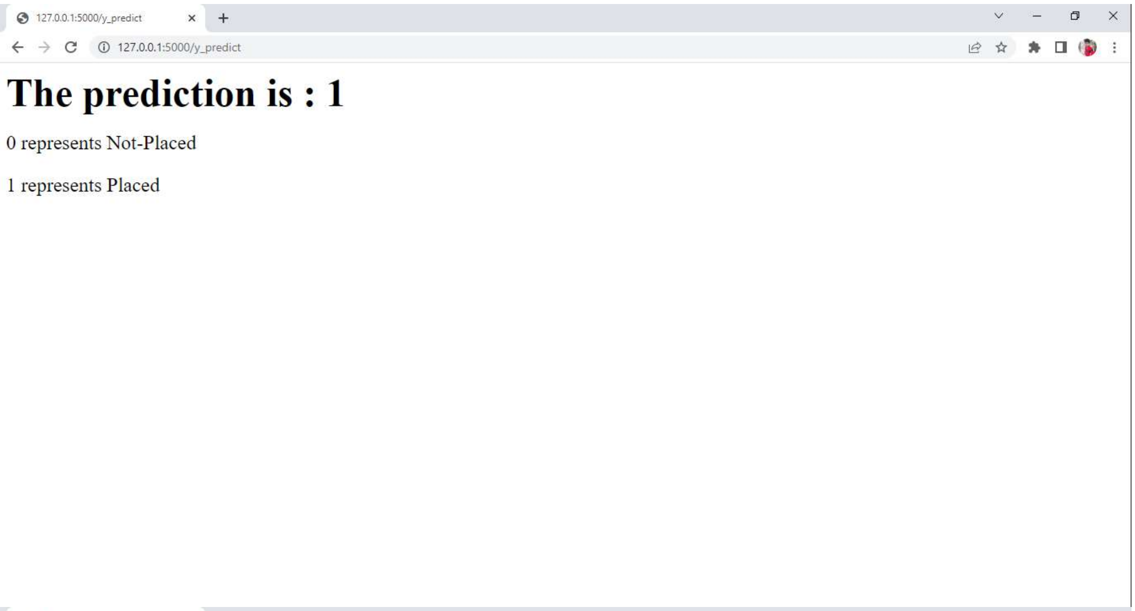
0

2

1

8

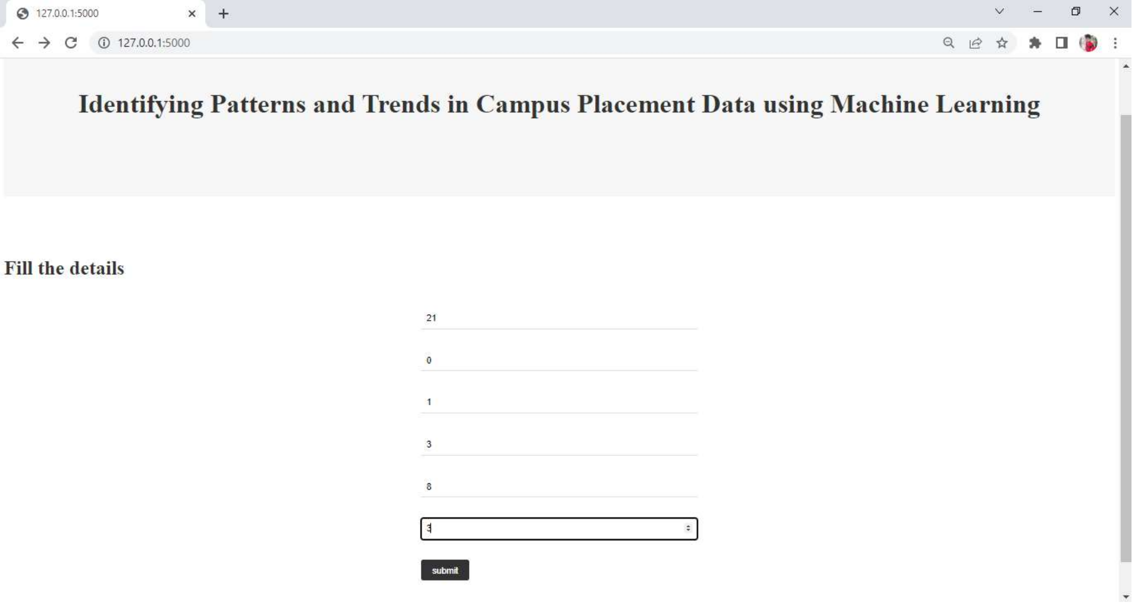
1



The prediction is : 1

0 represents Not-Placed

1 represents Placed



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

21

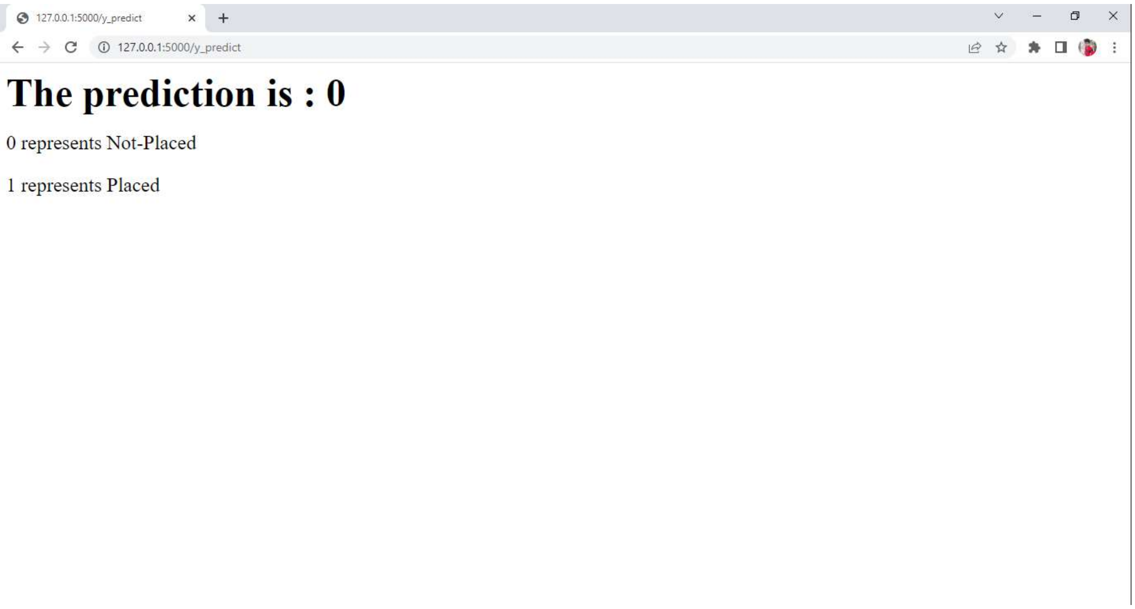
0

1

3

8

3



The prediction is : 0

0 represents Not-Placed

1 represents Placed

4. Advantages & Disadvantages

4.1 Advantages

* Provides insights into the placement process

* Helps in identifying factors that affect the placement status of candidates

* Predicts the placement status of future candidates

* Helps in making data-driven decisions

4.2 Disadvantages

* The accuracy of the machine learning model depends on the quality of the data

* The project may not be able to capture all the factors that affect the placement status of candidates

5. APPLICATIONS

The project can be applied in various domains, including:

* Educational institutions: To analyse the placement data of their students and improve the placement process.

* Recruiters: To predict the placement status of future candidates and make data-driven hiring decisions.

*Campus placement data can be used to identify skills gaps in the workforce.

*Campus placement data can provide insights into industry trends and changes in the job market.

6. CONCLUSION

The project provides a tool for analysing and understanding the campus placement data using machine learning techniques. The project helps in identifying patterns and trends in the data that can provide valuable insights into the placement process. The project also provides a machine learning model that can predict the placement status of future candidates.

7. FUTURE SCOPE

The project can be further improved by:

* Adding more data sources to improve the accuracy of the machine learning model

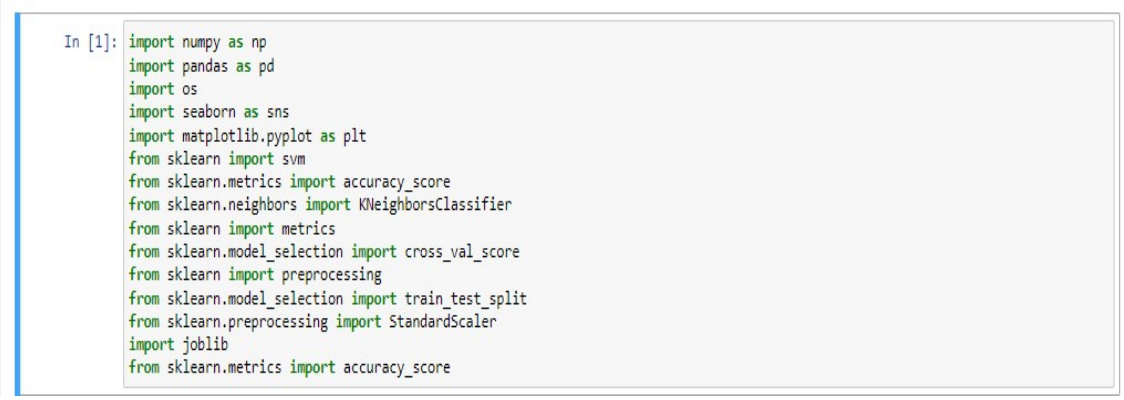
* Using advanced machine learning algorithms to improve the accuracy of the predictions

8. APPENDIX

A. Source Code

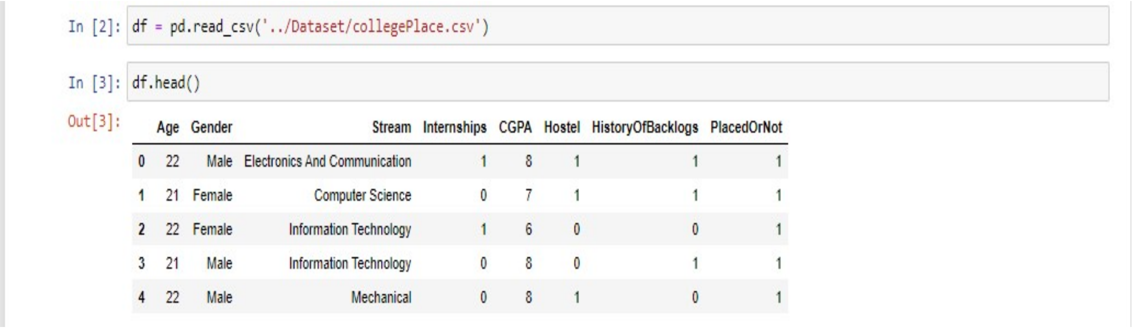
Milestone 2: Data Collection & Preparation

Importing the libraries:



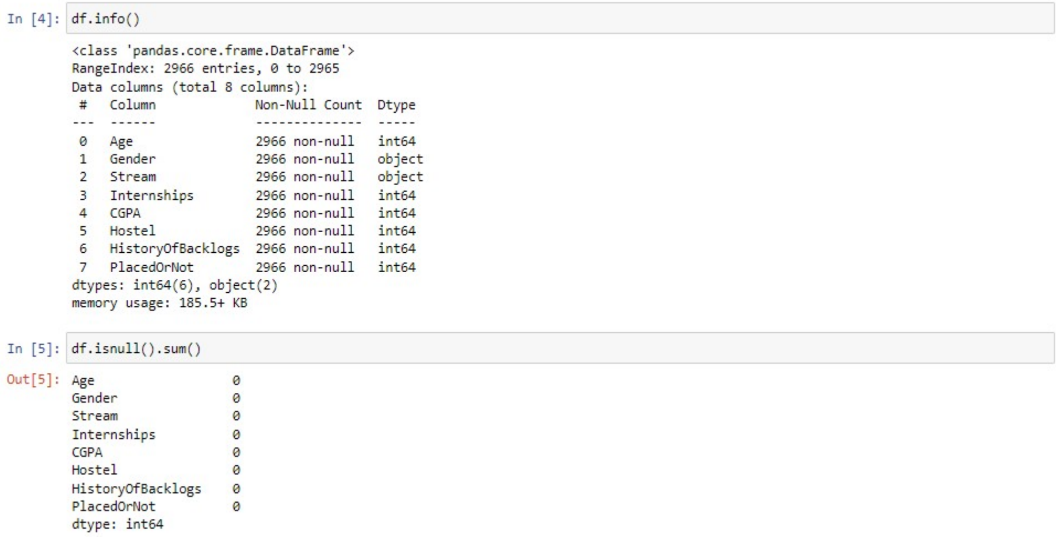
```
In [1]: import numpy as np
        import pandas as pd
        import os
        import seaborn as sns
        import matplotlib.pyplot as plt
        from sklearn import svm
        from sklearn.metrics import accuracy_score
        from sklearn.neighbors import KNeighborsClassifier
        from 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:



```
In [2]: df = pd.read_csv('../Dataset/collegePlace.csv')

In [3]: df.head()

Out[3]:
      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
```

Handling missing values:



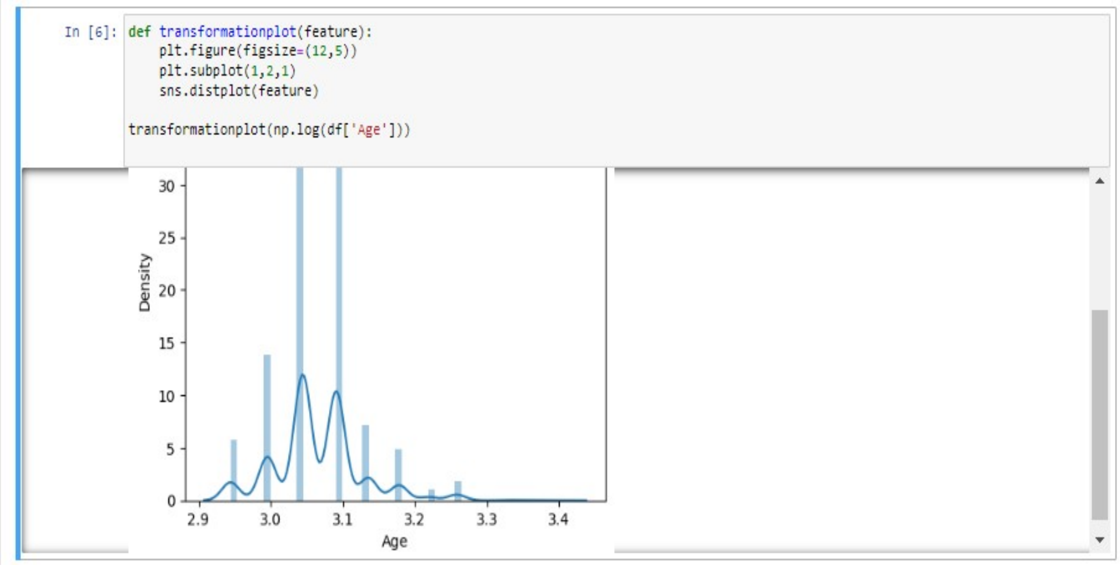
In [4]:
```
df.info()
```

```
<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
```

In [5]:
```
df.isnull().sum()
```

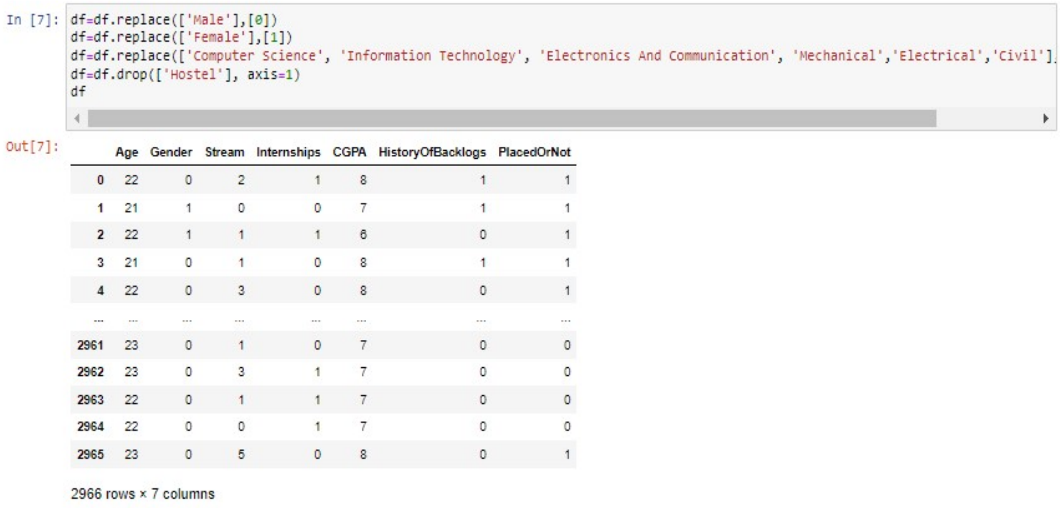
Out[5]:
```
Age                  0
Gender               0
Stream               0
Internships          0
CGPA                 0
Hostel               0
HistoryOfBacklogs    0
PlacedOrNot          0
dtype: int64
```

Handling outliers:



In [6]:
```
def transformationplot(feature):
    plt.figure(figsize=(12,5))
    plt.subplot(1,2,1)
    sns.distplot(feature)

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

Handling Categorical Values:



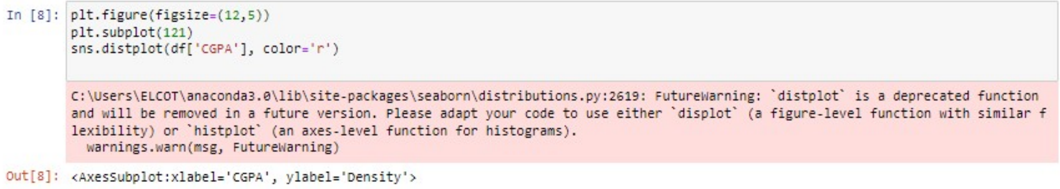
```
In [7]: df=df.replace(['Male'],[0])
        df=df.replace(['Female'],[1])
        df=df.replace(['Computer Science', 'Information Technology', 'Electronics And Communication', 'Mechanical','Electrical','Civil'].
        df=df.drop(['Hostel'], axis=1)
        df
```

Out[7]:

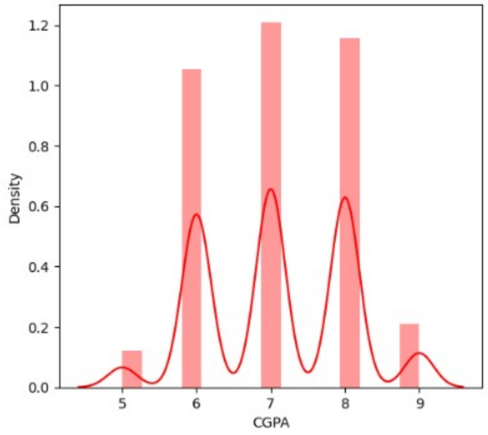
	Age	Gender	Stream	Internships	CGPA	HistoryOfBacklogs	PlacedOrNot
0	22	0	2	1	8	1	1
1	21	1	0	0	7	1	1
2	22	1	1	1	6	0	1
3	21	0	1	0	8	1	1
4	22	0	3	0	8	0	1
...	...	...	...	...	...	...	...
2961	23	0	1	0	7	0	0
2962	23	0	3	1	7	0	0
2963	22	0	1	1	7	0	0
2964	22	0	0	1	7	0	0
2965	23	0	5	0	8	0	1

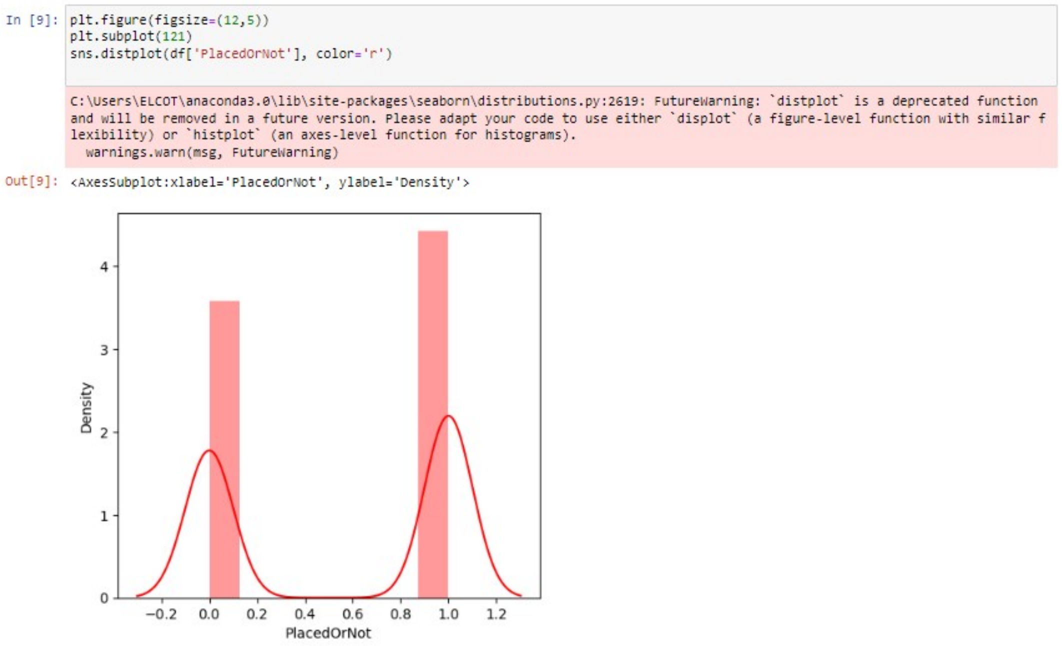
2966 rows × 7 columns

Milestone 3: Exploratory Data Analysis



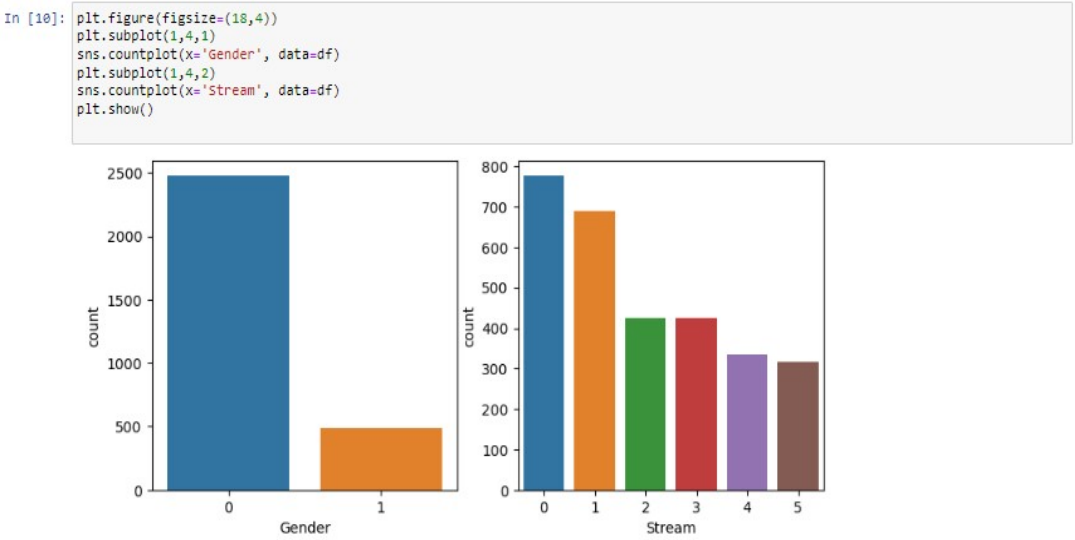
```
In [8]: plt.figure(figsize=(12,5))
        plt.subplot(121)
        sns.distplot(df['CGPA'], color='r')

        C:\Users\ELCOT\anaconda3.0\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function
        and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar f
        lexibility) or `histplot` (an axes-level function for histograms).
          warnings.warn(msg, FutureWarning)

Out[8]: <AxesSubplot:xlabel='CGPA', ylabel='Density'>
```





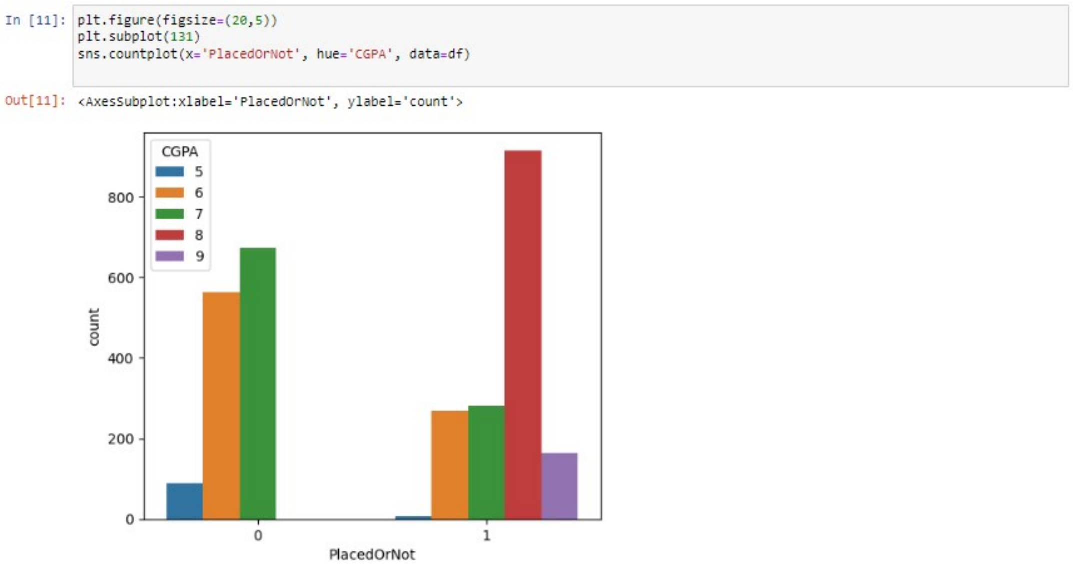
```
In [9]: plt.figure(figsize=(12,5))
        plt.subplot(121)
        sns.distplot(df['PlacedOrNot'], color='r')

        C:\Users\ELCOT\anaconda3.0\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function
        and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar f
        lexibility) or `histplot` (an axes-level function for histograms).
          warnings.warn(msg, FutureWarning)

Out[9]: <AxesSubplot:xlabel='PlacedOrNot', ylabel='Density'>
```

Bivariate analysis:

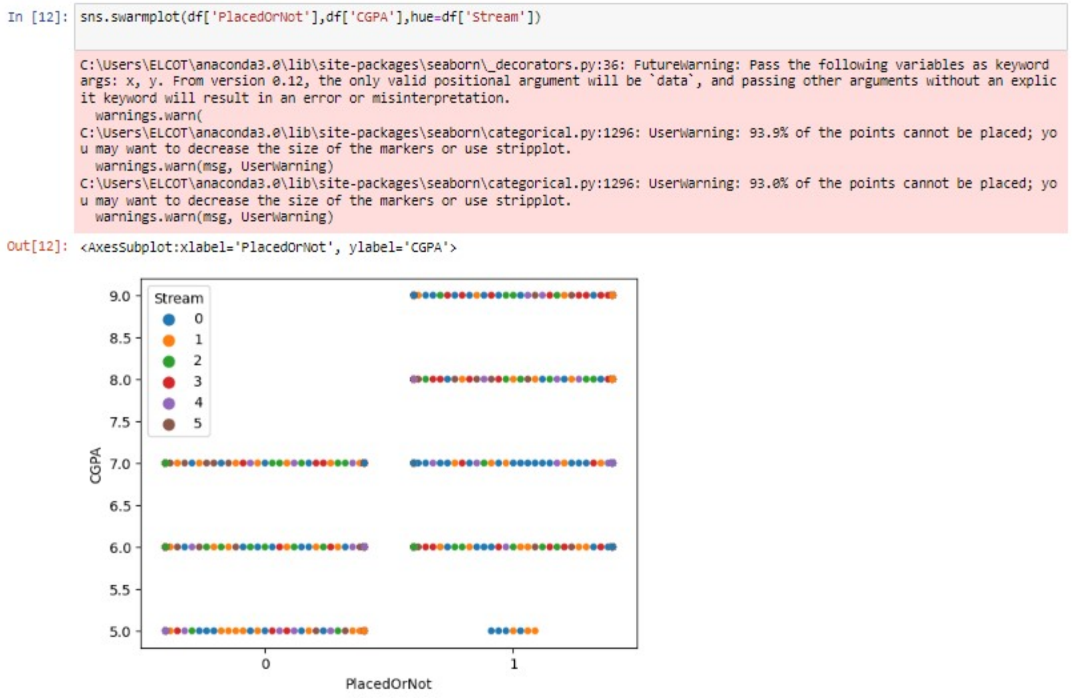


```
In [10]: plt.figure(figsize=(18,4))
         plt.subplot(1,4,1)
         sns.countplot(x='Gender', data=df)
         plt.subplot(1,4,2)
         sns.countplot(x='Stream', data=df)
         plt.show()
```

Multivariate analysis:

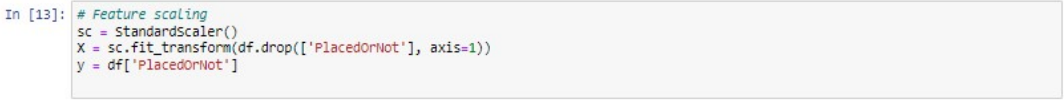


```
In [11]: plt.figure(figsize=(20,5))
         plt.subplot(131)
         sns.countplot(x='PlacedOrNot', hue='CGPA', data=df)

Out[11]: <AxesSubplot:xlabel='PlacedOrNot', ylabel='count'>
```



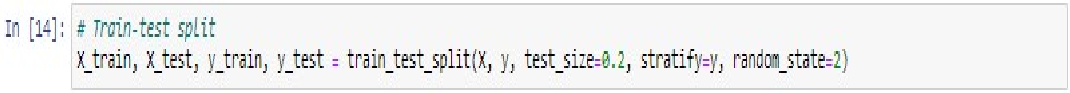
```
In [12]: sns.swarmplot(df['PlacedOrNot'],df['CGPA'],hue=df['Stream'])

         C:\Users\ELCOT\anaconda3.0\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword
         args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explic
         it keyword will result in an error or misinterpretation.
           warnings.warn(
         C:\Users\ELCOT\anaconda3.0\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 93.9% of the points cannot be placed; yo
         u may want to decrease the size of the markers or use stripplot.
           warnings.warn(msg, UserWarning)
         C:\Users\ELCOT\anaconda3.0\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 93.0% of the points cannot be placed; yo
         u may want to decrease the size of the markers or use stripplot.
           warnings.warn(msg, UserWarning)

Out[12]: <AxesSubplot:xlabel='PlacedOrNot', ylabel='CGPA'>
```

Scaling the data:



```
In [13]: # Feature scaling
         sc = StandardScaler()
         X = sc.fit_transform(df.drop(['PlacedOrNot'], axis=1))
         y = df['PlacedOrNot']
```

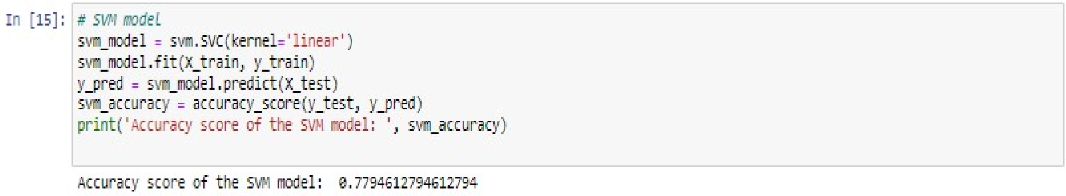
Splitting the data into train and test:



```
In [14]: # Train-test split
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=2)
```

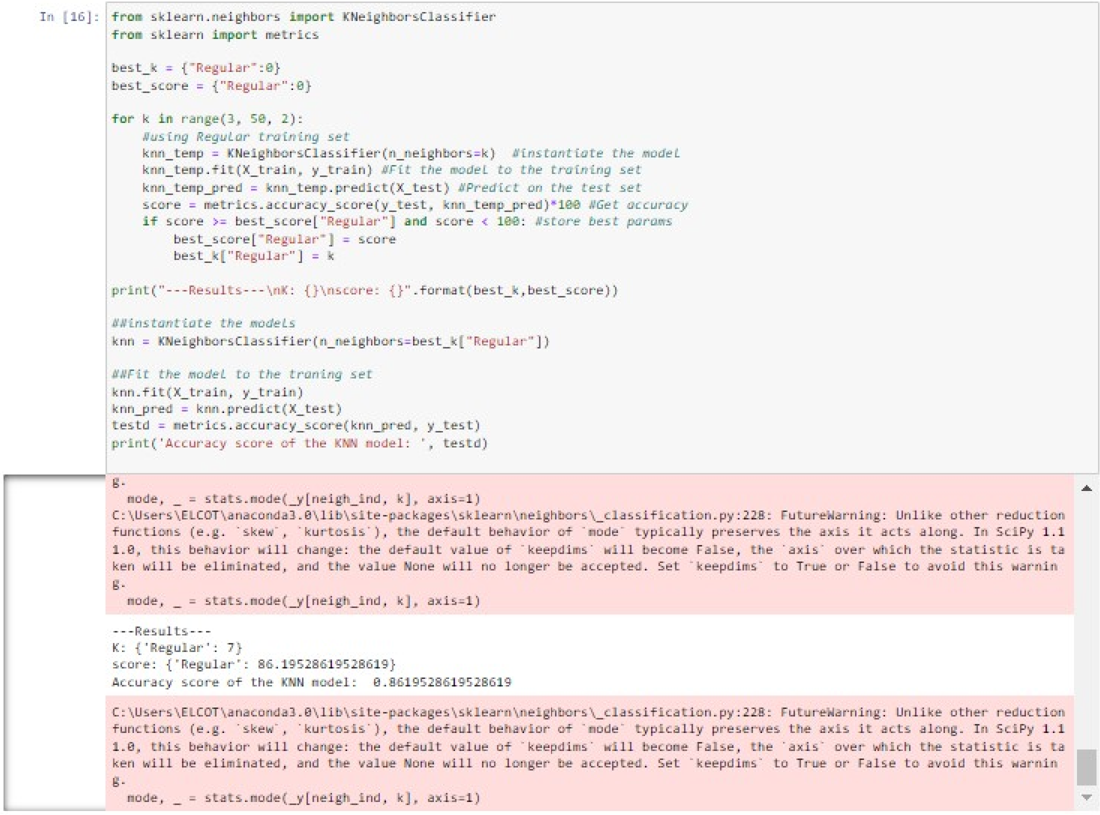
Milestone 4: Model Building

SVM model:



```
In [15]: # SVM model
         svm_model = svm.SVC(kernel='linear')
         svm_model.fit(X_train, y_train)
         y_pred = svm_model.predict(X_test)
         svm_accuracy = accuracy_score(y_test, y_pred)
         print('Accuracy score of the SVM model: ', svm_accuracy)

         Accuracy score of the SVM model:  0.7794612794612794
```

KNN model:



In [16]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

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["Regular"])

##Fit the model to the traning set
knn.fit(X_train, y_train)
knn_pred = knn.predict(X_test)
testd = metrics.accuracy_score(knn_pred, y_test)
print('Accuracy score of the KNN model: ', testd)
```

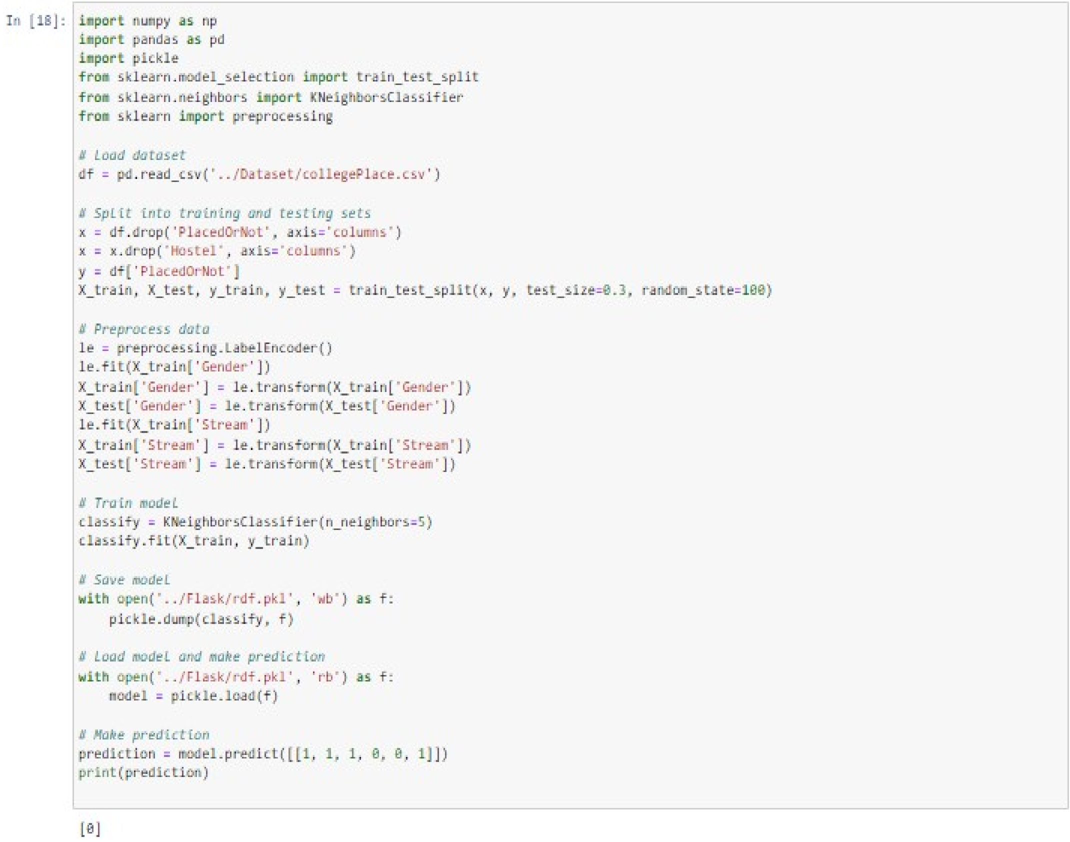
```
g.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\ELCOT\anaconda3.0\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.1
1.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is ta
ken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warnin
g.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

```
---Results---
K: {'Regular': 7}
score: {'Regular': 86.19528619528619}
Accuracy score of the KNN model:  0.8619528619528619
```

```
C:\Users\ELCOT\anaconda3.0\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.1
1.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is ta
ken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warnin
g.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Milestone 5: Model Deployment

Save the best model:



```
In [18]: import numpy as np
         import pandas as pd
         import pickle
         from sklearn.model_selection import train_test_split
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn import preprocessing

         # Load dataset
         df = pd.read_csv('../Dataset/collegePlace.csv')

         # Split into training and testing sets
         x = df.drop('PlacedOrNot', axis='columns')
         x = x.drop('Hostel', axis='columns')
         y = df['PlacedOrNot']
         X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=100)

         # Preprocess data
         le = preprocessing.LabelEncoder()
         le.fit(X_train['Gender'])
         X_train['Gender'] = le.transform(X_train['Gender'])
         X_test['Gender'] = le.transform(X_test['Gender'])
         le.fit(X_train['Stream'])
         X_train['Stream'] = le.transform(X_train['Stream'])
         X_test['Stream'] = le.transform(X_test['Stream'])

         # Train model
         classify = KNeighborsClassifier(n_neighbors=5)
         classify.fit(X_train, y_train)

         # Save model
         with open('../Flask/rdf.pkl', 'wb') as f:
             pickle.dump(classify, f)

         # Load model and make prediction
         with open('../Flask/rdf.pkl', 'rb') as f:
             model = pickle.load(f)

         # Make prediction
         prediction = model.predict([[1, 1, 1, 0, 0, 1]])
         print(prediction)

         [0]
```

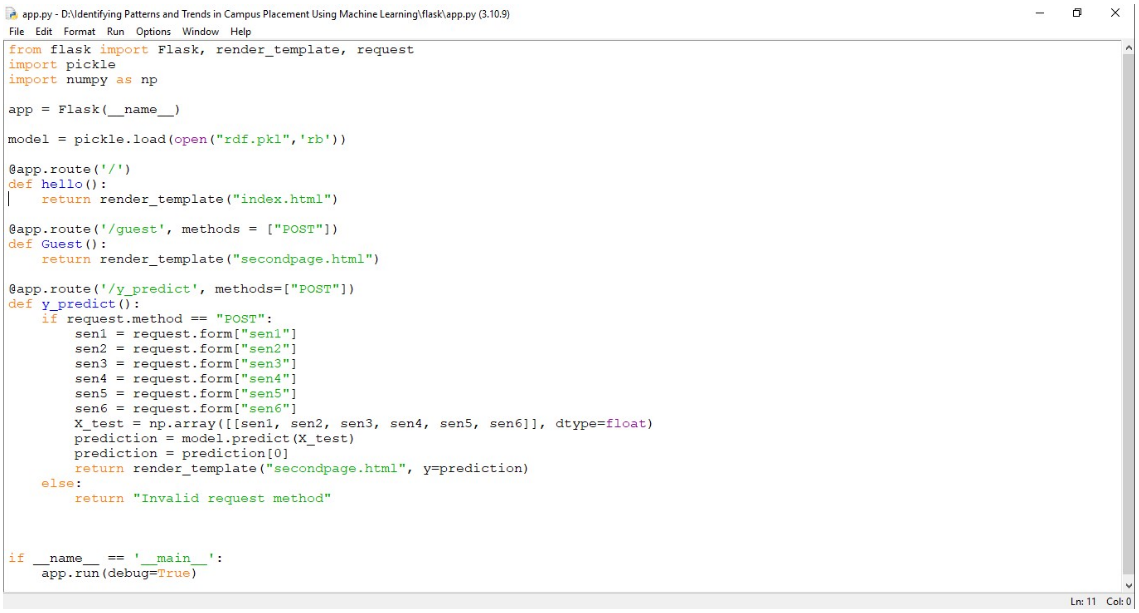
Index.html

```
<html>
    <head>
        <link rel="stylesheet" type="text/css"
href="../static/CSS/style.css">
    </head>
    <body>
<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-->

<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"><br><br>
                    <input type="number" id="sen2"
name="sen2" placeholder="Gender M(0),F(1)"><br><br>
                    <input type="number" id="sen3"
name="sen3" placeholder="Stream
CS(0),IT(1),ECE(2),Mech(3),EEE(4),Civil(5)"><br><br>
                    <input type="number" id="sen4"
name="sen4" placeholder="Internships"><br><br>
                    <input type="number" id="sen5"
name="sen5" placeholder="CGPA"><br><br>
                    <input type="number" id="sen6"
name="sen6" placeholder="Number of backlogs"><br><br>
                    <input type="submit"
value="submit">
                </form>
            </div>
        </div>
    </div>
</section><!--End about us section-->
    </body>
</html>
```

secondpage.html

```
<html>
    <head>
        <link rel="stylesheet" type="text/css"
href="../static/CSS/style2.css">
    </head>
    <body>
<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>The prediction is : {{y}}</h1>
                <h3>0 represents Not-Placed</h3>
                <h3>1 represents Placed</h3>
            </div>
        </div>
    </div>
</section><!--End Hero-->
    </body>
</html>
```

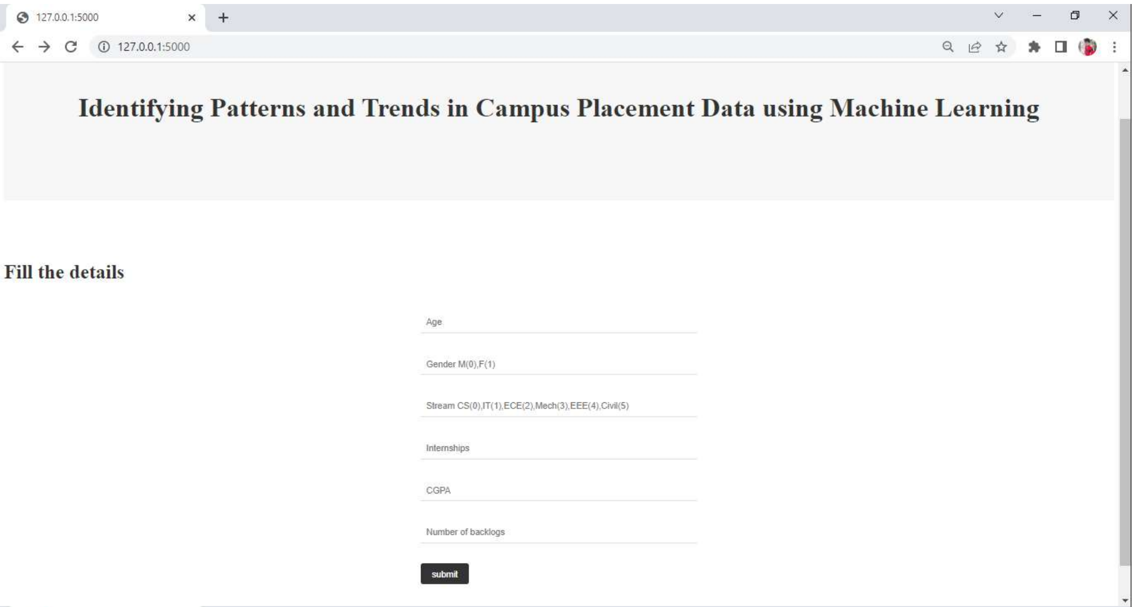
app.py


app.py - D:\Identifying Patterns and Trends in Campus Placement Using Machine Learning\flask\app.py (3.10.9)
File Edit Format Run Options Window Help
Ln: 11 Col: 0

```
from flask import Flask, render_template, request
import pickle
import numpy as np

app = Flask(__name__)

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

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

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

@app.route('/y_predict', methods=["POST"])
def y_predict():
    if request.method == "POST":
        sen1 = request.form["sen1"]
        sen2 = request.form["sen2"]
        sen3 = request.form["sen3"]
        sen4 = request.form["sen4"]
        sen5 = request.form["sen5"]
        sen6 = request.form["sen6"]
        X_test = np.array([[sen1, sen2, sen3, sen4, sen5, sen6]], dtype=float)
        prediction = model.predict(X_test)
        prediction = prediction[0]
        return render_template("secondpage.html", y=prediction)
    else:
        return "Invalid request method"



if __name__ == '__main__':
    app.run(debug=True)
```

```
from flask import Flask, render_template, request
import pickle
import numpy as np
app = Flask(__name__)
model = pickle.load(open("rdf.pkl",'rb'))
@app.route('/')
def hello():
return render_template("index.html")
@app.route('/guest', methods = ["POST"])
def Guest():
return render_template("secondpage.html")
@app.route('/y_predict', methods=["POST"])
def y_predict():
if request.method == "POST":
sen1 = request.form["sen1"]
sen2 = request.form["sen2"]
sen3 = request.form["sen3"]
sen4 = request.form["sen4"]
sen5 = request.form["sen5"]
sen6 = request.form["sen6"]
X_test = np.array([[sen1, sen2, sen3, sen4, sen5,
sen6]], dtype=float)
prediction = model.predict(X_test)
prediction = prediction[0]
return render_template("secondpage.html",
y=prediction)
else:
return "Invalid request method"
if __name__ == '__main__':
app.run(debug=True)
```

OUTPUTS :



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

Age

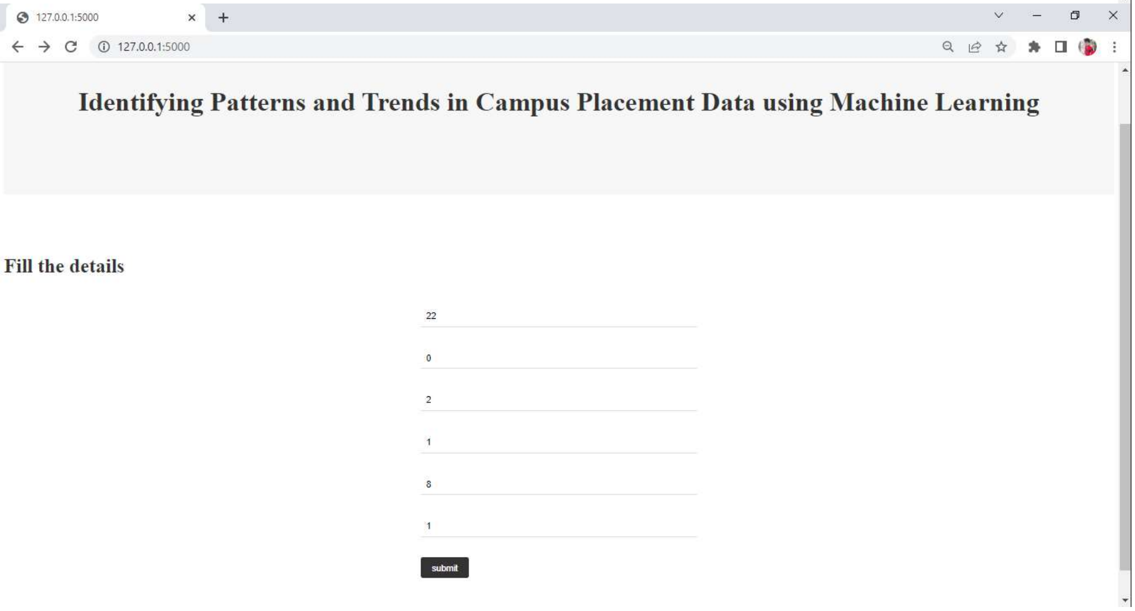
Gender M(0),F(1)

Stream CS(0),IT(1),ECE(2),Mech(3),EEE(4),Civil(5)

Internships

CGPA

Number of backlogs



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

22

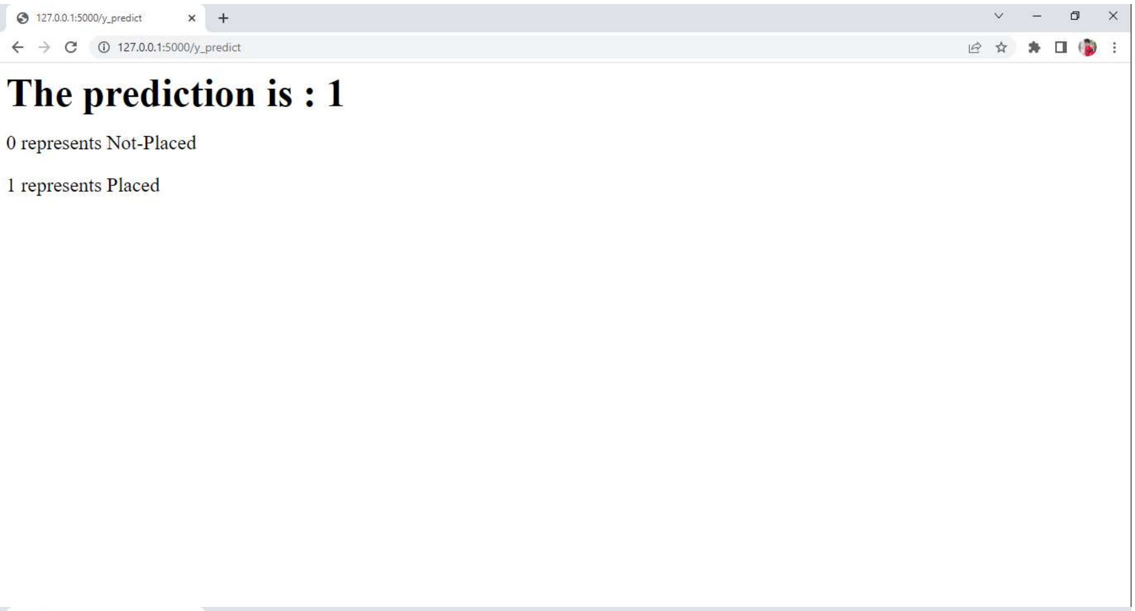
0

2

1

8

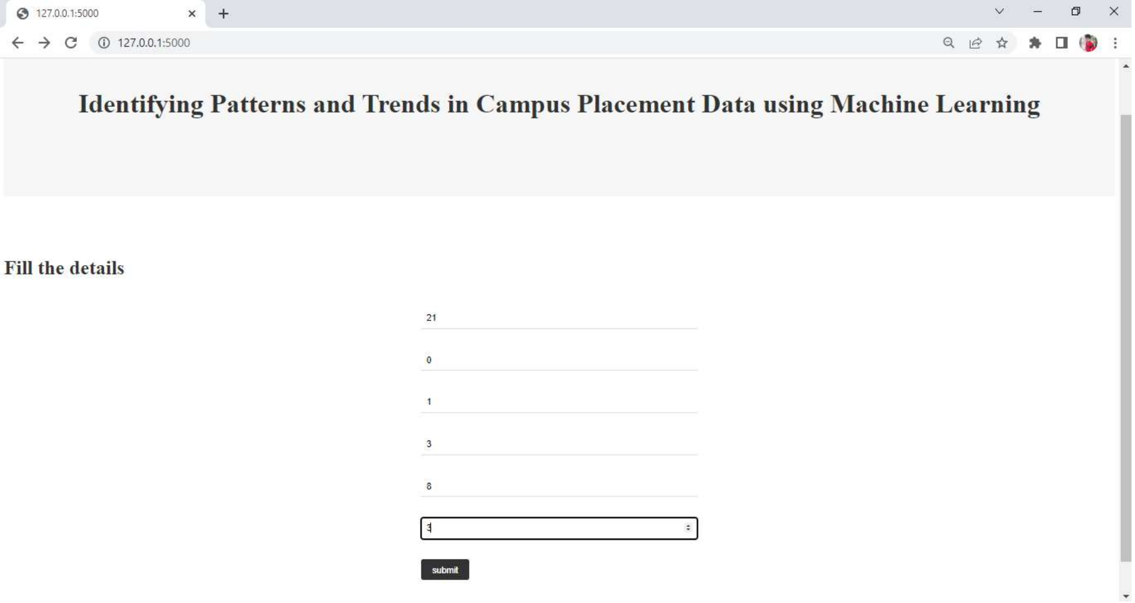
1



The prediction is : 1

0 represents Not-Placed

1 represents Placed



Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Fill the details

21

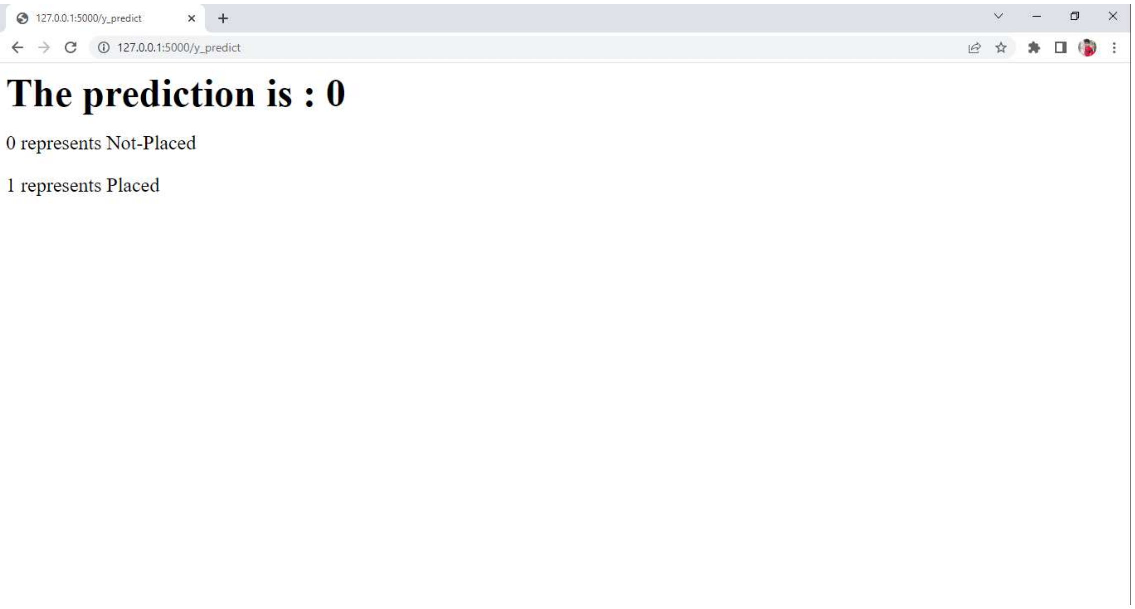
0

1

3

8

3



The prediction is : 0

0 represents Not-Placed

1 represents Placed