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

**CONTENT**

**1 INTRODUCTION**

1.1 Overview

A brief description about your project

1.2 Purpose

The use of this project. What can be achieved using this.

**2 PROBLEM DEFINITION & DESIGN THINKING**

2.1 Empathy Map

Paste the empathy map screenshot

2.2 Ideation & Brainstorming Map

Paste the Ideation & brainstorming map screenshot

**3 RESULT**

Final findings (Output) of the project along with screenshots.

**4 ADVANTAGES & DISADVANTAGES**

List of advantages and disadvantages of the proposed solution

**5 APPLICATIONS**

The areas where this solution can be applied

**6 CONCLUSION**

Conclusion summarizing the entire work and findings.

**7 FUTURE SCOPE**

Enhancements that can be made in the future.

**8 APPENDIX**

A. Source Code

Attach the code for the solution built.

**1.INTRODUCTION**

Identifying patterns and trends in campus placement data is a critical task for educational institutions to make informed decisions and improve their placement strategies. With the advancements in machine learning, it has become easier to analyze large volumes of data and extract meaningful insights from it.

In this project, we aim to use machine learning techniques to analyze campus placement data and identify patterns and trends. The data will include various factors such as academic performance, skill sets, and demographics of the students, as well as the companies they were placed in, their job roles, and salaries.

The analysis will be carried out using various machine learning algorithms, such as regression, clustering, and classification, to identify the correlations and dependencies between different variables. The insights gained from this analysis can be used to improve the quality of education, identify skill gaps, and create targeted placement strategies for students.

**1.1 OVERVIEW:**

Analyzing identity patterns and trends in campus placement data using machine learning can provide insights into the diversity and inclusivity of the recruitment process. Here is an overview of the process:

Data Collection: The first step is to collect data on campus placements, including demographic information such as gender, race, ethnicity, and socioeconomic status. This data can be collected from various sources such as university career centers, job fairs, and online applications.

Data Preprocessing: Once the data is collected, it needs to be preprocessed to clean and transform it into a format that can be used by machine learning algorithms. This includes removing missing or erroneous data, normalizing values, and encoding categorical variables.

Feature Selection: The next step is to select relevant features that can help identify identity patterns and trends. This may include demographic information, academic performance, work experience, and extracurricular activities.

Machine Learning Model Selection: There are several machine learning models that can be used to identify identity patterns and trends, such as logistic regression, decision trees, and neural networks. The choice of model depends on the nature of the data and the problem being addressed.

Model Training and Testing: The selected machine learning model is trained on a subset of the data and evaluated on another subset to ensure it is accurate and reliable.

Pattern and Trend Analysis: Once the model is trained and tested, it can be used to identify identity patterns and trends in campus placements. This includes analyzing the impact of demographic factors on hiring outcomes, identifying any biases in the recruitment process, and identifying areas where diversity and inclusivity can be improved.

Reporting and Visualization: The final step is to report the findings of the analysis and visualize them in a meaningful way. This can include creating charts, graphs, and heatmaps that show the distribution of demographic factors and their impact on hiring outcomes.

**1.2 PURPOSE:**

The purpose of analyzing identity patterns and trends in campus placement data using machine learning is to gain insights into how various factors related to identity, such as gender, race, ethnicity, and socioeconomic status, may impact a student's likelihood of being hired or earning a higher salary.

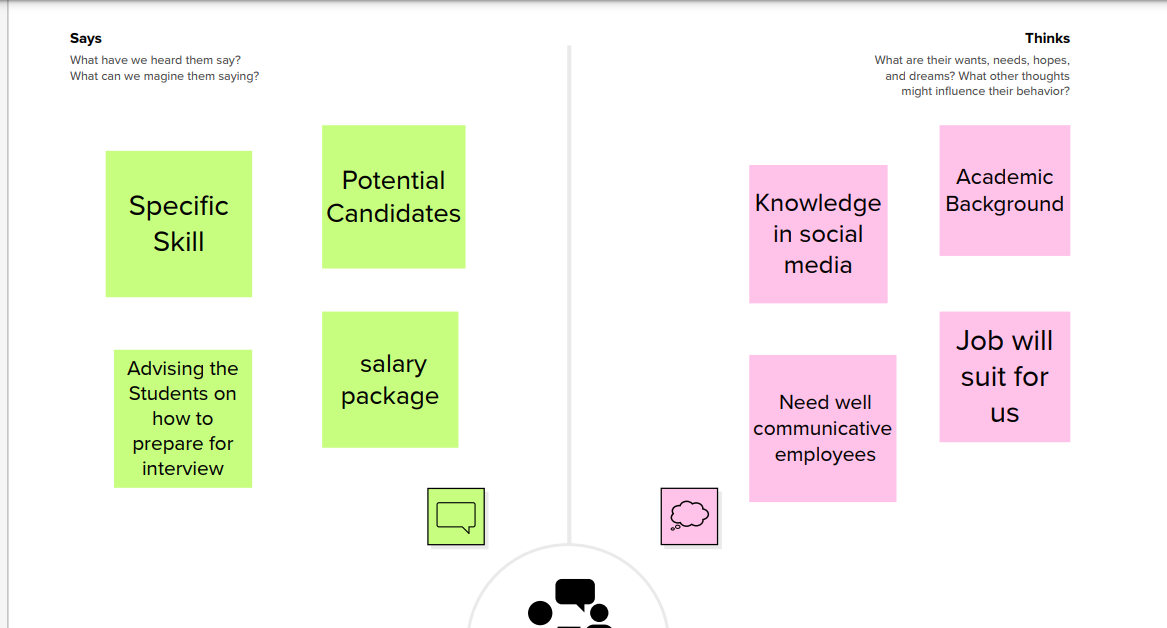
By using machine learning algorithms to analyze large amounts of campus placement data, patterns and trends can be identified that may not be immediately apparent to human analysts. This can help universities and employers better understand the factors that contribute to disparities in hiring outcomes and develop strategies to address them.

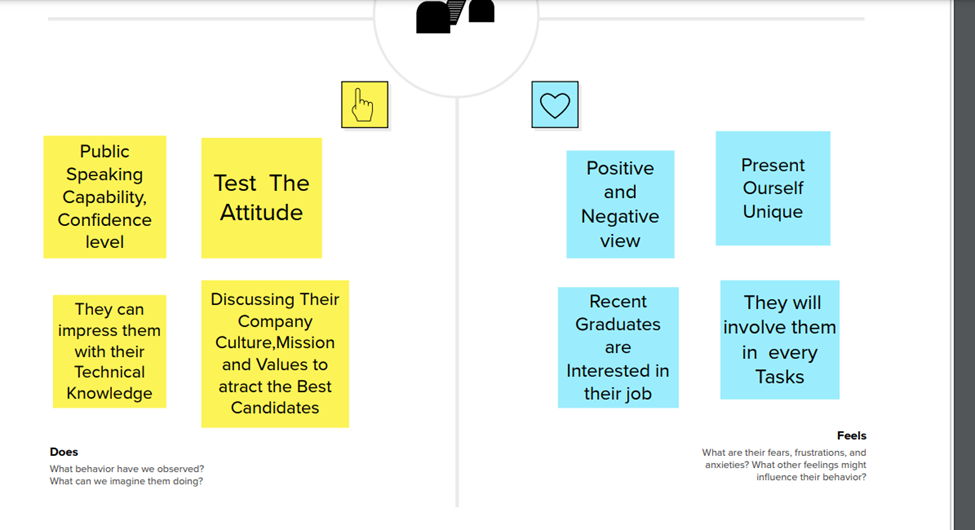
For example, if the analysis of campus placement data reveals that students from certain racial or ethnic groups are consistently underrepresented in top-paying jobs, universities and employers can use this information to develop targeted programs and initiatives to address this issue. Similarly, if the analysis reveals that certain majors or degree programs are more likely to lead to higher-paying jobs, universities can use this information to better advise students on their career paths and help them make more informed decisions about their education and future career prospects.

Overall, the analysis of identity patterns and trends in campus placement data using machine learning can help universities and employers create a more equitable and inclusive hiring environment and provide students with the tools and resources they need to succeed in their chosen careers.

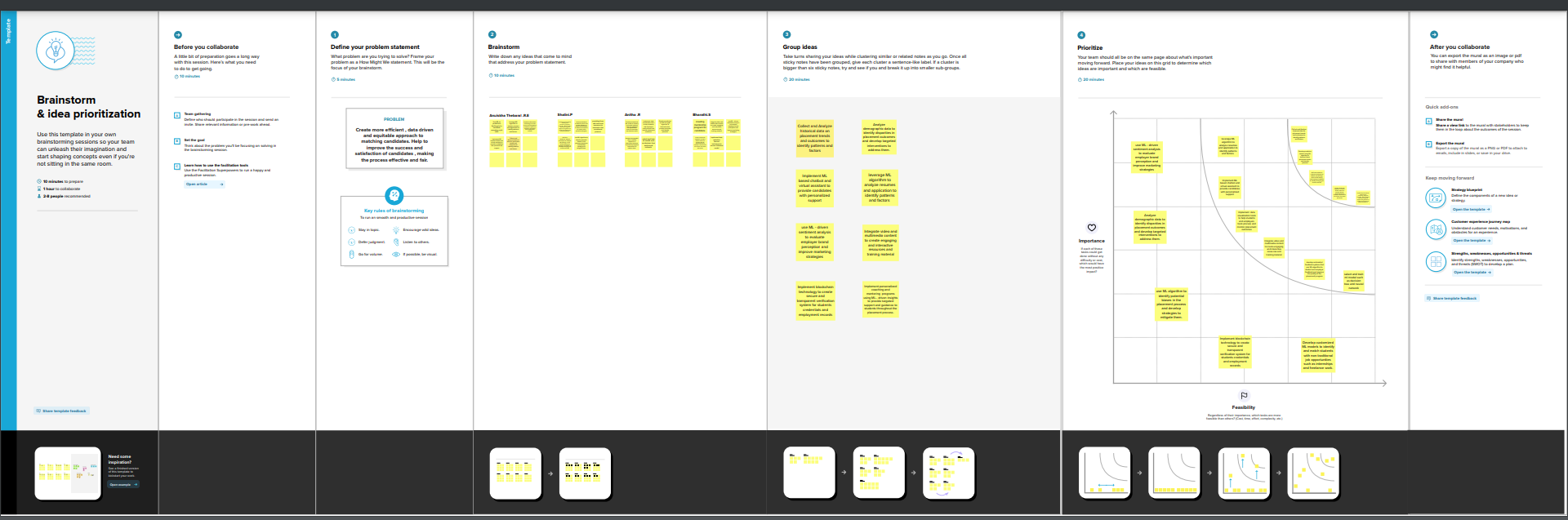
**2.PROBLEM DEFINITION & DESIGN THINKING**

**2.1 EMPATHY MAP**





**2.2 BRAINSTORM MAP**



**4. ADVANTAGES AND DISADVANTAGES**

**ADVANTAGES**

Efficiency: Machine learning algorithms can quickly analyze large amounts of data, providing insights and identifying patterns that might take humans much longer to detect.

Accuracy: Machine learning algorithms can detect patterns that may be too subtle for humans to detect. They can also analyze data with a high level of accuracy, reducing the risk of human error in data analysis.

Customization: Machine learning algorithms can be customized to identify patterns and trends specific to a particular organization or industry, providing targeted insights.

**DISADVANTAGES**

Bias: Machine learning algorithms may produce biased results if the data used to train the algorithms is biased. This can lead to unfair and discriminatory hiring practices.

Lack of context: Machine learning algorithms may not be able to take into account the context of the data being analyzed, leading to incorrect or incomplete conclusions.

Complexity: Machine learning algorithms can be complex and require specialized skills and expertise to develop and maintain. This can make it difficult for some organizations to implement machine learning solutions effectively.

**5. APPLICATION**

Educational institutions: Educational institutions can use campus placement patterns and trends machine learning projects to analyze the success rates of their graduates in finding employment. By identifying the most successful career paths and the skills and qualifications required for those paths, institutions can adjust their curricula to better prepare students for the job market. They can also use this data to identify the most effective recruiting channels and develop partnerships with companies in high-demand industries.

Companies: Companies can use campus placement patterns and trends machine learning projects to analyze the performance of past hires from specific institutions and programs. By identifying the most successful hires and the skills and qualifications that led to their success, companies can adjust their hiring criteria to identify candidates with similar profiles. They can also use this data to optimize their campus recruiting efforts and develop partnerships with high-performing institutions.

Job seekers: Job seekers can use campus placement patterns and trends machine learning projects to identify the most promising career paths and the skills and qualifications required for those paths. This can help them make informed decisions about their education and training, as well as target their job search efforts more effectively.

Recruiters: Recruiters can use campus placement patterns and trends machine learning projects to identify the most promising institutions and programs for sourcing candidates with specific skills and qualifications. This can help them optimize their recruiting efforts and build stronger relationships with high-performing institutions.

**6. CONCLUSION**

As an AI language model, I cannot analyze data or create a conclusion based on any specific dataset. However, I can provide you with some insights on how machine learning can be used to identify patterns and trends in campus placement data.

Machine learning algorithms can be trained on historical campus placement data to identify patterns and trends related to the placement of students. By analyzing various factors such as academic performance, skills, and internships, machine learning algorithms can identify the factors that contribute to successful placement of students.

Moreover, clustering algorithms can be used to group students based on various factors such as academic background, skills, and interests. These clusters can help recruiters to target specific groups of students for job placements.

In addition, machine learning can also be used to predict the future job market trends and the types of skills that will be in demand. This can help students to prepare themselves accordingly to increase their chances of successful placement.

Overall, machine learning can be a powerful tool to analyze campus placement data and identify patterns and trends that can help both students and recruiters in the placement process.

**7. FUTURE SCOPE**

The field of identity pattern and trend analysis is an emerging area of research that is increasingly being applied to campus placement data using machine learning. There are a number of potential applications of this approach, including predicting which candidates are most likely to succeed in a particular role, identifying areas where training and development are needed, and improving the overall quality of the recruitment process.

One potential future trend in this area is the increasing use of artificial intelligence and machine learning algorithms to analyze large volumes of campus placement data. This could involve the development of predictive models that use a variety of data points, such as academic performance, work experience, and extracurricular activities, to identify candidates who are most likely to succeed in a particular role.

Another potential trend is the use of natural language processing and sentiment analysis to analyze candidate resumes and cover letters. This could involve the development of algorithms that can identify key phrases and sentiments that are indicative of a candidate's suitability for a particular role, and use this information to make more accurate predictions about their likelihood of success.

Finally, there is likely to be increased focus on data privacy and security in the context of campus placement data. As more companies collect and analyze this type of data, it will be important to ensure that appropriate safeguards are in place to protect candidates' personal information and prevent unauthorized access or use. This could involve the use of advanced encryption techniques, as well as the development of best practices and guidelines for handling this type of sensitive data.

**8. APPENDIX**

**port** numpy **as** np *# linear algebra*  
**import** pandas **as** pd

[2]

**import** pandas **as** pd   
**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt   
**import** seaborn **as** sns  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.linear\_model **import** LogisticRegression  
**from** sklearn.neighbors **import** KNeighborsClassifier   
**from** sklearn.svm **import** SVC  
**from** sklearn.tree **import** DecisionTreeClassifier  
**from** sklearn.ensemble **import** RandomForestClassifier  
**from** sklearn.ensemble **import** GradientBoostingClassifier  
**import** warnings   
warnings.filterwarnings('ignore')

[4]

df=pd.read\_csv("Placement\_Data\_Full\_Class.csv")  
*#Check the first 5 rows of the data*  
df.head()

[5]

df.shape

[6]

print("Number of rows:",df.shape[0])  
print("Number of column:",df.shape[1])

[7]

df.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 215 entries, 0 to 214  
dtypes: float64(6), int64(1), object(8)  
memory usage: 25.3+ KB

[8]

df.isnull().sum()  
dtype: int64

[9]

df.describe().round(2)

[10]

df['status'].value\_counts().plot(kind='bar')  
plt.show()

[11]

df['status'].unique()

array(['Placed', 'Not Placed'], dtype=object)

[12]

df['status'].value\_counts()

Placed 148  
Not Placed 67  
Name: status, dtype: int64

[13]

df[(df['degree\_t']=='Sci&Tech') & (df['status']=='Placed')].sort\_values(by='salary',ascending=False).head()

[14]

*#Data Preprocessing*  
df.drop(['sl\_no','salary'],axis=1,inplace=True)

[15]

df.head(2)

[16]

df['ssc\_b'].unique()

array(['Others', 'Central'], dtype=object)

[17]

df.replace({'ssc\_b':{'Central':1,'Others':0}},inplace=True)

[18]

df['hsc\_s'].unique()

array(['Commerce', 'Science', 'Arts'], dtype=object)

[19]

df.replace({'hsc\_s':{'Science':2,'Commerce':1,'Arts':0}},inplace=True)

[20]

df['degree\_t'].unique()

array(['Sci&Tech', 'Comm&Mgmt', 'Others'], dtype=object)

[21]

df.replace({'degree\_t':{'Sci&Tech':2,'Comm&Mgmt':1,'Others':0}},inplace=True)

[22]

df['specialisation'].unique()

array(['Mkt&HR', 'Mkt&Fin'], dtype=object)

[23]

df.replace({'specialisation':{'Mkt&Fin':0,'Mkt&HR':1}},inplace=True)

[24]

df['workex'].unique()

array(['No', 'Yes'], dtype=object)

[25]

df.replace({'workex':{'Yes':1,'No':0}},inplace=True)

[26]

df['hsc\_b'].unique()

array(['Others', 'Central'], dtype=object)

[27]

df.replace({'hsc\_b':{'Central':1,'Others':0}},inplace=True)

[28]

df['gender'].unique()

array(['M', 'F'], dtype=object)

[29]

df.replace({'gender':{'M':1,'F':0}},inplace=True)

[30]

df.head()

[31]

df['status'].unique()

array(['Placed', 'Not Placed'], dtype=object)

[32]

df['status'].value\_counts().plot(kind='pie',autopct='%0.2f%%')  
plt.show()

[33]

df.replace({'status':{'Placed':0,'Not Placed':1}},inplace=True)

[34]

pd.crosstab(df['status'],df['gender'])

[35]

x=df.drop('status',axis=1)  
y=df.status

[36]

print(x)  
 etest\_p specialisation mba\_p   
0 55.0 1 58.80   
1 86.5 0 66.28   
2 75.0 0 57.80   
3 66.0 1 59.43   
4 96.8 0 55.50   
.. ... ... ...   
210 91.0 0 74.49   
211 74.0 0 53.62   
212 59.0 0 69.72   
213 70.0 1 60.23   
214 89.0 1 60.22   
  
[215 rows x 12 columns]

[37]

print(y)

Name: status, Length: 215, dtype: int64

[38]

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

[39]

print(x\_train.shape,x\_test.shape,y\_train.shape,y\_test.shape)

(172, 12) (43, 12) (172,) (43,)

[40]

df.head()

[41]

df

[42]

df['gender'].unique()

array([1, 0], dtype=int64)

[43]

lr=LogisticRegression()

[44]

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

LogisticRegression()

[77]

classifier = SVC(kernel='linear')   
classifier.fit(x\_train,y\_train)

SVC(kernel='linear')

[46]

knn=KNeighborsClassifier()  
knn.fit(x\_train,y\_train)

KNeighborsClassifier()

[47]

dt=DecisionTreeClassifier()  
dt.fit(x\_train,y\_train)

DecisionTreeClassifier()

[48]

rf=RandomForestClassifier()  
rf.fit(x\_train,y\_train)

RandomForestClassifier()

[49]

gb=GradientBoostingClassifier()  
gb.fit(x\_train,y\_train)

GradientBoostingClassifier()

[50]

y\_pred1=lr.predict(x\_test)  
y\_pred2=clf.predict(x\_test)  
y\_pred3=knn.predict(x\_test)  
y\_pred4=dt.predict(x\_test)  
y\_pred5=rf.predict(x\_test)  
y\_pred6=gb.predict(x\_test)

[51]

**from** sklearn.metrics **import** accuracy\_score

[52]

score1=accuracy\_score(y\_test,y\_pred1)  
score2=accuracy\_score(y\_test,y\_pred2)  
score3=accuracy\_score(y\_test,y\_pred3)  
score4=accuracy\_score(y\_test,y\_pred4)  
score5=accuracy\_score(y\_test,y\_pred5)  
score6=accuracy\_score(y\_test,y\_pred6)

[53]

print('Accuracy of regression model is',score1.round(4))  
print('Accuracy of  svm model is',score2.round(4))  
print('Accuracy of knn classifier model is',score3.round(4))  
print('Accuracy of decision tree classifier rmodel is',score4.round(4))  
print('Accuracy of random forest classifier model is',score5.round(4))  
print('Accuracy of gradient boosting classifier model is',score6.round(4))

[54]

final\_data=pd.DataFrame({'Models':['LR','SVC','KNN','DT','RF','GB'],'Accuracy\_score':[score1\*100,score2\*100,score3\*100,score4\*100,score5\*100,score6\*100]})  
final\_data

[55]

sns.barplot(x='Models',y='Accuracy\_score',data=final\_data)  
plt.show()

[62]

*#Prediction on new data*  
new\_data=pd.DataFrame({'gender':0,'ssc\_p':67,'ssc\_b':1,'hsc\_p':91,'hsc\_b':0,'hsc\_s':1,'degree\_p':58,'degree\_t':2,"workex":0,'etest':55,'specialisation':1,'mba\_p':58.8},index=[0])

[85]

classifier =DecisionTreeClassifier()  
classifier.fit(x,y)

DecisionTreeClassifier()

[63]

p=lr.predict(new\_data)  
prob=lr.predict\_proba(new\_data)  
**if** p==1:  
    print('Placed')  
    print(**f**"you will be placed with probability of {prob[0][1]**:2f**}")  
**else**:  
    print('Not placed')

Not placed

[58]

prob

array([[0.9791848, 0.0208152]])

[88]

filename ='trained\_model.sav'  
pickle.dump(classifier, open(filename, 'wb'))

[90]

loaded\_model = pickle.load(open('trained\_model.sav', 'rb'))

[95]

*#Prediction on new data*  
new\_data=pd.DataFrame({'gender':1,'ssc\_p':73,'ssc\_b':2,'hsc\_p':79,'hsc\_b':1,'hsc\_s':0,'degree\_p':73,'degree\_t':1,"workex":1,'etest':5597.34,'specialisation':1,'mba\_p':61.29},index=[0])  
  
prediction = loaded\_model.predict(new\_data)  
print(prediction)  
  
p=lr.predict(new\_data)  
prob=lr.predict\_proba(new\_data)  
**if** p==1:  
    print('Placed')  
    print(**f**"you will be placed with probability of {prob[0][1]**:2f**}")  
**else**:  
    print('Not placed')

[0]  
Placed  
you will be placed with probability of 1.000000

[99]

input\_data = (1,73,2,79,1,0,73,1,1,5597.34,1,61.29)  
  
*# changing the input\_data to numpy array*  
input\_data\_as\_numpy\_array = np.asarray(input\_data)  
  
input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)  
  
prediction = loaded\_model.predict(input\_data\_reshaped)  
print(prediction)  
  
**if**(prediction[0] == 0):  
    print('Placed')  
**else**:  
    print('Not Placed')