

GOVERNMENT ARTS COLLEGE

TIRUCHIRAPPALLI - 22



**DEPARTMENT OF COMPUTER
SCIENCE AND APPLICATION**

MACHINE LEARNING WITH PYTHON

*Project Title : Identifying Patterns and Trends in
Campus Placement Data using Machine Learning*

Team Id : NM2023TMID34444

Team Leader : KISHORE R

Team member: GOPALAKRISHNAN

Team member: SANJAY S

Team member: PRASANNA S

1.Introduction

1.1 overview

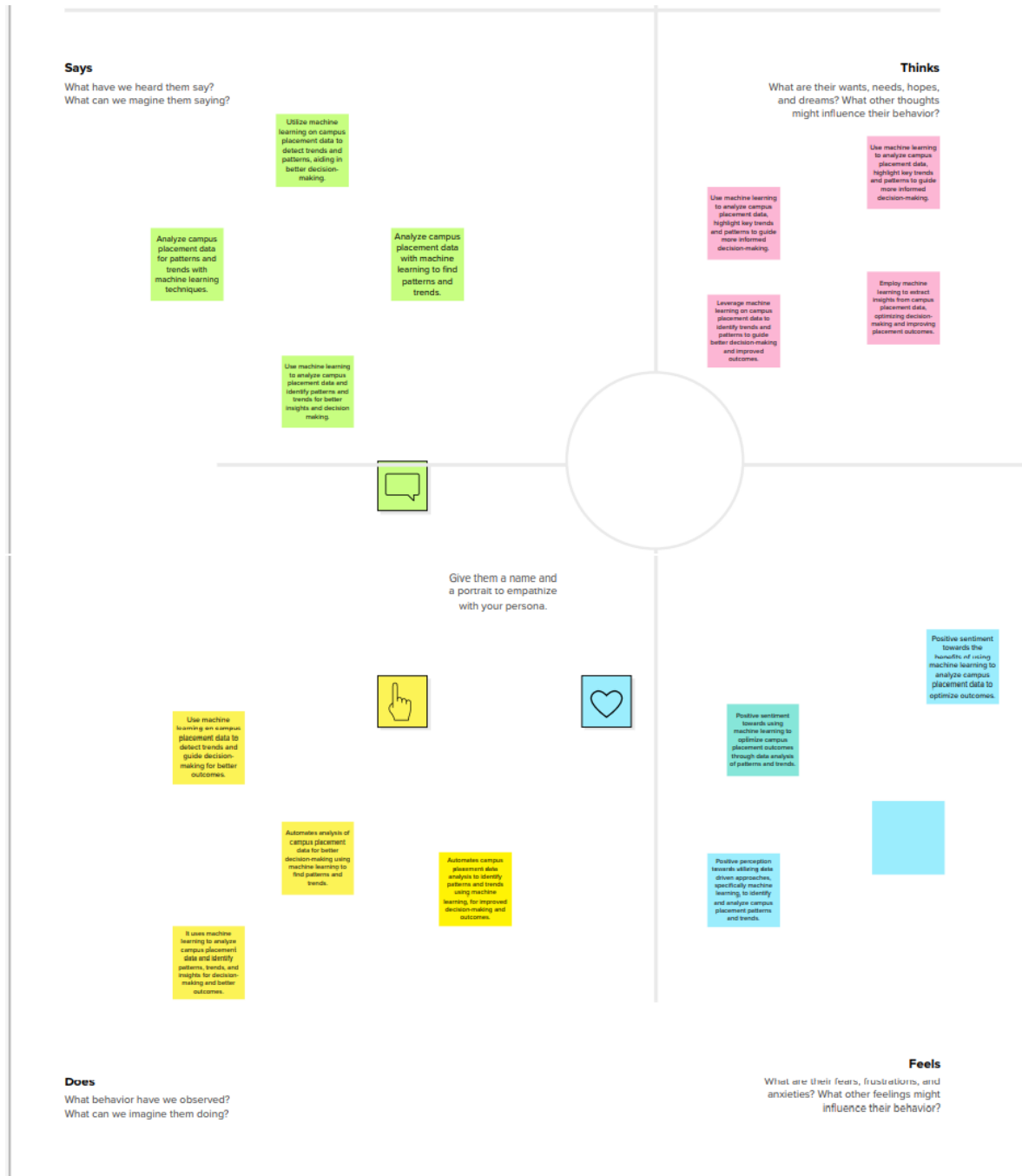
- The project aims to analyze campus placement data using machine learning to identify patterns and trends in the data.
- The project will use various machine learning algorithms such as regression, decision trees, and clustering to uncover meaningful insights from the data.
- The dataset used for this project will likely include information about students such as their academic performance, skills, and experience, as well as information about the companies offering job opportunities.
- The project's ultimate goal is to provide actionable insights to educational institutions and students to help them make better decisions regarding their career choices.
- The project's success will be evaluated based on the accuracy and relevance of the insights uncovered, as well as the usefulness of these insights to the target audience.

1.2 Purpose

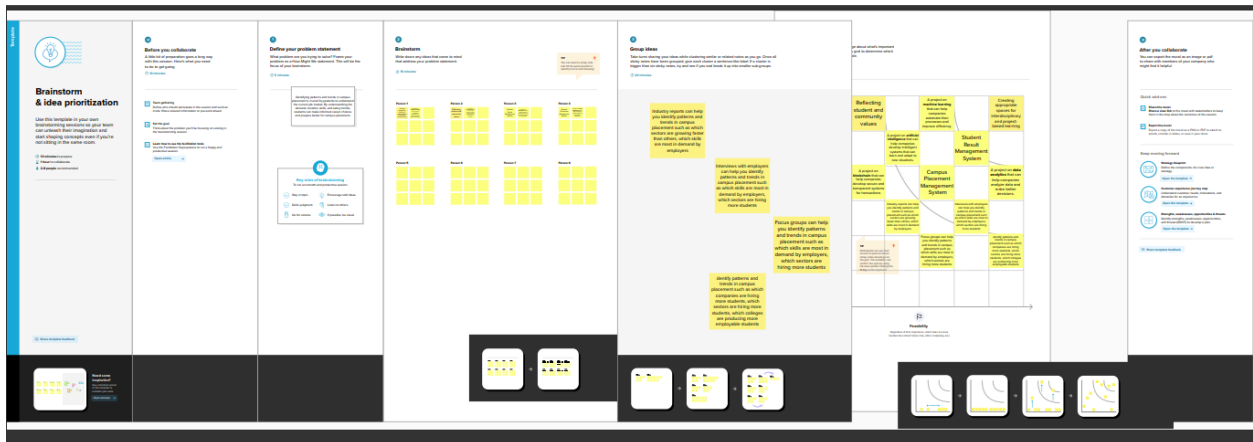
- To help educational institutions improve their placement programs and increase the number of students who get placed in desirable jobs.
- To assist students in making informed decisions about their career paths by identifying the skills, qualifications, and experiences that are most in demand in the job market.
- To provide companies with insights into the skills and qualifications of candidates who are most likely to succeed in their organization.
- To improve the overall efficiency and effectiveness of the campus recruitment process by using data-driven insights to guide decision-making.
- To contribute to the development of machine learning techniques and tools for analyzing large and complex datasets.

2.Problem Definition & Design Thinking

2.1 Empathy Map



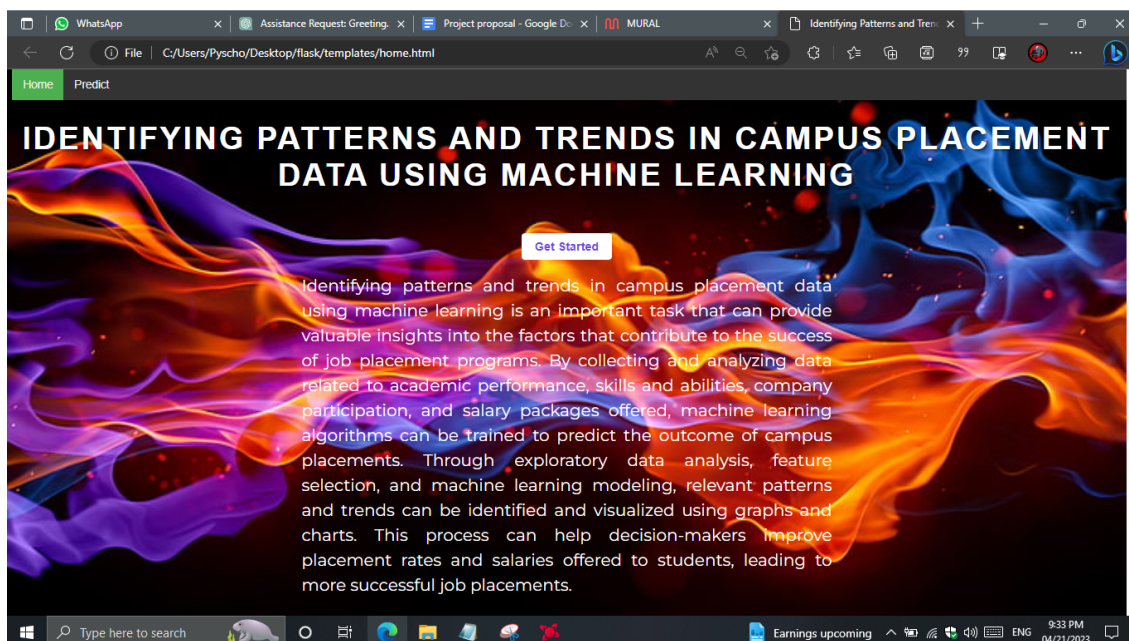
2.2 ideation & Brainstorming Map



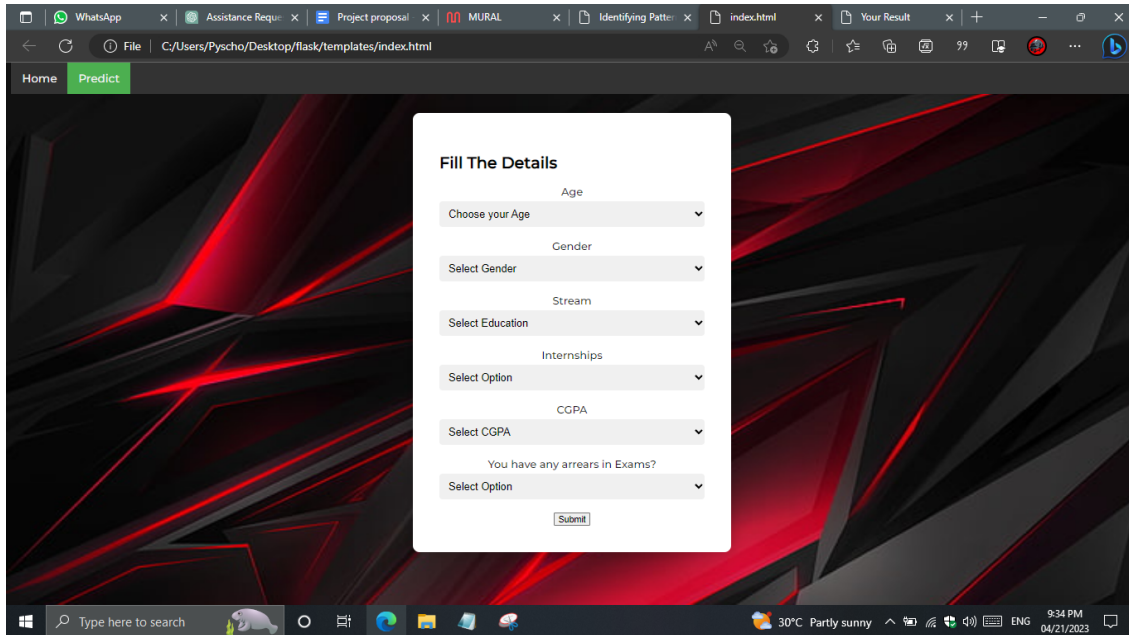
3.Result

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

→ Home.Html



→ Index.Html



The screenshot shows a web browser window with the URL `C:/Users/Psycho/Desktop/flask/templates/index.html`. The page has a dark background with red and black geometric patterns. A white form titled "Fill The Details" is centered on the page. The form contains several dropdown menus for selecting user information and a "Submit" button at the bottom.

Home Predict

Fill The Details

Age
Choose your Age

Gender
Select Gender

Stream
Select Education

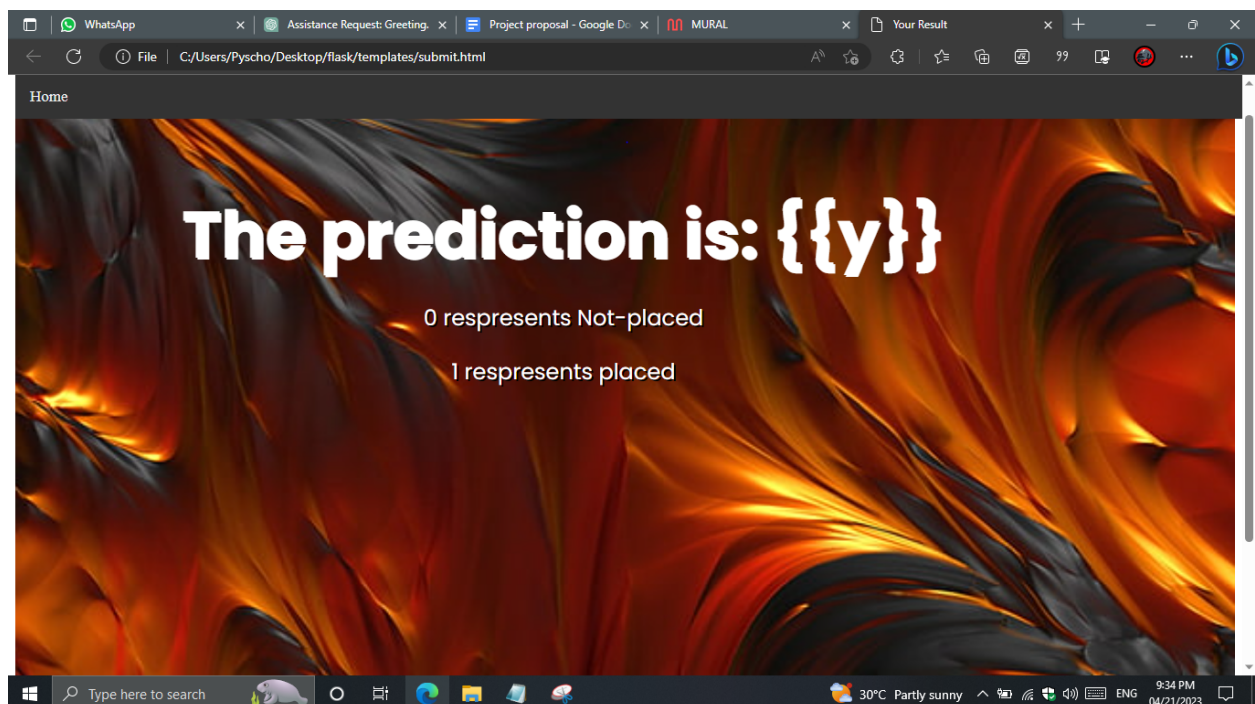
Internships
Select Option

CGPA
Select CGPA

You have any arrears in Exams?
Select Option

Submit

→ Submit.Html



4. Advantages & Disadvantages

Advantages:-

- Improved accuracy: Machine learning algorithms are designed to learn from data and make predictions or identify patterns based on that data. As a result, the insights generated by these algorithms are likely to be more accurate and reliable than those generated manually.
- Faster analysis: Machine learning algorithms can process large amounts of data quickly, making it possible to analyze campus placement data on a large scale and identify patterns and trends that might not be apparent through manual analysis.
- Objective insights: Machine learning algorithms are not biased by personal opinions or preconceptions, meaning that the insights they generate are likely to be more objective and impartial than those generated by humans.
- Better decision-making: By providing actionable insights into campus placement data, machine learning can help educational institutions, students, and companies make better decisions about their recruitment and career strategies.
- Innovation: Developing machine learning models for analyzing campus placement data can drive innovation in the field of data analysis and contribute to the development of new techniques and tools for working with large and complex datasets.

Disadvantages:-

- Data quality issues: Machine learning algorithms rely heavily on the quality of the data they are trained on. If the data is incomplete, inaccurate, or biased, the insights generated by the algorithm may also be flawed.
- Overfitting: Machine learning models can sometimes overfit to the data they are trained on, meaning that they become too specialized and may not generalize well to new data.
- Ethical concerns: The use of machine learning algorithms for analyzing campus placement data raises ethical concerns around privacy, fairness, and transparency. For example, algorithms may inadvertently perpetuate existing biases or discriminate against certain groups of people.
- Expertise requirements: Developing and implementing machine learning models requires specialized knowledge and skills in areas such as data science, statistics, and computer programming. This expertise may not be readily available in all educational institutions or companies.

- Cost and resource requirements: Implementing machine learning algorithms for analyzing campus placement data can be resource-intensive, requiring significant investments in hardware, software, and personnel. This may be a barrier for smaller educational institutions or companies with limited budgets.

5.Application

- Predictive modeling: Using machine learning algorithms to analyze past placement data can help predict which students are most likely to get placed in which companies and what types of jobs they are most likely to be successful in.
- Recommender systems: Machine learning algorithms can be used to build recommender systems that suggest job opportunities to students based on their skills, interests, and career aspirations.
- Skill and gap analysis: Machine learning can be used to identify the most in-demand skills in the job market and assess how well-prepared students are to meet these demands. This information can be used to inform curriculum development and career counseling.
- Employer analysis: Machine learning can be used to analyze data on companies offering job opportunities, such as their industry, size, location, and job requirements. This can help educational institutions and students better understand the job market and make informed decisions about career choices.
- Recruitment optimization: Machine learning can be used to optimize the recruitment process by identifying the most effective channels for reaching potential job candidates, evaluating the performance of recruitment campaigns, and predicting the likelihood of job candidates accepting job offers.

6.conclusion

- ★ The use of machine learning to analyze campus placement data has the potential to provide valuable insights for educational institutions, students, and companies. Machine learning algorithms can help identify patterns and trends in the data that might not be apparent through manual analysis, and generate accurate, objective insights that can inform decision-making. However, there are also potential challenges and drawbacks to using machine learning, including issues with data quality, overfitting, ethical concerns, expertise requirements, and cost and resource

requirements. Therefore, careful consideration and planning is necessary when implementing machine learning for campus placement data analysis, including attention to data quality and ethical considerations. With proper implementation, machine learning can be a powerful tool for improving the efficiency and effectiveness of campus placement programs, supporting students in making informed career decisions, and providing companies with the skilled talent they need to succeed.

7.Future Scope

- Integration with other data sources: In addition to analyzing campus placement data, machine learning algorithms can be used to integrate with other data sources, such as social media profiles or online job listings, to provide a more complete picture of the job market.
- Personalization and customization: Machine learning algorithms can be used to provide personalized recommendations and insights based on individual students' skills, interests, and career aspirations.
- Real-time analysis: Machine learning can be used to analyze campus placement data in real-time, providing up-to-date insights that can inform recruitment strategies and decision-making.
- Ethical and transparent algorithms: As concerns around ethical considerations of machine learning increase, there will be a growing demand for more transparent and ethical algorithms that can be trusted to generate unbiased insights.
- Integration with AI-powered chatbots: The integration of machine learning algorithms with AI-powered chatbots can help provide students with personalized career guidance and support, enhancing the overall campus placement experience.

8.Appendix

A.Source code:-

```
# IMPORT LIB  
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.neighbors import KNeighborsRegressor
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
import warnings
warnings.filterwarnings('ignore')
from scipy import stats

#read date set

df = pd.read_csv('/content/collegePlace.csv')
df.head()

"""# Missing values"""
df.info()
df.isnull().sum()
```

#Handling outliers

```
def transformationplot(feature):  
    plt.figure(figsize=(12,5))  
    plt.subplot(1,2,1)  
    sns.distplot(feature)  
transformationplot(np.log(df['Age']))
```

#categorical values

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

#Univariate analysis

```
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')
```

Bivariate analysis

```
df.drop_duplicates(inplace=True)  
df.isna().sum()
```



```
#ploting the count plot
```

```
plt.figure(figsize=(12,5))
plt.subplot(141)
sns.countplot(df['Gender'])
plt.subplot(142)
sns.countplot(df['Stream'])
plt.show()
```

```
#Multivariate analysis
```

```
plt.figure(figsize=(12,5))
plt.subplot(131)
sns.countplot(data=df, x="PlacedOrNot", hue="CGPA")

sns.swarmplot(x="PlacedOrNot", y="CGPA", hue="Stream", data=df)
```

```
# Scaling the data
```

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

```
pd.DataFrame(df)
df.head()
```

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

```
joblib.dump(X,"placement")
print(X)
```



```
print(y)
```

```
scaler = StandardScaler()
```

```
scaler.fit(X)
```

```
standardized_data = scaler.transform(X)
```

```
print(standardized_data)
```

```
X = standardized_data
```

```
y = df['PlacedOrNot']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.30 , stratify=y,  
random_state=100)
```

```
print(X.shape, X_train.shape, X_test.shape)
```

```
#Model Building
```

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

```
classifier.fit(X_train, y_train)
```

```
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)
```

```
#KNN Model
```

```
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 = 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))
```

```
print("---Results---\nk: {}\nScore: {}".format(best_k, best_score))
```

```
## Instantiate the models
```

```
knn = KNeighborsClassifier(n_neighbors=best_k["Regular"])
```

```
## 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)
```

```
knn_pred
```

```
X_train.shape
```

```
y_train.shape
```

```
"""#ANN model"""

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=7))
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'))

classifier.compile(optimizer='adam', loss='binary_crossentropy')

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)
```

#MODEL DEPLOYMENT

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