Rajalakshmi Engineering College

Student Placement Prediction: A Comparative Study of XGBoost, Random Forest, and Other Classifiers

by

Krishnavarthini K H
220701136
CSE - B

Guide

Dr. V.Auxilia Osvin Nancy.,M.Tech.,Ph.D.,

Introduction

This study presents a machine learning-based approach to predict student placement outcomes using features like academic performance, skills, and extracurricular involvement. By analyzing historical student data, various classification models such as Logistic Regression, SVM, Random Forest, and XGBoost were evaluated to determine the most effective model. The goal is to assist educational institutions and students in making informed career decisions through accurate placement predictions.

Literature Survey

A number of studies have investigated the use of machine learning for predicting student placement outcomes. Researchers have applied classification algorithms like SVM, Naive Bayes, and Random Forest to student datasets, often finding that model performance varies depending on feature selection and data quality. Some papers highlight the importance of including soft skills and extracurricular activities, in addition to academic scores, for better prediction. Recent studies have shown that ensemble methods such as XGBoost outperform traditional models in terms of accuracy and generalization. However, gaps remain in the use of data augmentation and real-world deployment readiness, which this project aims to improve upon.

Literature Survey

Previous research in student placement prediction has explored various machine learning techniques to forecast employment outcomes based on academic and non-academic features. Studies have shown that algorithms like Logistic Regression and Decision Trees offer interpretability, while ensemble methods such as Random Forest and XGBoost provide higher accuracy and robustness. Several works also emphasized the importance of features like CGPA, internships, soft skills, and aptitude scores in predicting placement. However, many lacked data augmentation techniques or comparative evaluations across multiple models, which this study aims to address.

Literature Survey

Recent advancements in educational data mining have led to the application of machine learning models for predicting student outcomes, including academic performance, dropout rates, and placement success. Early studies focused primarily on logistic regression and decision trees, using academic metrics like GPA and test scores. As datasets grew more complex, researchers began incorporating socio-demographic features, soft skills, and participation in extracurricular activities. Modern approaches emphasize ensemble methods—particularly Random Forest and XGBoost—for their ability to handle non-linear data and reduce overfitting. While promising results have been reported, many existing works lack robustness in real-time applicability and overlook the benefits of feature engineering and data augmentation, which our project addresses.

Objectives

Develop a Machine Learning Model

To design and implement classification models that can accurately predict student placement outcomes

Compare Model Performance

To evaluate and compare various algorithms such as Logistic Regression, SVM, Random Forest, and XGBoost based on performance metrics like accuracy, precision, recall, and F1-score.

Identify Key Features

To determine which academic, technical, and extracurricular attributes most strongly influence placement results.

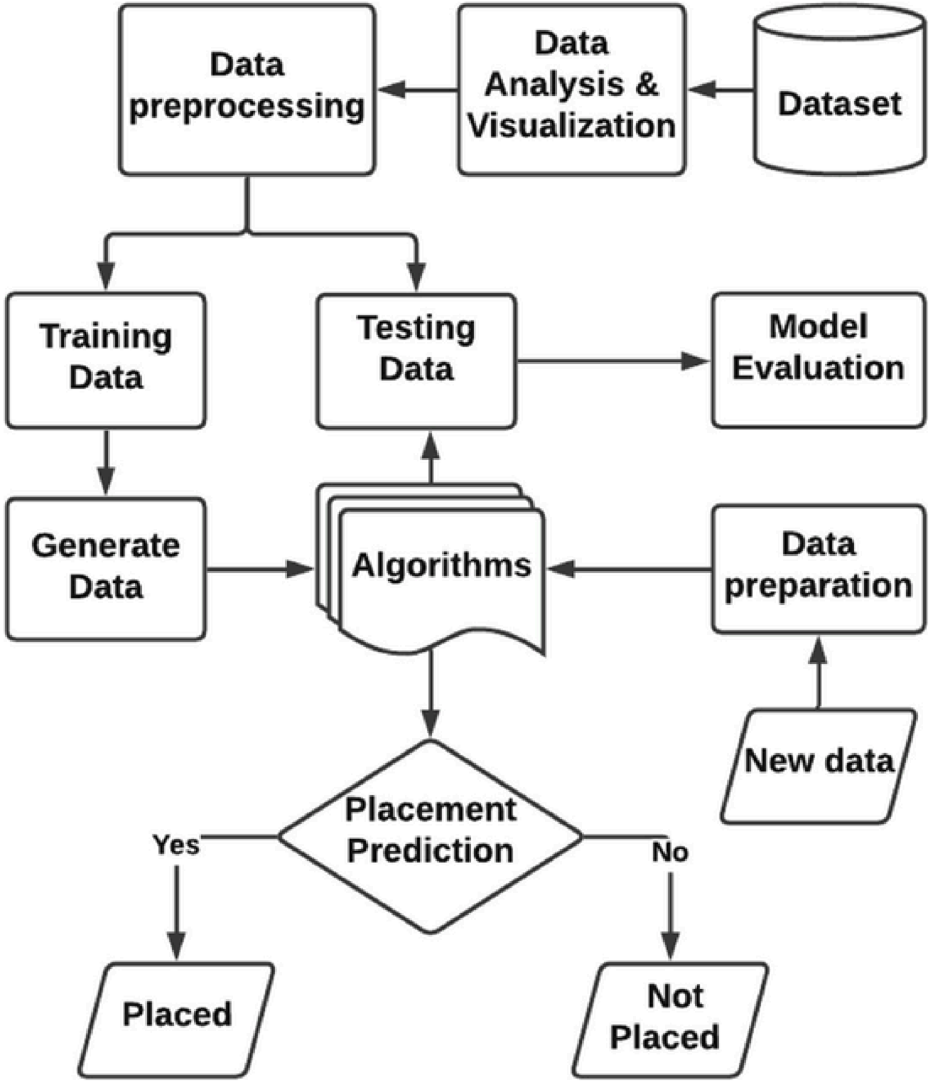
Apply Data Augmentation

To enhance model generalization using Gaussian noise-based data augmentation techniques.

Feature Analysis

To analyze which features (e.g., CGPA, internships, workshops) have the greatest impact on predicting placement success.

System Architecture



Dataset

Data Analysis & Visualization

Data preprocessing

Training Data

Testing Data

Model Evaluation

Generate Data

Algorithms

Data preparation

New data

Placement Prediction

Yes

No

Placed

Not Placed

Methodology

The methodology for the student placement prediction system involves several key steps:

1. **Data Collection:** The dataset includes student attributes like academic performance, skills, and extracurricular activities.
2. **Data Preprocessing:** Missing values are handled, and features are normalized and encoded for machine learning models.
3. **Feature Selection:** Relevant features are selected using techniques like correlation analysis and univariate feature selection.
4. **Model Selection: Four models are used:** Logistic Regression, Support Vector Machine (SVM), Random Forest, and XGBoost.

Methodology

5. **Data Augmentation:** Gaussian noise is added to improve model robustness and reduce overfitting.

6. **Model Training and Evaluation:** Models are trained and evaluated using accuracy, precision, recall, F1-score, and confusion matrix.

7. **Model Optimization:** Hyperparameter tuning is done using Grid Search or Random Search to improve performance.

8. **Deployment:** The best-performing model is deployed to predict placement outcomes and offer personalized career guidance.

Implementation

# Load the dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
data = pd.read_csv("placementdata.csv")

# View first few rows
data.head()
# data.tail()
```

#Preprocessing Data

```
data.drop(['StudentID'], axis=1,
inplace=True)

obj = (data.dtypes == 'object')
print("Categorical variables:",
len(list(obj[obj].index)))
```

#box plot of CGPA distribution

```
import seaborn as sns
import matplotlib.pyplot as plt

# Box plot for CGPA distribution based on Placement
status
sns.boxplot(x='PlacementStatus', y='CGPA', data=data,
hue='PlacementStatus', palette="Set2", legend=False)

# Add labels and title
plt.xlabel('Placed (0: Not Placed, 1: Placed)')
plt.ylabel('CGPA')
plt.title('CGPA Distribution for Placed vs Not Placed
Students')

# Show the plot
plt.show()
```

Implementation

```
#correlation heatmap
plt.figure(figsize=(12,6))
sns.heatmap(data.corr(), cmap='BrBG', fmt='.2f', linewidths=2, annot=True)
plt.show()
```

```
#handle missing data
for col in data.columns:
 data[col] = data[col].fillna(data[col].mean())

# Check missing values
data.isna().sum()
```

Implementation

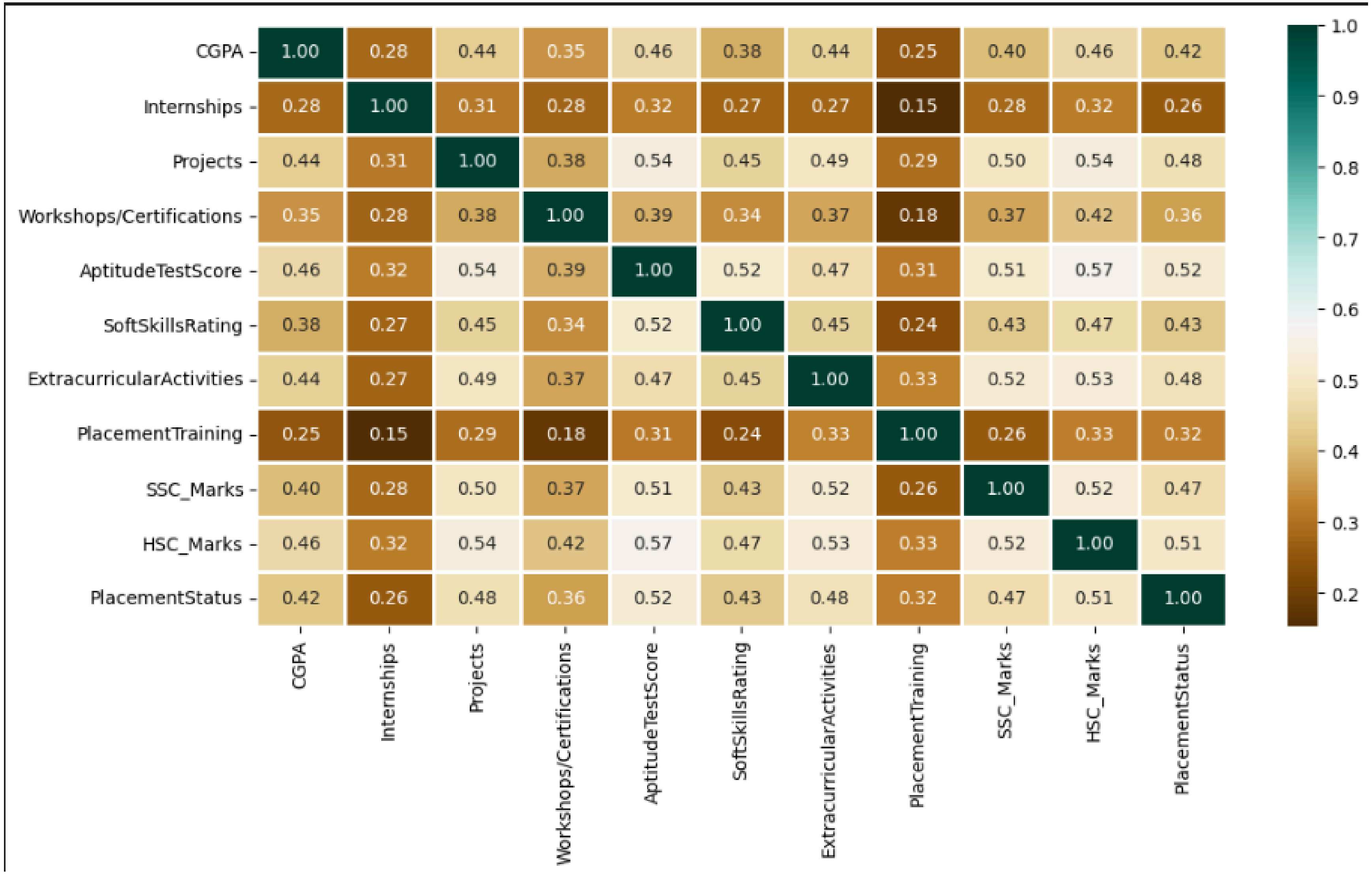
#splits the dataset into training and testing sets

```
from sklearn.model_selection
 import train_test_split

X = data.drop(['PlacementStatus'], axis=1)
Y = data['PlacementStatus']

X_train,X_test,Y_train,Y_test
=train_test_split(X, Y, test_size=0.4,
random_state=1)

X_train.shape, X_test.shape, Y_train.shape,
Y_test.shape
```

#Model Training and Evaluation (Training Accuracy)

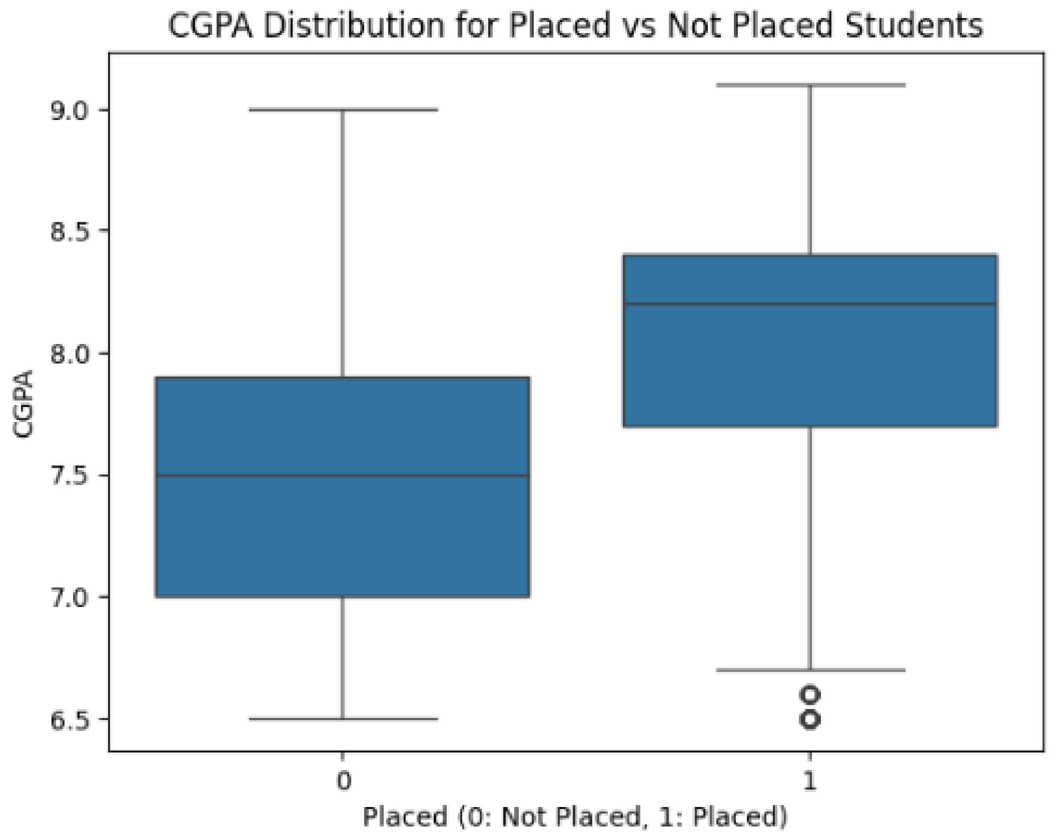
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)

# Define classifiers
knn = KNeighborsClassifier(n_neighbors=3)
rfc = RandomForestClassifier(n_estimators=7, criterion='entropy',
random_state=7)
svc = SVC()
lc = LogisticRegression(max_iter=2000) # increased max_iter
xgb = XGBClassifier(eval_metric='logloss', random_state=7)

print("Training Accuracy:\n")
for clf in (rfc, knn, svc, lc,xgb):
  clf.fit(X_train_scaled, Y_train)
  Y_pred_train = clf.predict(X_train_scaled)
  print("Accuracy score of", clf.__class__.__name__, "=", 100 *
metrics.accuracy_score(Y_train, Y_pred_train))
```

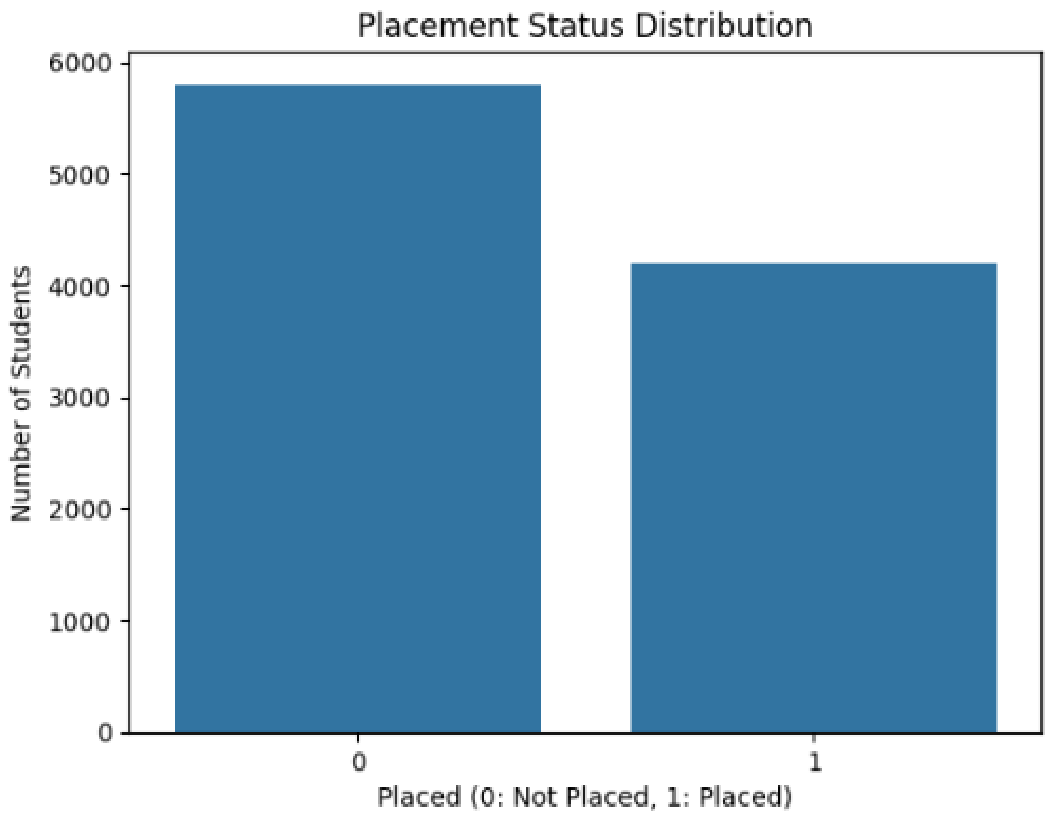
Results



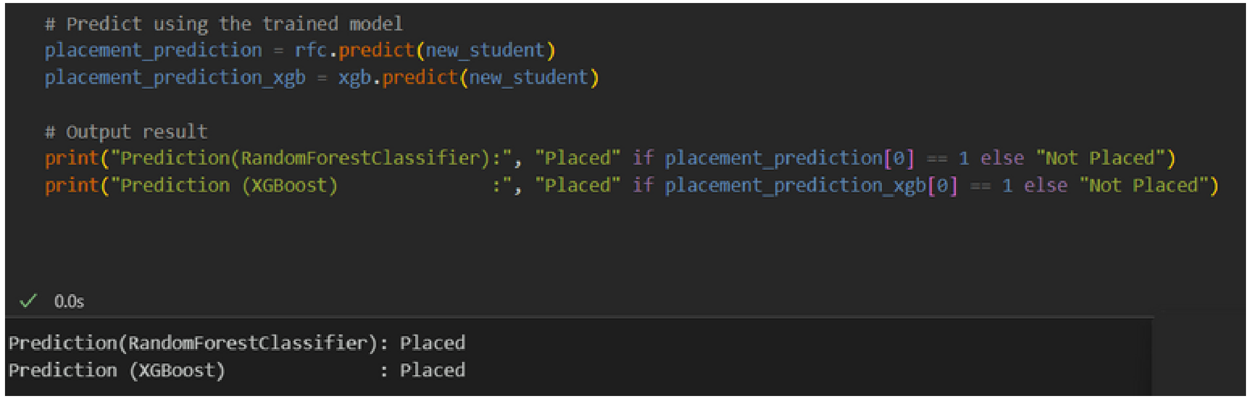
	CGPA	Internships	Projects	Workshops/Certifications	AptitudeTestScore	SoftSkillsRating	ExtracurricularActivities	PlacementTraining	SSC_Marks	HSC_Marks	PlacementStatus
CGPA	1.00	0.28	0.44	0.35	0.46	0.38	0.44	0.25	0.40	0.46	0.42
Internships	0.28	1.00	0.31	0.28	0.32	0.27	0.27	0.15	0.28	0.32	0.26
Projects	0.44	0.31	1.00	0.38	0.54	0.45	0.49	0.29	0.50	0.54	0.48
Workshops/Certifications	0.35	0.28	0.38	1.00	0.39	0.34	0.37	0.18	0.37	0.42	0.36
AptitudeTestScore	0.46	0.32	0.54	0.39	1.00	0.52	0.47	0.31	0.51	0.57	0.52
SoftSkillsRating	0.38	0.27	0.45	0.34	0.52	1.00	0.45	0.24	0.43	0.47	0.43
ExtracurricularActivities	0.44	0.27	0.49	0.37	0.47	0.45	1.00	0.33	0.52	0.53	0.48
PlacementTraining	0.25	0.15	0.29	0.18	0.31	0.24	0.33	1.00	0.26	0.33	0.32
SSC_Marks	0.40	0.28	0.50	0.37	0.51	0.43	0.52	0.26	1.00	0.52	0.47
HSC_Marks	0.46	0.32	0.54	0.42	0.57	0.47	0.53	0.33	0.52	1.00	0.51
PlacementStatus	0.42	0.26	0.48	0.36	0.52	0.43	0.48	0.32	0.47	0.51	1.00

Results


CGPA Distribution for Placed vs Not Placed Students
CGPA
9.0
8.5
8.0
7.5
7.0
6.5
0
1
Placed (0: Not Placed, 1: Placed)


Placement Status Distribution
Number of Students
6000
5000
4000
3000
2000
1000
0
0
1
Placed (0: Not Placed, 1: Placed)

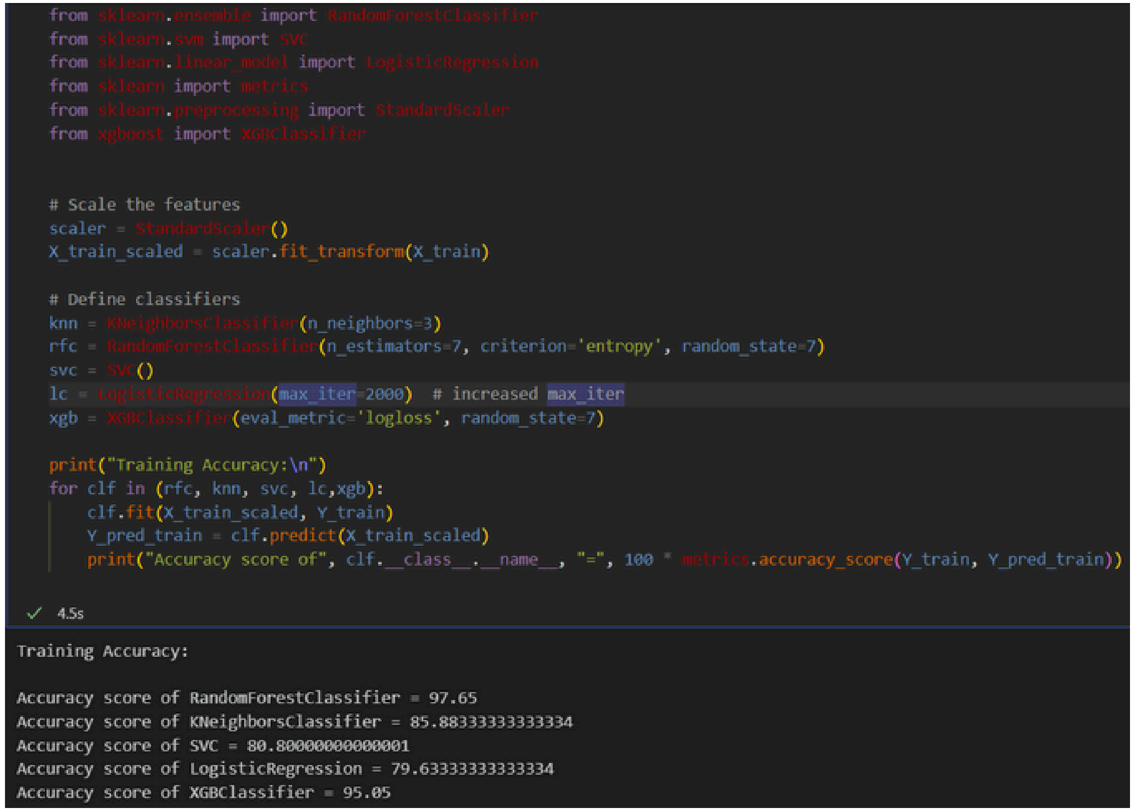
Results



```
# Predict using the trained model
placement_prediction = rfc.predict(new_student)
placement_prediction_xgb = xgb.predict(new_student)

# Output result
print("Prediction(RandomForestClassifier):", "Placed" if placement_prediction[0] == 1 else "Not Placed")
print("Prediction (XGBoost)               :", "Placed" if placement_prediction_xgb[0] == 1 else "Not Placed")
```

✓ 0.0s

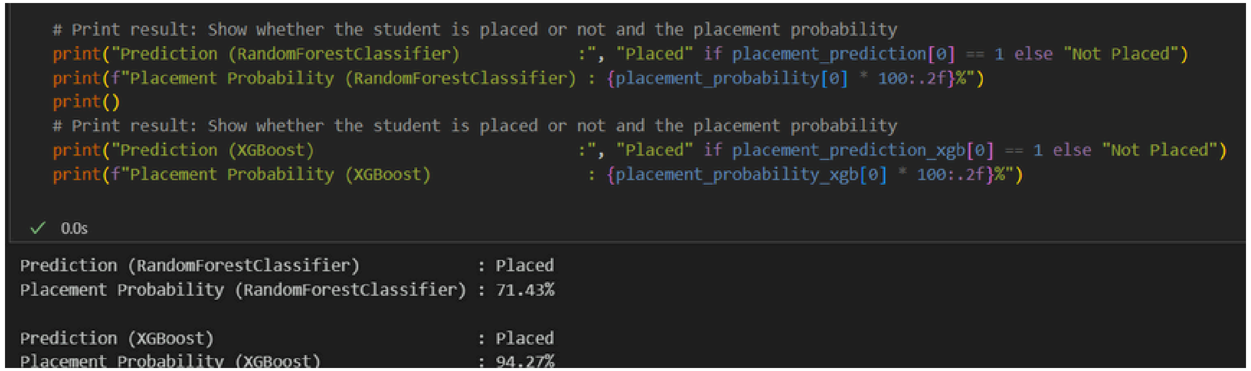
```
Prediction(RandomForestClassifier): Placed
Prediction (XGBoost)              : Placed
```



```
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.preprocessing import StandardScaler
from xgboost import XGBClassifier


# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)

# Define classifiers
knn = KNeighborsClassifier(n_neighbors=3)
rfc = RandomForestClassifier(n_estimators=7, criterion='entropy', random_state=7)
svc = SVC()
lc = LogisticRegression(max_iter=2000)  # increased max_iter
xgb = XGBClassifier(eval_metric='logloss', random_state=7)

print("Training Accuracy:\n")
for clf in (rfc, knn, svc, lc,xgb):
    clf.fit(X_train_scaled, Y_train)
    Y_pred_train = clf.predict(X_train_scaled)
    print("Accuracy score of", clf.__class__.__name__, "=", 100 * metrics.accuracy_score(Y_train, Y_pred_train))
```

✓ 4.5s

```
Training Accuracy:

Accuracy score of RandomForestClassifier = 97.65
Accuracy score of KNeighborsClassifier = 85.88333333333334
Accuracy score of SVC = 80.80000000000001
Accuracy score of LogisticRegression = 79.63333333333334
Accuracy score of XGBClassifier = 95.05
```



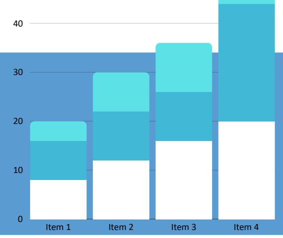
```
# Print result: Show whether the student is placed or not and the placement probability
print("Prediction (RandomForestClassifier)           :", "Placed" if placement_prediction[0] == 1 else "Not Placed")
print(f"Placement Probability (RandomForestClassifier) : {placement_probability[0] * 100:.2f}%")
print()
# Print result: Show whether the student is placed or not and the placement probability
print("Prediction (XGBoost)                          :", "Placed" if placement_prediction_xgb[0] == 1 else "Not Placed")
print(f"Placement Probability (XGBoost)                : {placement_probability_xgb[0] * 100:.2f}%")
```

✓ 0.0s

```
Prediction (RandomForestClassifier)            : Placed
Placement Probability (RandomForestClassifier) : 71.43%

Prediction (XGBoost)                           : Placed
Placement Probability (XGBoost)                : 94.27%
```

Comparison with existing work

Many existing studies on student placement prediction have focused primarily on limited variables such as academic grades or aptitude scores, using simple models like Decision Trees or Naive Bayes. While these models provide a basic level of accuracy, they often fail to capture the complex, non-linear relationships among multiple features. In contrast, our work integrates a broader feature set—including academic performance, technical skills, and extracurricular activities—and applies advanced machine learning models like Random Forest and XGBoost. These ensemble methods outperform traditional techniques by effectively handling feature interactions and reducing overfitting. Additionally, our use of data augmentation using Gaussian noise to improve generalization marks a novel enhancement rarely explored in prior work. This comprehensive approach results in higher accuracy and more reliable placement predictions.



Conclusion and Future Work

- Enhanced Data Collection: Incorporate data like resume quality, interview scores, and communication skills for more accurate predictions.
- Time-Series and Longitudinal Analysis: Use semester-wise or year-wise progression data to model placement trends over time.
- Automated Career Guidance: Extend the system to suggest career paths, skill development plans, and relevant job opportunities.
- Model Optimization: Improve performance using deep learning techniques or hybrid ensemble methods.
- Wider Deployment: Build scalable APIs or dashboards that can be used by multiple institutions.

Reference

[1] J. Luan, “Data mining and its applications in higher education”, New Dir. Inst. Res, 113:17–36, 2002.
[2] A.S. Sharma, S. Prince, S. Kapoor, K. Kumar, “PPS – Placement prediction system using logistic regression”, IEEE international conference on MOOC, innovation and technology in education (MITE), pp 337-341,2014.
[3] Thangavel, S.Bkaratki, P. Sankar, “Student placement analyzer: A recommendation system using machine learning”, Advances in Computing and Communication Systems (ICACCS-2017) International Conference on. IEEE, 2017. [4] R. Sangha, A. Satras, L. Swamy, G. Deshmukh, “Students Placement Eligibility Prediction using Fuzzy Approach”, International Journal of Engineering and Techniques , Volume 2, Issue 6, Dec 2016.

Reference

[5] H. Bhatt, S. Mehta, L. R. D'mello, “Use of ID3 Decision Tree Algorithm for Placement Prediction”, International Journal of Computer Science and Information Technologies (IJCSIT), vol. 6, pp. 4785-4789, 2023.

[6] T. Jeevalatha, N. Ananthi, D. Saravana Kumar, “Performance analysis of undergraduate students placement selection using Decision Tree Algorithms”, International Journal of Computer Applications, vol. 108, pp. 0975-8887, December 2023.

[7] Bharambe, Yogesh, “Assessing employability of students using data mining techniques”, Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on. IEEE, 2022

[8] P. Khongchai, P. Songmuang, “Random Forest for Salary Prediction System to Improve Students Motivation”, 12th International Conference on SignalImage Technology & Internet-Based Systems (SITIS), pp. 637-642, 2016.

THANK YOU