Volume: 11 Issue: V Month of publication: May 2023

DOI: https://doi.org/10.22214/ijraset.2023.52122

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

TnP Portal Using Web Development and Machine Learning

Prasad Khalkar[1], Aditya Muthal[2], Sujit Khopade[3], Lokesh Ghule[4], Prof. Tushar A. Rane[5]
[1, 2, 3, 4, 5]*Department of Information Technology Pune Institute of Computer Technology, Pune*

***Abstract:** In colleges, the training and placement cell is crucial. The creation of a web portal for the training and placement cell is intended to assist in automating the manual tasks of the placement cell, such as gathering candidate information for filling out the application form for any particular company. This application can keep all the student's data, including personal details and details about their education, abilities, and resume, among other things. Students can log on, see, and apply for any drives they're interested in. All student-related data that is necessary for them will be available to the college's TnP cell. Through the use of a classification model based on the data of previously placed students and an appropriate machine learning algorithm, the suggested system seeks to analyze applicant performance and provide the candidate's information regarding which type of company he is the best fit for. The TnP cell of the college may find this application to be a useful tool for managing student information related to the placement process. This tool aims to make the placement application process more organized. Additionally, this system will be able to give a summary of recent and historical placement data, thus the TnP cell and applicants will have a better insight into the same.*
***Index Terms:** Admin module, Classification Algorithms, Campus Placement, Data visualization, Student module, Recruitment portal,*

I. INTRODUCTION

The placement of a student on campus has a big impact on a college. Companies visit colleges during campus placement to identify qualified candidates before they graduate. The most important factors for a successful placement can be found by analyzing patterns and qualities in the massive volumes of student information that schools retain. The placement of engineering students beginning in their second year can be predicted, which can aid in the students' correct development. Students may be given access to an interface that enables them to submit applications to several businesses with a single click, obviating the need to update information that is already in the system. This can cut down on the time and work needed to verify the information given by the pupils.
The other proposed subsystem uses predictive analytics with machine learning classification algorithms to determine the likelihood of placement in a given industry, such as fintech, startups, products, and services. Additionally, it can identify key characteristics that influence a candidate's chances of landing a job in that industry, benefiting both the college and the student.

II. MOTIVATION

Our college's Training and Placement (T&P) division currently collects placement information using Google Forms and manually manages student records using Excel sheets. Here, TPO and his staff handle all work. Due to the human labor involved and the length of the procedure, there is a considerable potential for error. Therefore, we have chosen to work with the T&P department to develop a web-based training and placement tool. The T&P department's suggestions are being used to build the project.

III. RELATED WORK

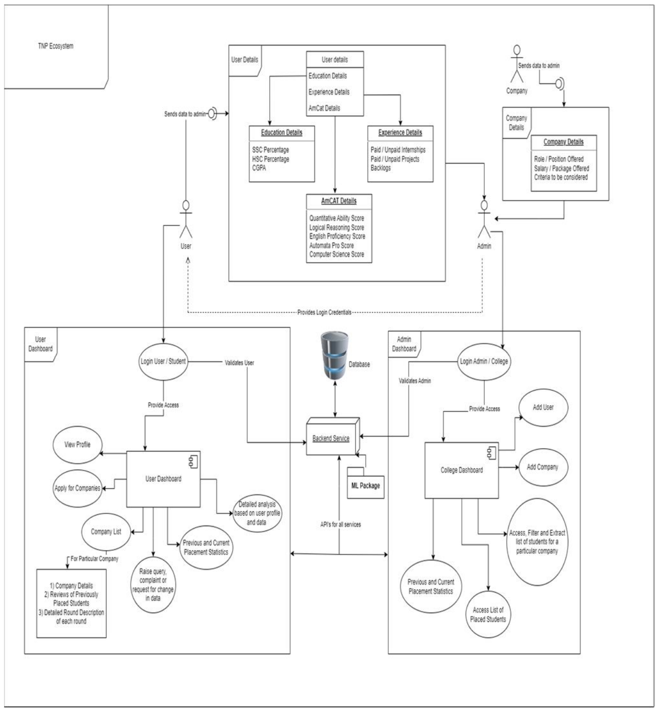
S. Taruna and Mrinal Pandey conducted an empirical analysis on predicting academic accomplishment using classification techniques or the mapping of data pieces into certain groups and classes using supervised learning [5]. They looked at five classification algorithms, including Decision Tree, Naïve Bayes, Naïve Bayes Tree, K-Nearest Neighbour, and Bayesian Network techniques, to anticipate students' grades, particularly for engineering students[1][6]. Syed Tanveer Jishan, Raisul Islam Rashu, Naheena Haque, and Rashedur M. Rahman used quizzes, labs, midterms, and attendance records to forecast students' success.
They also used three classification algorithms—Naïve Bayes, Decision Tree, and Neural Network—to enhance the prediction model by incorporating multiple preprocessing techniques, resulting in a prediction model that produces results that are more accurate. [2]
Saha and Goutam employed the Logistic Regression technique to assess the test result data for the University Grant Commission-funded research titled "Prospects and Problems of Educational Development (Higher Secondary Stage) in Tripura - An In-depth Study.

"[3] By selecting relevant variables based on a student's profile's quantitative and qualitative features, such as CGPA, academic performance, technical capabilities, and communication skills, Hitarthi Bhatt created a model that may forecast a student's placement using ID3[4][7]. Fahim and Torkey worked on enhancing the k-means clustering algorithm such that some information in each iteration is kept and used in the next iteration [8]. Emmanuel Ogor used various data mining techniques to monitor and evaluate student academic performance [9].
We would use ML classification algorithms like Logistic Regression, Random Forest, and Naïve Bayes for probabilistic prediction, taking into account the accuracy of the various techniques listed above.

IV. METHODOLOGY

The methodology is divided into 2 sub-parts wherein the first part contains the application overview and the second part contains the machine learning overview

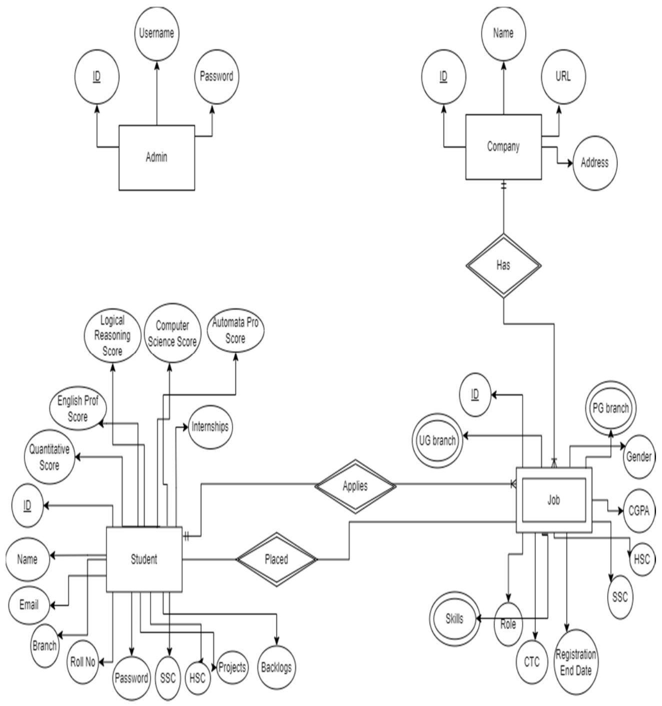
A. Architecture



The system consists of two major components and 3 major entities:
Components:
1) *User Dashboard:* This application is developed using Android Java and consists of all the features that are to be used by the student.
2) *Admin Dashboard:* This application is developed using Angular and consists of all the features that are to be used by the admin.
3) Other components: Machine learning components / APIs are developed using Flask and Backend or the Middleware service is developed using Spring Boot that provides APIs to carry out all the CRUD and other operations.

Entities:
1) *Student:* This entity has access to its own data as well as data of various campus visiting companies. The modification of the student is not directly provided to the student itself and hence requires raising a request regarding the same to admin
2) *Company:* This entity is in direct contact with the admin and doesn't have any access to the system as in the real world the information passes through the admin. So we thought of replicating the same in our system.
3) *Admin:* The following consists of access to all the systems. The modification and student list extraction requests are carried out by Admin for Student and Company respectively.

To further understand more about the entities and their relationship, let's take a look at the Entity Relationship diagram of the system.



As shown in the ER diagram, the entities and their attributes are as follows:

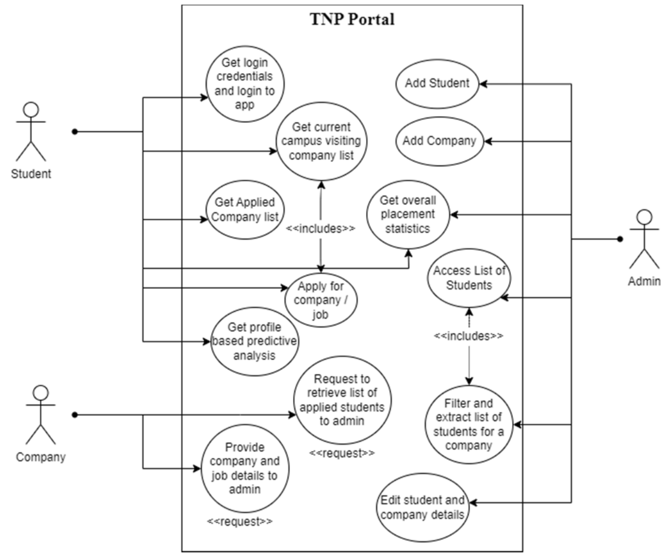
1) *Admin*
   a) ID: Auto-generated Primary Key Identifier
   b) Username
   c) Password: Hashed and stored in the database

2) *Company*
   a) ID: Auto-generated Primary Key Identifier
   b) Name: Name of the company
   c) URL: Company URL
   d) Address: Address or location of the company

3) *Job:* Job is a weak entity that is dependent upon the Company entity.
   a) ID: Auto-generated Primary Key Identifier
   b) Role: The position or role offered by the company
   c) CTC: The CTC offered by the company
   d) Company ID: This specifies the company to which this job instance belongs depicting a one-to-one relationship between Company and Job.
   e) Criteria-specific attributes are: HSC and SSC score, CGPA, Registration end date, Gender allowed to apply, UG and PG Branch allowed to apply, and list of skills which is a multi-valued attribute

4) *Student:*
   a) ID: Auto-generated Primary Key Identifier
   b) General information attributes are Roll No, Email, Branch, and Password.
   c) AMCAT data-based attributes are scores of Logical Reasoning, Quantitative Ability, Computer Science, Automata Pro, and English Proficiency.
   d) Experience-based attributes are the number of backlogs, internships, and projects the student is having.
   e) It also contains the list of applied jobs for the student which depicts a one-to-many mapping between Student and Job.
   f) It also consists of a Placed company that has a one-to-one mapping between Student and Job.

B. Use Case for TNP Portal


TNP Portal
Student
Company
Admin
Get login credentials and login to app
Get current campus visiting company list
Get Applied Company list
<<includes>>
Apply for company / job
Get profile based predictive analysis
Request to retrieve list of applied students to admin
<<request>>
Provide company and job details to admin
<<request>>
Add Student
Add Company
Get overall placement statistics
Access List of Students
<<includes>>
Filter and extract list of students for a company
Edit student and company details

The use case for all the defined entities are as follows:

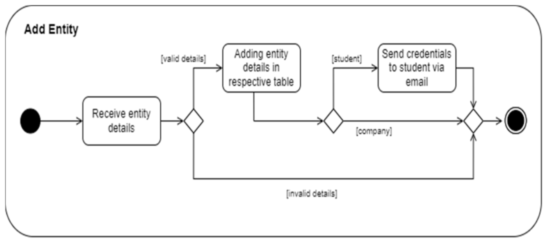
1) The Admin is able to:
a) Add a student and a Company,
b) Access, Filter, and extract a list of applied students for any Company
c) Edit Student and Company details.
d) Get overall placement analysis

2) The Student is able to:
a) Access list of current visiting companies and list of applied companies
b) Get profile-based and overall placement analysis
c) Request for modification of user data to Admin.

3) The Company is able to:
d) Provide company and job details to the admin
e) Request to retrieve the list of applied students.

Major services provided in TNP Portal:

- Email Service:

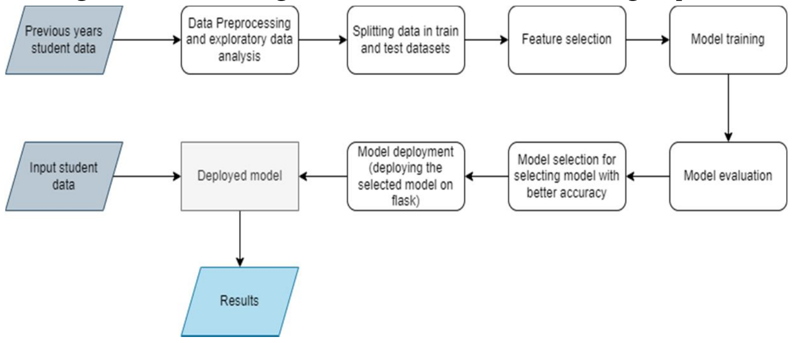


Once the request to add a Student to the system is received, validation of data is checked and after validation, the data is added to the database.
After which, an Email is triggered to the student consisting of the login username and password. This service uses JavaMailSender class, which takes the subject, recipient, and body of the mail to be sent.

- Other services are the modification and deletion service for each entity etc.

C. Machine Learning Overview

Implementation of Machine Learning Classification Algorithms consists of the following steps


Previous years student data
Data Preprocessing and exploratory data analysis
Splitting data in train and test datasets
Feature selection
Model training
Model evaluation
Model selection for selecting model with better accuracy
Model deployment (deploying the selected model on flask)
Deployed model
Input student data
Results

D. Data Collection

For the purposes of this research study, we have collected various types of data including academic performance, AMCAT scores, demographic information, and work experience of senior graduates from our institution within the last three years.

The academic information comprises the cumulative grade point average (GPA), active and passive backlogs, and study area (Computer Engineering, Information Technology, Electronics and Telecommunication Engineering).

The AMCAT scores contain a computer science score, a score for automata, a score for logical thinking, and a score for English proficiency.

Personal projects and internships are included in the job experience.

The gender is included in the demographic information since some on-campus employers exclusively hire female candidates.

Here, there are four categories for the tagged data, which is the sort of organization hiring student on-campus: FinTech, Startups, Product and Service based.

This data is made available by college’s Training and Placement office.

E. Data Preprocessing

The data preprocessing steps we performed are as follows:

1) *Data cleaning*: We initially eliminated any records that were duplicates or had values missing. This made it easier to make sure the data was reliable and accurate for analysis.
2) *Encoding category features*: We used one-hot encoding to transform category features into numerical features. This made it easier to make sure the category features were appropriate for the machine learning algorithms' examination.
3) *Feature Selection by domain knowledge:* Here we have dropped some columns like roll number and email ID as they are irrelevant for predicting student placement in a particular sector. By removing these columns, we reduced the dimensionality of the dataset and improved the computational efficiency of the model thus minimizing overfitting of the model and improving accuracy.
4) *Feature Scaling:* Using MinMaxScaler, we standardized the numerical features. Keeping all of the features on the same scale helped guarantee that no one feature dominated the analysis. A MinMaxScaler instance was initialized first, and it was then fitted to the training data. Then, to ensure consistency, we transformed both the training and testing data using the same scaler.
5) *Splitting data into train and test sets:* We divided the data into training and testing sets, with training data making up 80% of the total. The training set was used to train the machine learning algorithms, while the testing set was used to evaluate their performance.

F. Feature Selection

In this study, we investigated three different feature selection methods: Recursive Feature Elimination (RFE) for Logistic Regression, SelectFromModel for Random Forest Classifier, and SelectKBest for Naïve Bayes to find the most relevant features for predicting the job sector of candidates in campus placements.

Due to two key factors, we decided against fine-tuning the feature selection method and instead used the top 5 features from each technique. First, because they were discovered using a variety of feature selection techniques, the chosen features were highly relevant to the target variable.

Second, we might simplify the model and lower the chance of overfitting by minimizing the number of features. Even though we did not analyze it in our work, feature selection tuning is a crucial subject for future research to improve the efficacy and efficiency of feature selection. SelectKBest, for instance, can be tuned by utilizing a different scoring function or altering the quantity of features that are selected. Similar to how SelectFromModel can be tuned, RFE can also be fine-tuned by experimenting with various parameter settings or by using different feature importance metrics.

G. Splitting Data Into Train And Test Sets

We divided the data into training and testing sets, with training data making up 80% of the total. The training set was used to train the machine learning algorithms, while the testing set was used to evaluate their performance.

H. Model Training

In this study, we trained three different algorithms Random Forest, Logistic Regression, and Naïve Bayes on the training set. Due to their widespread use and successful track record of success in classification problems, these algorithms were chosen. We implemented the algorithms using scikit-learn, a well-known Python library for machine learning and to train the predictive models, we used a Jupyter notebook running Python 3.10 on an Intel Core i7 processor with 16GB of RAM.

I. Hyperparameter Tuning

During model training we explored the process of hyperparameter tuning using grid search for above mentioned classifiers - Logistic Regression, Random Forest, and Naïve Bayes.
For Logistic Regression we used scikit-learn's GridSearchCV class to search the hyperparameter space of the regularization strength (C) and the type of regularization (L1 or L2). We established a range of values for C as [0.001, 0.01, 0.1, 1, 10, 100] and for penalty as ['l1', 'l2']. GridSearchCV was used to grid search the hyperparameter space after we created a Logistic Regression model. Finally, we used our training data to fit the grid search object, and then we used our validation data to choose the hyperparameters that performed the best.
For Random Forest we used scikit-learn's GridSearchCV class to search over the hyperparameter space, including the number of trees (n_estimators), maximum depth of each tree (max_depth), minimum sample requirements for internal node splitting (min_samples_split), and minimum sample requirements for leaf nodes (min_samples_leaf). A range of parameters for n_estimators, max_depth, min_samples_split, and min_samples_leaf were defined as [10, 50, 100, 200], [2, 5, 10, 20, None], [2, 5, 10], and [1, 2, 4]. GridSearchCV was used to grid search the hyperparameter space after creating a Random Forest classifier. Then, after fitting the grid search object to our training data, we choose the hyperparameters that performed the best on our validation data.
For Naïve Bayes using Scikit-Learn's GridSearchCV class, we conducted a search over the smoothing parameter's (alpha) hyperparameter space. A range of alpha values was established as [0.1, 0.5, 1.0, 1.5, 2.0]. We built a Naïve Bayes classifier and used GridSearchCV to grid search the hyperparameter space. We then selected the hyperparameters that performed the best on our validation data after fitting the grid search object to our training data.

J. Model Evaluation

For Logistic Regression, Random Forest and Naïve Bayes, we evaluated the models using accuracy, precision, recall, and F1-score. Overall, the Logistic Regression, Random Forest, and Naïve Bayes classifiers showed good performance after hyperparameter tuning. The Logistic Regression model achieved an accuracy of 0.63, Random Forest achieved an accuracy of 0.67 and Naïve Bayes achieved an accuracy of 0.59.
In addition, we calculated the feature importance of the variables using the feature_importances_ attribute of the Random Forest classifier and it revealed that the top five features were 'hsc', 'quantitative_ability', 'logical_reasoning', 'english_proficiency', 'computer_science_score'.

K. Model Selection And Deployment

We discovered that Random Forest has the highest accuracy rating, followed by Logistics Regression and Naïve Bayes. So, we decide to implement the Random Forest Model.
We developed a Flask server that exposes APIs for predictions and feature importance in order to deploy our model. The probability of being placed in each of the four sectors is returned by the predictions API after receiving a set of features as input.

The trained Random Forest classifier was serialized and saved to a file using the pickle module, which was then loaded into the Flask server using the load method. As a result, we were able to use the trained model to make predictions using fresh data.
We have also developed a feature importance API, which returns the weights assigned to each feature in the Random Forest classifier, in addition to the prediction API. The feature importance were extracted using the Random Forest classifier's ‘feature_importances’ property, and they were then returned as a JSON object.

V. RESULTS

	Random Forest	Logistics Regression	Naïve Bayes
Accuracy	0.67	0.63	0.59
Precision	0.68	0.59	0.51
Recall	0.59	0.56	0.55
F1-Score	0.63	0.57	0.53

We found that the Random Forest algorithm performed the best in terms of accuracy, precision, recall, and F1-score among the three algorithms tested.
The superior performance of the Random Forest model could be attributed to a combination of its ability to handle complex interactions between features, reduce overfitting, and the specific characteristics of the dataset.

VI. CONCLUSION

Our research has demonstrated, in conclusion, that creating a TNP (Training and Placement) portal combining web development and machine learning strategies can have a major positive impact on educational institutions and the students they serve. Overall, implementing a TNP portal through web development and machine learning has the potential to revolutionize how educational institutions approach career guidance and training, ultimately resulting in better employment prospects for students and a stronger workforce for the industry.

REFERENCES

[1] S., Taruna & Pandey, Mrinal. (2014). An Empirical Analysis of Classification Techniques for Predicting Academic Performance. 10.1109/IAdCC.2014.6779379.
[2] Jishan, Syed Tanveer & Rashu, Raisul & Haq, Naheena & Rahman, Mohammad. (2015). Improving accuracy of students’ final grade prediction model using optimal equal width binning and synthetic minority over-sampling technique. Decision Analytics. 2. 10.1186/s40165-014-0010-2.
[3] Saha, Goutam. (2011). APPLYING LOGISTIC REGRESSION MODEL TO THE EXAMINATION RESULTS DATA. Journal of Reliability and Statistical Studies. 4.
[4] Bhatt, Hitarth B., Shraddha Mehta and Lynette R. D'mello. “Use of ID 3 Decision Tree Algorithm for Placement Prediction.” (2015).
[5] D. K. Arun, V. Namratha, B. V. Ramyashree, Y. P. Jain and A. Roy Choudhury, "Student Academic Performance Prediction using Educational Data Mining," 2021 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2021, pp. 1-9, doi: 10.1109/ICCCI50826.2021.9457021.
[6] T. R. Kumar, T. Vamsidhar, B. Harika, T. M. Kumar and R. Nissy, "Students Performance Prediction Using Data Mining Techniques," 2019 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, 2019, pp. 407-411, doi: 10.1109/ISS1.2019.8907945.
[7] Yusuf, A. and Lawan, A., 2018. Prediction of students’ academic performance using educational data mining technique: Literature review.
[8] Fahim, A.M., Salem, A.M., Torkey, F.A. et al. An efficient enhanced k-means clustering algorithm. J. Zhejiang Univ. - Sci. A 7, 1626–1633 (2006). https://doi.org/10.1631/jzus.2006.A1626
[9] E. N. Ogor, "Student Academic Performance Monitoring and Evaluation Using Data Mining Techniques," Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007), Cuernavaca, Mexico, 2007, pp. 354-359, doi: 10.1109/CERMA.2007.4367712.