Campus Placement Prediction

Revision Number: 2.0

Last date of revision: 05/03/2024

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issued | Version | Description | Author |
| 04/03/2024 | 1 | Initial HLD — V1.0 | Jatin Aggarwal |
| 05/03/2024 | 2 | Updated KPI — V1.1 | Jatin Aggarwal |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Document Version Control. 2](#_TOC_250021)

[Abstract. 4](#_TOC_250020)

1. [Introduction 5](#_TOC_250019)
   1. [Why this High-Level Design Document ?. 5](#_TOC_250018)
   2. [Scope. 5](#_TOC_250017)
   3. [Definitions 5](#_TOC_250016)
2. [General Description. 6](#_TOC_250015)
   1. [Product Perspective 6](#_TOC_250014)
   2. [Problem statement 6](#_TOC_250013)
   3. [PROPOSED SOLUTION 6](#_TOC_250012)
   4. [FURTHER IMPROVEMENTS 6](#_TOC_250011)
   5. [Technical Requirements. 6](#_TOC_250010)
   6. [Data Requirements 7](#_TOC_250009)
   7. [Tools used. 8](#_TOC_250008)
      1. Hardware Requirements. 8
      2. ROS(Robotic Operating System) 9
   8. [Constraints 9](#_TOC_250007)
   9. [Assumptions. 9](#_TOC_250006)
3. [Design Details 10](#_TOC_250005)
   1. [Process Flow. 10](#_TOC_250004)
      1. [Model Training and Evaluation 10](#_TOC_250003)
      2. [Deployment Process 11](#_TOC_250002)
   2. [Event log 11](#_TOC_250001)
   3. [Error Handling 11](#_TOC_250000)
   4. Performance. 12
   5. Reusability. 12
   6. Application Compatibility 12
   7. Resource Utilization 12
   8. Deployment. 12
4. Dashboards. 13
   1. KPIs (Key Performance Indicators) 13
5. Conclusion 14

## Abstract

In the modern educational landscape, campus placement plays a vital role in shaping the career trajectory of students. To optimize this process and enhance the efficiency of placements, leveraging machine learning techniques becomes imperative. This project aims to develop a Campus Placement Prediction System using machine learning algorithms.

The proposed system utilizes historical data of students' academic performance, their skill sets, and other relevant attributes to predict the likelihood of their successful placement in companies. Various machine learning models such as logistic regression, decision trees, random forests, and support vector machines are explored to identify the most accurate predictive model.

Furthermore, feature engineering techniques are employed to extract meaningful insights from the dataset, including feature selection, dimensionality reduction, and data normalization. The system also incorporates techniques for handling imbalanced data to ensure fair and unbiased predictions.

The developed system offers several benefits to educational institutions and students alike. For educational institutions, it provides valuable insights into factors influencing placement success, enabling them to tailor curriculum and guidance accordingly. For students, it offers personalized recommendations to improve their employability and increase their chances of securing desirable job placements.

Overall, the Campus Placement Prediction System serves as a powerful tool for optimizing the campus placement process, enhancing transparency, and fostering better outcomes for both students and educational institutions.

**1.Introduction**

###### Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

* + - Present all of the design aspects and define them in detail
    - Describe the user interface being implemented
    - Describe the hardware and software interfaces
    - Describe the performance requirements
    - Include design features and the architecture of the project
    - List and describe the non-functional attributes like: o Security
      * Reliability
      * Maintainability
      * Portability
      * Reusability
      * Application compatibility
      * Resource utilization
      * Serviceability

##### Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

* 1. **Definitions**

Term EDA

*Database*

*IDE AWS*

*Description*

Exploratory Data Analysis

Collection of all the information monitored by this system

Integrated Development Environment

Amazon Web Services

### 

### 2.General Description

#### Product Perspective

Campus Placement Prediction System brings transparency, efficiency, and fairness to the campus placement process, benefiting both educational institutions and students by aligning academic preparation with industry requirements and enhancing opportunities for successful placements.

* 1. **Problem statement**

This project aims to address these challenges by developing a Campus Placement Prediction System using machine learning techniques. The primary goal is to create a predictive model that can accurately forecast students' likelihood of successful placement based on their academic performance, skills, and other relevant attributes. By leveraging historical placement data and advanced machine learning algorithms, the system seeks to provide actionable insights to both students and educational institutions.

Key objectives of the project include:

1. Collecting and preprocessing a comprehensive dataset comprising students' academic - records, skills, internship experiences, and placement outcomes.

2. Exploring various machine learning algorithms and techniques to build predictive models capable of accurately forecasting students' placement probabilities.

3. Conducting rigorous evaluation and validation of the predictive models to ensure their reliability and effectiveness in real-world scenarios.

4. Developing a user-friendly interface for the Campus Placement Prediction System, enabling easy access and utilization by students, faculty, and recruiters.

5. Providing personalized recommendations and guidance to students based on the insights generated by the system, empowering them to enhance their employability and career prospects.

* 1. PROPOSED SOLUTION

The proposed solution involves developing a Campus Placement Prediction System that leverages machine learning algorithms to accurately forecast students' likelihood of successful placement. The system will utilize a comprehensive dataset comprising students' academic records, skills, internship experiences, and placement outcomes to train predictive models.

* 1. FURTHER IMPROVEMENTS

While the proposed solution provides a solid foundation for optimizing the campus placement process using machine learning techniques, several avenues for further improvement and enhancement can be explored:

Incorporating Advanced Algorithms, Ensemble Techniques, Fine-tuning Hyperparameters, Handling Imbalanced Data, Feature Engineering Refinement, Dynamic Updating, Interpretability, User Feedback Integration, Scalability and Efficiency, Ethical Considerations.

#### Technical Requirements

Programming Language: Choose a suitable programming language for implementing the system, such as Python, R, or Java, which offer extensive libraries and frameworks for machine learning and web development.

Machine Learning Libraries: Utilize popular machine learning libraries such as scikit-learn, TensorFlow, Pytorch, or XGboost for model development, training, and evaluation.

Web Development Framework: Select a web development framework like Django, Flask, or Node.js for building the user interface and backend functionality of the system.

Database Management System (DBMS): Choose a relational database management system (RDBMS) such as MySQL, PostgreSQL, or SQLite to store and manage the dataset and user data.

Data Visualization Tools: Integrate data visualization tools like Matplotlib, Seaborn, or Plotly to create interactive visualizations for exploring and analyzing the dataset.

User Interface Design Tools: Use frontend development tools and libraries such as HTML, CSS, JavaScript, Bootstrap, or React.js to design and develop an intuitive user interface for the system.

Deployment Platform: Select a deployment platform such as AWS (Amazon Web Services), Google Cloud Platform, Microsoft Azure, or Heroku for hosting the system and ensuring scalability and reliability.

Version Control: Implement version control using Git and GitHub or GitLab to manage codebase changes, collaborate with team members, and track project progress.

API Integration: Integrate external APIs for data retrieval, such as LinkedIn API for fetching student profiles or job listing APIs for gathering information about available job opportunities.

Security Measures: Implement security measures such as encryption, authentication, and authorization to protect sensitive data and ensure compliance with data privacy regulations.

Documentation: Create comprehensive documentation covering system architecture, installation instructions, API documentation, and user guides to facilitate system deployment and usage.

Testing Frameworks: Employ testing frameworks like pytest, Selenium, or Jasmine for unit testing, integration testing, and end-to-end testing to ensure the reliability and correctness of the system.

Monitoring and Logging: Set up monitoring and logging mechanisms to track system performance, detect errors, and troubleshoot issues in real-time, using tools like Prometheus, Grafana, or ELK Stack.

By fulfilling these technical requirements, you can ensure the successful development, deployment, and maintenance of the Campus Placement Prediction System, delivering a robust and user-friendly solution to optimize the campus placement process.

* 1. **Data Requirements**

Student Information: Collect comprehensive information about students, including demographic details (e.g., name, age, gender), academic background (e.g., GPA, degree program, specialization), and contact information (e.g., email, phone number).

Academic Performance: Gather data on students' academic performance, such as grades, courses taken, credits earned, academic awards, and extracurricular activities.

Skill Sets: Capture information about students' skills and competencies, including technical skills (e.g., programming languages, software tools), soft skills (e.g., communication, teamwork), and domain-specific skills (e.g., finance, marketing).

Internship Experiences: Record details about students' internship experiences, including internship duration, company name, role/title, responsibilities, projects undertaken, and performance evaluations.

Placement Outcomes: Document information about students' placement outcomes, including job offers received, companies selected, salary packages, job roles, and geographical locations.

Historical Placement Data: Acquire historical data on past placements, including placement trends, company profiles, industry sectors, and recruitment patterns, to identify relevant patterns and insights.

External Factors: Consider external factors that may influence placement outcomes, such as economic conditions, industry trends, job market demand, and geographic preferences of students.

Data Privacy and Security: Ensure compliance with data privacy regulations (e.g., GDPR, CCPA) and implement appropriate measures to protect sensitive information, such as anonymization, encryption, and access controls.

Data Quality Assurance: Validate the quality and integrity of the collected data through data cleansing, outlier detection, and error correction techniques to ensure the reliability and accuracy of the analysis.

Data Storage and Management: Establish a robust data storage and management system, using a relational database or data warehouse, to organize, store, and retrieve the data efficiently.

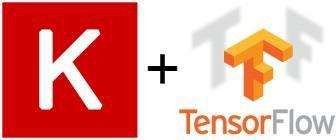
Data Integration and Consolidation: Integrate data from multiple sources, such as student databases, internship portals, and placement records, and consolidate them into a unified dataset for analysis.

Data Governance and Documentation: Implement data governance practices to maintain data consistency, integrity, and lineage, and document data definitions, metadata, and data lineage to facilitate data understanding and interpretation.

By fulfilling these data requirements, you can ensure the availability of high-quality data for developing and training predictive models in the Campus Placement Prediction System, enabling accurate forecasting of students' placement probabilities and informed decision-making by stakeholders.

* 1. Tools used

Python programming language and frameworks such as NumPy, Pandas, Scikit-learn, TensorFlow, Keras and Roboflow are used to build the whole model.





[This Photo](https://code.makery.ch/library/html-css/part1/) by Unknown Author is licensed under [CC BY](https://creativecommons.org/licenses/by/3.0/)



[This Photo](https://adrin.info/gitgithub-how-to-contribute-to-an-open-source-project-on-github.html) by Unknown Author is licensed under [CC BY-SA](https://creativecommons.org/licenses/by-sa/3.0/)

* + - VS code is used as IDE.
    - For visualization of the plots, Matplotlib, Seaborn and Plotly are used.
    - AWS is used for deployment of the model.
    - Tableau/Power BI is used for dashboard creation.
    - MySQL/MongoDB is used to retrieve, insert, delete, and update the database.
    - Front end development is done using HTML/CSS
    - Python Django is used for backend development.
    - GitHub is used as version control system.
    1. Hardware Requirements
       - RAM-4GB
       - SSD-better
       - PC (check you are system supports: https://7dfps.com/ros-system- requirements/)
       - Nice Battery Backup
       - Good Performance



* + 1. ROS (Robotic Operating System)

Robot Operating System is an open-source robotics middleware suite. Although ROS is not an operating system but a collection of software frameworks for robot software development, it provides services designed for a heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes, and package management.

#### Constraints

The Campus Placement Prediction solution system must be user friendly, as automated as possible and users should not be required to know any of the workings.

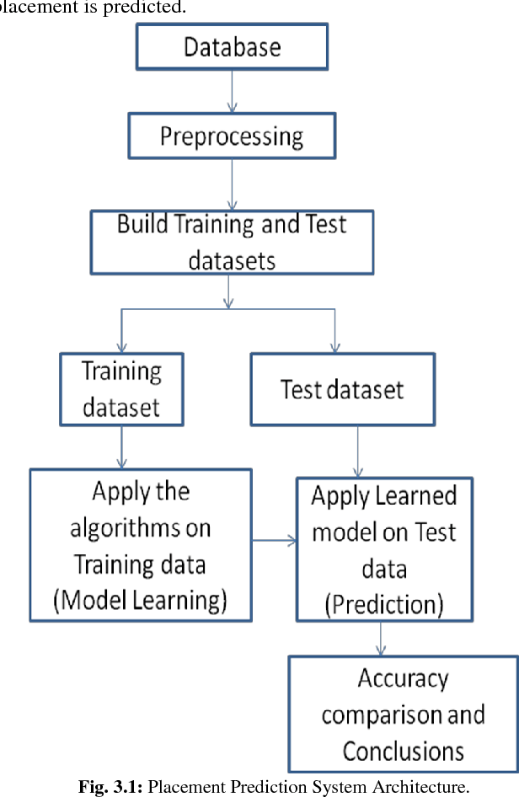
#### Assumptions

The main objective of the project is to implement the use cases as previously mentioned (2.2 Problem Statement) for new dataset that comes through Kaggel which has Previous Studies and other details for Predicting better results. Machine Learning based Prediction model is used for Predicting the above-mentioned use cases based on the input data. It is also assumed that all aspects of this project have the ability to work together in the way the designer is expecting.

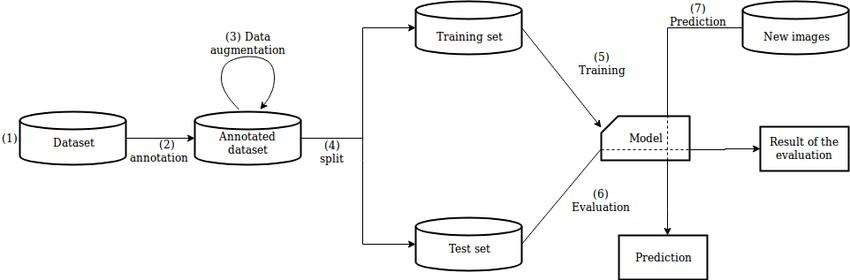
## 2.Design Details

##### Process Flow

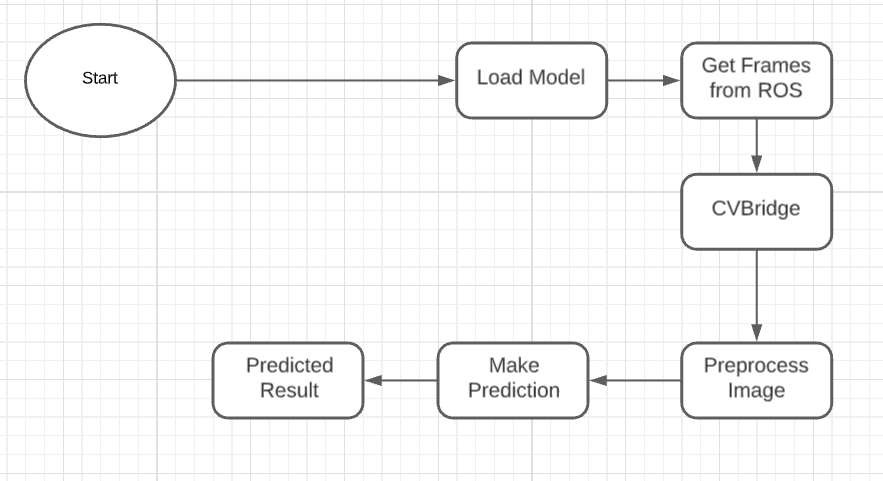
For identifying the different types of anomalies, we will use a deep learning base model. Below is the process flow diagram is as shown below.

Proposed methodology 

##### Model Training and Evaluation



##### Deployment Process



* 1. Event log

The system should log every event so that the user will know what process is running internally.

Initial Step-By-Step Description:

1. The System identifies at what step logging required
2. The System should be able to log each and every system flow.
3. Developer can choose logging method. You can choose database logging/ File logging as well.
4. System should not hang even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.
   1. **Error Handling**

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

## Performance

The Campus Placement solution is used for Prediction whenever companies come to hire students. it will used to predict the placements and takes necessary action, so it should be as accurate as possible. So that it will not mislead the concern authorities (like companies, universities, etc..). Also, model retraining is very important to improve the performance.

#### Reusability

The code written and the components used should have the ability to be reused with no

problems.

#### Application Compatibility

The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the Python to ensure proper transfer of information.

#### Resource Utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

#### Deployment



1. Dashboards

Dashboards will be implemented to display and indicate certain KPls and relevant indicators for the unveiled problems that if not addressed in time could cause catastrophes of unimaginable impact.



As and when, the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors.

#### KPls (Key Performance Indicators)

Key Performance Indicators (KPIs) for a Campus Placement Prediction System:

Prediction Accuracy: Measure the accuracy of placement predictions generated by the system compared to actual placement outcomes. KPIs under this category include:

Accuracy: The percentage of correct predictions out of total predictions made by the system.

Precision: The ratio of true positive predictions to the total predicted positives, indicating the system's ability to avoid false positives.

Recall: The ratio of true positive predictions to the total actual positives, indicating the system's ability to capture all positive instances.

F1-score: The harmonic mean of precision and recall, providing a balanced measure of prediction accuracy.

Model Performance: Evaluate the performance of machine learning models used in the system. KPIs under this category include:

Model Accuracy: The accuracy of individual machine learning models in predicting placement outcomes.

Training Time: The time taken to train the models on the dataset, indicating computational efficiency.

Inference Time: The time taken to generate predictions for new data, indicating the system's responsiveness.

User Satisfaction: Assess the satisfaction level of users interacting with the system. KPIs under this category include:

User Feedback Score: Feedback collected from users regarding the system's usability, effectiveness, and overall satisfaction.

System Response Time: The time taken for the system to respond to user inputs, indicating responsiveness and performance.

Fairness and Bias: Measure the fairness and bias of the predictive models in the system. KPIs under this category include:

Bias Metrics: Metrics such as disparate impact, equal opportunity difference, and statistical parity difference to assess bias in predictions across different demographic groups.

Fairness Scores: Scores indicating the system's adherence to fairness principles and mitigation of biases in prediction outcomes.

System Scalability: Evaluate the system's ability to scale and handle increasing volumes of data and user requests. KPIs under this category include:

Throughput: The number of predictions generated by the system per unit time, indicating its processing capacity.

Resource Utilization: The utilization of hardware resources such as CPU, memory, and storage, indicating scalability and efficiency.

Regulatory Compliance: Ensure compliance with data privacy regulations and ethical standards. KPIs under this category include:

Compliance Score: A score indicating the system's compliance with regulations such as GDPR, CCPA, and FERPA.

Data Privacy Measures: Measures implemented to protect user privacy, such as data encryption, anonymization, and access controls.

Placement Success Rate: Measure the success rate of placements facilitated by the system. KPIs under this category include:

Placement Rate: The percentage of students successfully placed in job roles recommended by the system.

Placement Time: The time taken for students to secure job placements after interacting with the system..

## Conclusion

Throughout this project, key considerations such as data quality, model accuracy, user experience, fairness, and regulatory compliance have been addressed to ensure the system's effectiveness and ethical integrity. By adhering to best practices in data collection, preprocessing, model development, and deployment, the system can deliver accurate predictions of students' placement probabilities while mitigating biases and safeguarding user privacy.

Moreover, the continuous monitoring, evaluation, and improvement of the system, guided by key performance indicators (KPIs) and user feedback, enable stakeholders to adapt and refine the system over time to meet evolving needs and requirements. By fostering collaboration and engagement among students, faculty, recruiters, and administrators, the system can facilitate better decision-making, optimize resource allocation, and ultimately enhance placement outcomes for all parties involved.

In essence, the Campus Placement Prediction System represents a significant step forward in modernizing and optimizing the campus placement process, aligning academic preparation with industry demands, and empowering students to achieve their career aspirations. As technology continues to evolve and new opportunities emerge, the system stands poised to make a meaningful impact in shaping the future of education and employment..

## References

1. Google.com
2. [kaggel](http://www.ros.org/).com
3. Github.com

4.[(PDF) Campus Placements Prediction & Analysis using Machine Learning (researchgate.net)](https://www.researchgate.net/publication/360130609_Campus_Placements_Prediction_Analysis_using_Machine_Learning)