Low-Level Document

Campus Placement Prediction

Written By	TRISHIT NATH THAKUR
Version	1.2
Date	25-06-2023

Document Change Control Record

Version	Date	Author	Comments
1.0	7/6/23	TRISHIT	Defined the architecture
1.1	14/6/23	TRISHIT	Architecture definition added
1.2	23/6/23	TRISHIT	Unit test Cases Added and defined

Index

1. Introduction

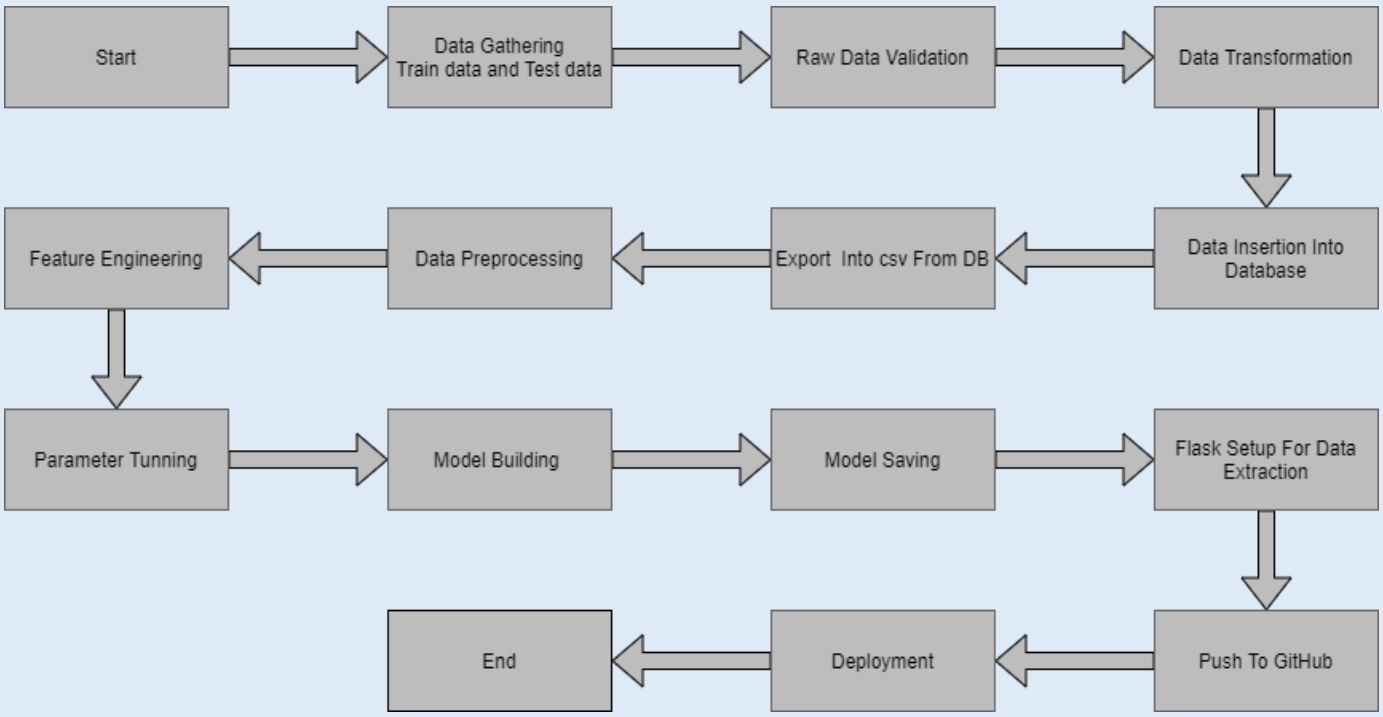
1.1 What is Low-Level Design Document.

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for **'Campus Placement Prediction'**. LLD describes the class diagrams with themethods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

1.2 Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required softwarearchitecture, source code, and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

2. Architecture :



2. Architecture Description

2.1 Data Description

File descriptions

- train.csv - the training set
- test.csv - the test set
- SampleSubmission.csv - a sample submission file in the correct format

Data fields

- gender - sex of the student
- secondary education percentage-marks obtained in secondary education
- higher secondary percentage-marks obtained in higher secondary education
- degree percentage-marks obtained in degree
- Under-graduation(Degree-type)-Field of degree education
- Work-experience • Employability-test-package
- specialization-field of study

Name	Data Type	Measurement
Gender	Integer	Gender of student
ssc_p	Float	Ssc percentage
ssc_b	Object	Ssc board
hsc_p	Float	Hsc percentage
Hsc_b	Object	Hsc board
Hsc_s	Object	Hsc stream
Degree_p	Float	Degree percentage
Degree_t	Object	Graduation stream
Workex	Object	Are they having any work experience
Etest_p	Float	Online test percentage
Specialisation	Object	Specialization the choosed in Mba
Mba_p	Float	Mba percentage
Status	Object	Are they placed or not placed. This the outcome column.

2.2 Data Transformation

In our dataset a lot of categorical values are present, we transformthose attributes into numerical values using one hot encoding. A one hot encoding is a representation of categorical variables as binary vectors. This first requires that the categorical values be mapped to integer values. Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1.

2.3 Data Preprocessing

Data preprocessing is a technique that is used to convert raw data into a clean dataset. The data is gathered from different sources is in raw format which is not feasible for the analysis. Pre-processing for this approach takes 4 simple yet effective steps. Attribute selection some of the attributes in the initial dataset that was not pertinent (relevant) to the experiment goal were ignored. The attributes name, roll no, credits, backlogs, whether placed or not, b.tech %, gender are not used. The main attributes used for this study are credit, back-logs, whether placed or not, b.tech %. Campus Placement Prediction Cleaning missing values in some cases the dataset contain missing values. We need to be equipped to handle the problem when we come across them. Obviously you could remove the entire line of data but what if you're inadvertently removing crucial information? After all we might not need to try to do that. One in every of the foremost common plan to handle the matter is to require a mean of all the values of the same column and have it to replace the missing data. The library used for the task is called Scikit Learn preprocessing. It contains a class called Imputer which will help us take care of the missing data. to split our dataset into two. Training set and a Test set. We will train our machine learning models on our training set, i.e our machine learning models will try to understand any correlations in our training set and then we will test the models on our test set to examine how accurately it will predict. A general rule of the thumb is to assign 80% of the dataset to Training and Test data splitting the Dataset into Training set and Test Set Now the next step is training set and therefore the remaining 20% to test set. Feature scaling the final step of data preprocessing is feature scaling. But what is it? It is a method used to standardize the range of independent variables or features of data. But why is it necessary? A lot of machine learning models are based on Euclidean distance. If, for example, the values in one column (x) is much higher than the value in another column (y), (x2-x1) squared will give a far greater value than (y2-y1) squared. So clearly, one square distinction dominates over the other square distinction.

2.4 Feature Engineering

After preprocessing it was we saw that there are few categorical columns present in the data. For converting the categorical column, I have created the column transformer object and inside it I performed one hot encoding on categorical columns. Afterwars I save the transformer object so that it can be used for transforming the test and prediction data in same way as train data. I have converted the target column categorical values into numerical using label encoder.

To handle imbalance data we have performed oversampling using imblearn smote function. In the same step we also performed feature scaling to bring down every feature on the same page in terms of value range.

2.5 Data Clustering

The campus placement activity is incredibly vital from institution point of view as well as student point of view. In this regard to improve the student’s performance, a dataset has

been analyzed and predicted using the classification algorithms like KNN algorithm, Random Forest, Naïve Bayes, Logistic Regression, Decision Tree and the SVM algorithm to validate the approaches. Models used: KNN ALGORITHM- K Nearest Neighbor (KNN) is intuitive to understand and an easy to implement the algorithm. It is a versatile algorithm also used for imputing missing values and resampling datasets. Random Forest- Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. o Naïve Bayes- Naïve Bayes is a probabilistic machine learning algorithm based on the Bayes Theorem, used in a wide variety of classification tasks. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. o Logistic Regression- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. o Decision Trees- Decision trees use multiple algorithms to decide to split a node into two or more subnodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes. o SVM Algorithm- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. o The goal of the SVM algorithm is to create the best line or decision boundary that can segregate ndimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

2.6 Parameter Tuning

I have used sklearn pipeline library with grid search cv to peform hyper parameter tuning. I have used different algorithms in pipeline and set different parameters for their important attributes.

2.7 Model Building

After doing all kinds of preprocessing operations mention above and performing scaling and hyperparameter tuning, the data set is passed into 3 models, SVC, XGB and Random Forest. It was found that Radom forest classifier performs best with the highest accuracy score equals 0.89. So ‘Random forest classifier’ performed well in this problem.

2.8 Model Saving

Model is saved using pickle library.

2.9 Django Setup for Data Extraction

After saving the model, the API building process started using Flask. Web application creation was created here. Whatever the data user will enter and then that data will be extracted by the model to predict the prediction of placement status, this is performed in this stage.

2.11 GitHub
The whole project directory will be pushed into the GitHub repository.

2.12 Deployment
The cloud environment was set up and the project was deployed from GitHub into the Heroku cloud platform.

3. Unit Test Cases.

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URLis accessed	1.Application URLis accessible 2.Application is deployed	The Application should load completely for the user when theURL is accessed
Verify whether the User is able to sign up in the application	1.Application is accessible	The User should be able to sign up in the application
Verify whether a user is able to enter the input values.	1.Application is accessible	User should be able to fill the input field
Verify whether a user gets predict button to submit the inputs	1.Application is accessible	Users should get Submit button to submit the inputs
Verify whether a user is presented with recommended results on clicking submit	1. Application is accessible 2.The user is able to see the submit button	Users should be presented with recommended results on clicking submit
Verify whether a result is in accordance with the input that the user has entered	1.Application is accessible 2.The user is able to see the submit button	The result should be in accordance with the input that the user has entered