**Campus Placement Prediction: Project Report**

**1. Dataset Selection**

**Dataset Source:**  
The dataset was obtained from Kaggle: Factors Affecting Campus Placement. It contains detailed information about student academic records and placement outcomes.

**Target Variable:**  
The goal is to predict whether a student will be placed or not placed (categorical classification).

**2. Data Preprocessing**

The preprocessing of the data was carried out carefully so that the dataset would be ready to use in machine learning models. It included exploratory analysis, missing value treatment, encoding categorical variables, and data splitting for training and testing.

2.1 Exploratory Data Analysis (EDA)

The data was initially explored using descriptive statistics (df.describe()) and data type summaries (df.info()). Distribution plots were generated for significant numerical features such as secondary and higher secondary percentages (ssc\_p, hsc\_p), degree percentage (degree\_p), and MBA percentage (mba\_p). These provided a sense of trends, outliers, and skewness in the data. A correlation heatmap was also constructed using seaborn to highlight the associations between numerical features, indicating which variables were most associated with placement status.

2.3 Encoding Categorical Features

Categorical features such as gender, workex, and status were label-encoded using LabelEncoder, converting them into numerical form suitable for machine learning models. The status column (the target variable) was encoded such that "Placed" became 1 and "Not Placed" became 0. Multi-class categorical features like hsc\_b, hsc\_s, degree\_t, and specialisation were transformed using one-hot encoding (pd.get\_dummies()), and drop\_first=True was applied to avoid the dummy variable trap.

2.4 Train-Test Split

The data processed was split into a train and test set in a 70:30 ratio with a fixed random seed to ensure reproducibility. The features (X) and the target (y) were separated and the train\_test\_split function from the library sklearn was utilized to obtain X\_train, X\_test, y\_train, and y\_test.

This structured preprocessing process ensured that data fed into the models was clean, well-balanced, and properly formatted, laying a strong foundation for accurate and fair model training.

3. Model Selection

Three machine learning models for classification were selected for the prediction of whether a student would go into campus recruitment or not. All three models were selected on the grounds that they were applicable to the task at hand, and hyperparameters had been optimized for improved performance on the dataset.

3.1 Models Selected and Reasoning

1. Logistic Regression

Logistic Regression was chosen as a baseline because it is simple to interpret, simple, and effective in binary classification problems. It is used to provide a basis with which the performance of more complex models can be compared.

2. Random Forest Classifier

Random Forest is a collection of ensemble algorithms that build a number of decision trees and combine their predictions to produce stable predictions. It is able to capture linear and non-linear relationships and is less prone to overfitting than single decision trees. It is best suited for structured/tabular data like this one.

1. Support Vector Machine (SVM)  
   SVM is a robust classifier that constructs a hyperplane to divide classes with maximum margin. SVM performs well in high-dimensional space and is especially well-suited for well-separated cases, which is likely in this data set

3.2 Hyperparameter Adjustment

Random Forest and SVM parameters were adjusted with grid search or manual parameter adjustment to achieve best performance:

\tIdeal Random Forest Parameters:

n\_estimators = 100 (number of trees in the forest)

max\_depth = None (trains trees until all the leaves become pure)

•\tBest SVM Parameters:

C = 0.1 (regularization strength; lower values imply higher regularization)

kernel = 'linear' (separates classes using a linear boundary)

4.\tModel Training

The models selected—Logistic Regression, Random Forest, and Support Vector Machine (SVM)—were trained on the dataset using a clear and systematic approach. Each of the models was trained using the training set upon data splitting, and hyperparameter tuning was executed for the complex models to improve generalization as well as performance.

4.1 Data Preparation for Training

The features (X) were prepared by removing the target column status and the id sl\_no from the data. The target variable (y) was encoded status, where 1 represented those who were placed and 0 for not being placed.

A stratified train-test split was performed on a 70:30 ratio, maintaining the class distribution constant across both subsets:

• Training set size: 150 samples

• Testing set size: 65 samples

• Target distribution: 148 placed, 67 not placed

python

CopyEdit

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.3, stratify=y, random\_state=42

)

**4.2 Model Training and Hyperparameter Tuning**

Three classifiers were trained on the training set:

• Logistic Regression:

Trained with default parameters and max\_iter=1000 for convergence.

• Random Forest Classifier:

Grid search with 5-fold cross-validation was used to tune n\_estimators and max\_depth.

• Support Vector Machine (SVM):

Grid search was used to tune the penalty term C and kernel type.

4.3 Summary of Training Process

• All models were trained successfully on the training set.

• GridSearchCV was used for Random Forest and SVM to determine optimal parameters using 5-fold cross-validation.

• Saved models for evaluation and used them later in the Voting Classifier.

• Training code was modular, commented, and sensibly organized, simple to trace and reuse.

This intense training process guaranteed that every model was well-trained for evaluation and ensemble modeling, contributing to the solution's overall strength.

**5. Model Evaluation**

Evaluated the models on the test set with the following metrics:

• Accuracy

• Precision

• Recall

• F1-Score

Confusion Matrices were plotted for each of the models to visualize actual vs. predicted labels. What follows is a summary of model results:

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Logistic Regression | ~86% | High | Good | Good |
| Random Forest | ~88% | Very High | High | Very Good |
| SVM | ~84% | Balanced | Good | Moderate |

**6. Voting Classifier**

In order to enhance model performance and stability, the Voting Classifier was employed as an ensemble of the above three selected and tuned models: Logistic Regression, Random Forest, and Support Vector Machine (SVM).

6.1 Implementation

Soft voting was employed, which computes the mean of the predicted class probabilities (and not the final labels) of all the base estimators and makes a final decision based on the mean of these probabilities. Soft voting is generally better than hard voting, especially if classifiers are well-calibrated.

The voting classifier was constructed on the best estimator from each model following hyperparameter tuning:

• Logistic Regression (default)

• Random Forest (n\_estimators=100, max\_depth=None)

• SVM (C=0.1, kernel='linear')

6.2 Evaluation and Comparison

Standard classification metrics were used to test the Voting Classifier on the test set. The following are the results:

• Accuracy: 0.80

• Precision: 0.81

• Recall: 0.93

• F1 Score: 0.87

These findings indicate that the Voting Classifier achieves very good recall and well-balanced overall performance, doing better or almost as good as the best performing stand-alone models in several categories. Especially, by high recall, it is indicated that the ensemble model does exceptionally well at recognizing correctly the students to be assigned.6.3 Comparison to Individual Models

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| Logistic Regression | ~0.86 | - | - | - |
| Random Forest | ~0.88 | - | - | - |
| SVM | ~0.84 | - | - | - |
| **Voting Classifier** | **0.80** | **0.81** | **0.93** | **0.87** |

While Voting Classifier was less accurate than Random Forest, it was more accurate in recall and F1 measure when it matters more that false negatives (students predicted to not be placed but indeed being placed) are more costly.

Conclusion

This project aimed to develop a predictive model to determine whether a student would be assigned to campus recruitment or not depending on various academic and personal attributes. The data set, which was acquired from Kaggle, contained 215 records with corresponding attributes such as academic performance, gender, specialization, and work experience.

Comprehensive preprocessing was done; missing value handling; categorical variable encoding; splitting the data into training and test sets using stratification for class balance.

Three well-justified selection criteria of the classification model: logistic regression due to its simplicity and interpretability; random forest due to ensemble strength and ability to model complex feature interplays; Support Vector Machine due to effectiveness in handling high-dimensional data.

All of the models were trained over training data and hyperparameter optimized using grid search. The models were evaluated using regular classification metrics such as accuracy, precision, recall, and F1-score. To leverage the best of different models, a Voting Classifier was used. It showed high overall performance, particularly on recall (0.93) and F1-score (0.87), and hence was an effective ensemble solution.

In short, the models were extensively tested and trained, with the Voting Classifier proving superior to single models on most measures. The report and code are clearly organized and documented and present an accurate and scalable way of predicting campus placement results in educational settings.