

Assignment 2: Campus Placement

Overview:

The goal of this project is to predict whether a student will be placed in a job based on academic performance, work experience, and other factors. We will build and evaluate three different classification models to determine the best-performing model.

Dataset:

The dataset contains student records with the following key attributes:

- Academic Performance: ssc_p (Secondary Education %), hsc_p (Higher Secondary %), degree_p (Degree %), mba_p (MBA %).
- Education Background: ssc_b (Board type), hsc_b (Board type), hsc_s (Specialization), degree_t (Degree type).
- Work Experience & Skills: workex (Yes/No), etest_p (Employability Test Score), specialisation (MBA specialization).
- Target Variable: status (Placed/Not Placed).
- Salary: salary - available only for placed students.

Data Preprocessing:

1. **Dropped unnecessary columns:** Dropped 'sl_no' and 'salary' columns as salary is dependent on our dependent variable 'status'.
2. **Checking Null values:** there are no null values in our dataset.

```
# checking Null values  
df.isnull().sum().sort_values(ascending=True)
```

```
gender      0  
ssc_p       0  
ssc_b       0  
hsc_p       0  
hsc_b       0  
hsc_s       0  
degree_p    0  
degree_t    0  
workex      0  
etest_p     0  
specialisation 0  
mba_p       0  
status      0  
dtype: int64
```

3. Encoding the Dependent Variable (status)

Since the target variable status is categorical with two unique values: "Placed" and "Not Placed", I applied Label Encoding to convert it into a numerical format. This transformation is essential for machine learning models, as they require numerical inputs.

We used Scikit-Learn's LabelEncoder to perform this encoding, where:

"Not Placed" -> 0

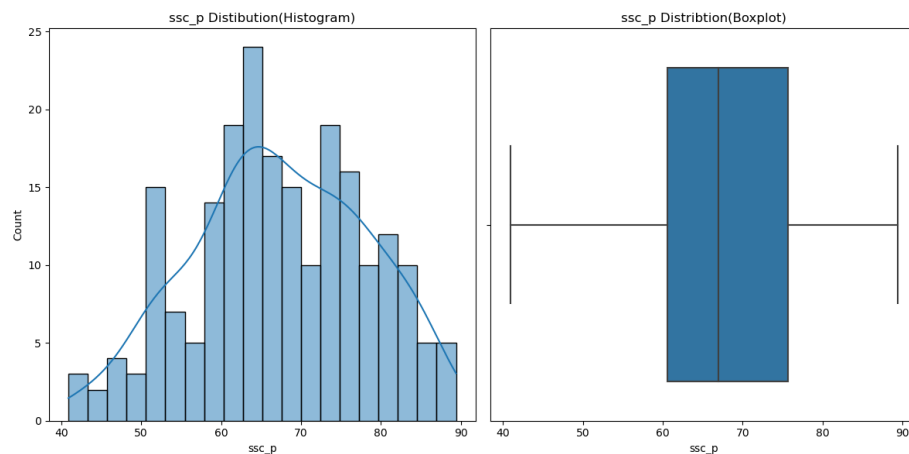
"Placed" -> 1

4. Exploratory Data Analysis:

- **Distribution of continuous variables**

Plotting the histogram with KDE and boxplot to visualize the distribution of continuous variables in the dataset, where histogram with KDE helps understand the overall distribution of a variable and boxplot identifies outliers and spread of data.

i. **ssc_p (Secondary School Percentage):**

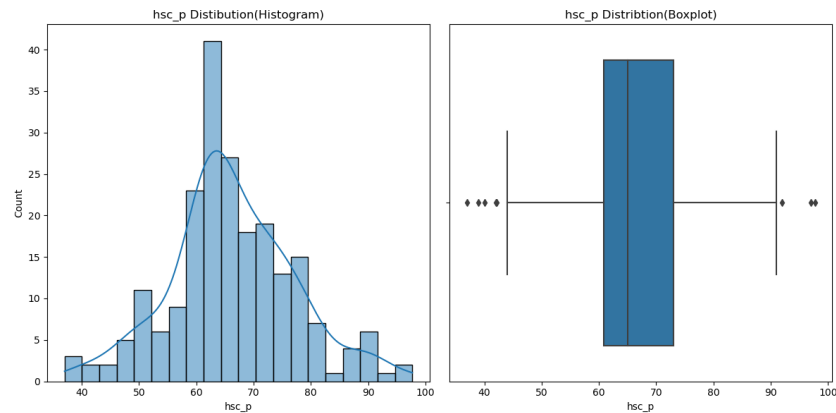


Histogram: The distribution appears roughly normal, centered around 60–70%, with a smooth peak near 65%.

Boxplot: The spread is symmetric, with no apparent outliers. The data is concentrated between 60 and 70, indicating consistent student performance.

Summary: Most students score within 60–70%, reflecting a generally uniform performance at the secondary school level.

ii. hsc_p (Higher Secondary Percentage):

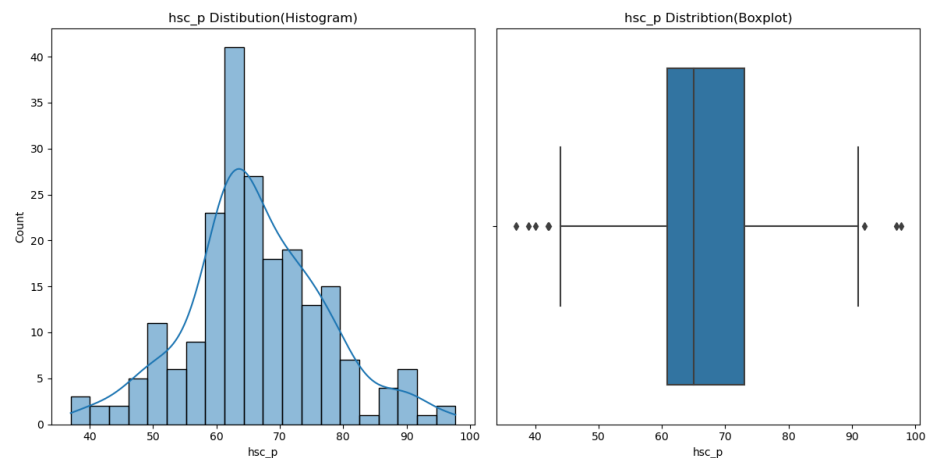


Histogram: The distribution shows a slight right skew, with most values falling between 60 and 70%. The peak occurs near 65%.

Boxplot: The median is around 65%, and the IQR spans from approximately 60 to 70. A few outliers are visible at both ends.

Summary: The higher secondary percentage indicates consistent performance, with some variability and a small number of outliers.

iii. degree_p (Degree Percentage):

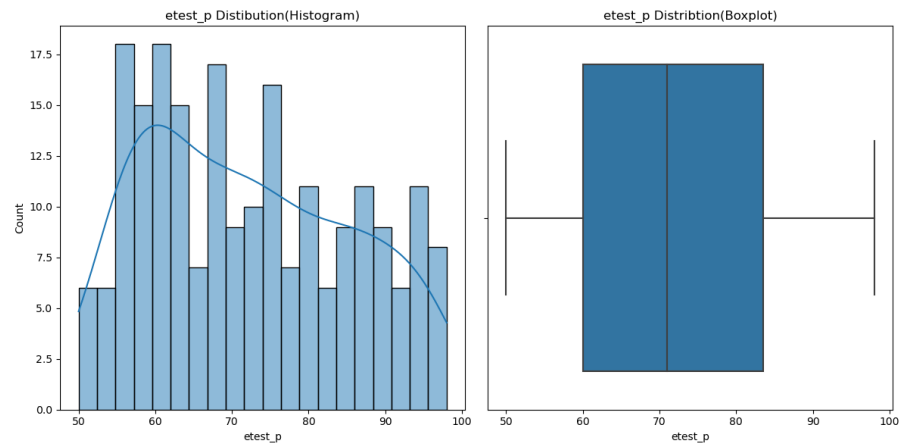


Histogram: The distribution is slightly skewed to the right, with most values clustered between 60 and 70%. A peak is observed around 65%.

Boxplot: The spread is similar to the histogram, with a median near 65%. A single outlier exists on the higher end, but the distribution is otherwise balanced.

Summary: Degree scores are concentrated around 65%, with minimal variation and one high outlier.

iv. **etest_p (Employability Test Percentage):**

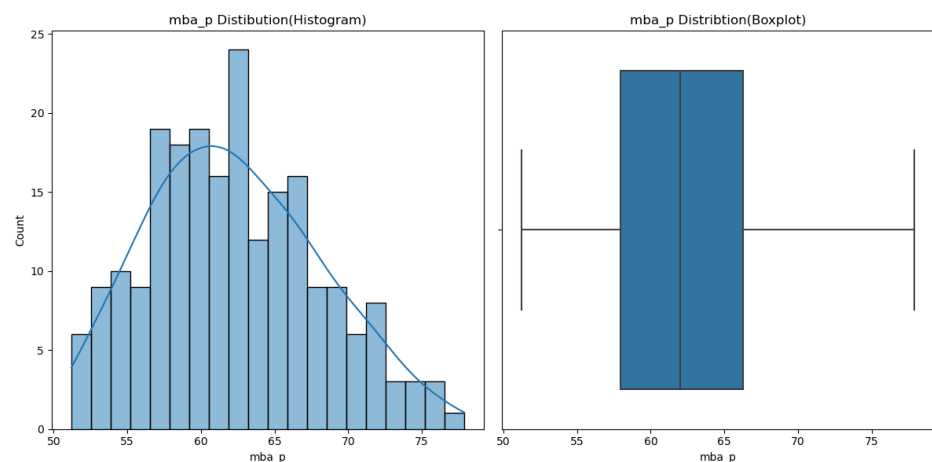


Histogram: The data is slightly skewed, with the majority of values concentrated between 50 and 80. A peak is observed near 60–70%.

Boxplot: The median is close to 70%, and the IQR spans from approximately 60 to 80. The whiskers extend symmetrically with no apparent outliers.

Summary: The employability test scores show a balanced distribution, with most values falling between 60 and 80%.

v. **mba_p (MBA Percentage):**



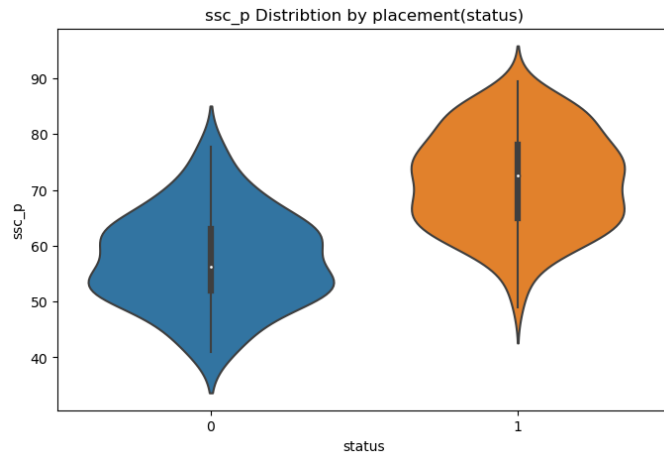
Histogram: The distribution is slightly skewed to the right, with most values concentrated between 55 and 70%. A peak occurs around 60–65%.

Boxplot: The median is near 62–65%, and the IQR spans 58 to 70. The whiskers extend without significant outliers, suggesting a well-distributed range.

Summary: MBA scores are centered around 60–65%, with slight skewness and a balanced spread.

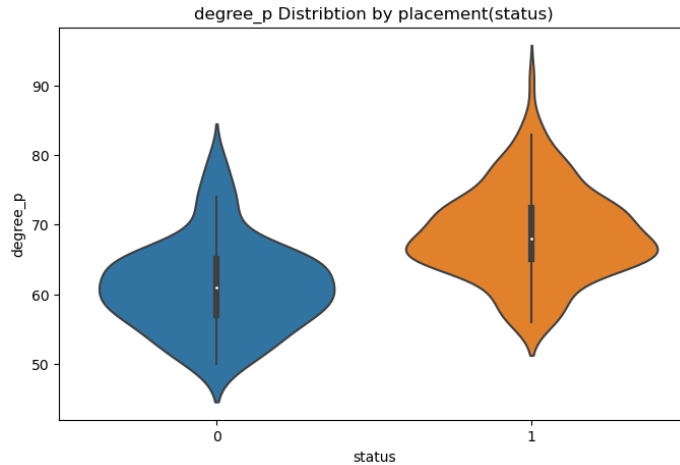
- **Summary of Placement Analysis**

**Violin Plot Analysis: Academic Performance & Placement:
High School Percentage (hsc_p):**



- Placed Students (status=1): Higher and more stable scores, showing that good academic performance helps in placement.
- Not Placed Students (status=0): Scores vary widely, with lower averages, suggesting that inconsistent or low performance may reduce placement chances.

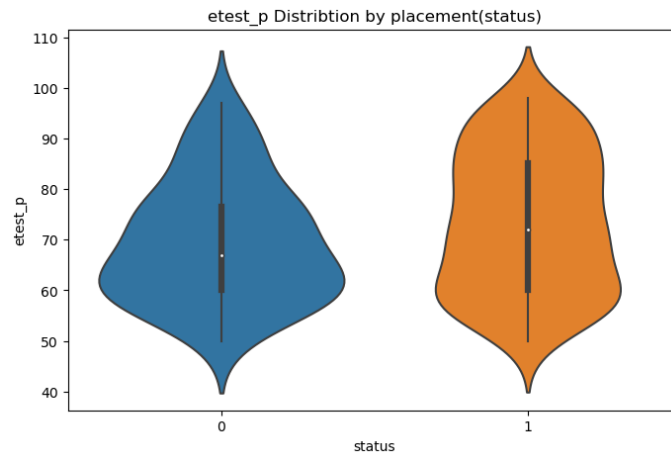
Degree Percentage (degree_p):



- Placed Students: Higher and more consistent scores, showing a strong link between degree performance and placement.
- Not Placed Students: More variation in scores with a lower average, meaning inconsistent academic performance could be a disadvantage.

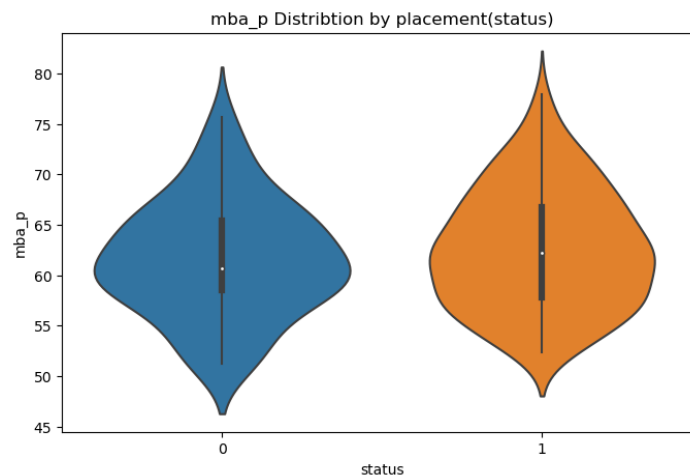
Violin Plot Analysis: Employability & MBA Performance

Employability Test Percentage (etest_p):



- Strong connection to placement—students with higher scores are more likely to be placed.
- Students who were not placed have lower and more varied scores.

MBA Percentage (mba_p):



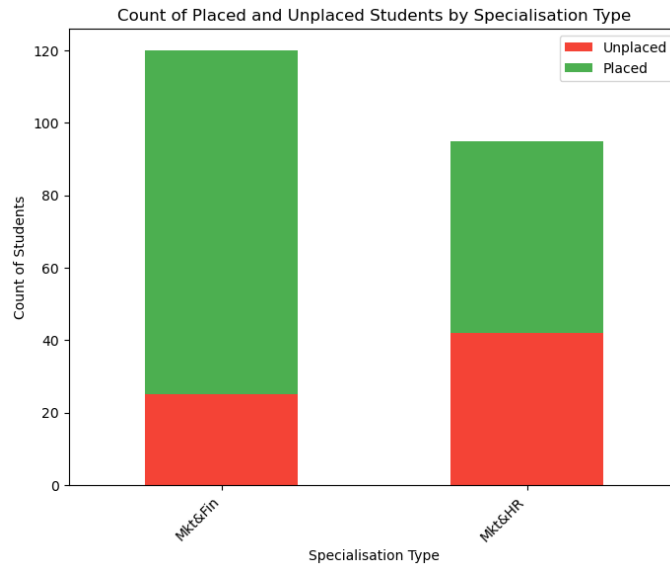
- Little difference between placed and not placed students, meaning MBA performance does not strongly impact placement chances.

Key Insights from Violin plots:

- Better academic performance in high school and degree studies increases placement chances.
- Employability test scores are an important factor in getting placed.

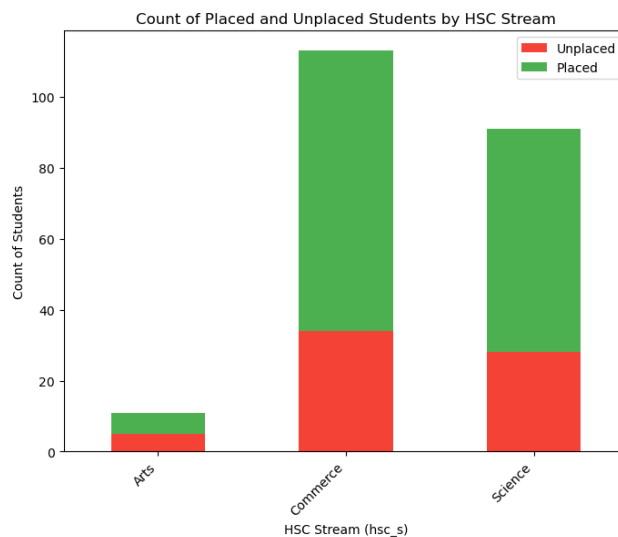
- MBA scores have less impact, suggesting companies focus more on degree and employability test results.

- **Count of placed and unplaced students by specialisation type**



It can be clearly seen that the ratio of students getting placed is higher in 'Mkt&Fin' with approximately 119 placed students out of 141 (119/22), compared to 'Mkt&HR' where the ratio is lower with 95 placed students out of 135 (95/40). This indicates that students specializing in Marketing & Finance have a higher placement rate compared to those specializing in Marketing & Human Resources.

- **Count of placed and unplaced student by HSC stream**

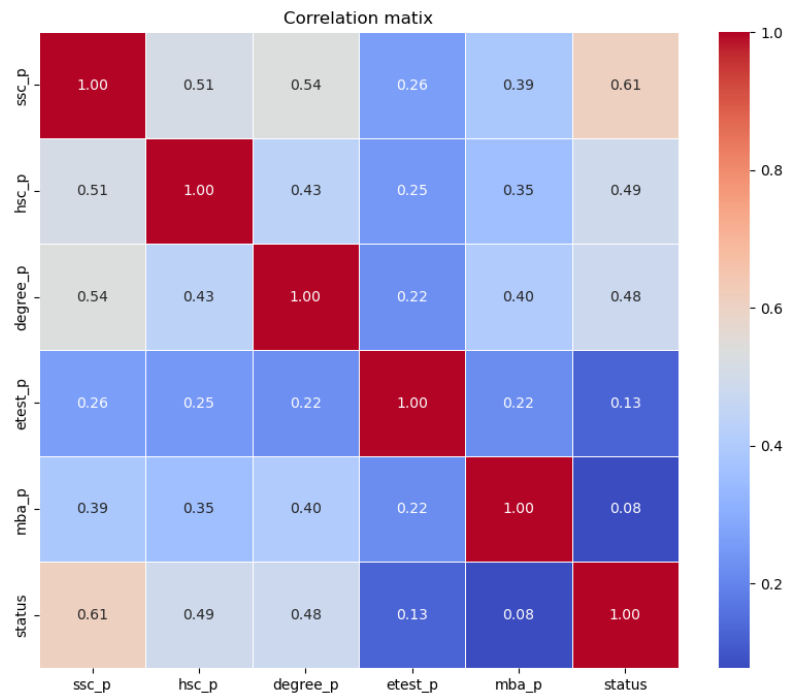


The placement ratio of students across different HSC streams shows varying trends:

- Arts Stream: The placement ratio is 1:1, indicating an equal number of placed and unplaced students in this stream.
- Commerce Stream: Students in the Commerce stream have a higher placement ratio of approximately 3.14:1, suggesting that placed students outnumber unplaced students by more than three times.
- Science Stream: The placement ratio for Science students is 3:1, demonstrating a stronger likelihood of placement compared to the Arts stream.

These ratios indicate that students from the Commerce and Science streams have significantly higher chances of placement compared to their peers in the Arts stream.

- **Correlation Matrix**



- Academic performance (ssc_p, hsc_p, and degree_p) is more strongly associated with placement than employability tests (etest_p) or MBA percentages (mba_p).
- ssc_p is the most significant individual factor influencing placement.

5. Encoding and Scaling the data using Column Transformer

```
# Column Transforming the data

num_cols = ['ssc_p', 'hsc_p', 'degree_p', 'etest_p', 'mba_p']
cat_cols = ['ssc_b', 'hsc_b', 'hsc_s', 'degree_t', 'workex', 'specialisation']

col_trans = ColumnTransformer(transformers=[('OneHot', OneHotEncoder(drop='first'), cat_cols),
                                           ('Standard', StandardScaler(), num_cols)], remainder='passthrough')
# remainder = passthrough means that remaining undefined cols will pass
```

Categorical Columns Transformation:

- **OneHotEncoder(drop='first')**: This transformer is applied to categorical columns (ssc_b, hsc_b, hsc_s, degree_t, workex, specialisation).
drop='first': This argument ensures that the first category for each categorical feature is dropped, avoiding the dummy variable trap (multicollinearity).

Numerical Columns Transformation:

- **StandardScaler()**: This is applied to numerical columns (ssc_p, hsc_p, degree_p, etest_p, mba_p) to standardize the values by removing the mean and scaling to unit variance.

6. **Splitting the dataset into Train/Test**: The dataset was split into an 80:20 ratio, where 80% of the data was used for training and 20% for testing, ensuring a proper evaluation of the model on unseen data. To improve model reliability and generalization, k-fold cross-validation with 5 folds (cv=5) was applied. This technique splits the training data into 5 subsets, training the model on 4 and testing on the remaining one, repeating the process for better performance estimates. This approach maximizes data usage and reduces the risk of overfitting.

Model Selection

Logistic Regression:

Simplicity: Logistic regression is a simple model, making it ideal for small datasets. It doesn't require a large amount of data to produce reliable results and can handle binary classification well.

Efficiency: It's computationally efficient and fast to train, which is beneficial when dealing with smaller datasets where you don't need overly complex models that may overfit.

Interpretability: Logistic regression produces coefficients that show the direct influence of each feature on the outcome, making it easier to understand and interpret, especially when the number of features is small.

Decision Tree:

Non-Linear Relationships: Decision trees can capture complex relationships between features and the target variable. They are particularly useful when the data includes non-linear relationships, which could be the case with academic and categorical data.

Handles Categorical Data Well: Since your dataset includes categorical columns (e.g., HSC stream, work experience), decision trees can efficiently handle this type of data without the need for encoding (although one-hot encoding is used in your case).

Small Dataset Friendly: Decision trees are not prone to overfitting with small datasets, especially when hyperparameters like depth and minimum samples per leaf are tuned properly.

Random Forest:

Improved Performance: Random Forest improves on decision trees by averaging multiple trees' predictions, which helps reduce overfitting and increases the robustness of the model, especially with small datasets.

Handles Overfitting: Random Forest can mitigate the risk of overfitting (which small datasets are prone to) by combining multiple trees, each trained on a different subset of the data, which leads to better generalization.

Works with Limited Data: Random Forest performs well even with fewer columns and smaller datasets as it doesn't rely on a single model's performance but aggregates predictions from several models.

Model Training and Hyper Parameter Tuning

Created and hyper tuned 3 model stated above:

1. Logistic Regression:

- **Best Parameters:** The best results were achieved with a regularization strength (C) of 1, using the 'saga' solver and 100 maximum iterations.
- **Performance:** The model achieved a solid cross-validation score of approximately 86.6%. This indicates that the model performed well in distinguishing between the target classes while avoiding overfitting.

Best Parameters: {'C': 1, 'max_iter': 100, 'solver': 'saga'}
Best Cross-Validation Score: 0.8658823529411764

2. Decision Tree Classifier:

- **Best Parameters:** The optimal configuration used the 'entropy' criterion for measuring splits, with no maximum depth, 10 samples required to split nodes, and 1 sample required at leaf nodes.

- **Performance:** This model achieved a cross-validation score of around 80.2%. It performed reasonably well but showed a bit more variance compared to Logistic Regression. The Decision Tree model could potentially be overfitting with the default settings, though tuning helped improve its performance.

Best Parameters: {'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 1, 'min_samples_split': 10}
Best Cross-Validation Score: 0.802016806722689

3. Random Forest Classifier:

- **Best Parameters:** The best-performing Random Forest model used 100 estimators, no maximum depth, and the 'sqrt' option for maximum features, with bootstrap sampling enabled.
- **Performance:** It had the highest cross-validation score of about 87.8%, which demonstrates that Random Forest can handle the dataset well, providing better generalization than the other models.

Best Parameters: {'criterion': 'entropy', 'max_depth': None, 'max_features': None, 'min_samples_leaf': 1, 'min_samples_split': 10}
Best Cross-Validation Score: 0.802016806722689

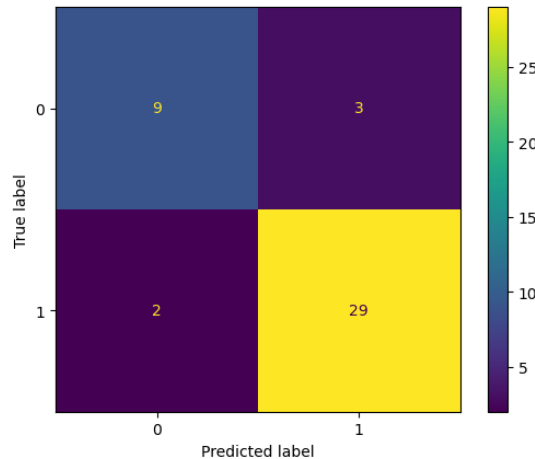
Summary:

- Random Forest outperformed both Logistic Regression and Decision Tree in terms of accuracy, likely due to its ensemble nature and ability to handle complex relationships in the data.
- Logistic Regression also performed well, showing its effectiveness in simpler linear classification problems.
- Decision Tree, while a bit more prone to overfitting in this case, still provided decent results and could be useful in certain contexts.

Model Evaluation:

1. Logistic Regression:

- **Accuracy Score:** 88.37%
- **Precision:** 90.63%
- **Recall:** 93.55%
- **F1 Score:** 92.06%
- **Confusion Matrix:**



- **True Positives (TP) = 29:** The model correctly predicted 29 positive cases (students placed).
- **False Positives (FP) = 3:** The model incorrectly predicted 3 negative cases as positive (students not placed but predicted as placed).
- **False Negatives (FN) = 2:** The model incorrectly predicted 2 positive cases as negative (students placed but predicted as not placed).
- **True Negatives (TN) = 9:** The model correctly predicted 9 negative cases (students not placed).

Interpretation:

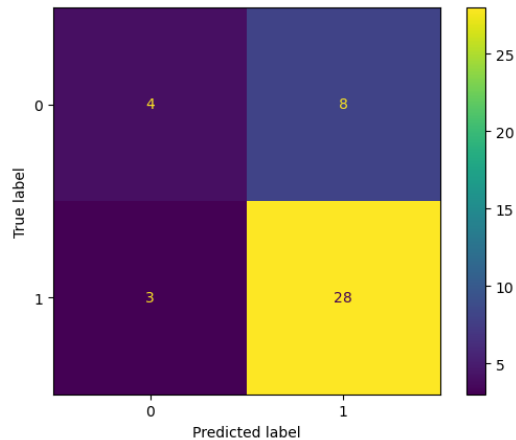
- The Logistic Regression model is **performing quite well** with an accuracy of 88.37%.
- The **precision (0.91)** and **recall (0.94)** are high, indicating the model is good at identifying placed students and rarely misidentifies unplaced students as placed.
- However, the **3 false positives** mean there are still some cases where students who were not placed are mistakenly predicted as placed.
- Similarly, the **2 false negatives** suggest that a few students who should have been predicted as placed were wrongly classified as unplaced.

Overall, Logistic Regression demonstrated the highest overall performance with strong precision and recall. The model correctly identified most of the positive

cases (high recall) while maintaining a good balance between false positives and true positives (high precision and F1 score).

2. Decision Tree:

- **Accuracy Score:** 74.42%
- **Precision:** 77.78%
- **Recall:** 90.32%
- **F1 Score:** 83.58%
- **Confusion Matrix:**



- **True Positives (TP) = 28:** The model correctly predicted 28 positive cases (students placed).
- **False Positives (FP) = 8:** The model incorrectly predicted 8 negative cases as positive (students not placed but predicted as placed).
- **False Negatives (FN) = 3:** The model incorrectly predicted 3 positive cases as negative (students placed but predicted as not placed).
- **True Negatives (TN) = 4:** The model correctly predicted 4 negative cases (students not placed).

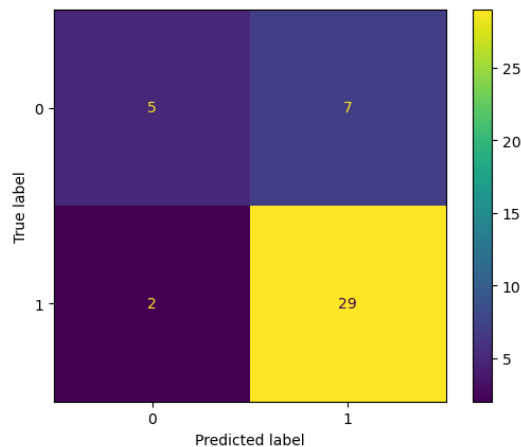
Interpretation:

- The **accuracy (74.42%)** is lower than Logistic Regression, indicating the Decision Tree model is not as accurate.
- The **precision (0.78)** and **recall (0.90)** are both decent, but the **high number of false positives (8)** indicates that the model is often mistakenly predicting unplaced students as placed.
- The **3 false negatives** mean that the Decision Tree is **missing** some students who should have been predicted as placed.
- This model might be overfitting since it has a complex structure, leading to **misclassifications** and not generalizing well.

While the Decision Tree had a lower accuracy score, it excelled in recall, identifying most of the positive cases. However, its precision was somewhat lower, suggesting it was more prone to false positives. The model's performance could be improved with further tuning.

3. Random Forest Classifier:

- **Accuracy Score:** 79.07%
- **Precision:** 80.56%
- **Recall:** 93.55%
- **F1 Score:** 86.57%
- **Confusion Matrix:**



- **True Positives (TP) = 29:** The model correctly predicted 29 positive cases (students placed).
- **False Positives (FP) = 7:** The model incorrectly predicted 7 negative cases as positive (students not placed but predicted as placed).
- **False Negatives (FN) = 2:** The model incorrectly predicted 2 positive cases as negative (students placed but predicted as not placed).
- **True Negatives (TN) = 5:** The model correctly predicted 5 negative cases (students not placed).

Interpretation:

- The **accuracy (79.07%)** is higher than the Decision Tree but lower than Logistic Regression.
- The **precision (0.81)** is also good, indicating that most of the predicted placed students are actually placed.
- The **recall (0.94)** is very high, meaning the model is excellent at identifying placed students with fewer false negatives (only 2).
- However, the **false positives (7)** indicate that some unplaced students are still being classified as placed.

The Random Forest model showed a good balance between accuracy, precision, and recall. While it didn't outperform Logistic Regression in terms of accuracy, it still provided strong results, with particularly high recall, ensuring it was effective at detecting positive cases.

Conclusion:

- Logistic Regression was the top performer in terms of overall accuracy and precision, making it the most reliable model for this dataset.
- Random Forest provided a good trade-off between accuracy and recall, being a strong candidate for situations where we care more about identifying positive cases accurately.
- Decision Tree was the least effective in terms of accuracy but performed well in recall. It may require further tuning or pruning to reduce overfitting and improve precision.

In summary, Logistic Regression stands out as the best model for this dataset, while Random Forest offers a solid alternative with its strong recall.

Model Used for App deployment:

Voting Classifier: combines the predictions from multiple models (Logistic Regression, Decision Tree, and Random Forest) to improve overall performance. With a cross-validated accuracy of 84.89%, the Voting Classifier performs well by reducing bias and variance compared to individual models. This ensemble approach enhances prediction reliability and generalizes better on unseen data, making it a powerful tool for improving model performance.

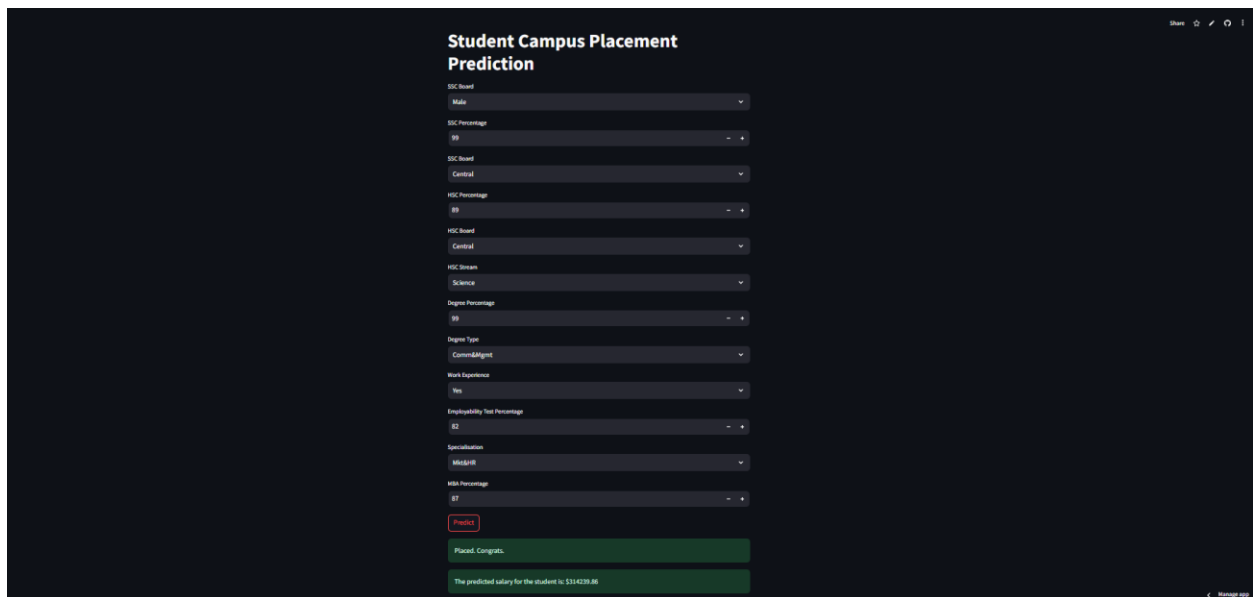
Fun Activity:

Created and tuned a random forest Regressor to predict the **salary** of students who got placed.

Model Deployment:

Created an app and deployed the model using 'streamlit'.

App url: <https://campus-placement-prediction.streamlit.app/>



The screenshot shows a web application titled "Student Campus Placement Prediction". It features a series of input fields for user data, each with a dropdown menu. The inputs are: SSC Board (Male), SSC Percentage (89), SSC Board (Central), HSC Percentage (89), HSC Board (Central), HSC Stream (Science), Degree Percentage (89), Degree Type (Comm&Agri), Work Experience (Yes), Employability Test Percentage (82), Specialization (MCA+IT), and MBA Percentage (87). Below these inputs is a red "Predict" button. After clicking, a green message box appears saying "Placed. Congrats." and another green box displays "The predicted salary for the student is: \$314339.86". The app has a dark theme and a sidebar on the right with icons for home, search, and other functions.

Github Repository: <https://github.com/GovindDogra/Campus-Placement-Prediction/tree/main>