Identifying Patterns and Trends in Campus Placement Data using Machine Learning

Project ID: SPS_PRO_4020

Abstract:

In order to determine whether a student will be placed in a job, this project analyzes and models a dataset for placement. The methodology uses a number of processes, such as preprocessing the data, oversampling, choosing the best features, and applying a Decision Tree Classifier to make predictions. Utilizing criteria like accuracy, precision, recall, F1 score, and ROC AUC score, the model's performance is assessed. The outcomes demonstrate how machine learning techniques can improve decision-making processes in educational and career planning, resulting in the placement of knowledgeable students and the most effective use of resources. The conclusions drawn from this code serve as an example of its applicability in situations where student employability is being evaluated in the real world.

Keywords: Accuracy, DecisionTreeClasiifier, F1 score, Precision, ROC AUC score.

1 Introduction:

1.1 Overview

Predicting the results of job placement is very important in the context of education and career aspirations. This program use machine learning to forecast students' placement status. The code makes use of the Decision Tree Classifier, a potent technique that builds a tree-like model based on input features to produce predictions. For educational institutions, students, and employers to make well-informed decisions, this model's prediction accuracy is essential.

The system analyzes features at each turn to choose the best course, which results in placement predictions at the end. The model can find complex correlations between features thanks to the nodes' hierarchical architecture, which opens the door to more precise predictions. Beyond its predictive skills, the algorithm's intuitive transparency enables teachers and students to understand the reasoning behind each prediction, promoting trust and allowing for ongoing strategy improvement.

The model training procedure and performance evaluation utilizing multiple metrics, including accuracy, precision, recall, F1 score, and ROC AUC score, are demonstrated in the code's following parts. These metrics give a thorough picture of how well the algorithm can categorize placement results. The trained model is also used to provide a sample prediction, giving users a taste of how it may be used in real-world scenarios.

In summary, this code incorporates a comprehensive strategy for utilizing machine learning to forecast student job placement outcomes. Its importance stems from its potential to assist educational institutions, learners, and employers in making knowledgeable decisions by predicting placement opportunities according to students' academic and personal characteristics. The environment of education and job placement can be optimized by embracing technology-driven solutions, bringing student goals into line with market demands.

1.2 Purpose

In order to anticipate student employment placements based on a variety of academic and personal characteristics, the project uses the power of machine learning, specifically a Decision Tree Classifier. With the use of this predictive capabilities, educational institutions may improve their practises, connect their curricula with market demands, and provide students with specialised support. The project also aims to offer tailored assistance to students, assisting them in selecting their career routes with knowledge. Additionally, it highlights the value of data-driven decision-making in the context of planning one's education and career, demonstrating how predictive analytics may successfully link goals for higher education with concrete professional accomplishments.

The ultimate goal of this project is to optimise the educational and professional landscape through the seamless integration of technology-driven solutions, making sure that student objectives are in line with those of the job market.

2 Literature Survey

2.1 Existing Approaches or Methods

Conventional methods for resolving this issue frequently entail manual evaluation and arbitrary judgement. Counsellors and advisers are frequently used by educational institutions to advise students based on their own experiences and knowledge. These methods might be inconsistent and subjective, though. Some institutions might base their decisions on straightforward rule-based systems or elementary statistical analysis, but these approaches might not adequately account for the nuanced nature and complexity of the job placement process.

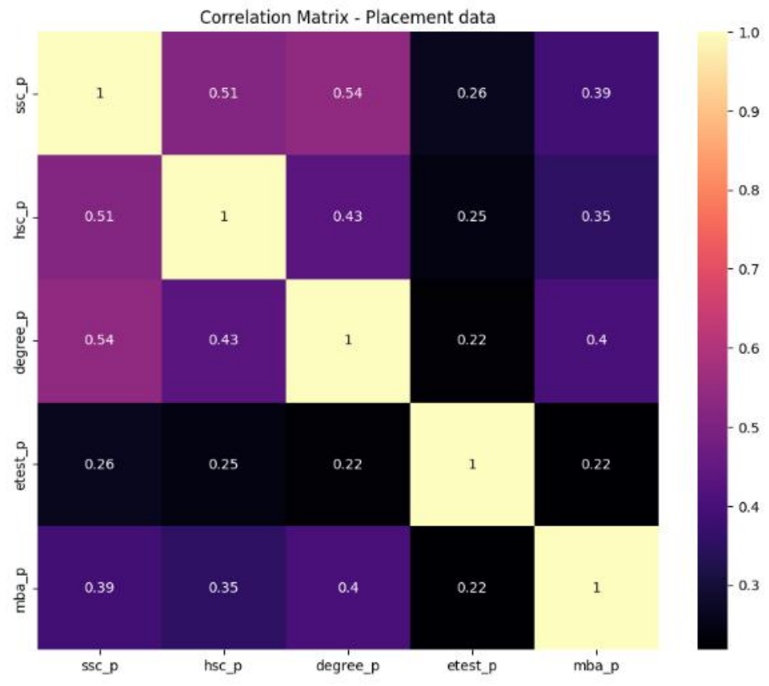
2.2 Proposed Solution:

In order to anticipate student employment placements, this project's proposed solution makes use of machine learning, more especially the Decision Tree Classifier. The algorithm seeks to deliver precise and impartial predictions by utilising a huge dataset with a variety of academic and personal factors. This approach has a number of benefits, including the capacity to spot intricate relationships between attributes and decision-making transparency. By providing a methodical, data-driven, and transparent strategy to help educational institutions, students, and companies make wise judgements on job placements, it tackles the issue that currently exists. In order to guarantee the efficacy of the predictive model, the suggested solution also incorporates model evaluation utilising measures like accuracy, precision, recall, F1 score, and ROC AUC score.

Methdology:

1. Data Reading and Preprocessing:

   Using the Pandas package, the dataset is loaded initially. By presenting a glimpse of the data, summarizing the statistics, and highlighting any missing values, exploratory data analysis is carried out. In order to better comprehend the correlations between attributes, the correlation matrix is shown using a heatmap. Utilizing label encoding, categorical variables are encoded, and superfluous columns ('sl_no' and'salary') are removed.



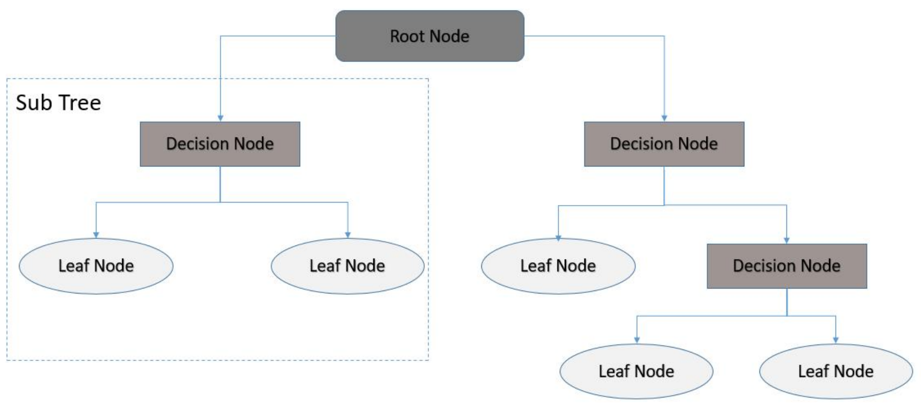
2. Handling Missing Values:

For a model to be effective, the imbalance between classes must be addressed. The 'Placed' and 'Not Placed' classes are balanced using the Synthetic Minority Over-sampling Technique (SMOTE), which improves the generalizability of the model.

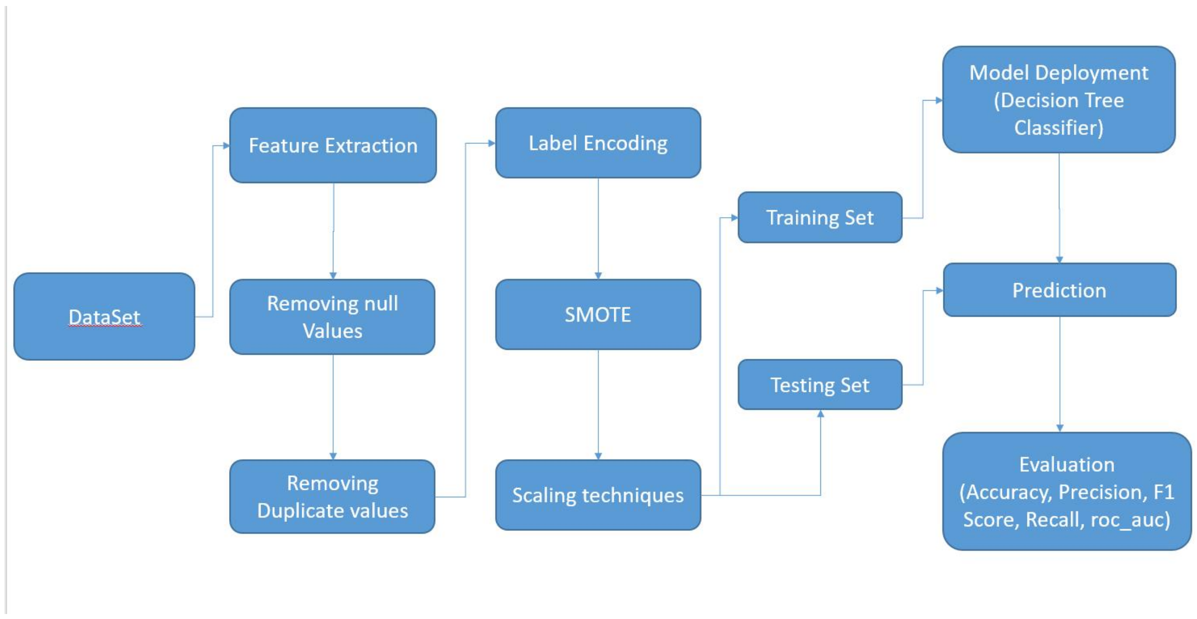
3. Label Encoding:
   In order to make categorical attributes (such as "gender," "ssc_b," "hsc_b," "hsc_s," "degree_t," "workex," "specialization," and "status") acceptable for the machine learning algorithm, label encoding is used. By preparing the data, it is ensured that it is formatted correctly for model training.
4. Feature Extraction:
   The dataset is split into input characteristics (X) and the target variable (y) following preprocessing. Each feature's importance is evaluated, and any potential skewness is considered.
5. Model Deployment and evolution:
   The train_test_split function divides the dataset into training and testing sets. The training data are used to create and train a decision tree classifier. The testing set's placement statuses are predicted using the trained model. To assess the model's predictive skills, a number of performance metrics, including accuracy, precision, recall, F1 score, and ROC AUC score, are calculated and printed.
6. Finding Metrics:
   The code calculates accuracy, precision, recall, F1 score, and ROC AUC score for the predictions made by the Decision Tree Classifier using the sklearn metrics module.

3 Theoretical Analysis

3.1 Block Diagram



Decision Tree


DataSet
Feature Extraction
Removing null Values
Removing Duplicate values
Label Encoding
SMOTE
Scaling techniques
Training Set
Testing Set
Model Deployment (Decision Tree Classifier)
Prediction
Evaluation (Accuracy, Precision, F1 Score, Recall, roc_auc)

Block Diagram

3.2 Hardware and Software Designing

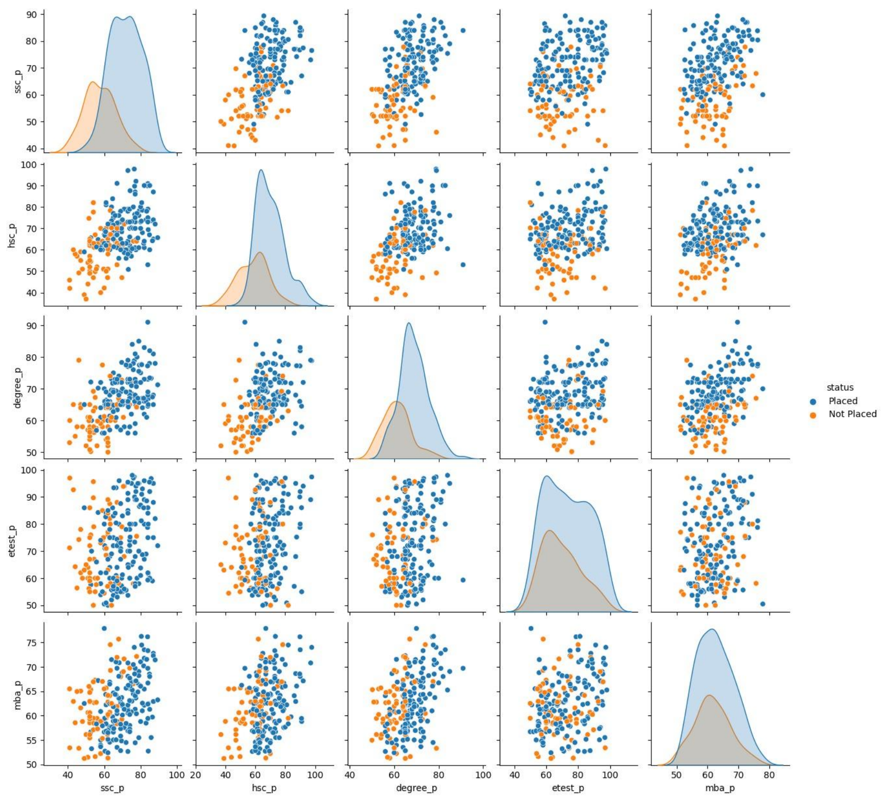
Hardware:

- Processor: Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz   2.11 GHz
- RAM: 8.00 GB

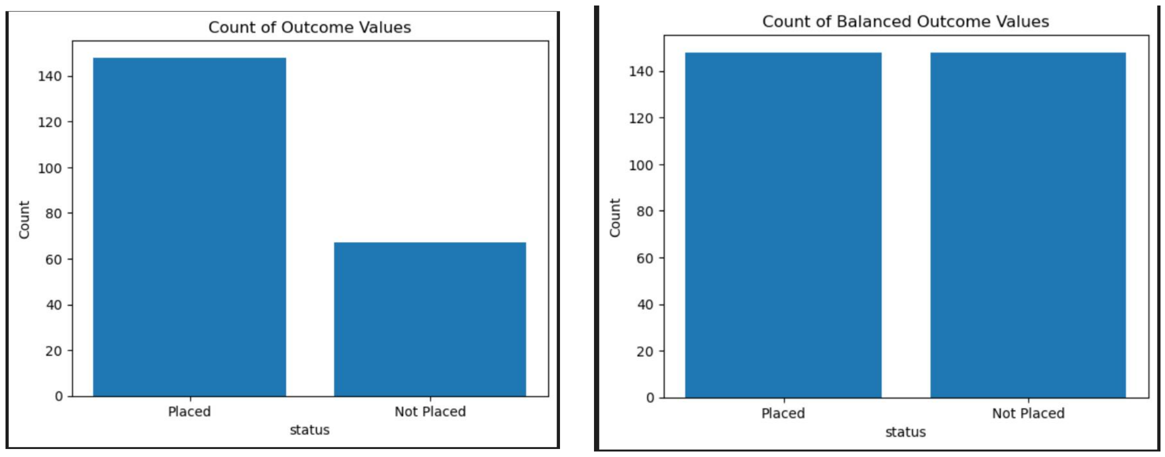
Software Requirements:

- Jupyter Notebook
- Google Colab
- IBM Watson Studio
- VS Code
-

4 Experimental Investigations



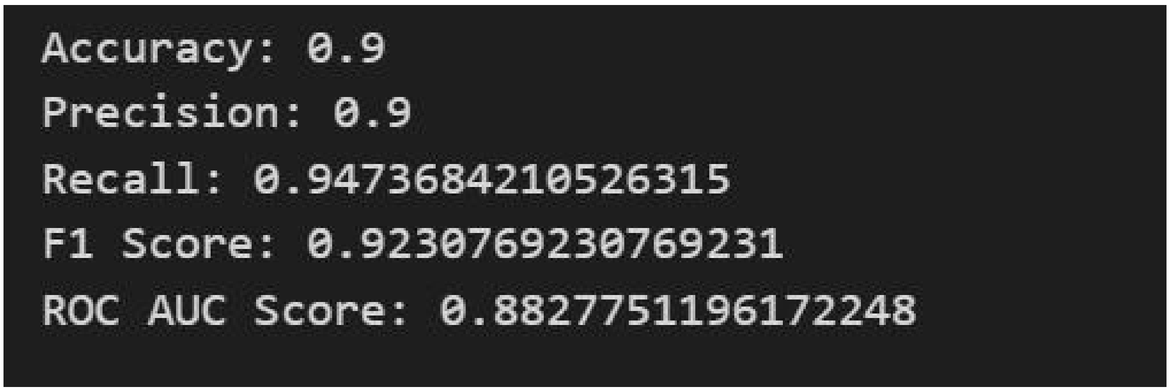
Pair Plot



Output Count Before and After applying SMOTE Analysis

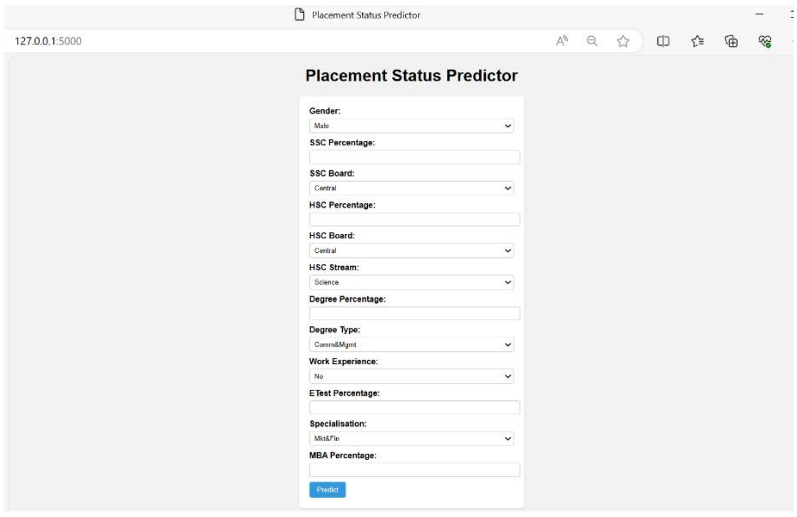
5 Result:

Evaluation metrics:



```
Accuracy: 0.9
Precision: 0.9
Recall: 0.9473684210526315
F1 Score: 0.9230769230769231
ROC AUC Score: 0.8827751196172248
```

Output:

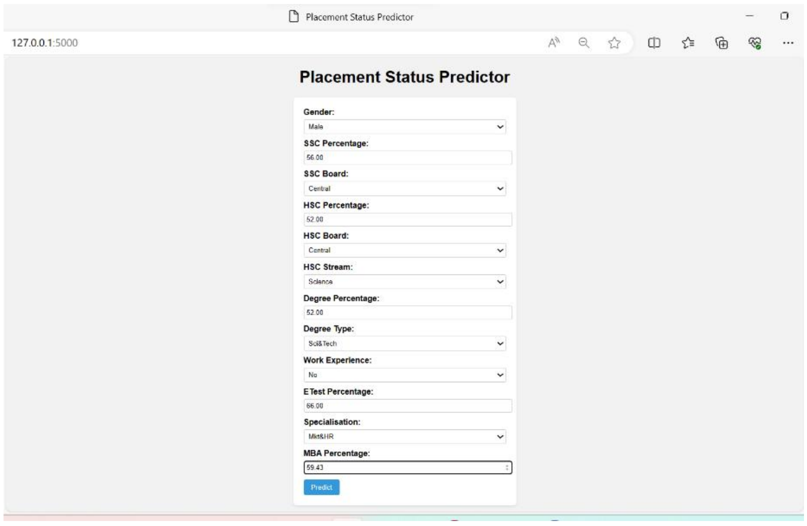


Placement Status Predictor

127.0.0.1:5000

Placement Status Predictor

Gender: Male
SSC Percentage:
SSC Board: Central
HSC Percentage:
HSC Board: Central
HSC Stream: Science
Degree Percentage:
Degree Type: Comm&Mgmt
Work Experience: No
E Test Percentage:
Specialisation: Mkt&Fin
MBA Percentage:

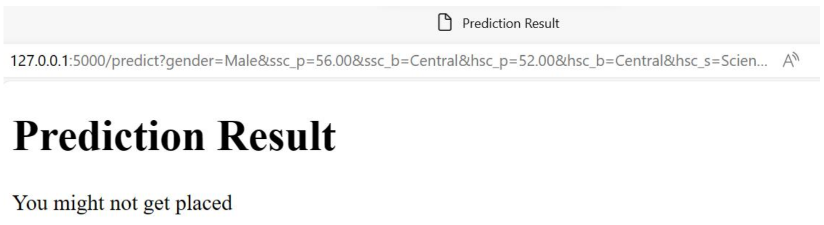


Placement Status Predictor

127.0.0.1:5000

Placement Status Predictor

Gender: Male
SSC Percentage: 56.00
SSC Board: Central
HSC Percentage: 52.00
HSC Board: Central
HSC Stream: Science
Degree Percentage: 52.00
Degree Type: Sci&Tech
Work Experience: No
E Test Percentage: 66.00
Specialisation: Mkt&HR
MBA Percentage: 59.43



Prediction Result

127.0.0.1:5000/predict?gender=Male&ssc_p=56.00&ssc_b=Central&hsc_p=52.00&hsc_b=Central&hsc_s=Scien...

Prediction Result

You might not get placed

6 Advantages and Disadvantages

Advantages:

Transparency: Decision trees are inherently transparent models, making it easy to understand the reasoning behind predictions. This transparency can build trust among users, including students, educators, and employers.

Interpretability: Decision trees provide interpretable rules, allowing stakeholders to grasp how specific attributes influence placement predictions. This can be valuable for educational institutions and students seeking actionable insights.

Balancing Class Imbalance: Model generalizability is achieved through SMOTE preprocessing technique and improved accuracy in predicting both "Placed" and "Not Placed" outcomes.

Feature Importance: Unnecessary feature columns like sl_no and salary are removed which helps in achieving more better results.

No Assumption of Linearity: Unlike some linear models, decision trees do not assume a linear relationship between features and outcomes. They can capture complex, nonlinear patterns in the data.

Disadvantages:

Overfitting: Decision trees are prone to overfitting, especially when the tree depth is not controlled. This can lead to poor generalization on unseen data. Careful pruning or using ensemble methods may be necessary to mitigate this issue.

Extra Features: More features columns can be added to the dataset which will help in analyzing the problem more.

7 Applications

Student Placement Services: This solution can be used by educational institutions, in particular colleges and universities, to improve their student placement services. They can give pupils more specialised career advise by forecasting employment placements.

Centres for career counselling: Predictive models can be used by career counselling centres to provide students with personalised career assistance that matches their interests and skills with appropriate career options.

Recruiters: Employers and staffing firms can use these algorithms to find and target applicants whose profiles match job specifications.

Enhanced Job Matching: By employing predictive models to more precisely match job seekers with appropriate job advertisements, job portals can enhance their recommendation systems.

HR: HR departments in organizations can use similar models to predict the suitability of job applicants for specific roles, streamlining the recruitment process.

Skills Gap Analysis: Programs aimed at skills development can employ predictive analytics to identify skill gaps among participants and tailor training programs accordingly.

Course Recommendations: EdTech platforms can suggest courses to users based on their career goals and predicted job placements.

Workforce Planning: Government agencies responsible for workforce development can use predictive models to plan educational and training initiatives that align with labor market demands.

Support for Disadvantaged Populations: Nonprofit organizations working with underprivileged or disadvantaged populations can use these models to provide targeted support and resources to individuals seeking employment opportunities.

Educational Research: Researchers can use the insights gained from predictive models to conduct studies on the effectiveness of different educational programs and policies in promoting job placements.

Matching Interns with Companies: Educational institutions and companies offering internships can use predictive models to match students with suitable internship opportunities.

Career Advancement: Services aimed at helping professionals advance in their careers can utilize predictive models to provide guidance on career progression.

Mentorship Programs: Matching Mentors and Mentees: Mentorship programs can use predictive models to match mentors with mentees based on career goals and compatibility.

Entrepreneurial Opportunities: Organizations supporting startups and entrepreneurship can use predictive models to identify individuals with skills and aspirations conducive to entrepreneurial ventures.

8 Conclusion:

In conclusion, this project makes use of the Decision Tree Classifier's prediction ability to provide insightful information about the consequences of student job placement. It helps universities to improve their methods, match curriculum with industry expectations, and offer individualized support by incorporating machine learning into the field of education and careers in a smooth manner. In return, students receive a data-driven compass for their career options, enabling them to make wise decisions. This work makes an important contribution to the future of educational and career planning by demonstrating the ability of predictive analytics to link ambitions for higher education with concrete professional achievements through thorough preprocessing, model deployment, and evaluation.

9 Future Scope

In the future, several enhancements can be made to further refine the project's predictive capabilities for student job placements using a Decision Tree Classifier. One promising avenue is the incorporation of ensemble methods like Random Forests or Gradient Boosting, which can collectively improve model performance and robustness. Additionally, conducting a more exhaustive hyperparameter tuning process can optimize the Decision Tree Classifier's parameters, fine-tuning its predictive accuracy. Exploring advanced feature engineering techniques to create new informative attributes and employing feature selection methods can help identify the most relevant factors impacting placements.

10 Bibilography

1. Naresh Patel K M, Goutham N M, Inzamam K A, Suraksha V Kandi, Vineet Sharan V R, 2022, Placement Prediction and Analysis using Machine Learning, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICEI – 2022 (Volume 10 – Issue 11),

2. https://www.academia.edu/89185650/Campus_Placements_Prediction_and_Analysis_using_Machine_Learning

3. https://www.irjmets.com/uploadedfiles/paper/issue_7_july_2022/28670/final/fin_irjmets1658646876.pdf

4. https://ijaem.net/issue_dcp/Future%20Technology%20Trends%20Prediction%20for%20the%20Job%20Market.pdf

5. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

6. https://scikit-learn.org/stable/modules/tree.html

11 Appendix

A. Source Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data
```

```
data.head()
```

```
data.tail()
```

```
data.describe()
```

```
data.info()
```

```
#Correlation Matrix
plt.figure(figsize=(10, 8))
corr_matrix = data.corr()
sns.heatmap(corr_matrix, annot=True, cmap='magma')
plt.title('Correlation Matrix - Placement data')
plt.show()
```

```
data = data.drop(columns=['sl_no'])
data = data.drop(columns=['salary'])
```

```
#Correlation Matrix
plt.figure(figsize=(10, 8))
corr_matrix = data.corr()
sns.heatmap(corr_matrix, annot=True, cmap='magma')
plt.title('Correlation Matrix - Placement data')
plt.show()
```

```
data.isnull().sum()
```

```
pip install missingno
```

```
import missingno as msno
p = msno.bar(data)
```

```
data[data.duplicated()==True]
```

```
data["status"].unique()
```

```
data.status.value_counts()
```

```
outcome_counts = data['status'].value_counts()
plt.bar(outcome_counts.index, outcome_counts.values)
plt.title('Count of Outcome Values')
plt.xlabel('status')
plt.ylabel('Count')
plt.show()
```

```
from sklearn import preprocessing
```

```
le = preprocessing.LabelEncoder()
data['gender'] = le.fit_transform(data['gender'])
data['ssc_b'] = le.fit_transform(data['ssc_b'])
data['hsc_b'] = le.fit_transform(data['hsc_b'])
data['hsc_s'] = le.fit_transform(data['hsc_b'])
data['degree_t'] = le.fit_transform(data['degree_t'])
data['workex'] = le.fit_transform(data['workex'])
data['specialisation'] = le.fit_transform(data['specialisation'])
data['status'] = le.fit_transform(data['status'])
```

```
data.head()
```

```
x = data.drop(columns=['status'])
x
```

```
y = data['status']
y
```

```
pip install imbalanced-learn
```

```
from imblearn.over_sampling import SMOTE
smote=SMOTE(sampling_strategy='minority')
x,y=smote.fit_resample(x,y)
y_df = pd.Series(y)
x_df = pd.DataFrame(x)
print(y_df.value_counts())
print(x_df.value_counts())
```

```
import matplotlib.pyplot as plt

balanced_data = pd.DataFrame({'status': y})
balanced_counts = balanced_data['status'].value_counts()
plt.bar(balanced_counts.index, balanced_counts.values)
plt.title('Count of Balanced Outcome Values')
plt.xlabel('status')
plt.ylabel('Count')
plt.xticks(balanced_counts.index, ['Not Placed', 'Placed'])  # Set the x-axis labels
plt.show()
```

```
skew = data.skew()
print('Skew of attribute distributions in the data:\n')
print(skew)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=30)
```

```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier(random_state=42)
dtc = dtc.fit(X_train,y_train)
y_pred = dtc.predict(X_test)
y_pred
```

```
from sklearn import metrics
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)

print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
print("ROC AUC Score:", roc_auc)
```

```
pred = dtc.predict([[1  ,80.000000, 1,  70.000000,  1,  1,  72.000000,  2,  0,  87.000000,  1,  71.040000]])
pred
```