STUDENT PERFORMANCE PREDICTION MODEL

By: Edith Luna, Ashley Chiang, Naiya Patel, and Alicia Kim

TABLE OF CONTENTS

01

DATASET OVERVIEW

Overview

Description

- "Student Performance" dataset from the UCI Machine Learning Repository
- Two datasets one for **math class** and Portuguese class

Dataset

- **Demographics:** sex, age, Pstatus
- **Academic factors:** studytime, failures, G1 (first period grade), absences
- **Support & behavior:** schoolsup, famsup, paid, internet, activities, goout, health

Inspection

- 395 rows and 33 columns
- 16 integers and 17 object data types
- No missing values
- No duplicates

02

MODEL OBJECTIVES

Business Question and Story

Can we predict whether a student will pass or fail their math course based on their third-period grade (G3) and other personal and academic factors?

- Can help both students and educators by identifying features that might have an impact on the final grade.
- Shows which students are struggling due to support, extra responsibilities or lack of encouragement.
- Schools can provide support such as tutoring and mentorship from the obtained data.

03

DATA CLEANING & PREPARATION

Data Cleaning

1. Remove Duplicates

```
[ ] df.duplicated().sum() #prints the total number of duplicates in the dataframe
```

```
np.int64(0)
```

2. Handle Missing Values

```
df.isnull().sum() #prints the sum of missing values in each column
```

	0
school	0
sex	0
age	0
address	0
famsize	0
Pstatus	0

3. Handle Outliers

```
[ ] #function which removes all outliers in the defined columns
def remove_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    print(len(df[(df[column] < lower_bound) | (df[column] > upper_bound)])) # total number of outliers removed in each column
    return df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]

numerical_columns = ['age', 'traveltime', 'studytime', 'failures', 'famrel', 'freetime', 'goout', 'health', 'G1','absences']

for column in numerical_columns:
    df = remove_outliers_iqr(df, column)
```

```
1
8
26
79
16
10
0
0
0
15
```

Creating a binary classification

Initially G3 (outcome variable) is a score that ranges from 1-20.
- G1 & G2 used as features

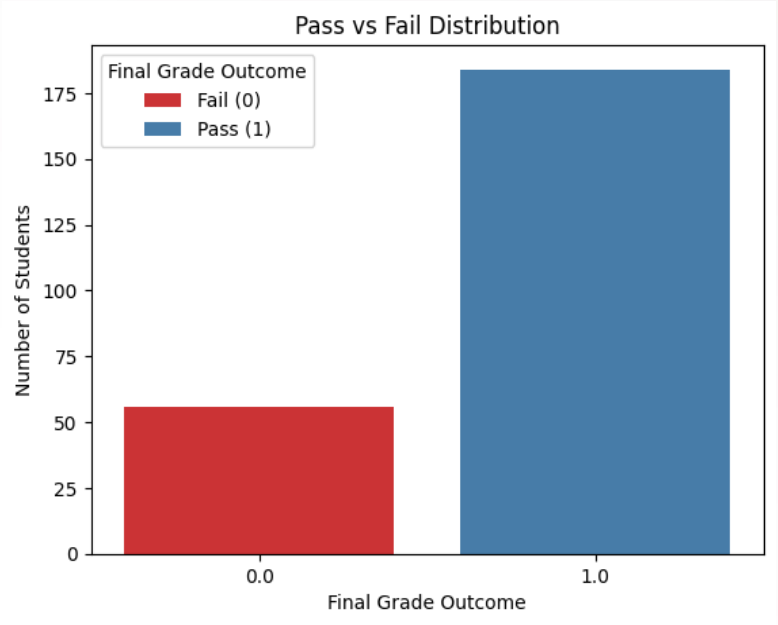
Create a 'Final_Grade' column with a threshold to indicate whether student will pass or fail.

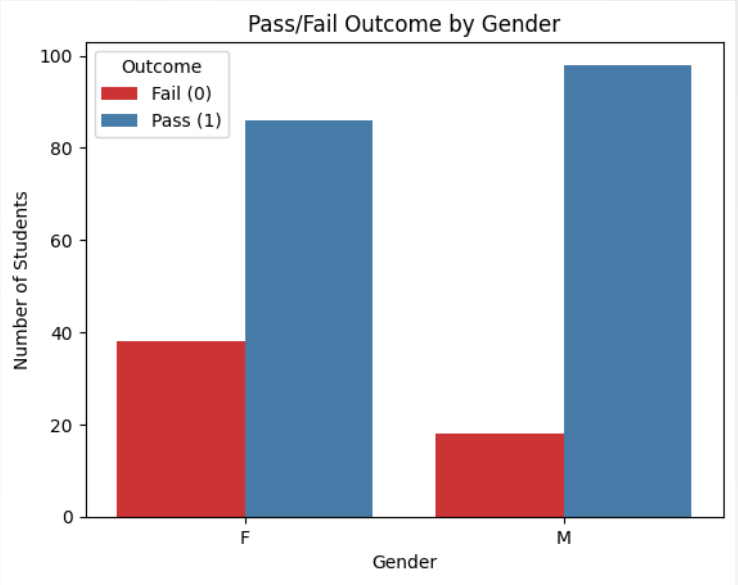
```
[ ]  # Converts the values for G3 to binary (1 = Pass, 0 = Fail) in column Final_Grade

     df.loc[df['G3'] >= 10, 'Final_Grade'] = 1
     df.loc[df['G3'] < 10, 'Final_Grade'] = 0
```

Criteria: A score of 10 or more means student will PASS= 1
A score of under 10 means student will FAIL =0

04 EDA INTERPRETATION

EDA- Understanding Dataset: Demographics





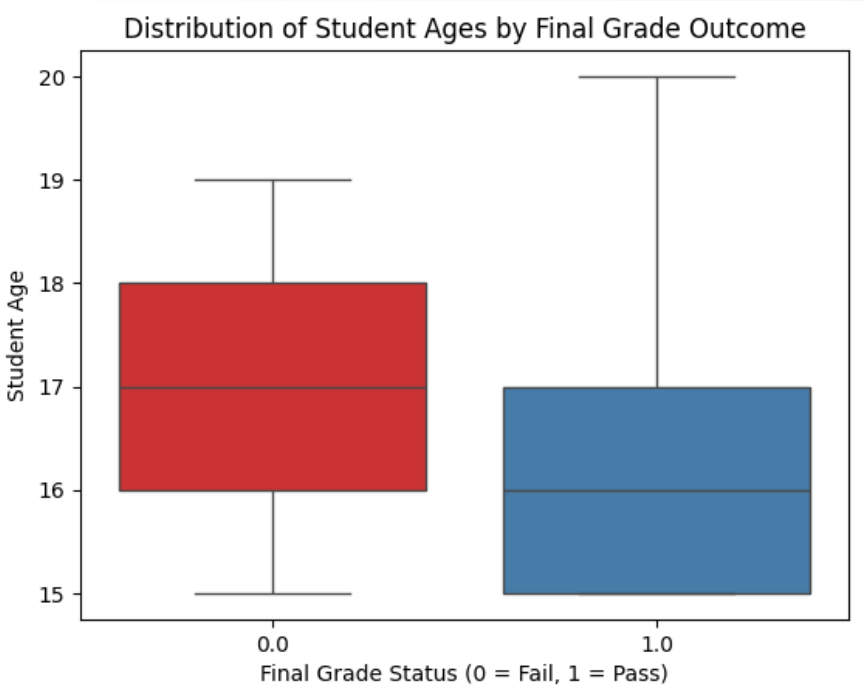
- Approximately how many students passed the course?

  The majority of students passed the course.
  Around **180 students passed** while **about 55 failed**.

Both males and females had higher pass rates, but males passed at a slightly higher rate.

EDA- Boxplot

- ❑ Double checkpoint for outliers
- ❑ Explore age trends in pass/fail outcomes

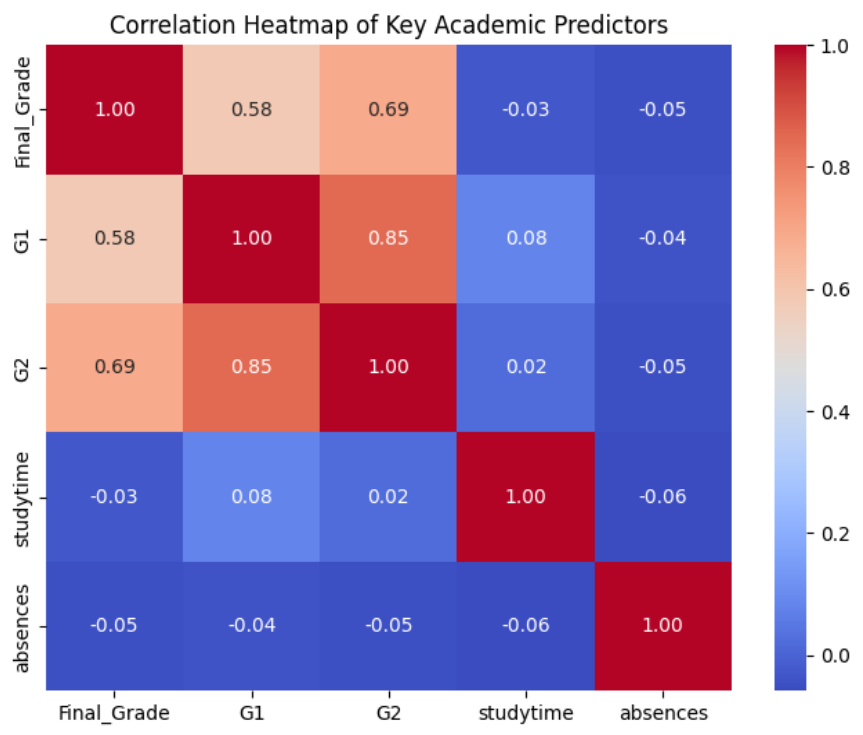


What does the boxplot suggest about age and academic success?

- Students who failed (red) tend to be slightly older on average
- Students who passed (blue) are slightly younger, with more variation

- ❑ What is the average age of those who 'fail'?

Key Academic Predictors



- Correlation matrix on numeric variables to identify key relationships

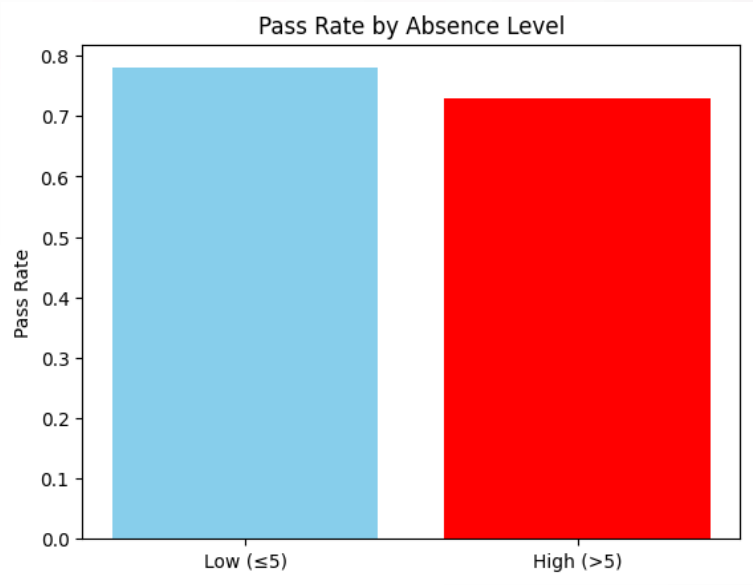
Subset of features relevant to **classroom behavior and academic** performance

Strong Academic Indicators:
G1 & G2: First and second period grades
→ Show strong correlation with the final grade (0.58 and 0.69)
→ Reflect ongoing academic performance

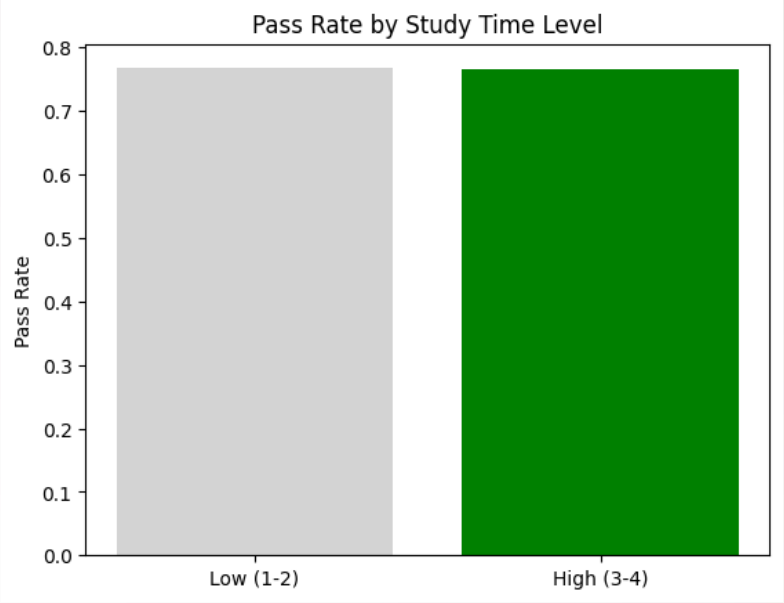
Behavioral Factors:
Studytime (weekly study hours)
→ Shows very weak correlation (-0.03), but still valuable to explore
Absences → Weak negative correlation (-0.05), but might matter in non-linear or threshold-based ways

Key Academic Predictors

Even though absences and study time showed weak correlation with final grades, I used thresholds to explore whether **students with low vs. high values** in these areas had different pass rates. This helped reveal patterns that a correlation matrix alone might miss.



- Students with low absences (≤5) have a higher pass rate (~78%)
- Students with high absences (>5) show a drop in pass rate (~73%)



- A very slight difference in pass rate, study time may not be a strong standalone predictor of final grade outcomes

05

TRAIN & EVALUATE MODELS

Dummy Coding and Partitioning

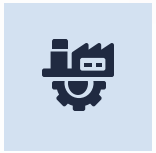
- Created new_df with features closely correlated to G3 / Final Grade
    - df_new = df[['sex', 'age', 'Pstatus', 'traveltime', 'studytime', 'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'higher', 'internet', 'famrel', 'freetime', 'goout', 'health', 'absences','Final_Grade']]
- Dummy coded using the new_df
- X = df_dummy.drop('Final_Grade', axis = 1)
- y = df_dummy['Final_Grade']
- Split data for training and testing
    - 20% of dataset for testing, 80% for training
    - Random state = 1; to ensure same split every time

MODELS USED

LOGISTIC REGRESSION

DECISION TREE

DECISION TREE GRID SEARCH



RANDOM FOREST

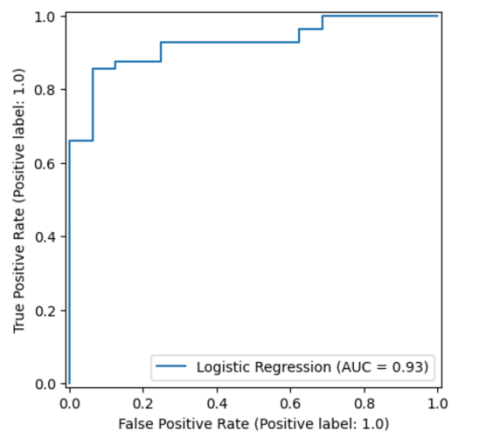
RANDOM FOREST GRID SEARCH

Logistic Regression

Confusion Matrix

```
Confusion Matrix (Accuracy 0.8929)

       Prediction
Actual   0   1
     0  30  10
     1   8 120
Confusion Matrix (Accuracy 0.8333)

       Prediction
Actual  0  1
     0  8  8
     1  4 52
```

```
 [[ 8  8]
 [ 4 52]]
Accuracy: 0.8333333333333334
Precision:  0.8666666666666667
Recall (Sensitivity):  0.9285714285714286
Specificity:  0.5
FNR:  0.0714285714285714
FPR:  0.5
F1 score:  0.896551724137931
AUC: 0.9252232142857142
```

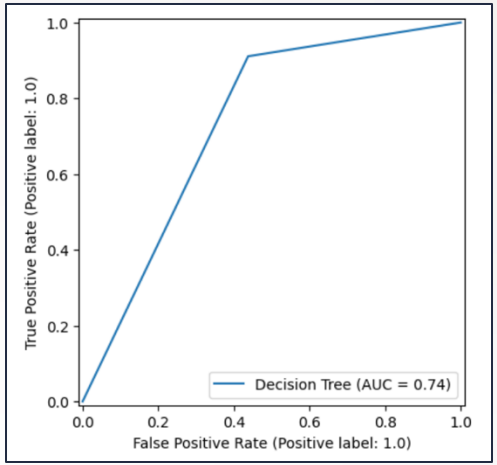


Interpretation

- Accuracy of 83% on test set indicates the model is good at predicting (can be better)
- High recall (0.92) – good at catching positives
- F1 score(0.89) – balance of both false positives and false negatives
- AUC(0.93) – good at identifying between classes
- Low specificity(0.50) – not good at identifying true negatives
- No overfitting as it performs similar on both training and test sets

Decision Tree

```
Confusion Matrix (Accuracy 1.0000)

       Prediction
Actual   0   1
     0  40   0
     1   0 128
Confusion Matrix (Accuracy 0.8333)

       Prediction
Actual  0  1
     0  9  7
     1  5 51
```




```
confusion matrix:
 [[ 9  7]
 [ 5 51]]
Accuracy: 0.8333333333333334
Precision:  0.8793103448275862
Recall (Sensitivity):  0.9107142857142857
Specificity:  0.5625
FNR:  0.0892857142857143
FPR:  0.4375
F1 score:  0.8947368421052632
AUC: 0.7366071428571428
```

INTERPRETATION

Sensitivity (91%)

Model is effective at correctly identifying true positives (few FNs)

F1 Score (89%)

Provides good balance between sensitivity and precision

False Positive Rate (43.75%)

High FPR → model mistakenly labels many actual negative cases as positive

Specificity (56.25%)

Model incorrectly labels many negatives as positive (many FPs)

AUC (0.74)

Model performance is moderate as it can somewhat distinguish between predicting pass/fail

Overfitting
Significantly higher accuracy on train vs test data

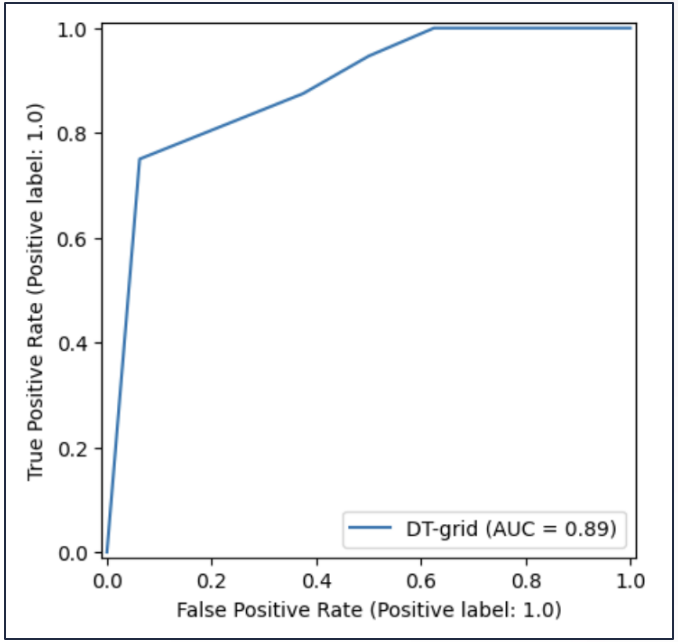
Decision Tree – GridSearchCV

```
param_grid_DT = {
    'max_depth': [5, 10, 12, 15, 20],
    'min_impurity_decrease': [0, 0.0001, 0.001, 0.005, 0.01],
    'min_samples_split': [10, 20, 30, 40, 50],
    'random_state': [1],
}
```

```
DT_grid.best_params_

{'max_depth': 5,
 'min_impurity_decrease': 0.01,
 'min_samples_split': 20,
 'random_state': 1}
```

```
Confusion Matrix (Accuracy 0.8750)

       Prediction
Actual   0   1
     0  32   8
     1  13 115
Confusion Matrix (Accuracy 0.8194)

       Prediction
Actual  0  1
     0 10  6
     1  7 49
```

```
confusion matrix:
 [[10  6]
 [ 7 49]]
Accuracy: 0.8194444444444444
Precision:  0.8909090909090909
Recall (Sensitivity):  0.875
Specificity:  0.625
FNR:  0.125
FPR:  0.375
F1 score:  0.8828828828828829
AUC: 0.8878348214285714
```



INTERPRETATION

Sensitivity (87.5%)

Model is effective at correctly identifying true positives (few FNs)

Precision (89.09)

Of all the positive predictions, most of them are correct → positive predictions are more likely to be correct

False Positive Rate (37.5%)

High FPR → model mistakenly labels many actual negative cases as positive

Specificity (62.5%)

Model incorrectly labels many negatives as positive (many FPs)

AUC (0.89)

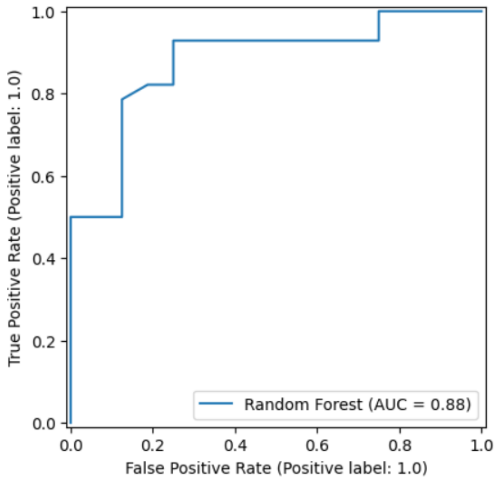
Model performance is good as it can somewhat distinguish between predicting pass/fail

Potential Overfitting
Higher accuracy on train vs test data

Random Forest

```
Confusion Matrix (Accuracy 1.0000)

       Prediction
Actual   0   1
     0  40   0
     1   0 128
Confusion Matrix (Accuracy 0.7778)

       Prediction
Actual  0  1
     0  4 12
     1  4 52
```

```
confusion matrix:
 [[ 4 12]
 [ 4 52]]
Accuracy: 0.7777777777777778
Precision:  0.8125
Recall (Sensitivity):  0.9285714285714286
Specificity:  0.25
FNR:  0.0714285714285714
FPR:  0.75
F1 score:  0.8666666666666667
AUC: 0.8783482142857144
```



Model Interpretations:

- Accuracy of 100% on test set and 77% on train set → possible overfitting.
    - Accuracy of 100% → no false positives or false negatives.

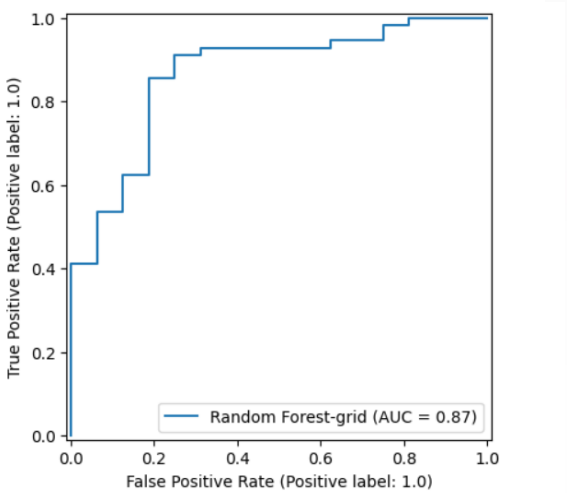
Test set interpretations:

- Accuracy 77% → decent overall performance, but not great.
- Precision 81% → not bad, 81% of predicted positives were correct.
- Recall 92% → model detects most of the actual positives well.
- Specificity 25% → only 25% of actual class 0s were correctly classified, very poor.
- FNR 0.071 → low, which is good (few class 1s missed).
- FPR 0.75 → too high, lots of false positives.
- F1 0.866 → good balance between precision and recall for class 1.
- AUC 0.88 → not bad, model separates classes reasonably.

Random Forest - GridSearchCV

```
Confusion Matrix (Accuracy 0.9107)

       Prediction
Actual   0   1
     0  25  15
     1   0 128
Confusion Matrix (Accuracy 0.8056)

       Prediction
Actual  0  1
     0  4 12
     1  2 54
```

```
confusion matrix:
 [[ 4 12]
 [ 2 54]]
Accuracy: 0.8055555555555556
Precision:  0.8181818181818182
Recall (Sensitivity):  0.9642857142857143
Specificity:  0.25
FNR:  0.0357142857142857
FPR:  0.75
F1 score:  0.8852459016393442
AUC: 0.8660714285714286
```



Model Interpretations:

- Accuracy of 91% on train set and 80% on test set.
    - Better model → less overfitting.

Test set interpretations:

- Accuracy 80% → higher, better performance.
- Precision 81% → not bad, 81% of predicted positives were correct.
- Recall 96% → model detects most of the actual positives well.
- Specificity 25% → only 25% of actual class 0s were correctly classified, very poor.
- FNR 0.035 → low, which is good (few class 1s missed).
- FPR 0.75 → too high, lots of false positives.
- F1 0.88 → good balance between precision and recall for class 1.
- AUC 0.87 → not bad, model separates classes reasonably.

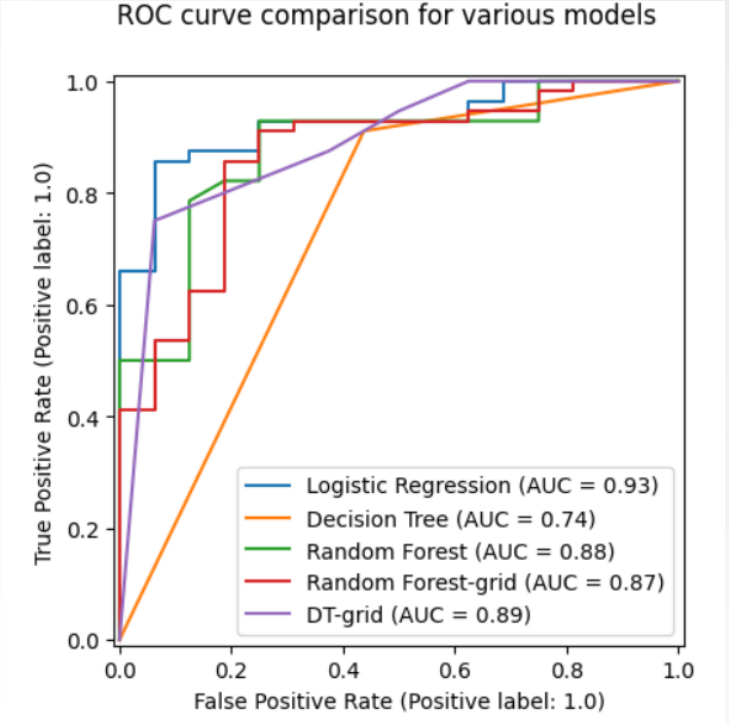
06

Summary

Summary

Best Model: Logistic Regression
- Accuracies on train and test sets are the most balanced.
    - Balanced model, not much overfitting or underfitting.
- Highest AUC score (0.93).
- Low FPR (0.5) and FNR (0.07).

Worst Model: Decision Tree
- Overfitting model with accuracies of 100% and 83% on train and test sets.
- Lowest AUC score (0.74).
- High FPR of 0.43 and low specificity of 0.56.



THANK YOU