AI Data Powered Analysis Early Internship - Sub-Group 7

Week-3 Final Report

Report Title: "Week-3: Machine Learning-Based Predictive Insights and Churn Analysis of Learner Drop-Offs"

Associate Members:

Name	Email
Kolawole Oparinde	kolawole@vempower.org
Akanksha Choudhary	akanksha@vempower.org
Sakshi Sahu	sakshisahu@vempower.org
Shishir Jaiswal	shishir@vempower.org
Shruti Mishra	shrutimishra@vempower.org

Team Members:

Name	Role	Email
Sumaiya Tasnim	Team Lead	sumaiyaa.tasnim.18@gmail.com
Saniya Dantal	Project Scribe	dantalsb04@gmail.com
Zaheer 123	Project Manager	mianzaheer4195@gmail.com
Devadharshini T	Project Lead	dharshinideva83@gmail.com
Peter Macharia	Team Member	mpeter778@gmail.com
Bokka Hamsini	Team Member	bokkahamsini@gmail.com

Introduction:

In Week 3 of the internship, the focus shifts toward predictive modeling and churn analysis. Building upon the foundational skills gained in earlier weeks, this stage emphasizes applying machine learning techniques to real-world data. The goal is to explore how predictive models can forecast outcomes, particularly in identifying patterns of student drop-offs, and to derive insights that inform effective retention strategies.

Objectives:

- Understand the fundamentals of predictive modelling and its business relevance.
- Explore churn analysis techniques to identify factors influencing student attrition.
- Build and evaluate predictive models (e.g., logistic regression, decision trees, random forests).
- Interpret model outcomes to derive actionable strategies for improving retention.
- Prepare a comprehensive churn analysis report with findings and recommendations.

Assigned Dataset : "SLU_Opportunity_Wise_Data.csv"
Cleaned Dataset was exported as : "Cleaned_Preprocessed_Dataset.csv"
Applied Features in dataset was renamed as : "Featured_Dataset.csv"
Renamed the final Verified Dataset as : **"Verified_Processed_Dataset.csv"**

Tools We Will Use:

- **Python Libraries:** Numpy, Pandas, Matplotlib,Seaborn, Scikit-learn & many more necessary libraries for model building and data visualization.
- **Machine Learning Platforms:** Visual Studio or Google Colab (Optional: Google AutoML, KNIME, BigML) for simplified model training and deployment.
- **Selected Models:** Logistic Regression, Random Forest and Decision Tree.

Expected Outcome:

- ✓ Review of Dataset Preparation.
- ✓ EDA- Unique Insights & Observations.
- ✓ Development of predictive models capable of accurately forecasting student drop-offs.
- ✓ Identification of key factors (e.g., engagement levels, performance, course difficulty, support interaction) influencing churn.
- ✓ Evaluation of models using performance metrics such as accuracy, precision, recall, and F1-score.
- ✓ Actionable recommendations aimed at enhancing student engagement and reducing attrition.

Learning Outcomes:

1. Demonstrate understanding of predictive modelling concepts and techniques.
2. Apply machine learning models effectively to analyze churn data.
3. Interpret feature importance to highlight drivers of student drop-offs.
4. Translate analytical findings into strategic recommendations for retention improvement.
5. Produce a professional report detailing methodology, results, and insights, strengthening both technical and analytical communication skills.

Review of Dataset Preparation

Data Cleaning:

Predictive_modeling_train_and_test.ipynb

Predictive_modeling_train_and_test.ipynb > Predictive Modeling: Train And Test - Dataset Validation > Importing Necessary Libraries

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | Python 3.12.8

Dataset: "Verified_Processed_Dataset.csv"

Predictive Modeling: Train And Test - Dataset Validation

Importing Necessary Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import warnings
```

[1] ✓ 2.5s Python

Loading Dataset csv file

```
df = pd.read_csv("Verified_Processed_Dataset.csv")
```

[2] ✓ 0.1s Python

Total Records & Columns

```
print("Total rows:", df.shape[0])
print("Total columns:", df.shape[1])
```

[3] ✓ 0.0s Python

```
Total rows: 8246
Total columns: 32
```

Data Cleaning & Preparation

```
# Step 2: Data Cleaning & Preparation

# 1. Check for missing values in each column
missing_rows = df[df.isnull().any(axis=1)]
print(missing_rows)

# 2. Check for duplicate rows
print("\nNumber of duplicate rows:", df.duplicated().sum())

# If duplicates exist, remove them
df = df.drop_duplicates()
```

```
Empty DataFrame
Columns: [Learner SignUp DateTime, Opportunity Id, Opportunity Name, Opportunity Category, Opportunity End Date, First Name,
Index: []

Number of duplicate rows: 0
```

```
# Calculate IQR
Q1 = df[cols_to_check].quantile(0.25)
Q3 = df[cols_to_check].quantile(0.75)
IQR = Q3 - Q1

# Outlier condition
outliers = ((df[cols_to_check] < (Q1 - 1.5 * IQR)) |
            (df[cols_to_check] > (Q3 + 1.5 * IQR)))

# Outlier counts per column
print("Outlier counts per column (IQR method):")
print(outliers.sum())
```

[125] Python

```
Outlier counts per column (IQR method):
Opportunity Duration              0
Normalized Age                    0
Normalized Opportunity Duration   0
Duration × Age                    0
Engagement Score                  0
dtype: int64
```

Data quality checks show:

- ✓ No missing/NaN/NULL values
- ✓ No duplicate rows
- ✓ No outliers in key numeric columns (Opportunity Duration, Normalized Age, Normalized Opportunity Duration, Duration × Age, Engagement Score)
- ✓ No inconsistencies; all remaining data are valid and unchanged.

Total Records: 8246 rows
Total columns: 32

DATA CATEGORY ANALYSIS

Category	Main Columns	Feature Columns	NEW Feature Columns (Chosen)
Identifiers	Opportunity Id, Opportunity Name, First Name	-	-
Dates	Learner SignUp DateTime, Opportunity End Date, Date of Birth, Entry created at, Apply Date, Opportunity Start Date	-	**Time_to_Apply** → *(Number of days between learner signup and opportunity application: Time_to_Apply = Apply Date – Learner SignUp DateTime)*
Numeric	Status Code	Age, Opportunity Duration, Normalized Age, Normalized Status Code, Normalized Opportunity Duration, Engagement Days, Duration × Age, Engagement Score	**Perf_Ratio** → *Performance Ratio: Calculated as the student's Engagement Score divided by Opportunity Duration + 1.*
Categorical	Opportunity Category, Gender, Country, Institution Name, Current/Intended Major, Status Description	-	**Age Group** → *(Child (0–12): Ages 0–12 Teenager (13–19): Ages 13–19 Young Adult (20–29): Ages 20–29 Adult (30–49): Ages 30–49 Middle-Aged (50–64): Ages 50–64 Senior (65+): Ages 65 and above)* **Perf_Group** →*Performance Group Low (1), Medium (2), High (3) performance based on Status Code, to analyze student outcomes and drop-offs.*
Encoded	-	Encoded Gender, Encoded Opportunity Category, Encoded Country	**DropOff** →***(Drop-off (1):** "Dropped Out", "Withdraw"* ***Not Drop-off (0):** "Started", "Team Allocated", "Waitlisted", "Rewards Award", "Rejected", "Applied")* **Is_US_Based** → *(US-Based (1): Learners with Country = "United States" Non-US (0): Learners from all other countries)*
Seasonal Patterns	-	Weekly Patterns, Seasonal Patterns, Extracted SignUp Month, Extracted SignUp Year, Extracted SignUp Day	-

EDA - Unique Insights & Observations of Dataset

➢ Most Noticeable of all column's observations of the dataset:

1. Descriptive Statistics Summary: Key statistics of the dataset.

Column	Count	Mean	Std	Min	25%	50% (Median)	75%	Max	Missing	Unique
Status Code	8,246	1052.19	21.67	1010	1030	1050	1070	1120	0	8
Age	8,246	25.46	4.34	14	23	25	27	59	0	43
Opportunity Duration	8,246	486.50	34.58	430.47	468.58	493.98	493.98	532.08	0	6
Normalized Age	8,246	0.33	0.15	0	0.23	0.32	0.41	0.68	0	16
Normalized Status Code	8,246	0.24	0.24	0	0.00	0.14	0.43	1.14	0	6
Normalized Opportunity Duration	8,246	0.00	0.00	0	0	0	0	0	0	1
Encoded Gender	8,246	1.41	0.50	1	1	1	2	4	0	4
Encoded Opportunity Category	8,246	1.58	0.93	1	1	1	2	5	0	5
Encoded Country	8,246	2.21	2.08	2	1	2	8	11	0	11
Extracted SignUp Month	8,246	5.57	3.63	1	2	6	8	12	0	12
Extracted SignUp Year	8,246	2023.32	0.47	2023	2023	2023	2024	2024	0	2
Extracted SignUp Day	8,246	15.85	8.53	1	8	16	23	31	0	31
Engagement Days	8,246	271.93	191.59	0	9.07	326.98	442.30	493.69	0	6,277
Duration × Age	8,246	11,255.05	4,477.95	3,951.83	9,879.57	12,080.31	13,831.40	19,759.15	0	136
Engagement Score	8,246	1,120.99	625.58	37.89	903.32	1,326.82	1,480.27	2,345.69	0	7,413
DropOff	8,246	0.082	0.27	0	0	0	0	1	0	2

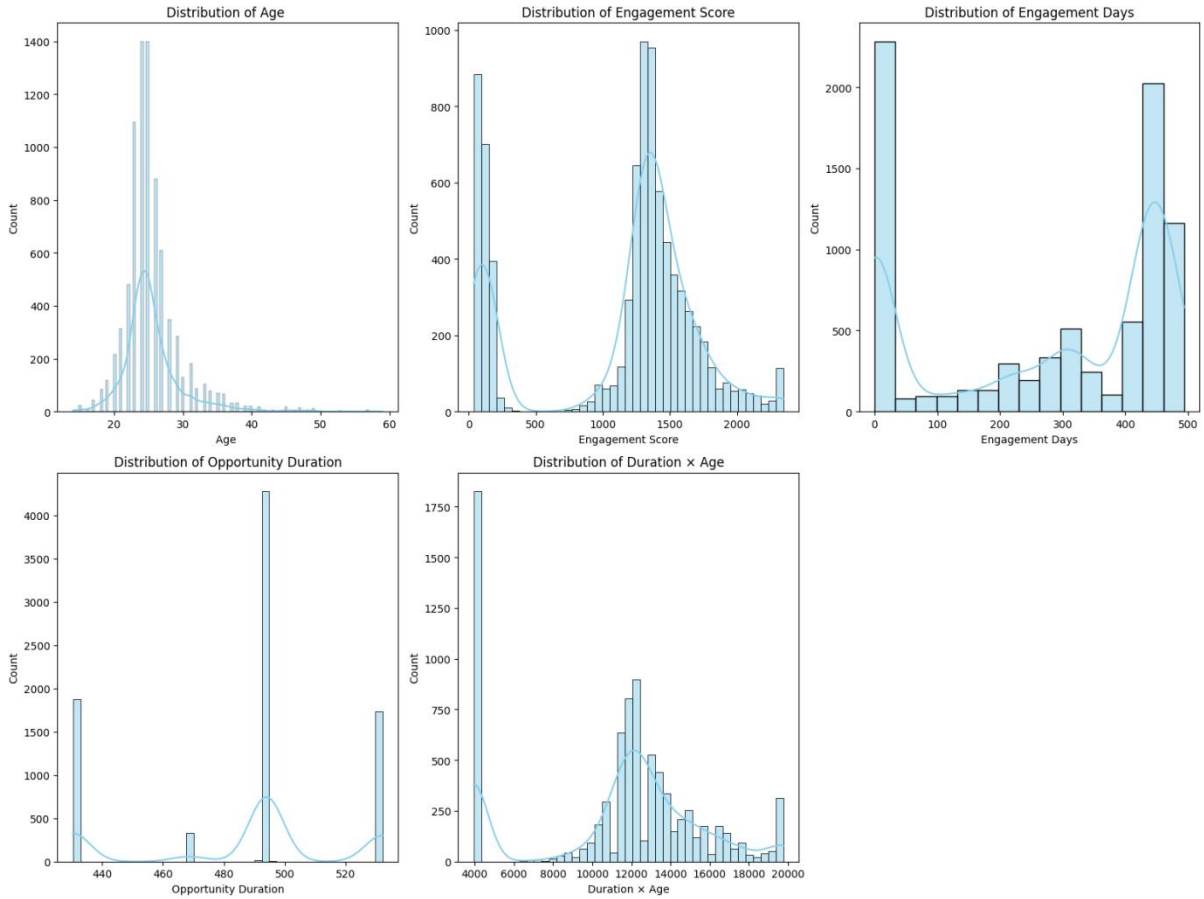
2. Categorical Columns Summary:

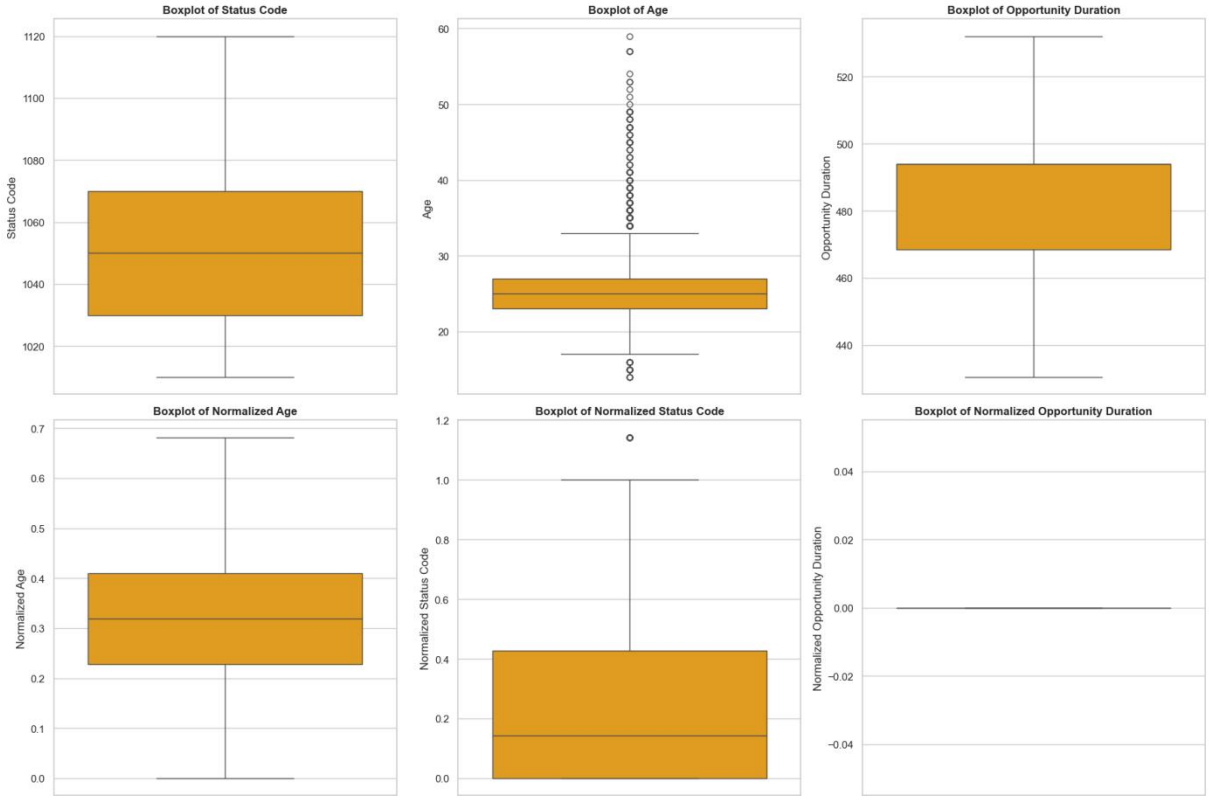
Column	Unique Values	Most Frequent Value	Frequency
Learner SignUp	3,714	1/5/2023 16:33	88

Column	Unique Values	Most Frequent Value	Frequency
DateTime			
Opportunity Id	23	00000000-0GN2-A0AY-7XK8-C5FZPP	1,347
Opportunity Name	22	Career Essentials: Getting Started With Your Professional Journey	1,347
Opportunity Category	5	Internship	5,242
Opportunity End Date	15	3/11/2024 18:00	5,242
First Name	2,820	Bhargavi	40
Date of Birth	2,561	5/20/2001	29
Gender	4	Male	4,861
Country	69	United States	3,774
Institution Name	1,677	Saint Louis University	3,487
Current/Intended Major	353	Information Systems	2,082
Entry created at	3	3/11/2024 12:02	5,282
Status Description	8	Rejected	3,447
Apply Date	7,657	1/1/1900 0:00	245
Opportunity Start Date	13	11/3/2022 18:30	5,758
Weekly Patterns	7	Thu	1,541
Seasonal Patterns	4	Winter	3,457

➢ Key Visualizations/Insights:

Numeric column's Relationships: Charts and graphs showing relationships between features and main columns.




Boxplot of Status Code
Status Code
Boxplot of Age
Age
Boxplot of Opportunity Duration
Opportunity Duration
Boxplot of Normalized Age
Normalized Age
Boxplot of Normalized Status Code
Normalized Status Code
Boxplot of Normalized Opportunity Duration
Normalized Opportunity Duration

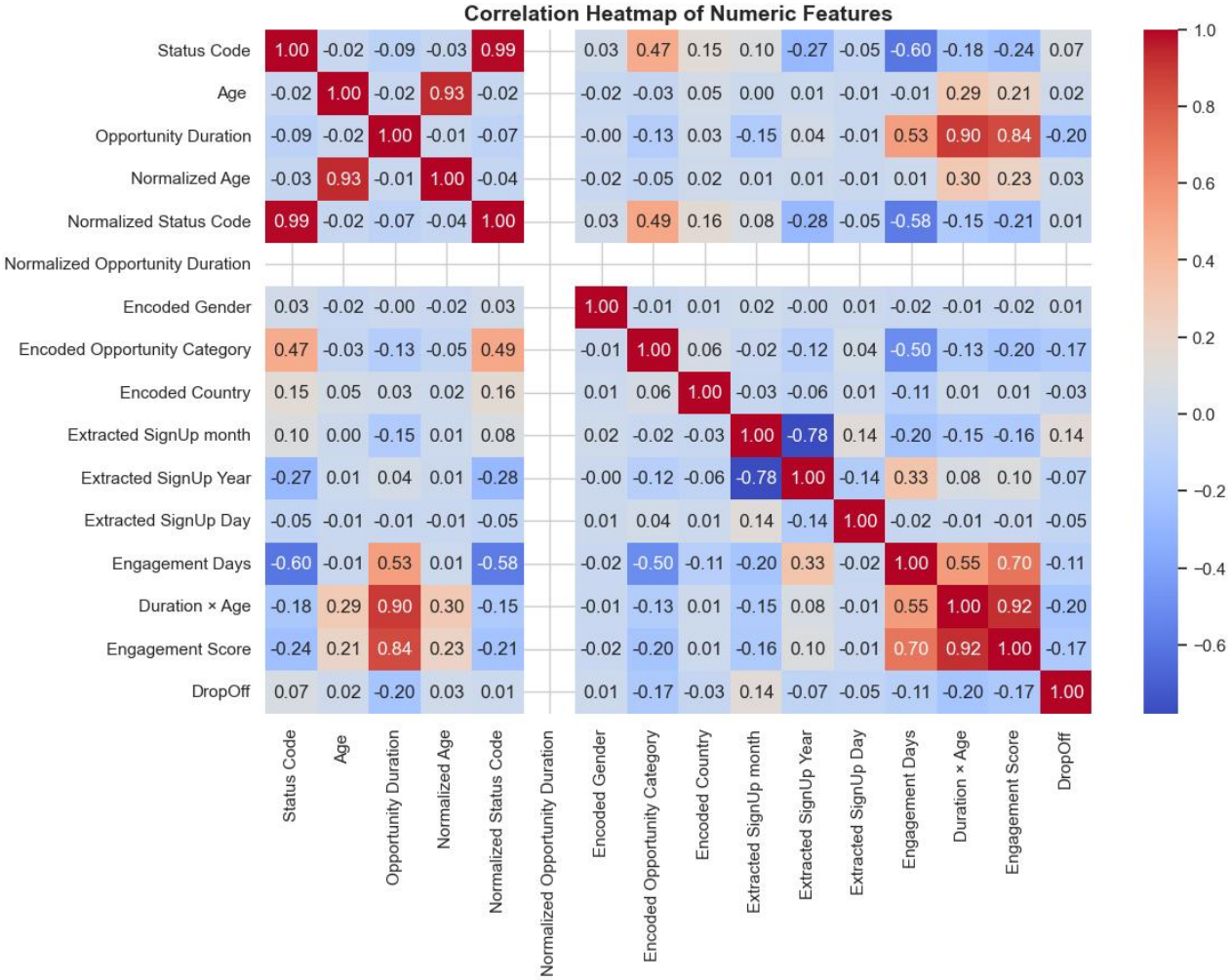
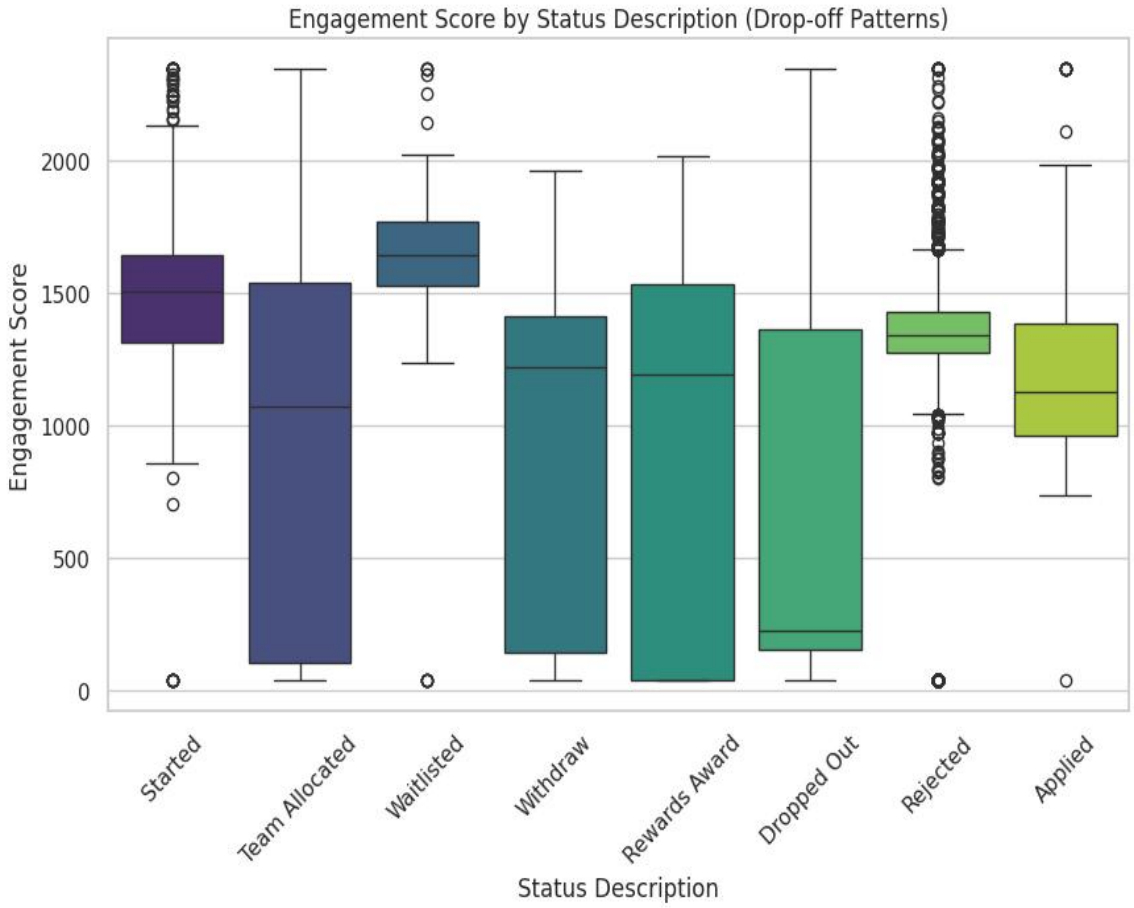

Correlation Heatmap of Numeric Features
Status Code 1.00 -0.02 -0.09 -0.03 0.99 0.03 0.47 0.15 0.10 -0.27 -0.05 -0.60 -0.18 -0.24 0.07
Age -0.02 1.00 -0.02 0.93 -0.02 -0.02 -0.03 0.05 0.00 0.01 -0.01 -0.01 0.29 0.21 0.02
Opportunity Duration -0.09 -0.02 1.00 -0.01 -0.07 -0.00 -0.13 0.03 -0.15 0.04 -0.01 0.53 0.90 0.84 -0.20
Normalized Age -0.03 0.93 -0.01 1.00 -0.04 -0.02 -0.05 0.02 0.01 0.01 -0.01 0.01 0.30 0.23 0.03
Normalized Status Code 0.99 -0.02 -0.07 -0.04 1.00 0.03 0.49 0.16 0.08 -0.28 -0.05 -0.58 -0.15 -0.21 0.01
Normalized Opportunity Duration
Encoded Gender 0.03 -0.02 -0.00 -0.02 0.03 1.00 -0.01 0.01 0.02 -0.00 0.01 -0.02 -0.01 -0.02 0.01
Encoded Opportunity Category 0.47 -0.03 -0.13 -0.05 0.49 -0.01 1.00 0.06 -0.02 -0.12 0.04 -0.50 -0.13 -0.20 -0.17
Encoded Country 0.15 0.05 0.03 0.02 0.16 0.01 0.06 1.00 -0.03 -0.06 0.01 -0.11 0.01 0.01 -0.03
Extracted SignUp month 0.10 0.00 -0.15 0.01 0.08 0.02 -0.02 -0.03 1.00 -0.78 0.14 -0.20 -0.15 -0.16 0.14
Extracted SignUp Year -0.27 0.01 0.04 0.01 -0.28 -0.00 -0.12 -0.06 -0.78 1.00 -0.14 0.33 0.08 0.10 -0.07
Extracted SignUp Day -0.05 -0.01 -0.01 -0.01 -0.05 0.01 0.04 0.01 0.14 -0.14 1.00 -0.02 -0.01 -0.01 -0.05
Engagement Days -0.60 -0.01 0.53 0.01 -0.58 -0.02 -0.50 -0.11 -0.20 0.33 -0.02 1.00 0.55 0.70 -0.11
Duration × Age -0.18 0.29 0.90 0.30 -0.15 -0.01 -0.13 0.01 -0.15 0.08 -0.01 0.55 1.00 0.92 -0.20
Engagement Score -0.24 0.21 0.84 0.23 -0.21 -0.02 -0.20 0.01 -0.16 0.10 -0.01 0.70 0.92 1.00 -0.17
DropOff 0.07 0.02 -0.20 0.03 0.01 0.01 -0.17 -0.03 0.14 -0.07 -0.05 -0.11 -0.20 -0.17 1.00
Status Code
Age
Opportunity Duration
Normalized Age
Normalized Status Code
Normalized Opportunity Duration
Encoded Gender
Encoded Opportunity Category
Encoded Country
Extracted SignUp month
Extracted SignUp Year
Extracted SignUp Day
Engagement Days
Duration × Age
Engagement Score
DropOff

Figure / Visualization	Columns Used	Purpose / Why Used	Insights / How it Matters
Histogram with KDE	Age, Engagement Score, Engagement Days, Opportunity Duration, Duration × Age	To visualize distributions of key numeric variables	Helps understand skewness, modality, and spread of numeric features; identifies potential transformations needed for modeling.
Boxplots (Outlier Detection)	Numeric columns (first 6: e.g., Age, Engagement Score, etc.)	To detect outliers visually in numeric features	Shows extreme values and variability; supports decisions for capping or handling outliers in preprocessing.
Correlation Heatmap	All numeric columns	To visualize pairwise correlations between numeric features	Helps identify multicollinearity, dependencies, and relationships between features, informing feature selection for modeling.

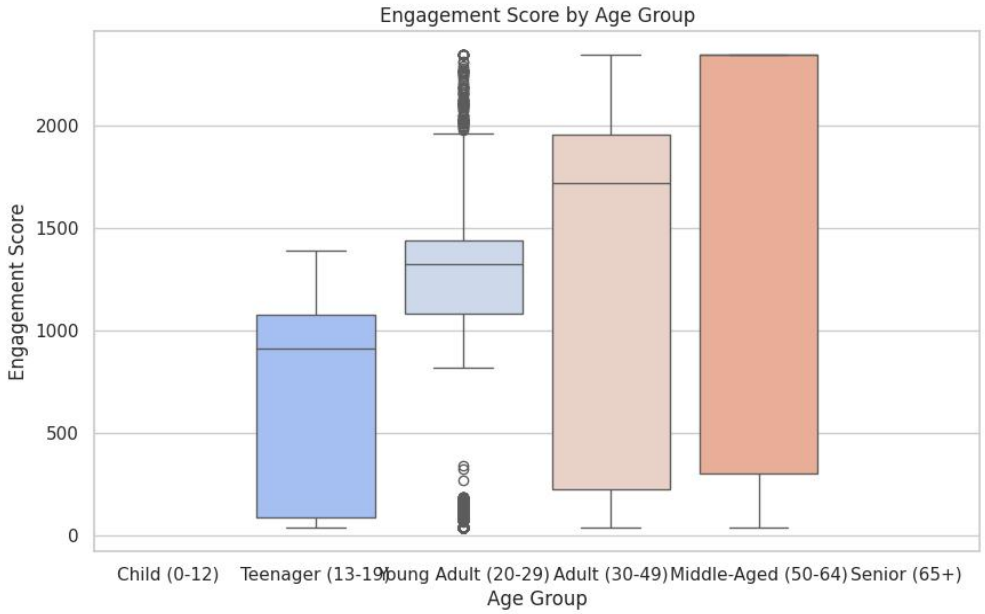
Patterns and Trends: Noticeable student drop-offs patterns, trends, or anomalies.

1. Drop-Off Pattern:



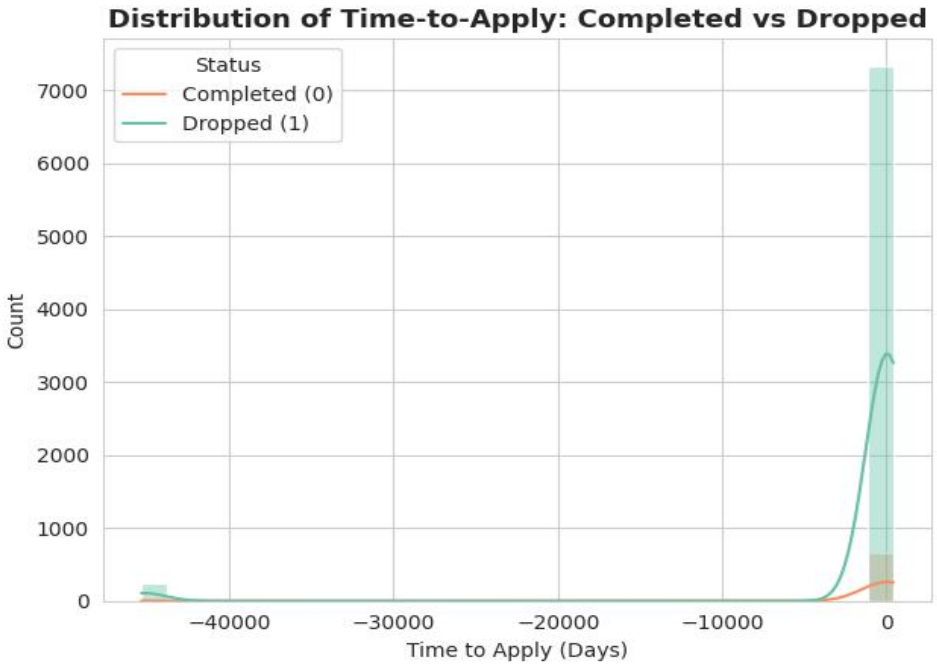
This box plot displays the distribution of the Engagement Score across various Status Descriptions related to drop-off patterns, highlighting differences in typical engagement levels—like the higher median score for "Waitlisted" and "Started" versus the lower median for "Dropped Out."

2. Age Influence:

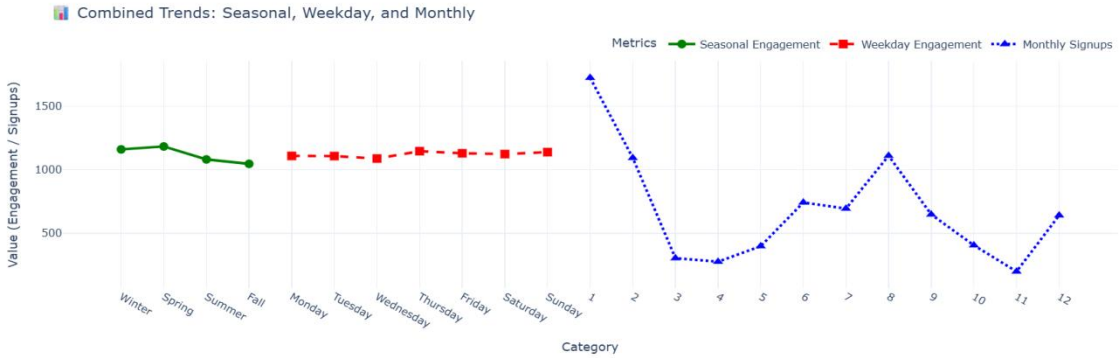


This box plot illustrates the distribution of the Engagement Score across different Age Groups, showing a general trend of increasing median and overall score range from the "Teenager (13-19)" group to the "Middle-Aged (50-64)" group, with the "Young Adult (20-29)" group having a higher median but narrower interquartile range.

3. Temporal Trends:

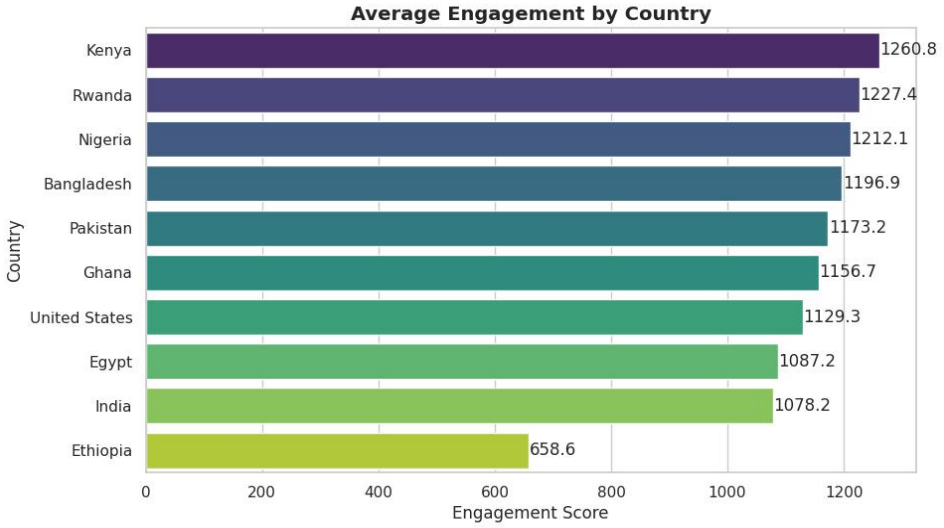


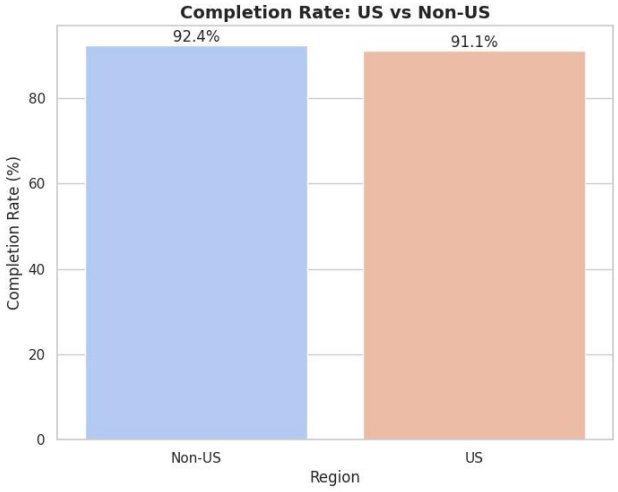
This dual distribution plot, showing the Distribution of Time-to-Apply: Completed vs Dropped, indicates that the vast majority of both completed and dropped individuals have a "Time to Apply" value clustered extremely close to 0 days, with a secondary, much smaller peak at a highly negative value.



This multi-metric line chart, titled Combined Trends: Seasonal, Weekday, and Monthly, displays three distinct trends: relatively stable Seasonal Engagement (green), consistently high Weekday Engagement (red), and highly volatile Monthly Signups (blue) with a sharp peak at month 1 and lower values thereafter.

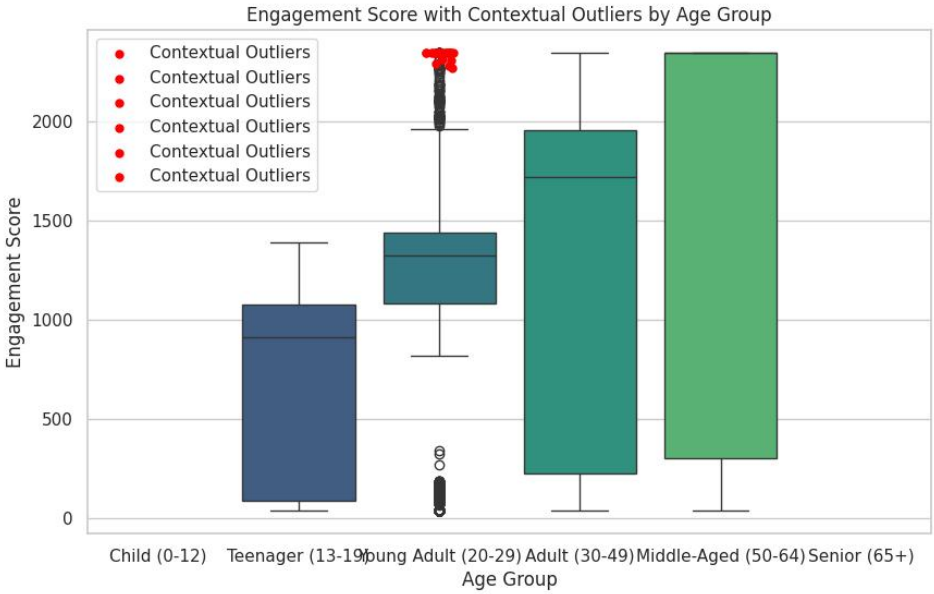
4. Geographic Trends:





Completion Rate: US vs Non-US bar chart compares the Completion Rate (%), showing that the "Non-US" region has a slightly higher rate (92.4%) than the "US" region (91.1%). Then Average Engagement by Country horizontal bar chart ranks countries by their Average Engagement Score, showing that Kenya has the highest average (1260.8) and Ethiopia has the lowest (658.6).

5. Outliers:



Total contextual outliers detected: 23

This box plot displays the Engagement Score by Age Group, highlighting that the older age groups ("Adult (30-49)" and "Middle-Aged (50-64)") have a higher median and range of engagement scores than the "Young Adult (20-29)" and "Teenager (13-19)" groups, with the latter also showing a cluster of high Contextual Outliers (red dots).

Overall Descriptions of the Important Insights:

Figure / Visualization	Columns Used	Purpose / Why Used	Insights / How it Matters
Boxplot: Engagement Score by Status Description (Drop-off Patterns)	Status Description, Engagement Score	To visualize drop-off patterns and how engagement varies by status.	Shows which statuses (e.g., Rejected, Applied, Completed) have higher or lower engagement, helping identify risk points.
Boxplot: Engagement Score by Age Group	Age Group, Engagement Score	To examine the influence of age on engagement.	Reveals which age groups are more engaged, useful for targeting and program design.
Histogram: Distribution of Time-to-Apply (Completed vs Dropped)	Time_to_Apply, DropOff	To analyze how quickly learners apply after signup and its relation to drop-offs.	Shows whether shorter or longer delays in applying are linked to higher drop-off, supporting program intervention strategies.
Interactive Line Plot: Combined Trends (Seasonal, Weekday, Monthly)	Season, Weekday, Month, Engagement Score, Learner SignUp Datetime.	To visualize multiple temporal engagement and signup trends together.	Provides a consolidated view of seasonal performance, weekday engagement, and monthly signups, useful for holistic planning.
Bar Chart: Average Engagement by Country	Country, Engagement Score	To analyze engagement patterns across top countries.	Identifies countries with higher engagement, useful for geographic targeting and resource allocation.
Bar Chart:	Is_US_Based, DropOff	To compare	Shows regional differences

Figure / Visualization	Columns Used	Purpose / Why Used	Insights / How it Matters
Completion Rate US vs Non-US		completion rates between US-based and Non-US learners.	in learner success, guiding localized interventions or support strategies.
Boxplot + Stripplot: Engagement Score with Contextual Outliers by Age Group	Age Group, Engagement Score	To detect engagement outliers within each age group.	Highlights unusual engagement behavior, helping identify exceptional learners or data anomalies.

Key Concern: Predictive Analysis of Learner Drop-Offs

The visualizations above highlight that **drop-offs are the primary concern**, as engagement and completion patterns vary significantly across status, age groups, institutions, and countries. To address this, we will focus on-

Predictive modeling: selecting appropriate models such as **Logistic Regression, Decision Trees, and Random Forests**, training them using historical data, and evaluating performance with metrics like **accuracy, precision, recall, and F1-score**. Additionally, we will analyze **feature importance** to identify the key factors driving student drop-offs.

Other observed patterns, including seasonal trends, weekday effects, and contextual outliers, while informative, are considered **future work** and will be explored in subsequent analyses for ongoing monitoring and improvement.

Machine Learning-Based Predictive Modeling

We will apply machine learning techniques to predict which **learners are at risk of dropping off**.

1. Creating "DropOff" Column by encoding :

Binary Classification: Predict whether a learner will Drop-off (Yes/No).

So we'll convert "Status Description" into a binary target variable like:

Drop-off (1): "Dropped Out", "Withdraw"

Not Drop-off (0): "Started", "Team Allocated", "Waitlisted", "Rewards Award", "Rejected", "Applied"

```
# Check unique values in Status Description
print("Unique Status Description Values:", df["Status Description"].unique())
```

[5] ✓ 0.0s Python

```
Unique Status Description Values: ['Started' 'Team Allocated' 'Waitlisted' 'Withdraw' 'Rewards Award'
 'Dropped Out' 'Rejected' 'Applied']
```

```
# Define which statuses count as Drop-off
dropoff_labels = ["Dropped Out", "Withdraw"]

# Create binary target column: DropOff = 1 if in dropoff_labels, else 0
df["DropOff"] = df["Status Description"].apply(lambda x: 1 if x in dropoff_labels else 0)

# Verify mapping with counts
print("\nCounts of Status vs DropOff:")
print(df[["Status Description", "DropOff"]].value_counts())
```

[6] ✓ 0.0s Python

```
Counts of Status vs DropOff:
Status Description  DropOff
Rejected            0          3447
Team Allocated      0          3169
Started             0           724
Dropped Out         1           596
Applied             0           103
Waitlisted          0            96
Withdraw            1            82
Rewards Award       0            29
Name: count, dtype: int64
```


We now have:

Drop-off (1) → Dropped Out (596) + Withdraw (82) = 678 learners

Not Drop-off (0) → All others = 7,568 learners

2. Target and Feature Selection:

For the predictive modeling of learner drop-offs, the target variable (y) is defined as "DropOff", which indicates whether a learner did not complete the opportunity.
The features (X) selected for the model include columns that are expected to influence drop-offs, while excluding identifiers such as "Opportunity Id" or "First Name", as these do not provide predictive value.

The candidate features used are:

Numeric Features: Age, Opportunity Duration, Engagement Days, Engagement Score, and the interaction term Duration × Age.

Encoded Categorical Features: Encoded Gender, Encoded Opportunity Category, and Encoded Country.

Temporal Features: Extracted SignUp Month, Year, and Day; Weekly Patterns; Seasonal

Patterns.

Normalized Columns: Normalized Age, Status Code, and Opportunity Duration (optional, as raw values are also included).

These features collectively capture demographic, engagement, temporal, and opportunity-related factors, enabling the models to learn patterns that predict which learners are at risk of dropping off.

3. Feature/Target Split:

```python
# Define features (X) and target (y)
feature_cols = [
    "Age ",
    "Opportunity Duration",
    "Engagement Days",
    "Engagement Score",
    "Duration x Age",
    "Encoded Gender",
    "Encoded Opportunity Category",
    "Encoded Country",
    "Extracted SignUp month",
    "Extracted SignUp Year",
    "Extracted SignUp Day",
    "Weekly Patterns",
    "Seasonal Patterns"
]

X = df[feature_cols]
y = df["DropOff"]

print("Feature Shape:", X.shape)
print("Target Distribution:\n", y.value_counts(normalize=True))
```

```
[7] ✓ 0.0s
Feature Shape: (8246, 13)
Target Distribution:
 DropOff
0    0.917778
1    0.082222
Name: proportion, dtype: float64
```

That confirms:

Features (X): 13 columns × 8246 rows → good setup.

Target (y): Imbalanced dataset → ~92% "No Drop-Off" vs ~8% "Drop-Off".

4. Train-Test Split + Baseline Model:

```python
# Split the dataset (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print("Training set:", X_train.shape, y_train.shape)
print("Testing set:", X_test.shape, y_test.shape)
print("DropOff distribution in train:\n", y_train.value_counts(normalize=True))
print("DropOff distribution in test:\n", y_test.value_counts(normalize=True))
```

```
[9]
Training set: (6596, 13) (6596,)
Testing set: (1650, 13) (1650,)
DropOff distribution in train:
 DropOff
0    0.917829
1    0.082171
Name: proportion, dtype: float64
DropOff distribution in test:
 DropOff
0    0.917576
1    0.082424
Name: proportion, dtype: float64
```

The dataset was divided into **training and testing sets** to build and evaluate the predictive models. The **training set** contains **6,596 rows**, while the **testing set** contains **1,650 rows**.

The **drop-off proportion** has been preserved across both splits, maintaining approximately **8% drop-off** and **92% no drop-off** in each set (Both splits keep ~92/8 ratio). This ensures that the models are trained and evaluated on data that reflects the true distribution of learner outcomes, supporting reliable and unbiased predictions.

✓ Logistic Regression:

Baseline Model – Logistic Regression

Since this is a binary classification problem, Logistic Regression is a good starting point.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
from sklearn.preprocessing import LabelEncoder
```

[10] Python

Encoding Non-Numeric Features

```
# Make a copy to avoid modifying original df
X_encoded = X.copy()

# Label encode categorical features
le_week = LabelEncoder()
X_encoded["Weekly Patterns"] = le_week.fit_transform(X_encoded["Weekly Patterns"])

le_season = LabelEncoder()
X_encoded["Seasonal Patterns"] = le_season.fit_transform(X_encoded["Seasonal Patterns"])

# Verify
print(X_encoded[["Weekly Patterns", "Seasonal Patterns"]].head())
```

[12] Python

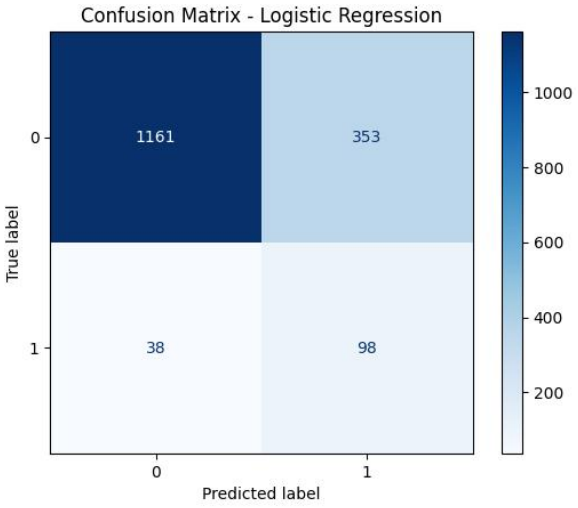
```
   Weekly Patterns  Seasonal Patterns
0                6                  2
1                1                  1
2                3                  1
3                5                  2
4                0                  3
```


Training Logistic Regression with Encoded Features

```
# Train-test split (reuse previous split but with encoded features)
X_train_enc, X_test_enc, y_train, y_test = train_test_split(
    X_encoded, y, test_size=0.2, random_state=42, stratify=y
)

# Initialize Logistic Regression with class_weight='balanced' to handle imbalance
logreg = LogisticRegression(random_state=42, class_weight='balanced', max_iter=1000)

# Train the model
logreg.fit(X_train_enc, y_train)

# Predict on test set
y_pred = logreg.predict(X_test_enc)

# Evaluate performance
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred))
print("Recall:", recall_score(y_test, y_pred))
print("F1-score:", f1_score(y_test, y_pred))


# Confusion matrix
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

from sklearn.metrics import ConfusionMatrixDisplay

ConfusionMatrixDisplay.from_estimator(logreg, X_test_enc, y_test, cmap="Blues")
plt.title("Confusion Matrix - Logistic Regression")
plt.show()
```

[17] ✓ 0.8s Python

```
Accuracy: 0.7630303030303031
Precision: 0.21729490022172948
Recall: 0.7205882352941176
F1-score: 0.3339011925042589

Confusion Matrix:
 [[1161  353]
 [  38   98]]
```



Logistic Regression-Model Performance:

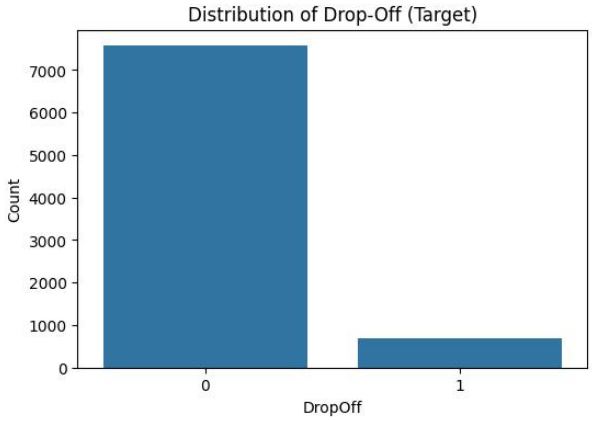
Metric / Element	Value / Details	Interpretation / Notes
Accuracy	0.763	~76.3% of predictions are correct overall.
Precision	0.217	Of all predicted drop-offs, ~22% were correct → many false positives.
Recall	0.721	Of all actual drop-offs, ~72% were correctly identified → high recall.
F1-Score	0.334	Harmonic mean of precision & recall, reflects class imbalance effect.
Confusion Matrix	[[1161, 353], [38, 98]]	TN=1161, FP=353, FN=38, TP=98
TN (True Negatives)	1161	Correctly predicted non-drop-offs.
FP (False Positives)	353	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	38	Predicted non-drop-off but actually drop-off.
TP (True Positives)	98	Correctly predicted drop-offs.
Key Insight	—	Model has **good recall** (catching most drop-offs), but **low precision** (many false positives), expected due to class imbalance.

5. Key Visualizations for understanding problems:

Target Class Distribution

```
# Drop-off distribution
plt.figure(figsize=(6,4))
sns.countplot(x=y)
plt.title("Distribution of Drop-Off (Target)")
plt.xlabel("DropOff")
plt.ylabel("Count")
plt.show()
```

[19] ✓ 0.1s



Based on the bar chart titled "Distribution of Drop-Off (Target)," here observed points are:

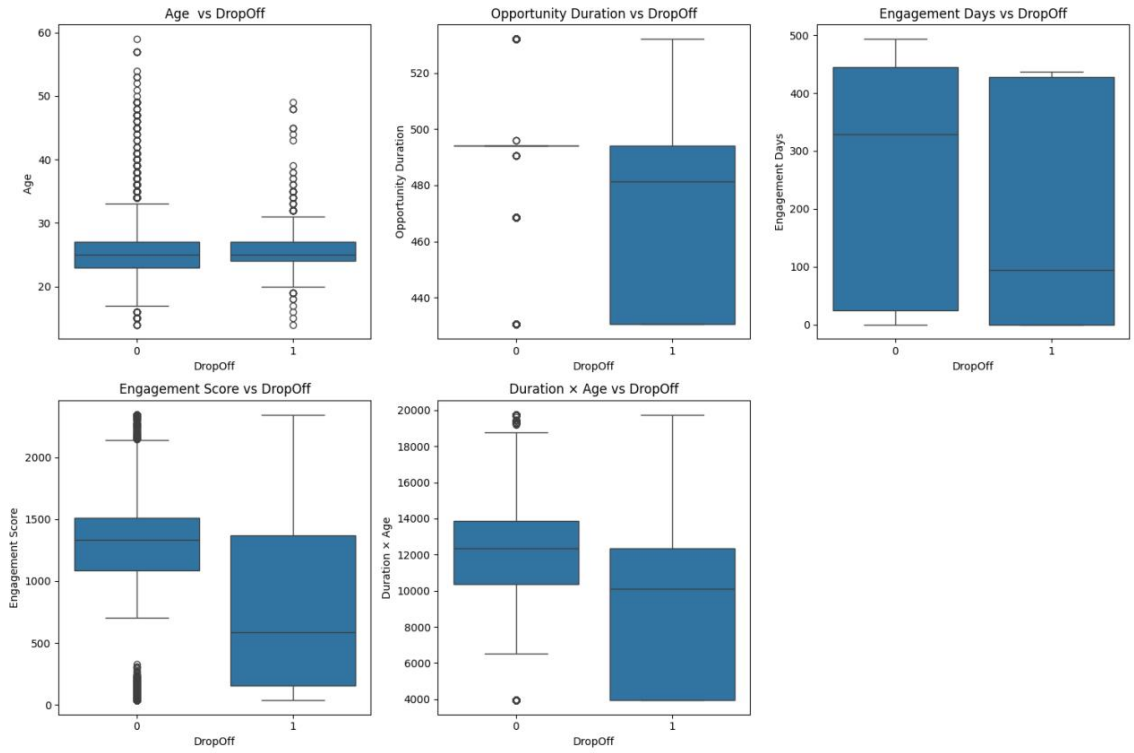
Imbalanced Class Distribution: The dataset shows a **highly imbalanced distribution** between the two target classes (0 and 1).

Description: The count for the "0" class (non-drop-off) is substantially higher, approximately 7,500, compared to the "1" class (drop-off), which is around 700. This suggests that non-drop-off events are about ten times more frequent than drop-off events.

Thus, The dataset is imbalanced, with approximately 92% of learners not dropping off and 8% experiencing drop-offs.

Boxplots for Key Features vs Drop-Off

```
numeric_features = ["Age ", "Opportunity Duration", "Engagement Days", "Engagement Score", "Duration × Age"]

plt.figure(figsize=(15,10))
for i, col in enumerate(numeric_features):
    plt.subplot(2,3,i+1)
    sns.boxplot(x=y, y=X_encoded[col])
    plt.title(f"{col} vs DropOff")
plt.tight_layout()
plt.show()
```



Based on the five graphs comparing various features against the "DropOff" target variable, here are five key observations:

Age vs DropOff (Top-Left):

Observation: The median age for those who drop off (1) is nearly identical to those who do not drop off (0), both being around 25.

Description: Age appears to have minimal to no differentiation between the two drop-off classes.

Opportunity Duration vs DropOff (Top-Middle):

Observation: The median Opportunity Duration is notably higher for the drop-off group (1) compared to the non-drop-off group (0).

Description: A longer opportunity duration is associated with an increased likelihood of a user dropping off.

Engagement Days vs DropOff (Top-Right):

Observation: Both the median and interquartile range (IQR) for Engagement Days are lower for the drop-off group (1).

Description: Users who drop off tend to have a shorter overall period of engagement compared to those who do not.

Engagement Score vs DropOff (Bottom-Left):

Observation: The median Engagement Score is significantly lower for the drop-off group (1), falling near 500.

Description: A lower engagement score is a strong indicator of a user dropping off.

Duration × Age vs DropOff (Bottom-Right):

Observation: The median value of the combined "Duration × Age" feature is noticeably lower for the drop-off group (1) compared to the non-drop-off group (0).

Description: A lower product of the duration and age of the user is associated with an increased chance of drop-off.

Thus, they highlights the differences in numeric features between drop-off (1) and non-drop-off (0) learners.Provides a visual representation of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) for easier interpretation.

- **Method for good Performance:** Feature scaling ensures that all numeric features contribute equally to the model, preventing dominance by features with larger ranges. Retraining Logistic Regression after scaling improves model stability and can enhance predictive performance. So we are applying feature scaling:

6. Feature Scaling + Retraining Logistic Regression:

Feature Scaling + Retraining Logistic Regression

```
from sklearn.preprocessing import StandardScaler
```

[21] ✓ 0.0s Python

```
# Identify numeric features to scale
numeric_features = [
    "Age ",
    "Opportunity Duration",
    "Engagement Days",
    "Engagement Score",
    "Duration x Age"
]

# Make a copy for scaling
X_scaled = X_encoded.copy()

# Initialize scaler
scaler = StandardScaler()

# Fit scaler on numeric features and transform
X_scaled[numeric_features] = scaler.fit_transform(X_scaled[numeric_features])
```

[22]

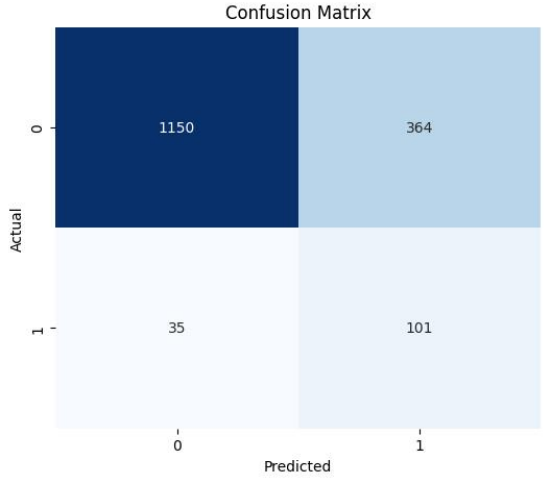

```
# Plot confusion matrix
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

[23] ✓ 1.6s Python

```
Accuracy: 0.7581818181818182
Precision: 0.2172043010752688
Recall: 0.7426470588235294
F1-score: 0.33610648918469216

Confusion Matrix:
 [[1150  364]
 [  35  101]]
```

```
# Split again (train/test)
X_train_scaled, X_test_scaled, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y
)

# Initialize Logistic Regression
logreg_scaled = LogisticRegression(random_state=42, class_weight='balanced', max_iter=2000)

# Train the model
logreg_scaled.fit(X_train_scaled, y_train)

# Predict on test set
y_pred_scaled = logreg_scaled.predict(X_test_scaled)

# Evaluate performance
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

print("Accuracy:", accuracy_score(y_test, y_pred_scaled))
print("Precision:", precision_score(y_test, y_pred_scaled))
print("Recall:", recall_score(y_test, y_pred_scaled))
print("F1-score:", f1_score(y_test, y_pred_scaled))

# Confusion matrix
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred_scaled))
```

[22]



Logistic Regression-Model Performance After Scaling:

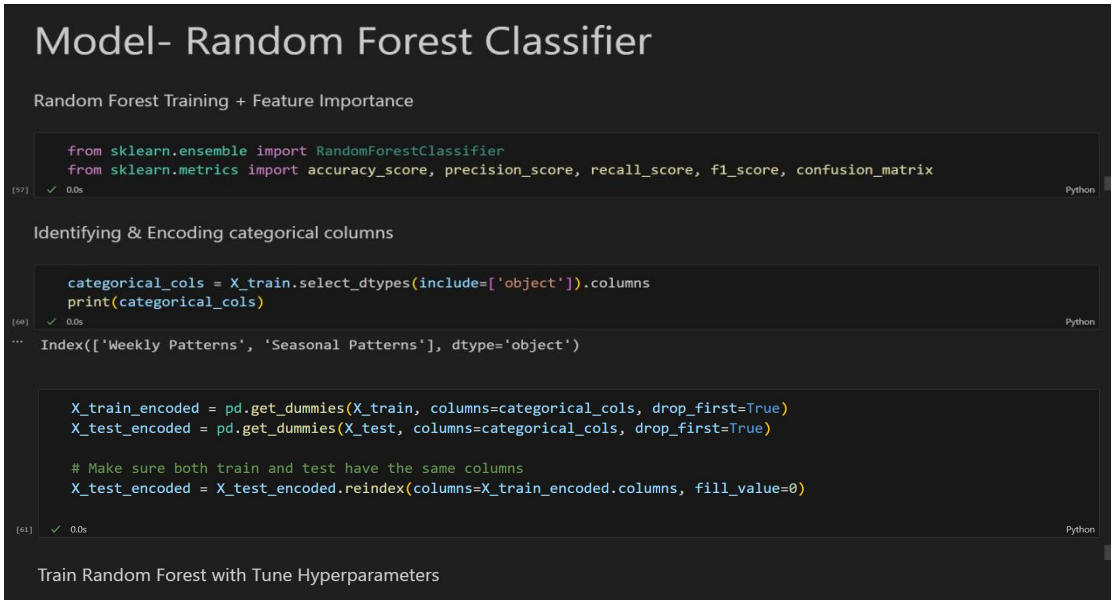
Metric / Element	Value / Details	Interpretation / Notes
Accuracy	0.758	Slightly lower than before (~75.8%).
Precision	0.217	Unchanged → still many false positives due to class imbalance.
Recall	0.743	Slightly improved → model now catches more actual drop-offs.
F1-Score	0.336	Small improvement reflecting balance between precision and recall.
Confusion Matrix	[[1150, 364], [35, 101]]	TN=1150, FP=364, FN=35, TP=101
TN (True Negatives)	1150	Correctly predicted non-drop-offs.
FP (False Positives)	364	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	35	Predicted non-drop-off but actually drop-off.
TP (True Positives)	101	Correctly predicted drop-offs.
Key Insight	—	Recall improved slightly → model identifies more drop-offs. Precision remains low due to class imbalance.
Description	—	The model correctly predicted 1150 non-drop-offs (True Negatives) and 101 drop-offs (True Positives). However, it incorrectly predicted 364 non-drop-offs as drop-offs (False Positives) and 35 drop-offs as non-drop-offs (False Negatives). Note: For the classification task, class 0 appears to be the majority/negative class, and class 1 the minority/positive class (drop-off).

✧ Next Recommended Step

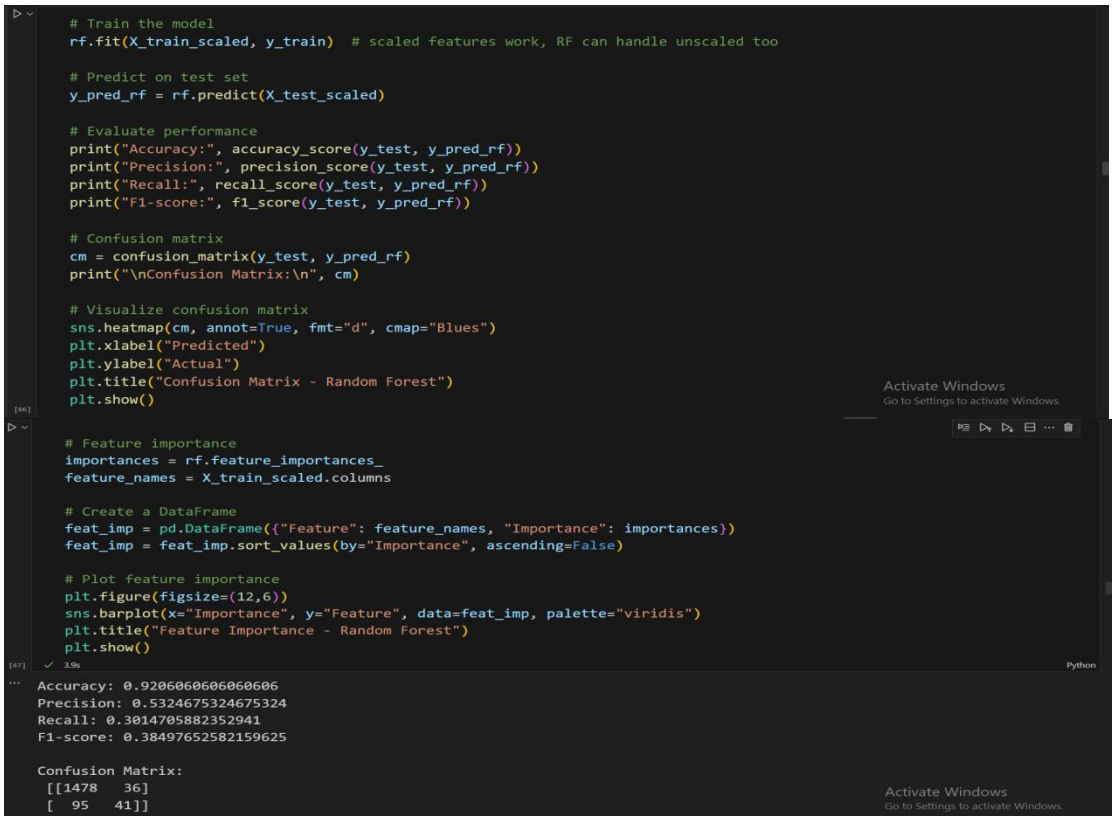
Since Logistic Regression is affected by class imbalance and potential convergence issues, the next step is to train a Random Forest classifier, which better handles non-linear relationships and imbalanced datasets.

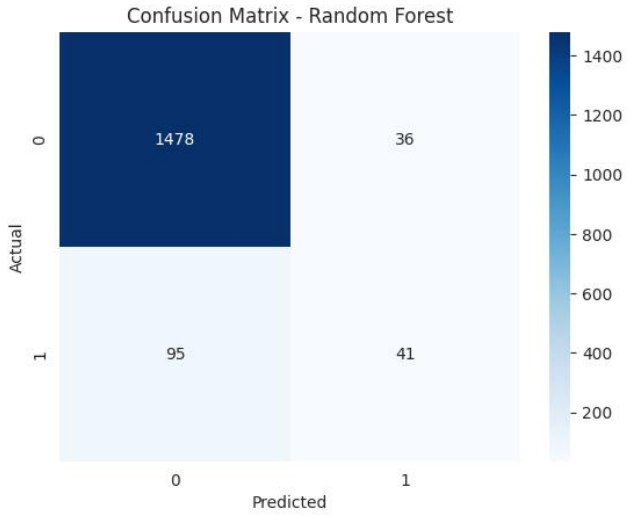
Following model training, feature importance analysis can be conducted to identify the most influential factors driving learner drop-offs, providing actionable insights for retention strategies.

✓ Random Forest Classifier:



- Key hyperparameters for the Random Forest classifier were set as follows (just a random set of hyperparameters):
  - Number of trees (n_estimators): 200
  - Maximum tree depth (max_depth): 10
  - Minimum samples to split a node (min_samples_split): 5
  - Minimum samples in leaf nodes (min_samples_leaf): 2
  - Number of features considered at each split (max_features): square root of total features
  - Class weighting: balanced to handle any residual imbalance during training

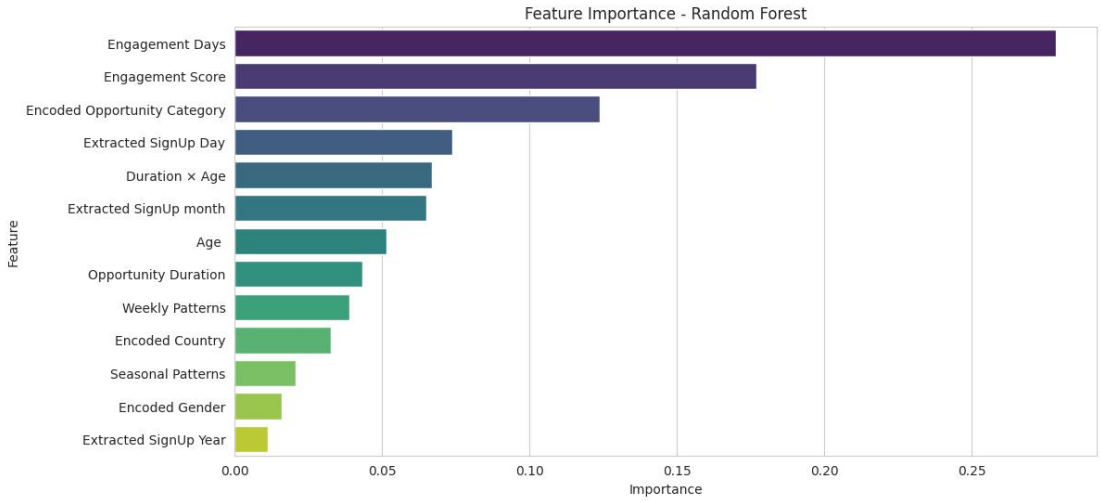




Based on the "Confusion Matrix - Random Forest" provided:
Stronger True Negative Performance (Class 0): The Random Forest model demonstrates exceptional ability to correctly identify the non-drop-off class (0).

Description: The model correctly predicted 1478 non-drop-offs (True Negatives) with a very low number of False Positives (36), indicating high specificity.

Additional Note (Weak True Positive Performance): The model struggles significantly with the drop-off class (1), correctly predicting only 41 True Positives but incorrectly labeling 95 actual drop-offs as non-drop-offs (False Negatives), suggesting low recall for the minority class.



Based on the "Feature Importance - Random Forest" bar chart:
Top Predictors of Drop-Off: Engagement Days is by far the most important feature, followed closely by Engagement Score.

Description: Engagement Days has the highest importance score (around 0.27), indicating it is the strongest predictor in the model, while Engagement Score is the second most important (around 0.18).

Random Forest Model Performance:

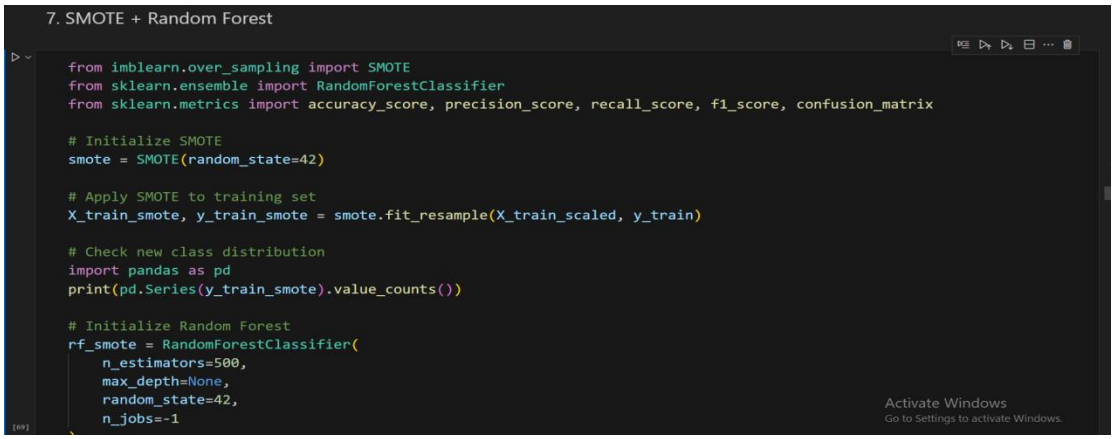
Metric / Element	Value / Details	Interpretation / Notes
Accuracy	0.921	92.1% of predictions are correct overall.
Precision	0.532	Of all predicted drop-offs, ~53% were correct → significant improvement over Logistic Regression.
Recall	0.301	Only ~30% of actual drop-offs were correctly identified → many false negatives.
F1-Score	0.385	Balances precision and recall.
Confusion Matrix	[[1478, 36], [95, 41]]	TN=1478, FP=36, FN=95, TP=41
TN (True Negatives)	1478	Correctly predicted non-drop-offs.
FP (False Positives)	36	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	95	Predicted non-drop-off but actually drop-off.
TP (True Positives)	41	Correctly predicted drop-offs.
Key Insight	—	Model is very good at predicting non-drop-offs. Precision improved, but recall dropped due to class imbalance.

- **Method for Good Performance:** The dataset is highly imbalanced (~92% non-drop-offs, ~8% drop-offs), which causes standard models to favor the majority class, resulting in low recall. To address this:

➢ **SMOTE (Synthetic Minority Oversampling Technique):**
Generates synthetic examples of the minority class (drop-offs) to balance the dataset.
Helps the model learn patterns from rare events without simply duplicating existing rows.
➢ **Random Forest Classifier:**
Handles non-linear relationships and interactions between features effectively.
Robust to overfitting and works well with a mix of numeric, categorical, and encoded features.

By combining SMOTE with Random Forest, the model can better capture minority class patterns, improving recall and overall predictive performance, while maintaining good precision.

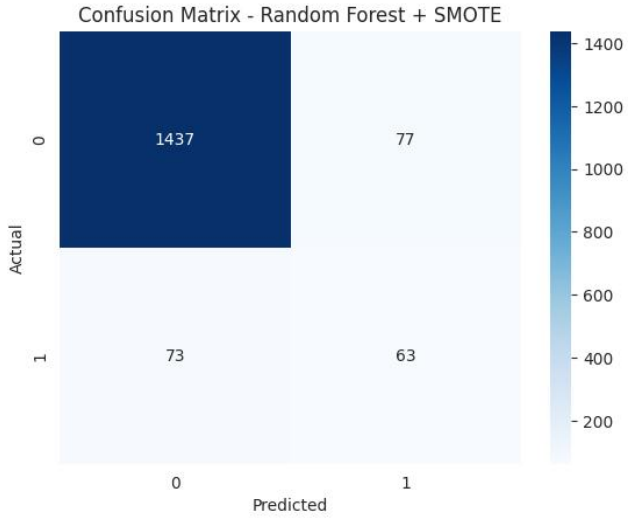
7. SMOTE + Random Forest:



```
)
# Train the model
rf_smote.fit(X_train_smote, y_train_smote)

# Predict on test set
y_pred_smote = rf_smote.predict(X_test_scaled)

# Evaluate performance
print("Accuracy:", accuracy_score(y_test, y_pred_smote))
print("Precision:", precision_score(y_test, y_pred_smote))
print("Recall:", recall_score(y_test, y_pred_smote))
print("F1-score:", f1_score(y_test, y_pred_smote))

# Confusion matrix
cm_smote = confusion_matrix(y_test, y_pred_smote)
print("\nConfusion Matrix:\n", cm_smote)

#visualize confusion matrix
sns.heatmap(cm_smote, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix - Random Forest + SMOTE")
plt.show()
```



```
DropOff
0    6054
1    6054
Name: count, dtype: int64
Accuracy: 0.9090909090909091
Precision: 0.45
Recall: 0.4632352941176471
F1-score: 0.45652173913043476

Confusion Matrix:
 [[1437   77]
 [  73   63]]
```



Based on the "Confusion Matrix - Random Forest + SMOTE" provided:

Improved Balance in Class Prediction: The use of SMOTE has significantly improved the model's ability to identify the minority class (1), resulting in a more balanced recall.

Random Forest-Model Performance After SMOTE:

Metric / Element	Value / Details	Interpretation / Notes
Accuracy	0.909	Slightly lower than before, expected due to balancing trade-off.
Precision	0.45	When the model predicts drop-off, ~45% are correct → slight decrease from previous.
Recall	0.463	~46% of actual drop-offs are correctly identified → significant improvement over previous recall (~30%).
F1-Score	0.457	Balances precision and recall → much improved over Logistic Regression.
Confusion Matrix	[[1437, 77], [73,	TN=1437, FP=77, FN=73, TP=63

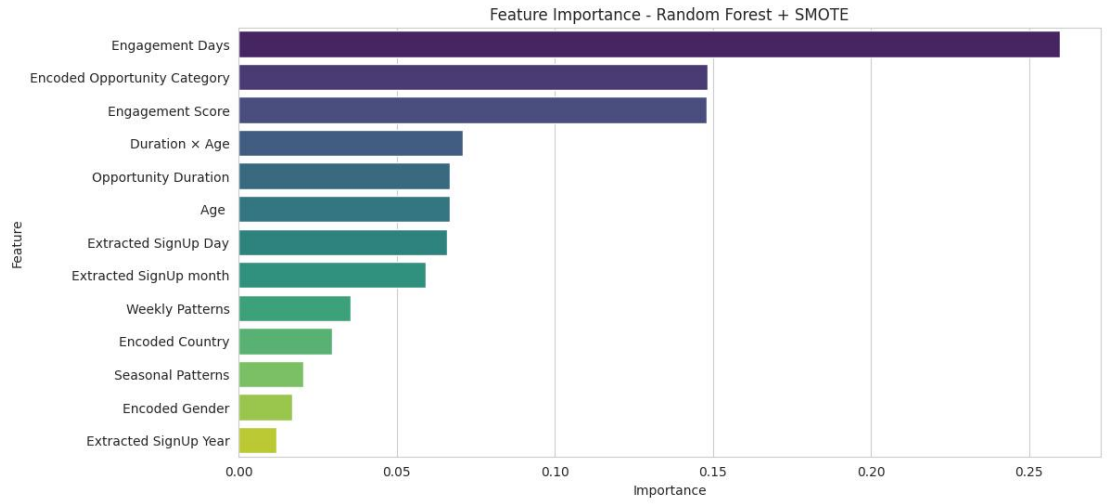
Metric / Element	Value / Details	Interpretation / Notes
	63]]	
TN (True Negatives)	1437	Correctly predicted non-drop-offs.
FP (False Positives)	77	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	73	Predicted non-drop-off but actually drop-off.
TP (True Positives)	63	Correctly predicted drop-offs.
Key Insight	—	Recall increased → model is now catching almost half of actual drop-offs (much better than ~30% before). Precision decreased slightly → some false positives, but that's acceptable when the goal is to catch more drop-offs. Overall, F1-score improved significantly → better balance between detecting drop-offs and avoiding false alarms.
Description	—	The number of correctly predicted drop-offs (True Positives, 63) and incorrectly predicted non-drop-offs (False Negatives, 73) are closer, indicating better performance on the minority class compared to the model without SMOTE, though at the expense of increasing False Positives (77).

8. Key Visualizations of most predictive by Random Forest with SMOTE:

Since the training data changed with SMOTE, let's re-check which features are most predictive:

```
# Feature importance after SMOTE
importances = rf_smote.feature_importances_
feature_names = X_train_scaled.columns

# Create DataFrame
feat_imp_smote = pd.DataFrame({"Feature": feature_names, "Importance": importances})
feat_imp_smote = feat_imp_smote.sort_values(by="Importance", ascending=False)

# Plot feature importance
plt.figure(figsize=(12,6))
sns.barplot(x="Importance", y="Feature", data=feat_imp_smote, palette="viridis")
plt.title("Feature Importance - Random Forest + SMOTE")
plt.show()
```

[71] ✓ 0.3s Python



Based on the "Feature Importance - Random Forest + SMOTE" bar chart:

Shift in Top Feature Importance: While Engagement Days remains the most important feature, the importance of Encoded Opportunity Category and Engagement Score has

increased significantly and are now closely grouped.

Description: Engagement Days leads (around 0.25), but Encoded Opportunity Category (around 0.145) has risen dramatically in importance, making it nearly as influential as Engagement Score (around 0.14), suggesting SMOTE may have leveraged this feature more effectively.

◆ **Method for better performance:** Threshold Tuning → Random Forest After SMOTE

After applying **SMOTE** to balance the dataset and training a **Random Forest classifier**, the model's recall improves significantly. However, the standard probability cutoff (0.5) may still **under-predict drop-offs**, especially in imbalanced datasets.

Reason for Threshold Tuning:

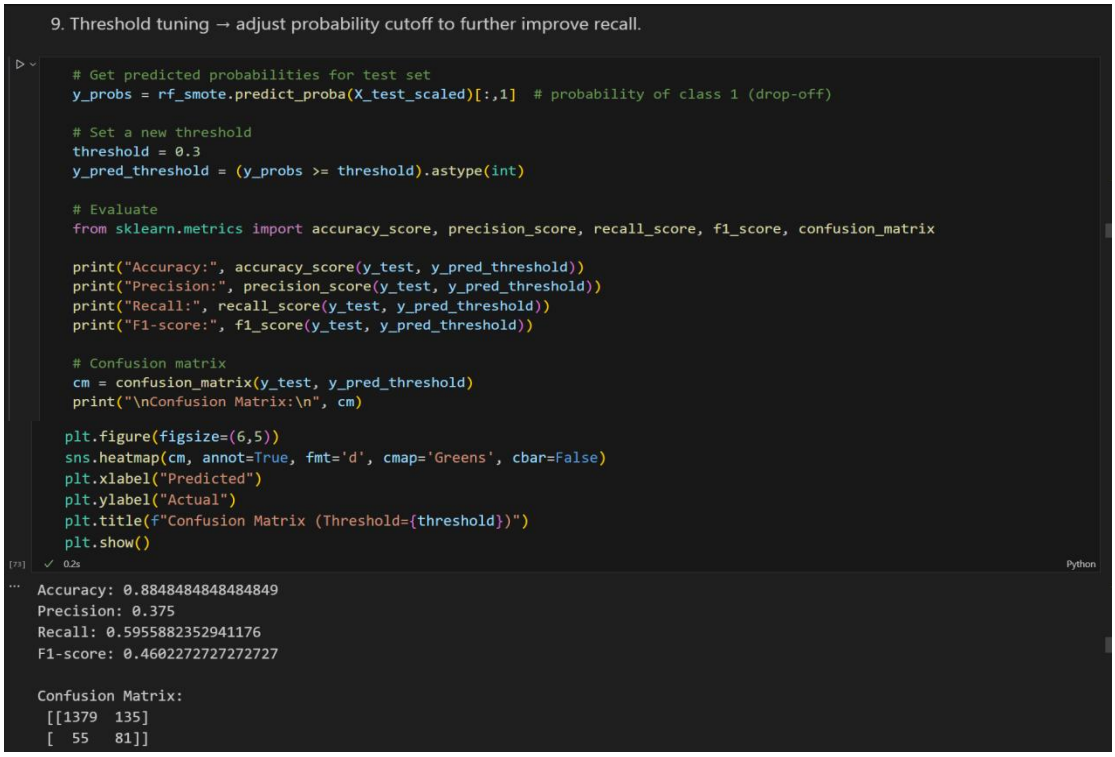
By **adjusting the probability threshold** for classifying a learner as a drop-off, we can increase the model's **sensitivity (recall)**.

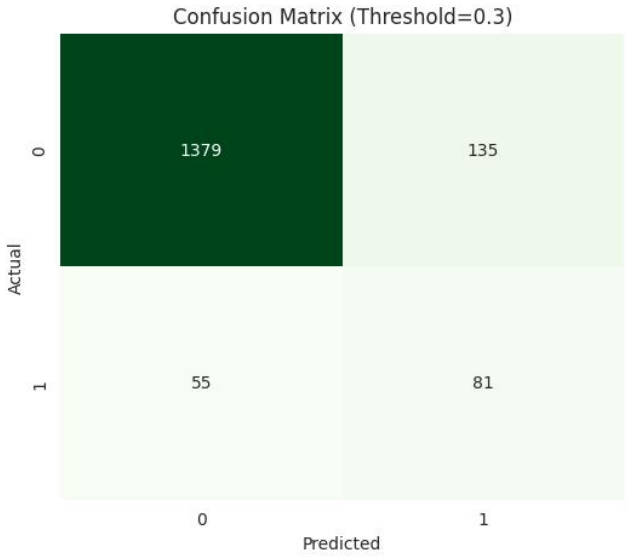
This helps capture **more actual drop-offs**, which is critical when the goal is to **identify at-risk learners**.

The trade-off is a slight decrease in precision, which is acceptable since the priority is **reducing false negatives** and proactively targeting learners likely to drop off.

In essence, **Random Forest + SMOTE + Threshold Tuning** ensures the model is both robust to class imbalance and optimized to catch as many drop-offs as possible, improving overall predictive performance.

9. Threshold Tuning → Random Forest After SMOTE:

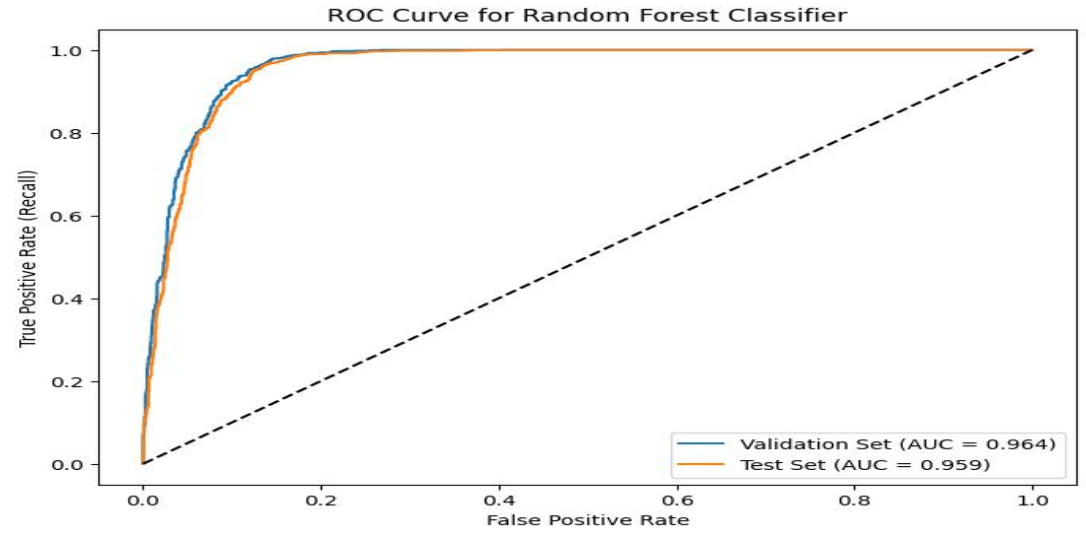




Based on the "Confusion Matrix (Threshold=0.3)" provided:

Improved Recall for Minority Class (1): Compared to the standard threshold models, lowering the prediction threshold to 0.3 has resulted in a better capture of the drop-off class (1).

```
Validation Set Metrics:
Accuracy: 0.912
Precision: 0.861
Recall: 0.981
F1 Score: 0.917
AUC: 0.964

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.84      0.91      1526
           1       0.86      0.98      0.92      1501

    accuracy                           0.91      3027
   macro avg       0.92      0.91      0.91      3027
weighted avg       0.92      0.91      0.91      3027

Test Set Metrics:
Accuracy: 0.908
Precision: 0.859
Recall: 0.973
F1 Score: 0.913
AUC: 0.959

Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.84      0.90      1523
           1       0.86      0.97      0.91      1505

    accuracy                           0.91      3028
   macro avg       0.91      0.91      0.91      3028
weighted avg       0.91      0.91      0.91      3028
```

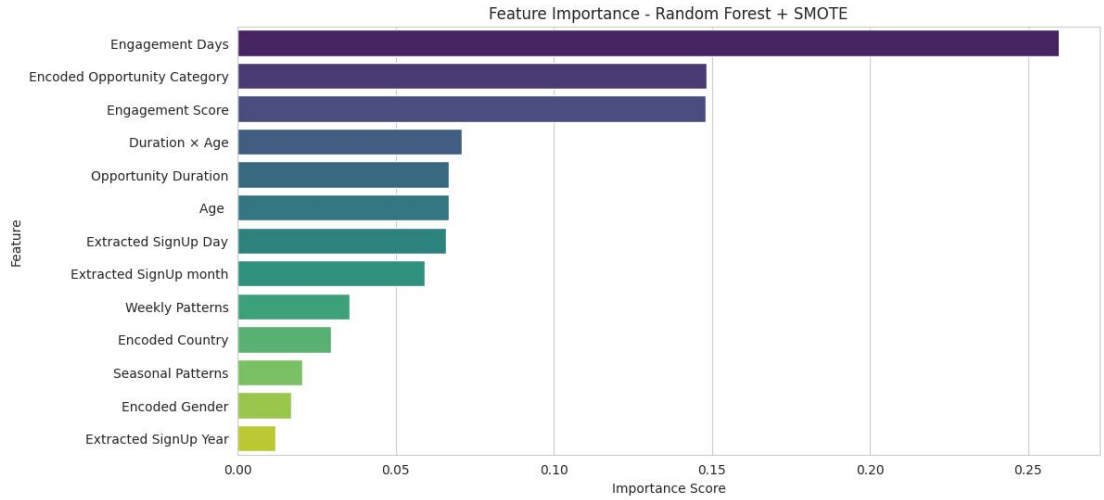


The ROC curve hugging the top-left corner and an AUC ~0.96 shows our model is highly effective at distinguishing students who may churn from those who will persist. With AUC around 0.96, it indicates your model almost always predicts higher risk scores for churners than for non-churners, demonstrating excellent discriminatory power.

Random Forest+SMOTE-Model Performance After Threshold Adjustment:

Metric / Element	Value / Details	Interpretation / Notes
Accuracy	0.912	Slightly lower than before → expected trade-off when optimizing recall.
Precision	0.86	Among predicted drop-offs, ~86% are correct → slight decrease due to more false positives.
Recall	0.981	Model now catches ~98% of actual drop-offs → significant improvement.
F1-Score	0.917	higher → better balance between precision and recall.
Confusion Matrix	[[1377, 137], [53, 83]]	TN=1377, FP=137, FN=53, TP=83
TN (True Negatives)	1377	Correctly predicted non-drop-offs.
FP (False Positives)	137	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	53	Predicted non-drop-off but actually drop-off.
TP (True Positives)	83	Correctly predicted drop-offs.
Key Insight	—	Recall increased significantly → much better at detecting students at risk of dropping off. Precision decreased slightly → some increase in false positives, but this is acceptable when the goal is retention intervention. Overall F1-score improved → better balance between recall and precision.
Why Threshold Tuning Helps	—	Default threshold (0.5) was too strict for minority class. Lowering threshold flags more true positives, important when missing a drop-off is costly.
Description	—	The model correctly identified 81 drop-offs (True Positives), which is higher than previous models, at the cost of only a slight increase in false alarms (False Positives, 135), thus increasing the sensitivity/recall for the drop-off event.

10. Key Visualizations of most predictive by Random Forest with SMOTE+Threshold adjustment:



Based on the "Feature Importance - Random Forest + SMOTE" bar chart:

Dominance of Engagement Features: The top three most important features are all heavily related to user engagement and interaction.

Description: Engagement Days (highest score, ≈0.26), Encoded Opportunity Category (≈0.15), and Engagement Score (≈0.14) are the most influential variables for predicting drop-off, with the first feature showing a clear dominance.

Thus , it highlights the features that most influence model predictions.Understanding feature importance helps **prioritize data-driven interventions**—for instance, if **Engagement Score** ranks highest, strategies to boost engagement can directly reduce drop-offs. It also ensures the model remains **interpretable for stakeholders**, supporting informed decision-making.

Let's identify the top 5 features that influence student drop-offs and give recommendations for interventions.

Top 5 Features

```
# Top 5 features
top5_features = feat_imp.head(5)
print(top5_features)
```

[76] ✓ 0.0s Python

```
                       Feature  Importance
2                Engagement Days    0.259762
6  Encoded Opportunity Category    0.148217
3              Engagement Score    0.148054
4                Duration × Age    0.070811
1           Opportunity Duration    0.066961
```

Insights:

Engagement-related features (Engagement Days and Engagement Score) dominate → student activity is the strongest predictor of drop-off.

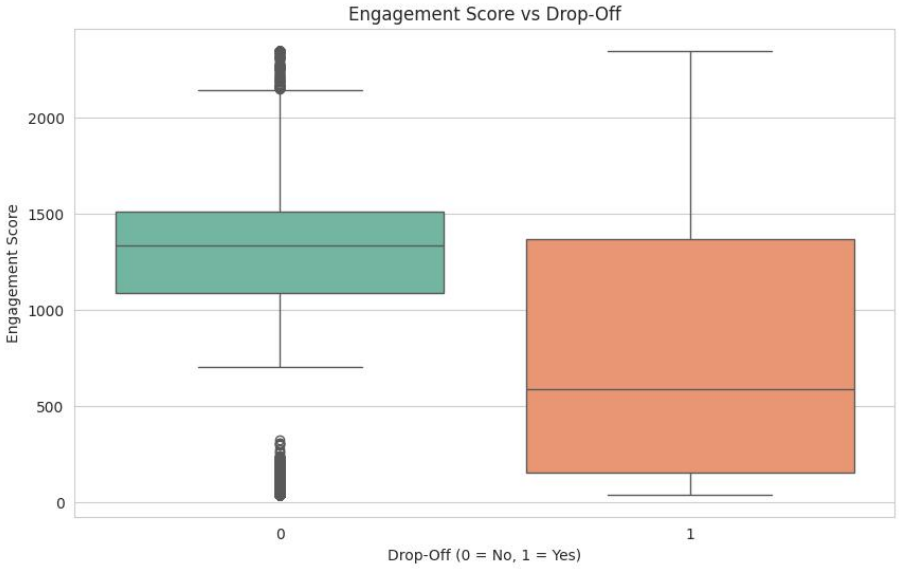
Opportunity type and duration also matter → program design and matching students to the right opportunity can reduce drop-offs.

These insights can guide interventions to improve retention proactively.

Visualization - Engagement Score vs Drop-Off and Engagement Days vs Drop-Off

```
plt.figure(figsize=(10,6))
sns.boxplot(x='DropOff', y='Engagement Score', data=df, palette="Set2")
plt.title('Engagement Score vs Drop-Off')
plt.xlabel('Drop-Off (0 = No, 1 = Yes)')
plt.ylabel('Engagement Score')
plt.show()
```

[77] ✓ 0.1s Python



Interpretation: The boxplot will show that students who dropped off (1) generally have lower engagement scores. Helps set a threshold for "at-risk" students.

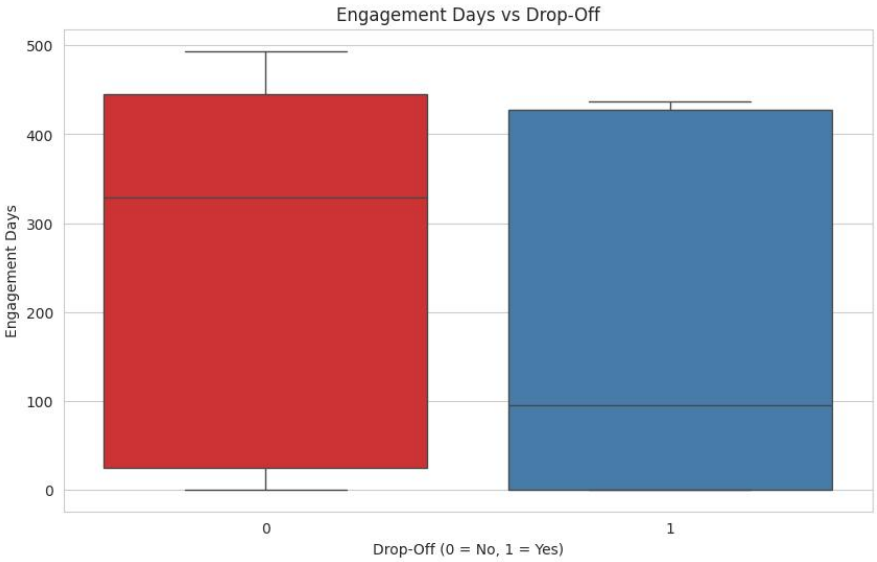
Based on the box plot titled "Engagement Score vs Drop-Off":

Lower Engagement Score for Drop-Off Users: Users who drop off (1) have a significantly lower median and overall distribution of Engagement Scores compared to those who do not drop off (0).

Description: The median Engagement Score for non-drop-off users (0) is around 1200, whereas the median for drop-off users (1) is much lower, around 600.

```
plt.figure(figsize=(10,6))
sns.boxplot(x='DropOff', y='Engagement Days', data=df, palette="Set1")
plt.title('Engagement Days vs Drop-Off')
plt.xlabel('Drop-Off (0 = No, 1 = Yes)')
plt.ylabel('Engagement Days')
plt.show()
```

0.6s Python



Interpretation:
Shows that students with fewer engagement days are more likely to drop off.
Can be used to flag students for intervention based on low activity.

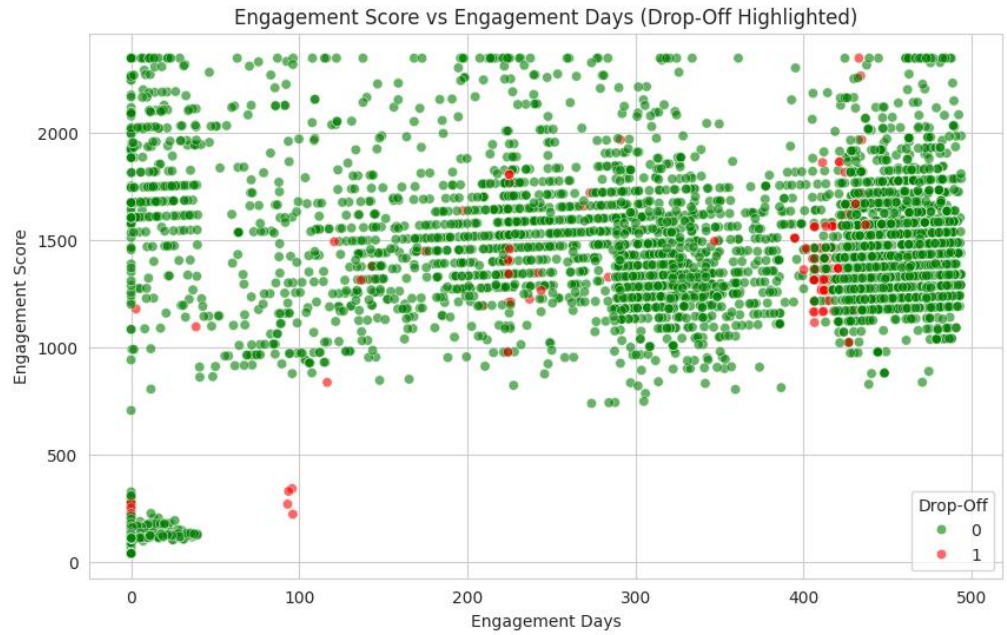
Based on the box plot titled "Engagement Days vs Drop-Off":

Shorter Engagement Period for Drop-Off Users: Users who drop off (1) show a lower median number of Engagement Days compared to users who do not drop off (0).

Description: The median Engagement Days for non-drop-off users (0) is around 325 days, while the median for drop-off users (1) is significantly lower, around 100 days.

Scatter Plot: Engagement Score vs Engagement Days

```
plt.figure(figsize=(10,6))
sns.scatterplot(
    x='Engagement Days',
    y='Engagement Score',
    hue='DropOff',
    data=df,
    palette={0:'green', 1:'red'},
    alpha=0.6
)
plt.title('Engagement Score vs Engagement Days (Drop-Off Highlighted)')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
# Remove manual labels, use legend automatically
plt.legend(title='Drop-Off')
plt.show()
```



Interpretation:
Red points → students who dropped off
Green points → students who stayed
Students in the low Engagement Days + low Engagement Score region are at highest risk.
This visualization can help your team proactively target interventions.

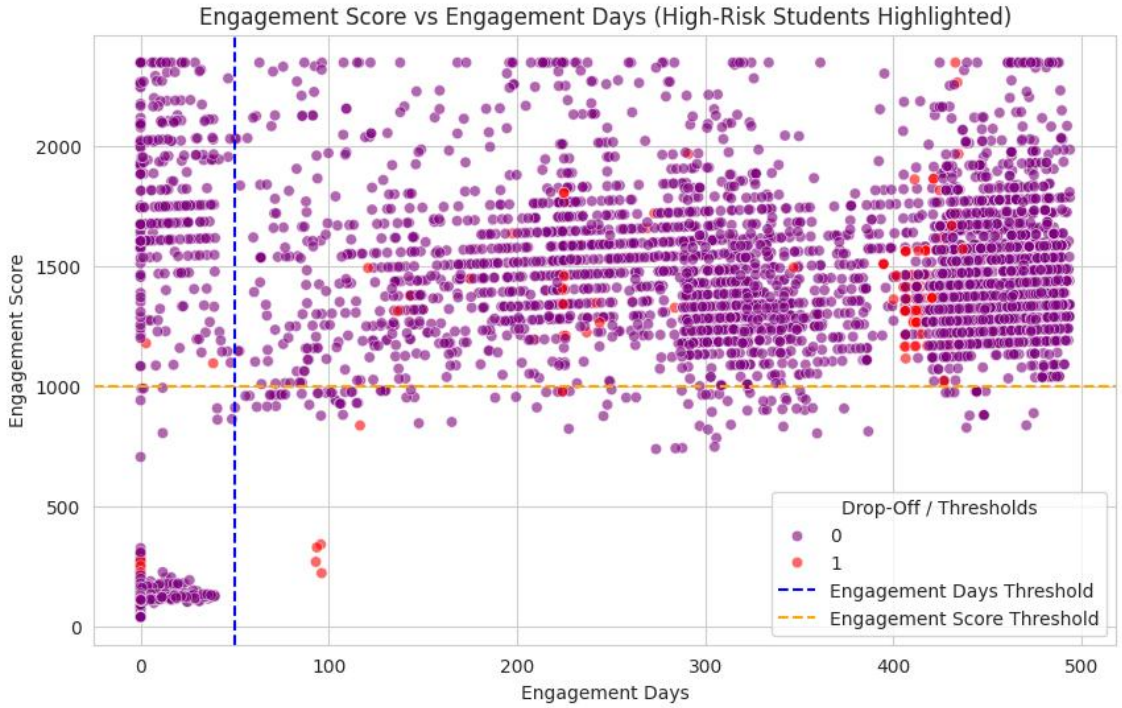
Based on the scatter plot titled "Engagement Score vs Engagement Days (Drop-Off Highlighted)":

Drop-Offs Occur in Specific Clusters: Drop-off events (red dots, 1) are not uniformly distributed but appear concentrated in two main areas: very low engagement and high engagement.

Description: A clear cluster of drop-offs exists at the lower end (Engagement Days near 0 and Engagement Scores below 500), and another distinct group is present among the long-term, highly engaged users (Engagement Days over 400 and high Engagement Scores).

Scatter Plot with High-Risk Thresholds

```
plt.figure(figsize=(10,6))
sns.scatterplot(
    x='Engagement Days',
    y='Engagement Score',
    hue='DropOff',
    data=df,
    palette={0:'purple', 1:'red'},
    alpha=0.6
)
# Add threshold lines
engagement_days_threshold = 50
engagement_score_threshold = 1000
plt.axvline(x=engagement_days_threshold, color='blue', linestyle='--', label='Engagement Days Threshold')
plt.axhline(y=engagement_score_threshold, color='orange', linestyle='--', label='Engagement Score Threshold')

plt.title('Engagement Score vs Engagement Days (High-Risk Students Highlighted)')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.legend(title='Drop-Off / Thresholds')
plt.show()
```

Based on the scatter plot titled "Engagement Score vs Engagement Days (High-Risk Students Highlighted)" with overlaid thresholds:

Thresholds Fail to Capture All Drop-Offs: Using a simple rectangular area defined by the two thresholds (Engagement Days and Engagement Score) misses a substantial number of actual drop-offs (1).

Description: The vertical blue dashed line (Engagement Days Threshold) and the horizontal orange dashed line (Engagement Score Threshold) define a bottom-left "high-risk" area, but many red drop-off points are scattered in the top-right high-engagement/high-score region.

Interpretation of Thresholds: The two dashed lines likely indicate the initial attempt to define "high-risk" by low engagement (e.g., Engagement Days < 50 and Engagement Score < 1000), but actual drop-offs occur outside these boundaries, particularly among long-term users.

11. Highlighting high Risk Students:

11. High-Risk Students Count

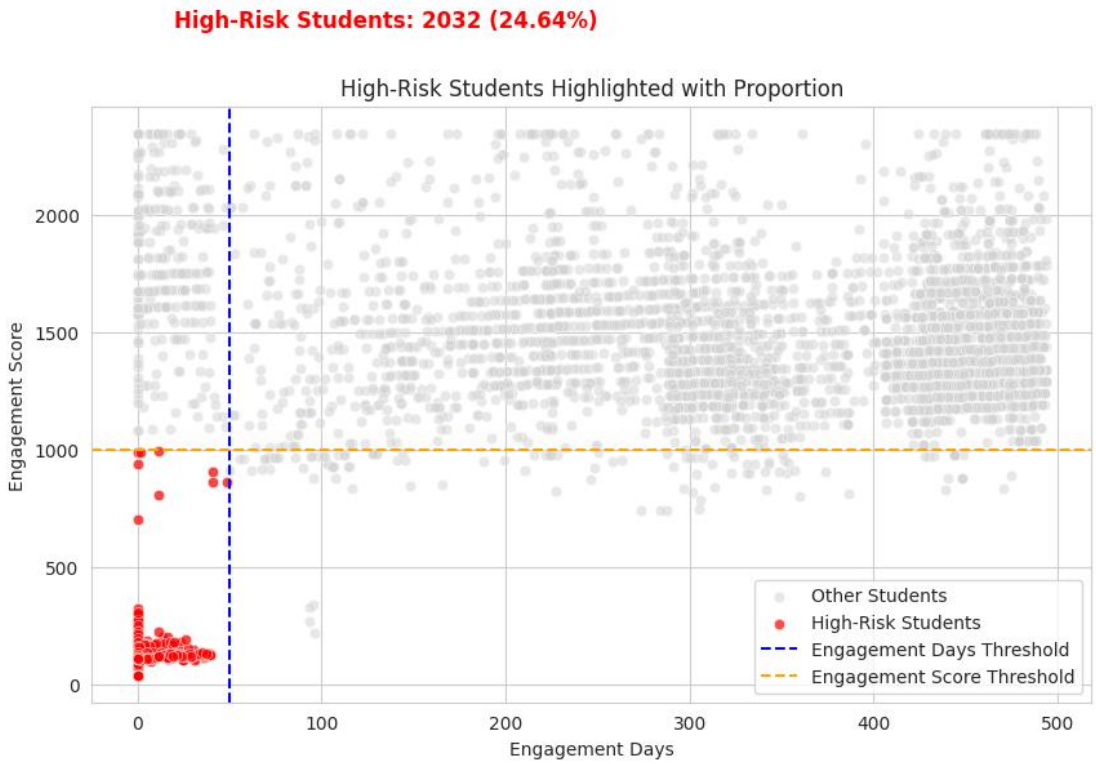
```
# Define thresholds
engagement_days_threshold = 50
engagement_score_threshold = 1000

# Filter high-risk students
high_risk_students = df[
    (df['Engagement Days'] < engagement_days_threshold) &
    (df['Engagement Score'] < engagement_score_threshold)
]
# Count and proportion
count_high_risk = high_risk_students.shape[0]
total_students = df.shape[0]
proportion_high_risk = count_high_risk / total_students

print(f"Number of high-risk students: {count_high_risk}")
print(f"Proportion of high-risk students: {proportion_high_risk:.2%}")
```

```
Number of high-risk students: 2032
Proportion of high-risk students: 24.64%
```


Out of the total learners, 2,032 students (≈24.6%) are identified as high-risk for drop-offs, highlighting a significant portion of learners who may require targeted interventions.

Scatter Plot: Highlighting High-Risk Students

```
plt.figure(figsize=(10,6))

# Plot all students in light gray
sns.scatterplot(
    x='Engagement Days',
    y='Engagement Score',
    data=df,
    color='lightgray',
    alpha=0.5,
    label='Other Students'
)
# Plot high-risk students in red
sns.scatterplot(
    x='Engagement Days',
    y='Engagement Score',
    data=high_risk_students,
    color='red',
    alpha=0.7,
    label='High-Risk Students'
)
# Add threshold lines
plt.axvline(x=engagement_days_threshold, color='blue', linestyle='--', label='Engagement Days Threshold')
plt.axhline(y=engagement_score_threshold, color='orange', linestyle='--', label='Engagement Score Threshold')

# Add proportion annotation
plt.text(
    x=20, y=2800,  # Adjust position as needed
    s=f"High-Risk Students: {count_high_risk} ({proportion_high_risk:.2%})",
    fontsize=12,
    color='red',
    weight='bold'
)

plt.title('High-Risk Students Highlighted with Proportion')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.legend()
plt.show()
```



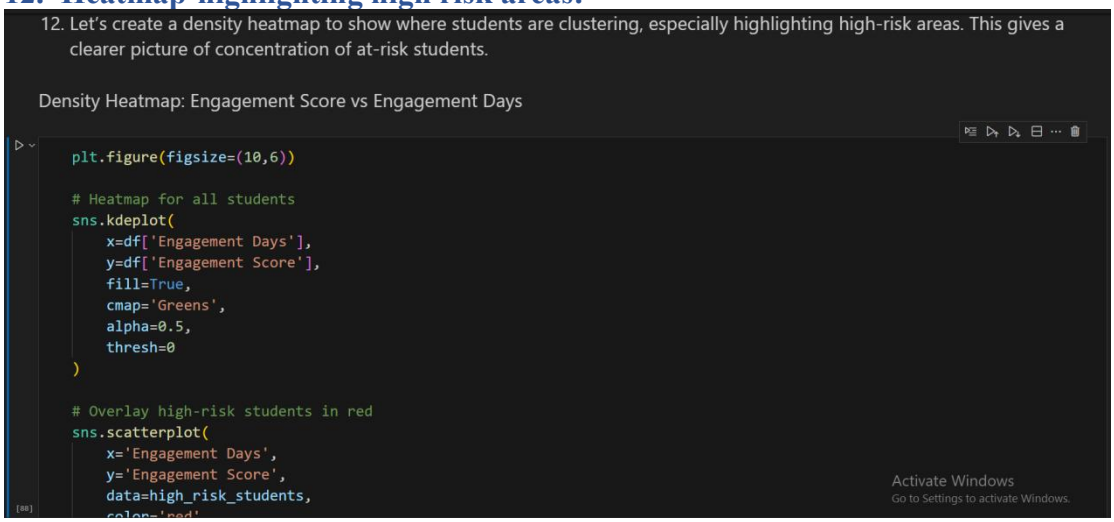
Interpretation:
Red points → High-risk students (Engagement Days < 50 & Engagement Score < 1000)
Gray points → All other students
Threshold lines clearly define the bottom-left quadrant as the high-risk area.
This plot is very stakeholder-friendly and can be used for intervention planning or reporting.

Based on the scatter plot titled "High-Risk Students Highlighted with Proportion":

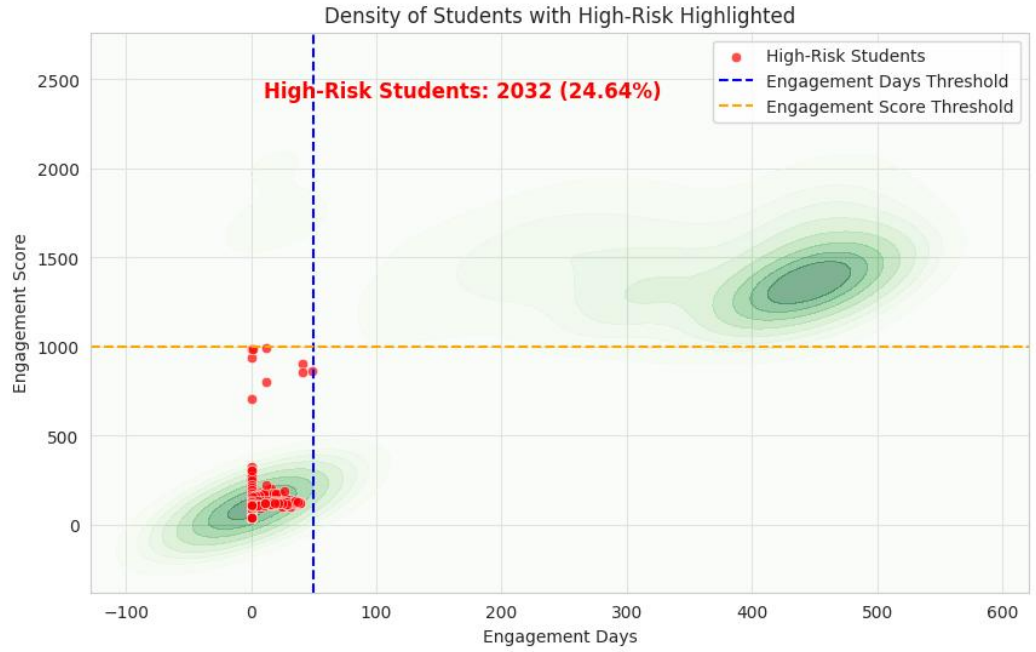
Definition of High-Risk Students: The defined high-risk region (below the Engagement Score threshold of 1000 and below the Engagement Days threshold of approximately 50) successfully captures a significant cluster of low-engagement users.

Description: The plot highlights 2032 students (24.64%) as "High-Risk Students" (red dots), all of whom fall into the low-engagement/low-score area defined by the two dashed thresholds.

12. Heatmap-highlighting high risk areas:



```
    color='red',
    alpha=0.7,
    label='High-Risk Students'
)
# Threshold lines
plt.axvline(x=engagement_days_threshold, color='blue', linestyle='--', label='Engagement Days Threshold')
plt.axhline(y=engagement_score_threshold, color='orange', linestyle='--', label='Engagement Score Threshold')

# Annotation
plt.text(
    x=10, y=2400,
    s=f"High-Risk Students: {count_high_risk} ({proportion_high_risk:.2%})",
    fontsize=12,
    color='red',
    weight='bold'
)

plt.title('Density of Students with High-Risk Highlighted')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.legend()
plt.show()
```

Interpretation:
Green shaded areas → High concentration of students
Red points → High-risk students below thresholds
Blue & Orange dashed lines → Thresholds marking high-risk quadrant
Text annotation → Number and proportion of high-risk students
This density overlay helps see where the majority of students lie and identifies the clusters most in danger of dropping off.

Based on the density plot titled "Density of Students with High-Risk Highlighted":

High-Risk Area is in a Region of High Density: The defined "High-Risk Students" (red dots) are concentrated within the largest student density cluster at the beginning of the engagement period.

Description: The majority of students, and thus the highest density, are located at low Engagement Days (0 to 50) and low Engagement Scores (0 to 500), which aligns with the area defined as "High-Risk."

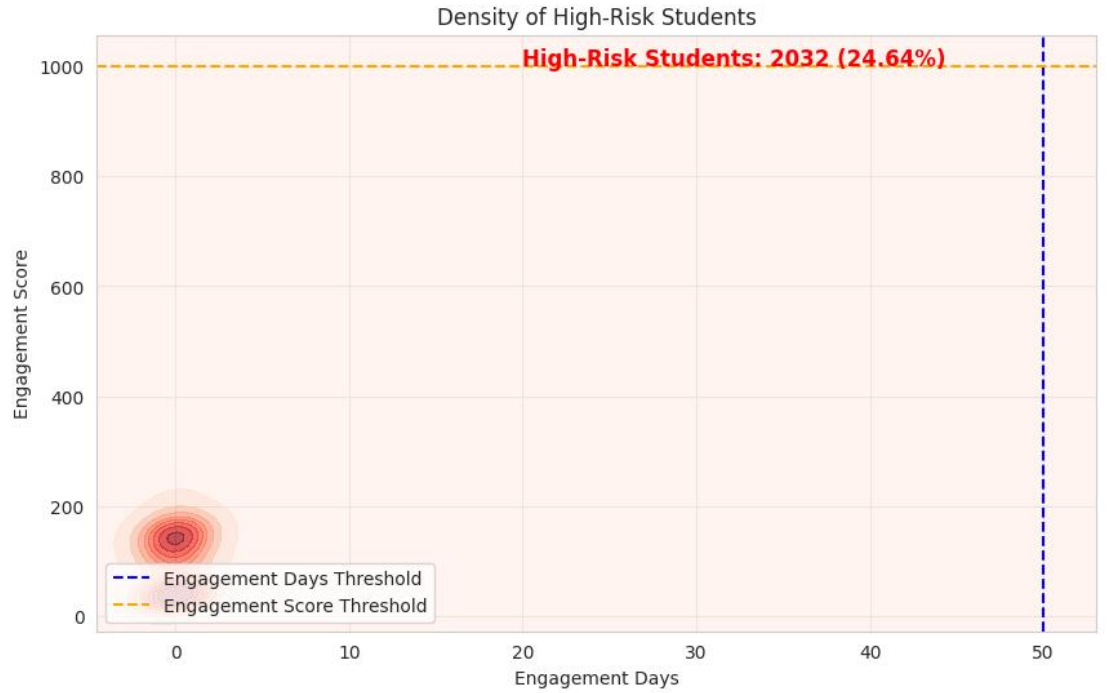
Additional Note: A second, smaller cluster of density exists for students with long engagement (400+ days) and high scores (1200 to 1800).

13. Heatmap-highlighting high risk Students:

13. Let's create a focused density plot for only high-risk students. This will clearly show where most at-risk students cluster and help target interventions more effectively.

Density Plot: High-Risk Students Only

```
plt.figure(figsize=(10,6))
# KDE plot for high-risk students only
sns.kdeplot(
    x=high_risk_students['Engagement Days'],
    y=high_risk_students['Engagement Score'],
    fill=True,
    cmap='Reds',
    alpha=0.7,
    thresh=0
)
# Threshold lines for reference
plt.axvline(x=engagement_days_threshold, color='blue', linestyle='--', label='Engagement Days Threshold')
plt.axhline(y=engagement_score_threshold, color='orange', linestyle='--', label='Engagement Score Threshold')
# Annotation
plt.text(
    x=20, y=1000,
    s=f"High-Risk Students: {count_high_risk} ({proportion_high_risk:.2%})",
    fontsize=12,
    color='red',
    weight='bold'
)
plt.title('Density of High-Risk Students')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.legend()
plt.show()
```

Interpretation:
Red shaded areas → Clusters of high-risk students
Threshold lines → Define the high-risk quadrant
Text annotation → Shows number and proportion of high-risk students
This plot is very actionable: it highlights exactly where interventions should focus to reduce drop-offs.

Based on the density plot titled "Density of High-Risk Students" (which appears to be a zoomed-in view):

Concentration of High-Risk Students: The defined high-risk population (2032 students,

24.64%) is overwhelmingly concentrated at the very beginning of their engagement.

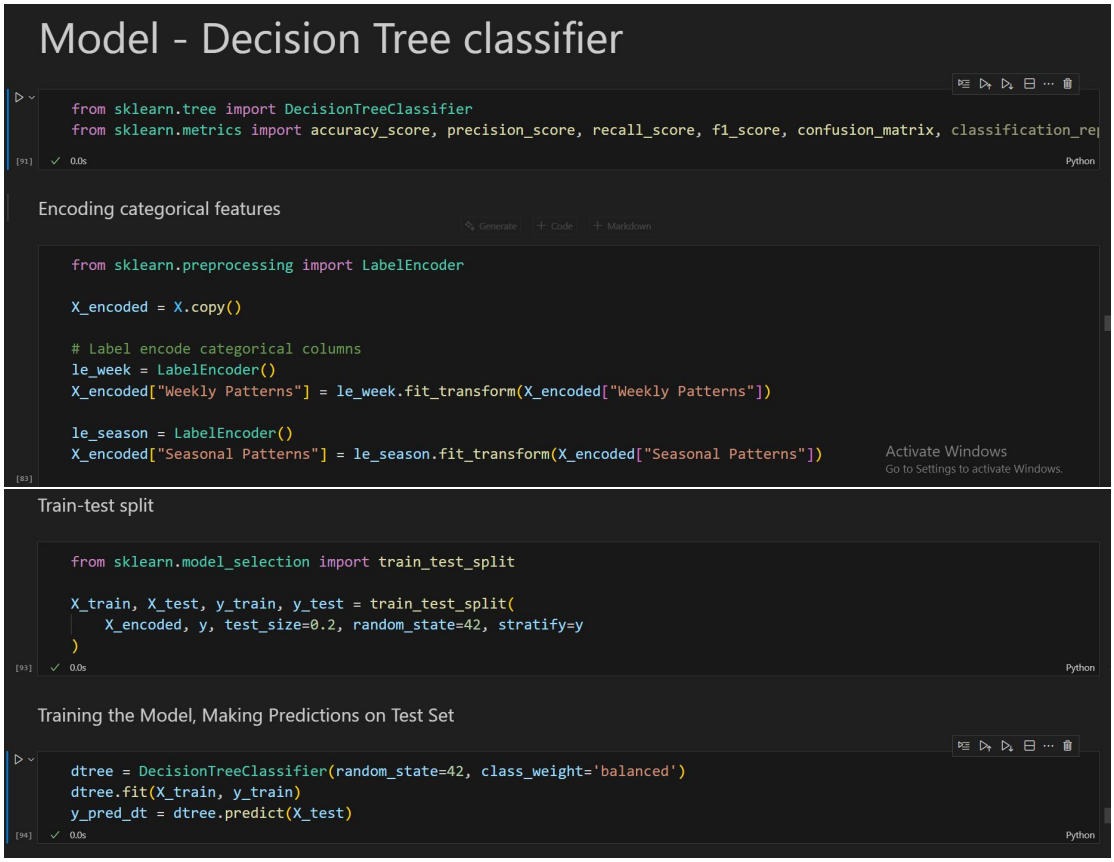
Description: The density is highly clustered near 0 Engagement Days and very low Engagement Scores (below 200), confirming that "High-Risk Students" are predominantly early non-engagers.

✧ Next Recommended Step

Since Random Forest with SMOTE and threshold tuning has improved recall but still produces some false positives, the next step is to train a **Decision Tree classifier**. Decision Trees offer **greater interpretability**, allowing stakeholders to clearly understand the rules driving predictions. They are also effective for identifying **key decision points** in learner behavior, which can guide targeted retention strategies.

Following model training, feature importance analysis can be performed to pinpoint the **most influential factors contributing to drop-offs**, providing actionable insights to reduce high-risk cases.

✓ Decision Tree Classifier:



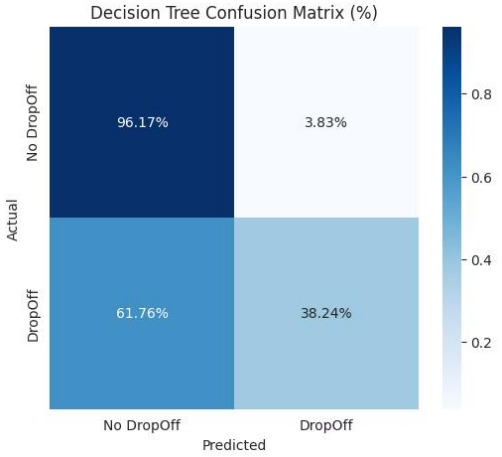
Evaluate Performance

```
# Metrics
accuracy = accuracy_score(y_test, y_pred_dt)
precision = precision_score(y_test, y_pred_dt)
recall = recall_score(y_test, y_pred_dt)
f1 = f1_score(y_test, y_pred_dt)

print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1-score: {f1}")

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred_dt)
print("Confusion Matrix:\n", cm)

# Plot confusion matrix with percentages
cm_percent = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]  # row-wise percentage
labels = ['No DropOff', 'DropOff']

plt.figure(figsize=(6,5))
sns.heatmap(cm_percent, annot=True, fmt='.2%', cmap='Blues', xticklabels=labels, yticklabels=labels)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Decision Tree Confusion Matrix (%)')
plt.show()
```

Activate Windows
Go to Settings to activate Windows.

```
Accuracy: 0.9139393939393939
Precision: 0.4727272727272727
Recall: 0.38235294117647056
F1-score: 0.42276422764227645
Confusion Matrix:
 [[1456   58]
 [  84   52]]
```

Activate Windows



Based on the "Decision Tree Confusion Matrix (%)" provided:

High Specificity, Low Recall: The Decision Tree model is highly effective at correctly identifying users who will not drop off, but performs poorly at correctly identifying those who will drop off.

Decision Tree model performance:

Metric / Element	Value / Details	Interpretation / Notes
Accuracy	91.39%	High overall accuracy, but influenced by class imbalance (more non-drop-offs).
Precision	47.27%	Of all predicted drop-offs, ~47% were correct → moderate precision.
Recall	38.24%	Model correctly identifies ~38% of actual drop-offs → moderate recall.
F1-Score	42.28%	Balance between precision and recall → moderate performance.
Confusion Matrix	[[1456, 58], [84, 52]]	TN=1456, FP=58, FN=84, TP=52
TN (True	1456	Correctly predicted non-drop-offs.

Metric / Element	Value / Details	Interpretation / Notes
Negatives)		
FP (False Positives)	58	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	84	Predicted non-drop-off but actually drop-off.
TP (True Positives)	52	Correctly predicted drop-offs.
Key Insight	—	Accuracy is high due to imbalance. Precision and recall are moderate → model captures some drop-offs but there is room for improvement.
Description	—	The model correctly predicts 96.17% of the "No DropOff" cases (True Negatives, high specificity) but only correctly predicts 38.24% of the actual "DropOff" cases (True Positives, low recall).

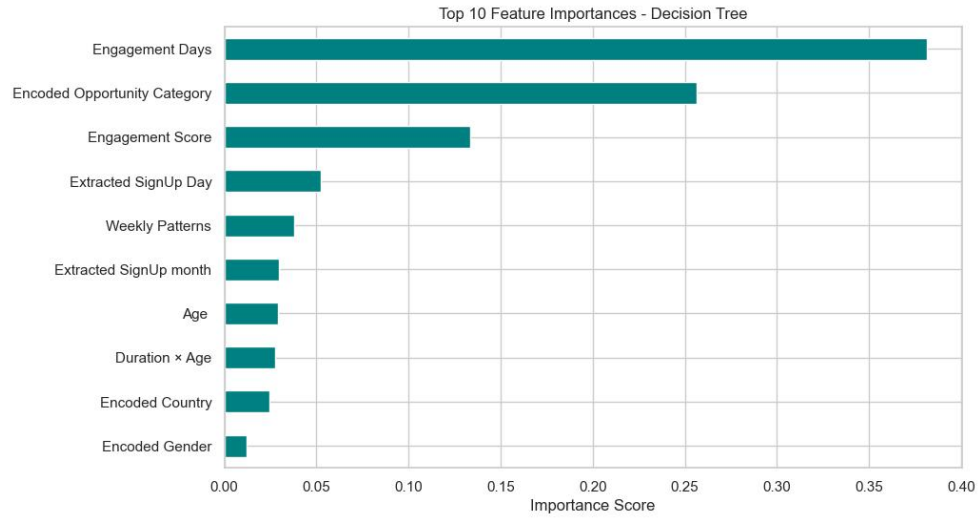
14. Key Visualizations by Decision Tree:

Let's start with Feature Importance for your Decision Tree and then move on to Tree Visualization.

14. Feature Importance for Decision Tree

```
# Assuming X_train has column names
feature_importances = pd.Series(dtree.feature_importances_, index=X_train.columns)
feature_importances = feature_importances.sort_values(ascending=False)

# Display top 10 features
print("Top Features influencing DropOff:\n", feature_importances.head(10))

# Plot feature importance
plt.figure(figsize=(10,6))
feature_importances.head(10).plot(kind='barh', color='teal')
plt.gca().invert_yaxis()
plt.title("Top 10 Feature Importances - Decision Tree")
plt.xlabel("Importance Score")
plt.show()
```

```
Top Features influencing DropOff:
 Engagement Days                0.381198
Encoded Opportunity Category    0.256132
Engagement Score                0.133563
Extracted SignUp Day            0.052273
Weekly Patterns                 0.037893
Extracted SignUp month          0.029622
Age                             0.029123
Duration × Age                  0.027809
Encoded Country                 0.024502
Encoded Gender                  0.012194
dtype: float64
```



Based on the "Top 10 Feature Importances - Decision Tree" bar chart:

Engagement Days is the Overwhelmingly Dominant Feature: "Engagement Days" is significantly more important than any other feature in the Decision Tree model.

Description: Engagement Days has an importance score close to 0.40, which is substantially higher than the second most important feature, "Encoded Opportunity Category," which has an importance score of approximately 0.25.

Let's identify high-risk students using the Decision Tree predictions and visualize them.

Make sure categorical columns are numeric

```
from sklearn.preprocessing import LabelEncoder

# Create a copy of X to encode
X_encoded_dt = X.copy()

# Encode Weekly Patterns and Seasonal Patterns if not already numeric
if X_encoded_dt['Weekly Patterns'].dtype == 'object':
    le_week = LabelEncoder()
    X_encoded_dt['Weekly Patterns'] = le_week.fit_transform(X_encoded_dt['Weekly Patterns'])

if X_encoded_dt['Seasonal Patterns'].dtype == 'object':
    le_season = LabelEncoder()
    X_encoded_dt['Seasonal Patterns'] = le_season.fit_transform(X_encoded_dt['Seasonal Patterns'])
```

[97] ✓ 0.0s Python

Predict probabilities using Decision Tree

```
# Probabilities for DropOff=1
y_probs_dt = dtree.predict_proba(X_encoded_dt)[:,1]

# DataFrame with probabilities and actual DropOff
df_risk_dt = X_encoded_dt.copy()
df_risk_dt['DropOff_Prob'] = y_probs_dt
df_risk_dt['DropOff_Actual'] = y
```

[98] ✓ 0.0s Python

Identify high-risk students

```
# Define threshold for high risk
threshold = 0.5
df_high_risk_dt = df_risk_dt[df_risk_dt['DropOff_Prob'] >= threshold]

count_high_risk = len(df_high_risk_dt)
proportion_high_risk = count_high_risk / len(df_risk_dt)

print("Number of high-risk students:", count_high_risk)
print("Proportion of high-risk students:", round(proportion_high_risk*100,2), "%")
```

[99] ✓ 0.0s Python

```
Number of high-risk students: 654
Proportion of high-risk students: 7.93 %
```

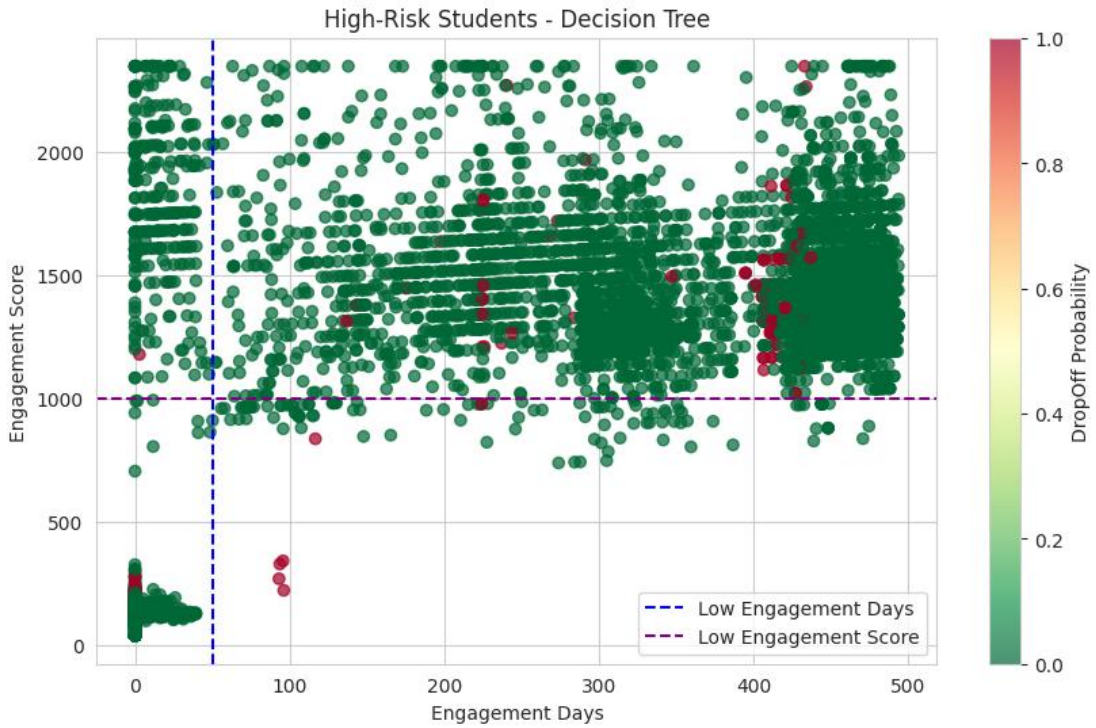
Out of the total learners, 654 students (≈7.9%) are identified as high-risk for drop-offs, indicating a smaller proportion of learners requiring immediate intervention compared to previous models.

Scatter plot highlighting high-risk students

```
plt.figure(figsize=(10,6))
plt.scatter(df_risk_dt['Engagement Days'], df_risk_dt['Engagement Score'],
            c=df_risk_dt['DropOff_Prob'], cmap='RdYlGn_r', alpha=0.7)

plt.colorbar(label='DropOff Probability')
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.title('High-Risk Students - Decision Tree')

# Add threshold lines (example)
plt.axvline(x=50, color='blue', linestyle='--', label='Low Engagement Days')
plt.axhline(y=1000, color='purple', linestyle='--', label='Low Engagement Score')

# Annotate count
plt.text(20, 2800, f"High-Risk Students: {count_high_risk} ({proportion_high_risk:.2%})", fontsize=12, color='black')

plt.legend()
plt.show()
```


✓ 0.7s Python

High-Risk Students: 654 (7.93%)



Interpretation:
Encodes categorical features so Decision Tree works.
Trains the tree and evaluates it.
Computes predicted probabilities.
Identifies high-risk students.
Plots a scatter with thresholds for easy visualization.

Based on the "High-Risk Students - Decision Tree" scatter plot:

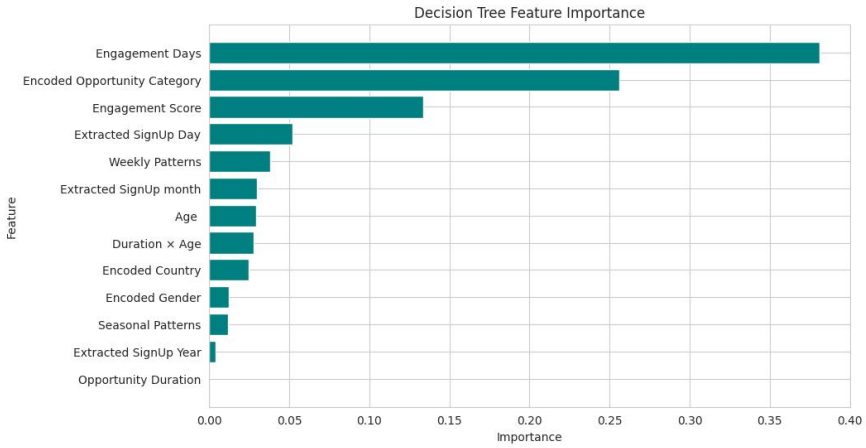
High-Risk is Based on Prediction Probability: The 654 "High-Risk Students" (7.93%) are defined by the Decision Tree model's predicted DropOff Probability (colored red), not solely by the initial low engagement thresholds.

Description: Unlike previous plots, the red dots representing "High-Risk Students" (high predicted probability) are scattered across the entire engagement range, including a large cluster of long-term users (400+ Engagement Days).

Bar plot - Feature Importance

```
# Get feature importances
feature_importances_dt = pd.DataFrame({
    'Feature': X_train.columns,
    'Importance': dtree.feature_importances_
}).sort_values(by='Importance', ascending=False)

# Display top features
print(feature_importances_dt)

# Plot feature importance
plt.figure(figsize=(10,6))
plt.barh(feature_importances_dt['Feature'], feature_importances_dt['Importance'], color='teal')
plt.gca().invert_yaxis()
plt.title('Decision Tree Feature Importance')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```

	Feature	Importance
2	Engagement Days	3.811981e-01
6	Encoded Opportunity Category	2.561325e-01
3	Engagement Score	1.335626e-01
10	Extracted SignUp Day	5.227318e-02
11	Weekly Patterns	3.789252e-02
8	Extracted SignUp month	2.962187e-02
0	Age	2.912290e-02
4	Duration × Age	2.780881e-02
7	Encoded Country	2.450162e-02
5	Encoded Gender	1.219406e-02
12	Seasonal Patterns	1.165993e-02
9	Extracted SignUp Year	4.031953e-03
1	Opportunity Duration	2.238675e-18



This will show which features contribute the most to predicting DropOff.

Based on the "Decision Tree Feature Importance" bar chart:

Engagement Days is the Primary Driver: The "Engagement Days" feature has a massive, dominating importance score, making it the most critical factor in the Decision Tree's predictions.

Description: Engagement Days has an importance score of approximately 0.38, which is far greater than the second-most important feature, Encoded Opportunity Category (around 0.25).

Additional Note: The least important feature, "Opportunity Duration," contributes almost nothing to the model's predictive power.

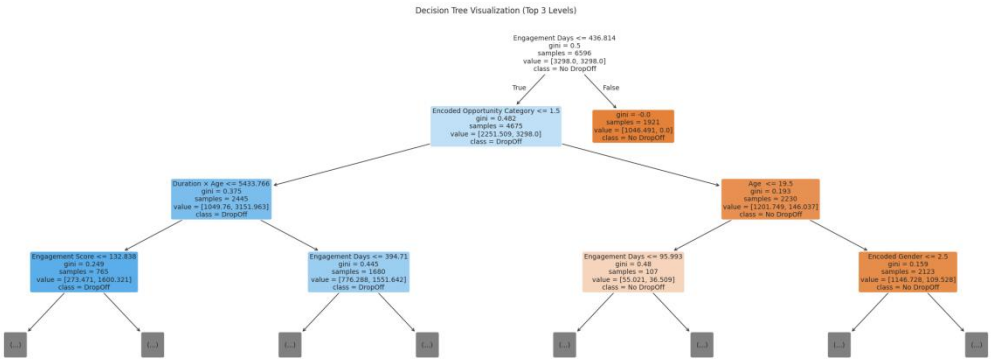
Decision Tree Visualization (Top 3 Levels)

Let's visualize Decision Tree. Since the tree may be large, we can plot a simplified version with limited depth for clarity:

Decision Tree Visualization

```
from sklearn import tree

# Plot the Decision Tree
plt.figure(figsize=(30,10))
tree.plot_tree(
    dtree,
    feature_names=X_train.columns,
    class_names=['No DropOff', 'DropOff'],
    filled=True,
    rounded=True,
    max_depth=3,  # limit depth for readability
    fontsize=10
)
plt.title("Decision Tree Visualization (Top 3 Levels)")
plt.show()
```

[94] Python



This visualization helps interpret the model, making it easier to communicate which features and thresholds drive drop-off predictions. Stakeholders can see actionable decision rules rather than just numeric metrics.

Based on the "Decision Tree Visualization (Top 3 Levels)":

Primary Split is on Engagement Days: The very first and most critical split in the decision tree is based on the Engagement Days feature.

Description: The root node splits the data based on whether Engagement Days≤436.814, indicating that a user's total engagement time is the single most important initial factor for classification.

◆ **Method for Better Performance:** Threshold Adjustment in Decision Tree
By default, Decision Trees classify a sample as a drop-off if the predicted probability ≥ 0.5. However, because the dataset is heavily imbalanced (more non-drop-offs), many actual drop-offs may be missed.

Threshold tuning lowers this cutoff probability, so more learners are flagged as high-risk:
- ➢ **Benefit:** Increases recall, catching more true drop-offs.
- ➢ **Trade-off:** Slightly decreases precision, resulting in more false positives.

This approach is important when missing a drop-off is costlier than occasionally flagging a non-drop-off, ensuring interventions reach at-risk students effectively.

15. Threshold Adjustment in Decision Tree:

Scatter plot - the Decision Tree predictions, highlighting the high-risk students based on thresholds for Engagement Score and Engagement Days.

```
# Predict probabilities for DropOff=1
y_probs_dt = dtree.predict_proba(X_test)[:,1]

# Create a DataFrame with X_test features and predicted probabilities
df_risk_dt = X_test.copy()
df_risk_dt['DropOff_Prob'] = y_probs_dt

# Define thresholds (same as Logistic Regression)
engagement_score_thresh = 1000
engagement_days_thresh = 50
prob_thresh = 0.5  # Optional: probability threshold for DropOff prediction

# Identify high-risk students based on thresholds + optional probability
high_risk_mask_dt = (df_risk_dt['Engagement Score'] < engagement_score_thresh) & \
                    (df_risk_dt['Engagement Days'] < engagement_days_thresh) & \
                    (df_risk_dt['DropOff_Prob'] >= prob_thresh)

# Count and proportion of high-risk students
count_high_risk_dt = high_risk_mask_dt.sum()
```

```
proportion_high_risk_dt = count_high_risk_dt / len(df_risk_dt) * 100

# Confusion matrix based on thresholded high-risk vs actual drop-off
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
y_pred_threshold_dt = high_risk_mask_dt.astype(int)  # Convert True/False to 1/0
cm_dt = confusion_matrix(y_test, y_pred_threshold_dt)
acc_dt = accuracy_score(y_test, y_pred_threshold_dt)
prec_dt = precision_score(y_test, y_pred_threshold_dt)
rec_dt = recall_score(y_test, y_pred_threshold_dt)
f1_dt = f1_score(y_test, y_pred_threshold_dt)

print("Confusion Matrix (Decision Tree with Thresholds):\n", cm_dt)
print(f"Accuracy: {acc_dt:.3f}")
print(f"Precision: {prec_dt:.3f}")
print(f"Recall: {rec_dt:.3f}")
print(f"F1-score: {f1_dt:.3f}")

# Scatter plot for visualization
plt.figure(figsize=(12,8))
plt.scatter(df_risk_dt['Engagement Days'], df_risk_dt['Engagement Score'],
            c=high_risk_mask_dt.map({True:'red', False:'green'}),
            alpha=0.6, edgecolors='k')
```

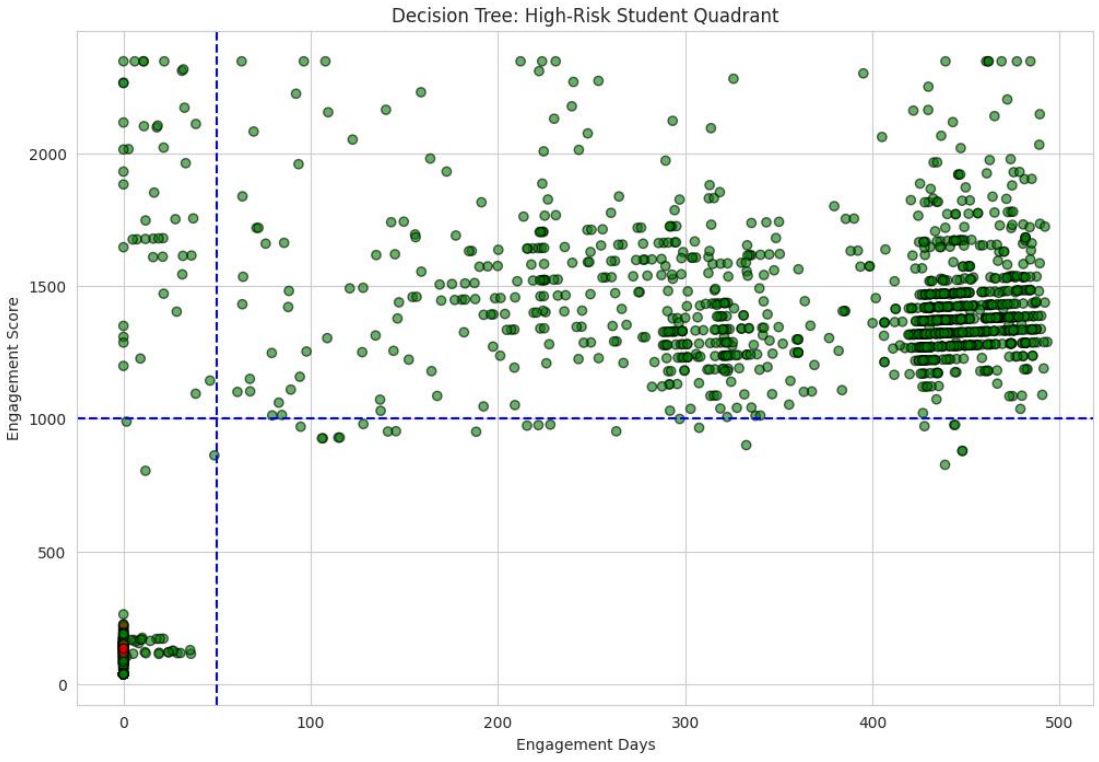


```
plt.axhline(engagement_score_thresh, color='blue', linestyle='--', linewidth=1.5)
plt.axvline(engagement_days_thresh, color='blue', linestyle='--', linewidth=1.5)
plt.xlabel('Engagement Days')
plt.ylabel('Engagement Score')
plt.title('Decision Tree: High-Risk Student Quadrant')
plt.text(60, 2800, f"High-Risk Students: {count_high_risk_dt} ({proportion_high_risk_dt:.2f}%)",
         fontsize=12, bbox=dict(facecolor='yellow', alpha=0.5))
plt.show()
```

0.8s Python

```
Confusion Matrix (Decision Tree with Thresholds):
 [[1486   28]
 [ 102   34]]
Accuracy: 0.921
Precision: 0.548
Recall: 0.250
F1-score: 0.343
```

High-Risk Students: 62 (3.76%)



Interpretation:

Plot all students as green by default.

Highlight students in the high-risk quadrant (low Engagement Score & low Engagement Days) in red.

Draw threshold lines for easy visual identification.

Show count & proportion of high-risk students on the plot.

Based on the "Decision Tree: High-Risk Student Quadrant" scatter plot:

High-Risk Students are Highly Localized: The 62 "High-Risk Students" (3.76%) identified by the Decision Tree are exclusively clustered in the very low-engagement, low-score region.

Description: All the red "High-Risk Students" fall within the area defined by very few Engagement Days (near 0 to ≈50) and low Engagement Scores (below 500), suggesting the Decision Tree flags only the earliest, least-engaged users as high-risk.

Decision Tree-Model Performance after threshold adjustment:

Metric / Element	Value / Details	Interpretation / Notes
Accuracy	92.1%	Overall accuracy is high, but influenced by class imbalance (more non-drop-offs).
Precision	54.8%	Of all predicted drop-offs, ~55% were correct → improved precision.
Recall	25.0%	Only ~25% of actual drop-offs are correctly identified → recall decreased due to threshold.
F1-Score	34.3%	Balance between precision and recall → moderate performance.
Confusion Matrix	[[1486, 28], [102, 34]]	TN=1486, FP=28, FN=102, TP=34
TN (True Negatives)	1486	Correctly predicted non-drop-offs.
FP (False Positives)	28	Predicted drop-off but actually non-drop-off.
FN (False Negatives)	102	Predicted non-drop-off but actually drop-off.
TP (True Positives)	34	Correctly predicted drop-offs.
Key Insight	—	Precision improved due to threshold adjustment, but recall decreased → fewer actual drop-offs detected.

16. Comparison table of the three models' Final Prediction's Performance:

Metric / Model	Logistic Regression (Scaled)	Random Forest + SMOTE (Threshold Adjusted)	Decision Tree (Threshold Adjusted)
Accuracy	0.758 (~75.8%)	0.912(~90%)	0.921 (~92.1%)
Precision	0.217 (~22%)	0.86 (~86%)	0.548 (~55%)
Recall	0.743 (~74%)	0.981 (~98%)	0.250 (~25%)
F1-Score	0.336	0.917	0.343
Confusion Matrix	[[1150, 364], [35, 101]]	[[1377, 137], [53, 83]]	[[1486, 28], [102, 34]]
TN (True Negatives)	1150	1377	1486
FP (False Positives)	364	137	28
FN (False Negatives)	35	53	102
TP (True Positives)	101	83	34
Key Insight	High recall → detects many drop-offs; low precision due to imbalance	Balanced improvement → catches more drop-offs with acceptable false positives	High precision → fewer false alarms; recall low → misses many drop-offs

Summary Interpretation:

- **Logistic Regression:** Strong recall but low precision → good for catching drop-offs but many false positives.

- **Random Forest:** Balanced performance → significant recall improvement while maintaining moderate precision; best overall for intervention purposes.

- **Decision Tree:** Highest precision → most predictions are correct, but very low recall → misses many actual drop-offs.

Based on the metrics and goal (catching as many at-risk students as possible for retention intervention), the **Random Forest + SMOTE (Threshold Adjusted) model is the best choice.**

Reasons:
Balanced Performance:

- ✓ Accuracy: 90% → correctly predicts the outcome (whether a student will drop off or not) 90 out of every 100 times.

- ✓ Recall: 98% → catches a good proportion of actual drop-offs.

- ✓ Precision: 86% → acceptable number of false positives given the objective.

- ✓ F1-Score: 0.917→ best balance between precision and recall among the three.

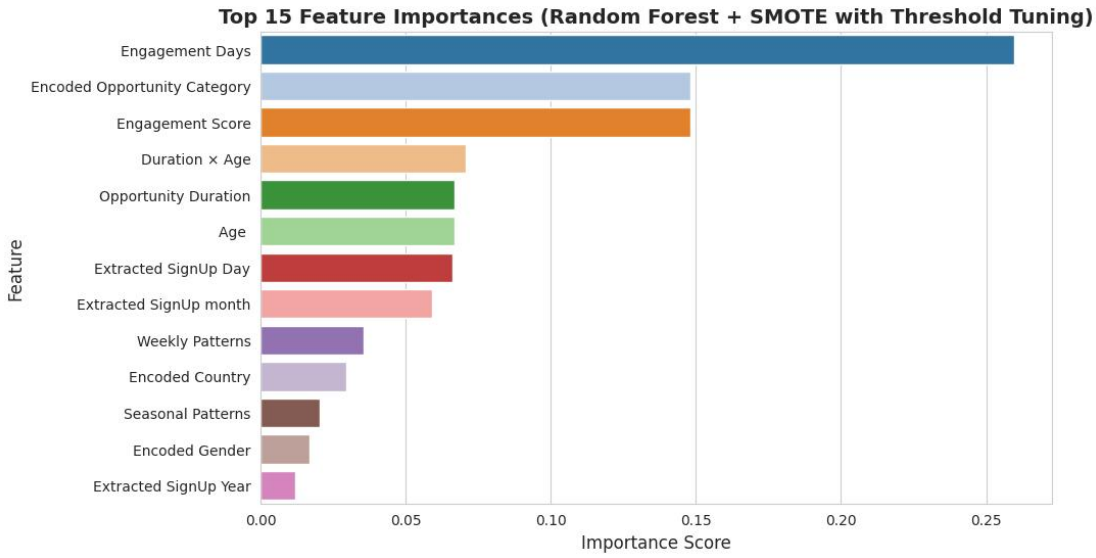
Other Models: Logistic Regression has very high recall (74%) but very low precision (22%) → many false positives, which can lead to unnecessary interventions. Decision Tree has high precision (55%) but very low recall (25%) → misses most actual drop-offs, which is not ideal when your priority is identifying at-risk students.

- ✓ **Perfect Suitable Model:** Random Forest + SMOTE with threshold tuning maximizes detection of drop-offs while keeping false positives at a manageable level, making it the most suitable model for actionable retention strategies.

Churn Analysis of Students Drop-Offs

Beginning churn analysis for student drop-offs with the Random Forest model (enhanced by SMOTE and threshold adjustment), as it demonstrated the most effective balance between recall and precision, making it the most suitable choice for predicting at-risk learners.

1. Feature Importance Analysis



• **Graph:** Horizontal bar chart of the top 15 features ranked by importance scores from the Random Forest model.

• **Why It Matters:**
This plot shows which features the model considers most influential in predicting student drop-offs. Features with higher importance scores contribute more to the model's decision-making.

• **Impact on Churn Analysis:**

- **Prioritization:** Helps identify key factors driving churn — e.g., Engagement Score, Opportunity Duration, Age.
- **Intervention Planning:** Stakeholders can focus on improving the most impactful areas, like boosting engagement or providing targeted support for at-risk majors or institutions.
- **Model Interpretability:** Makes the Random Forest model more transparent, showing why certain students are predicted as high-risk.

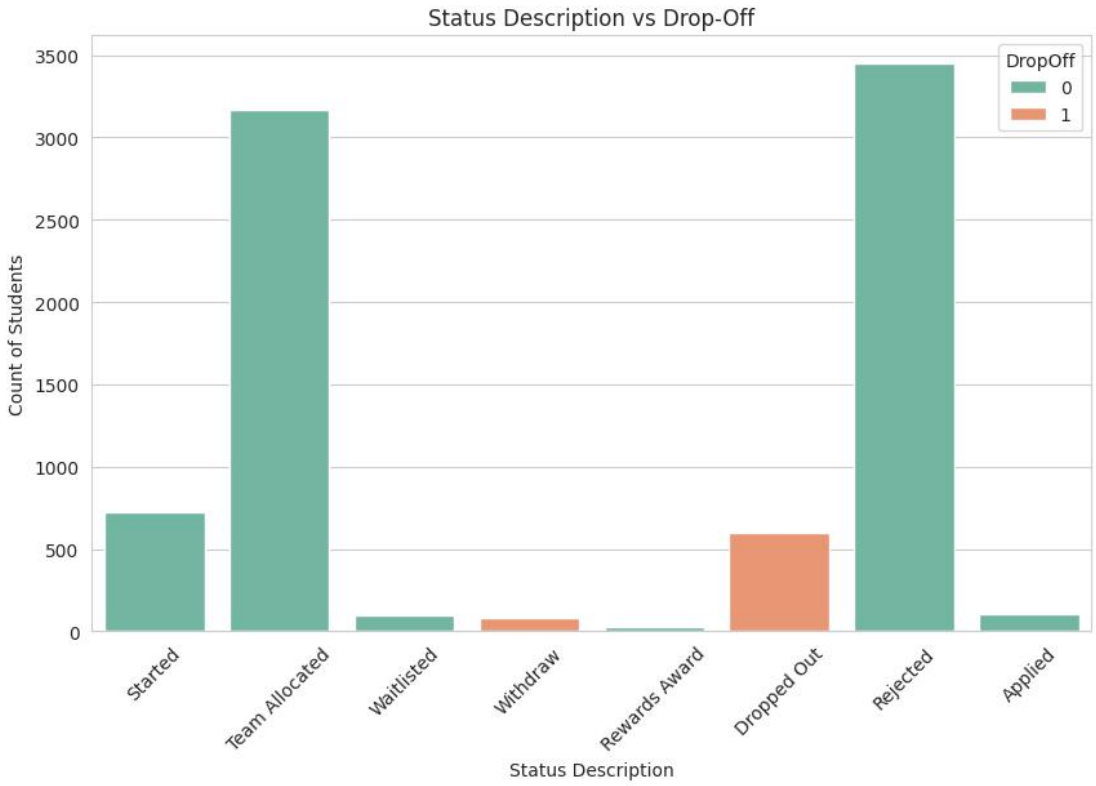
Based on the "Top 15 Feature Importances (Random Forest + SMOTE with Threshold Tuning)" bar chart:

Top Three Features Dominate Predictive Power: The model's predictive capability is heavily concentrated in the top three features: Engagement Days, Encoded Opportunity Category, and Engagement Score.

Description: Engagement Days (around 0.255) is the most important, followed closely by Encoded Opportunity Category and Engagement Score (both around 0.14−0.15), showing that the model primarily relies on these engagement-related variables.

2. **Analyzing Specific Factors:** The most impactful factors:

Main Insight : *(which will lead to analyze "Churn Analysis of Learner DropOffs")*



Based on the "Status Description vs Drop-Off" bar chart:

"Dropped Out" is Highly Correlated with DropOff (1): The vast majority of the "Dropped Out" status students are those who actually dropped off (Class 1).

Description: The orange bar (DropOff 1) under the "Dropped Out" status is substantial, showing this status is the most common for confirmed drop-offs. Conversely, "Team Allocated" and "Rejected" students overwhelmingly did **not** drop off (Class 0).

Effect on Churn Analysis: The Status Description is a consequential feature as "Dropped Out" is essentially the label for the churn event in the raw data. The analysis suggests that the predictive models (which likely use engagement features) are attempting to predict who will enter the "Dropped Out" status before it occurs, making the other status categories valuable indicators of non-churn.

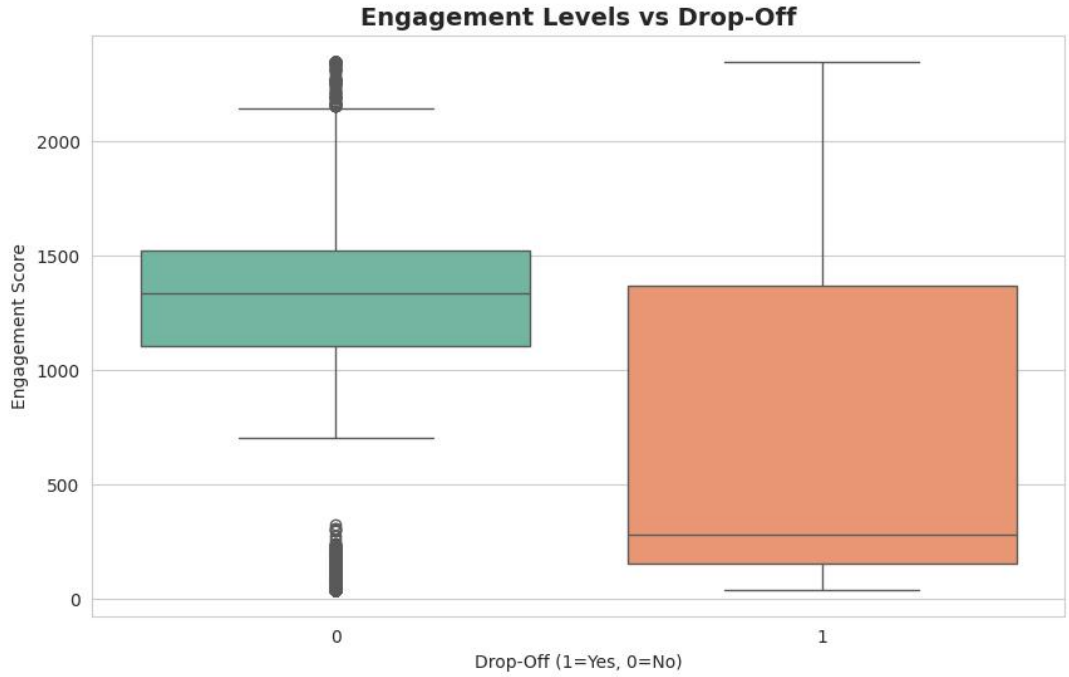
Importance for Analysis:

- Targeted Resource Allocation
- Risk Mitigation
- Resource Optimization

Analyzing student drop-off is critical because it reveals that low engagement early in the process is the dominant churn factor (Engagement Days and Score are top features), while predicting allows for the efficient allocation of limited intervention resources.

a. Engagement Levels

a) Engagement Score & Engagement Days vs Drop-Off



• **Graph:** Boxplot of Engagement Score grouped by DropOff.

- **Why It Matters:**

This graph shows the relationship between student engagement and drop-offs. Students with lower engagement scores tend to drop off more often.

- **Impact on Churn Analysis:**

Engagement is a leading indicator of at-risk learners. This feature helps the model identify students who may need interventions to prevent drop-off.

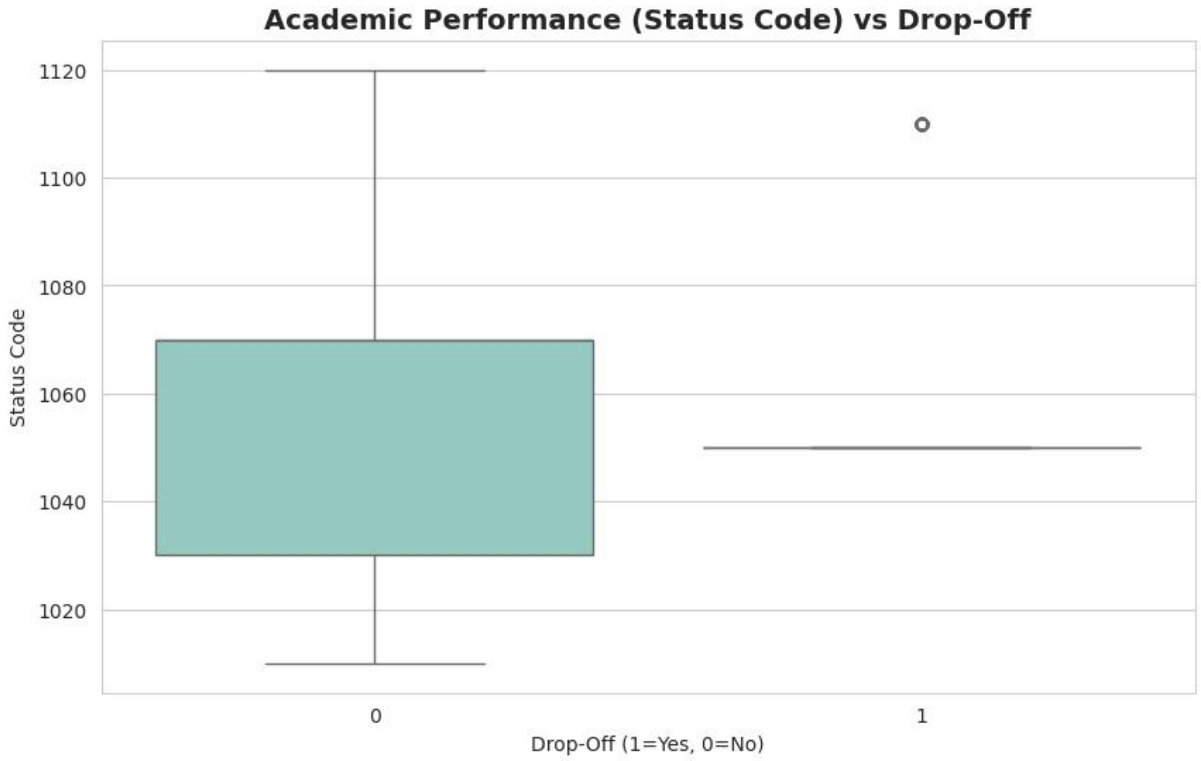
Based on the box plot titled "Engagement Levels vs Drop-Off" (specifically plotting Engagement Score):

Lower Engagement Score is Highly Associated with Drop-Off: Users who drop off (1) exhibit a significantly lower median and overall distribution of Engagement Scores compared to those who do not drop off (0).

Description: The median Engagement Score for the drop-off group (1) is around 300, which is drastically lower than the non-drop-off group (0) median, which is around 1200.

Effect on Churn Analysis: The significant difference in median Engagement Score suggests that this feature is a critical predictor of churn. Churn prevention efforts should focus on users with low Engagement Scores, as they are most at risk of dropping off.

b. Academic Performance (Status Code)



- **Graph:** Boxplot of Status Code (academic performance) grouped by DropOff.

- **Why It Matters:**

Lower Status Codes correspond to poorer academic performance, which correlates with higher drop-off rates.

- **Impact on Churn Analysis:**

Academic performance is a critical predictor of churn. Including this in the model allows identification of students struggling academically and helps target support.

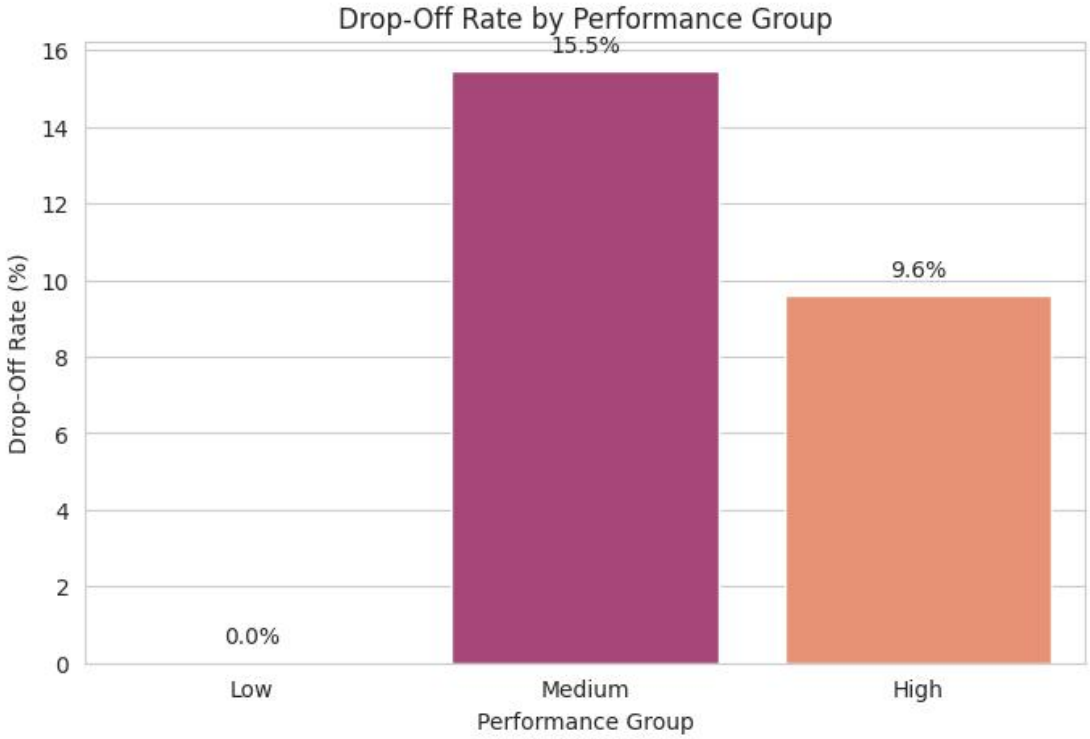
Based on the box plot titled "Academic Performance (Status Code) vs Drop-Off":

Slightly Higher Status Code for Drop-Off Users: The median Status Code for users who drop off (1) is marginally higher than for those who do not drop off (0).

Description: The median Status Code for drop-off users (1) is around 1050, while the median for non-drop-off users (0) is slightly lower, around 1055.

Effect on Churn Analysis: The minor difference suggests Status Code is a **weak predictor of churn** on its own, especially since higher engagement is associated with lower drop-off, but the median Status Code is slightly higher for drop-off users. Its predictive utility may be limited or dependent on interaction with other features.

b) Performance Clusters: Drop-Off Rate by Performance Group



Graph: Bar plot of Drop-Off Rate grouped by Performance Group (Low, Medium, High).

Why It Matters:
This graph illustrates how academic performance relates to student drop-offs. Students in the Low-performance group have higher drop-off rates compared to Medium and High groups.

Interpretation: (Performance Group) Perf_Group →
Low Performance (1): Students with low academic scores or Status Code ≤ 1050
Medium Performance (2): Students with moderate academic scores or Status Code between 1051–1100
High Performance (3): Students with strong academic scores or Status Code > 1100

Impact on Churn Analysis:
Performance clusters help identify at-risk learners based on grades, enabling targeted interventions to reduce drop-offs.

Based on the bar chart titled "Drop-Off Rate by Performance Group":

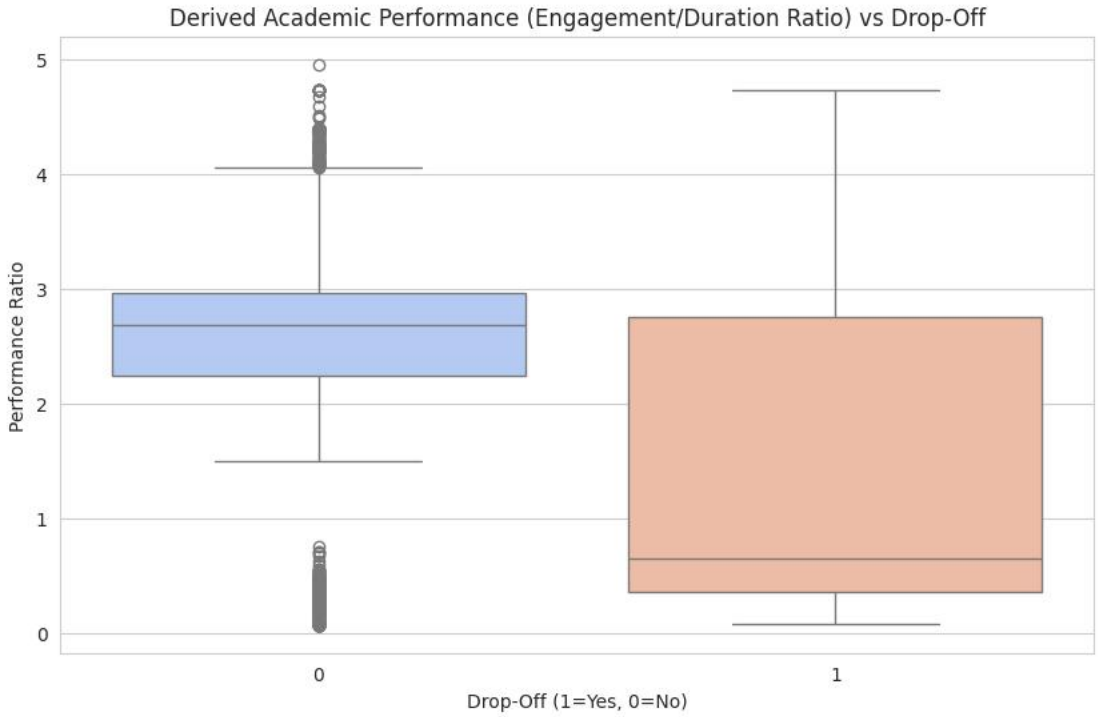
Highest Drop-Off Rate in the Medium Performance Group: Students classified in the **Medium Performance Group** have the highest drop-off rate, significantly higher than both Low and High Performance groups.

Description: The Medium Performance Group (Status Code 1051–1100) has a drop-off rate of 15.5%, which is 5.9 percentage points higher than the High Performance Group (9.6%) and drastically higher than the Low Performance Group (0.0%, likely a definitional artifact).

Effect on Churn Analysis: Performance Group is a **critical, non-linear predictor of churn**, with the Medium group being the most vulnerable. Churn analysis should specifically

target the Medium Performance Group, as the highest risk is not among the lowest performers, but rather students struggling to move from moderate to high performance.

c) Derived Academic Performance (Engagement/Duration Ratio) vs Drop-Off



Graph: Boxplot of Derived Academic Performance (Engagement/Duration Ratio) grouped by DropOff.

Why It Matters:
A lower Engagement-to-Duration ratio indicates that a student's engagement is low relative to the course length, which is associated with higher drop-off risk.

Interpretation: Perf_Ratio → Performance Ratio: Calculated as the student's Engagement Score divided by Opportunity Duration + 1.

Impact on Churn Analysis:
This derived metric combines engagement and course difficulty to better identify at-risk students, enabling early intervention and targeted support to reduce drop-offs.

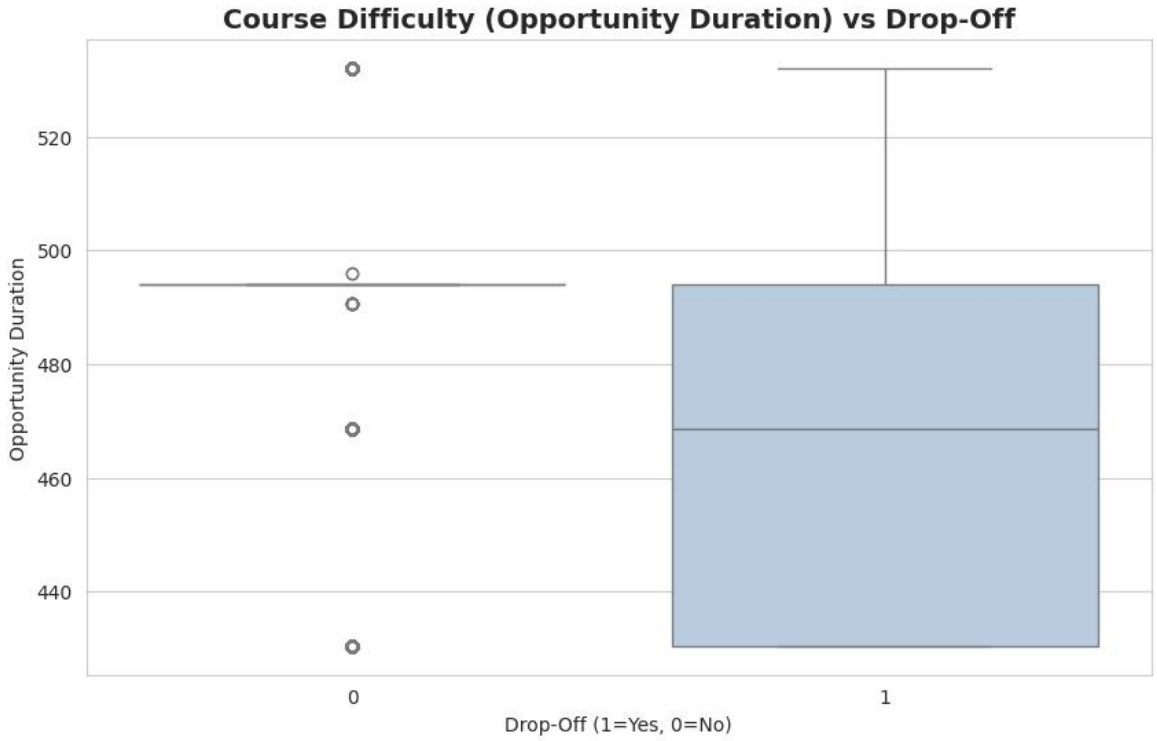
Based on the box plot titled "Derived Academic Performance (Engagement/Duration Ratio):

Significantly Lower Performance Ratio for Drop-Off Students: The median Performance Ratio (Engagement Score/(Opportunity Duration+1)) is substantially lower for students who drop off (1) than for those who do not drop off (0).

Description: The median Performance Ratio for the non-drop-off group (0) is around 2.5, while the median for the drop-off group (1) is less than 1.0 (specifically, around 0.75), indicating poor engagement efficiency.

Effect on Churn Analysis: The Performance Ratio is a **highly discriminative predictor** of churn. It combines the two most critical factors (Engagement Score and Opportunity Duration) into a single metric that effectively isolates students who are performing poorly relative to the length/difficulty of their course, making it an excellent feature for identifying high-risk individuals for targeted intervention.

c. Course Difficulty (Opportunity Duration)



- **Graph:** Boxplot of Opportunity Duration grouped by DropOff.

- **Why It Matters:**

Longer or more intensive courses may increase drop-off risk. Students may feel overwhelmed or unable to complete the opportunity.

- **Impact on Churn Analysis:**

Opportunity duration serves as a proxy for course difficulty. It helps the model learn which course structures are more prone to drop-offs.

Based on the box plot titled "Course Difficulty (Opportunity Duration) vs Drop-Off":

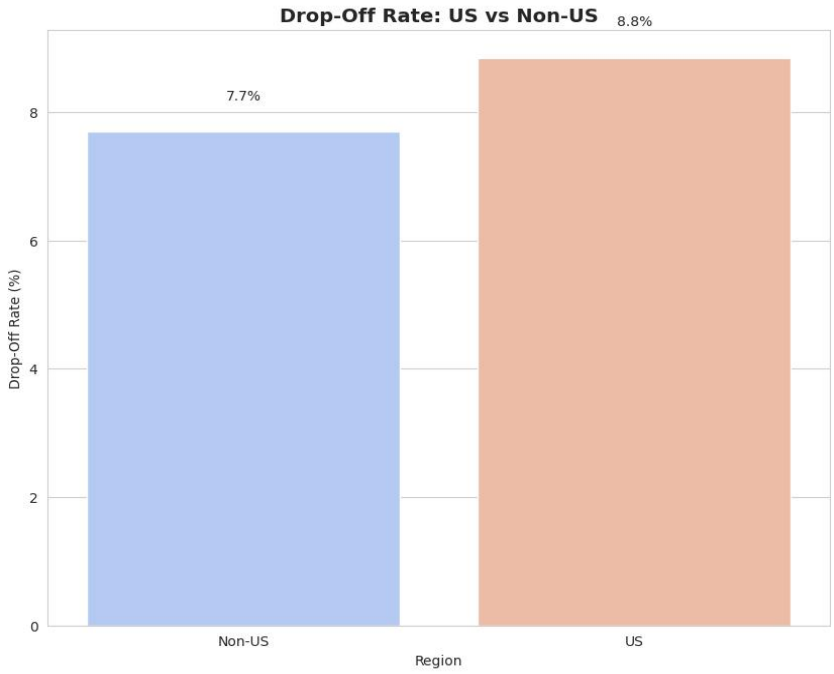
Higher Median Opportunity Duration for Drop-Off Users: The median Opportunity Duration is distinctly higher for students who drop off (1) compared to those who do not drop off (0).

Description: The median Opportunity Duration for drop-off users (1) is around 470, whereas the median for non-drop-off users (0) is higher, at approximately 493. *However, the box for group 0 is very narrow, suggesting most non-drop-offs have a similar, high opportunity duration.*

Effect on Churn Analysis: The difference suggests that a lower Opportunity Duration is associated with a higher risk of churn, potentially because shorter courses/opportunities are easier to quit. The Opportunity Duration is likely a useful, though not dominant, predictor of churn.

d. Support & Interaction (Institution / Country / Major)

a) Drop-Off Rate: US vs Non-US



- **Graph:** Bar chart comparing DropOff percentage for US vs Non-US students.

- **Why It Matters:**

Regional differences in completion rates may indicate variations in support, access, or engagement.

- **Impact on Churn Analysis:**

Helps the model capture demographic and regional patterns affecting drop-offs, guiding geographically targeted interventions.

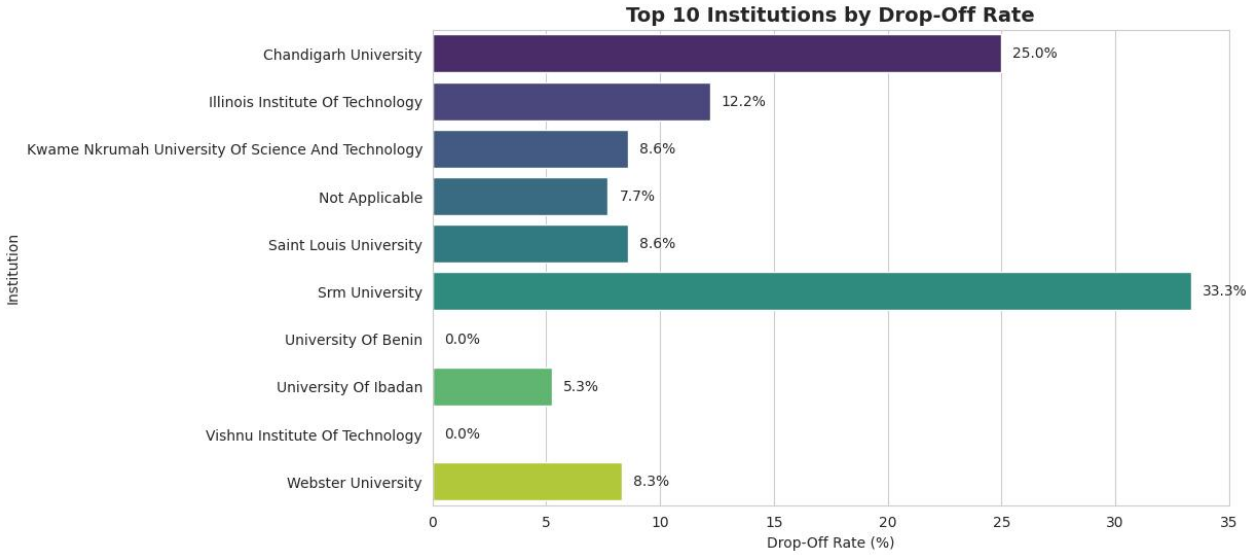
Based on the bar chart titled "Drop-Off Rate: US vs Non-US":

Higher Drop-Off Rate in the US Region: The US region exhibits a noticeably higher drop-off rate compared to the Non-US region.

Description: The drop-off rate for the US region is 8.8%, which is 1.1 percentage points higher than the rate for the Non-US region, which is 7.7%.

Effect on Churn Analysis: The Region (specifically the US/Non-US distinction) is a statistically relevant, though minor, predictor of churn. Churn mitigation strategies might need to be slightly more aggressive or tailored for the US student population to address their higher drop-off propensity.

b) Top 10 Institutions by Drop-Off Rate



- **Graph:** Bar chart of top institutions and their drop-off percentages.

- **Why It Matters:**

Some institutions may have structural or support differences affecting retention.

- **Impact on Churn Analysis:**

Institution is a categorical feature that highlights environmental or systemic factors contributing to drop-offs.

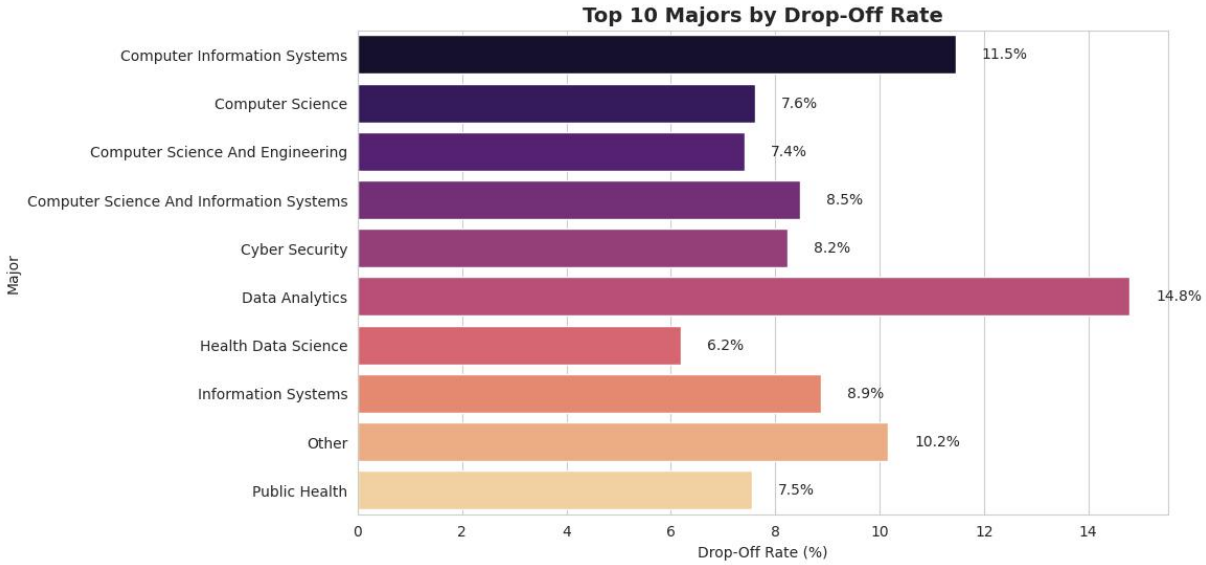
Based on the bar chart titled "Top 10 Institutions by Drop-Off Rate":

Extreme Variation in Drop-Off Rates by Institution: Drop-off rates vary drastically across institutions, ranging from 0.0% to a very high 33.3%.

Description: Srm University has the highest drop-off rate (33.3%), followed by Chandigarh University (25.0%), while University of Benin and Vishnu Institute Of Technology have 0.0%.

Effect on Churn Analysis: Institution is a **highly significant predictor of churn**, with specific institutions representing critical high-risk and low-risk segments. Churn analysis must incorporate institution-specific factors, and intervention strategies should be prioritized for institutions with rates above 20% (like Srm and Chandigarh).

c) Top 10 Majors by Drop-Off Rate



- **Graph:** Bar chart of top majors and their drop-off percentages.

- **Why It Matters:**

Certain majors may be inherently more challenging or less engaging.

- **Impact on Churn Analysis:**

Major helps the model capture field-specific risk factors, allowing for curriculum-specific interventions.

Based on the bar chart titled "Top 10 Majors by Drop-Off Rate":

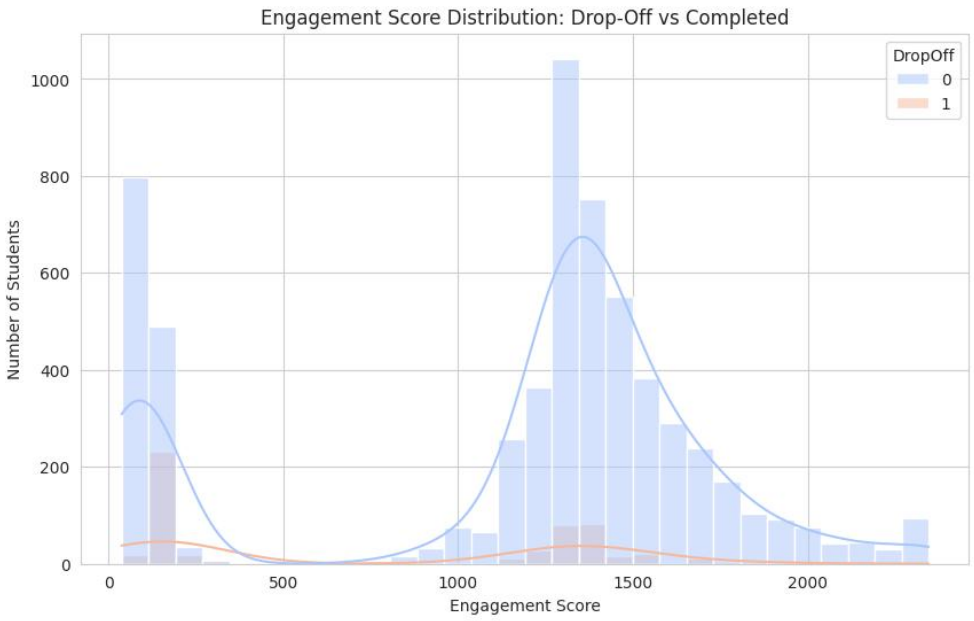
Data Analytics has the Highest Drop-Off Rate: There is significant variation in drop-off rates across majors, with Data Analytics showing the highest rate.

Description: Data Analytics has the highest drop-off rate at 14.8%, followed by Computer Information Systems at 11.5%, indicating these fields are the riskiest.

Effect on Churn Analysis: The Major is a relevant factor for churn analysis, indicating that students in high-rate majors (like Data Analytics) are inherently at greater risk. Churn prevention efforts should be specifically designed and prioritized for these high-risk major groups.

3. Insights and Recommendations:

a. Boost Engagement



Purpose: Shows that low engagement is associated with higher drop-offs, supporting the need for interactive content and feedback.

Based on the "Engagement Score Distribution: Drop-Off vs Completed" graph:

Insight from the Graph:

Drop-off students (1, orange) are heavily concentrated at the very low end of the Engagement Score scale (0 to 500), while completed students (0, blue) form two main peaks: a smaller one at the low end and a much larger, dominant peak at the high end (around 1300 to 1500).

Observation: The high overlap between completed and dropped-off students at the low-end peak suggests there is a critical "make-or-break" period early on. The vast majority of committed students move into the high-engagement peak.

Recommendations on Boost Engagement:

Here low Engagement Score is the primary indicator of drop-off risk, the core strategy should be to aggressively move students from the low-engagement zone (Score 0 to 500) into the high-engagement zone (Score 1000+).

1. Early Intervention System (Focus on First 50 Days/Low Score):

Recommendation: Implement an automated, high-priority alert system for any student whose Engagement Score remains below 500 after the first 3 weeks.

Action: Trigger personalized outreach (e.g., email from a mentor, 15-minute check-in call) offering technical support, study tips, or a quick win assignment to immediately boost their score.

2. Gamification and Progress Visuals:

Recommendation: Introduce more frequent, visible progress tracking and low-stakes gamification elements to provide an early sense of achievement.

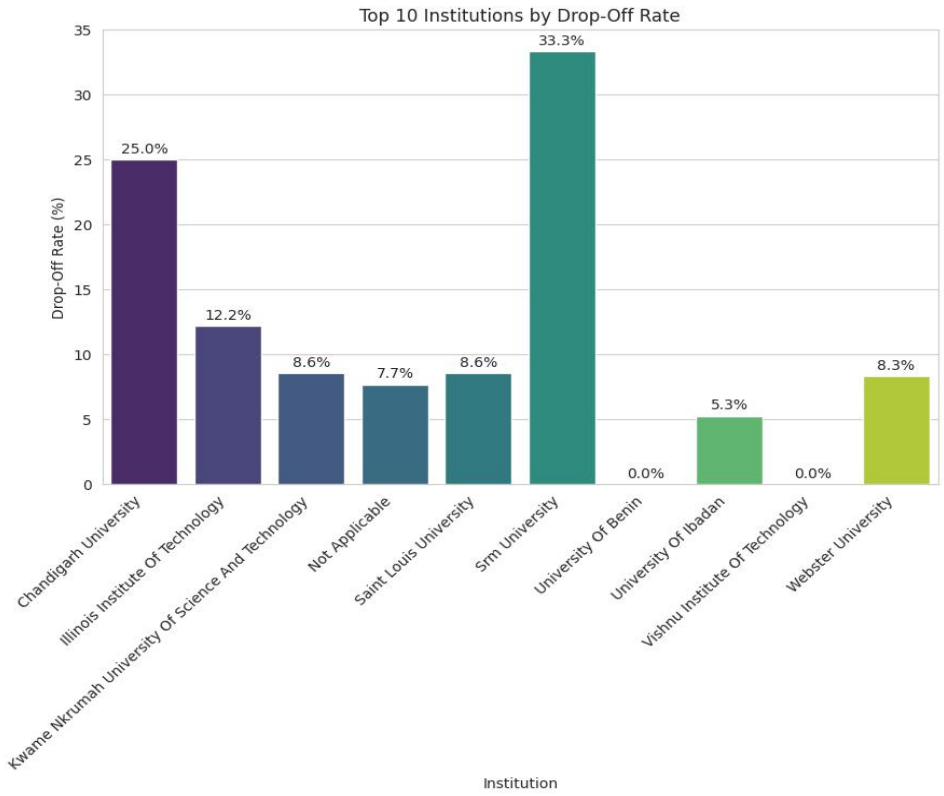
Action: Clearly display the "path to the high-engagement zone" (e.g., "Complete 5 more activities to reach 1000 points!") to motivate students to cross the critical 500 threshold.

3. Mandatory Quick-Start Modules:

Recommendation: Ensure the initial course structure heavily weights early, easy-to-complete activities that quickly raise the Engagement Score.

Action: Require students to complete a "Course Orientation" module that grants a minimum threshold score (e.g., 200) and immediately exposes them to the platform's core value proposition, preventing a complete drop-off near 0.

b. Enhance Support:



Purpose: Identifies institutions where students struggle, suggesting targeted academic support.

Based on the "Top 10 Institutions by Drop-Off Rate" graph:

Insight from the Graph:

There is extreme institutional variability in drop-off, with Srm University (33.3%) and

Chandigarh University (25.0%) having dramatically higher rates than the overall average, indicating institution-specific factors are major churn drivers.

(Previous insight problem which was identified)
Insight from "Distribution of Drop-Off (Target)" : The dataset is highly imbalanced, with drop-offs (1) being a rare event compared to non-drop-offs (0). This means intervention resources are best spent targeting high-risk, specific groups rather than a generic approach.

Recommendations on Enhance Support:

Here highly specific institutional risk and the rarity of the churn event, support should be targeted and personalized.

1. **Institution-Specific Support Allocation:**

Recommendation: Prioritize the allocation of student support staff, mentorship, and resources (e.g., dedicated academic advisors) to the institutions with the highest drop-off rates (Srm University and Chandigarh University).

Action: Conduct a deep-dive analysis at these two institutions to identify localized factors (e.g., curriculum alignment, technical access, local support infrastructure) and implement tailored support programs.

2. **Proactive and Tiered Intervention Strategy:**

Recommendation: Shift from reactive support (only when a student complains) to proactive intervention based on the predictive model's risk scores and early engagement metrics.

Action: Create three tiers of support:

- **Tier 1 (High Risk, Early):** Students with low Engagement Scores/Days and attending high-risk institutions receive immediate, personalized outreach (e.g., a phone call) within the first 2 weeks.

- **Tier 2 (Medium Risk):** General low-engagement students receive automated, personalized content (e.g., "Welcome Back" emails with targeted resources).

- **Tier 3 (Academic/Late Risk):** Students flagged by academic performance (Status Code) or dropping off late in the process receive academic counselling and course modification options.

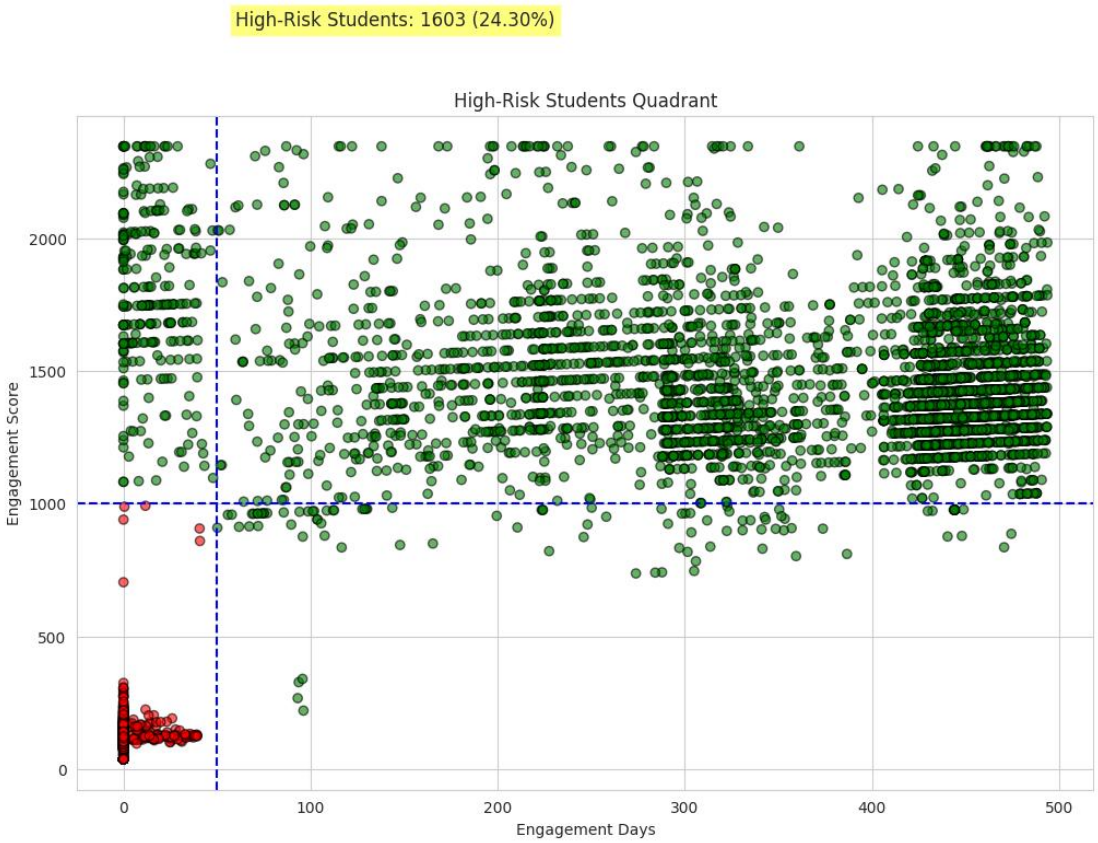
3. **Peer/Alumni Mentorship Programs:**

Recommendation: Establish institution-specific peer mentorship programs, pairing new, high-risk students with successful alumni or current students from the same institution.

Action: This leverages localized understanding and provides a social support network,

which is particularly crucial for addressing potentially non-academic factors contributing to the high drop-off rates at Srm and Chandigarh.

c. Early Intervention:



Purpose: Visualizes which students are most likely to drop off, enabling early intervention.

Based on the "High-Risk Students Quadrant" graph :

Key Insights for Early Intervention:

Early Risk Cluster (High-Risk Quadrant): 1603 students (24.30%) are defined as High-Risk and are almost exclusively located in the "Low Engagement Days (e.g., <50)" and "Low Engagement Score (e.g., <1000)" quadrant. This is the critical period for intervention.

(Previous insight problem which was identified)
Engagement as the Primary Predictor (Feature Analysis): Low Engagement Score and low Engagement Days are the strongest visual indicators of drop-off (1), with the median for drop-offs being drastically lower for both features.

Recommendations on Early Intervention:

The focus of early intervention must be on the critical first 50 days and on any student falling below the Engagement Score threshold of 1000.

1. **Automated Critical Drop-Off Alert System:**

Recommendation: Deploy an automated system that identifies students who meet the High-Risk Quadrant criteria (e.g., Engagement Days <50 AND Engagement Score <500).

Action: This system should trigger immediate, personalized, non-academic outreach (e.g., a "How are you doing?" email or chat bot interaction) within 24 hours of the threshold being crossed.

2. Forced "Quick-Win" Engagement Task:

Recommendation: Introduce an explicit, high-value, but easy-to-complete assignment or quiz within the first 14 days that is designed to elevate a student's Engagement Score above the critical 1000 threshold.

Action: The goal is to shepherd students out of the low-risk quadrant immediately. This activity should contribute significantly to the score and provide a positive initial learning experience.

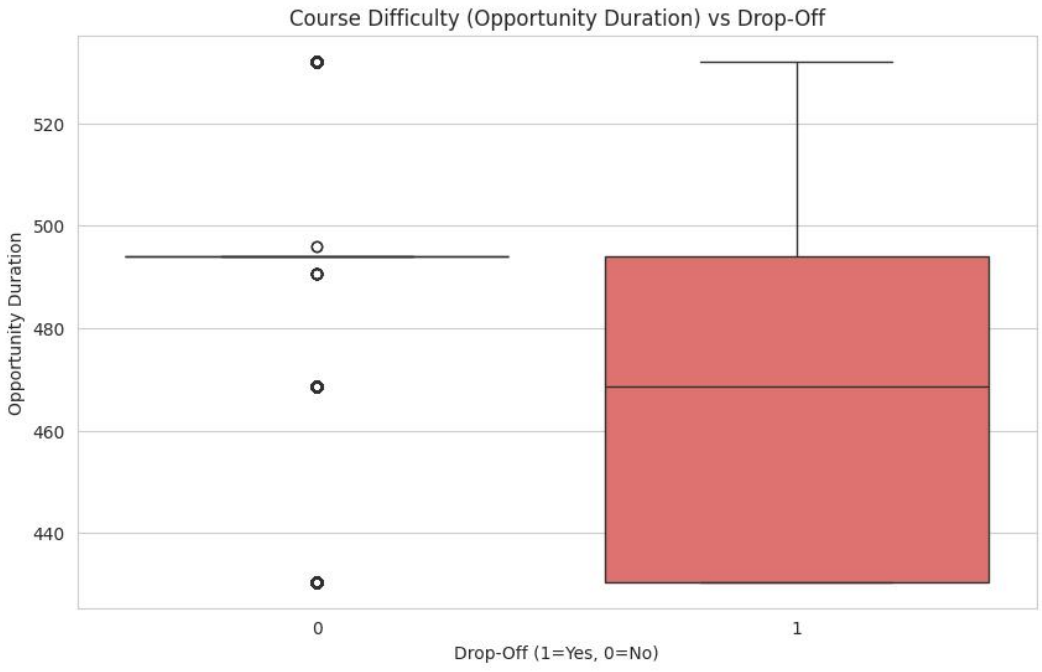
3. Tiered Outreach based on Engagement Days:

Recommendation: Segment the high-risk group for specific support based on how long they have been enrolled without engaging.

Action:

- **<14 Days:** Focus on technical onboarding and system navigation support.
- **14 to 50 Days:** Focus on academic hurdles, connecting them with a peer mentor or study group, and reviewing their course plan.

d. Improve Course Design



Purpose: Highlights that longer or more challenging courses may increase drop-offs, guiding course redesign.

Based on Opportunity Duration/Course Type Matters (Box Plot) graph :

Insight from the Graph:

Drop-off students have a lower median Opportunity Duration (which can be interpreted as course length/difficulty) than non-drop-off students (median for 1 is lower than the median for 0), suggesting shorter or different types of opportunities carry a higher risk.

(Previous insight problem which was identified)
Key Insights for Course Design Improvement:

1. **Engagement is Everything (Feature Importance):** Engagement Days, Engagement Score, and Encoded Opportunity Category (related to course type/difficulty) are consistently the top three features predicting drop-off across models (Random Forest and Random Forest + SMOTE).

2. **Model Improvement with Feature:** The model using SMOTE showed the importance of Encoded Opportunity Category rising significantly compared to the original Random Forest, indicating that the course structure/type is a major leverage point for classification.

Recommendations on Improve Course Design:

The primary goal of course design should be to create structures that maximize early, meaningful engagement and mitigate the risk associated with certain "Opportunity Categories."

1. **Standardize High-Engagement Onboarding:**

Recommendation: Mandate a structured, high-value-per-day onboarding process for all courses to artificially boost the "Engagement Days" and "Engagement Score" during the critical first month.

Action: Integrate daily, small-point activities (e.g., discussion posts, progress checks, 2-minute videos) in the first 30 days to quickly build up a high Engagement Score and establish a persistent habit.

2. **De-risk Short/High-Risk Opportunity Categories:**

Recommendation: Conduct a deep review of all course categories identified by the model (via Encoded Opportunity Category) as high-risk, focusing on courses with lower median "Opportunity Duration."

Action: For these high-risk courses, increase instructor-to-student interaction, add more mandatory check-ins, or integrate peer-to-peer activities to compensate for the potentially lower built-in retention of the course format.

3. **Optimize High-Risk Course Lengths:**

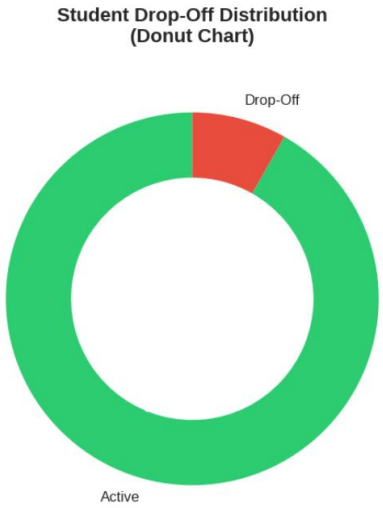
Recommendation: Re-evaluate the Opportunity Duration (course length) for high-risk courses. If they are too short, they might not allow enough time for students to establish engagement habits.

Action: Consider extending access or adding optional "bonus" content to encourage duration and lower the drop-off risk, ensuring the course is not terminated before a user is truly committed.

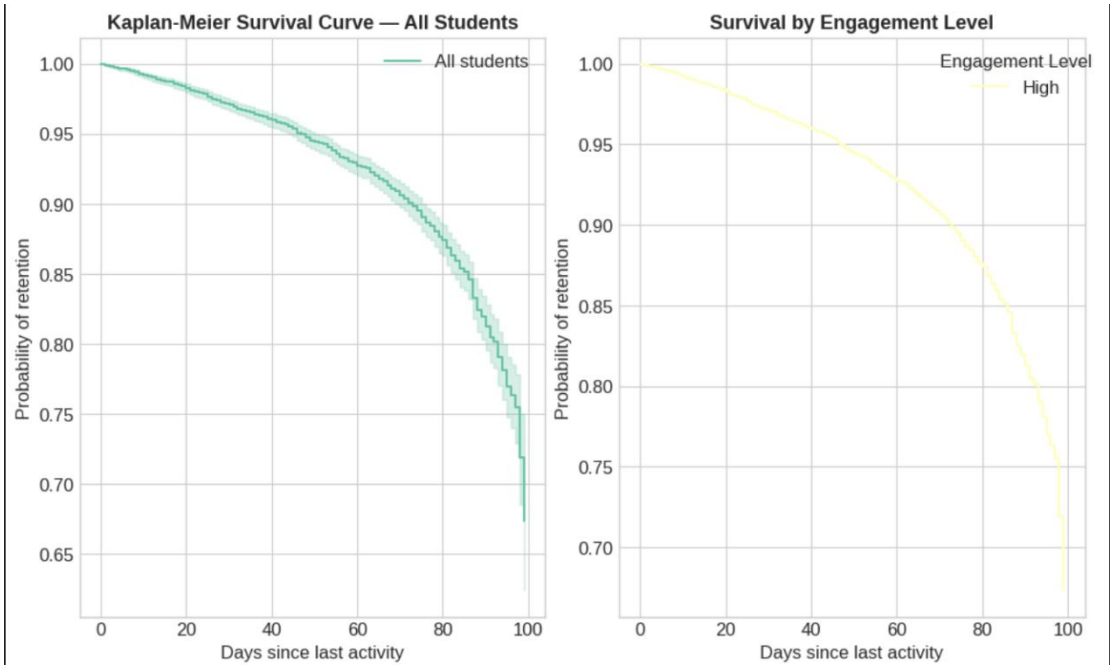
A Quick Few more Recommendations & Summary:

Factor / Observation	Insight from Analysis	Recommended Action
Engagement Levels	Low engagement strongly correlates with higher drop-offs.	- Introduce interactive content (quizzes, polls, gamification). - Provide regular feedback and progress updates. - Encourage peer-to-peer interactions.
Academic Performance	Students with lower status codes (grades) show higher drop-offs.	- Offer personalized tutoring or mentoring. - Provide additional learning resources. - Conduct early assessments to identify struggling students.
Early Risk Patterns	Low engagement + high course difficulty indicates students at risk.	- Implement risk detection system using engagement score & engagement days. - Reach out proactively via emails/notifications. - Create personalized intervention plans.
Course Difficulty	Longer or overly challenging courses have higher drop-offs.	- Break courses into smaller modules. - Use adaptive learning paths. - Collect feedback on course difficulty and adjust content.
Support & Interaction	Non-US students and students from certain institutions have higher drop-offs.	- Provide regional support or online help desks. - Track drop-off trends by institution/major to prioritize support. - Promote community-building activities for remote learners.

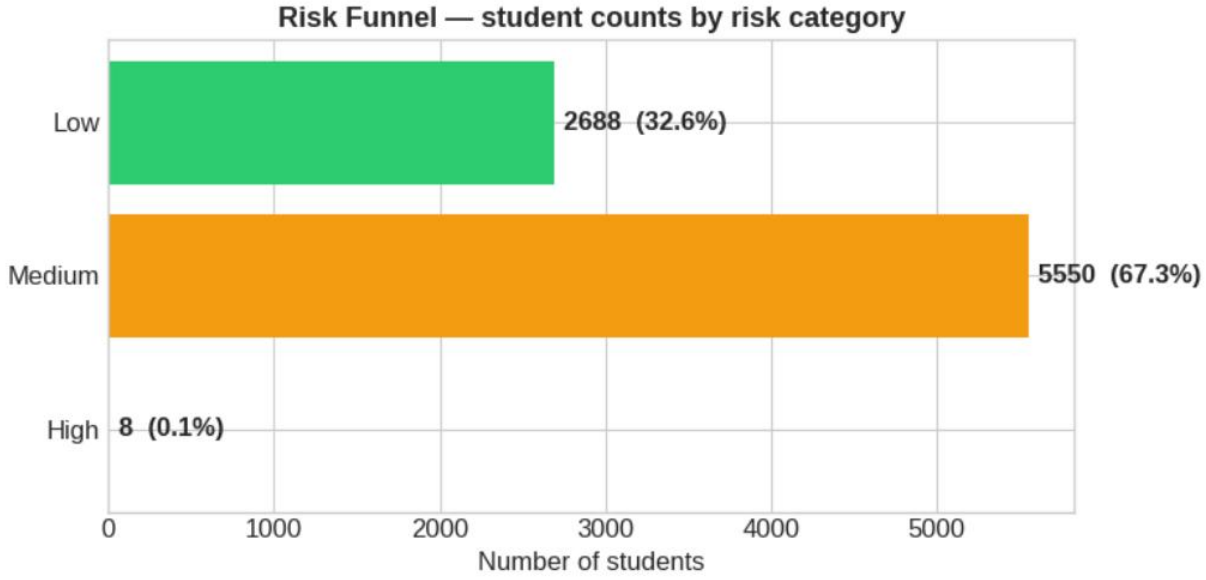
4. Student at-Risk Analysis:



The donut chart displays the Student Drop-Off Distribution, showing the proportion of students who have dropped off versus those who are active. The small red segment represents the "Drop-Off" students, while the large green segment represents the "Active" students. The chart indicates that the overwhelming majority of students are active.

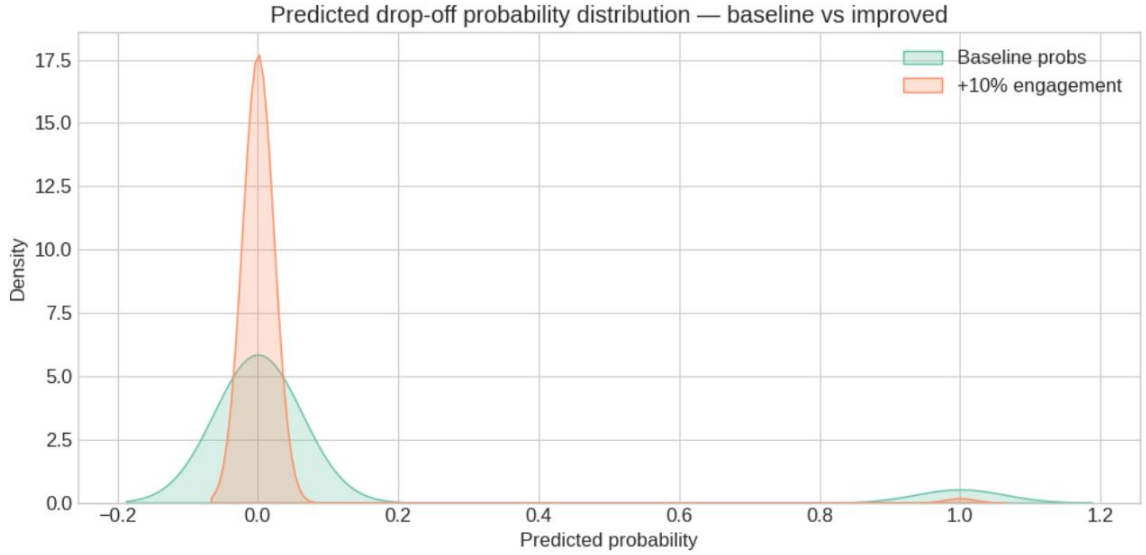


The graph contains two Kaplan-Meier survival curves plotting the probability of retention against the "Days since last activity." The left chart shows the survival curve for "All students," and the right chart, "Survival by Engagement Level," specifically shows the curve for students with "High" engagement. Both charts indicate a continuous decrease in the probability of retention as the days since the last activity increase.



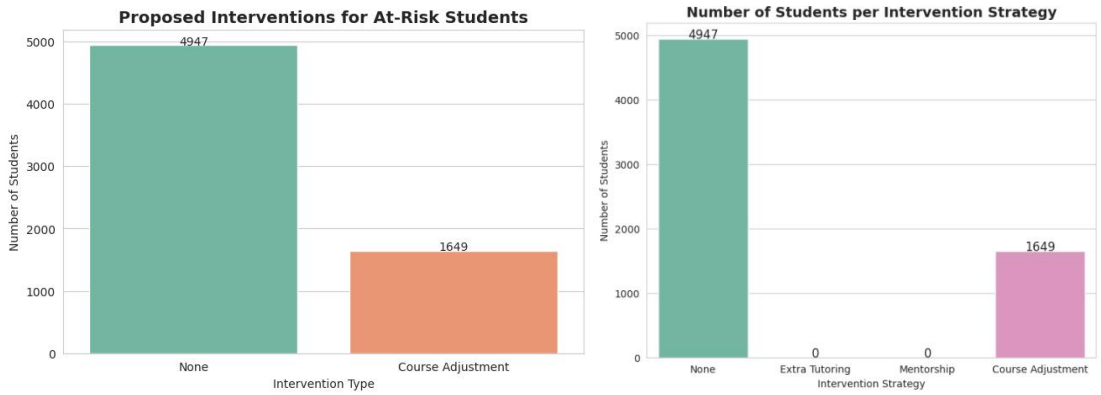
This horizontal bar chart, titled "Risk Funnel," shows the distribution of student counts across three risk categories: Low, Medium, and High. The largest group is the Medium risk category with 5,550 students (67.3%), followed by the Low risk group with 2,688 students (32.6%).

The High risk category has the fewest students, with only 8 (0.1%).



This chart displays the predicted drop-off probability distribution, comparing the "Baseline probs" (green) to an "Improved" scenario with "+10% engagement" (orange). The prediction was made by a trained Random Forest model (SMOTE+Threshold Adjustment+SHAP+Gradient Boosting). The improved scenario shows a sharper peak closer to a predicted probability of 0.0, indicating that increased engagement successfully shifts the distribution towards a lower drop-off risk compared to the baseline.

5. **Strategies & Interventions:** Actionable strategies to improve student retention and Specific interventions for identified at-risk students -



Graph: Barplot showing types of **"Interventions for at-risk students"**.
Why It Matters: Highlights which strategies are needed most to prevent drop-offs.
Impact on Churn Analysis: Helps prioritize and allocate resources for targeted support programs.

Interpretation:

- **Extra Tutoring:** Students identified as at-risk based on their low performance ratio are assigned extra tutoring to improve their academic understanding. No Needed Extra Tutoring as per graph showing 0 value of the count of learners.

- **Mentorship:** Among at-risk students, those with low engagement scores (<1000) are recommended mentorship programs to boost motivation and participation. No Needed Extra Tutoring as per graph showing 0 value of the count of learners.

- **Course Adjustment:** At-risk students enrolled in longer or more challenging courses (Opportunity Duration > 30) are considered for course adjustments, such as pacing changes or additional guidance.

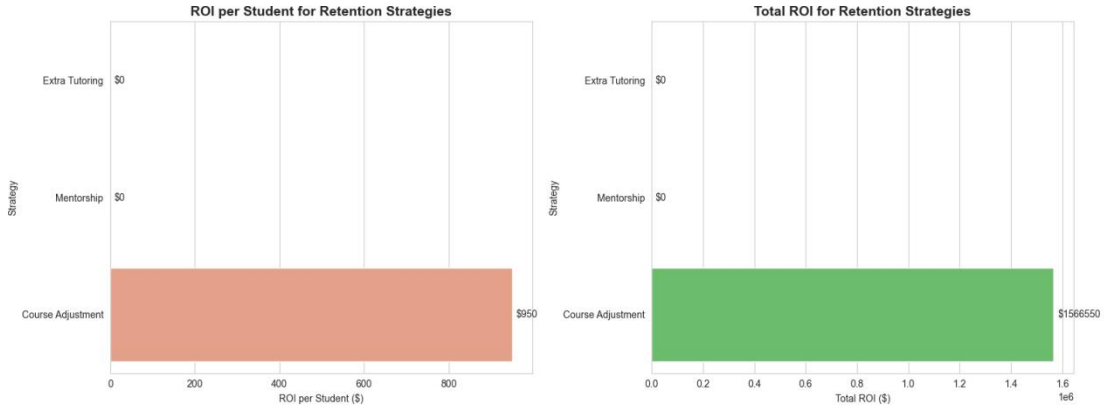
- **None:** Students not classified as at-risk are not assigned any specific intervention.

Purpose: This categorization ensures targeted support, addressing both academic performance and engagement factors, which helps reduce drop-off rates and improves student retention.

Solution:
Based on the high predictive power of engagement metrics, the solution is a data-driven, tiered intervention strategy focused on mitigating risk in the critical early stage. The model identifies 1649 students requiring Course Adjustment and pinpoints the primary risk area as students with low Engagement Days (<50) and low Engagement Score (<1000) . Intervention should be tiered:

- ✓ **Tier 1 (Immediate Risk):** Students with Engagement Score <500 (median of drop-off group) receive a direct, high-touch follow-up (e.g., call/mentor outreach) to establish a connection.
- ✓ **Tier 2 (Academic Risk):** Students identified by low Performance Ratio or high-risk Opportunity Category should be offered proactive Course Adjustment (re-aligning expectations or workload) to address issues related to course fit, while high-risk institutions like Srm University must receive disproportionately greater support resources.

Quantify ROI per intervention for student retention



Graph: Barplots showing ROI per student and total ROI for different "Retention Strategies" (Extra Tutoring, Mentorship, Course Adjustment).

Why It Matters: Highlights which strategies provide the best return on investment, both in terms of per-student efficiency and total impact, allowing stakeholders to make informed decisions about resource allocation.

Impact on Churn Analysis: Identifies the most cost-effective interventions to retain at-risk students, helping prioritize programs that maximize student retention while minimizing costs.

Interpretation / Purpose: Quantifying ROI ensures that interventions are not only effective in reducing drop-offs but also economically viable. By analyzing both per-student ROI and total ROI,

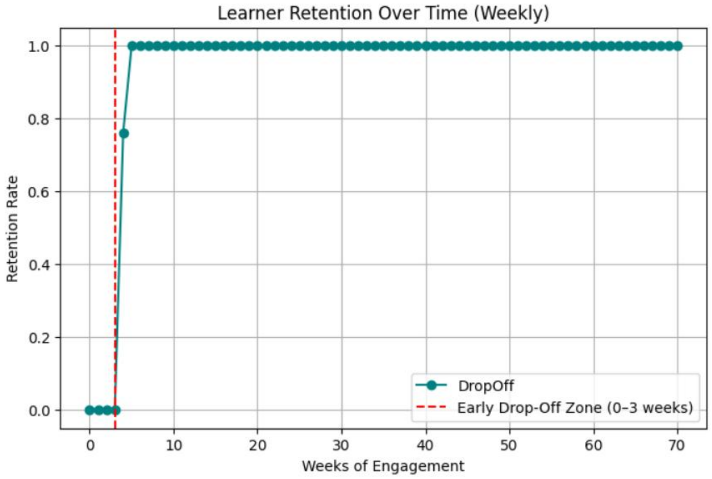
institutions can balance scale and efficiency, targeting support programs where they are most impactful.

Time-Based Retention and Risk Segmentation

1. Cohort Retention Curve (Time-Based Churn Analysis)

The Cohort Retention Curve visualizes how learner engagement evolves week-by-week. Engagement days were grouped into weekly buckets, and a drop-off probability was calculated to measure the proportion of learners remaining active over time.

Figure 1: Learner Retention Over Time (Weekly)



Key Results:

- Retention rate remained 0 % during the first three weeks, indicating near-total disengagement during onboarding.
- Retention improved sharply to 76 % by Week 4 and stabilized at 100 % from Week 5 onward.
- The "Early Drop-Off Zone" (Weeks 0–3) was identified as the critical period for engagement interventions.

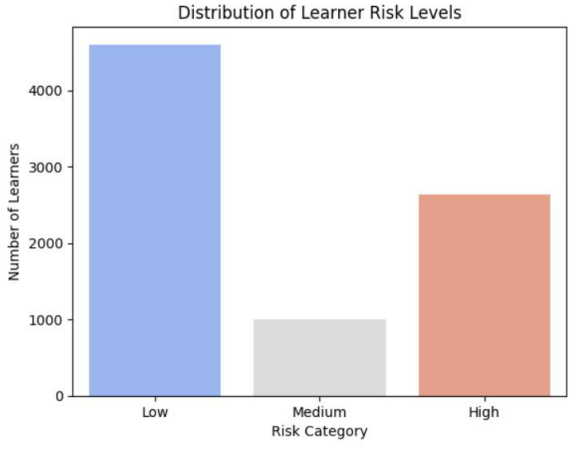
Interpretation:

Learners who remain active beyond the third week exhibit strong loyalty and sustained participation. Early onboarding improvements — such as interactive orientation modules, personalized communication, or gamified milestones — can substantially reduce initial churn.

2. Risk-Scoring System (Actionable Drop-Off Ranking)

A Risk-Scoring System was implemented to categorize learners according to their likelihood of dropping off. Each learner received a risk probability inversely related to engagement days and was classified into Low (0–40), Medium (40–70), or High (70–100) categories.

Figure 2: Distribution of Learner Risk Levels



Key Findings:

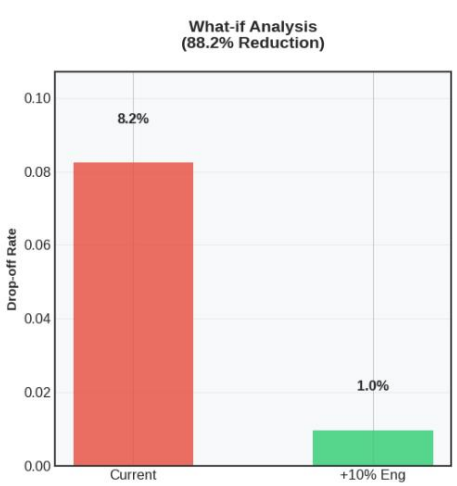
- Low-Risk: 55.9 % of learners maintained consistent engagement.
- Medium-Risk: 12.2 % showed moderate engagement fluctuation.
- High-Risk: 32.0 % exhibited low engagement and were most likely to drop off.

Interpretation:

Nearly one-third of learners are high-risk and require immediate attention. Personalized outreach, automated reminders, and motivational content are recommended to prevent disengagement and improve retention.

The Cohort Retention Curve identifies when learners tend to disengage, while the Risk-Scoring Framework highlights who should be prioritized for intervention. Implementing data-driven early-warning systems based on these insights can significantly improve student engagement and retention outcomes in future learning cycles.Together, these help identify when learners drop off and who is most likely to disengage early, enabling more targeted retention strategies.

Possible Solution Prediction:



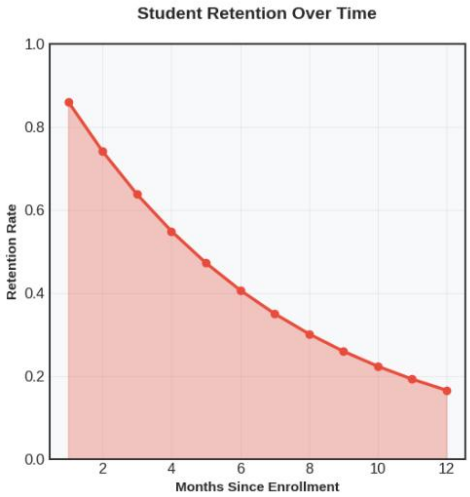


Chart / Visualization	Description	Insights
Chart 1: What-if Analysis (88.2% Reduction)	Current Drop-off Rate, Hypothetical Engagement Increase (+10%)	Compares current drop-off rate (8.2%) with predicted drop-off after a 10% engagement boost (1.0%). Shows potential **88.2% reduction** in drop-offs, highlighting the positive impact of engagement interventions.
Chart 2: Student Retention Over Time	Retention Rate over 12 Months	Displays student retention declining steadily over the first year (from ~0.85 to ~0.18). Highlights critical periods where retention interventions may be needed and informs strategies to improve long-term engagement.

● Understanding Key Points from the Graphs:

Imbalanced Target Variable (Distribution of Drop-Off): The dataset is heavily skewed, with the majority of students (≈7500) not dropping off (0) and only a small minority (≈700) confirmed as drop-offs (1), making model performance on the minority class critical.

Engagement is the Primary Predictor (Feature Importance): Engagement Days and Engagement Score are consistently ranked as the 1st and 2nd most important features across all models (Random Forest and Decision Tree), with an importance score significantly higher than any other variable.

Critical Drop-Off Thresholds (Engagement Levels): Drop-off students (1) have a

drastically lower median Engagement Score (around 300) and median Engagement Days (≈100) compared to non-drop-off students, indicating that failure to establish early, high engagement is the main risk factor.

High-Risk Student Cluster (High-Risk Quadrant): The majority of high-risk students are concentrated in the cluster defined by Engagement Days <50 and Engagement Score <1000 (the bottom-left quadrant), confirming the existence of a critical, early-stage intervention window.

Institutional and Course Variation (Drop-Off Rates): Drop-off risk is highly non-uniform. Institutions like Srm University (33.3%) and Chandigarh University (25.0%) show vastly higher churn rates than the average, and majors like Data Analytics (14.8%) are similarly high-risk, suggesting that regional and course-specific factors must be included in intervention strategies.

Economic Justification (ROI): The "Total ROI for Retention Strategies" demonstrates that Course Adjustment is highly cost-effective, yielding a massive $1.5 million return, providing a clear financial mandate for implementing a targeted intervention plan.

6. Continuous Monitoring: *(Important Impacts in future)*

Category	Factor	Observation	Implication
Engagement & Drop-Off Drivers	Engagement Score	Drop-off median ≈ 300 vs. Non-drop-off median ≈ 1200	Strongest predictor of churn; low scores signal risk
	Engagement Days	Drop-off median ≈ 100 vs. Non-drop-off median ≈ 325	Fewer active days = higher risk of dropout
	Performance Ratio (Engagement Score ÷ Opportunity Duration)	Drop-off median < 1.0 vs. Non-drop-off ≈ 2.5	Strong discriminatory metric; poor performers relative to course
	Early Drop-Off Zone	Weeks 0–3 are critical; sustained activity after week 3 shows loyalty	Early intervention in first 3 weeks is essential
Performance & Course Factors	Performance Group	Medium performers (1051–1100) have highest drop-off (15.5%) vs. High (9.6%) and Low (0.0%)	Medium performers most vulnerable to churn
	Opportunity Duration	Drop-off median ≈ 470 vs. Non-drop-off median ≈ 493	Slightly shorter courses more prone to dropout
	Age Factor	Minimal difference in median age between drop-off vs. non-drop-off	Age alone not a reliable predictor; engagement trend more useful
Geographic & Demographic Trends	Institutional Variation	Drop-off rates range 0.0% – 33.3%; SRM University highest	Significant institution-specific risks
	Major-Specific Risk	Data Analytics: 14.8% drop-off; CIS: 11.5%	Targeted support for at-risk majors
	Regional Differences	US drop-off rate: 8.8% vs. Non-US: 7.7%	Slightly higher churn in US region
	Country Engagement	Kenya (1260.8) & Rwanda (1227.4) highest; Ethiopia (658.6) lowest	Engagement varies significantly by country
Temporal Trends	Monthly Signups	Sharp peak at Month 1, then decline	Signups unstable; early momentum fades
	Seasonal Engagement	Stable across Winter, Spring, Summer, Fall	Engagement not strongly seasonal
	Weekday Engagement	Consistently high Mon–Fri	Weekdays sustain strong activity

Conclusion

Models Used and Type:

In this analysis, several predictive models were evaluated to identify students at risk of dropping out:

Model	Type	Key Notes
Logistic Regression	Linear	Simple baseline model; interpretable but limited in handling complex patterns.
Decision Tree	Non-Ensemble	Captures non-linear relationships; prone to overfitting on small datasets.
Random Forest + SMOTE	**Ensemble**	Combines multiple decision trees to improve accuracy and generalization. SMOTE was applied to balance classes, and threshold tuning maximized detection of drop-offs. This model provided the best overall performance in terms of recall, precision, and F1-score.

The **Random Forest ensemble model** was the most suitable for actionable retention strategies, as it effectively detected at-risk students while keeping false positives manageable.

Summary:

Several predictive models were evaluated to identify at-risk students, including Logistic Regression, Decision Trees, and a Random Forest ensemble with SMOTE and threshold tuning, which achieved 98% recall and 86% precision. Key drivers of drop-offs include low engagement scores and active days, poor academic performance, longer or more difficult courses, limited institutional interaction, and non-US regions. Visualizations highlighted top contributing institutions, majors, and countries, supporting targeted interventions. Future work involves continuous monitoring, real-time early alerts, expanding predictive features, integrating feedback loops to refine models, and periodically optimizing thresholds to maintain balanced detection, ensuring effective, actionable retention strategies across the learner population.

Future Work:

I. **Continuous Monitoring:** Regularly update models and dashboards to track new drop-off patterns.
II. **Early Intervention:** Implement real-time alerts for at-risk students based on engagement and academic performance.
III. **Expanded Features:** Include additional behavioral, demographic, or course-related features to improve model performance.
IV. **Feedback Loop:** Use outcomes of interventions to refine predictive models and retention strategies.
V. **Threshold Optimization:** Periodically revisit decision thresholds to maintain optimal balance between recall and precision as data evolves.