LARANA, INC.

PREDICTING STUDENT DROPOUTS

OUR TEAM

COLLABORATING TO SOLVE REAL-WORLD PROBLEMS THROUGH DATA INSIGHTS.

Gabriel Mancillas

Data Scientist

Amayrani Balbuena

Data Scientist

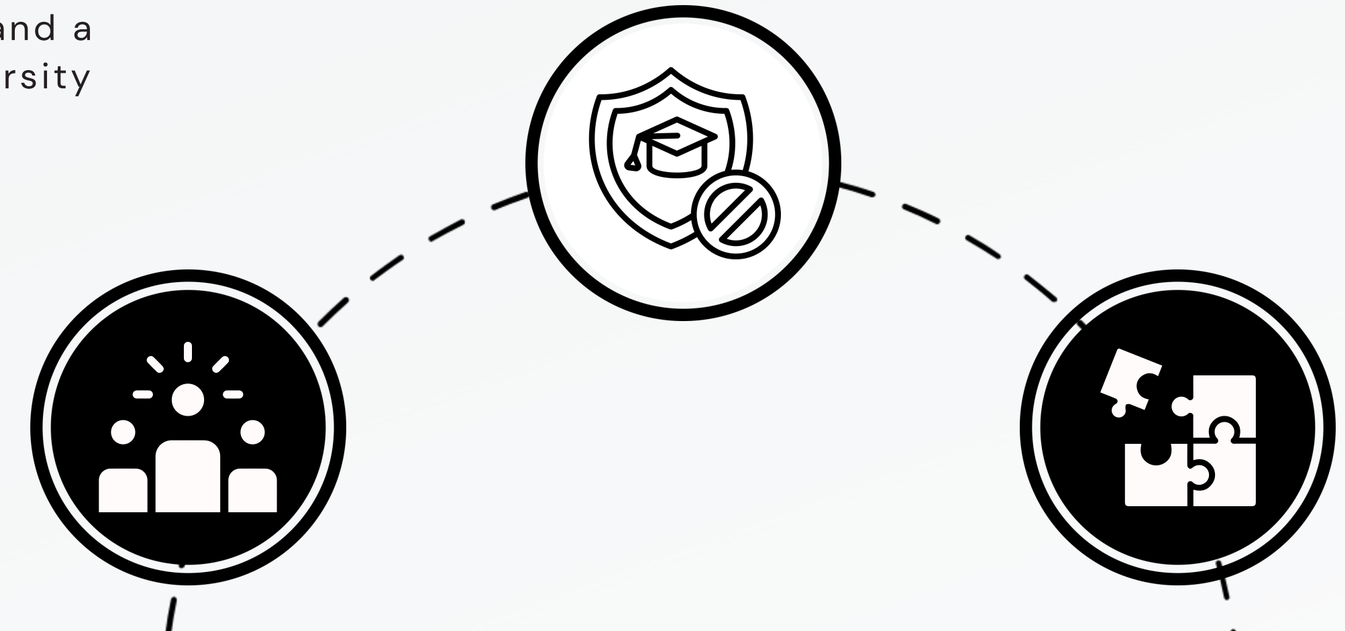
THE PROBLEM: STUDENT DROPOUTS

Universities face high dropout rates

High dropout rates lead to lost tuition revenue, lower student success rates, and a decline in university rankings

Impact on the university

Dropouts result in **significant financial losses** and lower rankings.



Why it matters

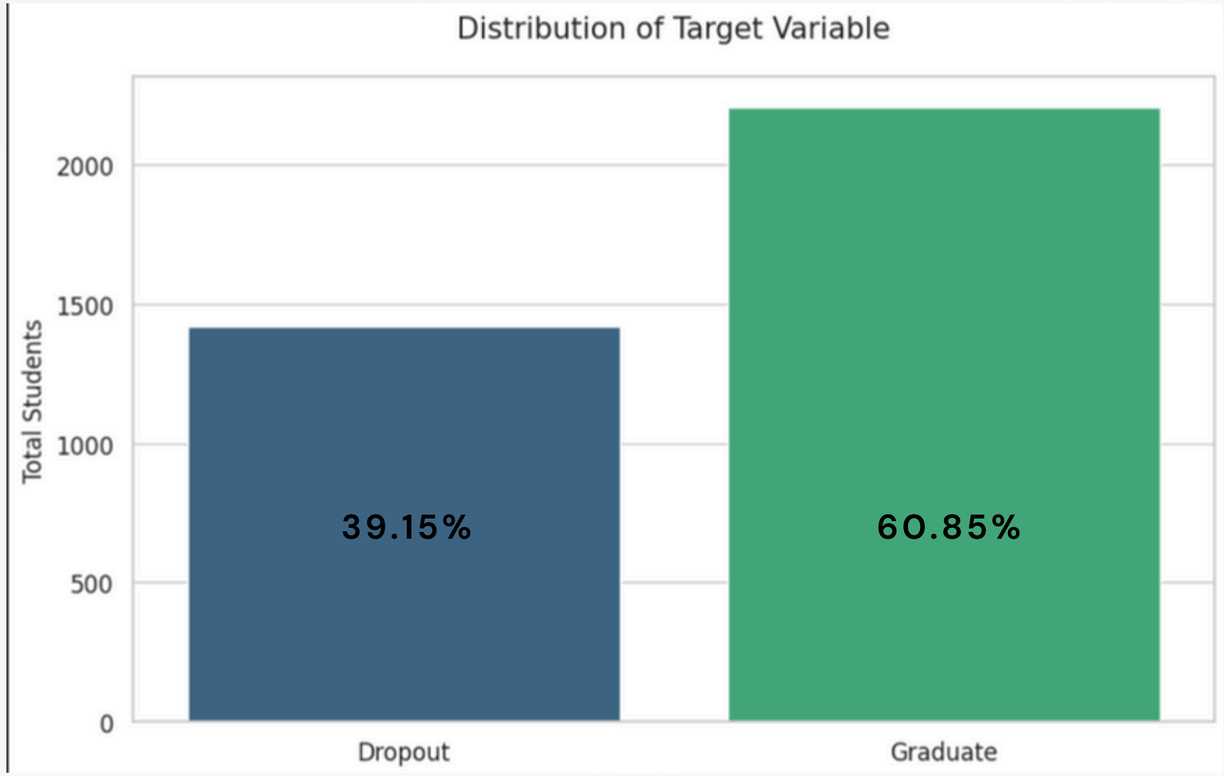
Early identification allows universities to offer support and prevent students from dropping out, improving overall retention

DATASET SUMMARY

Total of 4,424 student records and 35 variables including demographic, academic, and financial data.

Key summary statistics

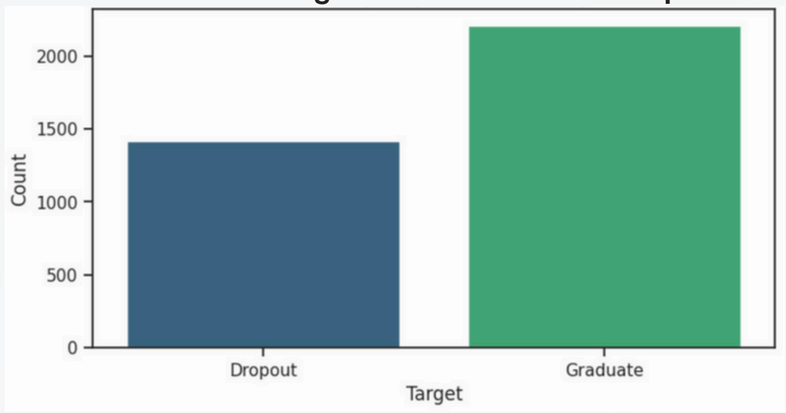
Variable	Key Statistics
Curricular Units 2nd Sem (Grades)	Mean: 10.23, Min: 0.00, Max: 18.57, Std: 5.21
Age at Enrollment	Mean: 20.92, Min: 17, Max: 45
Scholarship Holders	1,000 holders (23%)
Debtor Status	800 debtors (18%)
Unemployment Rate	Mean: 11.57%, Min: 7.60%, Max: 16.20%
Dropout Rate	Dropouts: 39.15%, Graduates: 60.85%



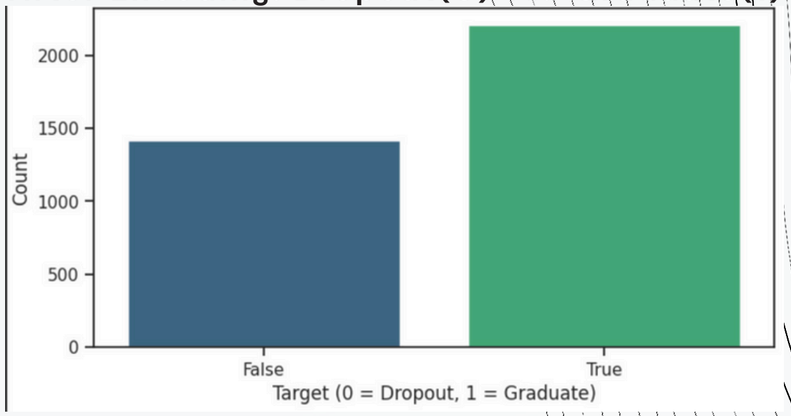
DATA PREPROCESSING

- Categorical Variables:
  - **One-hot encoding** used for variables like **marital status, attendance**, and **enrollment type**
- Handling Missing Data:
  - Dataset contained **no missing values**, ensuring full data completeness.
- Feature Scaling:
  - **Features like age, GPA, and curricular units passed** were standardized for consistency across models.
- Removed irrelevant columns like:
  - Nationality
  - International Status
  - Application mode
  - Curricular units without evaluations

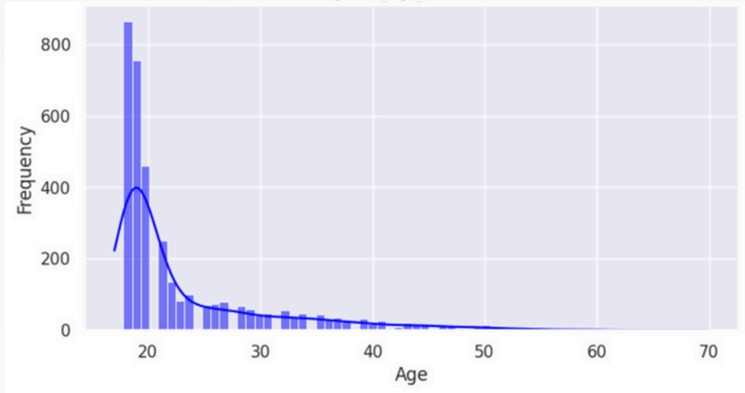
Before Encoding: Graduate vs. Dropout



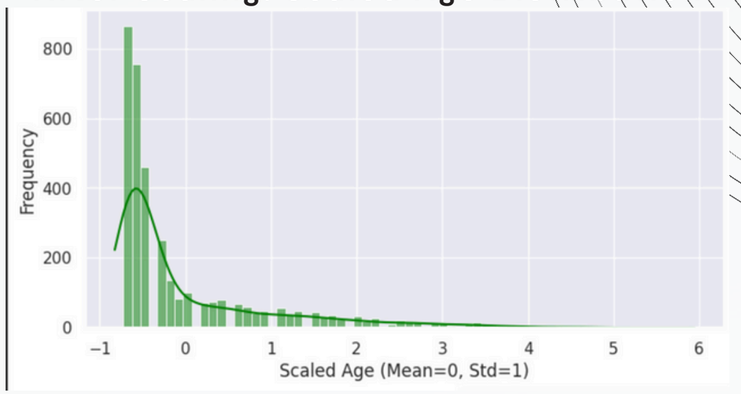
After Encoding: Dropout(0) vs. Graduate(1)



Before Scaling: Age Distribution

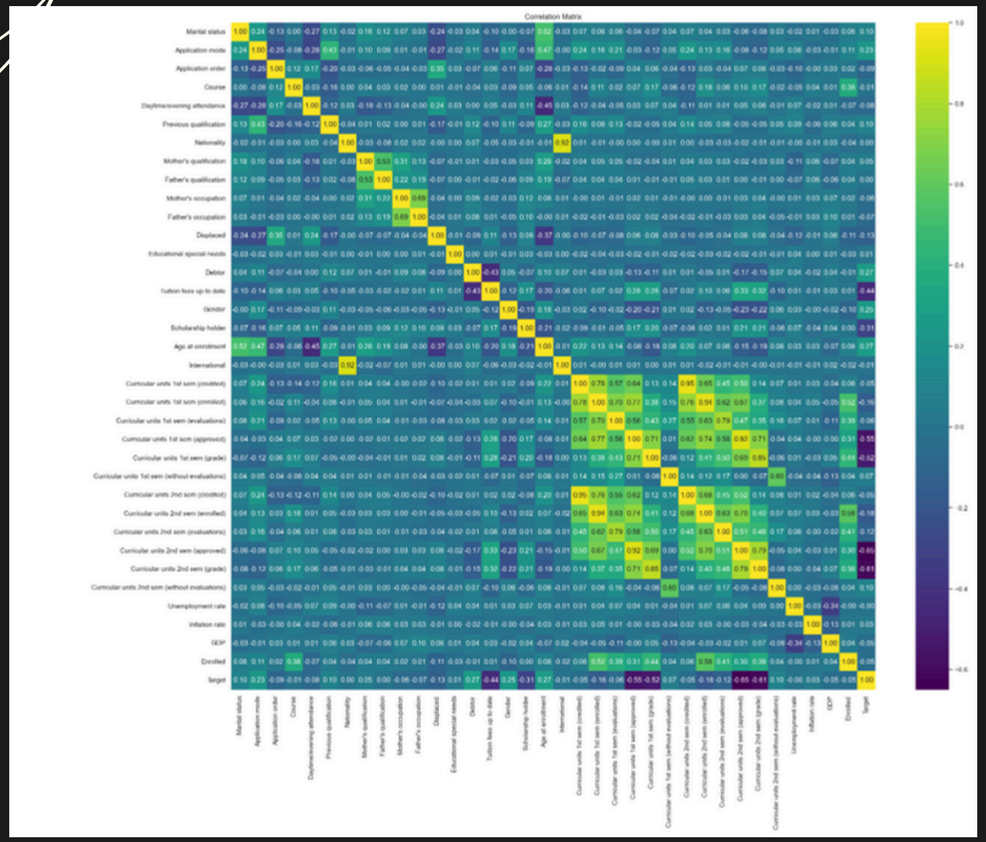


After Scaling: Scaled Age Distribution

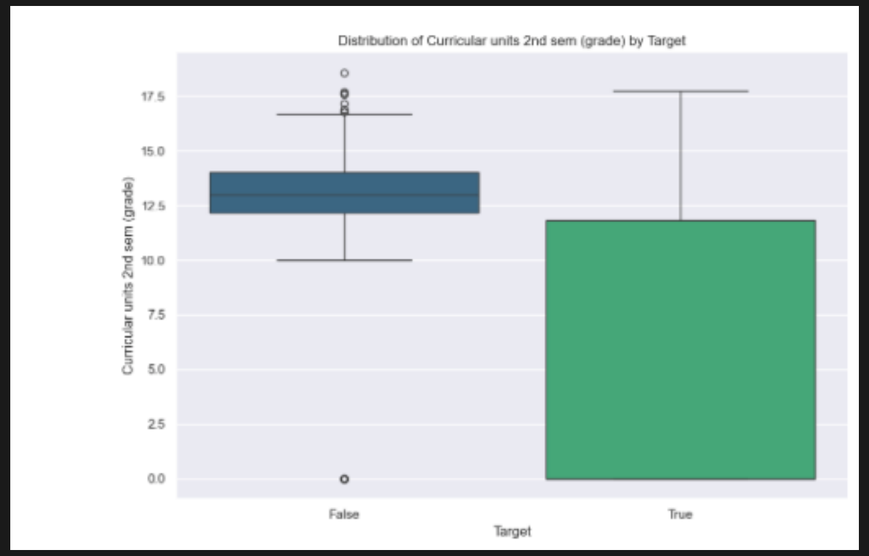


EXPLORATORY DATA ANALYSIS

Low academic performance correlates with higher dropout risk



- **Older students** and those with **financial issues** have a higher probability of dropping out



- Students with **high absenteeism** or who **fail key courses** are at increased risk of dropping out

LIBRARY OVERVIEW AND PREPROCESSING STEPS

- **Import libraries:**
  - LogisticRegression, RandomForestClassifier, GradientBoostingClassifier, SVC, KNeighborsClassifier, DecisionTreeClassifier for model building.
  - train_test_split for splitting data.
  - StandardScaler for feature standardization.
- Step 2: Load and preprocess data:
  - Features (X) are selected by dropping the “Target” column.
  - Target variable (y) is assigned from the “Target” column.
- Step 3: Split the data:
  - Use train_test_split to divide the data into training and testing sets with a test size of 20% and a random seed for reproducibility.
- Step 4: Standardize features:
  - StandardScaler is applied to scale the feature variables for both training and testing data.

```python
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier

# Step 2: Load and preprocess the data
# Assuming df is your DataFrame and 'Target' is the column you want to predict
X = df.drop(columns="Target")  # Features
y = df["Target"]  # Target variable

# Step 3: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Step 4: Standardize the feature variables
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

MODEL SELECTION

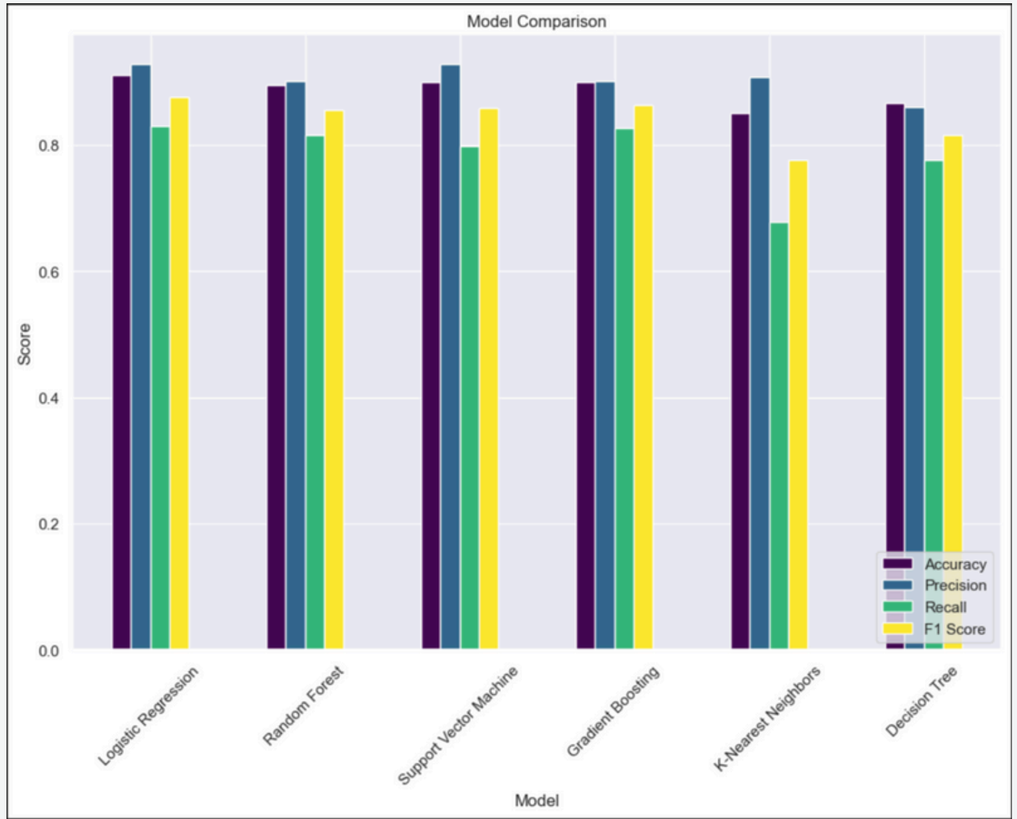
Summary of Tested Models:

- Logistic Regression: Simple, interpretable, accuracy of 85%.
- Random Forest: Complex data handling, accuracy of 91%.
- Support Vector Machine (SVM): Good accuracy but difficult to interpret
- Gradient Boosting: High accuracy but more prone to overfitting and slower than Random Forest.

Model	Accuracy	Precision	Recall	F1-score	Interpretability
Logistic Regression	90%	91%	91%	0.88	High
Random Forest	**90%**	**90%**	**92%**	**0.86**	**Medium**
Gradient Boosting	90%	90%	93%	0.86	Low
Support Vector Machine	90%	90%	91%	0.87	Low

Final Model Selection:

Random Forest was selected for its ability to handle complex data, its strong accuracy (90%), and its balance between performance and interpretability, making it suitable for deployment.



```python
# Step 5: Define the models and their hyperparameter grids
models = {
    "Logistic Regression": {
        "model": LogisticRegression(max_iter=500, random_state=42),
        "params": {
            'C': [0.01, 0.1, 1, 10, 100],
            'penalty': ['l1', 'l2'],
            'solver': ['liblinear', 'saga']
        }
    },
    "Random Forest": {
        "model": RandomForestClassifier(random_state=42),
        "params": {
            'n_estimators': [50, 100, 200],
            'max_depth': [None, 10, 20, 30],
            'min_samples_split': [2, 5, 10]
        }
    },
    "Support Vector Machine": {
        "model": SVC(random_state=42),
        "params": {
            'C': [0.1, 1, 10, 100],
            'kernel': ['linear', 'rbf', 'poly'],
            'gamma': ['scale', 'auto']
        }
    },
    "Gradient Boosting": {
        "model": GradientBoostingClassifier(random_state=42),
        "params": {
            'n_estimators': [50, 100, 200],
            'learning_rate': [0.01, 0.1, 0.2],
            'max_depth': [3, 5, 7]
        }
    },
    "K-Nearest Neighbors": {
        "model": KNeighborsClassifier(),
        "params": {
            'n_neighbors': [3, 5, 7, 9],
            'weights': ['uniform', 'distance'],
            'metric': ['euclidean', 'manhattan']
        }
    },
    "Decision Tree": {
        "model": DecisionTreeClassifier(random_state=42),
        "params": {
            'max_depth': [None, 10, 20, 30],
            'min_samples_split': [2, 5, 10],
            'criterion': ['gini', 'entropy']
        }
    }
}
```

```python
# Step 6: Perform Grid Search with cross-validation for each model
best_models = {}
for name, model_info in models.items():
    grid_search = GridSearchCV(model_info["model"], model_info["params"], cv=5, scoring='accuracy')
    grid_search.fit(X_train_scaled, y_train)
    best_models[name] = grid_search.best_estimator_
    print(f"{name} - Best Hyperparameters: {grid_search.best_params_}")

# Step 7: Train each model on the training data and evaluate on the test set
for name, model in best_models.items():
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_test_scaled)

    # Evaluate the model
    accuracy = accuracy_score(y_test, y_pred)
    print(f"\n{name} - Test Set Accuracy: {accuracy:.2f}")

    # Confusion matrix
    conf_matrix = confusion_matrix(y_test, y_pred)
    print(f"{name} - Confusion Matrix:")
    print(conf_matrix)

    # Classification report
    class_report = classification_report(y_test, y_pred)
    print(f"{name} - Classification Report:")
    print(class_report)
```

MODEL PERFORMANCE & HYPERPARAMETER TUNING

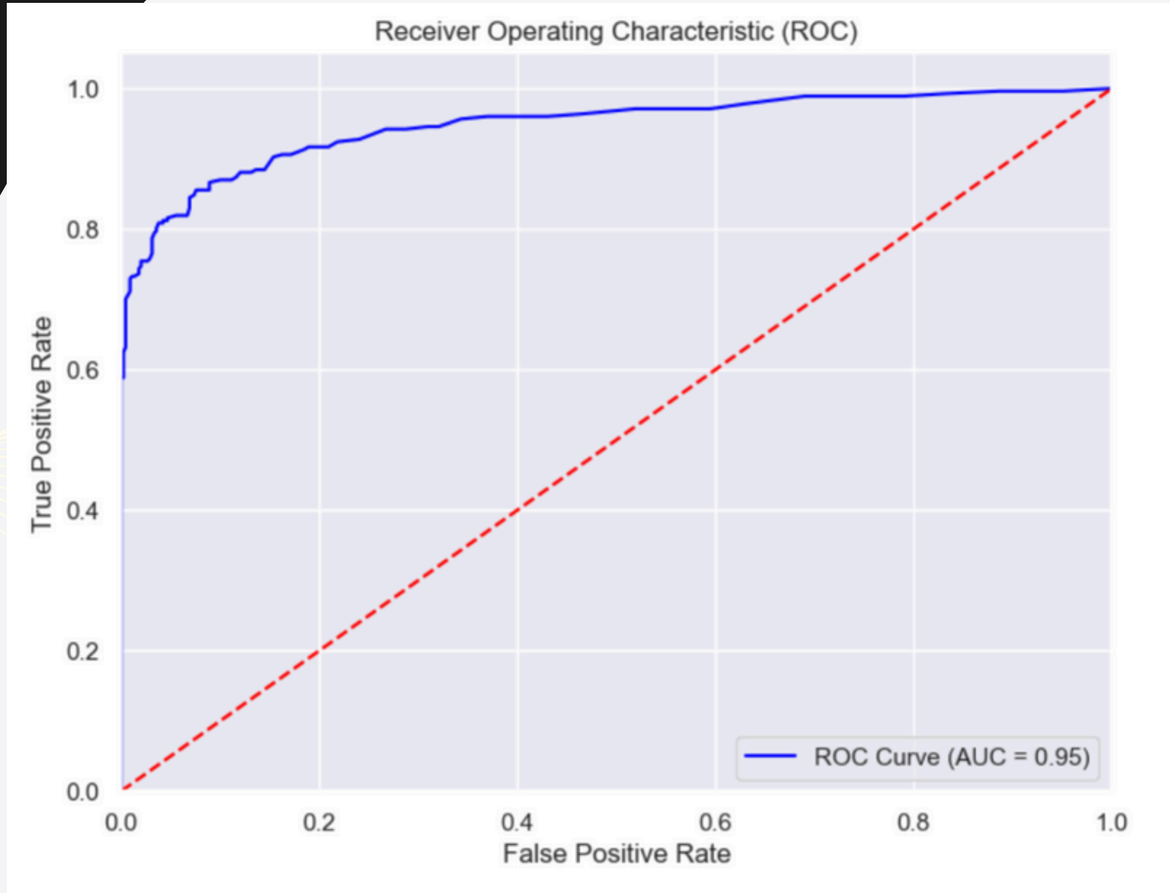
2. GRID SEARCH AND CROSS-VALIDATION:
WE APPLIED 5-FOLD CROSS-VALIDATION WITH GRIDSEARCHCV TO ENSURE THAT THE SELECTED HYPERPARAMETERS GENERALIZED WELL TO UNSEEN DATA. THE ACCURACY SCORING METRIC WAS USED TO GUIDE OPTIMIZATION WHILE PREVENTING OVERFITTING.

3. TRAINING AND MODEL EVALUATION:
AFTER TUNING, EACH MODEL WAS TRAINED ON SCALED DATA AND EVALUATED USING:

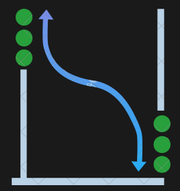
- ACCURACY SCORE FOR OVERALL CORRECTNESS.
- CONFUSION MATRIX TO ASSESS TRUE/FALSE POSITIVES AND NEGATIVES.
- CLASSIFICATION REPORT (PRECISION, RECALL, F1-SCORE) FOR DEEPER PERFORMANCE INSIGHTS, ESPECIALLY WITH IMBALANCED DATA.

MODEL VALIDATION

- Cross-validation: 5-fold, avoiding overfitting
- Data Split: 80% training, 20% testing
- ROC Curve: AUC = 0.95
- Final Metrics: Accuracy = 90%, Precision = 90%, Recall = 92%, F1-score = 0.86

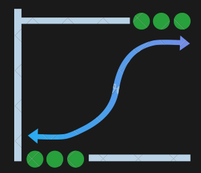


MODEL PERFORMANCE AND RESULTS



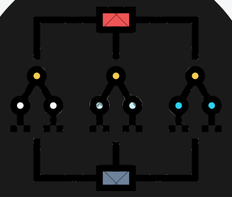
- Accuracy: N/A
- Precision: N/A
- Recall: N/A
- F1-Score: 0.87

XGBOOST AND AMAZON SAGEMAKER



- Accuracy: 91%
- Precision: 91%
- Recall: 84%
- F1-Score: 0.88

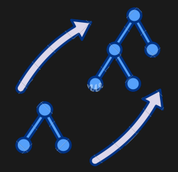
LOGISTIC REGRESSION



- Accuracy: 90%
- Precision: 90%
- Recall: 92%
- F1-Score: 0.86

Selected as the final model due to its high accuracy, strong precision, and ability to handle complex relationships.

RANDOM FOREST



- Accuracy: 90%
- Precision: 91%
- Recall: 83%
- F1-Score: 0.87

GRADIENT BOOSTING

FROM TECHNICAL INSIGHTS TO BUSINESS IMPACT

Our predictive model achieved 91% accuracy in identifying at-risk students. This allows universities to act proactively, leading to improved retention rates and enhanced financial outcomes

91% accuracy in predicting at-risk students

TECHNICAL INSIGHTS

Allows universities to proactively identify at-risk students

EARLY IDENTIFICATION

Efficient allocation of resources to improve student outcomes

TARGETED INTERVENTION

IMPROVED RETENTION/ FINANCIAL OUTCOMES

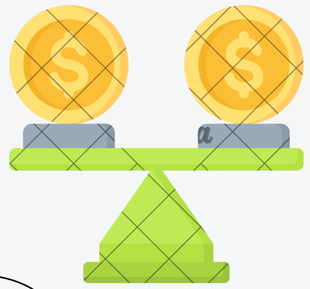
KEY INSIGHTS FOR UNIVERSITIES

- Insight 1: Early identification of at-risk students provides a **91% accurate** prediction, enabling timely interventions.
- Insight 2: **Low academic performance** is a key indicator of dropout risk.
- Insight 3: Students facing **financial difficulties** are more likely to drop out.
- Insight 4: **Older students** have an increased risk of dropout, particularly those balancing work and study.

BUSINESS IMPACT

Improved Retention

Early identification of at-risk students allows for **timely interventions**, improving student retention

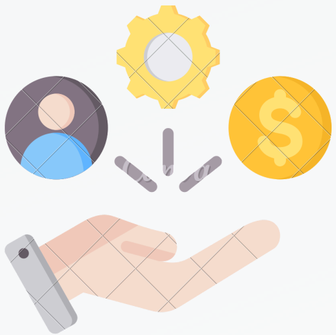


Financial Stability

Reducing dropout rates leads to **increased tuition revenue**, supporting the university's financial goals

Reputation

Higher retention rates enhance the **university's reputation** and rankings, attracting more students



Resource Allocation

Focusing resources on at-risk students ensures support efforts are more effective and targeted

IMPLEMENTATION AND NEXT STEPS

Integrate the predictive model with the university's student information system and ensure seamless data updates.

PHASE 1: DATA INTEGRATION

Deploy the model in a live environment for real-time predictions.

PHASE 2: MODEL DEPLOYMENT

Train staff on how to use the model's predictions for early intervention strategies with at-risk students.

PHASE 3: STAFF TRAINING

Continuously monitor model performance and update as needed with new data to improve predictions

PHASE 4: ONGOING MONITORING AND UPDATES

CALL TO ACTION

- Partner with Us: Collaborate with us to implement this solution and improve your university’s retention rates

- Schedule a Demo: Contact us today to schedule a demo or consultation to discuss how we can customize the model for your needs.

- Start Improving Retention: Take immediate steps to reduce dropout rates, increase revenue, and improve student outcomes.

THANK'S FOR WATCHING

LARANA, INC.

OUR CALLOBORATION

This project was a collaborative effort, with both team members contributing across the entire workflow. Below is a breakdown of the key areas where we collaborated.

Task	Contribution
Data Preprocessing	Both team members handled data cleaning, feature engineering, and scaling.
Exploratory Data Analysis	Both collaborated to explore the data, generate insights, and visualize results.
Model Building	Both worked together to build, tune, and evaluate the models.
Presentation	Both shared the responsibility of creating the final report and presentation.