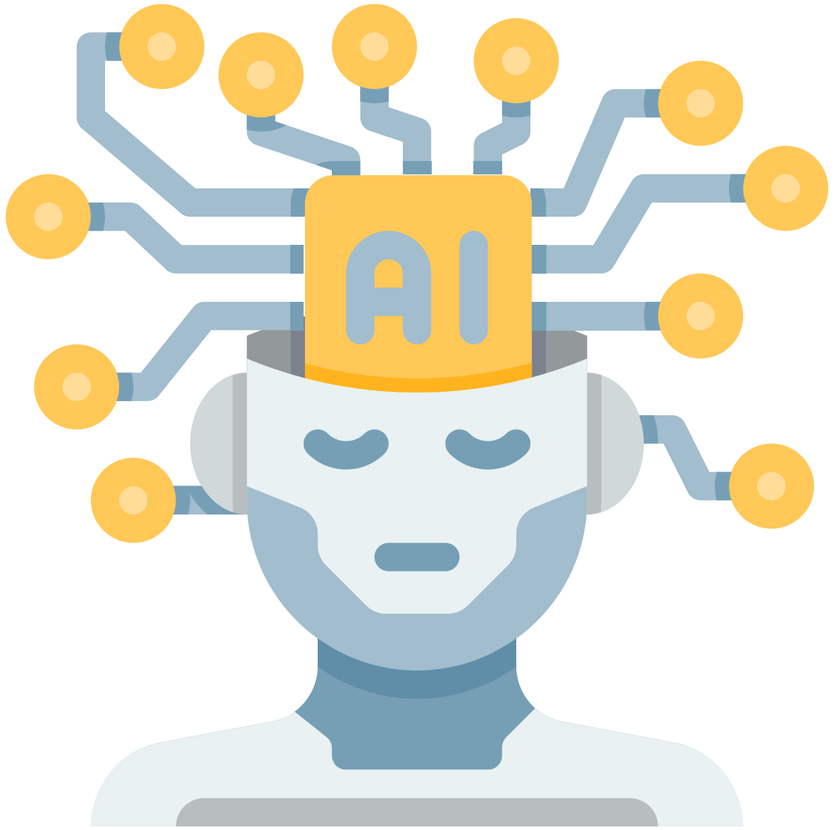
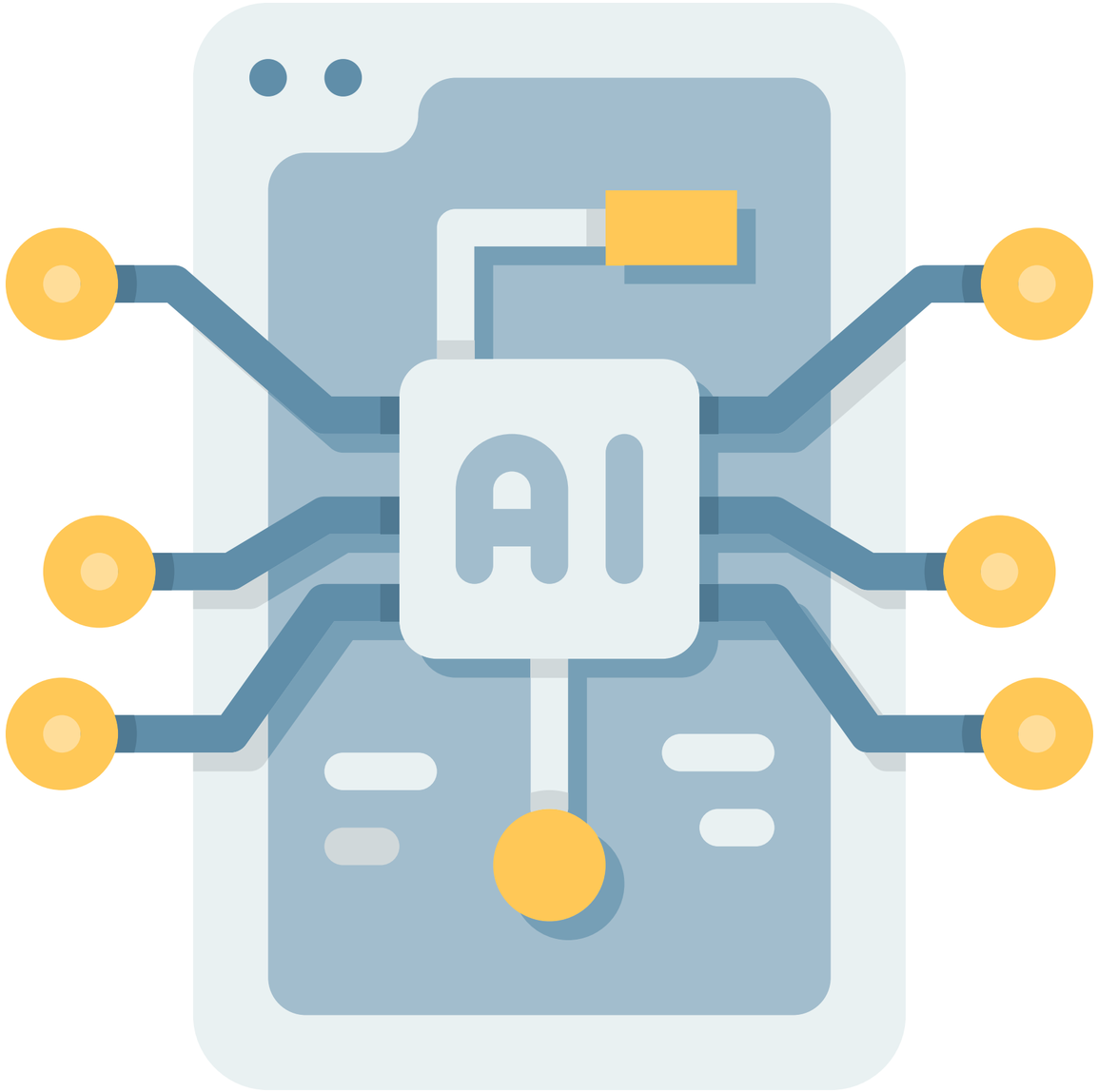
Predicting Student Success Using Machine Learning

Nicole, Mehdi, Tania, Houleye





Project overview

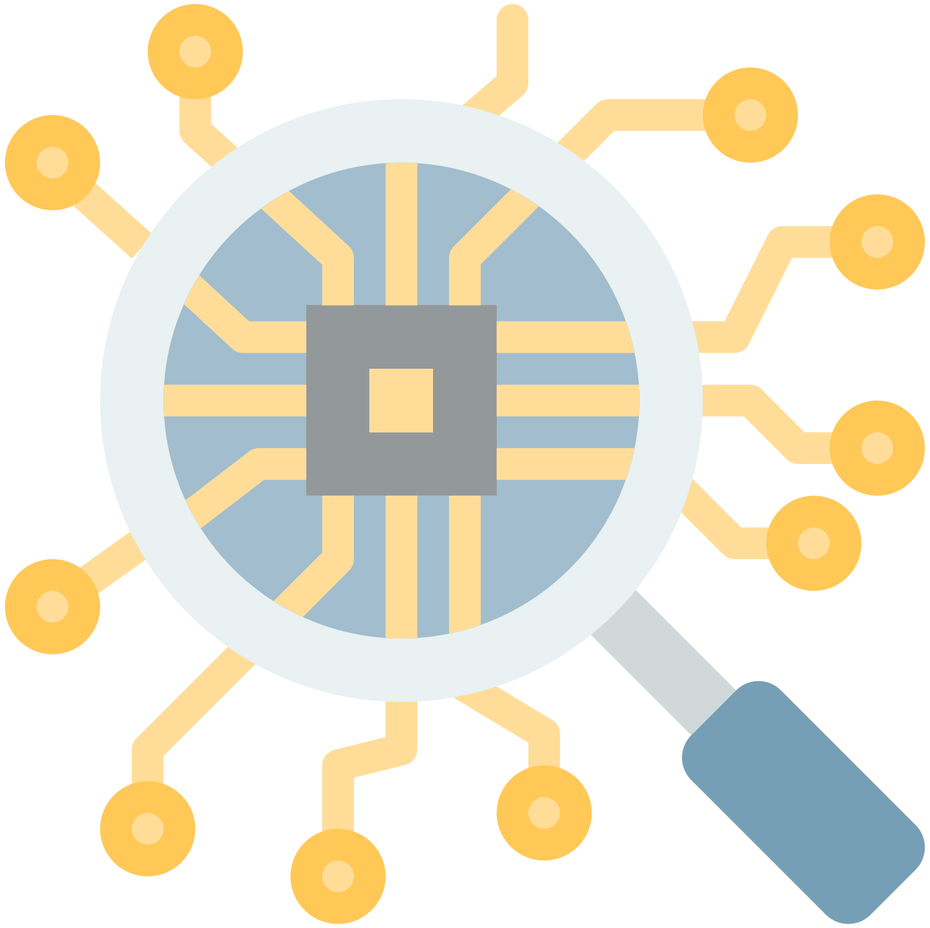
Problem:

Our goal is to predict student academic success based on various features. A classification model is developed to forecast the likelihood of successful graduation (Pass/Fail).

Objective:

Improve decision-making in educational institutions by identifying at-risk students early and providing targeted support.

The Process



Data Processing

Exploratory Data Analysis

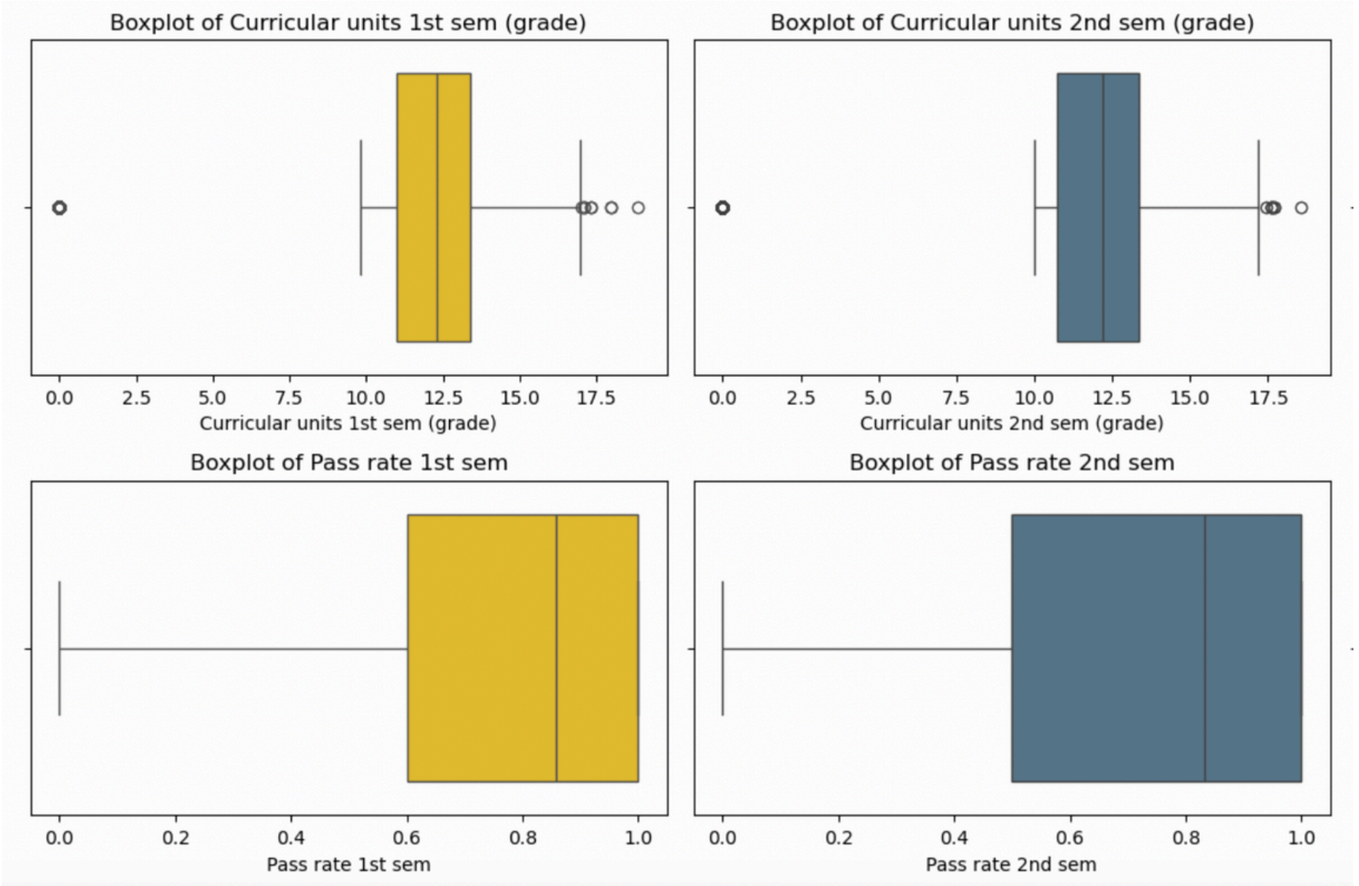
Target Variable Encoding

Feature Engineering and Standardization

Developp a KNN model

Train and Evaluate KNN Model

Data Selection and Preparation



Dataset:

The dataset includes academic performance data, such as grades, attendance, and demographic information from 4,424 students.

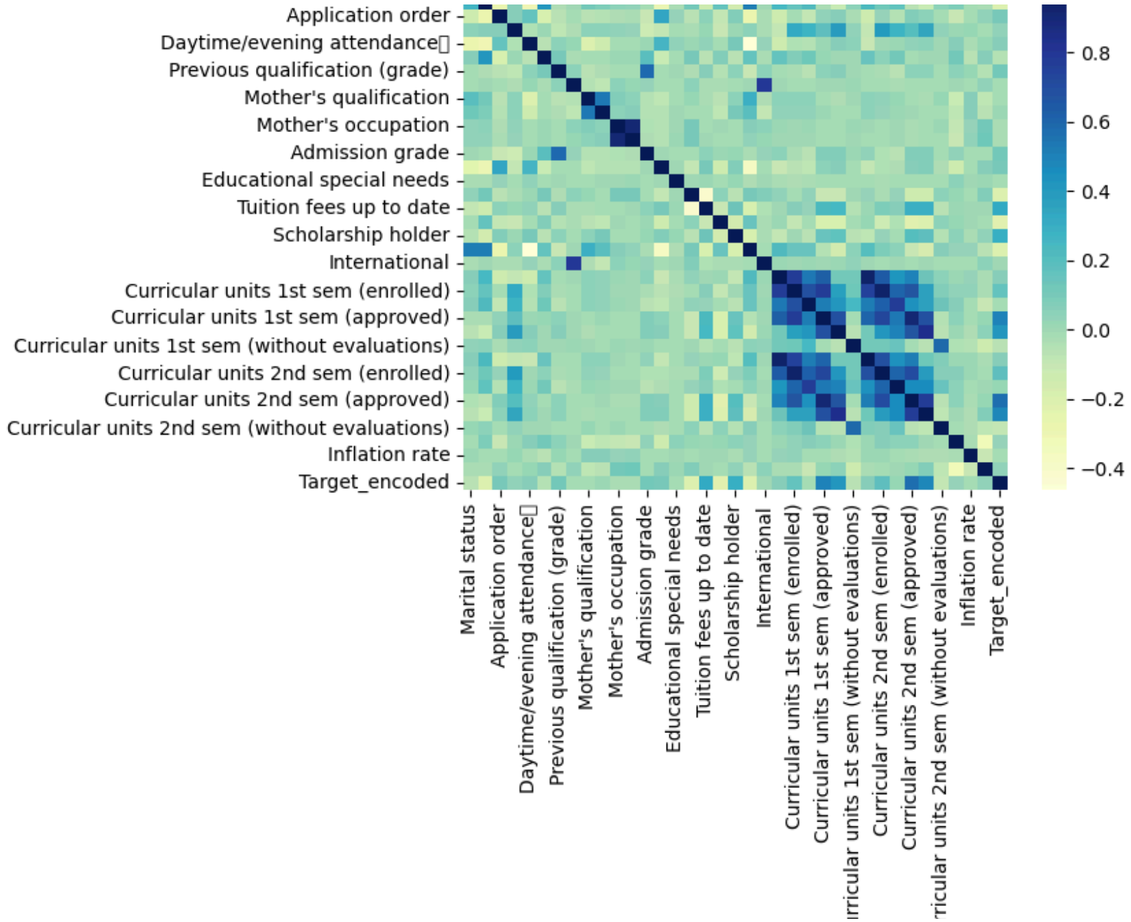
Data Cleaning:

Missing values were identified and either replaced or removed, ensuring data integrity.

Outliers:

Outliers in specific numerical features, such as 'Pass Rate', were identified and removed, reducing the dataset to 4,338 rows.

Features :Correlation Matrix



socio-economic factors student's academic path

We then calculated correlations between these features and our target variable—whether a student passed or failed—and selected 16 relevant features based on a correlation threshold of 0.1.

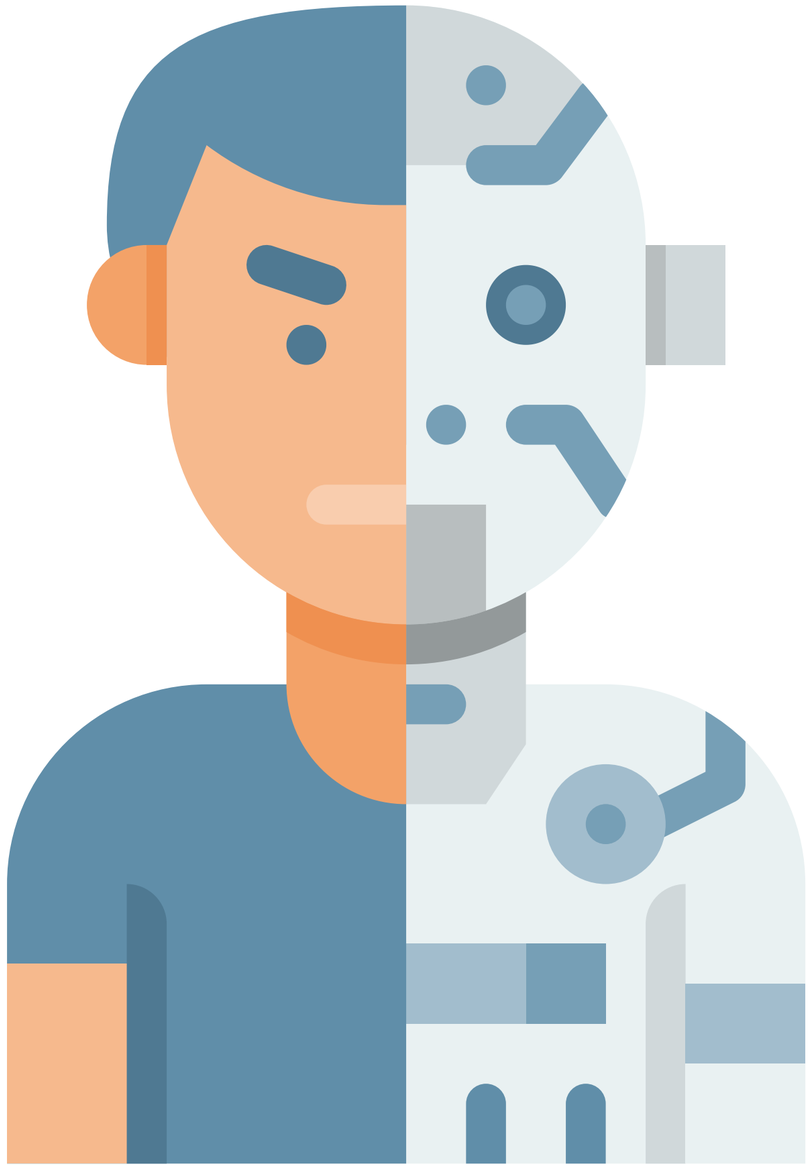
Feature Engineering and Selection

```
# Create a new feature: Average grade across both semesters
data['Average grade'] = (data['Curricular units 1st sem (grade)'] + data['Curricular units 2nd sem (grade)']) / 2

# Create new features: Pass rate for the first and second semesters
data['Pass rate 1st sem'] = data['Curricular units 1st sem (approved)'] / data['Curricular units 1st sem (enrolled)']
data['Pass rate 2nd sem'] = data['Curricular units 2nd sem (approved)'] / data['Curricular units 2nd sem (enrolled)']

# Display the first few rows to confirm new features are added
print(data[['Average grade', 'Pass rate 1st sem', 'Pass rate 2nd sem']].head())
```

```
   Average grade  Pass rate 1st sem  Pass rate 2nd sem
0       0.000000                NaN                NaN
1      13.833333           1.000000           1.000000
2       0.000000           0.000000           0.000000
3      12.914286           1.000000           0.833333
4      12.666667           0.833333           1.000000
```

New Features:

New features, such as 'Average Grades' and 'Pass Rates', were engineered to enhance the model's predictive power.



Model Building: KNN

Model:

A k-Nearest Neighbors **(KNN)** model was used as the baseline model.
Hyperparameter Tuning:
Grid Search was used to find the best hyperparameters for the KNN model.

Results:

The final KNN model achieved an accuracy of 81.34%, with the best parameters being n_neighbors=9, metric=manhattan, and weights=distance.

Model Building: Random Forest

Model:

A Random Forest model was built to compare against the KNN model.

Results:

Random Forest outperformed KNN with an accuracy of 83.64%.

```python
# Initialize the Random Forest model
rf_model = RandomForestClassifier(random_state=42)

# Train the model on the scaled training data
rf_model.fit(X_train_scaled, y_train)

# Make predictions on the test data
rf_y_pred = rf_model.predict(X_test_scaled)

# Evaluate the performance of the Random Forest model
rf_accuracy = accuracy_score(y_test, rf_y_pred)
rf_f1 = f1_score(y_test, rf_y_pred)

# Print the results
print(f"Random Forest Accuracy: {rf_accuracy * 100:.2f}%")
print(f"Random Forest F1 Score: {rf_f1:.2f}")
```

```
Random Forest Accuracy: 83.64%
Random Forest F1 Score: 0.84
```

Model Building: Gradient Boosting

Model:

Gradient Boosting was tested as an ensemble model.

Results:

The Gradient Boosting model achieved the highest accuracy of 84.10%.

```
# Initialize the Gradient Boosting model
gb_model = GradientBoostingClassifier(random_state=42)

# Train the model on the scaled training data
gb_model.fit(X_train_scaled, y_train)

# Make predictions on the test data
gb_y_pred = gb_model.predict(X_test_scaled)

# Evaluate the performance of the Gradient Boosting model
gb_accuracy = accuracy_score(y_test, gb_y_pred)
gb_f1 = f1_score(y_test, gb_y_pred)

# Print the results
print(f"Gradient Boosting Accuracy: {gb_accuracy * 100:.2f}%")
print(f"Gradient Boosting F1 Score: {gb_f1:.2f}")
```

```
Gradient Boosting Accuracy: 84.10%
Gradient Boosting F1 Score: 0.85
```

Hyperparameter Tuning: Grid Search

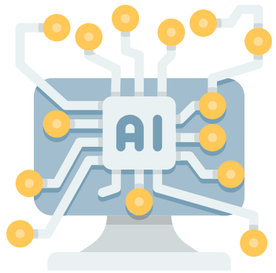
Description:

Grid Search was employed to tune the hyperparameters for both the KNN and Random Forest models.

Results:

Optimized parameters for KNN and Random Forest improved the model accuracy.

Model Comparison

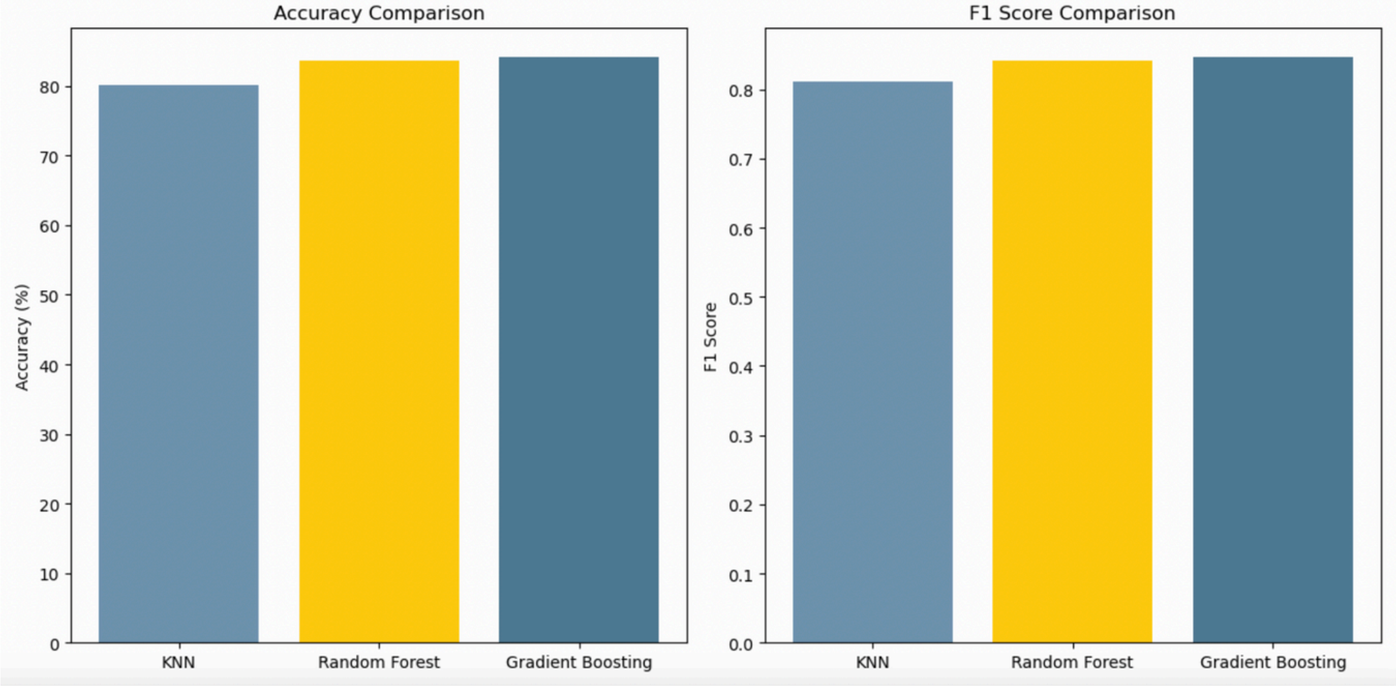


Comparison

Comparing KNN, Random Forest, and Gradient Boosting models using accuracy and F1 score showed that Gradient Boosting performed the best with an accuracy of 84.01%.

Visualization:

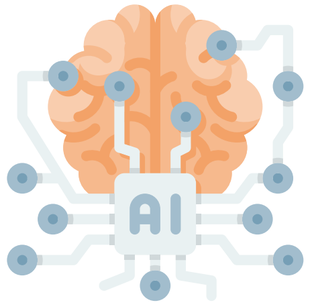
(Display bar charts of model accuracy and F1 score here.)



Real-World Application and Impact

Application:

- Early Identification
- Targeted Support
- Resource Efficiency
- Improved Graduation Rates



Impact:

- Lower dropout rates
- Higher graduation rates
- Cost savings for students and universities
- Enhanced student well-being

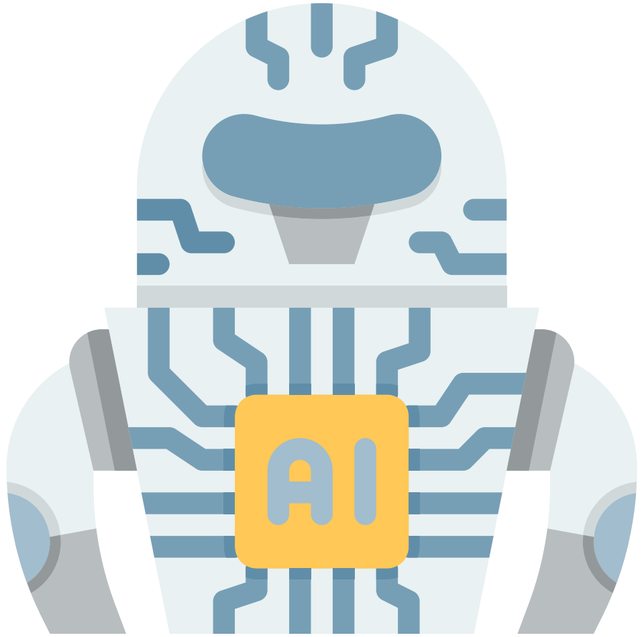
Ethical Considerations:

- Data Privacy: Protect sensitive student information.
- Bias and Fairness: Ensure predictions don't reflect or perpetuate societal biases.
- Autonomy: Avoid stigmatization of students labeled "at-risk."

Challenges and Learnings

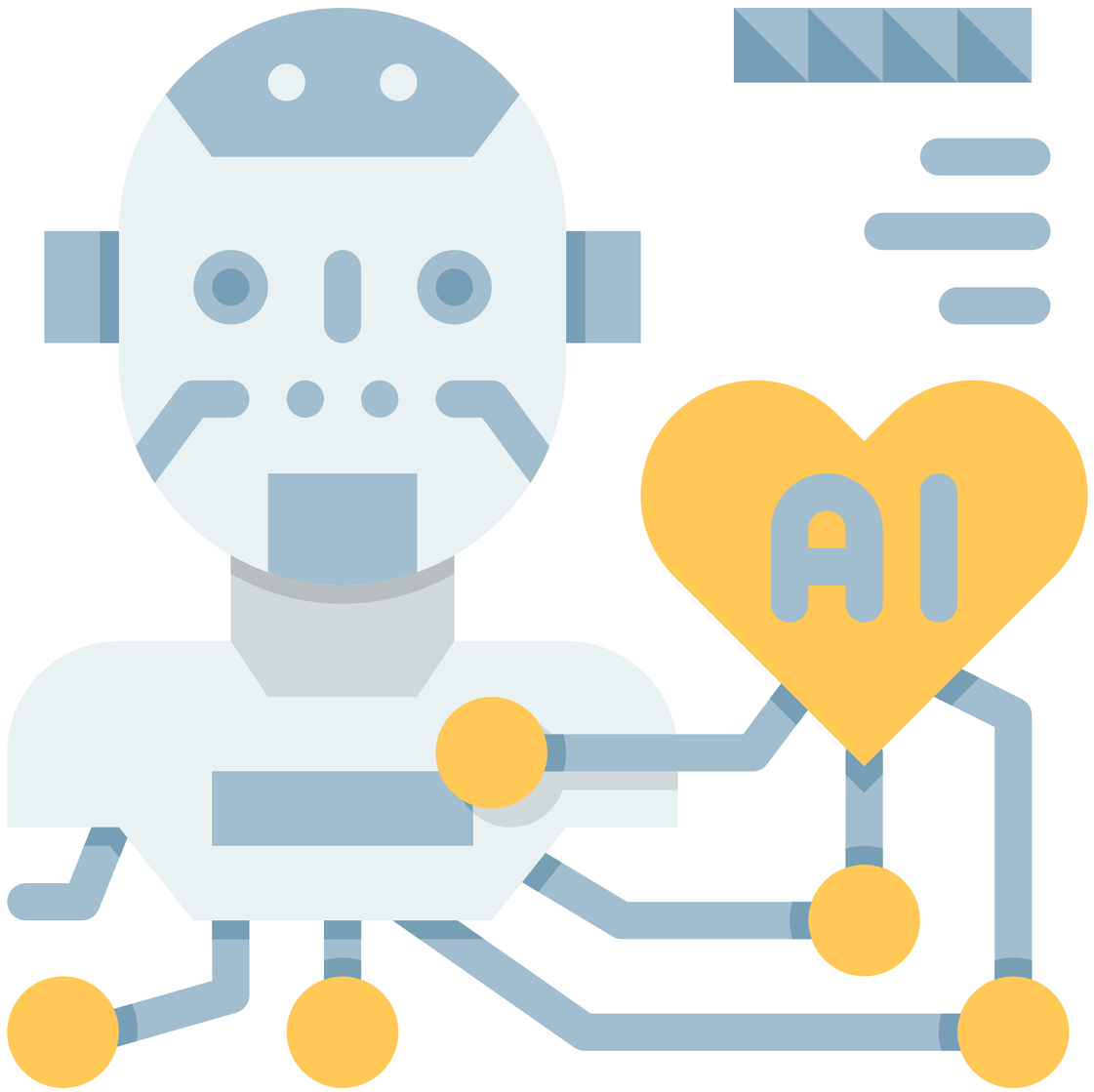
Challenges:

Handling missing values and outliers in the dataset.
Identifying the best model and optimizing hyperparameters required significant experimentation.



Learnings:

Feature engineering and selection are critical to the success of a machine learning model.
Choosing the right model and hyperparameters significantly impacts model performance.



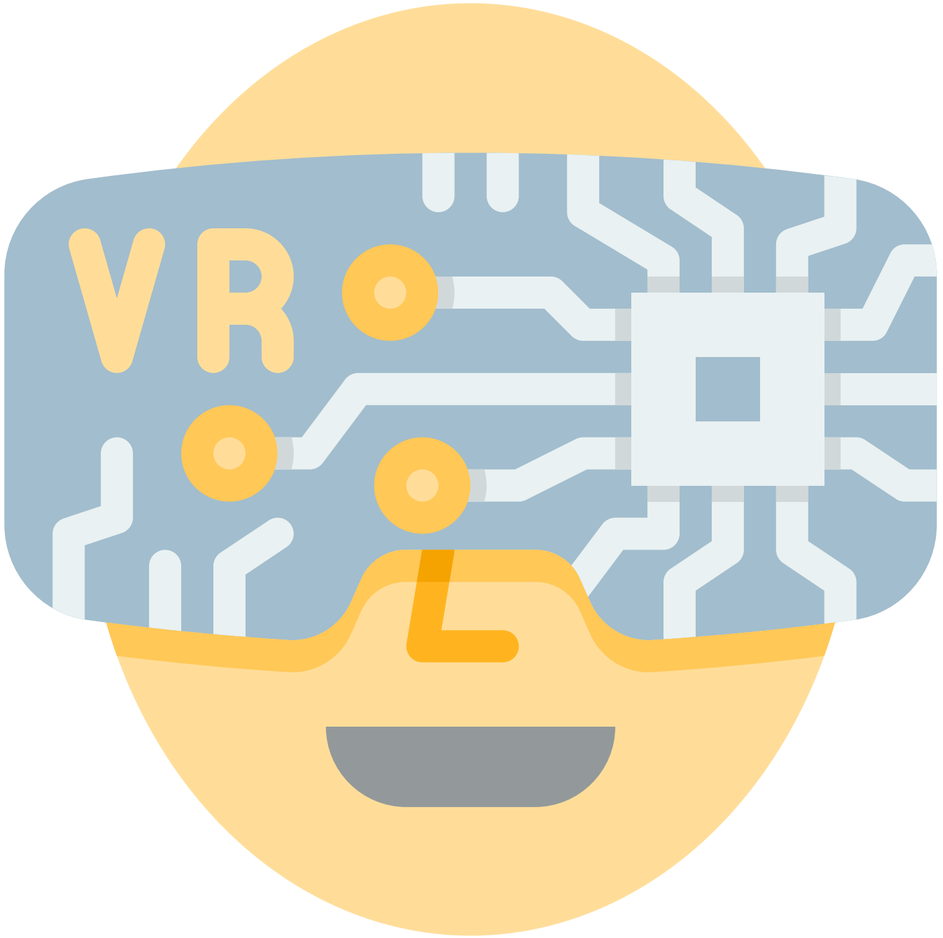
Future Work and Improvements

Future Work:

- Incorporate Post-enrollment and Behavioral Data
- Advanced Model Architectures
- Continuous Monitoring and Updating of Data

Improvements:

- Developing a real-time early warning system to provide predictions for students in real-time.



Predicting Student Success Using Machine Learning

Team Members:

Nicole, Mehdi, Tania, Houleye

Thank you for your attention. Any questions?