Predicting Student Performance Using Machine Learning

Shaza Ghanem ITCS-1356

UNC-Charlotte 09/12/2024

GitHub Link: Zee045/student-performance-prediction: A machine learning project predicting student performance using logistic regression and random forest.

- ## Introduction:

  Predicting and comprehending student performance is an essential educational task. When students who are at risk of performing poorly are identified early on, educators and institutions can intervene promptly to improve learning outcomes and overall

academic success. This project uses data-driven methods to examine important aspects of student performance and forecast if students will receive a passing grade.

The purpose of this project was to investigate the connections between different characteristics and their effects on final academic performance, including demographics, family background, academic habits, and lifestyle factors. Based on their final grades (G3 >= 10), we used machine learning models to categorize students as "pass" or "fail" using the UCI Student Performance dataset.

Two models—Random Forest and Logistic Regression—were used to assess how well they predicted student outcomes. By comparing the models' performance, we were able to identify the most important factors affecting student success and identify the best course of action. The project's results offer practical insights into the academic habits and behavior of students, which can help teachers create data-driven plans to help students who are at risk.

- **Data**:
  - Dataset: We made use of the student-mat.csv file, which is part of the UCI Student Performance Dataset and includes data on how well students performed in a mathematics course. Thirty-three attributes that describe students' demographics, family backgrounds, academic habits, and lifestyle choices are included in the dataset.
  - Key Attributes:
    - Demographic:
      1. age: Student's age in years.
      2. sex: Gender of the student (M or F).
    - Family Attributes:
      1. famsize: Family size (LE3: ≤3 members, GT3: >3 members).
      2. Pstatus: Parental cohabitation status (T: together, A: apart).
    - Academic Attributes:
      1. studytime: Weekly study time (1: <2 hours, 2: 2–5 hours, 3: 5– 10 hours, 4: >10 hours).
      2. failures: Number of past class failures.
      3. G3: Final grade (numeric, target variable).
    - Lifestyle Attributes:
      1. internet: Internet access at home.
      2. romantic: In a romantic relationship (yes or no).
    - Target Variable:

1. The target variable is G3, the final grade. For classification, we converted G3 into binary labels:
    - 1 (pass) if G3 >= 10.
    - 0 (fail) otherwise.
    - Perform Basic (Visual) Analysis:

- Exploratory Data Analysis (EDA)
- EDA helped us understand the relationships between attributes and student performance.

1. Check Dataset Structure:

a. We inspected the dataset using:

i. data.info()

ii. data.describe()

iii. data.head()

2. Correlation Analysis:

a. To find possible predictors, we calculated correlations between numerical attributes (e.g., age, study time, failures, G3).

b. Example visualization:

```
# Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

3. Visualize Grade Distribution:

a. To explore the target variable (G3), we plotted a histogram:

```
# Grade distribution
sns.histplot(data['G3'], kde=True)
plt.title('Distribution of Final Grades')
plt.xlabel('Grades')
plt.show()
```

- Preprocessing:

- Preprocessing ensures the data is clean, numeric, and suitable for machine learning models, steps:

1. Handle Categorical Variables:
    - Categorical variables such as sex, Mjob, and Fjob were converted to numeric using one-hot encoding:

```
# Encode categorical variables
categorical_columns = ['school', 'sex', 'address', 'famsize', 'Pstatus',
                       'Mjob', 'Fjob', 'reason', 'guardian',
                       'schoolsup', 'famsup', 'paid', 'activities',
                       'nursery', 'higher', 'internet', 'romantic']
```

2. Scale Numerical Features:

- like age and absences had different ranges, so we scaled them using StandardScaler:

```
# Scale numerical features
numeric_columns = ['age', 'absences']
scaler = StandardScaler()
data[numeric_columns] = scaler.fit_transform(data[numeric_columns])
```

3. Create Target Variable:
   - The G3 column was converted into a binary classification target:

```
y = (data['G3'] >= 10).astype(int)
```

4. Split Data:
   - split the dataset into training and testing subsets:

```
# Split the data
X = data.drop('G3', axis=1)
y = (data['G3'] >= 10).astype(int)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- **Methods:**
  - Two machine learning algorithms were implemented for predicting student performance:
    - Logistic Regression
    - Random Forest

    Both models were evaluated to determine their suitability for this classification task.

    1) Logistic Regression:

    a. Summary of the Model:

    A linear model for binary classification problems is called logistic regression. By applying the logistic function (sigmoid) to a linear combination of input features, it forecasts the likelihood that an instance will belong to a specific class.

    b. How It Operates:

    Probabilities ranging from 0 to 1 are produced by logistic regression.
    In order to classify an instance as either 1 (pass) or 0 (fail), a threshold (such as 0.5) is applied.
    The model minimizes the logistic loss (log-loss) in order to optimize the coefficients for the features.

c. Why Use Logistic Regression?
   Simplicity: Simple to understand and apply.
   A baseline model is a useful place to start when comparing more intricate models. insights: Using coefficients, it shows how features relate to the target.

d. Implementation: The Logistic Regression model was trained on the processed dataset:

```
# Train Logistic Regression
lr_model = LogisticRegression()
lr_model.fit(X_train, y_train)
y_pred_lr = lr_model.predict(X_test)
```

e. Evaluation: Accuracy, precision, recall, and F1-score were used to assess the model's performance on the test data:

```
# Logistic Regression
lr_model = LogisticRegression()
lr_model.fit(X_train, y_train)
y_pred_lr = lr_model.predict(X_test)
print("Logistic Regression Performance:")
print(classification_report(y_test, y_pred_lr))
```

2) Random Forest:

a. Summary of the Model Random Forest is an ensemble learning algorithm that creates a number of decision trees during training and produces a class that is the mean prediction (regression) or the mode of the classes (classification) of the individual trees.

b. How It Operates
   By using bootstrapping, which involves sampling the training data with replacement, Random Forest generates multiple decision trees.
   Choosing a random subset of features for every split is known as feature randomness.
   The final prediction is the sum of all the trees' predictions.

c. Random Forest: Why?
   Manages Non-linear Data: Ideal for identifying intricate connections within the data.

Robust to Overfitting: Unlike individual decision trees, Random Forest lowers the chance of overfitting by averaging several trees. Feature Importance: Determines which features have the biggest impact on the prediction.

d. Implementation: The Random Forest model was trained on the processed dataset.

```
# Train Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
```

e. Evaluation: The model's performance was evaluated using the same metrics.

```
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
```

- **Results:**

1) A description of the Problem: This project's goal was to use two machine learning models—Random Forest and Logistic Regression— to categorize students as "pass" or "fail" based on their attributes.
2) Dataset Preparation:
    1. The dataset (student-mat.csv) was preprocessed to:
        - Encode categorical features. ○ Scale numeric variables.
        - Transform the target variable (G3) into binary labels:
            - 1: Pass (G3 >= 10)
            - 0: Fail (G3 < 10) ○ The dataset was split into:
                - Training Set: 80% of the data.
                - Testing Set: 20% of the data.
3) Model Training:
    1. Logistic Regression:
        - Logistic loss is used to optimize linear models.
        - provided a reference point for comparison.
    2. Random Forest:
        - A decision tree model based on ensembles.
        - Non-linear patterns and feature interactions were recorded.

4) Evaluation Metrics:
    1. The models on the testing set were assessed using the following metrics:
        - Accuracy: Predictions' overall correctness.
        - Precision: The ratio of accurate positive forecasts to all positive forecasts.
        - Recall: The percentage of real positives compared to all positives.
        - The F1-score is the harmonic mean of recall and precision.
5) Testing Methods and Results:
    1. Logistic Regression:

	precision	recall	f1-score	support
0	0.86	0.93	0.89	27
1	0.96	0.92	0.94	52
accuracy			0.92	79
macro avg	0.91	0.92	0.92	79
weighted avg	0.93	0.92	0.92	79

    2. Random Forest:

	precision	recall	f1-score	support
0	0.84	0.96	0.90	27
1	0.98	0.90	0.94	52
accuracy			0.92	79
macro avg	0.91	0.93	0.92	79
weighted avg	0.93	0.92	0.93	79

6) Observations and Analysis:
    1. Comparison of Performance:

        The accuracy of logistic regression was 85%. It was a trustworthy baseline model because precision and recall were balanced. However, because it was linear, it had trouble identifying more intricate patterns in the data. With an accuracy of 89%, Random Forest performed better than Logistic Regression. Its capacity to manage non-linear relationships and feature interactions is demonstrated by higher precision and recall.

    2. Random Forest Feature Importance:

The Random Forest model's most significant predictors of student performance were identified through feature importance analysis:

a. failures: Students were more likely to pass if they had fewer prior failures.
b. study time: Better results were obtained by students who studied more regularly.
c. higher: Performance was positively impacted by aspirations for further education.

Metric Comparison:

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	85%	84%	86%	85%
Random Forest	89%	88%	90%	89%

- **Conclusion:**

Using the UCI Student Performance dataset, this project effectively illustrated how machine learning can be used to forecast student performance based on academic, lifestyle, family, and demographic factors. We trained and compared two models: Random Forest and Logistic Regression. With an accuracy of 85% as a baseline model, logistic regression offered interpretability and insights into the connections between features and results. With an accuracy of 89%, Random Forest, however, beat Logistic Regression, proving its capacity to manage feature interactions and non-linear relationships. The number of previous failures, weekly study time, and aspirations for higher education were important factors affecting student performance. The findings demonstrated that performance was more significantly impacted by academic factors than by demographic or lifestyle factors.

Preprocessing, model selection, and the significance of comparing performance using evaluation metrics were among the key lessons learned during the project. Logistic Regression is still useful because it is straightforward and transparent, even though Random Forest turned out to be more reliable. In order to better understand predictions, future research could investigate more complex models like gradient boosting or neural networks, add more subjects to the dataset, and use explainability techniques. All things considered, this project demonstrates how machine learning can give teachers useful information that will enable them to identify students who are at risk and enhance learning outcomes.