Junyi Academy Dataset Analysis & Prediction

Shri Prakash Yadav
Ilenia Santagata
Hardware & Software for Big Data Mod B

Abstract:

The importance of online learning has surged, especially highlighted by the COVID-19 pandemic, which has demonstrated its necessity for equitable quality education when physical campuses close. The Junyi Academy Foundation, a non-profit in Taiwan dedicated to providing quality education through technology, has released a dataset of over 16 million exercise attempts by 72,000 students from August 2018 to July 2019. This dataset aims to facilitate research into personalized learning experiences and promote interdisciplinary contributions to the future of online education.

1.Problem Statement:

In this project we have been provided with three different dataset from Junyi academy. On top of these datasets, we have to perform an EDA and train a classification model to predict whether any user is expected to solve a given problem based on his previous learnings.

2. Introduction

The Junyi academy dataset consists of three files InfoUser.csv , InfoContent.csv and LogProblem.csv. The description of the datasets are below:

Log_Problem.csv recorded 16,217,311 problem attempts of 72,630 selected students for a year from 2018/08 to 2019/07.

Info_Content.csv describes the metadata of the exercises, each exercise is a basic unit of learning consisted of many problems.

Info_UserData.csv describes the metadata of the selected registered students in Junyi Academy.

3. Steps performed in the analysis

3.1 Exploratory Data Analysis

After loading the dataset, we visualized the data using various types of plots, such as bar plots, line plots, pie charts, and Probability Density Functions (PDFs). These visualizations allowed us to analyze the dataset comprehensively and uncover hidden patterns and insights.

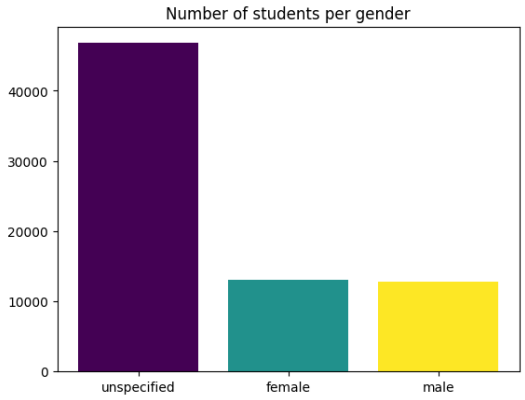
Subsequently, we merged the three separate datasets into a single cohesive dataset. This merging process enabled us to perform more complex analyses, such as tracing the learning paths of individual users. By examining this unified dataset, we could identify intricate patterns and relationships that were not apparent when the datasets were analyzed separately.

During our exploratory data analysis (EDA), we also plotted the distributions of all numerical variables. This step was crucial for understanding how these variables varied across the dataset. It helped us identify potential outliers, skewness, and other characteristics of the data distributions that could inform our subsequent analyses.

Furthermore, we created a correlation matrix to examine the relationships between different features and our target variable. The correlation matrix revealed which features had strong correlations with the target variable and which had weaker or negligible correlations. This information was vital for feature selection and understanding the underlying structure of our data, guiding us in building more accurate and effective predictive models.

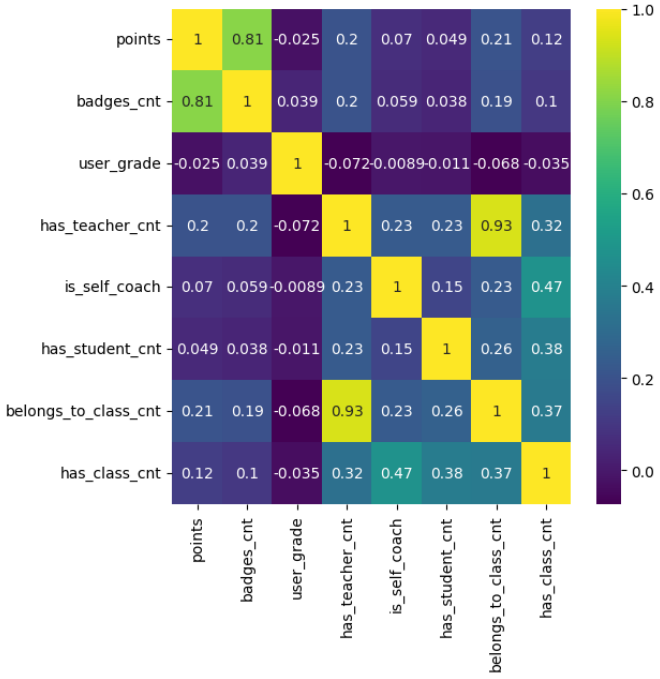
Some of the EDA plots are added below:

- Number of users from different genders.



We had `39853 null values` for the gender variable, meaning that many students prefer to not disclose this detail, therefore we imputed the rows where gender was ‘null’ with unspecified.

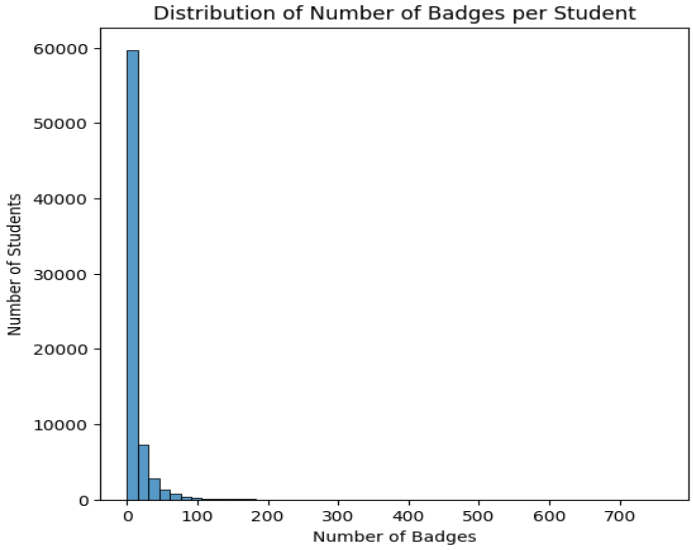
- The correlation matrix shows how the features are correlated with each other



The main insights that we can obtain from the correlation matrix are:

1) There is a strong positive correlation between `points` and `badges_cnt`, indicating that users with higher points tend to have more badges. This relationship suggests that earning badges is closely tied to accumulating points.
2) There is an extremely strong positive correlation between `belongs_to_class_cnt` and `has_teacher_cnt`, indicating that users who belong to more classes are also those who are more frequently counted as having teachers. This could imply a structured learning environment where being part of a class is associated with having a teacher.

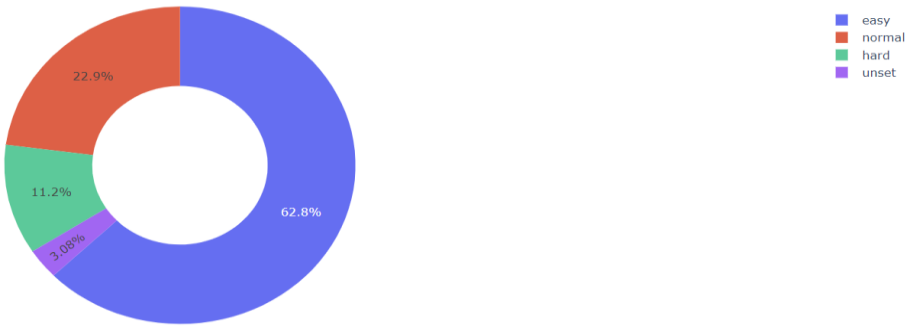
- The plot below shows that very few number of students has high number of badges.



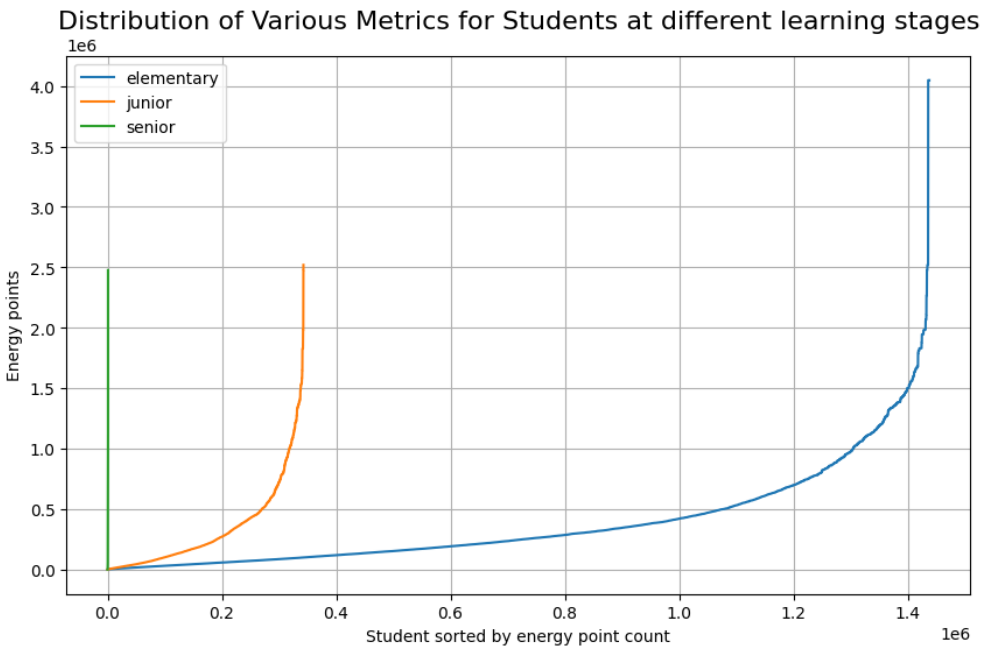
A large proportion of students have earned between `0` and `20 badges`. The bar representing students with very few badges (especially around 0) is the tallest, indicating that many students have earned only a small number of badges.

There is a long tail extending towards the right, indicating that a small number of students have earned a significantly higher number of badges, some even exceeding 100 badges. This suggests that while most students earn a few badges, there are outliers who are highly engaged or have achieved a lot.

- The plot below shows that most of the content in the `df_InfoContent` dataset belongs to the `easy category`.



- The graph below compares the energy points of students across three educational stages: elementary, junior, and senior.



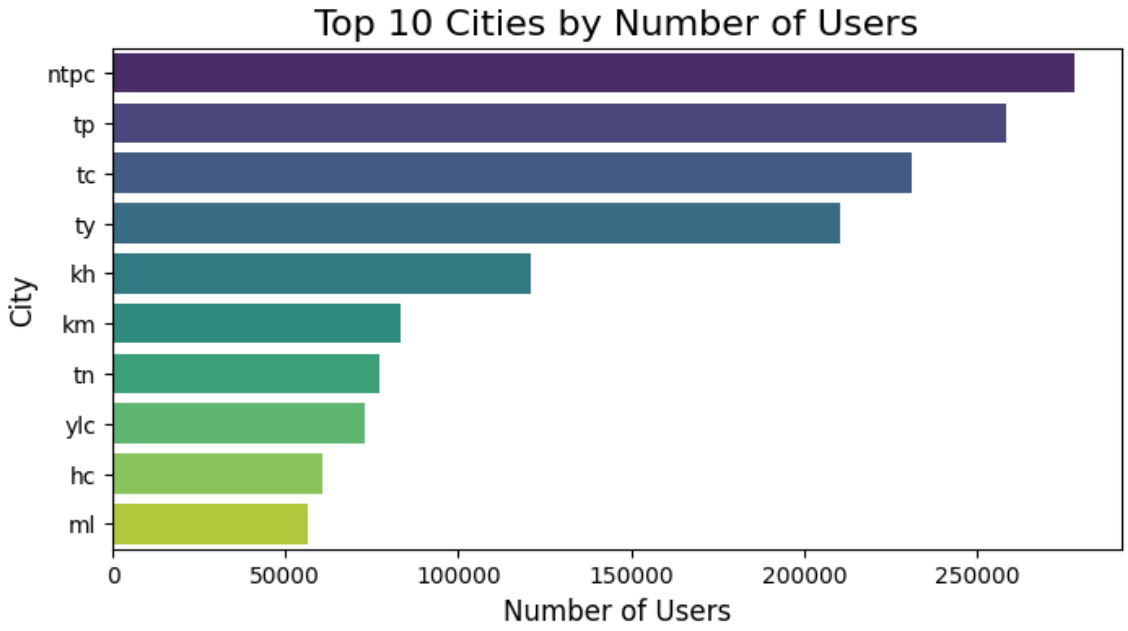
Insights that we can gather for the graph:

`Elementary stage:` the energy points for elementary students show a gradual increase across the student population, with a significant spike towards the end. This indicates that while most elementary students accumulate a moderate number of points, a small subset accumulates a very high number, suggesting a few high achievers or highly engaged students in this group.

`Junior stage:` junior students exhibit a steep increase in energy points early on, followed by a plateau (constant frequency over time). This suggests that junior students quickly reach a high level of energy points, which then stabilizes. This could indicate activities that rapidly engage students to a certain level of achievement or participation.

1) `Senior stage:` the energy points for senior students start very low and remain flat across most of the student population, with a very sharp rise at the very end. This pattern suggests that the majority of senior students do not accumulate many points, but a very small number of students accumulate an extremely high number of points, possibly indicating a group of students excelling in a specific area.

- The plot below shows the user cities ordered by the number of users.



3.2 Encoding Categorical Features

In our dataset, there were several categorical features that required transformation for analysis and modeling. We employed one-hot encoding to convert these categorical variables into a numerical format suitable for visualization and training purposes. This method creates new binary columns for each category, allowing us to represent the presence or absence of each category with 1s and 0s.

One-hot encoding was crucial for several reasons:

1. Visualization and correlation analysis: for instance, it was useful to create correlation matrices, which revealed hidden patterns and insights.
2. Model Training: Machine learning algorithms typically require numerical input data. One-hot encoding ensured that our categorical features could be integrated into various models, improving their performance and accuracy. This preprocessing step was essential to use the decision tree.

3.3 Handling Null Values

Our dataset contained numerous null values that needed to be addressed to ensure accurate analysis and model performance. We implemented the following strategy to handle these missing values:

1) **Dropping Features with High Null Percentage:**

   We identified features with a high percentage of null values. Specifically, if a feature had more than 70% of its values missing, we decided to drop it from the dataset. This was the case for the features `is_downgrade` and `is_upgrade`. Dropping these features helped streamline the dataset by removing columns that had insufficient data to provide meaningful insights or contribute to model training.

2) **Imputing Missing Values for Remaining Features:**

   For the remaining features with missing values, we imputed the null values with appropriate substitutes.

This preprocessing step was crucial for maintaining the integrity of our analysis and ensuring that our models were trained on accurate and representative data.

3.4 Upsampling

After merging the three datasets, we observed a significant imbalance in our target variable `is_correct`. This imbalance in fact affected the performance of our machine learning model (decision tree), leading to biased predictions towards the majority class. To address this issue, we employed the upsampling technique to balance our dataset.

1) **Identifying Imbalance:**

   We first identified the extent of the imbalance by examining the distribution of the target variable. It was clear that one class was significantly underrepresented compared to the other, in particular, we had a lower number of cases for is_correct = false compared to the true value of the variable.

2) **Upsampling the Minority Class:**

   To rectify this imbalance, we used the upsampling technique. This involved increasing the number of instances in the minority class by randomly sampling with replacement. By duplicating existing instances of the minority class, we generated a dataset where the target classes were more evenly distributed.

3.5 Fitting different models

In our analysis, we utilized two classification models: Logistic Regression and Decision Tree, to classify and predict outcomes based on our dataset.

Logistic Regression is a fundamental statistical model that predicts the probability of a binary outcome based on input variables. It's widely used due to its simplicity and interpretability.

Decision Trees, on the other hand, offer a more intuitive approach to classification. They recursively partition the data into subsets based on features, creating a tree-like structure where each internal node represents a feature, each branch represents a decision rule, and each leaf node represents a class label or outcome.

Initially, we considered expanding our model selection to include more complex algorithms. However, during our evaluation phase, the Decision Tree model consistently demonstrated exceptional performance.

In alignment with the principles of model `parsimony` and `efficiency`, we decided against introducing additional complexity beyond the Decision Tree.

3.6 Feature Importance

Subsequently, we employed the Decision Tree model, which exhibited superior performance, to identify the most influential features in predicting our target variable.

Decision Trees are particularly advantageous for feature importance analysis due to their inherent ability to partition data based on feature importance during training. This

methodically segregates features according to their predictive power, making it straightforward to discern which variables significantly influence the model's predictions.

4. Algorithms

4.1 Logistic Regression

In our project, Logistic Regression was used in predicting whether students would answer exercises correctly or not, using a binary target variable (`is_correct`).

Logistic Regression is a statistical model that predicts the probability of a binary outcome based on input variables. In our case, the model estimated the probability that a student's answer to an exercise would be correct. This probability is modeled using the logistic function, which transforms the linear combination of input features into a probability score.

Formula:

- The logistic regression model uses the logistic function (sigmoid function) to transform the linear combination of input features into a probability score between 0 and 1.

$$P = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\beta_3 X_3+\ldots+\beta_n X_n)}}$$

  - β0,β1,…,βp are coefficients estimated from the training data, representing the impact of each feature on the probability of Y=1.
  - e is the base of the natural logarithm (approximately equal to 2.718).
  - X1,X2,…,Xp are the input features of the model.

4.2 Decision Tree Classifier

A Decision Tree Classifier is a ML model used for both classification and regression tasks.

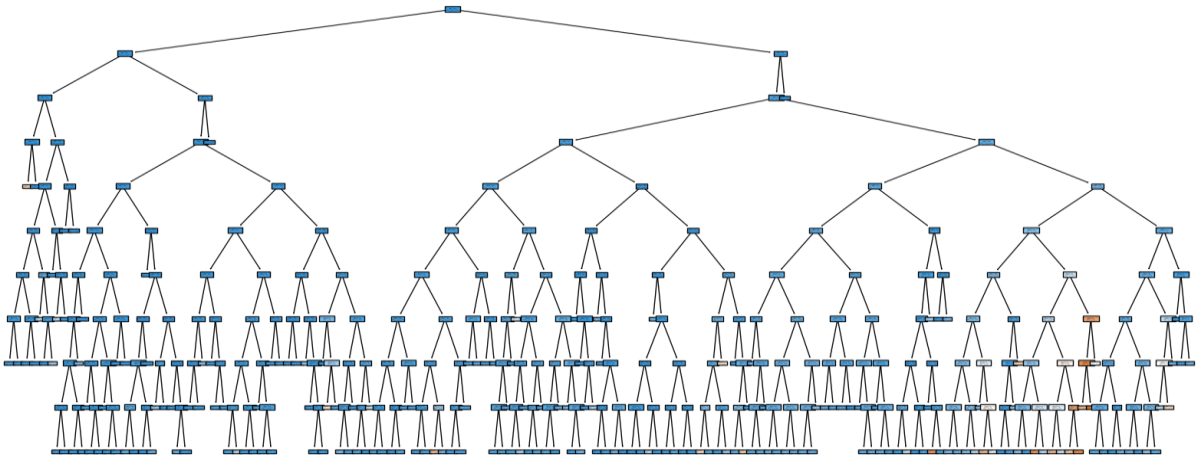
Structure: Decision trees consist of nodes, branches, and leaves. Each node represents a feature or attribute, each branch represents a decision rule based on that feature, and each leaf node represents the predicted outcome.

Splitting Criteria: In a decision tree classifier, the dataset is split at each node based on the feature that maximizes the information gain or reduces the impurity of the subsets.

Information gain measures the reduction in entropy or Gini impurity after splitting the dataset based on a particular feature.

Information Gain: Information gain quantifies how much a particular feature contributes to improving the purity of the subsets. Entropy is a measure of randomness or uncertainty in the dataset, while Gini impurity measures the probability of incorrectly classifying a randomly chosen element in the dataset if it were randomly labeled according to the class distribution.

- Decision tree structure for our dataset.



5. Model performance Metrics

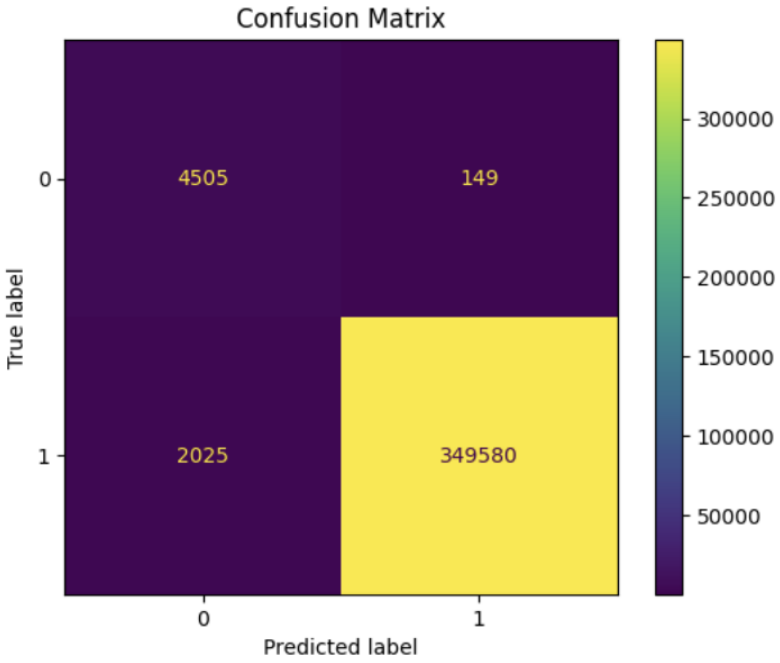
5.1 Confusion Matrix

The confusion matrix is a fundamental tool for evaluating the performance of a classification model by summarizing its predictions against the actual classes of the data.

It is a table that has rows representing the actual classes and columns representing the predicted classes. Each cell in the matrix represents the count (or proportion) of instances that fall into a particular combination of predicted and actual classes:

- **True Positives**: Instances where the model correctly predicts the positive class (predicted = Positive, actual = Positive).
- **True Negatives**: Instances where the model correctly predicts the negative class (predicted = Negative, actual = Negative).
- **False Positives**: Instances where the model incorrectly predicts the positive class (predicted = Positive, actual = Negative).
- **False Negatives**: Instances where the model incorrectly predicts the negative class (predicted = Negative, actual = Positive).

These were the results for the decision tree model:



5.2 F1 Score

The F1 Score is a single metric that combines both precision and recall into a single value, providing a balanced measure of a model's performance in binary classification tasks.

- **Precision**: Precision measures the accuracy of positive predictions. It is the ratio of true positive predictions to the total number of positive predictions made by the model.
- **Recall (Sensitivity)**: Recall measures the proportion of actual positives that were correctly identified by the model. It is the ratio of true positive predictions to the total number of actual positive instances in the data.

In our model we had an high value for the F1 Score around 0.99.

5.3 Accuracy

Accuracy is given by the number of correctly classified examples divided by the total number of classified examples. In terms of the confusion matrix, it is given by: TP+TN/TP+TN+FP+FN

Conclusion

In this project, we conducted exploratory data analysis (EDA) and trained various models on the Junyi dataset to predict outcomes effectively. After experimentation, the Decision Tree model emerged as the optimal model for our task.

Model Performance:

- **Accuracy Score:** The Decision Tree classifier achieved an impressive accuracy score of 0.99, indicating its ability to correctly classify instances.
- **F1 Score:** The F1 Score, a harmonic mean of precision and recall, also stood at 0.98. This score reflects a balanced performance in terms of both identifying positives and negatives.

Feature Importance:

- From the Decision Tree model, we identified several key features that significantly influenced predictions:
    - **total_attempt_cnt:** The total number of attempts made by users.
    - **is_hint_used:** The total number of hints used by users.
    - **points the user has:** The points accumulated by users during interactions.
    - **problem_number:** The problem user is going to solve.