Discovering Enrollment Patterns for Course Planning

Zubaer Chowdhury, Tasmia Alamgir, Matthew Goldman

1. Problem Statement

With many factors impacting the effectiveness of advising, college students are encouraged to plan out their semesterly and four-year plans by themselves. Personalizing an academic plan for the first time in a new educational institution while fulfilling the degree requirements, however, can be a struggle. Figuring out how to prioritize which classes to take in this situation can be a frustrating and confusing process. Our team identifies a need in an interpretation of the enrollment data that helps college students identify trends in course enrollment by sampling and analyzing a university's student information system.

Thirty percent of U.S. college students change their college major within three years of undergraduate coursework (Dept. Of Education, 2018). Some universities, like the University of Rochester, often encourage their students to broaden their curriculum with interdisciplinary or cross-college studies through open curriculum and the cluster system; about 32% of undergraduates ended up pursuing a double major in 2020 (Mccaslin, 2024). Changes such as these may introduce issues in the student's expected graduation date. According to a 2022 report released by the National Center for Education Statistics, only about 46 percent of U.S. college students graduate with a bachelor's degree within four years. While there are many factors affecting degree progress, course registration can be a source of stress for many college students, especially those who are worried about graduating on time.

This project addresses the challenge of predicting **how fast seats fill up in courses** during registration periods. Understanding enrollment velocity is critical for:

- Students making informed registration decisions
- Academic advisors guiding course selection timing
- University administrators planning section offerings and capacity allocation

This is formulated as a **Time Series Forecasting/Regression problem** where:

- **Target variable:** Fill percentage at time *t* (percentage of total capacity filled)
- **Prediction type:** Continuous value representing current enrollment status
- **Core question:** Given current enrollment trends and course characteristics, what will the fill percentage be at any given point during the registration period?

The model enables both real-time monitoring of enrollment status and short-term forecasting to predict when courses will reach critical capacity thresholds.

2. Dataset Description

Source of Data

Data was collected through a Python web scraper that systematically retrieved course enrollment information from the University of Miami's registration portal. The scraper operated at regular intervals throughout the registration period to capture temporal enrollment dynamics.

Type of Data

The dataset comprises **1,000,000 time-series observations** for Spring 2025 semester courses, stored in comma-separated values (CSV) format with **162 columns** capturing comprehensive course and enrollment information.

The column attributes in the CSV file exported from MongoDB are:

1. **_id**
   A unique alphanumeric ID assigned to each BSON file in MongoDB
2. **name**
   The name of the course being offered.
3. **subjectName**
   The subject group name for the course (e.g. Accounting Business Administration)
4. **subjectCode**
   The course subject group name code (e.g. ACC)
5. **catalogNumber**
   The course code number assigned postfix of subjectCode (e.g. 211)
6. **academicCareer**
   Listing label as either Undergraduate or Graduate category for a course
7. **semester**
   The semester the course is being offered at (e.g. Spring)
8. **year**
   The year the course is being offered on
9. **sectionType**
   Label for whether it is a Lecture section or Lab section

10. **sectionCode**
    A 4 character alphanumeric code assigned for a particular day-time combination a course uses (e.g. *1U* for Mon-Thur 9:00AM – 10:15AM)
11. **classNumber**
    An unique 5 digit number assigned to each listing for courses. (e.g. *8429* for ACC211 on Mon-Thur 9:00AM – 10:15AM)
12. **session**
    A label to identify the type of duration a course runs on (e.g. *Regular Academic* for Fall and Spring courses)
13. **days[0-6]**
    Stores the day names in a week a course takes place (e.g. Monday)
14. **timeStart**
    Stores the time when the class starts in each day is takes place in date-time format (e.g. 1900-01-01T18:35:00.000Z)
15. **timeEnd**
    Stores the time when the class starts in each day is takes place in date-time format (e.g. 1900-01-01T21:20:00.000Z)
16. **classroom**
    The name of the classroom where the class takes place or either the label “Online” for online class or “Arranged Arranged” if the classrooms/place is dynamic.
17. **instructor[0-13]**
    Name(s) of the instructor teaching a course
18. **startDate**
    The day a course begins in date-time format (e.g. 2025-01-13T00:00:00.000Z)
19. **endDate**
    The day a course ends in date-time format (e.g. 2025-04-28T00:00:00.000Z)
20. **capacity**
    The numeric classroom capacity for the course
21. **waitlistCapacity**
    The numeric classroom waitlist capacity for the course
22. **multipleMeetings**
    A Boolean label indicating whether a course section has an additional set of day-time during the semester. The course can run in one day-time combination for half the semester and later change to a different combination
23. **dateTimeRetrieved**
    Stores the date-time when the data in the row was collected
24. **days[0-8][0-6]**
    Additional fields to store days for courses which has *Multiple Meetings* timings in a semester

25. **timeStart[0-8]**
    Additional fields to store the start date-time for a course which has *Multiple Meetings* timings in a semester
26. **timeEnd[0-8]**
    Additional fields to store the end date-time for a course which has *Multiple Meetings* timings in a semester
27. **classroom[0-8]**
    Additional fields to store the classroom location for a course which has *Multiple Meetings* timings in a semester
28. **instructor[0-8][0-5]**
    Additional fields to store the instructor name(s) for a course which has *Multiple Meetings* timings in a semester
29. **startDate[0-8]**
    Additional fields to store the start date for a new day and time combination for a course which has *Multiple Meetings* timings in a semester
30. **endDate[0-8]**
    Additional fields to store the end date date-time for a new day and time combination for a course which has *Multiple Meetings* timings in a semester
31. **reservedSeatsAvailable**
    Number of reserved seats available from reserved seats capacity for graduate students or honors students
32. **reservedSeatsCapacity**
    Total number of seats reserved from total course capacity for graduate students or honors students
33. **topic[0-4]**
    Unused buffer columns which remains as artifacts from the web scraper's initial implementation

Key Features/Attributes

Raw Features:

- **Time-series enrollment data:** `seatsAvailable`, `capacity`, `waitlistCapacity`, `waitlistAvailable`
- **Timestamps:** `dateTimeRetrieved` capturing observation time at regular intervals
- **Course identifiers:** `classNumber` (unique 5-digit code), `subjectCode`, `catalogNumber`, `name`
- **Academic classification:** `semester`, `year`, `academicCareer` (Undergraduate/Graduate)

- **Course logistics:** `sectionType` (Lecture/Lab), `sectionCode`, `session`, `days`, `timeStart`, `timeEnd`, `classroom`
- **Instructional information:** `instructor` fields
- **Enrollment status:** `status` (Open/Closed), `reservedSeatsAvailable`

Engineered Features:

- **Temporal progression:** Days/hours since registration opened, observation sequence
- **Enrollment metrics:** Current fill percentage, seats taken, seats remaining percentage
- **Velocity indicators:** Fill rate per hour/day, rolling average velocity, fill acceleration
- **Temporal patterns:** Day of week, hour of day, weekend/weekday flags, cyclical encodings (sine/cosine transformations)
- **Course characteristics:** Subject encoding, course level, early morning/evening class indicators, online/in-person format
- **Waitlist dynamics:** Waitlist ratio, waitlist availability indicator

These engineered features capture the **rate of change** in enrollment, which is critical for understanding "how fast" seats fill alongside contextual factors that influence enrollment behavior.

Data Size and Format

- **Format:** Comma-separated values (CSV)
- **Volume:** ~1,000,000 rows (time-series observations)
- **Dimensions:** 162 original columns plus engineered features
- **Temporal coverage:** Complete Spring 2025 registration period with multiple observations per course

3. Algorithm Choices/Ideas

(a) Gradient Boosting (XGBoost/LightGBM)

Rationale: Gradient boosting methods are optimal for this problem due to:

1. Designed for structured data with mixed feature types
2. Enrollment patterns exhibit complex, non-linear dynamics that gradient boosting captures effectively through decision tree ensembles

3. Built-in feature importance metrics reveal which factors most strongly drive enrollment velocity—providing actionable insights for stakeholders
4. Optimized implementations (XGBoost/LightGBM) handle 1M+ row datasets efficiently with parallel processing
5. Inherently resistant to outliers and handles missing values gracefully without extensive preprocessing
6. Industry-standard approach for time-series regression with mixed features

Implementation Considerations:

- **Time-based validation:** Time-series split ensuring training on earlier data and testing on later periods (no data leakage)
- **Hyperparameter tuning:** Learning rate, tree depth, regularization parameters optimized via cross-validation
- **Ensemble potential:** Multiple models can be combined for improved robustness

Alternative Approaches Considered:

- **Random Forest:** Similar performance but typically slower and less accurate than gradient boosting

(b) Statistical Aggregation and Ranking

We want to identify and rank the most popular courses within specific academic levels (200, 300, 400 level) for selected subjects such as Computer Science (CSC) and Electrical & Computer Engineering (ECE). Understanding course popularity patterns is valuable for:

- Students: Making informed decisions about which elective courses to prioritize during registration
- Academic departments: Identifying high-demand courses that may require additional sections or capacity
- Curriculum planners: Understanding which courses attract the most student interest for resource allocation
- Academic advisors: Providing data-driven guidance on course selection strategies

Analytical pipeline:

1. Data Preparation and Filtering

- Extract academic level from `catalogNumber`
- Filter dataset for target subjects (CSC, ECE)

- Group observations by unique course (`classNumber`)
- Identify registration start date (earliest `dateTimeRetrieved` per course)

2. Popularity Metric Calculation

For each course, we will calculate the following metrics:

- Fill Velocity Score
- Time-to-Fill Score
- Waitlist Demand Score
- Final Fill Rate
- Early Rush Score
- Composite Popularity Score

3. Aggregation and Ranking

- Group courses by subject and academic level (e.g., 200-level CSC, 300-level ECE)
- Sort courses within each group by composite popularity score
- Generate ranked lists identifying top courses per level

4. Comparative Analysis

- We will compare popularity distributions across academic levels
- Analyze subject-specific patterns (CSC vs ECE enrollment behavior)
- Identify trends (e.g., do upper-level courses consistently fill faster?

4. Model Evaluation

Primary Regression Metrics

Mean Absolute Error (MAE):

- Measures average prediction error in percentage points
- **Interpretation:** "On average, predictions are off by X percentage points"
- MAE < 5% indicates strong performance for practical applications

Root Mean Squared Error (RMSE):

- Penalizes larger errors more heavily than MAE
- Useful for identifying if model has systematic large errors

- RMSE < 8% acceptable for enrollment forecasting

R^2 Score (Coefficient of Determination):

- Proportion of variance in fill percentage explained by the model
- **Range:** 0 to 1, where 1 is perfect prediction
- $R^2 > 0.85$ indicates the model captures enrollment dynamics well

Business-Relevant Metrics

Within-Tolerance Accuracy:

- Percentage of predictions within ±5 percentage points of actual fill rate
- **Most interpretable** for stakeholders: "90% of predictions are within 5% of actual"
- **Target:** >85% of predictions within tolerance threshold

Stage-Specific Performance:

- Evaluate error separately for:
    - **Early stage** (0-25% full): Predictions when enrollment just begins
    - **Mid stage** (25-75% full): Active enrollment period
    - **Late stage** (75-100% full): Critical capacity approaching
- Ensures model performs consistently across enrollment lifecycle

Time-to-Fill Derivation:

- Using regression predictions, calculate "days until X% capacity"
- Validates model's ability to answer survival analysis questions
- Enables alerts like "Course will fill in 2 days"

Validation Strategy

Time-Series Cross-Validation:

- Multiple train/test splits preserving temporal order
- Prevents data leakage and evaluates generalization to future semesters
- Tests model on truly unseen future enrollment patterns

Since this is descriptive analytics rather than predictive modeling, evaluation focuses on **metric validity and result reliability** instead of traditional machine learning performance metrics.

For Statistical Aggregation and Ranking

We will verify that popularity rankings align with intuitive expectations:

- Do courses known anecdotally as "high-demand" rank highly?
- Do courses that close quickly or maintain large waitlists receive high scores?
- Do elective courses rank differently than required core courses?

Evaluation criteria:

- Stakeholder review by department chairs and academic advisors
- **Target:** >70% agreement that top-ranked courses match experiential knowledge

5. References

Advisor to student ratio/caseload resources. NACADA. (n.d.). https://nacada.ksu.edu/Resources/Clearinghouse/View-Articles/Advisor-to-Student-Ratio-Caseload-Resources.aspx

Amos, D. (2024, March 28). *A practical introduction to web scraping in Python*. Real Python. https://realpython.com/python-web-scraping-practical-introduction/

Digest of Education Statistics, 2021. National Center for Education Statistics (NCES) Home Page, a part of the U.S. Department of Education. (n.d.). https://nces.ed.gov/programs/digest/d21/tables/dt21_326.10.asp

University of Miami (2021). CaneLink. https://canelink.miami.edu/