



GHULAM ISHAQ KHAN INSTITUTE OF ENGINEERING SCIENCES AND TECHNOLOGY - GIKI

Student Grading System

‘PROJECT REPORT’

DS-221 (Inferential Statistics)

Proposal Submitted to:

Sir Sajid

Faculty of Computer Sciences and Electrical Engineering - FCSE

Team Members:

Section: BSCS

Muhammad Ammar Saleem

2023378

Raja Bilal Khurram

2023591

Daanish Ahmad Mufti

2023171

Semester Project Documentation (DS-221)

Semester Project Title: Student Grading System)

1. Introduction

The grading system in universities is a crucial aspect of evaluating students' performance. This project focuses on designing and implementing a **Relative Grading System** that adjusts student grades based on a statistical distribution. The system can also apply **Absolute Grading** based on fixed thresholds, depending on the instructor's preference. The primary goal is to ensure fairness and statistical validity in grade adjustments while providing instructors with detailed analytics, including descriptive statistics and visualizations. This project adheres to the guidelines set by the **Higher Education Commission (HEC)** for grade distributions.

2. Problem Statement

Universities often use either an **absolute grading system** or a **relative grading system**. Absolute grading assigns fixed grades based on predefined thresholds (e.g., grades $\geq 90\%$ = A), while relative grading adjusts grades according to a specified distribution, such as a normal distribution or percentile-based scaling.

The task is to create a system that:

1. Allows instructors to input student grades (via CSV/Excel).
2. Enables instructors to choose between absolute or relative grading.
3. Provides statistical analysis of the grades before and after adjustment.
4. Includes visualizations to justify the adjustments made.

3. Key Features of the System

a. Input Module

- **Grade Input:** The system allows the instructor to upload student grades via a CSV or Excel file.
- **Grading Method:** The instructor can select between **absolute grading** (fixed thresholds) or **relative grading** (adjusted to match a statistical distribution).
- **Grade Distribution:** If relative grading is selected, the instructor can specify the desired distribution of grades (e.g., percentages for A, B, C, etc.). For absolute grading, the instructor can define thresholds (e.g., A for $\geq 90\%$).

b. Statistical Analysis

- **Relative Grading:** Calculates **mean**, **variance**, and **skewness** of the input grades to understand the grade distribution.

- **Visualization:** Plots histograms or density plots of the grade distribution before and after adjustment. This helps to understand how the grades are distributed and how the adjustments affect the final results.
- **Absolute Grading:** Provides summary statistics (e.g., the percentage of students in each grade category) based on fixed thresholds.

c. Grade Adjustment

- **Relative Grading:** The grades are adjusted using algorithms like **Z-score scaling**, **curve fitting**, or **percentile scaling** to ensure that the final distribution meets the desired grade percentages.
- **Absolute Grading:** Assigns grades based on the thresholds defined by the instructor or based on the **HEC guidelines**.

d. Reporting and Visualization

- **Detailed Report:** The system generates a comprehensive report that includes the original and adjusted grades.
- **Visualizations:** Displays histograms, boxplots, and bar charts comparing the original and adjusted grade distributions.
- **Summary Statistics:** Provides insights into the distribution, including how many students moved between different grade categories after adjustment.

e. Error Handling


- The system handles missing data, invalid input types, and incorrectly formatted files gracefully. It prompts the user with clear error messages if something goes wrong.

f. User Interface (Optional)

- A **simple GUI** or web interface is implemented for ease of use. The interface allows instructors to upload grades, select grading methods, adjust thresholds, and view results interactively.


Grading System

Choose a file (CSV or Excel)



Drag and drop file here
Limit 200MB per file • CSV, XLSX

Browse files



grades.csv 65.0B
 ×

Uploaded Grades:

	StudentID	Grade
0	1	85
1	2	90
2	3	78
3	4	92
4	5	88
5	6	76
6	7	95
7	8	80

Select grading method

☒ Absolute Grading

☐ Relative Grading

Choose Absolute Grading Type:

☒ Predefined Grading System

☐ Custom Grading System

4. Methodology

a. Statistical Methods for Relative Grading

1. **Z-Score Scaling:** This method normalizes the grades by converting them into Z-scores based on the **mean** and **standard deviation**. Each student's grade is then mapped to a new grade based on these Z-scores.
 - Formula: $Z = \frac{X - \mu}{\sigma}$, where X is the student's score, μ is the mean, and σ is the standard deviation.
2. **Percentile-Based Scaling:** This method adjusts grades to ensure that a specified percentage of students fall within certain grade categories (e.g., 30% A's, 40% B's).
 - The system calculates percentiles based on the grade distribution and assigns grades accordingly.
3. **Curve Fitting:** This method fits the grade distribution to a normal curve, adjusting the grades to ensure a bell-shaped distribution that matches the desired grade percentages.

b. Statistical Methods for Absolute Grading

1. **Threshold-Based Grading:** This method assigns grades based on fixed thresholds. For example:
 - $\geq 90\% = A$
 - $\geq 80\% = B$
 - $\geq 70\% = C$
 - $< 70\% = F$

5. Statistical Calculations and Visualizations

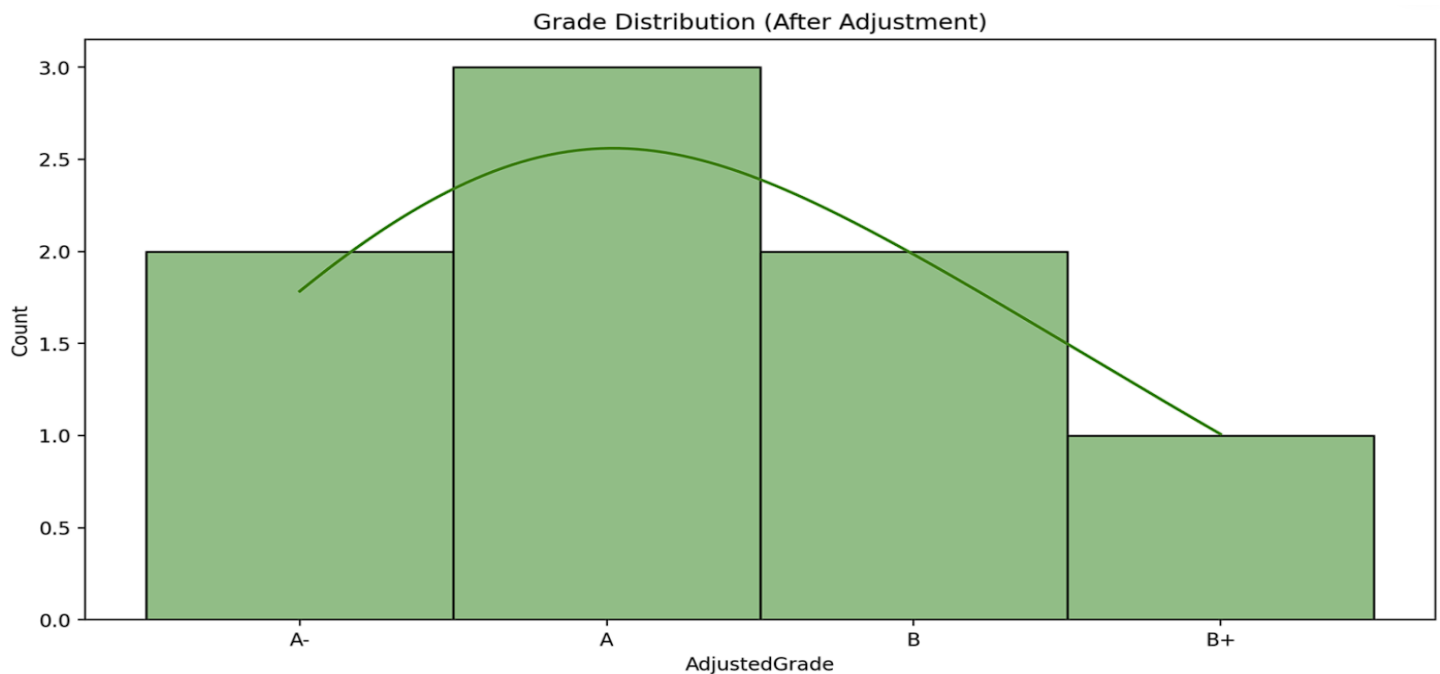
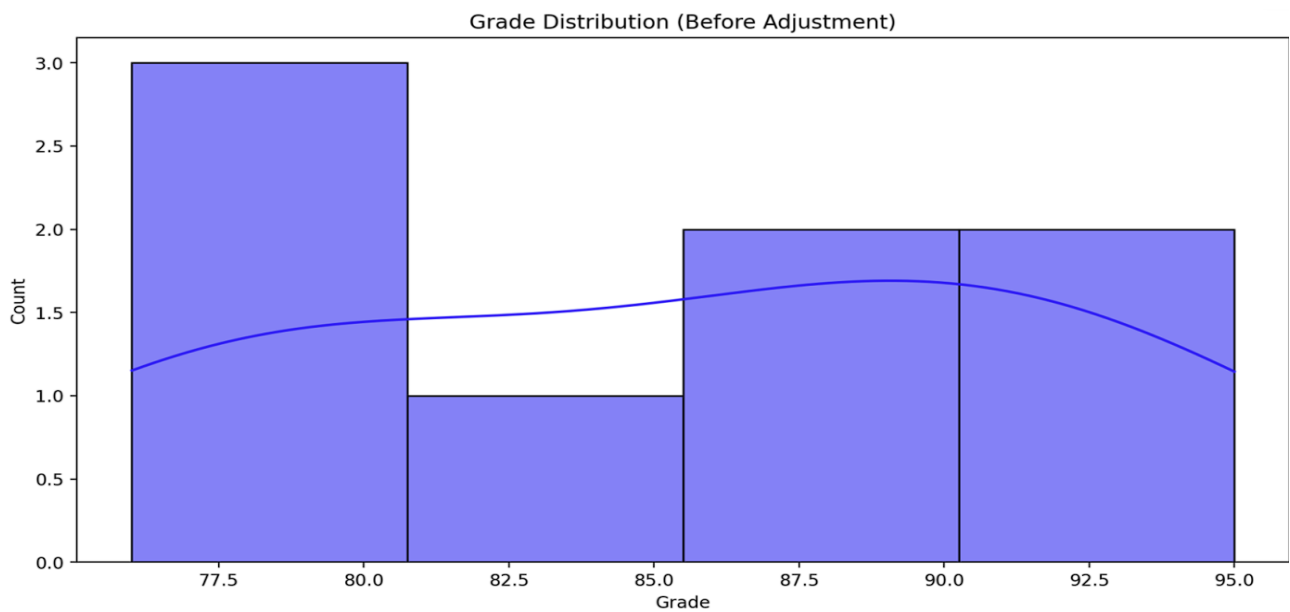
a. Descriptive Statistics

For both relative and absolute grading, the system calculates the following descriptive statistics:

- **Mean:** The average grade of the class.
- **Variance:** The spread or variability of grades.
- **Skewness:** The asymmetry of the grade distribution, which helps in identifying if the distribution is skewed towards higher or lower scores.

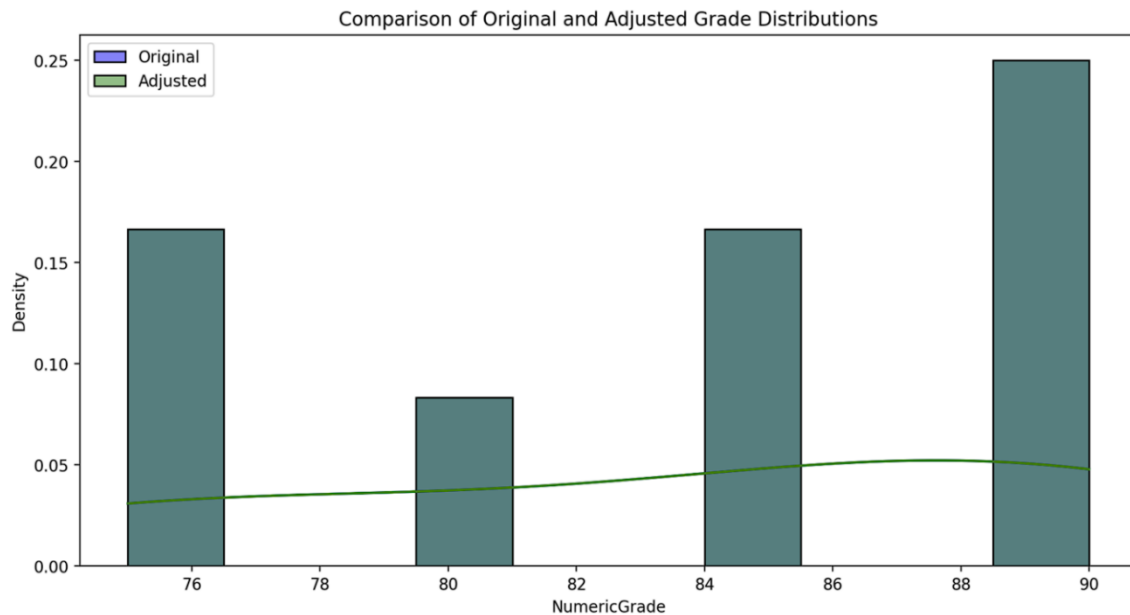
b. Visualizations

- **Histogram:** Plots of grade distributions before and after adjustment to show the shape and spread of the data.
- **Density Plot:** A smooth curve that shows the distribution of grades, highlighting the peak and tails.
- **Boxplot:** A visualization showing the quartiles, median, and potential outliers in the grade distribution.



c. Comparison of Original and Adjusted Grades

The system generates side-by-side visualizations comparing the original grade distribution with the adjusted.



6. Error Handling and Data Integrity

The system is designed to handle errors such as:

- **Missing Grades:** Automatically fills in missing values with the average grade.
- **Invalid File Formats:** If the uploaded file is not in CSV or Excel format, the system prompts the user to upload a valid file.
- **Incorrect Thresholds:** If thresholds for absolute grading are incorrectly entered (e.g., not in descending order), the system raises an error and asks for correct input.

7. Challenges Faced

During this project, our team learned so much. From writing python code from scratch to spending countless hours understanding the syntax of python codes. Selecting and importing libraries for our required needs, was one of the hardest tasks. Libraries like NumPy, Matplotlib, and Pandas were particularly daunting at first, as understanding statistical concepts and implementing them in code required a steep learning curve. Integrating our backend code with frameworks like Streamlit and Seaborn to provide a user-friendly front end was another major hurdle, as it involved merging design with functionality.

Reading and writing files in Python was also a very challenging task as we had to handle edge cases like empty files etc. Writing efficient codes for custom absolute grading and Grade percentages for relative gradin turned out to be significantly difficult task. After countless nights of working we finally derived a way to efficiently achieve this. We divided the task into different parts and amongst ourselves to achieve this. One of us was working on understanding HEC policies, and the other two were working on deriving an efficient piece of code function to achieve this.

Edge cases, such as handling user inputs exceeding 100%, preventing index errors, and ensuring graceful error messages for faulty files, demanded significant time and collaboration. Debugging these issues consumed late nights, but as a group, we divided tasks, brainstormed solutions, and tested scenarios rigorously to overcome each obstacle. Through persistence and teamwork, we successfully tackled each challenge and grew as problem solvers in the process.

8. Future use

In the future, the Student Grading System will integrate with Learning Management Systems (LMS) for automatic grade imports, saving time and reducing errors. Machine learning will predict at-risk students, enabling early interventions. Cloud deployment will ensure scalability and remote access for large institutions. Advanced error handling will improve grade accuracy by detecting issues like overlapping thresholds. Customizable reporting will help instructors analyze grade trends, and an enhanced user interface will make the system more intuitive and accessible, benefiting both instructors and students.

9. Conclusion

This **Relative Grading System** ensures a fair and statistically sound method for adjusting student grades. It adheres to the **HEC guidelines** and offers flexibility for instructors to choose between **absolute** or **relative grading methods**. The system provides useful analytics, including descriptive statistics and visualizations, to justify the adjustments. In addition to the core grading functionality, the system handles missing data and invalid inputs effectively, ensuring a smooth user experience. This project aims to offer a comprehensive solution for grading that is both fair and transparent.

Libraries Used and Their Implementation

This **Relative Grading System** leverages several Python libraries to handle data input/output, statistical analysis, and visualizations. Below are the libraries used and their specific purposes in the project:

1. Pandas

Purpose:

Pandas is used for handling data input, manipulation, and cleaning. It is especially useful for reading CSV/Excel files and performing data analysis on the grade data.

Usage:

- **Reading Data:** To load grades from a CSV or Excel file into a DataFrame.
- **Data Cleaning:** To handle missing values and filter out invalid data.

2. NumPy

Purpose:

NumPy is used for numerical operations such as calculating the **mean**, **variance**, and **standard deviation** of the grades, as well as performing statistical transformations (e.g., z-score scaling).

Usage:

- **Statistical Calculations:** Calculating mean, standard deviation, and variance for the relative grading algorithm.
- **Z-Score Calculation:** To adjust grades using z-scores.

3. Matplotlib

Purpose:

Matplotlib is used for creating visualizations such as histograms, box plots, and bar charts to compare the original and adjusted grade distributions.

Usage:

- **Histograms:** To visualize the grade distribution before and after adjustment.
- **Boxplots:** To identify the spread of grades and outliers.
- **Bar Charts:** To compare the percentage of students in each grade category.

4. Seaborn

Purpose:

Seaborn is an extension of matplotlib and is used for creating more complex visualizations such as **density plots** and **boxplots** with enhanced aesthetics.

Usage:

- **Density Plots:** To show the smoothed distribution of grades before and after adjustments.
- **Boxplots:** To display the spread and outliers of grades.

5. SciPy

Purpose:

SciPy is used for advanced statistical operations, particularly when implementing **Z-score scaling** or performing **curve fitting** for relative grading. It is also useful for calculating skewness and other statistical measures.

Usage:

- **Skewness:** To measure the asymmetry of the grade distribution.
- **Curve Fitting:** To fit the data to a normal distribution or another statistical distribution..

6. Openpyxl (For Excel File Handling)

Purpose:

openpyxl is used to handle Excel files (.xlsx) in case the instructor prefers this format over CSV.

Usage:

- **Reading Excel Files:** If the input grades are stored in an Excel file, openpyxl allows reading and manipulation..

7. Tkinter (Optional GUI)

Purpose:

Tkinter is used to create a simple graphical user interface (GUI) for the grading system, making it easier for instructors to interact with the program without requiring command-line knowledge.

Installation:

Tkinter comes pre-installed with Python in most distributions, but if needed, you can install it using:

Usage:

- **File Uploads:** Provides an interface to select and upload student grades.
- **Grade Method Selection:** Allows instructors to choose between absolute and relative grading through a graphical interface.

Conclusion

In summary, these libraries work together to provide a complete solution for the grading system:

- **Pandas** for data input and manipulation.
- **NumPy** and **SciPy** for statistical calculations.
- **Matplotlib** and **Seaborn** for visualizing grade distributions.
- **Tkinter** for an optional GUI interface.