Grade Prediction using Gaussian and Linear Models

1 Introduction

Predicting student grades based on their current scores is a valuable task in educational analytics. This document explores two different approaches:

1. **Gaussian (Normal) Distribution Model** – Assumes scores follow a bell curve.
2. **Linear Model** – Assumes scores are evenly distributed over a fixed range.

Both models aim to map a student's score to a grade between 3 and 10.

2 Gaussian Distribution Model

A normal distribution is a continuous probability distribution that is symmetric around its mean. The probability density function (PDF) is given by:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{1}$$

where:

- x is the student's current score,
- μ is the mean (average) score,
- σ is the standard deviation.

Grades are assigned based on standard deviation intervals:

$$
\begin{aligned}
\text{Grade 10} &: x \geq \mu + 3\sigma, \\
\text{Grade 9} &: \mu + 2\sigma \leq x < \mu + 3\sigma, \\
\text{Grade 8} &: \mu + \sigma \leq x < \mu + 2\sigma, \\
\text{Grade 7} &: \mu \leq x < \mu + \sigma, \\
\text{Grade 6} &: \mu - \sigma \leq x < \mu, \\
\text{Grade 5} &: \mu - 2\sigma \leq x < \mu - \sigma, \\
\text{Grade 4} &: \mu - 3\sigma \leq x < \mu - 2\sigma, \\
\text{Grade 3 and below} &: x < \mu - 3\sigma.
\end{aligned}
$$

```python
import numpy as np

mu = 7  # Mean grade
sigma = 1  # Standard deviation

grade_cutoffs = {
    10: mu + 3 * sigma,
    9: mu + 2 * sigma,
    8: mu + sigma,
    7: mu,
    6: mu - sigma,
    5: mu - 2 * sigma,
    4: mu - 3 * sigma
}

def predict_grade_gaussian(score):
    for grade, cutoff in sorted(grade_cutoffs.items(), reverse=True):
        if score >= cutoff:
            return grade
    return 3  # Lowest grade

test_scores = [5.5, 6.2, 7.8, 9.1]
predicted_grades = [predict_grade_gaussian(score) for score in test_scores]
print(predicted_grades)
```

Listing 1: Gaussian Grade Prediction Algorithm

3 Linear Distribution Model

A linear model assumes that scores are evenly distributed over a fixed range. The grade is computed using:

$$G(x) = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \times (G_{\max} - G_{\min}) + G_{\min} \tag{2}$$

where:

- x is the student's score,
- $x_{\min}$ and $x_{\max}$ are the minimum and maximum scores,
- $G_{\min} = 3$ and $G_{\max} = 10$ are the lowest and highest grades.

```python
import numpy as np

x_min = 0   # Minimum possible score
x_max = 10  # Maximum possible score

def predict_grade_linear(score, x_min=0, x_max=10):
    grade = (score - x_min) / (x_max - x_min) * 7 + 3
    return round(grade)  # Round to nearest integer

test_scores = [3.5, 5.2, 7.8, 9.1]
predicted_grades = [predict_grade_linear(score) for score in
    test_scores]
print(predicted_grades)
```

Listing 2: Linear Grade Prediction Algorithm

4 Performance Analysis

Both models have different advantages and disadvantages. We analyze them using the following seven points:

1. **Assumptions:** The Gaussian model assumes a normal distribution, while the linear model assumes an even spread of scores.
2. **Error Measurement:** We can compare predictions with actual grades using Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).
3. **Range Sensitivity:** The Gaussian model depends on the correct estimation of μ and σ, while the linear model depends on accurate values of $x_{\min}$ and $x_{\max}$.
4. **Handling Outliers:** The Gaussian model naturally restricts values within 3σ from μ, while the linear model scales everything between $x_{\min}$ and $x_{\max}$.
5. **Grade Distribution:** The Gaussian model assigns more students to mid-range grades, while the linear model distributes grades evenly across the range.
6. **Cross-validation:** Splitting data into training and testing sets is essential for verifying the robustness of both models.
7. **Comparison:** If scores are normally distributed, the Gaussian model is better; if they are uniformly spread, the linear model is preferable.

5 Conclusion

Both models provide a structured approach to grade prediction:

- The **Gaussian model** is useful when scores naturally follow a bell curve.
- The **linear model** is better for evenly distributed scores.
- The best model depends on actual data distribution.