Statistics and Data Analysis
Unit 01 – Lecture 02 Notes
Feature Scaling and Feature Engineering Basics

Tofik Ali

February 9, 2026

What You Will Learn

In this lecture we focus on practical preprocessing steps that are used *before* modeling:

- scaling (normalization and standardization),
- transformations (especially for skewed variables),
- and basic feature engineering (creating helpful new variables).

1. Feature Scaling

1.1 Why scaling is needed

Many datasets contain features with different units and magnitudes, for example:

- attendance (%) typically ranges 0–100,
- study hours per week might range 0–25,
- income could range 20–200 (in thousands) or much more.

If we use distance (like Euclidean distance), the largest-scale feature can dominate. This is why scaling is important for:

- kNN (nearest neighbors),
- k-means clustering,
- many gradient-based optimization methods.

1.2 Min–max normalization

Min–max scaling maps a feature into $[0, 1]$:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Interpretation: $x' = 0$ means the minimum observed value; $x' = 1$ means the maximum observed value.

Exercise 1 (solution). If min=20, max=80, and $x = 50$:

$$x' = \frac{50 - 20}{80 - 20} = \frac{30}{60} = 0.5$$

1.3 Standardization (z-score)

Standardization uses mean and standard deviation:

$$z = \frac{x - \mu}{\sigma}$$

Interpretation: z is the number of standard deviations x is above/below the mean.

Exercise 2 (solution). If $\mu = 60$, $\sigma = 10$, and $x = 80$:

$$z = \frac{80 - 60}{10} = 2$$

So the score 80 is 2 standard deviations above the mean.

1.4 Important warning: data leakage

When we scale, we compute parameters (min/max or mean/std). These parameters must be estimated on the **training data only**. Then we use the same parameters to transform the test set.

If we use the whole dataset (including test) to compute scaling parameters, then test information "leaks" into training. That can make performance look better than it truly is.

Exercise 3 (solution).

- kNN: Yes (distance-based)
- k-means: Yes (distance-based)
- Linear regression (gradient descent): often yes (helps optimization)
- Decision tree: usually no (split rules are scale-invariant)

2. Transformations

2.1 Why transform?

Scaling changes the *units* or spread, but it does not necessarily fix skewness. For right-skewed features (like income), many values are small and a few values are extremely large.

Transformations can:

- reduce skewness,
- make relationships more linear,
- reduce the impact of extreme values.

2.2 Log transform

A common transform for positive values is:

$$x' = \log(1 + x)$$

We use $\log(1 + x)$ instead of $\log(x)$ to handle zero values safely.

Exercise 4 (solution). For right-skewed income, a reasonable first transform is $\log(1 + x)$.

3. Feature Engineering Basics

3.1 What is feature engineering?

Feature engineering means creating new variables (features) from raw columns. Good engineered features can:

- capture patterns more directly,
- improve model performance,
- and make interpretations clearer.

3.2 Common patterns

- **Encoding categorical variables**: one-hot encoding for nominal categories.
- **Buckets/bins**: convert continuous variables to categories (e.g., attendance low/medium/high).
- **Interactions**: multiply/add features when combined effect matters (effort = hours × attendance).
- **Ratios**: e.g., marks per hour, cost per unit.
- **Date parts**: month, weekday, time-of-day.
- **Text features**: word count, length, keyword flags (basic features).

Exercise 5 (example answers).

- `income_log1p`: reduces skew and outlier impact.
- `effort_index` = hours × attendance/100: single "effort" score.
- `join_month`: captures seasonal effects.

Exercise 6 (solution). If `program` has categories CSE/ECE/AIML, the one-hot columns can be:

`program_CSE`, `program_ECE`, `program_AIML`

Each row has exactly one 1 (its category) and 0 for the others.

4. Mini Demo (Python)

Run from the lecture folder:

```
python demo/scaling_feature_engineering_demo.py
```

The demo:

- scales three features (attendance, hours, income) and prints how distances change after scaling,
- engineers features (month, weekday, income log, effort index, backlog flag, one-hot program),
- saves `data/student_features_engineered.csv`,
- saves plots in `images/`.

References

- McKinney, W. *Python for Data Analysis*. O'Reilly, 2022.
- Kuhn, M., & Johnson, K. *Feature Engineering and Selection*. CRC Press, 2019.
- Montgomery, D. C., & Runger, G. C. *Applied Statistics and Probability for Engineers*. Wiley, 7th ed., 2020.