Feature Scaling

- ML model may behave incorrectly if one feature has higher value than other
- Numerical features should be transformed such that they have similar range
- Below, without scaling Salary dominates learning as distance gets biased.
- Scaling improves accuracy, speed up training and prevent dominant of large value feature
- Types: MinMaxScaler, StandardScaler

Age	Experience	Salary	Position
23	3	30000	Junior
20	1	35000	Intern
30	7	50000	Senior

MinMax Scaling

- Transforms data into range [0, 1]
- Preserves original distribution shape
- Sensitive to outliers
- Formulae: $\bar{X} = \frac{X - Xmin}{X_{max} - Xm_{in}}$

Age	Experience	Salary	Position	Age Scaled	Exp Scaled	Salary Scaled
23	3	30000	Junior	0.3	0.33	0
20	1	35000	Intern	0	0	0.25
30	7	50000	Senior	1	1	1

Standard Scaling

- Also called Z – Score Scaling
- After transformation new data is centered around μ **= 0** & σ **= 1**
- Widely used in ML
- Mean will be 0, Value above mean will be +ve & below mean will be -ve
- Formulae: $\grave{X} = \frac{X - \mu}{\sigma}$

Age	Experience	Salary	Position	Age Scaled	Exp Scaled	Salary Scaled
23	3	30000	Junior	-0.32	-0.27	-0.98
20	1	35000	Intern	-1.03	-1.07	-0.39
30	7	50000	Senior	1.35	1.34	1.37

Robust Scaling

- Uses median and IQR for data transformation
- Less affected by outlier
- Best when data has extreme value
- Formulae: $\grave{X} = \frac{X - Median}{IQR}$

Age	Experience	Salary	Position	Age Scaled	Exp Scaled	Salary Scaled
23	3	30000	Junior	0	0	-0.5
20	1	35000	Intern	-0.6	-0.67	0
30	7	50000	Senior	1.4	1.33	1.5

Max Absolute Scaling

- Transforms data into range [-1, 1]
- Preserves zero values
- Useful for sparse data (most value as 0)
- Formulae: $\grave{X} = \frac{X}{|X_{max}|}$

Age	Experience	Salary	Position	Age Scaled	Exp Scaled	Salary Scaled
23	3	30000	Junior	0.77	0.43	0.6
20	1	35000	Intern	0.67	0.14	0.7
30	7	50000	Senior	1	1	1

Summary

Scaling Method	Range	Outliers	Best For
Min-Max	0 to 1	❌ No	Image data
Standard	Mean 0, Std 1	❌ No	Most ML models
Robust	No fixed	✅ Yes	Outlier-heavy data
MaxAbs	-1 to 1	❌ No	Sparse data

Use Scaling For	Don't Scale For
Linear Regression	Decision Tree
Logistic Regression	Random Forest
SVM	XGBoost
KNN	
K – Means Clustering	
Neural Network	

Python Demo Code

Encoding

- Scikit learn doesn't accept categorical feature
- Categorical features has to be encoded numerically
- Convert data to dummy variables.
- Some common encoding techniques are:
  - Label Encoding
  - One-Hot Encoding
  - Binary Encoding
  - Target Mean Encoding
  - Frequency / Count Encoding
  - Embedding

Ordinal Encoding

- Technique to convert categorical variables (ordinal) into numeric form
- Useful for ordinal data where order matters
- Not good for nominal data as model may assume numeric relationships
- Example: [“large”, “small”, “medium”] → [2, 0, 1]

Label Encoding

- Technique to convert categorical variables into numeric form
- Assigns label alphabetically. Example: apple → 0, ball → 1 as apple < ball
- Can mislead in nominal data as model may assume numeric relationships
- Example: [“red”, “blue”, “yellow”] → [0, 1, 2]

One - Hot Encoding

- Creates a binary column for each category
- Example: [“red”, “blue”, “green”] →

red	blue	green
1	0	0
0	1	0
0	0	1

- Works well for nominal data
- Downside: Creates many columns if category are numerous

Binary Encoding

Binary Encoding

- First convert categories into integers, then into binary
- After its separated into multiple columns (of binary digit)
- Example: “red” → 1 → 01; “blue” → 2 → 10; “Green” → 3 → 11
- More compact that One Hot Encoding
- For n values $\log_2(n)$ new columns are sufficient to hold encoded data
- Works well for high cardinality feature

Other Encoding

Target Mean Encoding

- Replace categories with the mean of target variable for each
- Some smoothing is applied so we get non-intuitive number instead of mean

Frequency Encoding

- Replace categories with their frequency or count
- Works well with high cardinality feature
- City: "London" (1000 times), "Paris" (500 times) → 1000, 500

Embedding (For Deep Learning)

- Represent category in dense vector space
- Used in NLP and recommendation system

Python Code