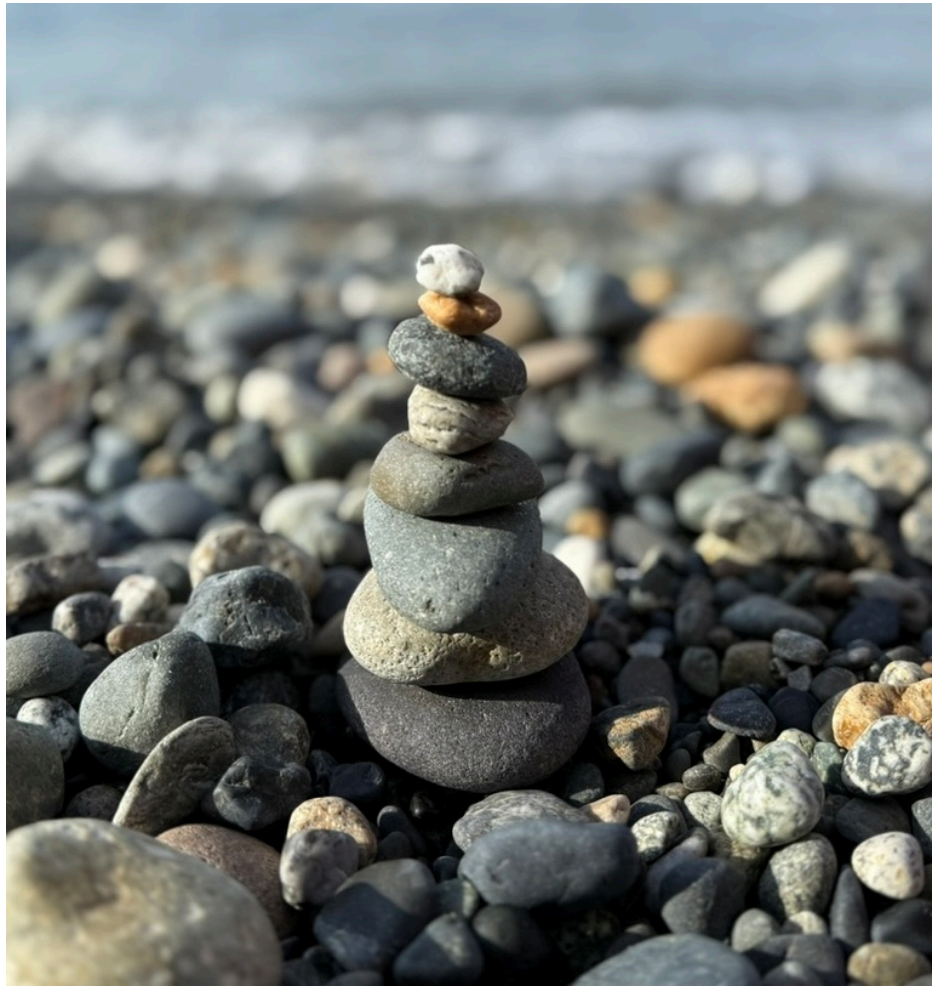
Lecture 5: Preprocessing and sklearn pipelines

Varada Kolhatkar

Focus on the breath!



Announcements

- HW1 grades will be posted soon.
- Homework 3 (hw3) has been released (Due: Oct 5th, 11:59 pm)
  - You can work in pairs for this assignment.
- My OH: Tuesdays and Thursdays right after the class.

Supervised ML workflow

Next two lectures: how do we get data into a useful representation?

iClicker: where does the time go?

Take a guess: how much of an ML project's time might go into data preparation and transformation?

a. Less than 20%

b. 20–40%

c. More than 40%

d. None; most time goes into model building

Often a **substantial share**: understanding, cleaning, and transforming data takes time. The exact proportion varies by project.

Inspired by Google's ML Crash Course

Remember this from last lecture?

- You're trying to find a suitable date based on age and social media followers.
- You are 30 years old and have 250 followers.

Person	Age	# Followers	Euclidean Distance Calculation	Distance
A	25	400	$\sqrt{5^2 + 150^2}$	150.08
B	27	300	$\sqrt{3^2 + 50^2}$	50.09
C	30	500	$\sqrt{0^2 + 250^2}$	250.00
D	60	250	$\sqrt{30^2 + 0^2}$	30.00

Based on the distances, the two nearest neighbors (2-NN) are:

- **Person D** (Distance: 30.00)
- **Person B** (Distance: 50.09)

What's the problem here?

Which models are affected?

In the age-and-followers example, which models are sensitive to the **relative scales of the two features**? Why?

- **Decision trees**
- **k-NN**
- **RBF SVM**

k-NN and RBF SVM use distances, so feature scales affect their neighbours or similarities.

Decision trees split on one feature at a time; changing units allows an equivalent split threshold.

Same people, different units

Suppose we express age in **months instead of years**, leaving follower counts unchanged.

Person	Distance using years	Distance using months
B	$\sqrt{3^2 + 50^2} \approx 50.09$	$\sqrt{36^2 + 50^2} \approx 61.61$
D	$\sqrt{30^2 + 0^2} = 30$	$\sqrt{360^2 + 0^2} = 360$

Which of these two people is closer now?

B replaces D as the closer person. **Only the units changed, not the people.**

Why data representation matters

Models operate on the representation we give them.

- **Units:** age in years $\longrightarrow$ months changes distances without changing the people.
- **Scales:** ages might range from 18–60, but follower counts from 10–1000. Differences in follower counts can dominate distances.
- **Missing entries:** someone leaves their age blank (`age = NaN`). How do we calculate their distance?
- **Category codes:** suppose we add a favourite hobby: hiking = 0, reading = 1, gaming = 2. Why should hiking be “closer” to reading than gaming?

Today's big questions

- What **representation** does the model need?
- What do transformations **learn**, and which data may they learn from?
- How do we combine **preprocessing and prediction** safely?

Class demo: restaurant survey responses

Open the preprocessing demo

Task: infer a respondent's `like` / `dislike` label from their completed survey.

As we work, watch for:

1. What does each transformation change?
2. What does `fit` learn?
3. Which rows are allowed to influence that learning?

Recap: choosing transformations

Problem	Demo features	Transformation
Missing values	`price`, `food_type`, `noise_level`	Impute with a training median or most frequent value
Different scales	`age`, `n_people`, `price`	Standardize numeric features
Unordered categories	`food_type`, `north_america`	One-hot encode
Ordered categories	`noise_level`	Ordinally encode with an explicit order

Choose the representation to match the meaning of the feature.

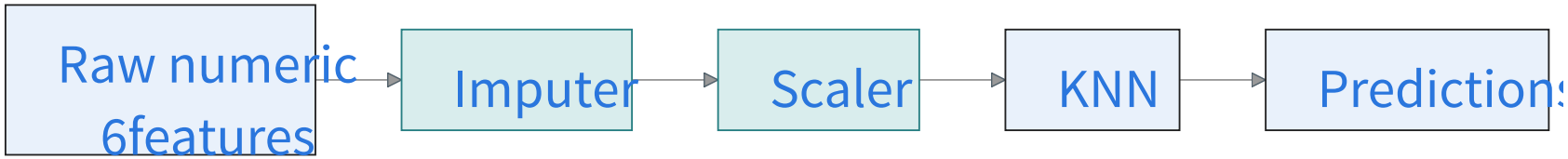
Transformers and predictive estimators

	Transformer	Predictive estimator
Purpose	Change feature representation	Predict a target
Learn	`fit(X)`	`fit(X, y)`
Apply	`transform(X)`	`predict(X)`
Examples	Imputer, scaler, encoder	k-NN, SVC

Both are estimators: both implement `fit`.

`fit_transform` fits and transforms the same data. On unseen data, **transform only**.

A pipeline combines preprocessing and prediction



During training	On unseen data
Imputer: **fit and transform**	Imputer: **transform**
Scaler: **fit and transform**	Scaler: **transform**
KNN: **fit** with labels	KNN: **predict**

A sequence of preprocessing steps followed by a predictive estimator.

The golden rule in preprocessing

Never call `fit` on the validation portion for preprocessing or the model.

- Learn medians, scales, and categories from **that fold's training rows**.
- Apply the fitted transformations to its validation rows.
- Pass the **whole pipeline and raw features** to cross-validation.

Keeping the final test set separate is necessary, but does not prevent leakage between CV folds.

Back to the full workflow

Inside each CV fold: raw features → fitted preprocessing → fitted classifier.

Cross-validate the whole pipeline: each fold fits preprocessing and the model using only its training rows. This keeps validation data out of `fit` and preserves the golden rule.

Why `ColumnTransformer`?

In the demo, we trained on feature groups separately:

- **Numeric:** impute → scale
- **Categorical:** impute → one-hot encode
- **Ordinal:** impute → encode with a specified order

How do we combine these transformed columns for one classifier?

`ColumnTransformer` applies different transformations to selected columns and joins the results into one feature matrix.

Summary

- **Representation:** handle missing values, feature scales, and categories according to what the features mean.
- **Learning:** fit transformations using only the available training rows, including within each CV fold.
- **Combination:** a pipeline keeps preprocessing and prediction together; cross-validate the whole procedure.

Next lecture: combine feature groups with `ColumnTransformer`.

Chapter 5: Preprocessing and pipelines

Exit question

A colleague imputes and scales **all training rows**, then cross-validates a KNN classifier on the transformed matrix.

“The test set is untouched, so there’s no leakage.”

Do you agree? What would you change?