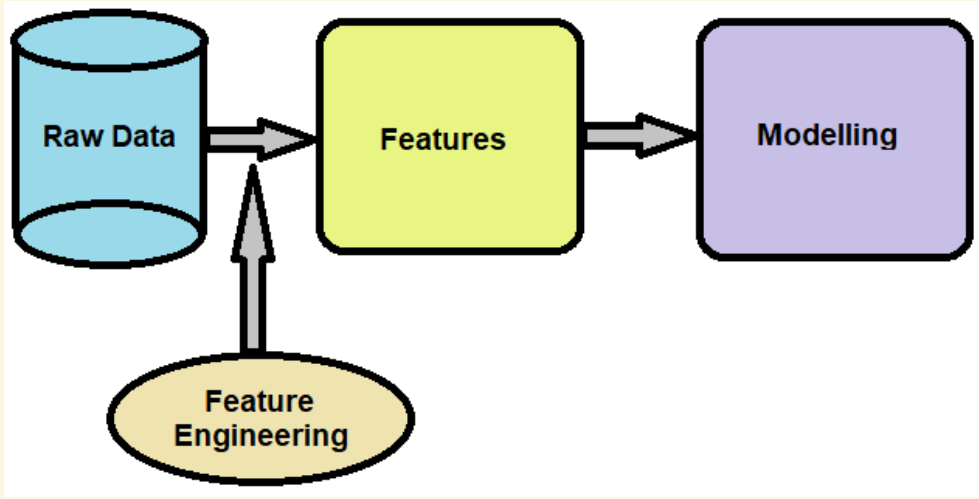
Feature Engineering

What is Feature Engineering?

Feature engineering is the process of **using domain knowledge to create or transform variables** (features) from raw data into formats that help machine learning models learn better.



Goal

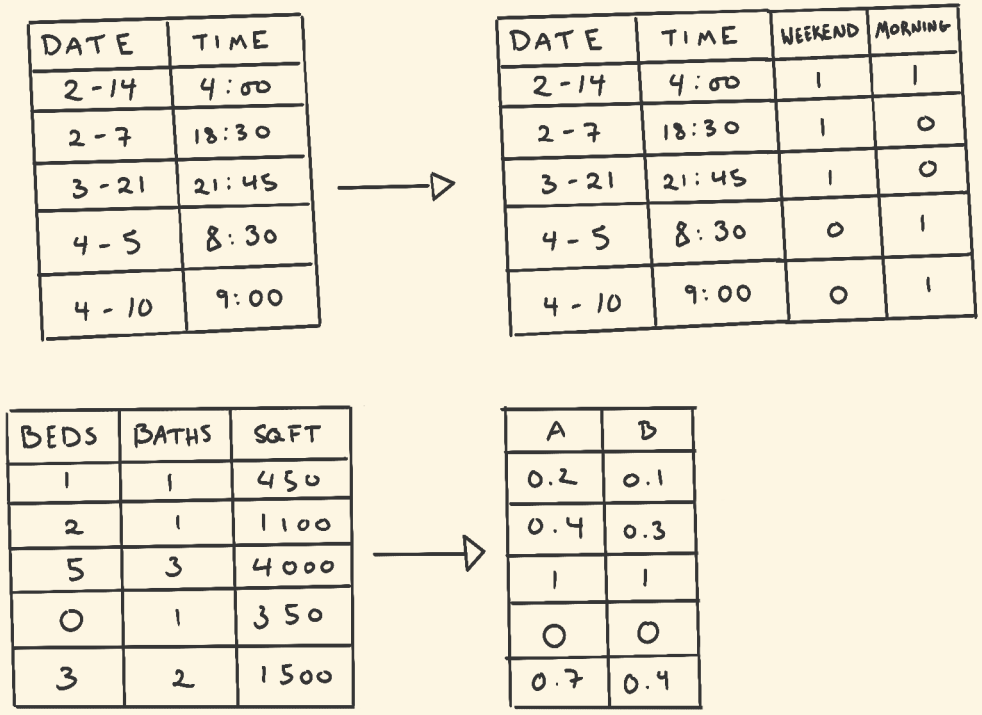
1. **Improve model performance** and interpretability.
2. **Reduce complexity** by transforming or encoding data more effectively.
3. **Leverage domain insights** so that the features *best capture underlying patterns*.

Why is Feature Engineering Important?

1. **Enhances Model Performance**
   Good features can significantly improve accuracy and reduce error.

2. **Reduces Data Complexity**
   Proper transformations can help algorithms converge faster and more reliably.

3. **Domain Knowledge**
   Adds “expert insight” that purely algorithmic approaches might miss.



Polynomial Features

Polynomial Transformation

Create new features by raising existing numeric features to a power (e.g., x^2, x^3).

Interaction Term

Multiply or combine features (e.g., $x_1 \times x_2$, x_1/x_2).

```
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly.fit_transform(X)
```

We've already seen something similar to this in Mean Squared Error. The polynomial degree is the same concept, but applied to the features themselves.

Interaction Features

Combine features in pairs or more to capture potential relationships.

Examples

- Multiplying two features: $x_1 \times x_2$
- Dividing features: x_1 / x_2
- Combining text-based features (e.g., bigrams, trigrams in NLP).

Real-World Examples

- `price_per_sqft` = `price` / `sqft`
- `BMI` = `weight` / `height` ^{2}
- `force` = `mass` × `acceleration`

AGE	HEIGHT(m)	WEIGHT(KG)
21	1.88	74.84
32	1.83	83.91
27	1.63	61.23
40	1.78	79.38
35	1.8	102.06

+

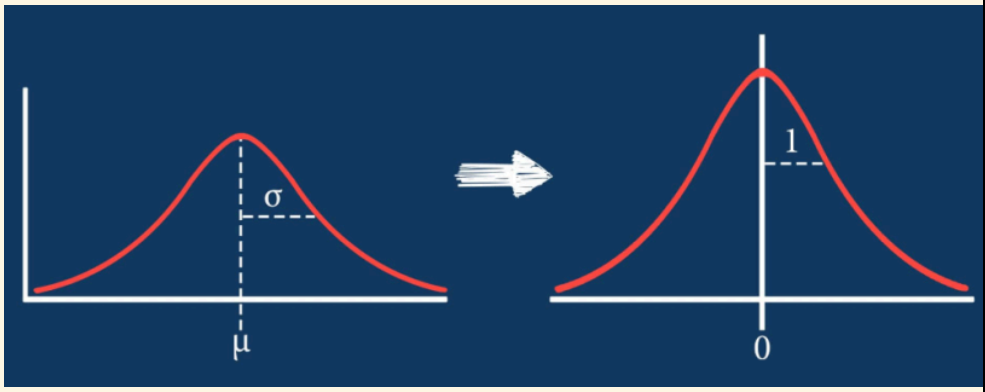
BMI
21.18
25.09
23.17
25.11
31.38

Normalization & Standardization

Normalization

Scale features to a fixed range. Useful for algorithms that rely on distances or magnitude (e.g., KNN, neural networks)

$$x_{norm} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$



Standardization

Transform features to have mean 0 and variance 1. Useful for linear models, logistic regression, SVM.

$$x_{std} = \frac{x - \mu}{\sigma}$$

De-Skewing & Transformation

Reduce skewness, handle outliers, or stabilize variance.

Log Transform

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

("+1" to handle zero or near-zero values)

Box-Cox / Yeo-Johnson

More general transformations that handle negative and zero values gracefully.

```
from sklearn.preprocessing import PowerTransformer

pt = PowerTransformer(method='yeo-johnson')
X_trans = pt.fit_transform(X)
```

One-Hot Encoding (Categorical Data)

Converts each category into a separate binary feature (0 or 1) so numerical algorithms can process them.

This is closely related to bag-of-words in NLP.

Binning (Discretization)

Converting continuous features into categories by splitting the range into intervals (bins).

Example

Age → 0-18, 19-35, 36-50, 51+.

Usage

- Helps capture *non-linear relationships* or reduce outlier impact.
- But can cause loss of granularity.

Dealing with Missing Data

Simple Methods

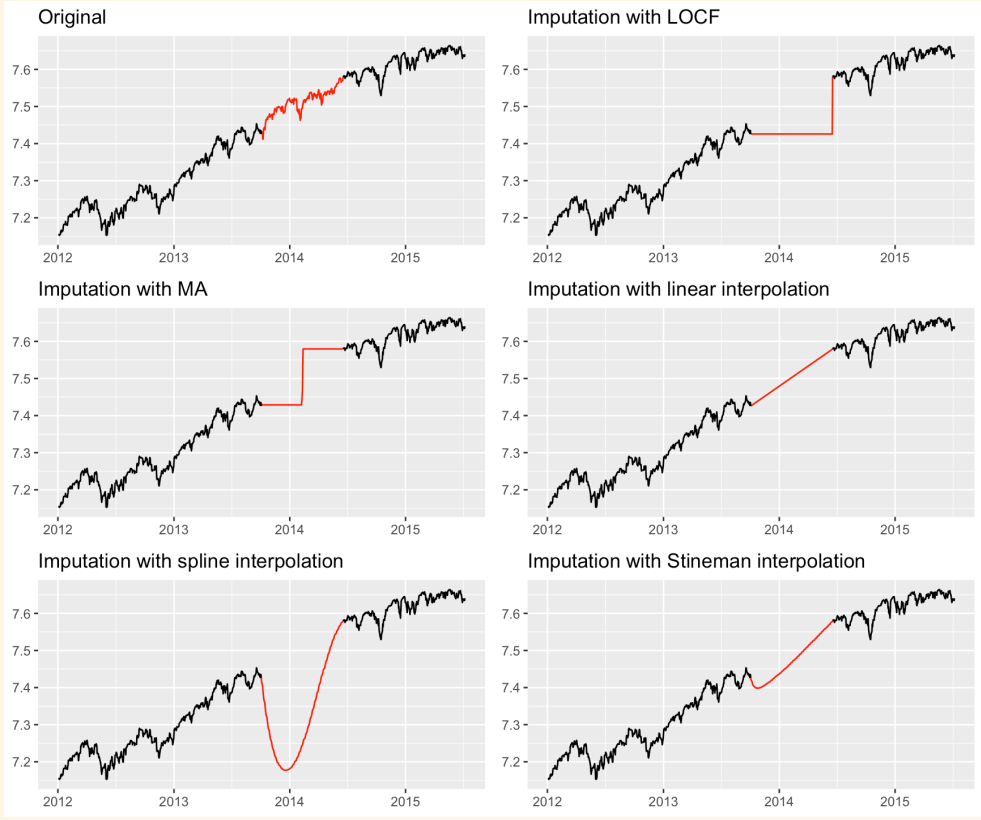
Replace missing values with mean, median, or mode.

Advanced Methods

KNN imputation, iterative imputer, or domain-specific rules.

Indicator Variables

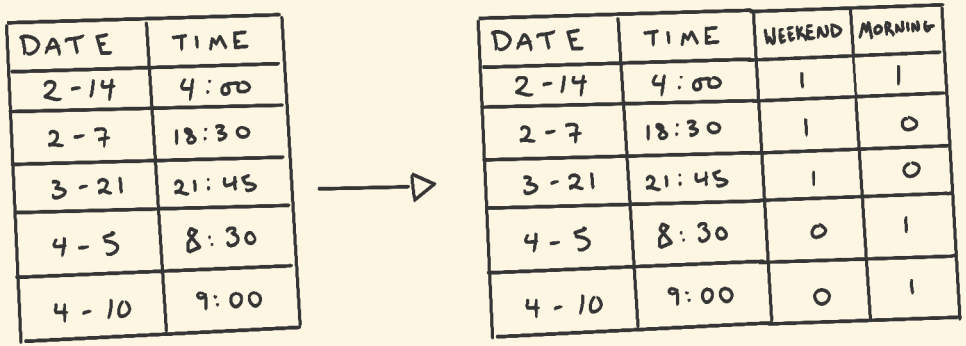
Create a binary feature “was_missing” to capture whether data was missing.



Date/Time Feature Engineering

Extract Components

Day of week, month, hour, whether it's a holiday, etc.



DATE	TIME
2-14	4:00
2-7	18:30
3-21	21:45
4-5	8:30
4-10	9:00

DATE	TIME	WEEKEND	MORNING
2-14	4:00	1	1
2-7	18:30	1	0
3-21	21:45	1	0
4-5	8:30	0	1
4-10	9:00	0	1

Cyclical Encoding

When dealing with periodic data (hours in a day, days in a week, months in a year), transform them into sine/cosine pairs.

$$x_{sin} = \sin \frac{2\pi \times \text{hour}}{24}, \quad x_{cos} = \cos \frac{2\pi \times \text{hour}}{24}$$

```
df['hour_sin'] = np.sin(2 * np.pi * df['hour']/24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour']/24)
```

Text Feature Engineering

As we saw in NLP, text data has multiple transformations into numerical features.

Bag-of-Words

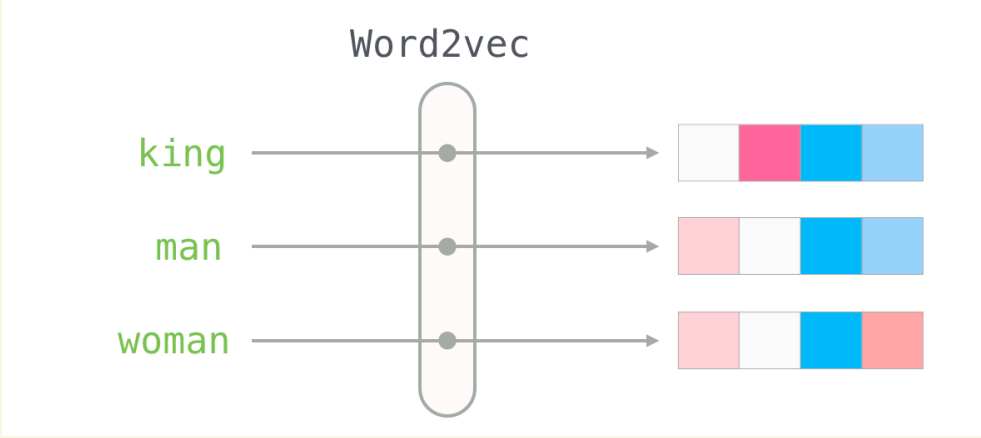
Converts text into counts of each word.

TF-IDF

Accounts for frequency across documents.

Embeddings

Word2Vec / GloVe / BERT embeddings for deeper semantic representation.



Exercise: Classify That Pokemon

bigd103.link/classify-pokemon