Featurization

- **Featurization** is the process of **converting raw data into numerical features** that a machine learning model can understand.
- Key idea
- Transform **raw input → machine-readable features**
- Examples
- Text → **Bag of Words, TF-IDF, Word2Vec**
- Categorical data → **Label Encoding, One-Hot Encoding**
- Images → **Pixel values, HOG, CNN embeddings**
- Date/Time → **Day, Month, Hour**
- Audio → **MFCCs**

Feature Engineering

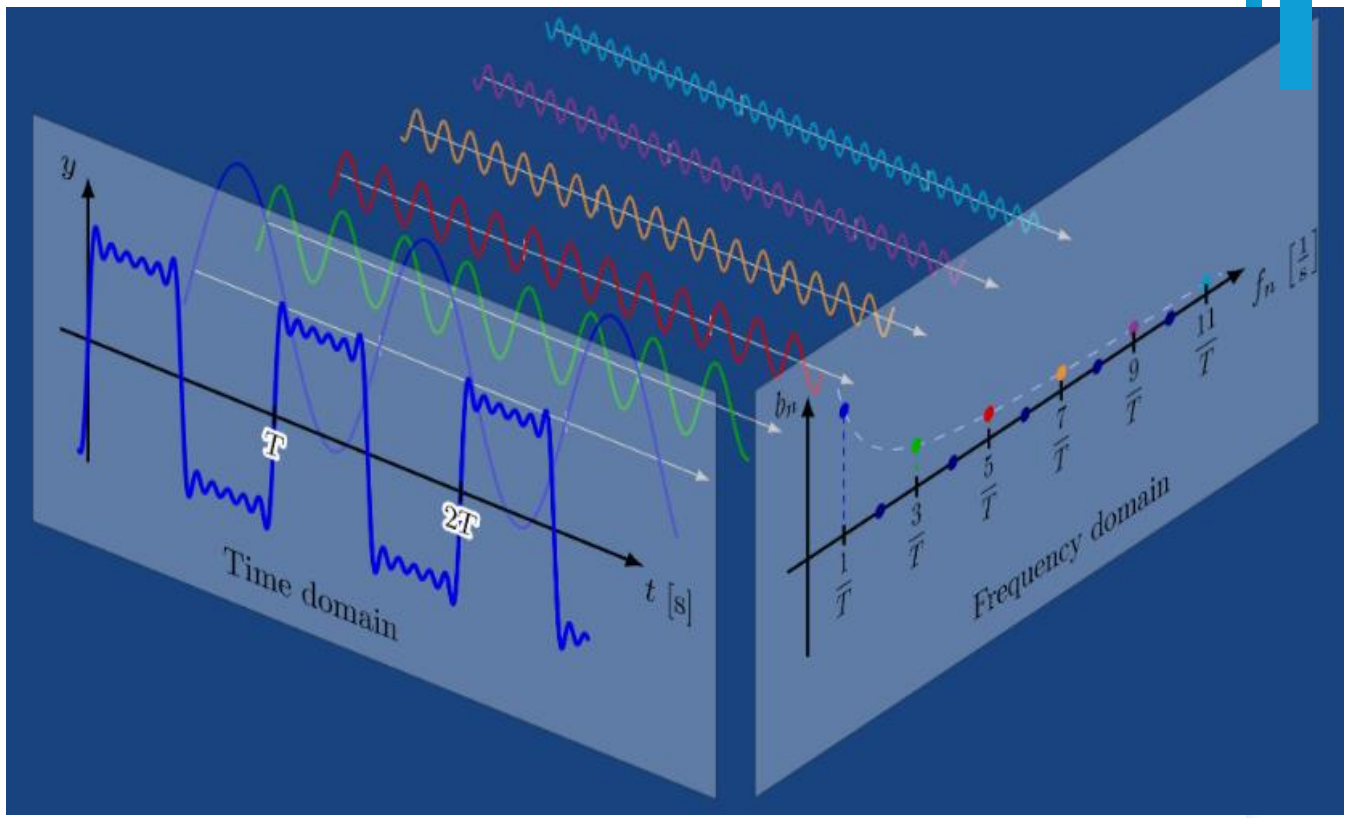
- **Feature Engineering** is the process of **creating, modifying, selecting, or combining features** to **improve model performance**.
- Key idea
- Use **domain knowledge + creativity** to make features more informative
- Examples
- Creating `Age = Current Year – Birth Year`
- Combining features: `Total Spend = Price × Quantity`
- Polynomial features: x^2, x^3
- Interaction features: `Income × Education`
- Feature selection: removing irrelevant features
- Binning: converting continuous → categorical
- Characteristics
- **Creative and problem-dependent**
- Uses **domain knowledge**
- Strongly impacts **model accuracy & interpretability**
- Includes:
  - Feature creation
  - Feature transformation
  - Feature selection

Moving Window in Time Series

- Technique that analyzes fixed-size, consecutive subsets of time series data
- **Purpose:** Capture local patterns, smooth noise, and enable forecasting
- **How it works:** Window moves step-by-step over the series
- **Types:**
- **Sliding (Fixed) Window** – constant size
- **Expanding Window** – grows over time
- **Rolling Window** – used for statistics
- **Common Operations:** Moving average, rolling std, min/max, lag features
- **Forecasting Use:** Converts time series → supervised learning (past values → future value)
- **Advantages:** Simple, interpretable, captures short-term dependencies
- **Limitations:** Window size selection is critical; too small = noisy, too large = slow response

Fourier Transform

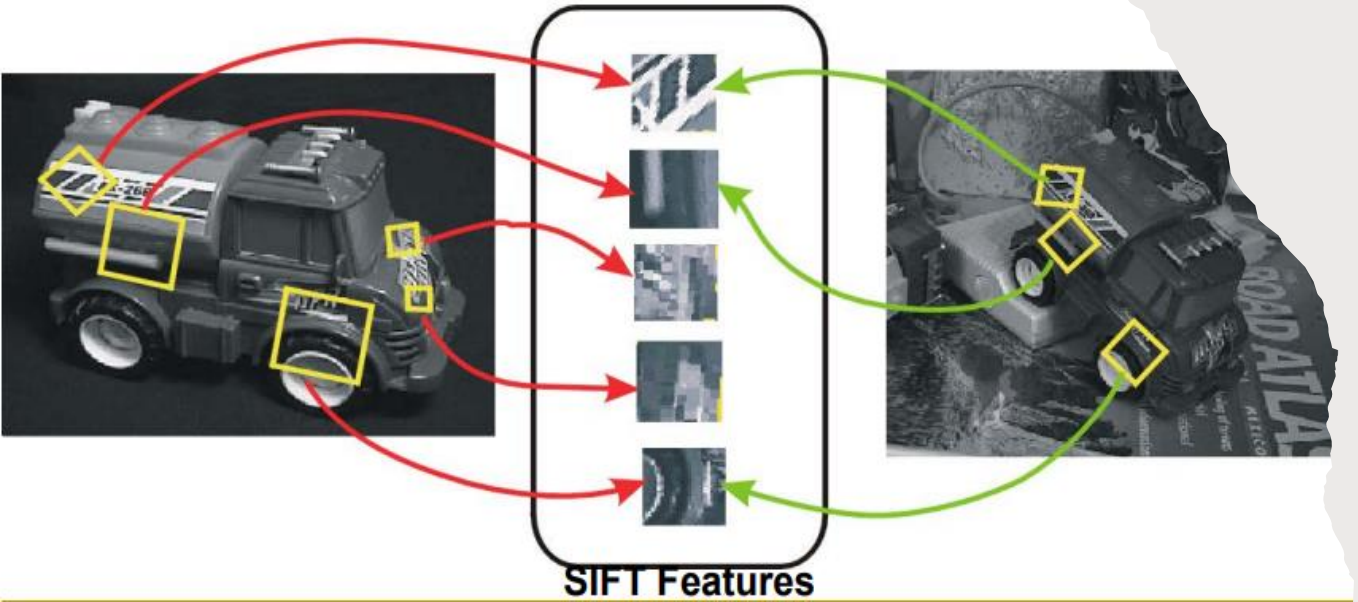
- Decomposes a signal/time series into a sum of **sinusoids (sine & cosine)** with different frequencies
- **Core Idea:** Any periodic signal = combination of simple waves
- **Components:**
- **Frequency** – how fast the wave oscillates
- **Amplitude** – strength of the frequency
- **Phase** – shift in time
- **Mathematical Tool: Fourier Transform (FT) / Discrete Fourier Transform (DFT)**
- **Output:** Frequency-domain representation (time → frequency)
- **Uses:**
- Identify **seasonality & cycles**
- Noise removal (filtering)
- Signal compression



Scale-Invariant Feature Transform (SIFT)

- Feature detection algorithm used to identify **distinctive, scale- and rotation-invariant keypoints** in images
- **Purpose:** Robust feature extraction for **object recognition & image matching**
- **Key Properties:**
- **Scale invariant**
- **Rotation invariant**
- Partially invariant to illumination & affine changes
- **Main Steps:**
- Scale-space extrema detection (Difference of Gaussian)
- Keypoint localization
- Orientation assignment
- Keypoint descriptor generation (128-dimensional vector)

- Image content is tra
coordinates that are invariant to tra
scale, and other imaging parameters



Relational Data Featurization

- Transforming data from **multiple related tables/entities** into features for ML models
- **Core Idea:** Use **relationships (joins, links, graphs)** to enrich features
- **Data Structure:** Tables connected via **foreign keys** or **networks/graphs**
- **Common Techniques:**
- Aggregations (count, sum, mean, max) over related records
- Time-based aggregations (last 7 days, last month)
- Join-based features
- Graph features (degree, centrality)
- **Examples:**
- Customer → total orders, average spend
- User → number of interactions, recency
- **Applications:** Fraud detection, recommendation systems, customer analytics

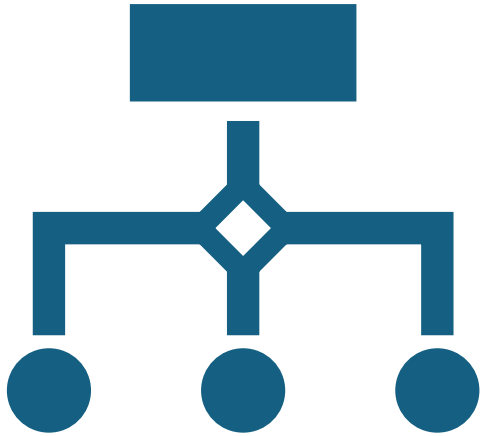
Graph Data Featurization

- Converting **graph-structured data** (nodes, edges, relationships) into numerical features for ML models
- **Graph Elements:** Nodes (entities), Edges (relationships), Weights, Directions
- **Node Features:** Degree, centrality, clustering coefficient, PageRank
- **Edge Features:** Weight, frequency, type of relationship
- **Global Graph Features:** Number of nodes/edges, density, diameter
- **Embedding Methods:** Node2Vec, Deep Walk, Graph SAGE
- **Applications:** Social networks, fraud detection, recommendations, knowledge graphs
- **Advantages:** Captures **structural & relational information**
- **Challenges:** Scalability, dynamic graphs, feature dimensionality

Interaction Variables

- Features created by **combining two or more variables** to capture their **joint effect** on the target
- **Purpose:** Model relationships where the effect of one variable **depends on another**
- **Common Forms:**
- Product terms (X1×X2X_1 \times X_2X1 ×X2)
- Polynomial interactions (X1×X2×X3X_1 \times X_2 \times X_3X1 ×X2 ×X3)
- Categorical interactions (One-Hot × Numerical)
- **Examples:**
- Income × Education
- Price × Discount
- Age × Gender
- **Used In:** Linear/Logistic Regression, Polynomial models, Tree-based models
- **Advantages:** Capture **non-additive effects**, improve model expressiveness
- **Limitations:** Feature explosion, multicollinearity, harder interpretation

Feature Orthogonality



- Features are **orthogonal** when they are **uncorrelated / independent** (dot product = 0)
- **Core Idea:** Each feature provides **unique, non-redundant information**
- Reduces **multicollinearity**
- Improves **model stability & interpretability**
- Faster and more reliable optimization
- **Where Used:** Linear regression, PCA, optimization-based models
- **How to Achieve:**
- Feature selection
- PCA / SVD (creates orthogonal components)
- Regularization (Ridge)
- **Example:** PCA components are mutually orthogonal