UNIVERSITÀ DEGLI STUDI DI TRIESTE

Class 4: Python ML Project Feature Engineering

Master Course:

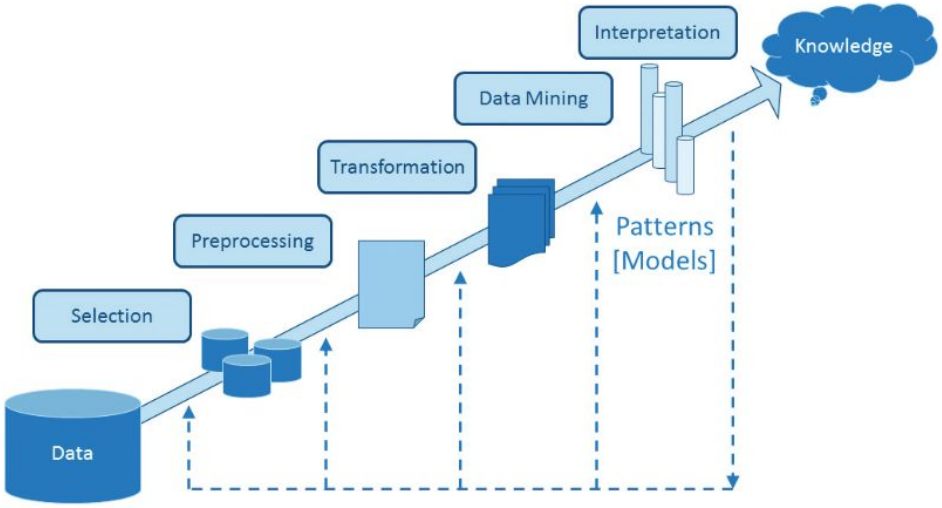
Data-driven Systems Engineering (ML Operations)

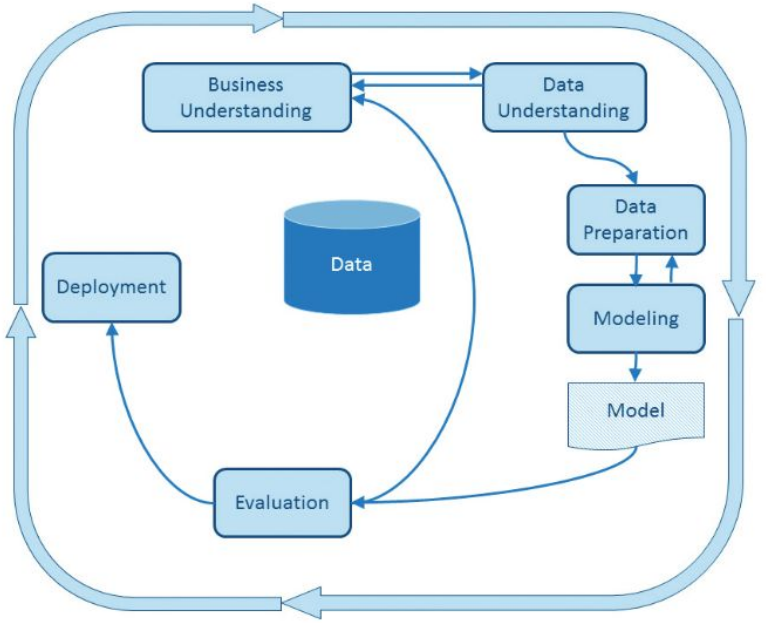
440MI and 305SM

Agenda & Learning Goals

- Feature Engineering
- Why Feature Engineering matters?
- Creating new variable from raw data
- Handling Categorical and Numerical Features
- Scaling and Transformations
- Feature Pipelines and MLOps Integration

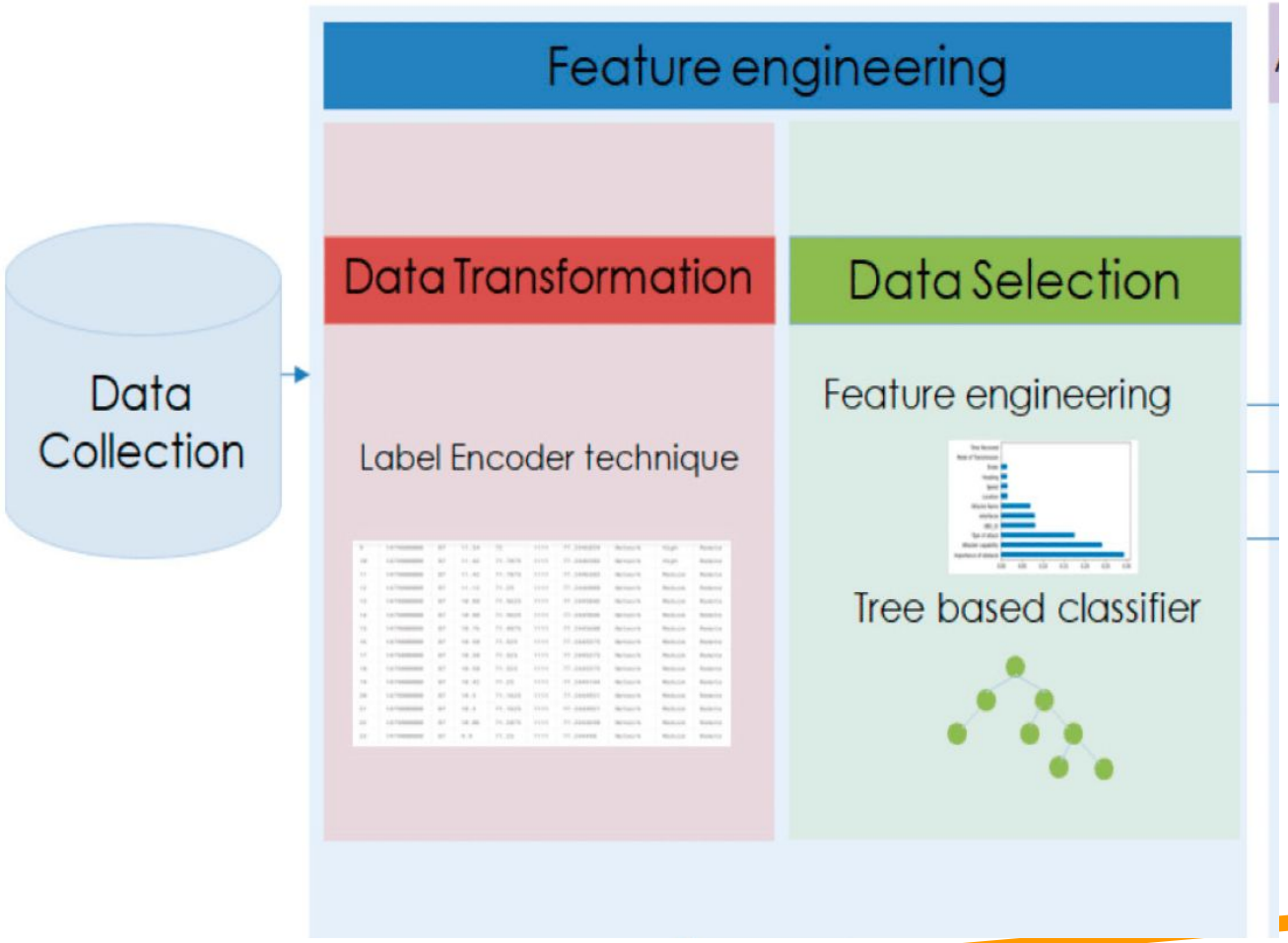
KDD vs CRISP-DM





Cross-Industry Standard Process for Data Mining

Feature Engineering?



Feature Engineering

- Definition: Feature engineering is the process of creating, transforming, and selecting variables to improve model performance and interpretability (Zheng & Casari, 2018).
- Role in MLOps: In production systems, feature engineering is not only a data science task but also an engineering challenge — ensuring reproducibility, scalability, and consistency across training and deployment (Hummer et al., 2019).
- Historical perspective: The performance of ML models often depends more on the quality of features than on the choice of algorithm (Domingos, 2012).

Feature Engineering

Feature Selection:

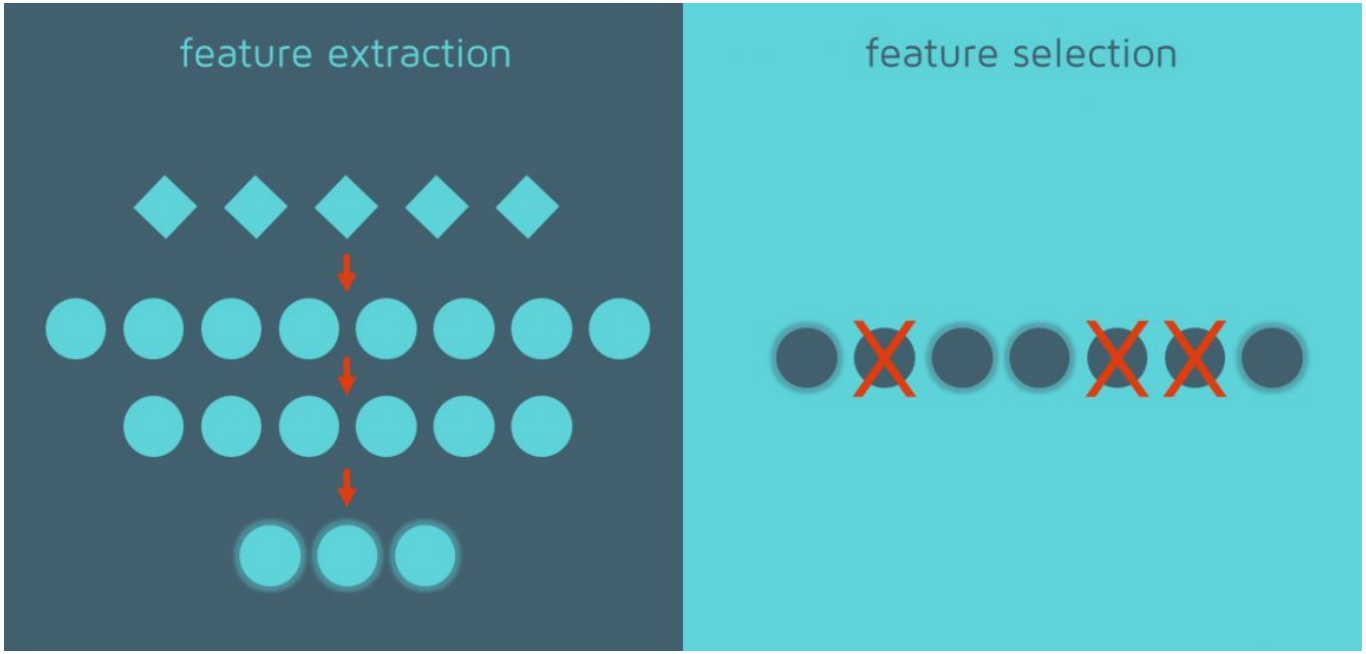
- Process of choosing a subset of existing features.
- Removes redundant, irrelevant, or noisy variables.
- Techniques:
  - Filter methods (correlation, chi-square, mutual information).
  - Wrapper methods (e.g., recursive feature elimination).
  - Embedded methods (Tree-based importance).
- **Goal**: Keep the most informative features → reduce dimensionality while preserving interpretability.

Feature Engineering:

Feature Extraction:

- Process of transforming original features into new ones.
- Creates lower-dimensional representations.
- Techniques:
  - Linear: PCA, LDA.
  - Nonlinear: Kernel PCA, t-SNE, autoencoders.
  - Domain-driven: Fourier/MFCC (signal), embeddings (NLP).
- **Goal**: Compress information into a new feature space, often at the cost of interpretability.

Feature Engineering:



Feature Engineering:

Data Cleaning:

- Garbage in → garbage out (Redman, 1998).
- Data quality directly impacts model performance.
- Typical Tasks:
  - Handling missing value
    - Deletion (listwise, pairwise).
    - Imputation (mean, median, mode, model-based).
  - Outlier detection & treatment
    - Z-scores, IQR, robust scaling.
  - Noise reduction
    - Smoothing (binning, moving averages).
- Data consistency
  - Correcting typos, harmonizing formats (e.g., “M” vs. “Male”).

Feature Engineering?

Step	Goal	Main Actions	Examples
Data Cleaning	Fix problems in the raw data	Handle missing values, remove duplicates, correct errors, manage outliers	Replace missing ages with median, remove duplicated customer IDs
Data Curation	Organize & document datasets for long-term usability	Collect from sources, integrate, add metadata, version control, ensure quality	Build a labeled dataset with clear documentation, provenance, and licensing
Feature Selection	Keep the most relevant features	Filter (correlation, MI), wrapper (RFE), embedded (Lasso, trees)	From 100 variables, keep the 10 most predictive
Feature Extraction	Transform data into new representations (often lower-dim)	PCA, t-SNE, embeddings, Fourier/MFCCs, CNN features	Convert 28×28 images into 50 latent dimensions with PCA
Feature Engineering	Create new features using domain knowledge	Transform, bin, combine, time-based features, ratios	From a timestamp → “day of week,” “is weekend,” “hour”

Curation → Cleaning → Engineering → Extraction → Selection → Modeling

Creating New Variables from Raw Data

- Feature Construction: Deriving higher-level features from raw attributes (Heaton, 2016).
  - Temporal data: extracting weekday/weekend, cyclic encoding of hour of day.
  - Text data: n-grams, TF-IDF, embeddings.
  - Time-series: moving averages, Fourier/MFCC features.
- Case study: In predictive maintenance, vibration signals are transformed into frequency-domain features (Randall, 2011).

Handling Categorical and Numerical Features

- Categorical:
  - One-hot encoding, ordinal encoding, target encoding (Micci-Barreca, 2001).
  - Challenges with high cardinality (e.g., ZIP codes, user IDs).
- Numerical:
  - Transformations for skewness (log, Box–Cox; Osborne, 2010).
  - Outlier treatment and winsorization.

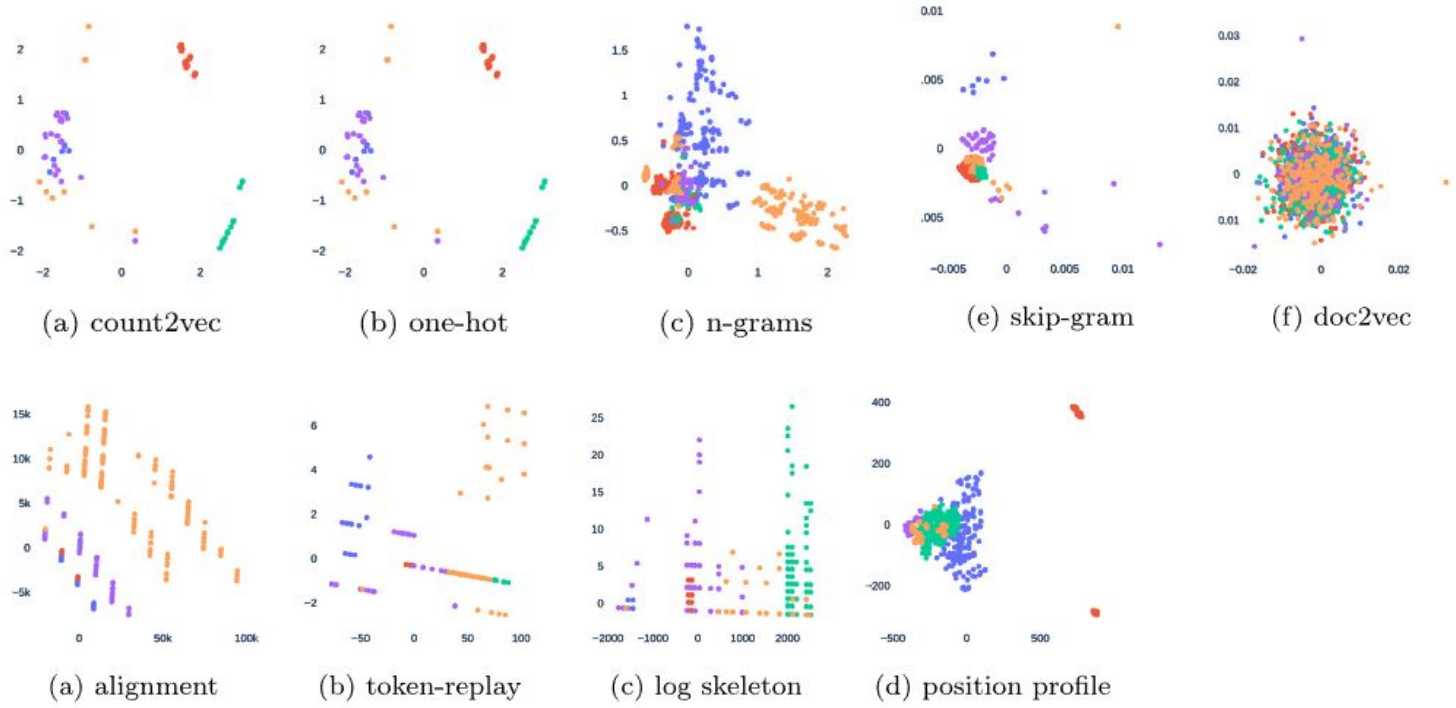
Scaling and Transformations

- Scaling:
  - Standardization (z-score), min–max scaling, robust scaling.
  - Essential for distance-based models and PCA (Jolliffe & Cadima, 2016).
- Transformations:
  - Polynomial and interaction features (Friedman, Hastie & Tibshirani, 2001).
  - Normalization for neural networks.

Encoding vs Embedding

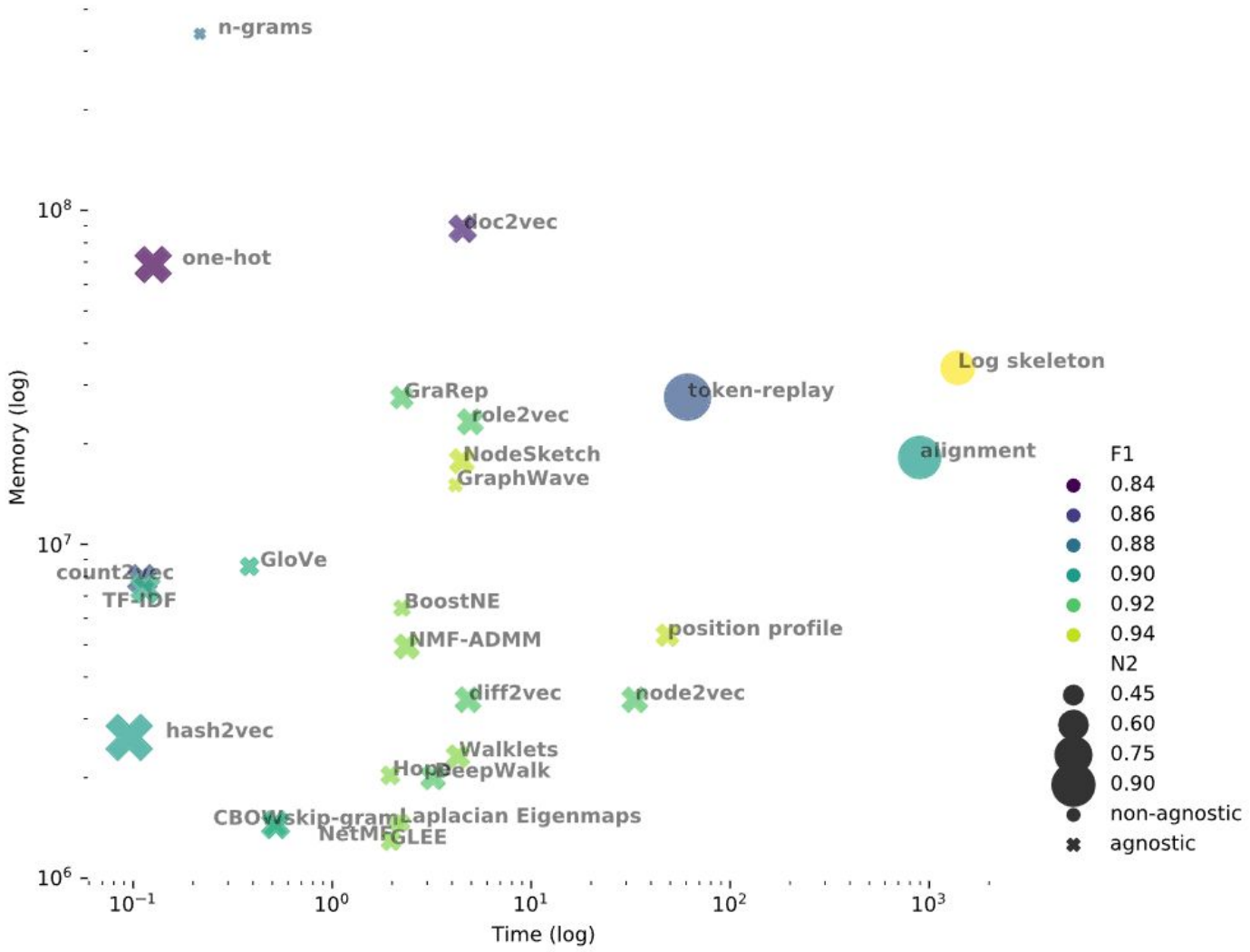
Aspect	Encoding	Embedding
Definition	Convert categories to numbers (rules-based)	Learn dense vectors (model-based)
Dimensionality	High (e.g., one-hot)	Low (compressed)
Sparsity	Sparse (lots of zeros)	Dense (all values used)
Semantic Info	No relationships preserved	Captures similarities/meaning
Examples	One-hot, label encoding	word2vec, BERT, entity embeddings

Encoding vs Embedding



TAVARES, Gabriel M., et al. Trace encoding in process mining: A survey and benchmarking. *Engineering Applications of Artificial Intelligence*, 2023, 126: 107028.

Encoding vs Embedding



Feature Pipelines and MLOps Integration

- Reproducibility: Feature pipelines must be serialized (e.g., via Pickle, Joblib, MLflow) to ensure training–inference consistency (Schelter et al., 2019).
- Feature Stores: Industrial systems often use centralized repositories (Uber's Michelangelo, TFX) to guarantee consistency (Hammad et al., 2021).
- Pitfalls: Data leakage, improper scaling, inconsistent transformations between training and production.

Summary & Key Takeaways

- Discussion:
  - How does feature engineering interact with AutoML?
  - What are the trade-offs between manual feature design and automated feature generation (e.g., Deep Feature Synthesis)?
- Feature quality > Model complexity (Domingos, 2012 – CACM)
- Creating new variables from raw data improves signal (Heaton, 2016).
- Categorical & numerical handling: choose encodings and transformations carefully
- Scaling & transformations are critical for PCA, kNN, and neural nets (Jolliffe & Cadima, 2016).
- Pipelines in MLOps ensure reproducibility and prevent data leakage (Schelter et al., 2019).
- Feature stores enable consistent training/inference across teams and systems (Hammad et al., 2021).

References:

- Zheng, A., & Casari, E. (2018). Feature Engineering for Machine Learning. O'Reilly.
- Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. Communications of the ACM, 55(10).
- Hummer, W., et al. (2019). "Automated Data Science in MLOps: Towards Reproducible Model Pipelines." arXiv:1907.04418.
- Heaton, J. (2016). An Empirical Analysis of Feature Engineering for Predictive Modeling. In IEEE SoutheastCon.
- Randall, R. B. (2011). Vibration-based Condition Monitoring. Wiley.
- Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A Review and New Perspectives. IEEE TPAMI.
- Zheng & Casari (2018). Feature Engineering for Machine Learning.
- Schelter, S., et al. (2019). On Challenges in Machine Learning Model Management. IEEE Data Engineering Bulletin.
- Hammad, M., et al. (2021). The Feature Store: A Data Engineering Infrastructure for Machine Learning. arXiv:2107.13000.
- Jolliffe, I. T., & Cadima, J. (2016). Principal Component Analysis: A Review and Recent Developments. Philosophical Transactions of the Royal Society A.
- Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning. Springer.
  Micci-Barreca, D. (2001). A Preprocessing Scheme for High-Cardinality Categorical Attributes in Classification and Prediction Problems. SIGKDD Explorations.
- Osborne, J. (2010). Improving Your Data Transformations: Applying the Box-Cox Transformation. Practical Assessment, Research, and Evaluation