Feature Selection: A Data Perspective

Zachary Jacokes

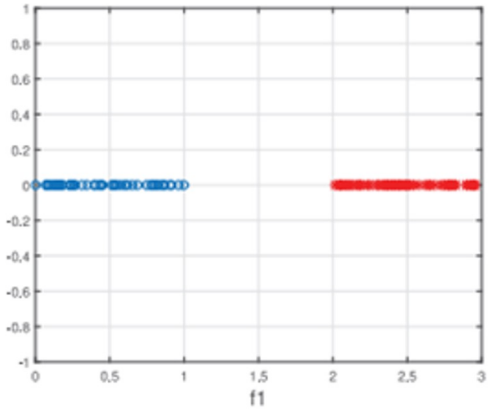
DS7406

9/21/2022

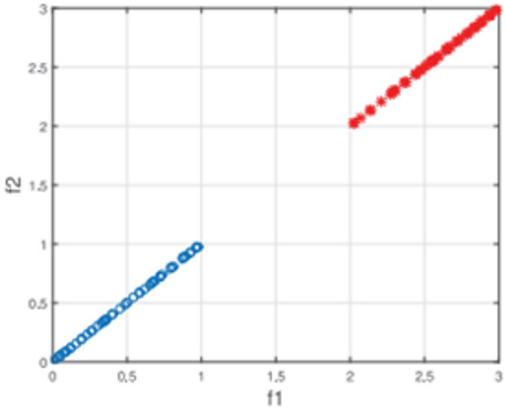
General Overview

- High-dimensional data common in modern datasets
- Curse of dimensionality: p > n
  - Data become sparser in high-dim space
  - Algorithms less stable in high-dim space
  - More features = higher chance of overfitting training data
  - Storage requirements and computational costs
- Feature engineering can help
  - Feature extraction: project high-dim features to new low-dim space
  - Feature selection: directly select relevant features
- Trade-off between interpretability and feature relevance

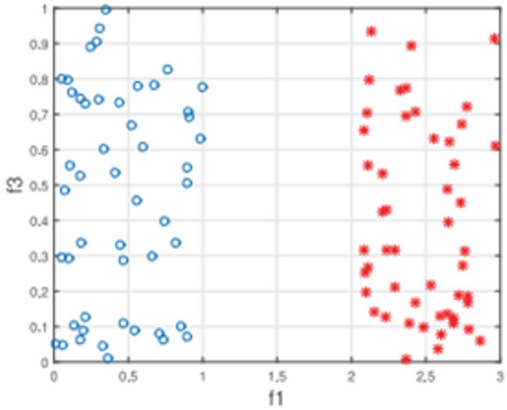
Relevant, Redundant, Irrelevant Features



(a) relevant feature f_1



(b) redundant feature f_2



(c) irrelevant feature f_3

Removing f_2 and f_3 should not impact model performance

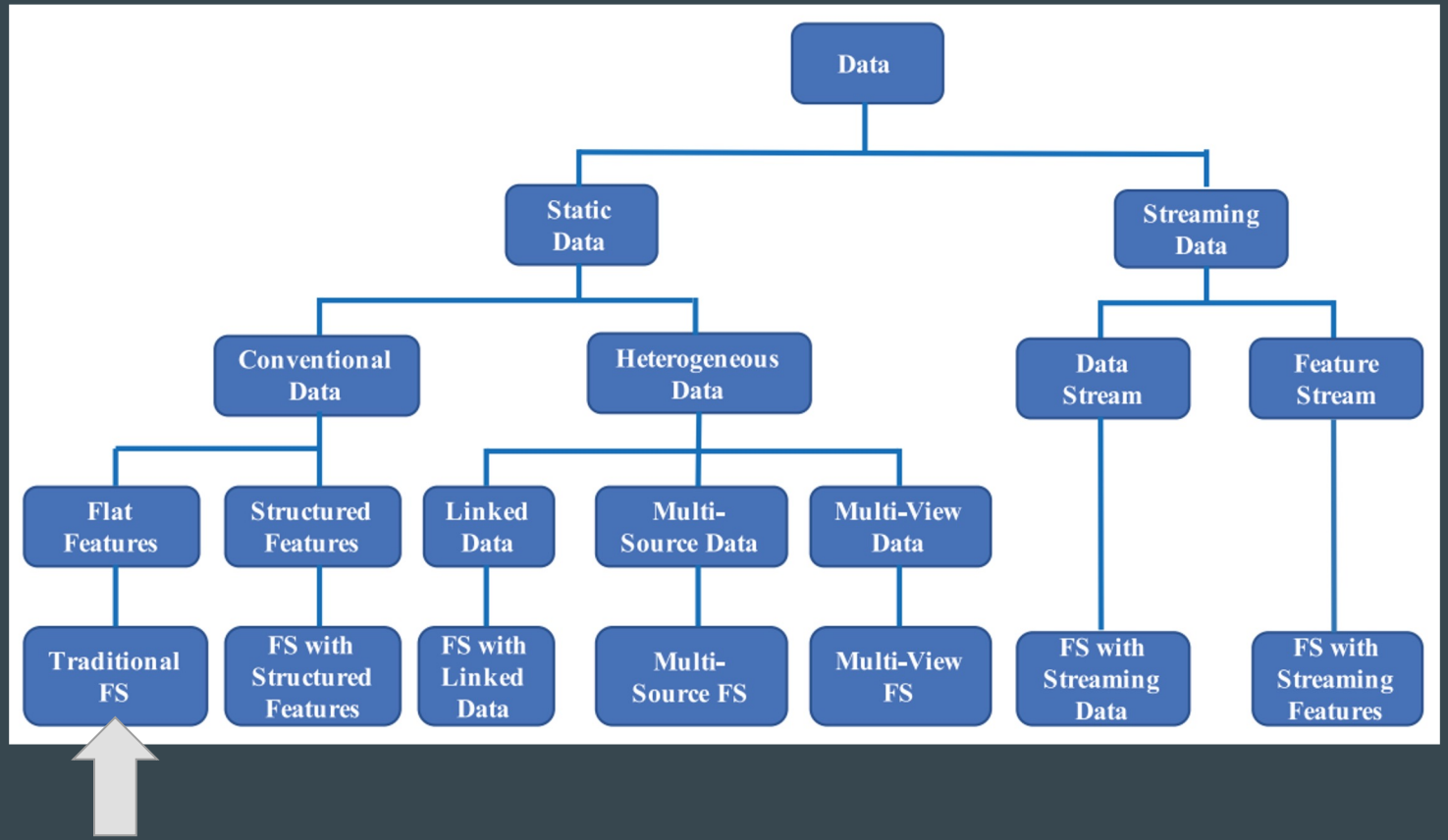
Feature Selection: Data Categorization

- Supervised
  - Class labels available
  - Best subset of features determined based on how effectively they discriminate between classes
- Unsupervised
  - No class labels
  - Clusters data based on criteria other than labels
- Semi-supervised
  - Combination of both
  - Real-world datasets commonly have labeled and unlabeled data

Feature Selection: Strategy

- Wrapper method
  - Check performance of model to evaluate quality of feature selection
  - Search for subset of features, evaluate performance
  - Repeat iteratively until desired performance achieved
  - Rarely used: search space for d features is 2^d
- Filter method
  - Use characteristics of the data to determine feature importance
  - Ranked based on various criteria, low-ranked features are filtered out
    - Discriminative ability, feature correlation, mutual information, manifold structure preservation, ability to reconstruct original data
- Embedded method
  - Combination of wrapper and filter methods
  - Interaction between learning algorithm and feature importance
  - Force feature coefficients to be small/zero
  - Common embedding method: regularization (LASSO, ridge, elastic net)

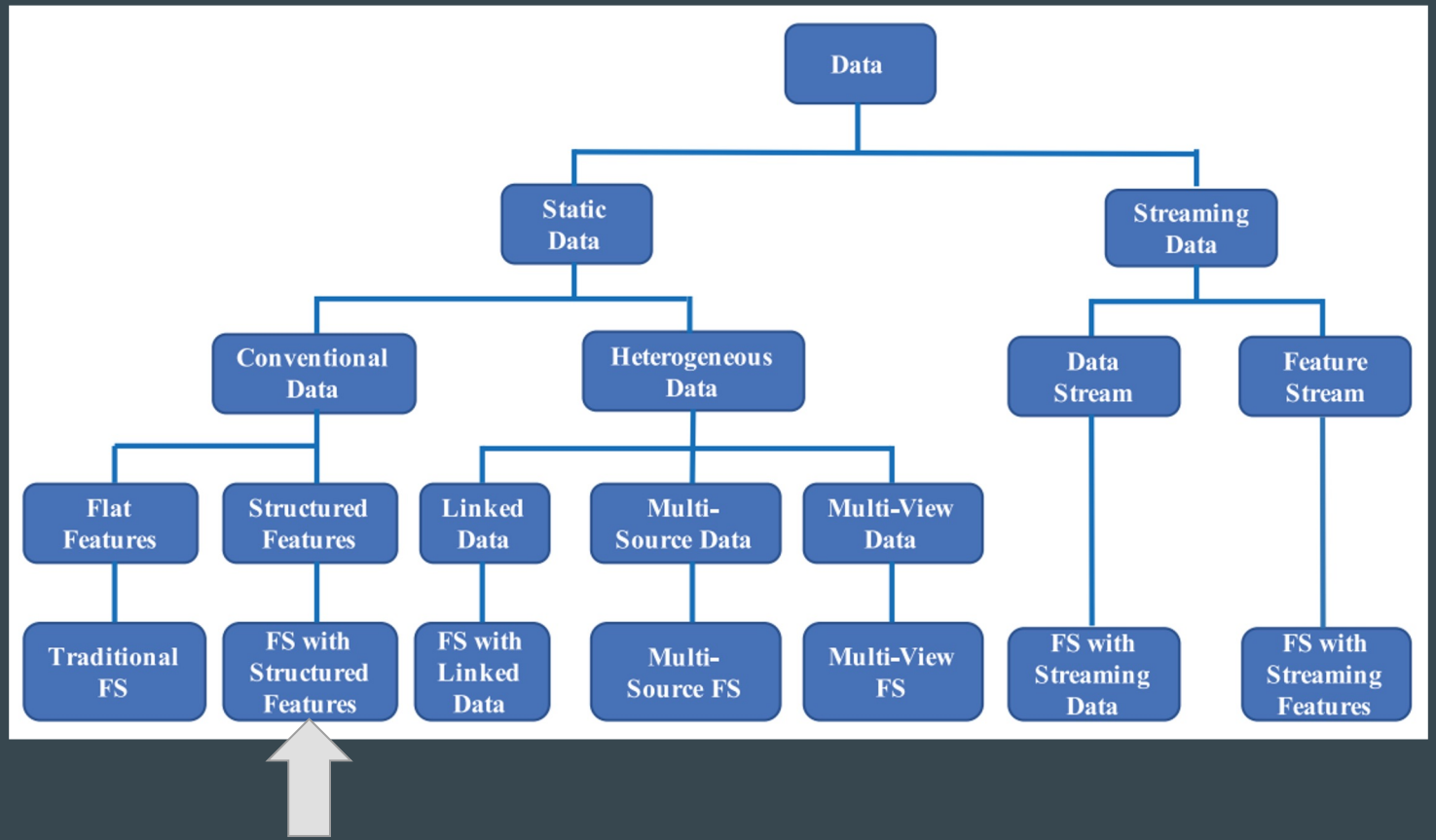
Data Types


Data
Static Data
Streaming Data
Conventional Data
Heterogeneous Data
Data Stream
Feature Stream
Flat Features
Structured Features
Linked Data
Multi-Source Data
Multi-View Data
Traditional FS
FS with Structured Features
FS with Linked Data
Multi-Source FS
Multi-View FS
FS with Streaming Data
FS with Streaming Features

Feature Selection: Traditional Data

- Similarity-based methods
  - Ability to preserve similarity to original data structure
  - Struggles with feature redundancy; often selects highly-correlated features
    - Laplacian score, SPEC, Fisher score, Trace Ratio Criterion, ReliefF
- Information-theoretical-based methods
  - Maximize feature relevance, minimize feature redundancy/information shared
  - Relevance measured by correlation to class labels
  - Mostly used in supervised algorithms; data must be discrete/discretized
    - Mutual information maximization, conditional infomax feature extraction, joint mutual information, informative fragments
- Sparse-learning-based methods
  - Embedded feature selection; good interpretability
  - Heavy computational cost
    - REFS, l_p-norm regularizer, $l_{p,q}$-norm regularizer, nonnegative spectral analysis
- Statistical-based methods
  - Filter based; typically analyze features individually thus ignore feature redundancy
  - Simple, low computational cost, sometimes used as a pre-processing step
    - Chi-square, Gini index, low-variance

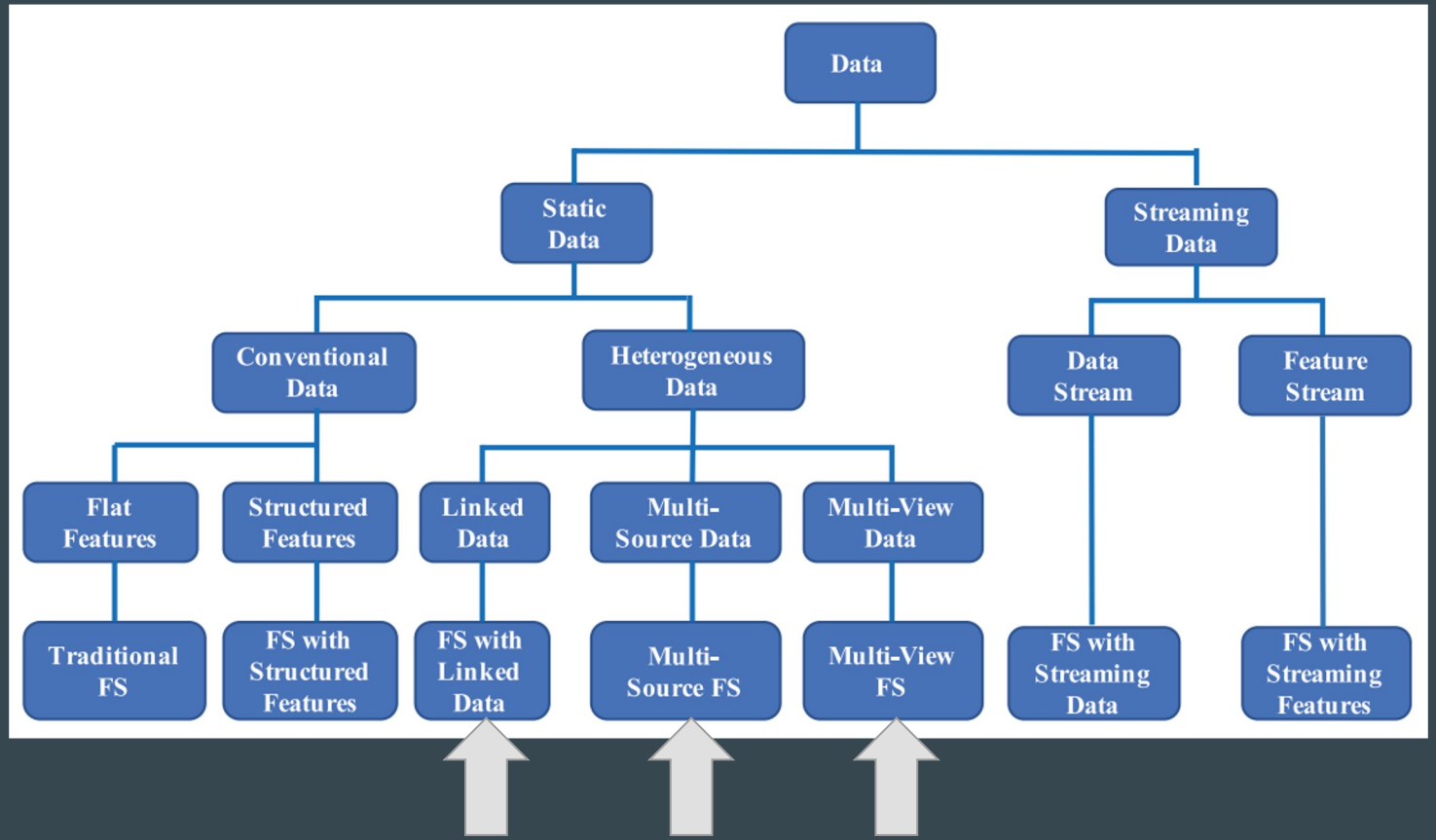
Structured Features



Feature Selection: Structured Features

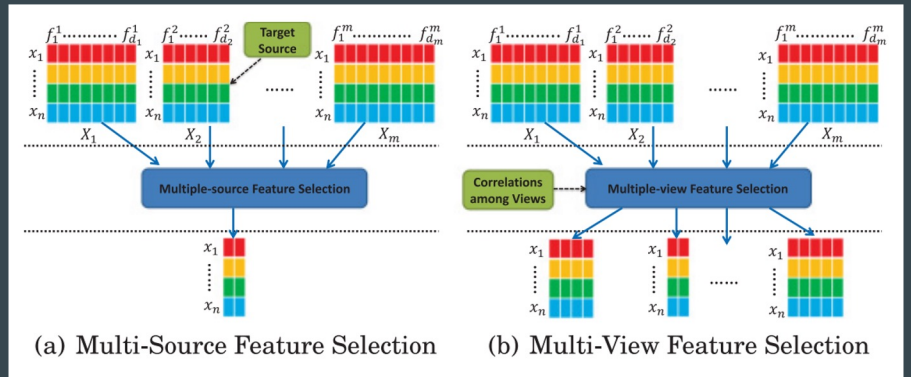
- Sometimes features are not independent; structure is important to quantify
- Three types:
  - Groups; i.e. genes with similar functions acting together
  - Trees; i.e. facial recognition, root = whole face, pixels = leaves, etc.
  - Graphs; i.e. natural language processing, word = feature, relationships = synonym/antonyms/etc., undirected graph
- Feature selection comparable to sparse-learning (regularization: LASSO)
  - Specific structure type taken into account as prior info
- Difficult task to automatically infer data structure
- Computationally expensive

Heterogeneous Data

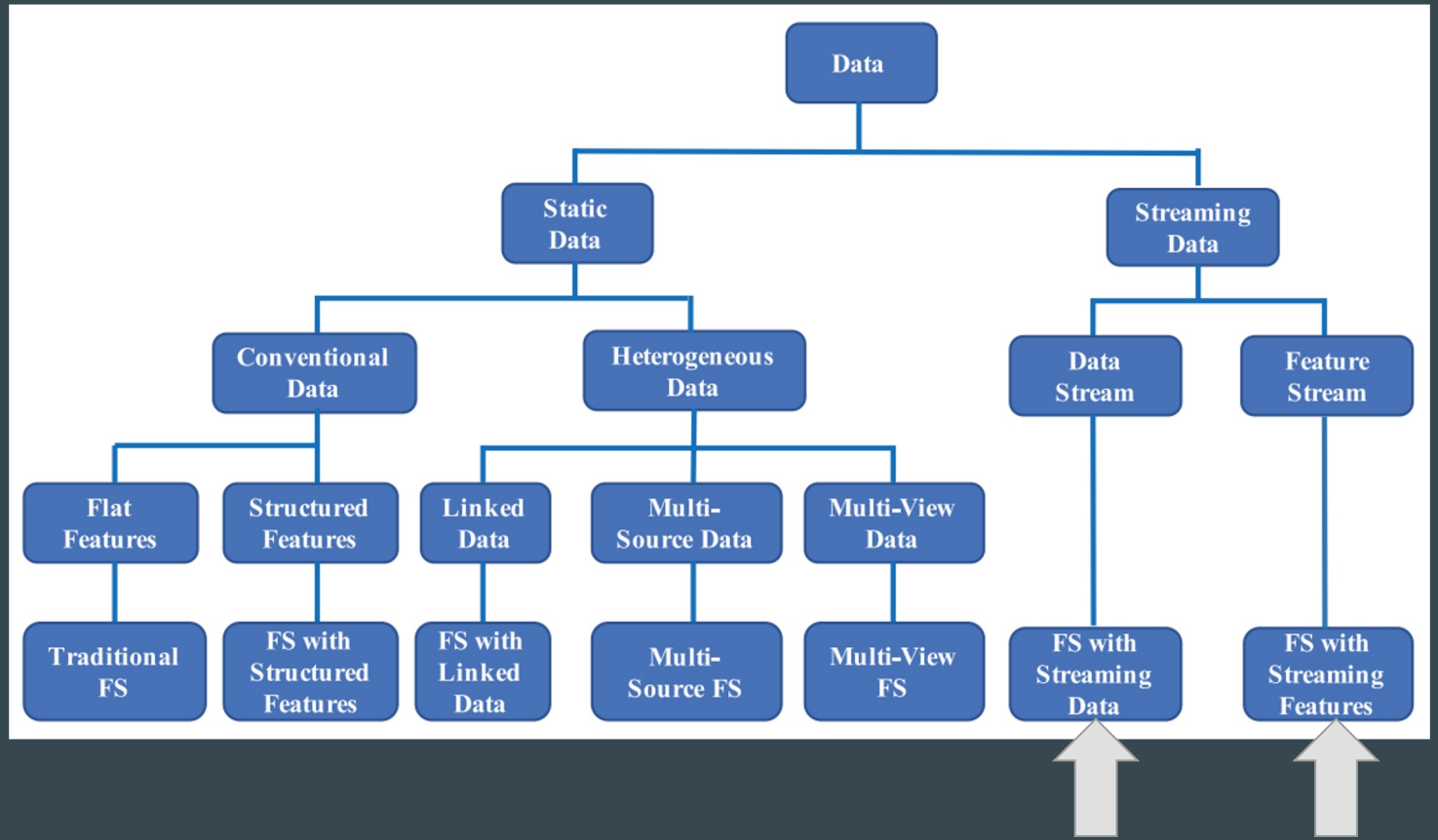


Feature Selection: Heterogeneous Data

- Combined data from different sources
- Linked data
  - Social media: user-post interactions; biological systems: protein interactions
  - Handled similarly to graph feature selection
- Multi-source data
  - Features selected from original feature space; ignores correlations between sources
- Multi-view data
  - Features selected from all views simultaneously; exploits relationships from different sources



Streaming Data



Feature Selection: Streaming Data

- Streaming data is constantly refreshing
  - Data size unknown; sometimes infinite
  - Single snapshot of streaming data desired to save memory
- Twitter: newly created slang words = new potential features
- Feature streams
  - Instances are constant, candidate features arrive one by one
  - Algorithm either accepts or rejects newly arrived feature
  - Discards existing features to make room (sometimes)
- Data streams
  - Instances and candidate features arriving constantly
  - Algorithm accepts/rejects new instance
  - Discards existing data streams (sometimes)

Challenges

- Scalability
  - Most methods = quadratic/cubic time complexity; inefficient for large datasets
  - Some distributed frameworks exist, but more work needs to be done
- Stability
  - Defined as sensitivity of algorithm to fluctuations in training data
  - Affected by the underlying structure of datasets
  - Much easier to assess in labeled datasets
    - Clustering in streaming contexts can change entirely when presented with new data
- Model Selection
  - Must specify number of features to be selected, but optimal number is unknown
  - Grid search to assess accuracy at different feature numbers - computationally expensive

Conclusions and Recommendation

- Effective preprocessing step for dimensionality reduction
- Essential for many ML applications
- Greater understanding about data characteristics = better features
  - Also introduces greater complexity into the process
- Authors provide a repository for many of the discussed algorithms
- Limited commentary on how to improve
- Lack of detail on deep learning/NN pruning, given the other papers