Feature Selection

Selecting a useful subset from all the features

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization
- Removal of features can increase (relative) margin (and generalization)

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization
- Removal of features can increase (relative) margin (and generalization)
- Reduces data set and resulting model size

Feature Selection

Selecting a useful subset from all the features

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization
- Removal of features can increase (relative) margin (and generalization)
- Reduces data set and resulting model size
- Note: Feature Selection is different from Feature Extraction
  - The latter transforms original features to get a small set of new features
  - More on feature extraction when we cover **Dimensionality Reduction**

Feature Selection Methods

- Methods agnostic to the learning algorithm

Feature Selection Methods

- Methods agnostic to the learning algorithm
  - Preprocessing based methods
    - E.g., remove a binary feature if it's ON in very few or most examples

Feature Selection Methods

- Methods agnostic to the learning algorithm
  - Preprocessing based methods
    - E.g., remove a binary feature if it's ON in very few or most examples
  - Filter Feature Selection methods
    - Use some ranking criteria to rank features
    - Select the top ranking features

Feature Selection Methods

- Methods agnostic to the learning algorithm
  - Preprocessing based methods
    - E.g., remove a binary feature if it's ON in very few or most examples
  - Filter Feature Selection methods
    - Use some ranking criteria to rank features
    - Select the top ranking features
- Wrapper Methods (keep the learning algorithm in the loop)

Feature Selection Methods

- Methods agnostic to the learning algorithm
  - Preprocessing based methods
    - E.g., remove a binary feature if it's ON in very few or most examples
  - Filter Feature Selection methods
    - Use some ranking criteria to rank features
    - Select the top ranking features
- Wrapper Methods (keep the learning algorithm in the loop)
  - Requires repeated runs of the learning algorithm with different set of features

Feature Selection Methods

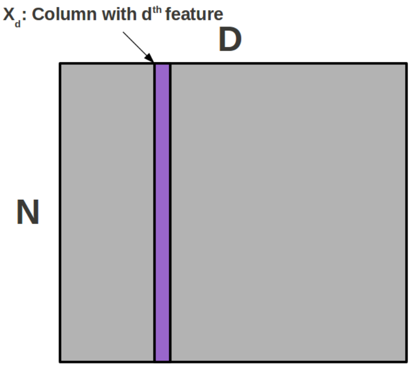
- Methods agnostic to the learning algorithm
  - Preprocessing based methods
    - E.g., remove a binary feature if it's ON in very few or most examples
  - Filter Feature Selection methods
    - Use some ranking criteria to rank features
    - Select the top ranking features
- Wrapper Methods (keep the learning algorithm in the loop)
  - Requires repeated runs of the learning algorithm with different set of features
  - Can be computationally expensive

Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods

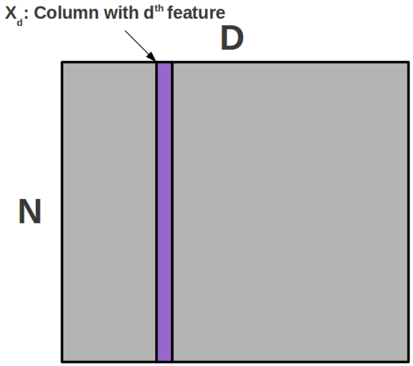
Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods



Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods

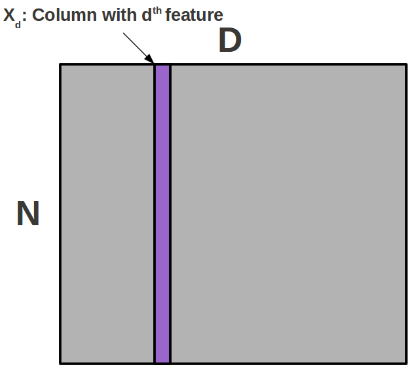


- **Correlation Critera:** Rank features in order of their correlation with the labels

$$R(X_d, Y) = \frac{cov(X_d, Y)}{\sqrt{var(X_d)var(Y)}}$$

Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods



- **Correlation Critera:** Rank features in order of their correlation with the labels

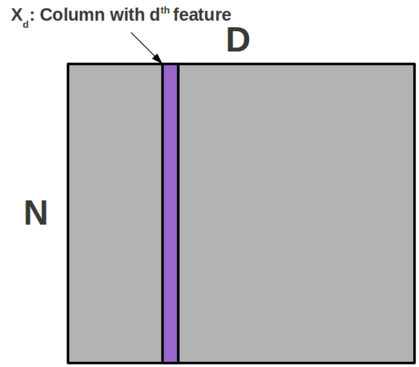
$$R(X_d, Y) = \frac{cov(X_d, Y)}{\sqrt{var(X_d)var(Y)}}$$

- **Mutual Information Criteria:**

$$MI(X_d, Y) = \sum_{X_d \in \{0,1\}} \sum_{Y \in \{-1,+1\}} P(X_d, Y) \frac{\log P(X_d, Y)}{P(X_d)P(Y)}$$

Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods



- **Correlation Critera:** Rank features in order of their correlation with the labels

$$R(X_d, Y) = \frac{cov(X_d, Y)}{\sqrt{var(X_d)var(Y)}}$$

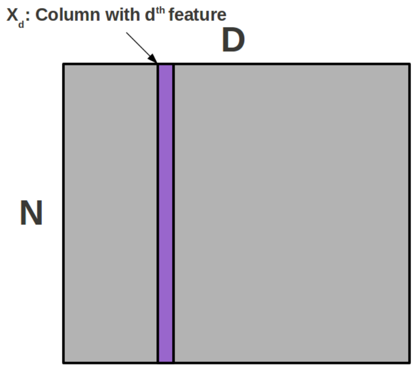
- **Mutual Information Criteria:**

$$MI(X_d, Y) = \sum_{X_d \in \{0,1\}} \sum_{Y \in \{-1,+1\}} P(X_d, Y) \frac{\log P(X_d, Y)}{P(X_d)P(Y)}$$

  - High mutual information mean high relevance of that feature

Filter Feature Selection

- Uses heuristics but is much faster than wrapper methods



- **Correlation Critera:** Rank features in order of their correlation with the labels

$$R(X_d, Y) = \frac{cov(X_d, Y)}{\sqrt{var(X_d)var(Y)}}$$

- **Mutual Information Criteria:**

$$MI(X_d, Y) = \sum_{X_d \in \{0,1\}} \sum_{Y \in \{-1,+1\}} P(X_d, Y) \frac{\log P(X_d, Y)}{P(X_d)P(Y)}$$

  - High mutual information mean high relevance of that feature
  - Note: These probabilities can be easily estimated from the data

Wrapper Methods

- Two types: Forward Search and Backward Search

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature
    - Stop when selected the desired number of features
  - **Backward Search**

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature
    - Stop when selected the desired number of features
  - **Backward Search**
    - Start with all the features

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature
    - Stop when selected the desired number of features
  - **Backward Search**
    - Start with all the features
    - Greedily remove the least relevant feature

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature
    - Stop when selected the desired number of features
  - **Backward Search**
    - Start with all the features
    - Greedily remove the least relevant feature
    - Stop when selected the desired number of features

Wrapper Methods

- Two types: Forward Search and Backward Search
  - **Forward Search**
    - Start with no features
    - Greedily include the most relevant feature
    - Stop when selected the desired number of features
  - **Backward Search**
    - Start with all the features
    - Greedily remove the least relevant feature
    - Stop when selected the desired number of features
  - Inclusion/Removal criteria uses cross-validation

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:
    - Estimate model's error on feature set $\mathcal{F} \bigcup f$ (using cross-validation)

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:
    - Estimate model's error on feature set $\mathcal{F} \bigcup f$ (using cross-validation)
  - Add f with lowest error to $\mathcal{F}$
- **Backward Search**
  - Let $\mathcal{F} = \{\text{all features}\}$

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:
    - Estimate model's error on feature set $\mathcal{F} \bigcup f$ (using cross-validation)
  - Add f with lowest error to $\mathcal{F}$
- **Backward Search**
  - Let $\mathcal{F} = \{\text{all features}\}$
  - While not reduced to desired number of features
  - For each feature $f \in \mathcal{F}$:

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:
    - Estimate model's error on feature set $\mathcal{F} \bigcup f$ (using cross-validation)
  - Add f with lowest error to $\mathcal{F}$
- **Backward Search**
  - Let $\mathcal{F} = \{\text{all features}\}$
  - While not reduced to desired number of features
  - For each feature $f \in \mathcal{F}$:
    - Estimate model's error on feature set $\mathcal{F} \backslash f$ (using cross-validation)

Wrapper Methods

- **Forward Search**
  - Let $\mathcal{F} = \{\}$
  - While not selected desired number of features
  - For each unused feature f:
    - Estimate model's error on feature set $\mathcal{F} \bigcup f$ (using cross-validation)
  - Add f with lowest error to $\mathcal{F}$
- **Backward Search**
  - Let $\mathcal{F} = \{\text{all features}\}$
  - While not reduced to desired number of features
  - For each feature $f \in \mathcal{F}$:
    - Estimate model's error on feature set $\mathcal{F} \backslash f$ (using cross-validation)
  - Remove f with lowest error from $\mathcal{F}$