Feature Selection

He He

CDS, NYU

Feb 16, 2021

Complexity of Hypothesis Spaces

Trade-off between approximation error and estimation error:

- Bigger $\mathcal{F}$: better approximation but can overfit
- Smaller $\mathcal{F}$: less likely to overfit but can be far from the "true" model

To control the "size" of $\mathcal{F}$, we need some measure of its complexity:

- Number of variables / features
- Depth of a decision tree
- Degree of polynomial

General Approach to Control Complexity

1. Learn a sequence of models varying in complexity from the training data

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \mathcal{F}_n \cdots \subset \mathcal{F}$$

   Example: Polynomial Functions
   - $\mathcal{F} = \{\text{all polynomial functions}\}$
   - $\mathcal{F}_d = \{\text{all polynomials of degree } \leqslant d\}$

2. Select one model according to some "score" (e.g. validation error)

Feature Selection

Nest sequence of hypothesis spaces: $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \mathcal{F}_n \cdots \subset \mathcal{F}$

- $\mathcal{F} = \{\text{linear functions using all features}\}$
- $\mathcal{F}_d = \{\text{linear functions using fewer than } d \text{ features}\}$

Best subset selection:

- Choose the subset of features that is best according to the score (e.g. validation error)
  - Example with 2 features: Train models using $\{\}$, $\{X_1\}$, $\{X_2\}$, $\{X_1, X_2\}$, respectively
- No efficient algorithm for large number of features

Greedy Selection Methods

Forward selection:

1. Start with an empty set of features S
2. For each feature i not in S
   - Learn a model using features $S \cup i$
   - Compute score of the model: α_i
3. Find the candidate feature with the highest score: $j = \arg\max_i \alpha_i$
4. If α_j improves the current best score, add feature j: $S \leftarrow S \cup j$; return S otherwise.

Backward Selection:

- Start with all features and remove the one that maximally improves the score in each iteration

Complexity Penalty

Goal: score a subset of features based on its size and prediction performance

Subset selection aims to find the smallest subset that minimizes the validation error. Can we formalize the objective?

$$\text{score}(S) = \text{training\_loss}(S) + \lambda |S| \tag{1}$$

λ balances the training loss and the number of features used:

- Adding 1 feature must be justified by at least λ improvement in training loss
- Larger λ penalizes complex models more heavily
- Different values gives different criterion (e.g. AIC, BIC)

Summary

- Number of features as a measure of complexity
- General approach to feature selection
  - Define a score that balances training error and complexity
  - Find the subset of features that maximize the score
- In practice, forward selection is usually used
- Exercise caution when interpreting the features