Fundamentals of Learning

Kaiqi Zhao
The University of Auckland

Slides are partially based on the materials from the University of British Columbia

Validation Error

Validation Error

- How do we decide the tree depth?
- We care about test error, but we can't look at test data
- So what do we do?
- One answer: Use part of the training data to approximate test error
- Split training examples into training set and validation set:
  - Train model based on the training data
  - Test model based on the validation data.

Validation Error

$$X = \left[\begin{array}{cccccc|c} 0.0 & 0.7 & 0.0 & 0.3 & 0.0 & 0.00 & \dots \\ 0.3 & 0.7 & 0.0 & 0.6 & 0.0 & 0.01 & \dots \\ 0.0 & 0.0 & 0.0 & 0.8 & 0.0 & 0.00 & \dots \\ \hline 0.3 & 0.7 & 1.2 & 0.0 & 0.1 & 0.01 & \dots \\ 0.3 & 0.0 & 1.2 & 0.3 & 0.1 & 0.01 & \dots \end{array}\right] \qquad y = \left[\begin{array}{c} 1 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{array}\right] \begin{array}{l} \} \text{ training} \\ \\ \\ \} \text{ validation} \end{array}$$

1. Train: $model = train(X_{train}, Y_{train})$
2. Predict: $\hat{y} = predict(model, X_{validate})$
3. Validate: $error = \Sigma(\hat{y} \neq y_{validate})$

- Note: if examples are ordered, split should be random

Validation Error

- IID data: validation error is an unbiased approximation of test error

$$\mathbb{E}[E_{valid}] = \mathbb{E}[E_{test}]$$

- Course analogy
  - You have 2 practice exams
  - You hide one exam, and spend a lot of time working through the other
  - You then do the other practice exam, to see how well you'll do on the test

- We can use the validation error for choosing hyperparameters

Parameters and Hyperparameters

- The decision tree rule values are called parameters
  - Parameters control how well we fit a dataset
  - We train a model by trying to find the best parameters on training data

- The maximum decision tree depth is a called a hyperparameter
  - Hyperparameters control how complex our model is
  - We can't train a hyperparameter
    - You can always fit training data better by making the model more complicated
  - We validate a hyperparameter using a validation score

Choosing Hyperparameters with Validation Sets

- So to choose a good value of depth (hyperparameter), we could?
  - Try a depth-1 decision tree, compute validation error
  - Try a depth-2 decision tree, compute validation error
  - Try a depth-3 decision tree, compute validation error
  - ...
  - Try a depth-20 decision tree, compute validation error
  - Return the depth with the lowest validation error
- After you choose the hyperparameter, we usually re-train on the full training set with the chosen hyperparameter

Revisit Decision Trees

- What makes a decision tree overfit?
  - A large tree depth – mainly caused by stopping criteria

- Mitigating the overfitting problem with a validation set!
  - **Post-pruning** – prune some subtrees after the tree is built
    - E.g., Reduced Error Pruning (REP) - remove some branches that do not decrease the accuracy on the validation set

  - **Pre-pruning** – stop early when you build the tree
    - Stop when the tree reach the maximum tree depth
    - Stop when a node has less than a number of examples
    - Stop when the information gain is below a threshold

Hyperparameters

Pre-pruning → hyperparameter tuning

Post-pruning: Reduced Error Pruning

- **Idea**: Check the rules in a bottom-up manner. If a rule does not increase the accuracy (on the validation dataset), the rule can be pruned.

Input: decision Tree T; validation data D

Output: Pruned tree T'

for every internal node N of T, starting from the bottom **do**

$T_N \leftarrow$ subtree of T rooted at N;

$D_N \leftarrow \{x \in D | x \text{ is covered by } N\}$;

if accuracy of T_N over D_N is worse than majority class in D_N **then**

replace T_N in T by a leaf labelled with the majority class in D_N;

end

end

return pruned version T

Optimization Bias

- Another name for overfitting is optimization bias
  - How biased is an error that we optimized over many possibilities?
- Optimization bias of parameter learning
  - During learning, we could search over tons of different decision trees
  - So we can get lucky and find one with low training error by chance
    - Overfitting of the training error
- Optimization bias of hyperparameter tuning
  - Here, we might optimize the validation error over 20 values of depth
  - One of the 20 trees might have low validation error by chance
    - Overfitting of the validation error

Example of Optimization Bias

- Consider a multiple-choice (a, b, c, d) test with 10 questions
  - If you choose answers randomly, expected grade is 25% (no bias)
  - If you fill out two tests randomly and pick the best, expected grade is 33%
    - Optimization bias of ~8%
  - If you take the best among 10 random tests, expected grade is ~47%
  - If you take the best among 100, expected grade is ~62%
  - If you take the best among 1000, expected grade is ~73%
  - If you take the best among 10000, expected grade is ~82%
    - You have so many chances that you expect to do well
- But on new questions the random choice accuracy is still 25%

Factors Affecting Optimization Bias

- If we instead used a 100-question test then
  - Expected grade from best over 1 randomly-filled test is 25%
  - Expected grade from best over 2 randomly-filled test is ~27%
  - Expected grade from best over 10 randomly-filled test is ~32%
  - Expected grade from best over 100 randomly-filled test is ~36%
  - Expected grade from best over 1000 randomly-filled test is ~40%
  - Expected grade from best over 10000 randomly-filled test is ~47%
- The optimization bias grows with the number of things we try
  - Complexity of the set of models we search over
- But, optimization bias shrinks quickly with the number of examples
  - But it's still non-zero and growing if you over-use your validation set

Overfitting to the Validation Set?

- Validation error usually has lower optimization bias than training error
  - Might optimize over 20 values of depth, instead of millions+ of possible trees
- But we can still overfit to the validation error (common in practice)
  - Validation error is only an unbiased approximation if you use it once
  - Once you start optimizing it, you start to overfit to the validation set
- This is most important when the validation set is small
  - The optimization bias decreases as the number of validation examples increases
- Remember, our goal is still to do well on the test set (new data), not the validation set (where we already know the labels)

Examples: Should you trust them?

- Scenario 1
  - I built a model based on the data you gave me
  - It classified your data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Should you trust them?
- Probably not:
  - They are reporting training error
  - This might have nothing to do with test error
  - E.g., they could have fit a very deep decision tree.
- Why probably?
  - If they only tried a few very simple models, the 98% might be reliable
  - E.g., they only considered decision stumps with simple 1-variable rules

Examples: Should you trust them?

- Scenario 2
  - I built a model based on half of the data you gave me
  - It classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error once
  - This is an unbiased approximation of the test error
  - Trust them if you believe they did not violate the golden rule

Examples: Should you trust them?

- Scenario 3
  - I built 10 models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error a small number of times
  - Maximizing over these errors is a biased approximation of test error
  - But they only maximized it over 10 models, so bias is probably small
  - They probably know about the golden rule.

Examples: Should you trust them?

- Scenario 4
  - I built 1 billion models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably not
  - They computed the validation error a huge number of times
  - They tried so many models, one of them is likely to work by chance
- Why probably?
  - If the 1 billion models were all extremely-simple, 98% might be reliable.

Examples: Should you trust them?

- Scenario 5
  - I built 1 billion models based on the first third of the data you gave me
  - One of them classified the second third of the data with 98% accuracy
  - It also classified the last third of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the first validation error a huge number of times
  - But they had a second validation set that they only looked at once
  - The second validation set gives unbiased test error approximation
  - This is ideal, as long as they didn't violate golden rule on the last third
  - And assuming you are using IID data in the first place.

Validation Error and Optimization Bias

- Optimization bias is small if you only compare a few models
  - Best decision tree on the training set among depths 1, 2, 3,..., 10
  - Risk of overfitting to validation set is low if we try 10 things
- Optimization bias is large if you compare a lot of models
  - All possible decision trees of depth 10 or less
  - Here we are using the validation set to pick between a billion+ models
    - Risk of overfitting to validation set is high: could have low validation error by chance
  - If you did this, you might want a second validation set to detect overfitting
- And optimization bias shrinks as you grow size of validation set.