Lecture 18: Loose Ends and False Discovery Rate

David Carlson

FEEDBACK ON MIDTERM + PROJECTS

Midterm

- The uncurved average was 75
  - This is lower than desired; every student will get an additional 10 points, to a max of a 100, making the class average roughly 85.
  - The midterm is only 20% of your grade; focus on making sure you do well on the remaining homeworks and the project
- People largely struggled with penalized regression
  - Will briefly revisit because this is an important topic
- Any regrade (see syllabus) must be requested by 3/26/19

Project Proposals

- Project proposals were on average good
- Every group has feedback on Sakai
  - Some have little feedback, which is a good sign
  - Some groups have *extensive* feedback. This needs to be addressed prior to the progress report
    - Talk to me or a TA if you need help figuring out how to address it
- Grading gets harsher if points are not addressed…

Scientific Questions

- A repeated issue in the project proposal was either:
  - A lack of a clear goal or scientific question
  - How the proposed methodology would answer that question
- Also lack of related work in some projects

Example 1:

- Scientific goal: We hypothesize that body temperature can predict later hospital admission.
- Methodology: We will address this hypothesize by predicting hospital admission from body temperature using a decision tree.

Example 2:

- Specific goal: We hypothesize than a random forest method can predict surgical complications better than existing logistic regression models.
- Methodology: we will evaluate AUC on a clinical dataset using a random forest and logistic regression to evaluate whether there is an improvement in predictive performance

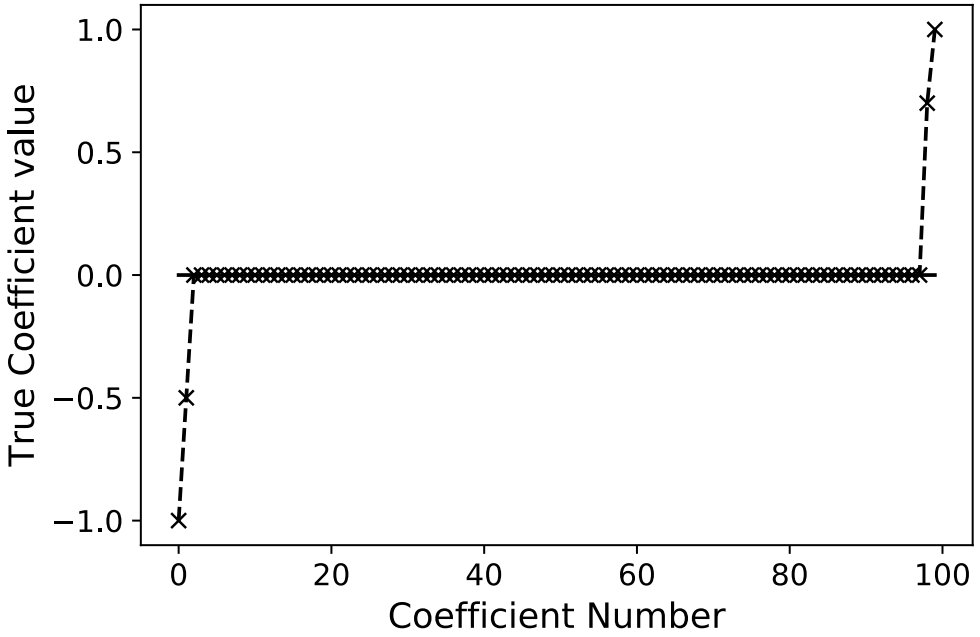
REVISITING PENALIZED REGRESSION

True linear regression weights

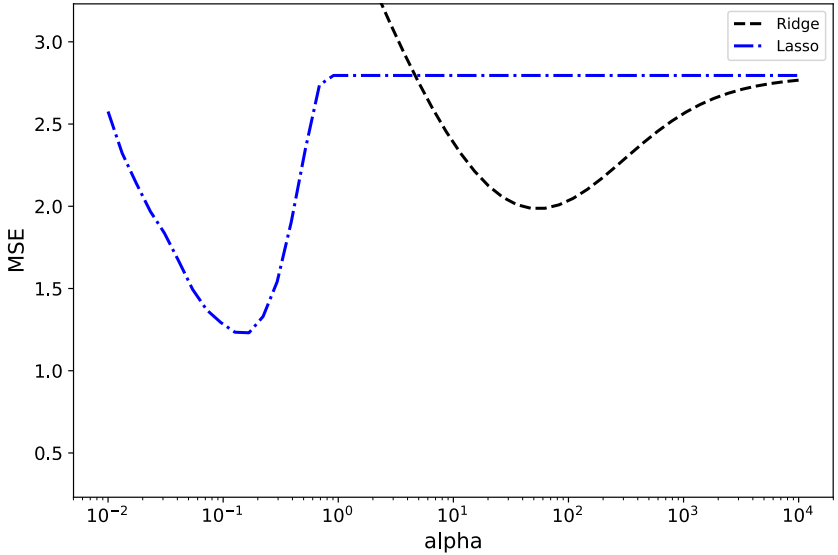
If the linear regression model is true with the following procedure, what would be the better method?

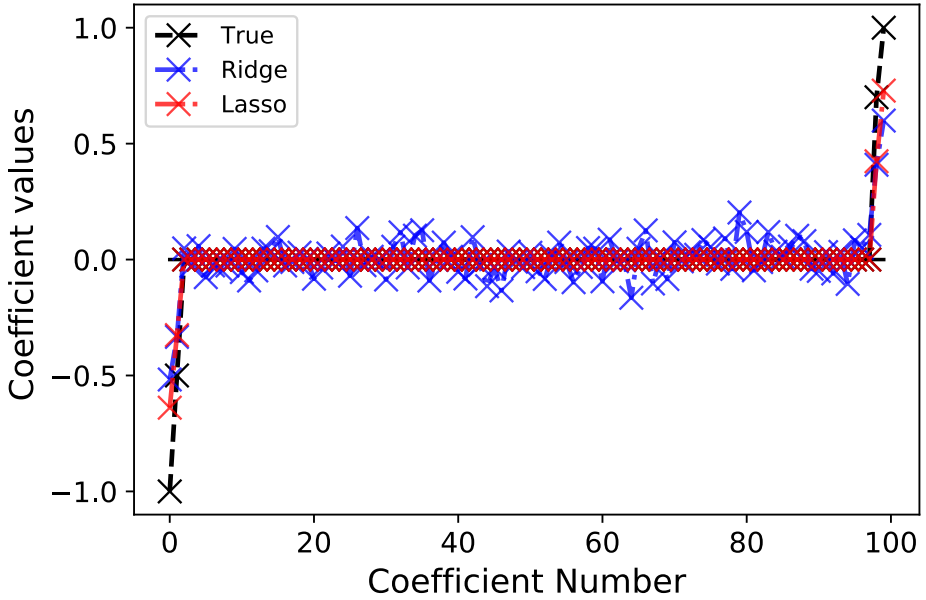
Ridge Regression?

Lasso Regression?



Mean Squared Error and Inferred Weights



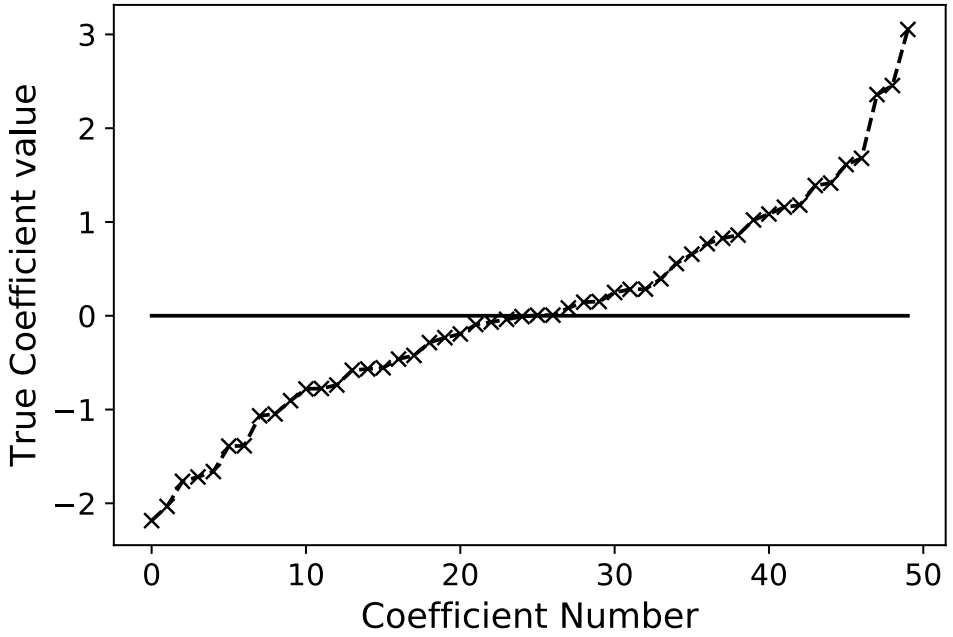


True linear regression weights

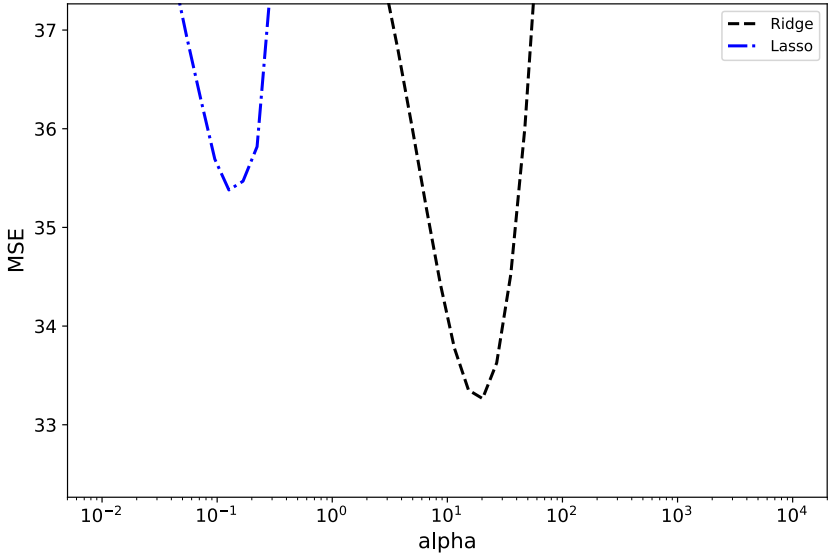
If the linear regression model is true with the following procedure, what would be the better method?

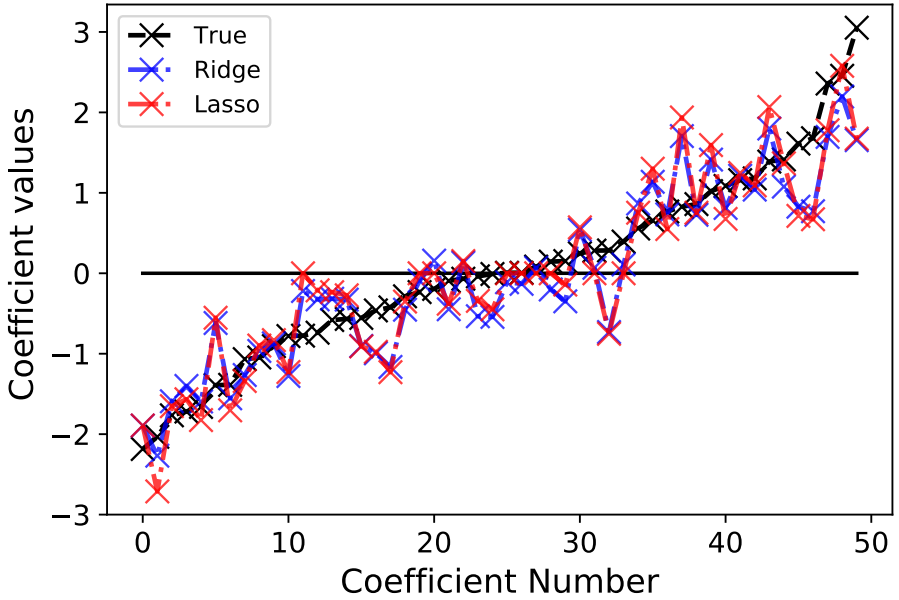
Ridge Regression?

Lasso Regression?



Mean Squared Error and Inferred Weights



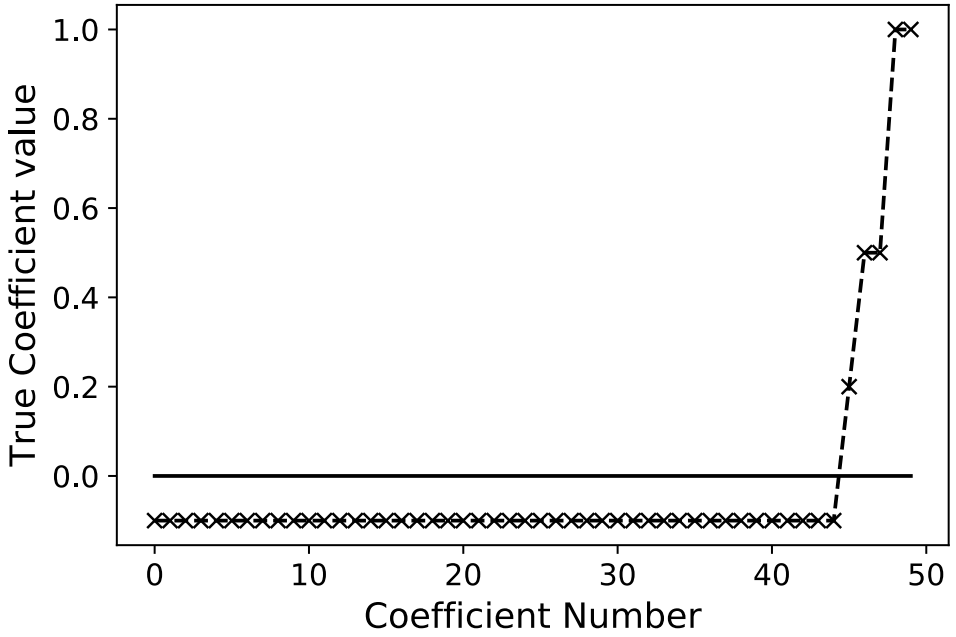


True linear regression weights

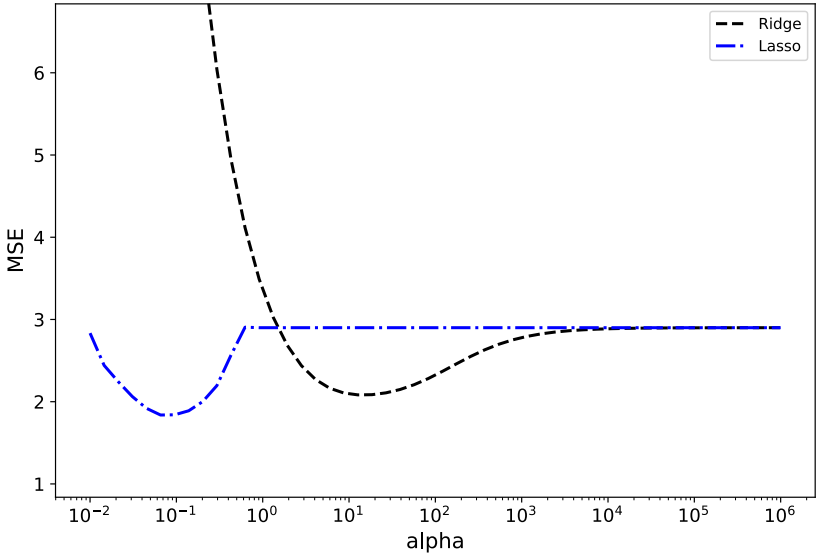
If the linear regression model is true with the following procedure, what would be the better method?

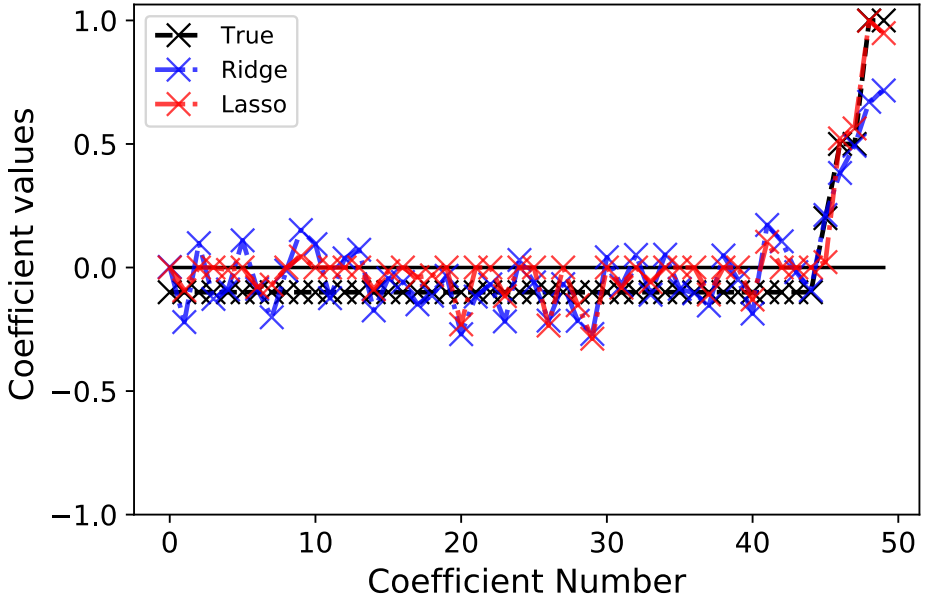
Ridge Regression?

Lasso Regression?



Mean Squared Error and Inferred Weights





Unclear which one is really better here…

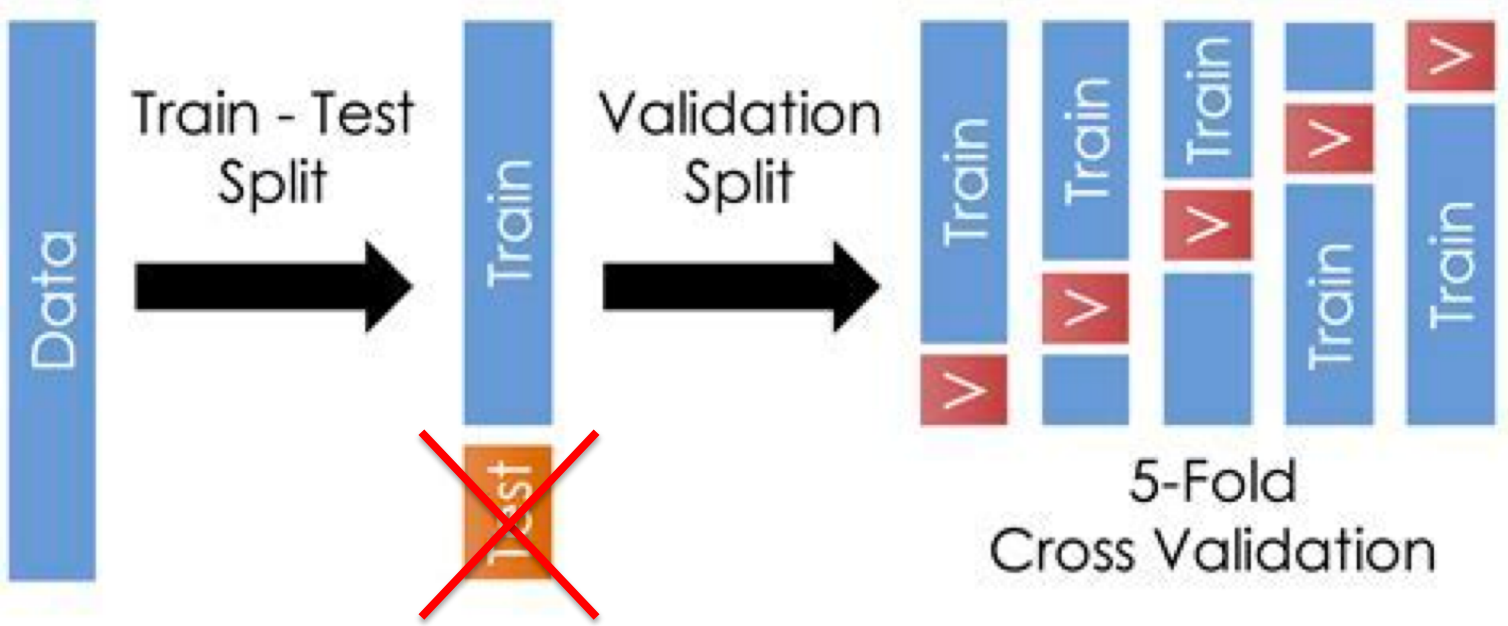
RETURNING TO STATISTICAL TESTING

What to do if our test set is really small?

Small samples have lots of uncertainty

- If our test set is 20 data points, this can make our performance estimates very uncertain
- In these situations, people often just report cross-validation results
- This can be problematic and overconfident

Let's return to cross-validation



Permutation Testing

- One way of evaluating whether the machine learning algorithm is performing above chance in the cross-validation procedure is *permutation testing*
- Essentially, this asks the question: how high would the estimate of cross-validation be on *random* values

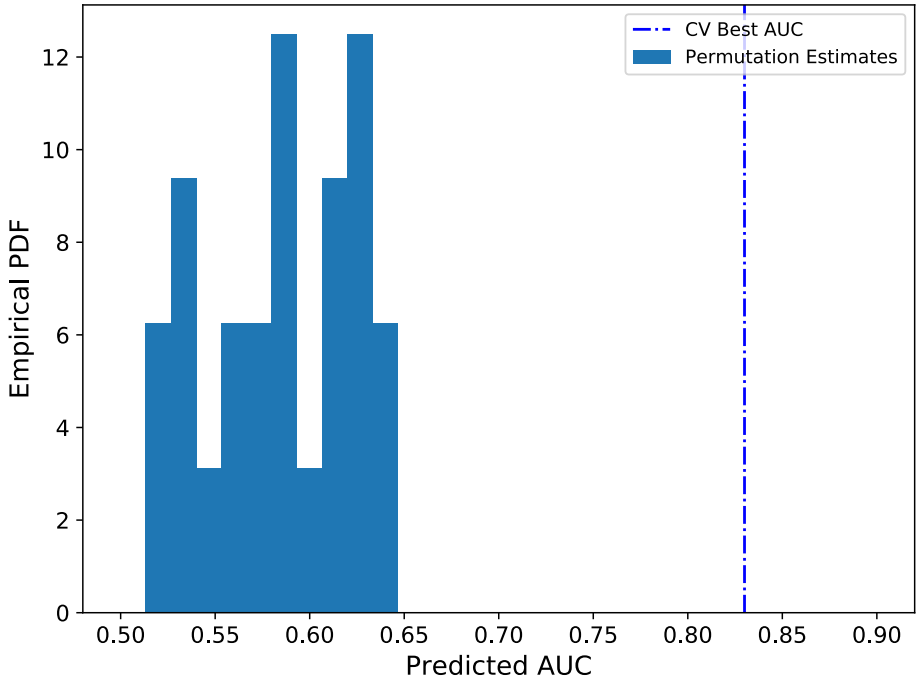
Permutation Testing Approach

- How high would the estimate of cross-validation be on *random* values? To assess this, try many times:
  - Randomly permute the labels of the data
  - Run the cross-validation grid search procedure
  - Record the best estimated performance
- This creates an empirical distribution; then the data can be used to create a p-value

Visualization of Permutation Testing

This is a visualization of the permutation test in action.

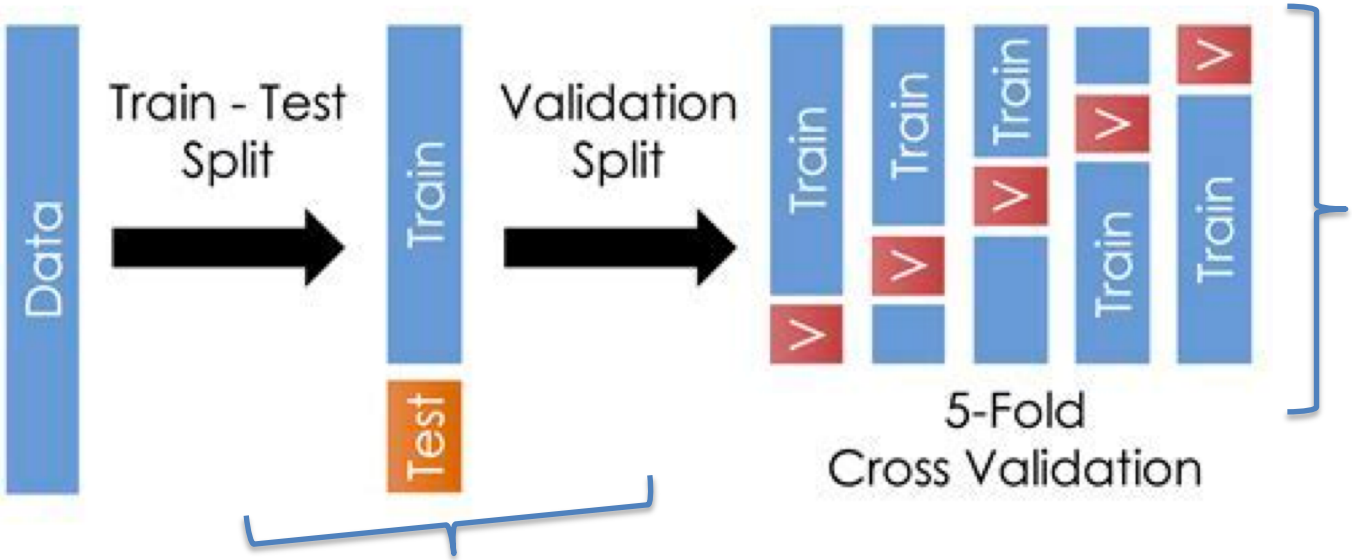
This is the breast cancer dataset limited to 100 data points fitting a decision tree. Note that *all* premutation AUCs are above .5 (greater than chance)!



“Nested” Cross-Validation

- Another approach tries to solve the problem of a small test set by using the entire dataset as the test set
- This seems problematic, but can be done *if care is taken to set this up correctly!*

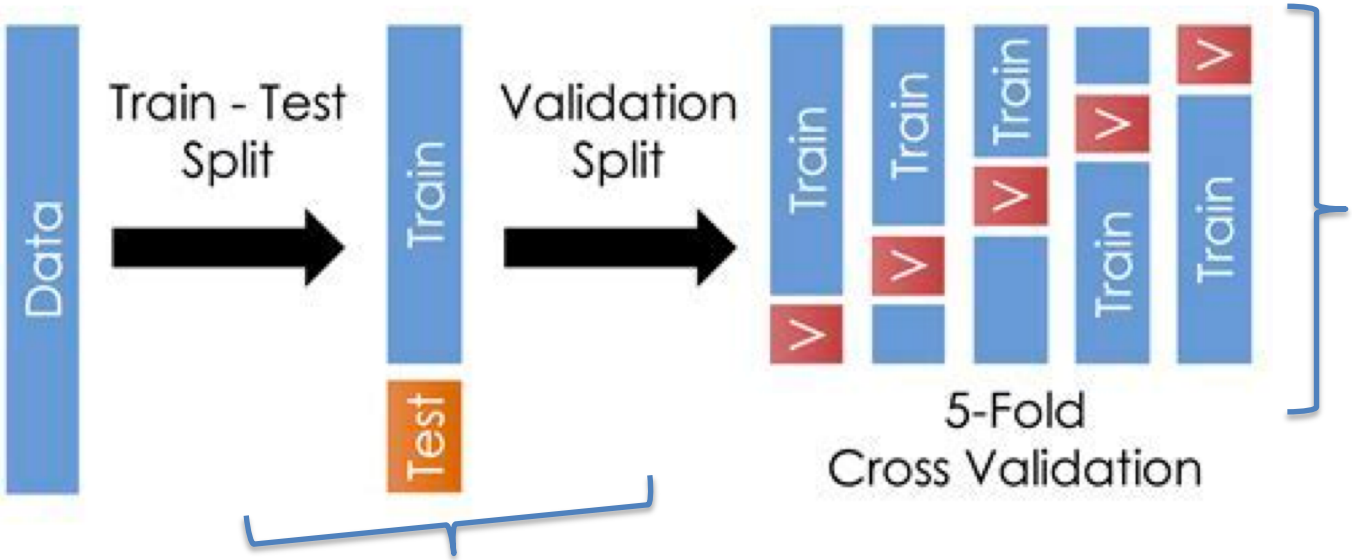
Let's return to cross-validation



We typically rotate data on our validation dataset, allow us to reduce variance on our validation data.

We only have a predefined, single, small test set.

Let's return to cross-validation



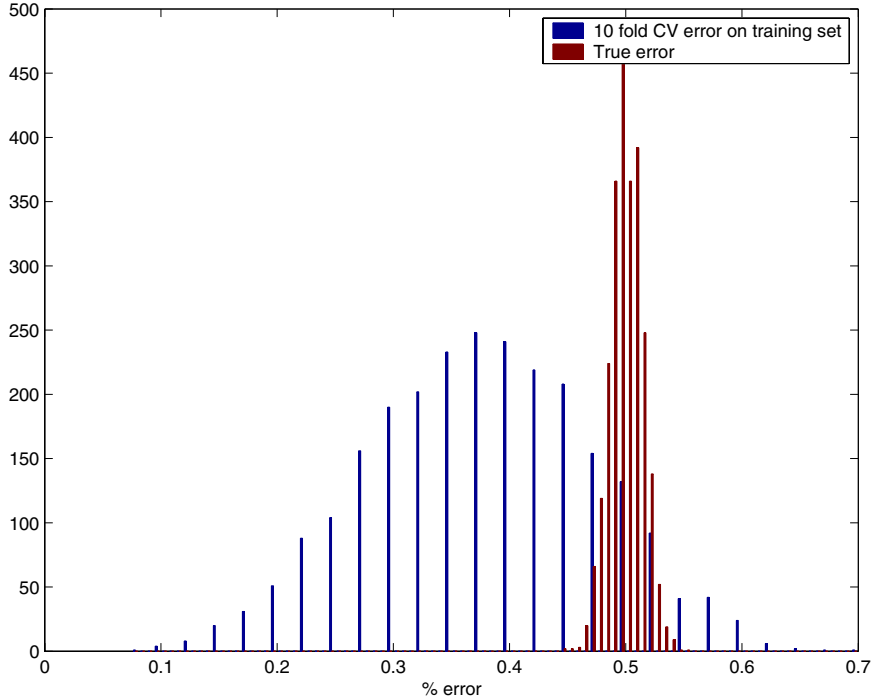
We typically rotate data on our validation dataset, allow us to reduce variance on our validation data.

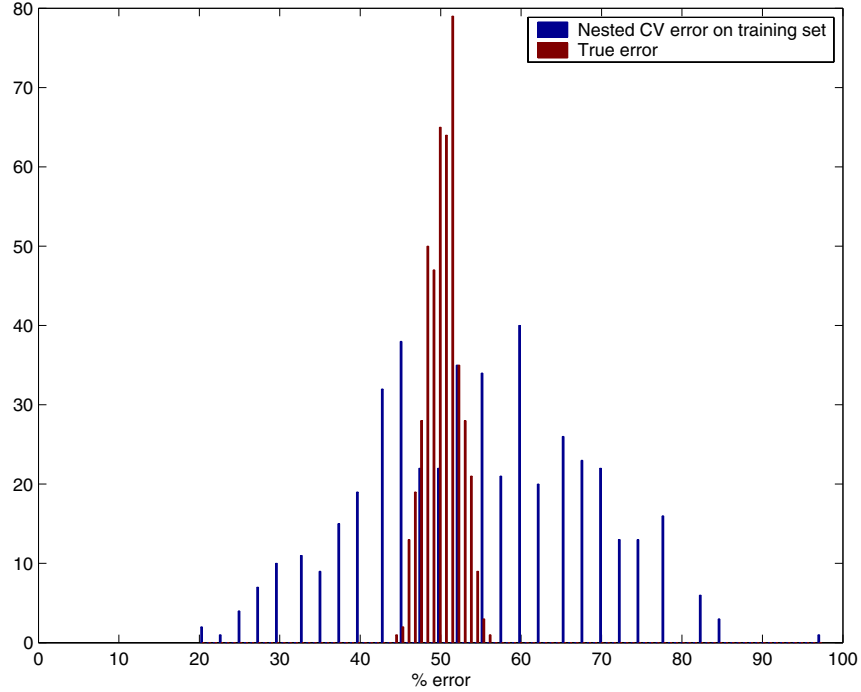
The test set is rotated also.

Is this a problem?

- This seems like a problem because now our test set is being used as a training dataset part of the time
- However, each test set is only used once and that test set doesn't influence itself
- This procedure is theoretical sounds ***IF:***
  - The entire model selection procedure (i.e. which model, grid search, etc.) is set and standardized beforehand
- Can be very useful in practice
- Really the big downside here is coding and *computational* complexity

Reduces Bias in Practice





Varma and Simon 2006

Intermediate Conclusions

- Permutation Testing and Nested Cross-Validation are extremely helpful when data sizes are very small
  - Permutation testing allows for the development of a empirical p-value (but does not remove bias!)
  - Nested CV helps reduce bias

FALSE DISCOVERY RATE

How do we address multiple comparisons?

- Recall what a p-value means:
  - Chance that a statistic that extreme occurred under the null hypothesis
- But we run into issues when trying repeatedly trying statistical tests
  - Need to correct for multiple comparisons
- https://xkcd.com/882/

Bonferroni Correction

- Suppose that we have tried *M* different algorithms on the test set, and we have found one that is better than chance.
- Want to control Type 1 error that we make a false claim ("Familywise Error Rate"
- Simple version is that if we have *M* different tests, we should correct the p-value by multiplying it by *M*.
  - This is the "Bonferroni Correction"
  - This procedure guarantees that the null hypothesis is rejected by chance at less than or equal to the desired significance level.
- *Better* corrections exist with additional knowledge or assumptions, hard to use in general

Bonferroni is conservative

- Bonferroni controls the rate that we make ***any*** false claim
- Suppose we are looking at gene expression:
  - Maybe 20,000 genes that we can look at
  - If you test each gene, Bonferroni means that you multiple the each p-value by *20000!*

Bonferroni is conservative

- Bonferroni controls the rate that we make ***any*** false claim
- Suppose we are looking at gene expression:
  - Maybe 20,000 genes that we can look at
  - If you test each gene, Bonferroni means that you multiple the each p-value by *20000!*

Two Different Statistical Approaches

- Familywise Error Rate
  - Bonferroni Correction
  - Controls the chance that *any* reported value is false
- False Discovery Rate
  - Controls the chance that each reported value is false

A Practical Example

- Let's consider a hypothetical significance level of .05 testing 400 gene expressions
- Under Bonferroni:
  - Report 2 genes as significant
  - Chance that both are real findings is >.95
- Under False Discovery Rate:
  - Report 100 genes as significant
  - Expect that 5% are false (5 findings are incorrect)
- Which is better depends on the situation, but often the second is reasonable in practice

Benjamini-Hochberg

- The Benjamini-Hochberg method for multiple comparisons is a standard procedure
  - Choose a significance level α
  - Suppose we have *M* tests
  - Order the p-values from smallest to largest (p_1 is smallest)
  - Determine the largest value k such that $p_k < \frac{k}{M}\alpha$
  - Mark the tests with the smallest k p-values as significant
- We will visualize and go through this in code to make this procedure clearer

Conclusions

- Statistical testing is critical in science
- Often less critical in industrial data science applications where conclusions are easier to test
- Very important to understand the hypothetical uncertainty
- We will go through code on a number of these procedures.
  - They do get technical, but are necessary
- If time remains, we will go through algorithm/method comparison tests

COMPARING ALGORITHMS

How do we compare algorithms?

- Often, instead of comparing whether we can predict at all, we want to know whether there is evidence that algorithm B is in fact different in performance than algorithm A.
  - Is our complex algorithm better than our simple one?
- Instead of comparing the decision statistic correctly, needs to compare the *loss* (or some other quantity where only one side is good).
  - i.e. in logistic regression, if can write down our logistic loss
    - Small loss -> good
    - Large, negative loss -> bad
- A simple question: is the loss in algorithm A different from the loss in algorithm B?

Comparing algorithms with Mann–Whitney *U*

- We can frame this as a similar question as when we wanted to know if our algorithm was significantly different from chance.
- Now we're trying to estimate

$$E_{\{x_A,y_A\}\sim P,\{x_B,y_B\}\sim P}[\ell(f_A(x_A),y_A) > \ell(f_B(x_B),y_B)]$$

- Can run the same Mann-Whitney U test to see if these distributions are actually different.
- Some notes:
  - The loss must be the same
  - Doesn't test for mean – that's a different statistical test that is sometimes more relevant, but often not as robust
- Often we actually have *matched samples*, such that each example given to algorithm A is the exact same as the example given to algorithm B. Does this matter?

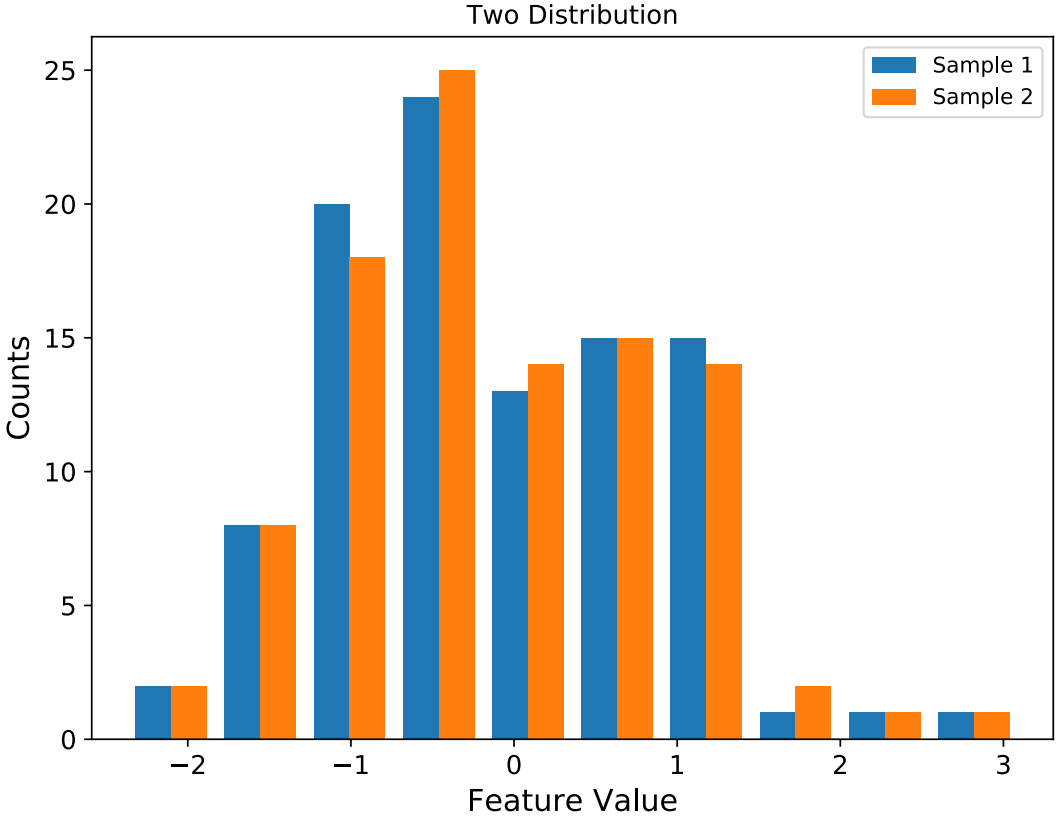
A simple example:

- Suppose that we have data samples, $\{x_i\}_{i=1,\ldots,N}$, and the performance for each sample is given by $\{z_i^A\}_{i=1,\ldots,N}$ for algorithm A and $\{z_i^B\}_{i=1,\ldots,N}$ for algorithm B.
- If $z_i^A \sim N(0,1)$ and $z_i^B = z_i^A$+.01, how many samples would it take until we realized the means were different?

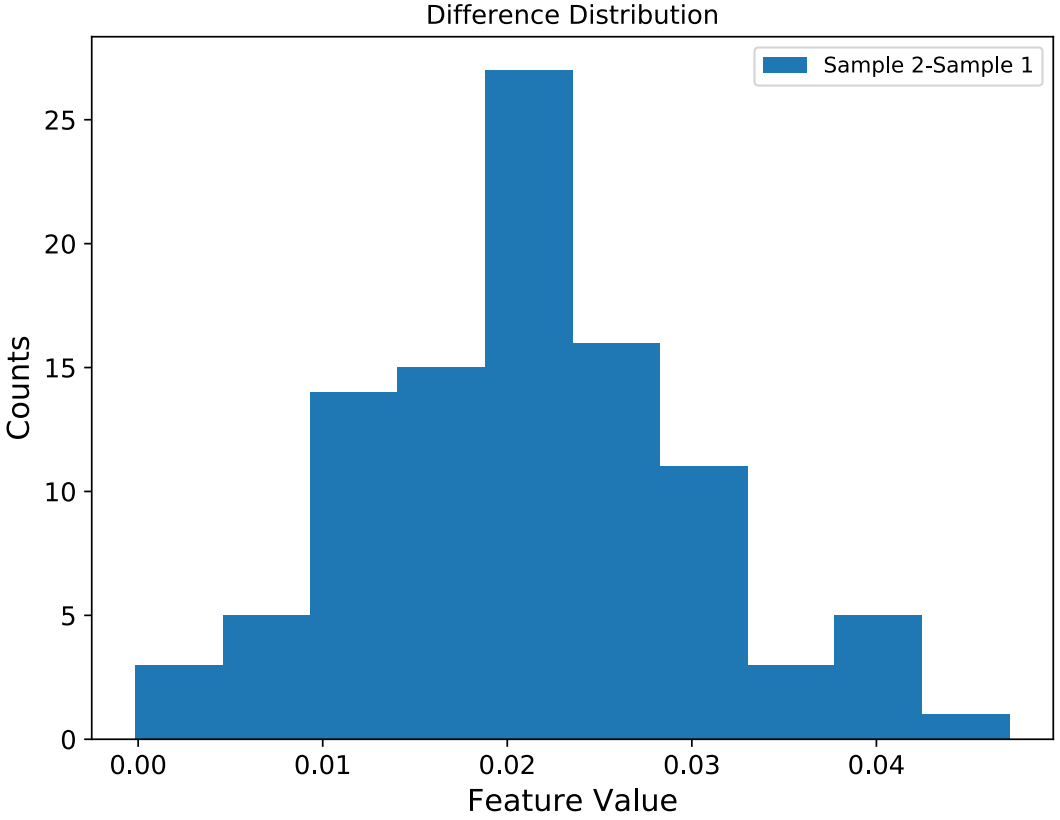
- What if instead of comparing the means from algorithm A and algorithm B, we compare their results on the exact same samples?

$$z_i^B - z_i^A = .01$$

Two Distributions



Pairwise Distances



Wilcoxon Signed-Rank Test

- Instead of comparing *average* performance of an algorithm, we can use the fact that the algorithms are evaluated *on the same data*.
- The most common way of doing this is a *Wilcoxon Signed-Rank* test.

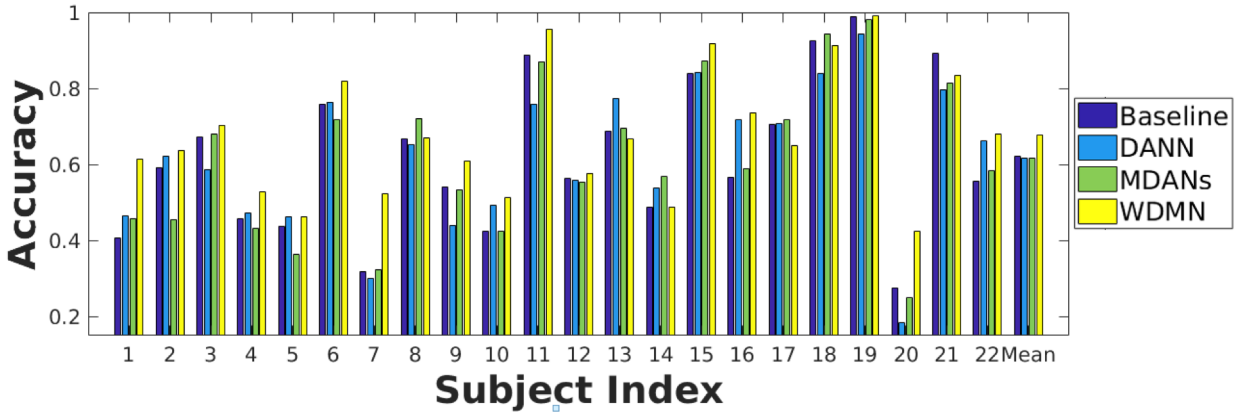
- Some important details:
  - Very commonly appropriate in data science where we evaluate on the same data
  - Less common in experiments – rare that we get to try two strategies on the same quantities (i.e. can't try each patient with two different types of surgery)
  - Same question as the rank-sum test
  - Can have *drastically* better statistical power

A Real Example

A comparison tracking neurological changes in children diagnosed with Autism Spectrum Disorder.

P-value for testing whether the means are different: 0.3

P-value on with paired statistics: 0.002



Conclusions

- Statistical testing is very necessary to understand in the context of machine learning
- Estimating performance accurately is important
- Different from many tests run in statistics; primarily caring about predictive ability