# Experimentation

Jaime Arguello
INLS 613: Text Data Mining
jarguell@email.unc.edu

# Outline

Parameter Tuning

Cross-Validation

Significance tests

# Evaluation

- The goal of evaluation is to determine a model's performance on previously unseen data
  - Parameter-tuning
  - Comparing between alternative approaches
  - Feature-ablation studies

# Parameter Tuning

## motivation

- Supervised machine learning algorithms have lots of moving parts
- We can think of these parameters as “knobs” that need to be tweaked or tuned
- The goal is to set these parameter values such that we maximize performance
- We need to do this for both systems, not just the one we want to win!
- Can you think of some example parameters?

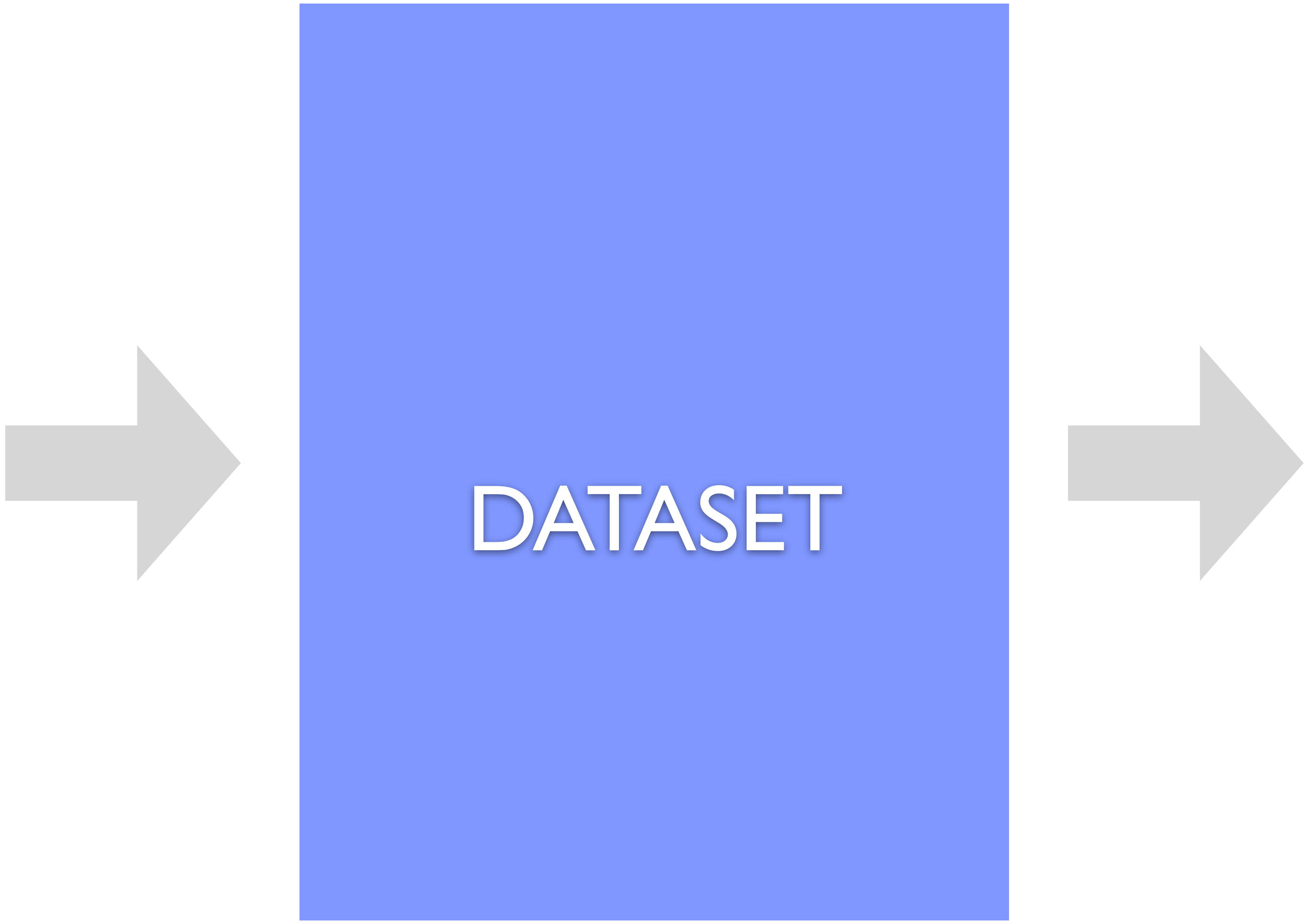
# Parameter Tuning

- K-nearest Neighbor
  - Compute the similarity between a previously unseen instance and all the instances in the training set
  - Assign the majority class associated with its K nearest neighbors
- Parameter K determines the number of training set instances that are used in the voting
- Goals:
  - How do we set K?
  - What is the expected performance of the system with a good value of K?

# Parameter Tuning

- How should we determine the value of K?
- Option -1: roll the dice, close your eyes, and hope for the best
- Option 0: take a conservative guess (e.g., K = 5)?
- Option 1: try out a range of values (e.g., K = 1, 5, 10, 20, 50, 100) and set it to the value that maximizes performance based on a sensible metric?

# Parameter Tuning



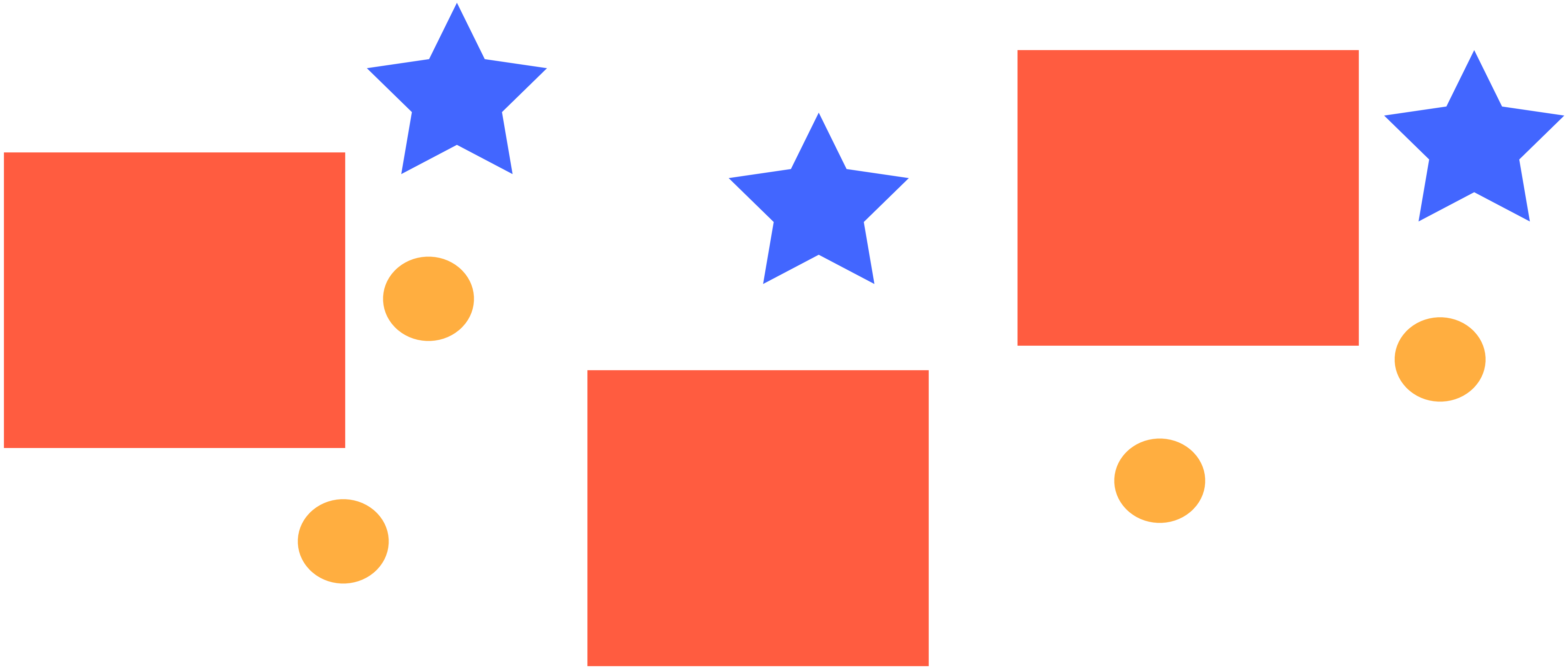
| K = | F-measure = |
|---|---|
| 1 | 0.25 |
| 2 | 0.27 |
| 3 | 0.29 |
| 4 | 0.35 |
| 5 | 0.45 |
| 6 | 0.50 |
| **7** | **0.55** |
| 8 | 0.47 |
| 9 | 0.35 |
| 10 | 0.20 |

Why is this a bad idea?

# Parameter Tuning

## toy example

- Objective: distinguish between stars, squares, and circles



- Parameters: the relative importance between (1) size, (2) color, and (3) number of sides

# Parameter Tuning

- The goal is to set parameter values such that we maximize performance
- What is the performance that we are really interested in?
- We care about performance on previously unseen data
- We care about generalization performance!
- Our training set may contain regularities that are not meaningful
- We care about those regularities that are meaningful for the overall population!

# Parameter Tuning

K = 5

K = 10

THE WORLD

F-measure = 0.60

F-measure = 0.55

# Parameter Tuning

- Option 2:

1. divide the data set into two sets
    - training set: a set used to find the best parameter values (e.g., 80%)
    - test set: a held-out set used to evaluate model performance (e.g., 20%)
2. train: find the parameter value that maximize performance on the training set
3. test: evaluate the model (with the best training-set parameter value) on the test set

# Parameter Tuning

DATASET

# Parameter Tuning

- Split the data into two sets.
- Find the parameter value that maximizes performance on the training set.
- Evaluate the system with that parameter value on the test set.

TRAINING SET (80%)

K = 5

TEST SET (20%)

F = 0.50

# Parameter Tuning

- Split the data into two sets.
- Find the parameter value that maximizes performance on the training set.
- Evaluate the system with that parameter value on the test set.

TRAINING SET (80%)

K = 5

TEST SET (20%)

F = 0.50

Advantages and Disadvantages?

# Single Train/Test Split

- Advantage
  - the data used to find the optimal parameter value is not the same data used to test!
  - we are testing generalization performance.
- Disadvantage
  - we are putting all our eggs in one basket!
  - out of pure coincidence, the training set may have regularities that don't generalize to the test set

# Parameter Tuning

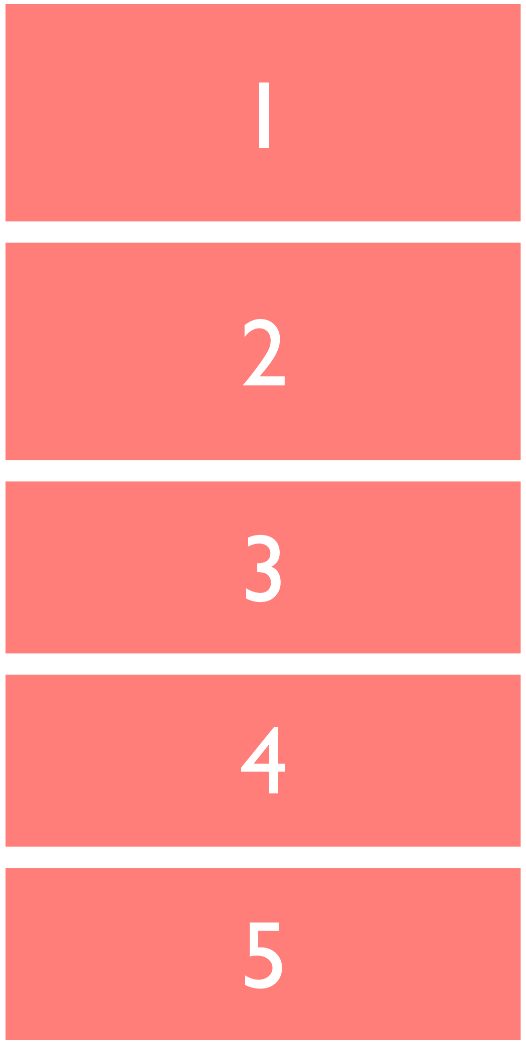
- Option 3: cross-validation
  1. divide the data into N sets of instances
  2. use the union of N-1 sets to find the best parameter values
  3. measure performance (using the best parameters) on the held-out set
  4. do steps 2-3 N times
  5. average performance across the N held-out sets
- This is called N-fold cross-validation (usually, N=10)

# Cross-Validation

DATASET

# Cross-Validation

- Split the data into N = 5 folds



1

2

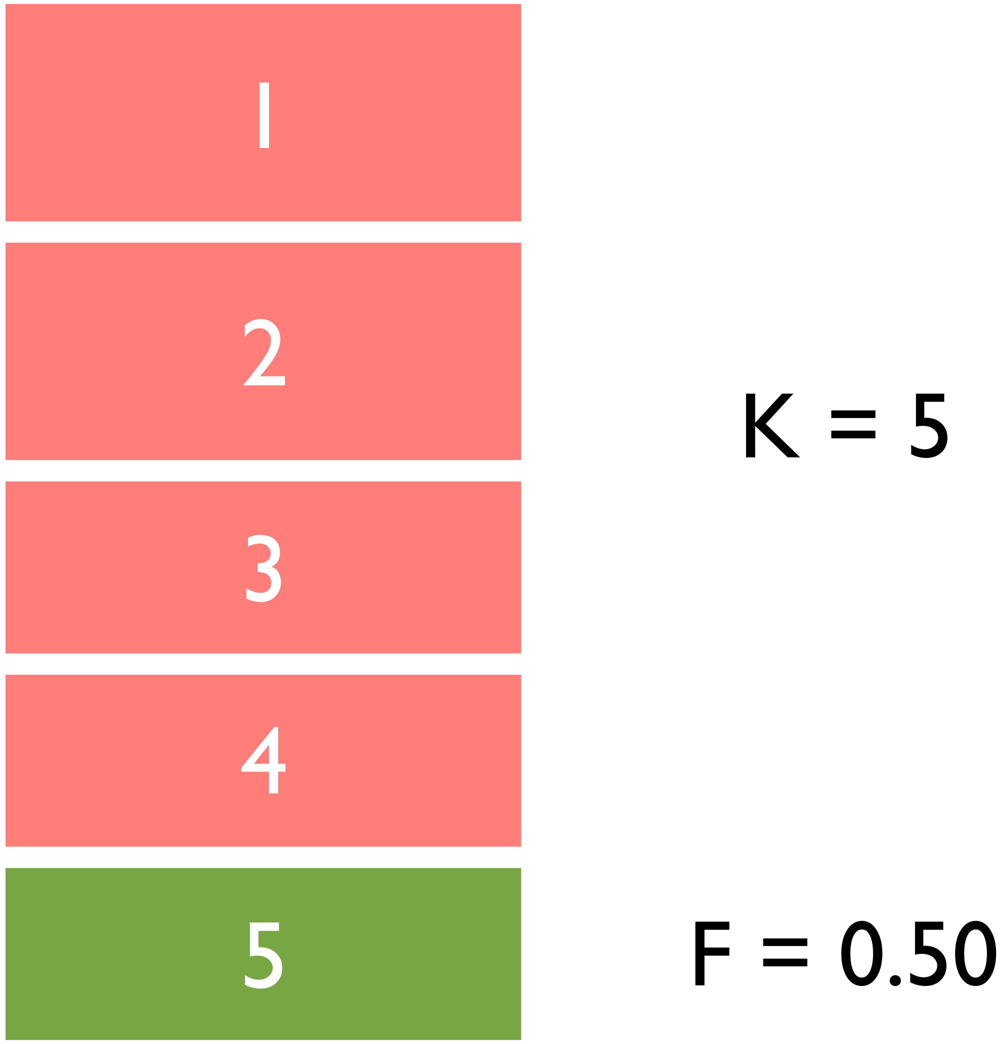
3

4

5

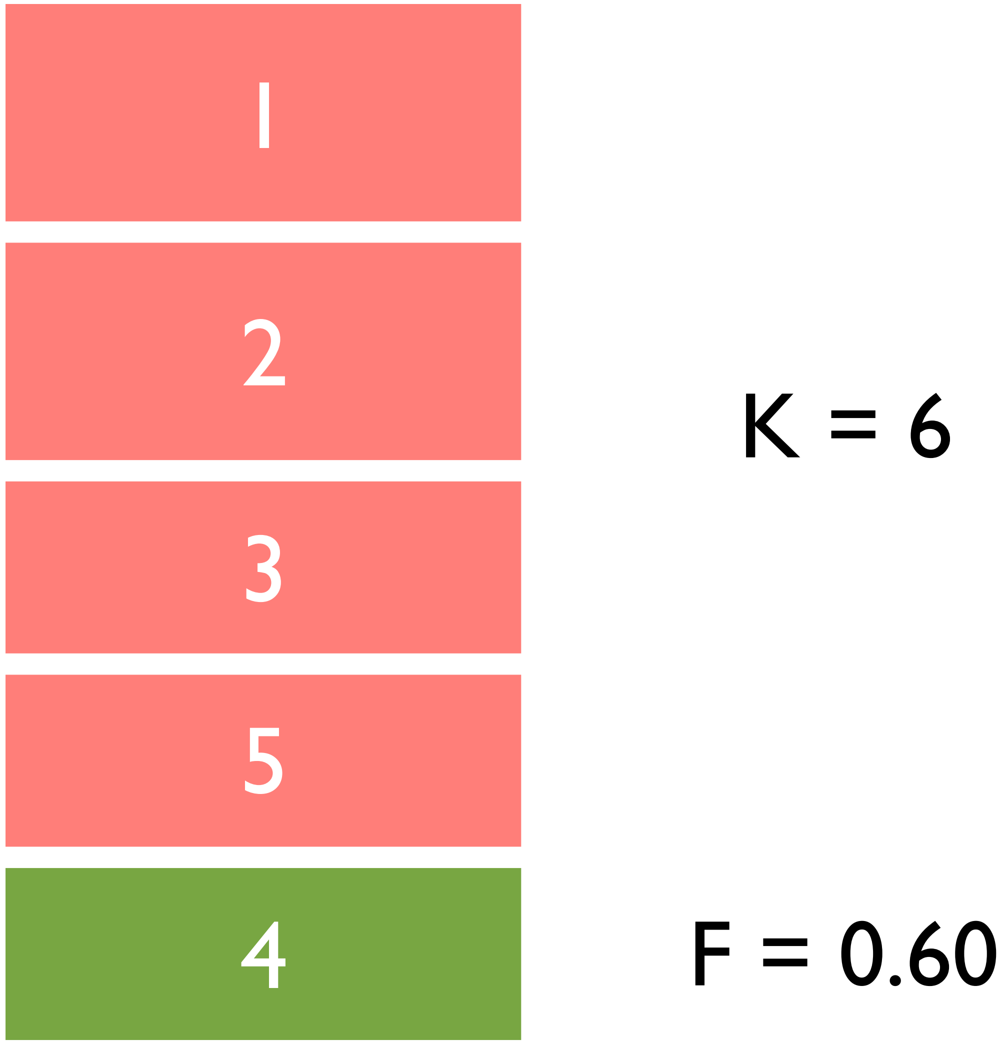
# Cross-Validation

- For each fold, find the parameter value that maximizes performance on the union of N - 1 folds and test (using this parameter value) on the held-out fold.



# Cross-Validation

- For each fold, find the parameter value that maximizes performance on the union of N - 1 folds and test (using this parameter value) on the held-out fold.



1

2

3

5

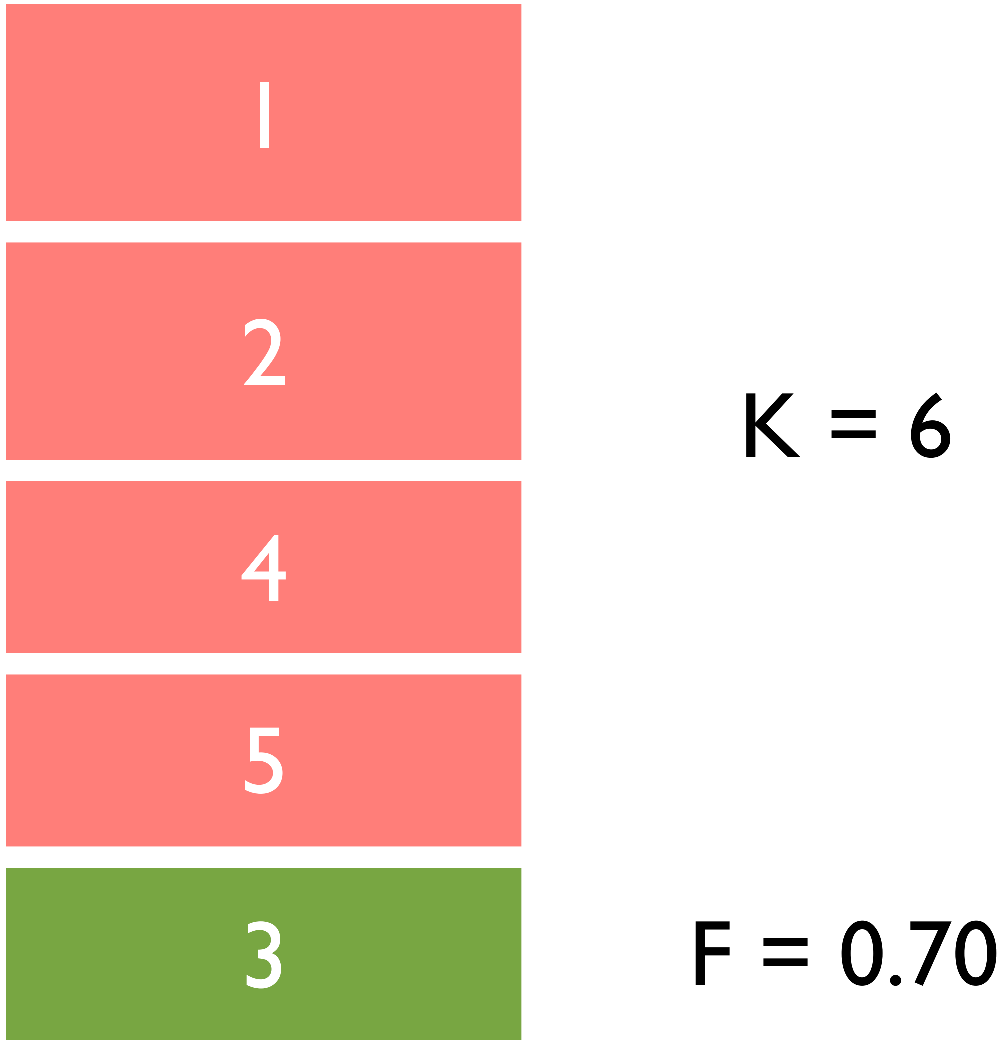
4

K = 6

F = 0.60

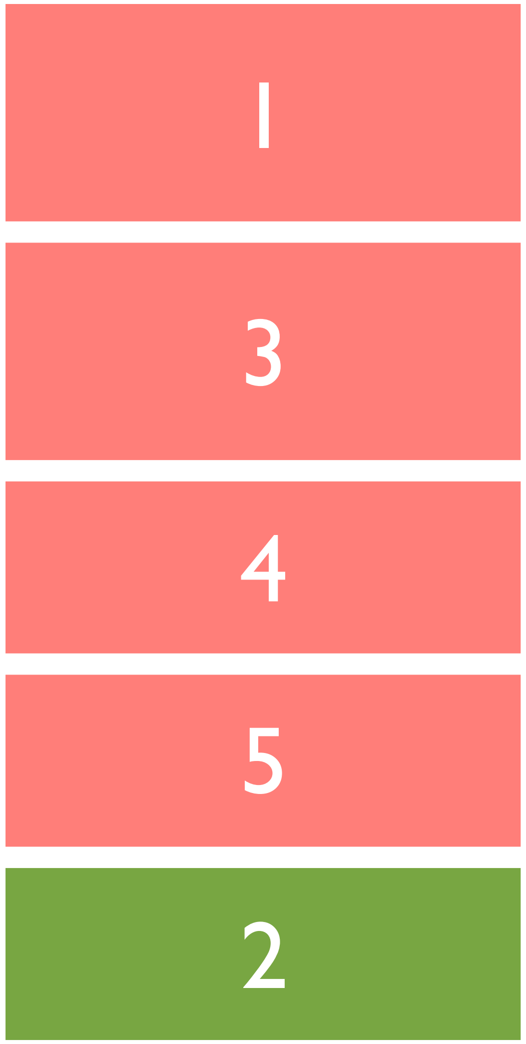
# Cross-Validation

- For each fold, find the parameter value that maximizes performance on the union of N - 1 folds and test (using this parameter value) on the held-out fold.



# Cross-Validation

- For each fold, find the parameter value that maximizes performance on the union of N - 1 folds and test (using this parameter value) on the held-out fold.

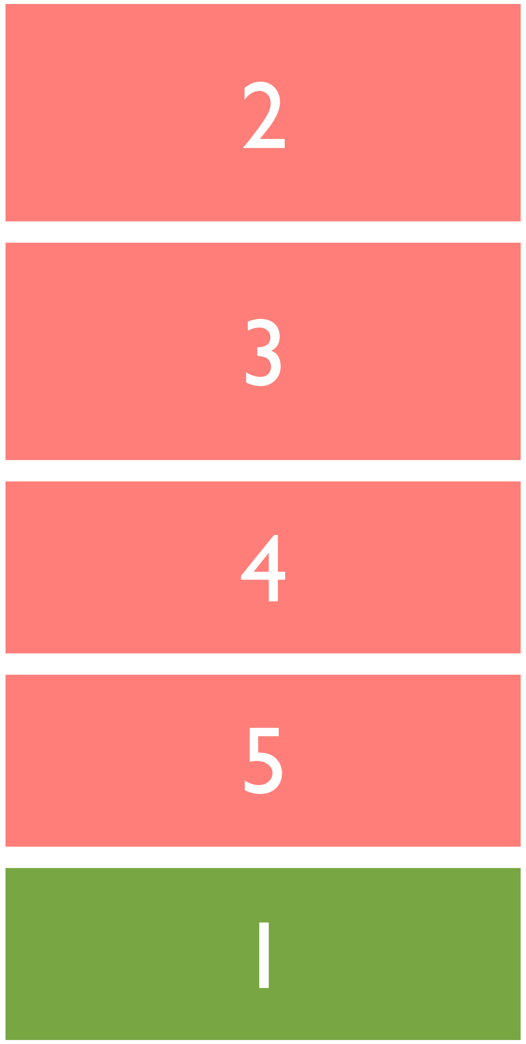


K = 4

F = 0.60

# Cross-Validation

- For each fold, find the parameter value that maximizes performance on the union of N - 1 folds and test (using this parameter value) on the held-out fold.


2

3

4

5

1

K = 5

F = 0.50

# Cross-Validation

- Average the performance across held-out folds

| | |
|---|---|
| 1 | F = 0.50 |
| 2 | F = 0.60 |
| 3 | F = 0.70 |
| 4 | F = 0.60 |
| 5 | F = 0.50 |
| Average | **F = 0.58** |

# Cross-Validation

- Average the performance across held-out folds

| | |
|---|---|
| 1 | F = 0.50 |
| 2 | F = 0.60 |
| 3 | F = 0.70 |
| 4 | F = 0.60 |
| 5 | F = 0.50 |
| Average | **F = 0.58** |

Advantages and Disadvantages?

# N-Fold Cross-Validation

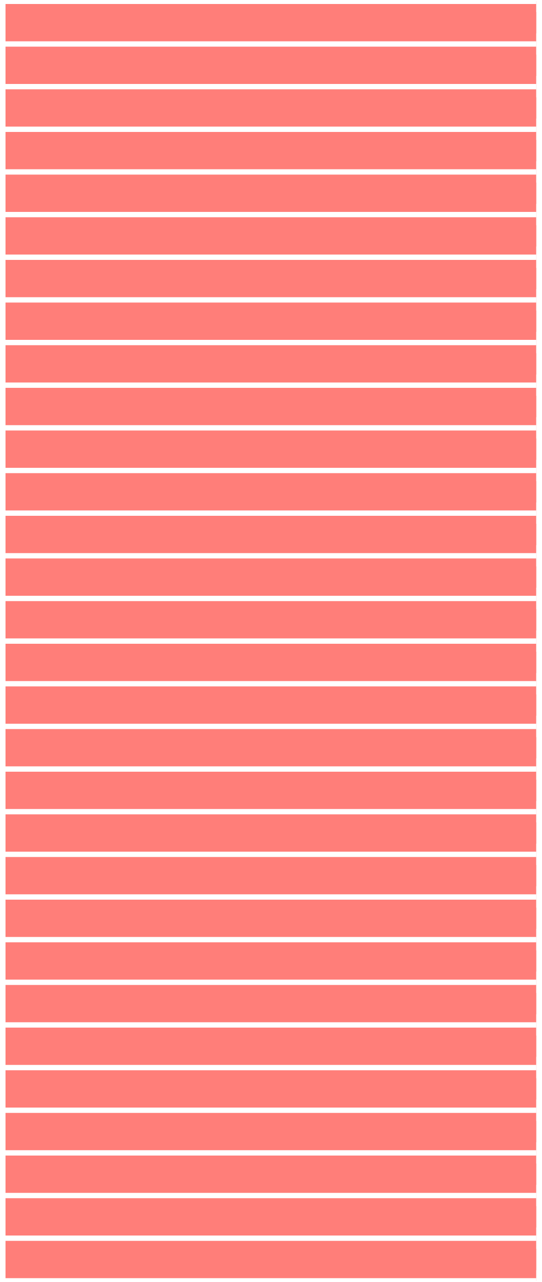
- Advantage
  - multiple rounds of generalization performance.
- Disadvantage
  - ultimately, we'll tune parameters on the whole dataset and send our system into the world.
  - a model trained on 100% of the data should perform better than one trained on 80%.
  - thus, we may be underestimating the model's performance!

# Leave-One-Out Cross-Validation

DATASET

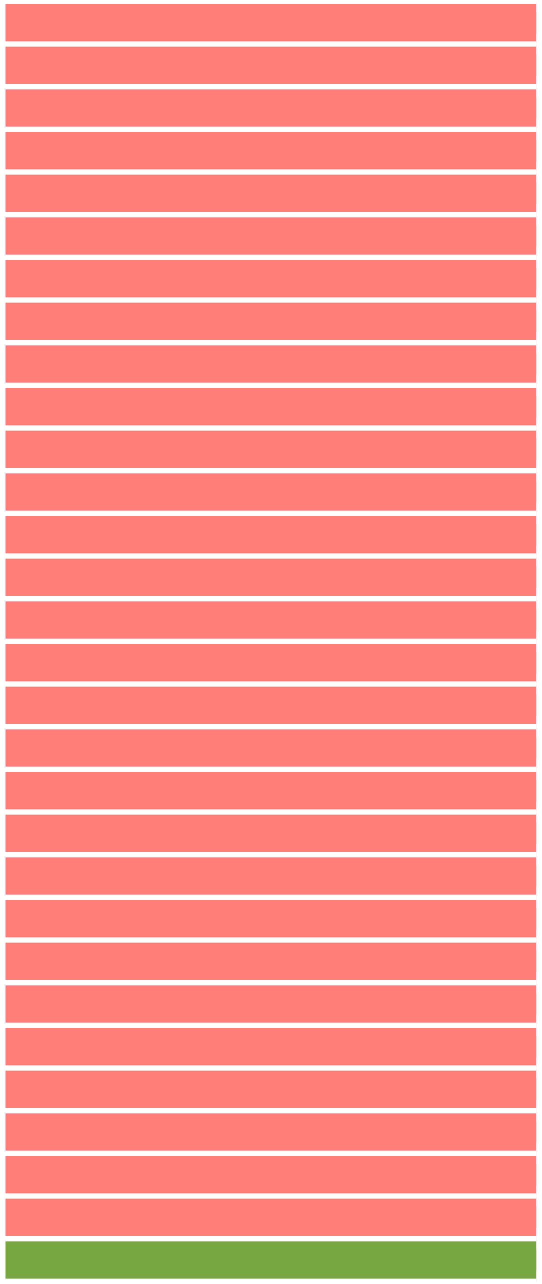
# Leave-One-Out Cross-Validation

- Split the data into N folds of 1 instance each



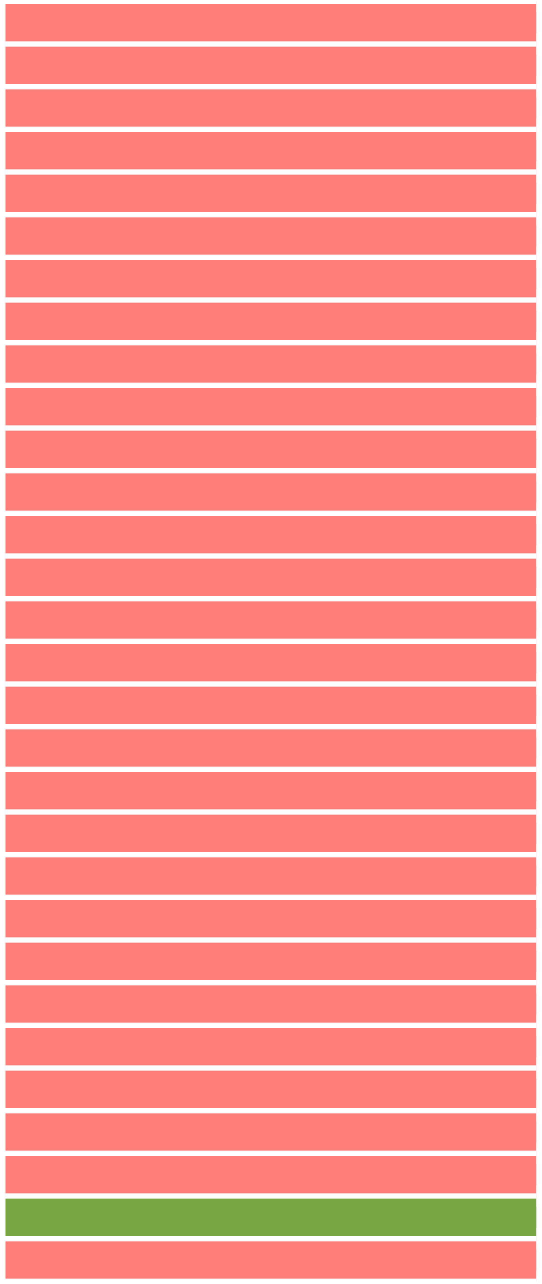
# Leave-One-Out Cross-Validation

- For each instance, find the parameter value that maximize performance on for the other instances and and test (using this parameter value) on the held-out instance.



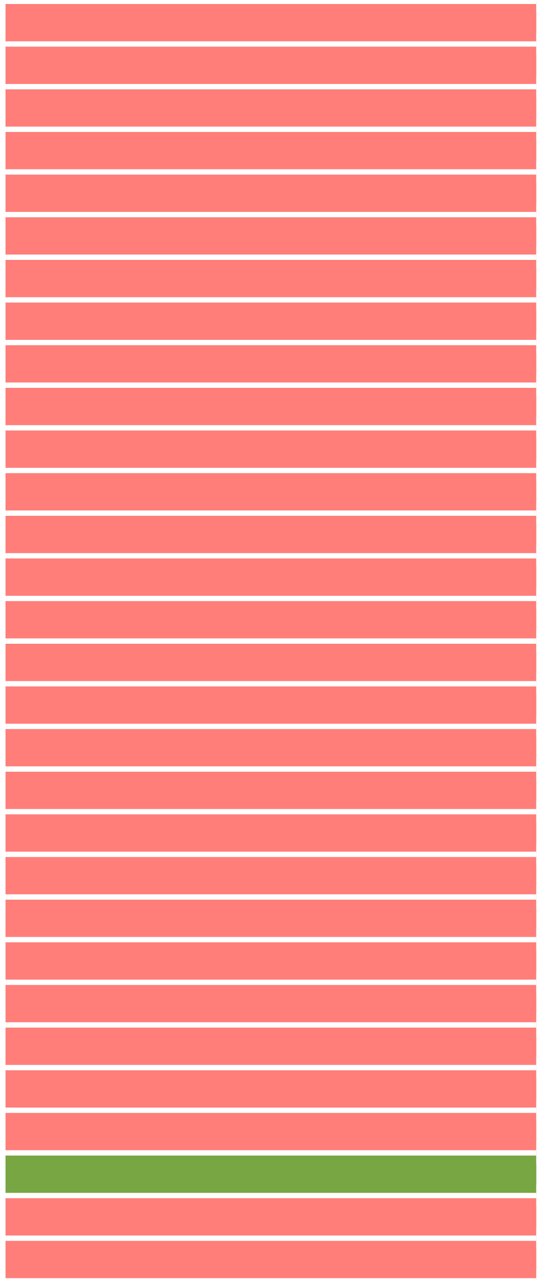
# Leave-One-Out Cross-Validation

- For each instance, find the parameter value that maximize performance on for the other instances and and test (using this parameter value) on the held-out instance.



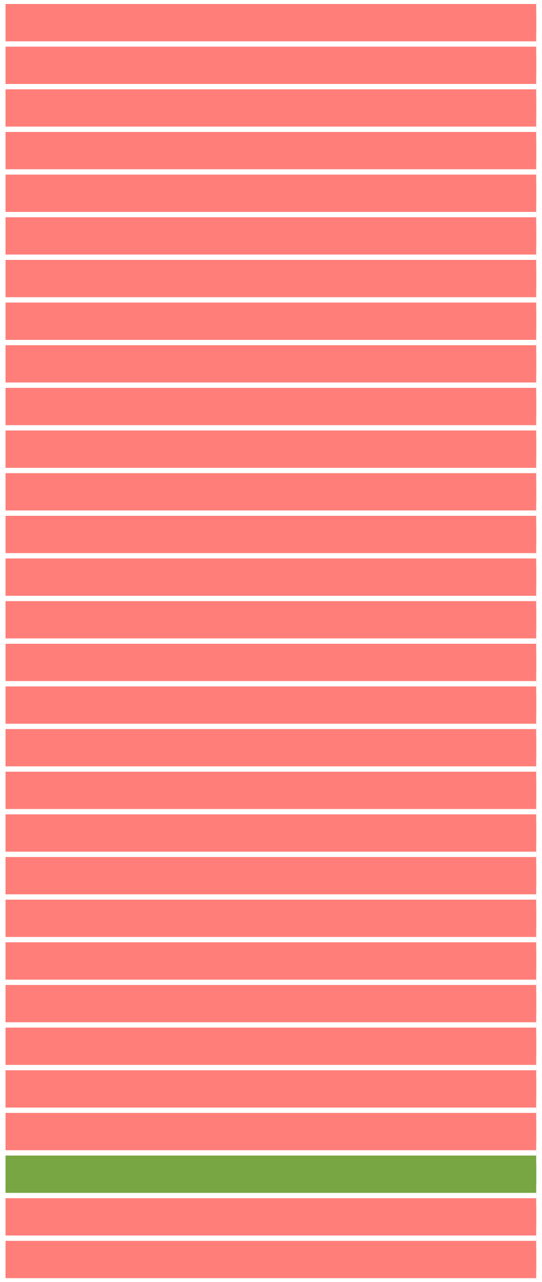
# Leave-One-Out Cross-Validation

- For each instance, find the parameter value that maximize performance on for the other instances and and test (using this parameter value) on the held-out instance.
- And so on ...
- Finally, average the performance for each held-out instance



# Leave-One-Out Cross-Validation

- For each instance, find the parameter value that maximize performance on for the other instances and and test (using this parameter value) on the held-out instance.
- And so on ...
- Finally, average the performance for each held-out instance



Advantages and Disadvantages?

# Leave-One-Out Cross-Validation

- Advantages
    - multiple rounds of generalization performance.
    - each training fold is as similar as possible to the one we will ultimately use to tune parameters before sending the system out into the world.
- Disadvantage
    - our estimate of generalization performance may still be artificially high
    - why?

# Leave-One-Out Cross-Validation

- Advantages
  - multiple rounds of generalization performance.
  - each training fold is as similar as possible to the one we will ultimately use to tune parameters before sending the system out into the world.
- Disadvantage
  - our estimate of generalization performance may still be artificially high
  - we are likely to try lots of different things and pick the one with the best “generalization” performance
  - still indirectly over-training to the dataset (sigh...)

# Experimentation

Jaime Arguello
INLS 613: Text Data Mining
jarguell@email.unc.edu

October 23, 2017

# Outline

Parameter Tuning

Cross-Validation

Significance tests

# Comparing Systems

- Train and test both systems using 10-fold cross validation
- Use the same folds for both systems
- Compare the difference in average performance across held-out folds

| **Fold** | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.41 | 0.48 |
| | Difference | 0.07 |

# Significance Tests

## motivation

- Why would it be risky to conclude that System B is better System A?
- Put differently, what is it that we're trying to achieve?

# Significance Tests

## motivation

- In theory: that the average performance of System B is greater than the average performance of System A for all possible test sets.
- However, we don’t have all test sets. We have a sample
- And, this sample may favor one system vs. the other!

# Significance Tests

## definition

- A significance test is a statistical tool that allows us to determine whether a difference in performance reflects a true pattern or just random chance

# Significance Tests

## ingredients

- Test statistic: a measure used to judge the two systems (e.g., the difference between their average F-measure)
- Null hypothesis: no “true” difference between the two systems
- P-value: take the value of the observed test statistic and compute the probability of observing a value that large (or larger) under the null hypothesis

# Significance Tests

## ingredients

- If the p-value is large, we cannot reject the null hypothesis
- That is, we cannot claim that one system is better than the other
- If the p-value is small ($p<0.05$), we can reject the null hypothesis
- That is, the observed test-statistic is not due to random chance

# Comparing Systems

- P-value: the probability of observing a difference **equal to or greater than** 0.07 under the null hypothesis (i.e., the systems are actually equally good).

| **Fold** | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.41 | 0.48 |
| | Difference | $\boxed{0.07}$ |

# Fisher's Randomization Test

## procedure

- **Inputs:** counter = 0, N = 100,000
- Repeat N times:

  **Step 1:** for each fold, flip a coin and if it lands 'heads', flip the result between System A and B

  **Step 2:** see whether the test statistic is equal to or greater than the one observed and, if so, increment counter

- **Output:** counter / N

# Fisher's Randomization Test

| Fold | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.41 | 0.48 |
|  | Difference | 0.07 |

# Fisher's Randomization Test

| Fold | System A | System B |
|---|---|---|
| 1 | **0.5** | **0.2** |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | **0.9** | **0.8** |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | **0.5** | **0** |
| 10 | 0.9 | 0.8 |
| Average | 0.5 | 0.39 |
| | Difference | -0.11 |

at least 0.07?

iteration = 1 counter = 0

# Fisher's Randomization Test

| Fold | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | **0.1** | **0.1** |
| 4 | 0.4 | 0.4 |
| 5 | **1** | **1** |
| 6 | 0.8 | 0.9 |
| 7 | **0.1** | **0.3** |
| 8 | **0.2** | **0.1** |
| 9 | 0 | 0.5 |
| 10 | **0.08** | **0.9** |
| Average | 0.318 | 0.5 |
| | Difference | 0.182 |

at least 0.07?

iteration = 2 counter = 1

# Fisher's Randomization Test

| Fold | System A | System B |
|---|---|---|
| 1 | **0.5** | **0.2** |
| 2 | 0.3 | 0.3 |
| 3 | **0.1** | **0.1** |
| 4 | **0.4** | **0.4** |
| 5 | 1 | 1 |
| 6 | **0.9** | **0.8** |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | **0.5** | **0** |
| 10 | 0.9 | 0.8 |
| Average | 0.5 | 0.39 |
|  | Difference | -0.11 |

at least 0.07?

iteration = 100,000 counter = 25,678

# Fisher's Randomization Test

## procedure

- **Inputs:** counter = 0, N = 100,000
- Repeat N times:

  **Step 1:** for each query, flip a coin and if it lands 'heads', flip the result between System A and B

  **Step 2:** see whether the test statistic is equal to or greater than the one observed and, if so, increment counter

- **Output:** counter / N = (25,678/100,00) = 0.25678

# Fisher's Randomization Test

- Under the null hypothesis, the probability of observing a value of the test statistic of 0.07 or greater is about 0.26.
- Because $p > 0.05$, we cannot confidently say that the value of the test statistic is <u>not</u> due to random chance.
- A difference between the average F-measure values of 0.07 is not significant

# Fisher's Randomization Test

## procedure

- **Inputs:** counter = 0, N = 100,000
- Repeat N times:

  **Step 1:** for each query, flip a coin and if it lands 'heads', flip the result between System A and B

  **Step 2:** see whether the test statistic is equal to or greater than the one observed and, if so, increment counter

- **Output:** counter / N = (25,678/100,00) = 0.25678

This is a one-tailed test (B > A).
How can we modify it to be a two-tailed test (B != A)

# Fisher's Randomization Test

## procedure

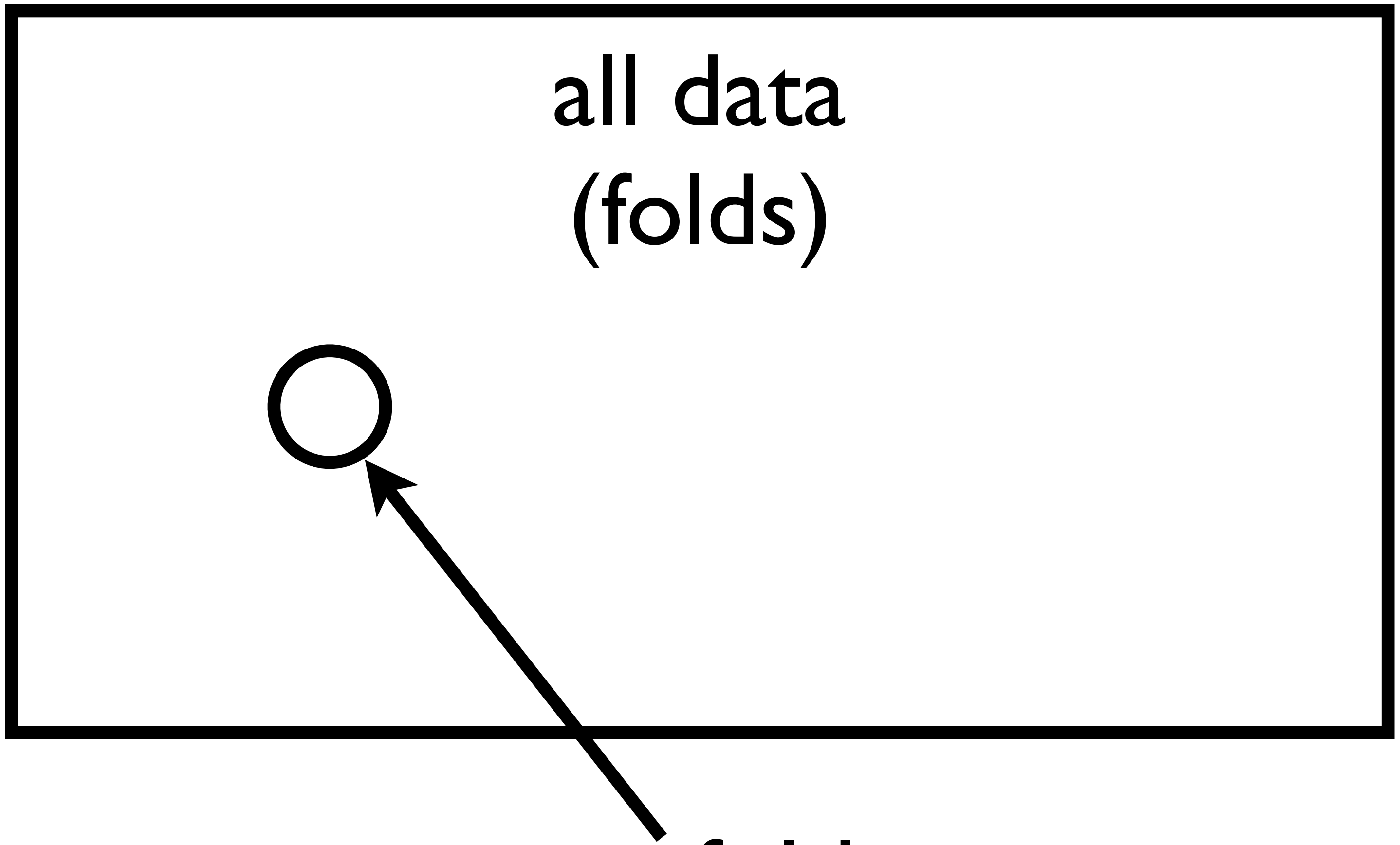
- P-value: the probability of observing a difference *in the absolute value* **equal to or greater than** 0.07 under the null hypothesis (i.e., the systems are actually equal).

| **Fold** | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.41 | 0.48 |
| | Difference | 0.07 |

# Bootstrap-Shift Test

## motivation

- Our sample is a representative sample of all data



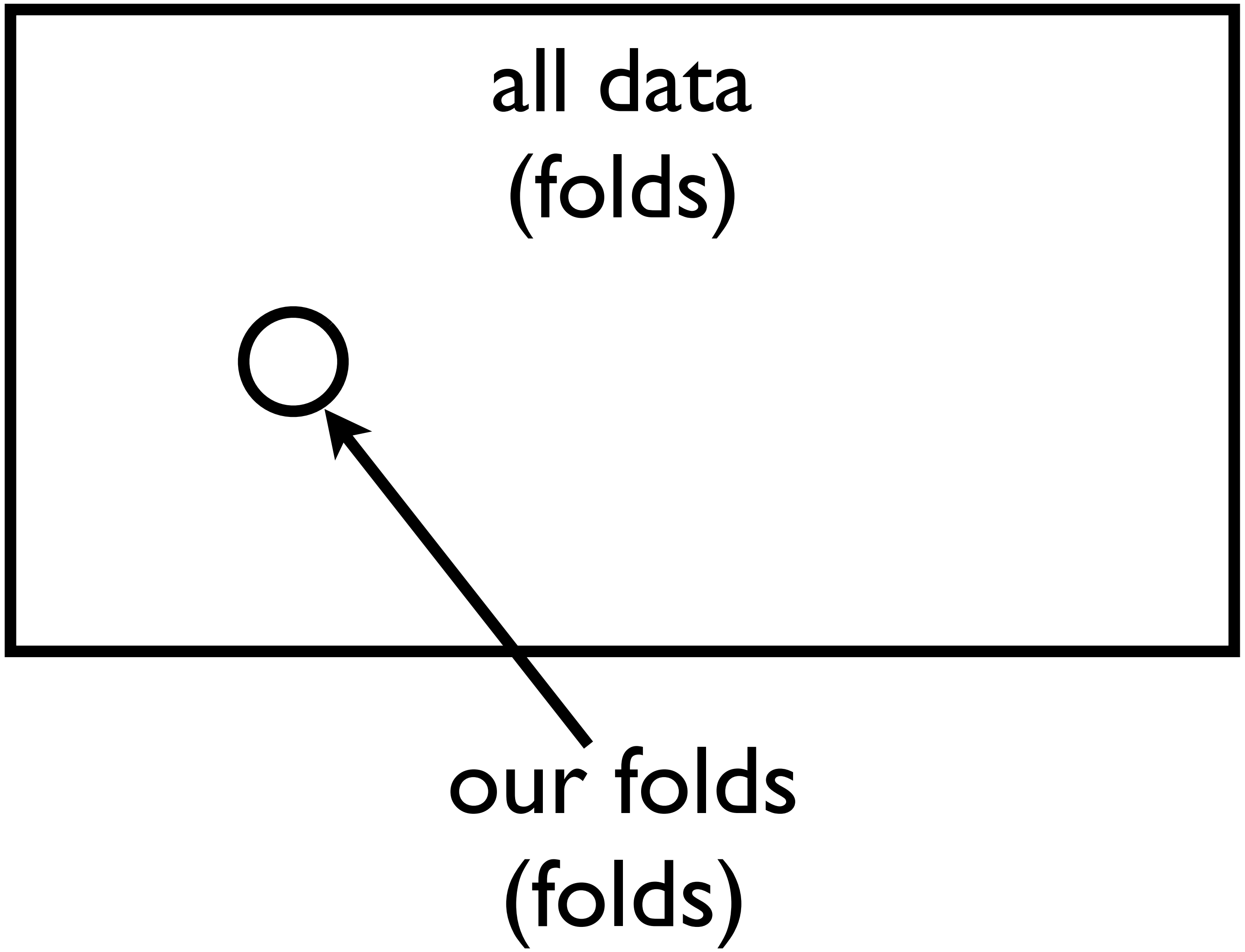
all data
(folds)

our folds
(folds)

# Bootstrap-Shift Test

## motivation

- If we sample (with replacement) from our sample, we can generate a new representative sample of all data



all data
(folds)

our folds
(folds)

# Bootstrap-Shift Test

## procedure

- **Inputs:** Array T = {}, N = 100,000
- Repeat N times:

  **Step 1:** sample 10 folds (with replacement) from our set of 10 folds (called a subsample)

  **Step 2:** compute test statistic associated with new sample and add to T

- **Step 3:** compute <u>average</u> of numbers in T
- **Step 4:** reduce every number in T by <u>average</u>
- **Output:** % of numbers in T greater than or equal to the observed test statistic

# Bootstrap-Shift Test

## procedure

- **Inputs:** Array T = {}, N = 100,000
- Repeat N times:

  **Step 1:** sample 10 folds (with replacement) from our set of 10 folds (called a subsample)

  **Step 2:** compute test statistic associated with new sample and add to T

- **Step 3:** compute average of numbers in T
- **Step 4:** reduce every number in T by average
- **Output:** % of numbers in T greater than or equal to the observed test statistic

# Bootstrap-Shift Test

| Fold | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 5 | 1 | 1 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.41 | 0.48 |
|  | Difference | 0.07 |

# Bootstrap-Shift Test

| Fold | System A | System B | **sample** |
|---|---|---|---|
| 1 | 0.2 | 0.5 | **0** |
| 2 | 0.3 | 0.3 | **1** |
| 3 | 0.1 | 0.1 | **2** |
| 4 | 0.4 | 0.4 | **2** |
| 5 | 1 | 1 | **0** |
| 6 | 0.8 | 0.9 | **1** |
| 7 | 0.3 | 0.1 | **1** |
| 8 | 0.1 | 0.2 | **1** |
| 9 | 0 | 0.5 | **2** |
| 10 | 0.9 | 0.8 | **0** |

iteration = 1

# Bootstrap-Shift Test

| Fold | System A | System B |
|---|---|---|
| 2 | 0.3 | 0.3 |
| 3 | 0.1 | 0.1 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 4 | 0.4 | 0.4 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 9 | 0 | 0.5 |
| Average | 0.25 | 0.35 |
| | Difference | **0.1** |

T = {**0.10**}

iteration = 1

# Bootstrap-Shift Test

| Fold | System A | System B | **sample** |
|---|---|---|---|
| 1 | 0.2 | 0.5 | **0** |
| 2 | 0.3 | 0.3 | **0** |
| 3 | 0.1 | 0.1 | **3** |
| 4 | 0.4 | 0.4 | **2** |
| 5 | 1 | 1 | **0** |
| 6 | 0.8 | 0.9 | **1** |
| 7 | 0.3 | 0.1 | **1** |
| 8 | 0.1 | 0.2 | **1** |
| 9 | 0 | 0.5 | **1** |
| 10 | 0.9 | 0.8 | **1** |

$T = \{\mathbf{0.10}\}$

iteration = 2

# Bootstrap-Shift Test

| Fold | System A | System B |
|---|---|---|
| 3 | 0.1 | 0.1 |
| 3 | 0.1 | 0.1 |
| 3 | 0.1 | 0.1 |
| 4 | 0.4 | 0.4 |
| 4 | 0.4 | 0.4 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 9 | 0 | 0.5 |
| 10 | 0.9 | 0.8 |
| Average | 0.32 | 0.36 |
|  | Difference | **0.04** |

T = {**0.10**, **0.04**}

iteration = 2

# Bootstrap-Shift Test

| Fold | System A | System B |
|---|---|---|
| 1 | 0.2 | 0.5 |
| 1 | 0.2 | 0.5 |
| 4 | 0.4 | 0.4 |
| 4 | 0.4 | 0.4 |
| 4 | 0.4 | 0.4 |
| 6 | 0.8 | 0.9 |
| 7 | 0.3 | 0.1 |
| 8 | 0.1 | 0.2 |
| 8 | 0.1 | 0.2 |
| 10 | 0.9 | 0.8 |
| Average | 0.38 | 0.44 |
|  | Difference | **0.06** |

iteration = 100,000

T = {**0.10**,
**0.04**,
……,
**0.06**}

# Bootstrap-Shift Test

## procedure

- **Inputs:** Array T = {}, N = 100,000
- Repeat N times:

  **Step 1:** sample 10 folds (with replacement) from our set of 10 folds (called a subsample)

  **Step 2:** compute test statistic associated with new sample and add to T

- **Step 3:** compute average of numbers in T
- **Step 4:** reduce every number in T by average
- **Output:** % of numbers in T' greater than or equal to the observed test statistic

# Bootstrap-Shift Test

## procedure

- For the purpose of this example, let’s assume N = 10.

T = {**0.10**,
**0.04**,
**0.21**,
**0.20**,
**0.13**,
**0.09**,
**0.22**,
**0.07**,
**0.03**,
**0.11**}

**Step 3**

Average = **0.12**

**Step 4**

T’= {**-0.02**,
**-0.08**,
**0.09**,
**0.08**,
**0.01**,
**-0.03**,
**0.10**,
**-0.05**,
-**0.09**,
**-0.01**}

# Bootstrap-Shift Test

## procedure

- **Inputs:** Array T = {}, N = 100,000
- Repeat N times:

  **Step 1:** sample 10 folds (with replacement) from our set of 10 folds (called a subsample)

  **Step 2:** compute test statistic associated with new sample and add to T

- **Step 3:** compute average of numbers in T
- **Step 4:** reduce every number in T by average
- **Output:** % of numbers in T' greater than or equal to the observed test statistic

# Bootstrap-Shift Test

## procedure

- **Output:** (3/10) = **0.30**

T = {**0.10**,
**0.04**,
**0.21**,
**0.20**,
**0.13**,
**0.09**,
**0.22**,
**0.07**,
**0.03**,
**0.11**}

**Step 3**

Average = **0.12**

**Step 4**

T’= {**-0.02**,
**-0.08**,
**0.09**,
**0.08**,
**0.01**,
**-0.03**,
**0.10**,
**-0.05**,
-**0.09**,
**-0.01**}

# Bootstrap-Shift Test

## procedure

- **Output:** (3/10) = **0.30**

T = {**0.10**,
**0.04**,
**0.21**,
**0.20**,
**0.13**,
**0.09**,
**0.22**,
**0.07**,
**0.03**,
**0.11**}

This is a one-tailed test. How can we modify it to be a two-tailed test?

**Step 3**

Average = **0.12**

**Step 4**

T'= {**-0.02**,
**-0.08**,
**0.09**,
**0.08**,
**0.01**,
**-0.03**,
**0.10**,
**-0.05**,
-**0.09**,
**-0.01**}

# Significance Tests

## summary

- Significance tests help us determine whether the outcome of an experiment signals a “true” trend
- The null hypothesis is that the observed outcome is due to random chance (sample bias, error, etc.)
- There are many types of tests
- Parametric tests: assume a particular distribution for the test statistic under the null hypothesis
- Non-parametric tests: make no assumptions about the test statistic distribution under the null hypothesis
- The randomization and bootstrap-shift tests make no assumptions, are robust, and easy to understand