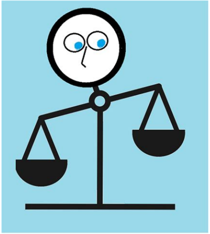
# MS Project Presentation

## Bias-Variance Tradeoff and Cross-Validation

Funfay Jen

2022-01-03



Section 1

# Background and Motivation

# What is the Bias-Variance Tradeoff?

- A central problem in **supervised learning**
- Desires a model that simultaneously:
  - Captures the regularities in its training data
  - Generalizes well to unseen data
- Unfortunately,
  - High-variance learning methods may be able to represent their training set well but are at risk of overfitting to noisy or unrepresentative training data.
  - In contrast, algorithms with high bias typically produce simpler models that may fail to capture important regularities (i.e. underfit) in the data.

# Mathematics

- $f$: fixed but unknown function
- $X$: can be univariate or multivariate
- $Y$: univariate
- $\epsilon$: zero mean, finite variance

# Mathematics

$$E_{D,\epsilon}[(Y-\hat{f}(X;D))^2] = (Bias_D[\hat{f}(X;D)])^2 + Var_D[\hat{f}(X;D)]) + \sigma^2 \quad (1)$$

where

$$Bias_D[\hat{f}(X;D) = E_D[\hat{f}(X;D)]] - f(X) \quad (2)$$

and

$$Var_D[\hat{f}(X;D)] = E_D[(E_D[\hat{f}(X;D)] - \hat{f}(X;D))^2] \quad (3)$$

# Mathematics

Since all three terms are non-negative, the irreducible error forms a lower bound on the expected error on unseen samples.

- $\sigma^2$: lowest achievable predictive MSE
- bias and variance: adds to it to result in a total MSE
- tuning parameter: not visible in Equations (1), (2), (3), but modulates the relative contributions of these terms

Section 2

# Method of Simulating the Bias and Variance Tradeoff

# At a high level

- Visualized as a plot of the total expected mean squared error (MSE) along with its three components (bias, variance and the irreducible error) against one or more hyperparameters.
  - Compute the expectations in Equations (1), (2), and (3) via Monte Carlo simulation
  - Do this for each set of hyperparameters to trace out a path (or high-dimensional surface)
  - Visualize the resulting bias-variance tradeoff

# Procedurally

(Refer to the report for details)

1. Generate Training Sets
2. Generate Test Sets
3. Deploy the Statistical Method at a given set of hyperparameters
4. Compute Bias, Variance and MSE according to Equations (2), (3), and (1)
5. Repeat steps 1-4 over a grid of (sets of) hyperparameters and visualize the resulting mean squared error profile and how it is decomposed into its components.

Section 3

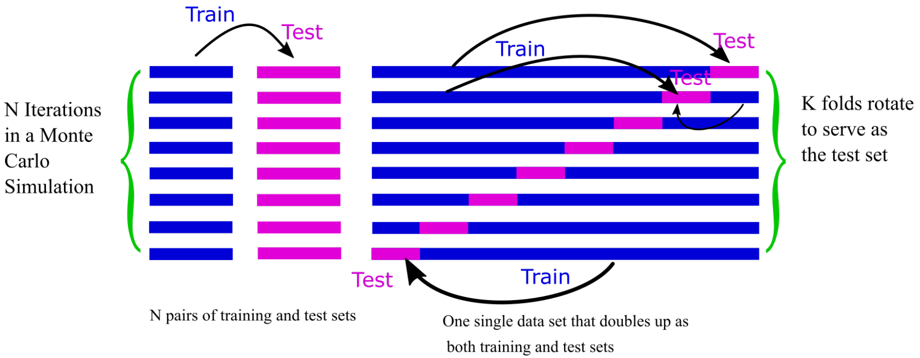
# Method of Cross-Validation

# Setup

- Generate *one* single set of data containing observations
- Randomly divide the set of observations into $K$ folds of approximately equal size.
- Each fold $i$, $1 \leq i \leq K$ is treated as a test set and the method is fit on the remaining $K-1$ folds. The mean squared error, $MSE_i$, is computed on the observations in the held-out fold.
- The K-fold CV estimate is computed by averaging these values,

$$CV_{(K)} = \frac{1}{K}\sum_{i=1}^{K} MSE_i \tag{4}$$

# # Relationship between the Bias-Variance Tradeoff and Cross-Validation



# Recap of the Main Differences

- With CV, we cannot determine the bias component since in practice we do not know the true underlying function $f$;
- With CV, we cannot determine the variance component since in practice we often only have one set of data available and the $K$ folds typically do not have the same $X$ values–we cannot properly define variance in this setting;
- With CV, we can still compute an average MSE over the folds. However, one issue is that the MSE's on different folds become correlated.

Section 4

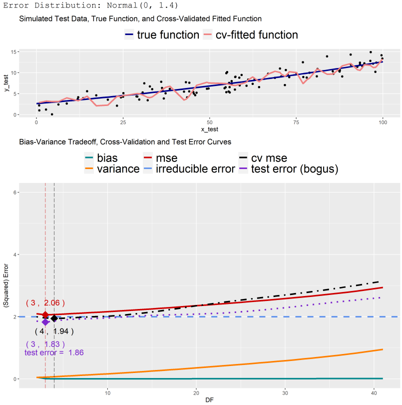
# Simulation Studies

# Exploring Smoothing Splines, Penalized Regression, Boosting, and SVM

- Draw $X$ from a (univariate or multivariate) uniform distribution and the error from a normal distribution (or other distributions) with the same variance.
- Overlay the cross-validated MSE curve obtained from a separate training set.
- Overlay the test MSE curve from an independent test set
  - This is “bogus” since in real scenarios We wouldn’t have been able to find the MSE since the response values are typically unknown.
  - We do this to get a sense of how close we would have been to the (bogus, but practical) optimum for that particular test set if we used the hyperparameter from CV.

# Smoothing Splines

$f_1$



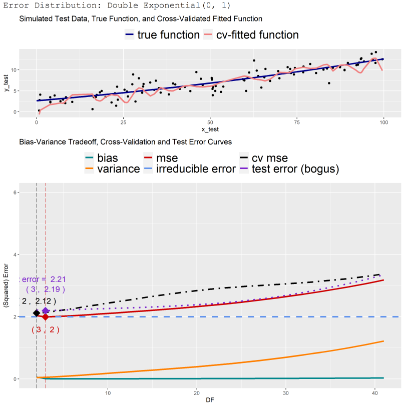




**Figure 1:** Normal Errors (Left) and Double Expenential Errors (Right)

# Smoothing Splines

$f_2$

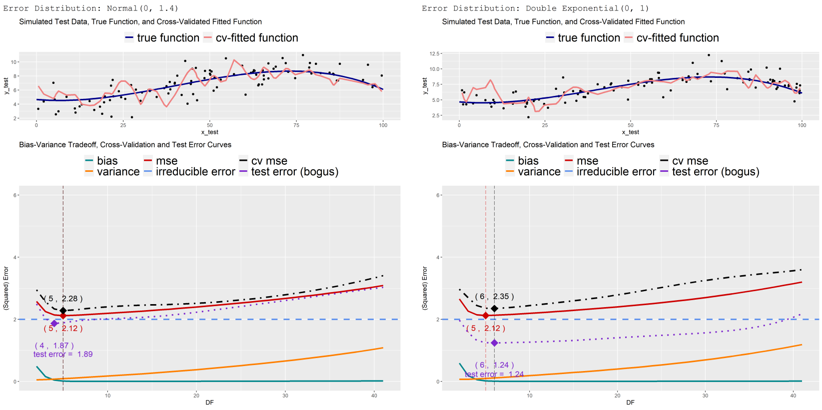



**Figure 2:** Normal Errors (Left) and Double Expenential Errors (Right)

# Smoothing Splines

$f_3$

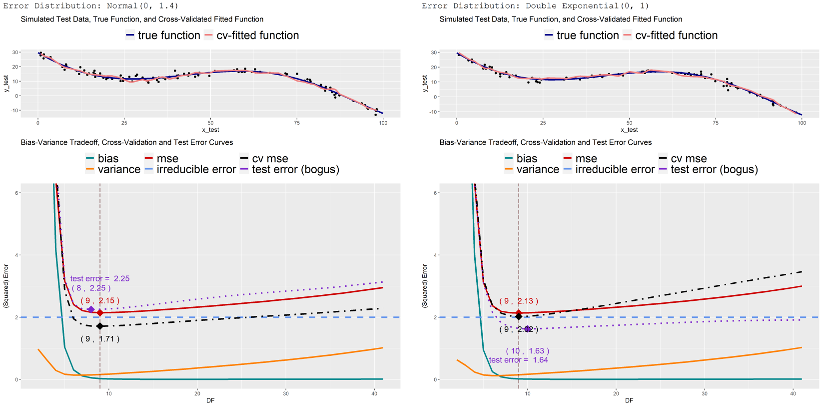



**Figure 3:** Normal Errors (Left) and Double Expenential Errors (Right)

# Remarks

# Penalized Regression $f_1$

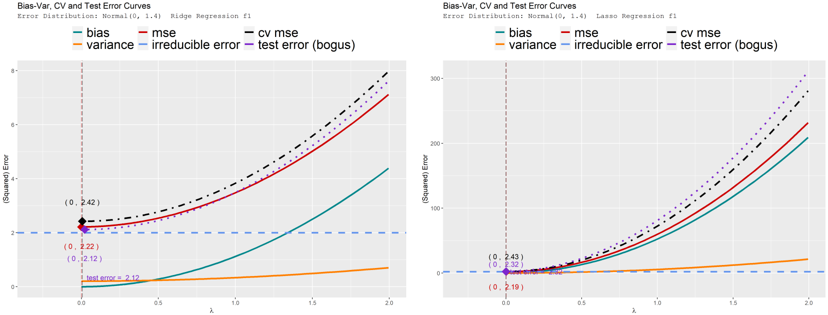



**Figure 4:** RIdge (Left) and Lasso (Right)

# Penalized Regression $f_2$

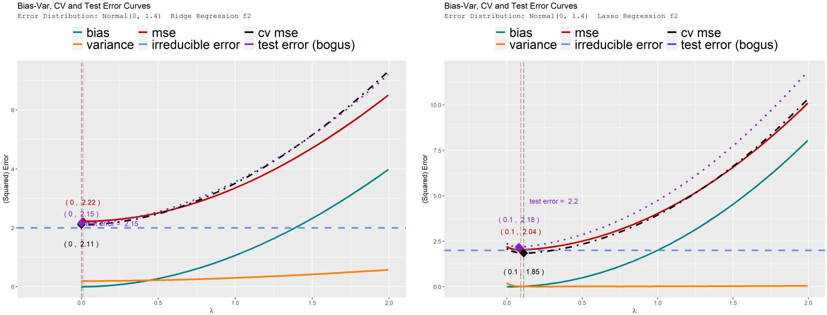



**Figure 5:** RIdge (Left) and Lasso (Right)

# Penalized Regression $f_3$

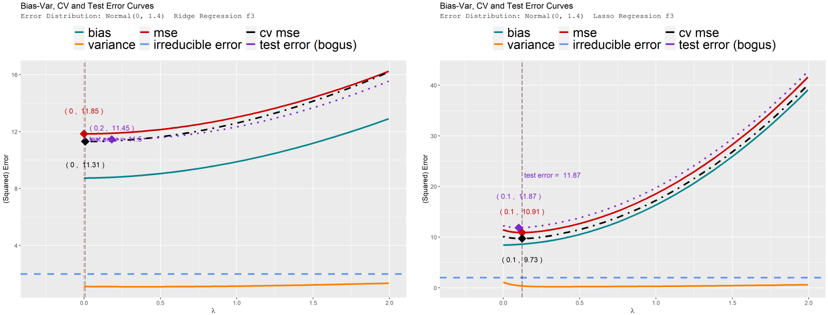



**Figure 6:** RIdge (Left) and Lasso (Right)

# Penalized Regression $f_4$

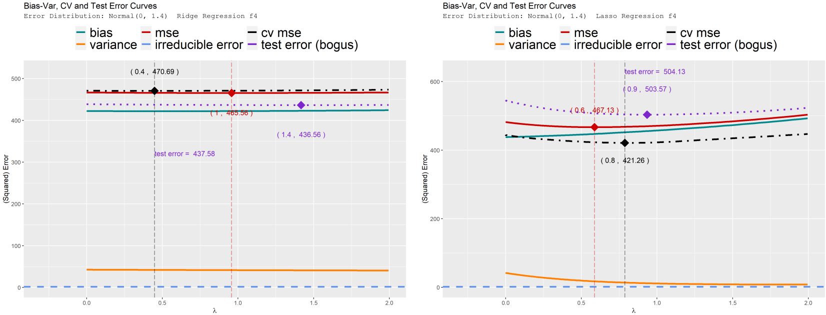
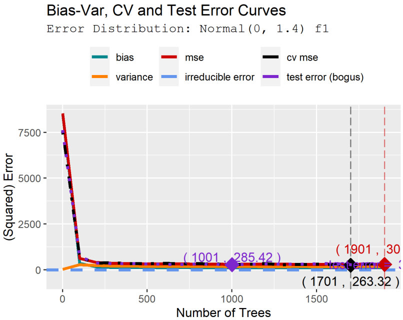



**Figure 7:** RIdge (Left) and Lasso (Right)

# Remarks

# Boosting $f_1$, $f_2$



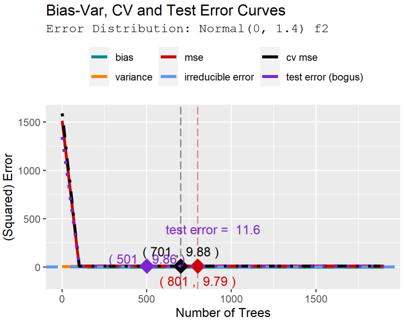




**Figure 8:** Boosting

# Boosting $f_3$, $f_4$

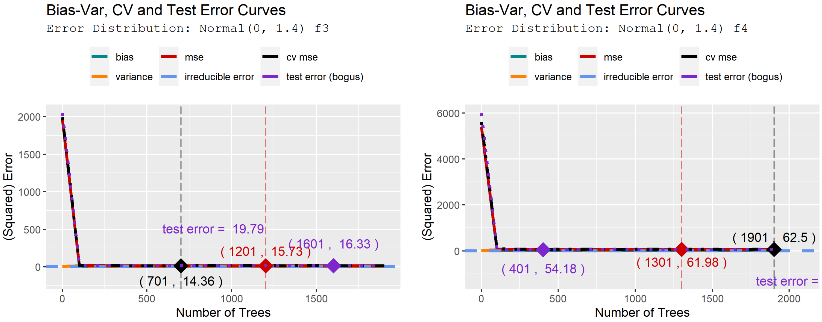
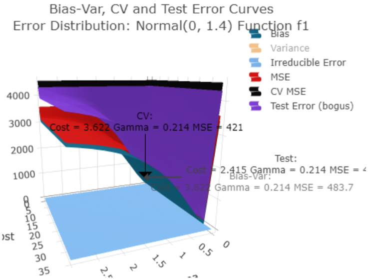



**Figure 9:** Boosting

# Remarks

# SVM $f_1$, $f_2$



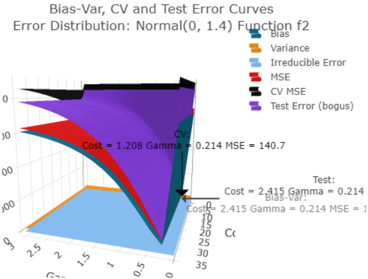




**Figure 10:** SVM

# SVM $f_3$, $f_4$

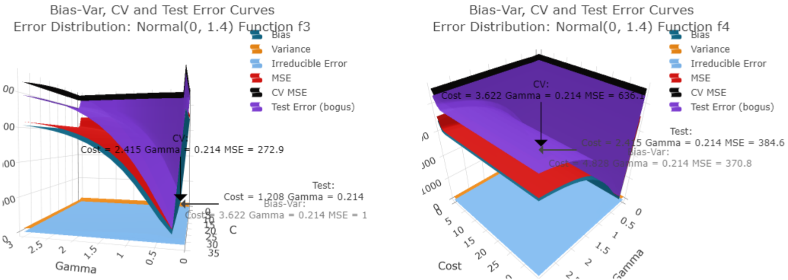



**Figure 11:** SVM

# Remarks

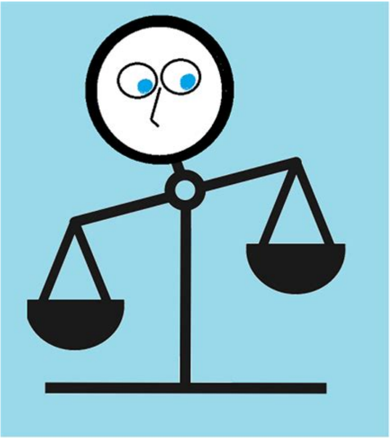


**Figure 12:** It's all about tradeoffs!

Section 5

# Q & A

# Thank You!