Model Selection

Model Selection

Box: “All models are wrong, but some are useful.”

Occam’s Razor: “It is futile to do more with what can be done with less.”
“The simplest explanation is best!”

Einstein: “Everything should be made as simple as possible, but no simpler.”

Bias

- Suppose $\boldsymbol{\theta}^*$ is our model parameter, and $\boldsymbol{\theta}$ is our estimator.
- The function $\boldsymbol{\theta}$ applied to data $x \sim X|\boldsymbol{\theta}^*$ yields a point estimate.
- $E_{x \sim X|\boldsymbol{\theta}^*}[\boldsymbol{\theta}(x)]$ is the expected value of the estimator.
- $\text{Bias}[\boldsymbol{\theta}, \boldsymbol{\theta}^*] = E_{x \sim X|\boldsymbol{\theta}^*}[\boldsymbol{\theta}(x)] - \boldsymbol{\theta}^*$

Mean-squared Error

- Suppose $\boldsymbol{\theta}^*$ is our model parameter, and $\boldsymbol{\theta}$ is our estimator.
- The function $\boldsymbol{\theta}$ applied to data $x \sim X|\boldsymbol{\theta}^*$ yields a point estimate.
- Mean-squared error is the expected residual error.
- $\mathrm{MSE}(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \mathrm{E}_{x \sim X|\boldsymbol{\theta}^*}[(\boldsymbol{\theta}(x) - \boldsymbol{\theta}^*)^2]$

Bias-Variance Decomposition

- Theorem: $\text{MSE}(\boldsymbol{\theta}, \boldsymbol{\theta}^*) = \text{Bias}^2[\boldsymbol{\theta}, \boldsymbol{\theta}^*] + \text{Var}[\boldsymbol{\theta}]$
- So MSE is a combination of bias and variance in our estimator.
- Ideally, we would reduce both, but this is often impossible.
- Instead, we usually trade off one against the other.

Proof of Bias-Variance Decomposition

$\text{Bias}^2[\theta, \theta^*] = (E_X[\theta(x)] - \theta^*)^2 = (E_X[\theta(x)] - \theta^*)(E_X[\theta(x)] - \theta^*) = (E_X[\theta(x)])^2 - 2\theta^* E_X[\theta(x)] + (\theta^*)^2$

$\text{Var}[\theta] = E_X[(\theta(x) - E_X[\theta(x)])^2] = E_X[\theta^2(x)] - (E_X[\theta(x)])^2$

$\text{Bias}^2[\theta, \theta^*] + \text{Var}[\theta] = (E_X[\theta(x)])^2 - 2\theta^* E_X[\theta(x)] + (\theta^*)^2 + E_X[\theta^2(x)] - (E_X[\theta(x)])^2$

$= -2\theta^* E_X[\theta(x)] + (\theta^*)^2 + E_X[\theta^2(x)]$ (the first and last terms cancel)

$\text{MSE}(\theta, \theta^*) = E_X[(\theta(x) - \theta^*)^2] = E_X[\theta^2(x) - 2\theta^*\theta(x) + (\theta^*)^2] = E_X[\theta^2(x)] - 2\theta^* E_X[\theta(x)] + (\theta^*)^2$

Bias-Variance Tradeoff

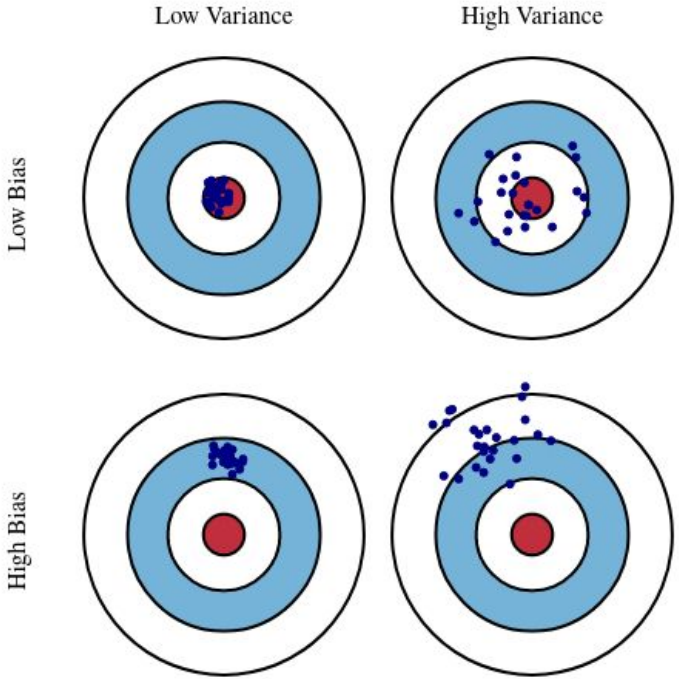


Image Source

Model Selection

Find a model that appropriately balances complexity and generalization capabilities: i.e., that optimizes the bias-variance tradeoff.

- High bias, low variance (underfitting)
  - Build a model with only very few variables/features
  - Assume a simple relationship among variables (e.g., linear)
  - Make strong structural assumptions—so many, the data can barely be heard
- Low bias, high variance (overfitting)
  - Make weak structural assumptions
  - Allow for a complex relationship among variables (e.g., highly non-linear)
  - The analyst defers to the data almost entirely; their own domain expertise is suppressed
  - As for variables/features, throw in everything in the kitchen sink!

Overfitting

- Problem: Models are always biased towards training data
- A model overfits when it “memorizes” the training data
- Overfit models cannot generalize well to test data
- Solution: Use test data to evaluate models to mitigate the risk that they overfit

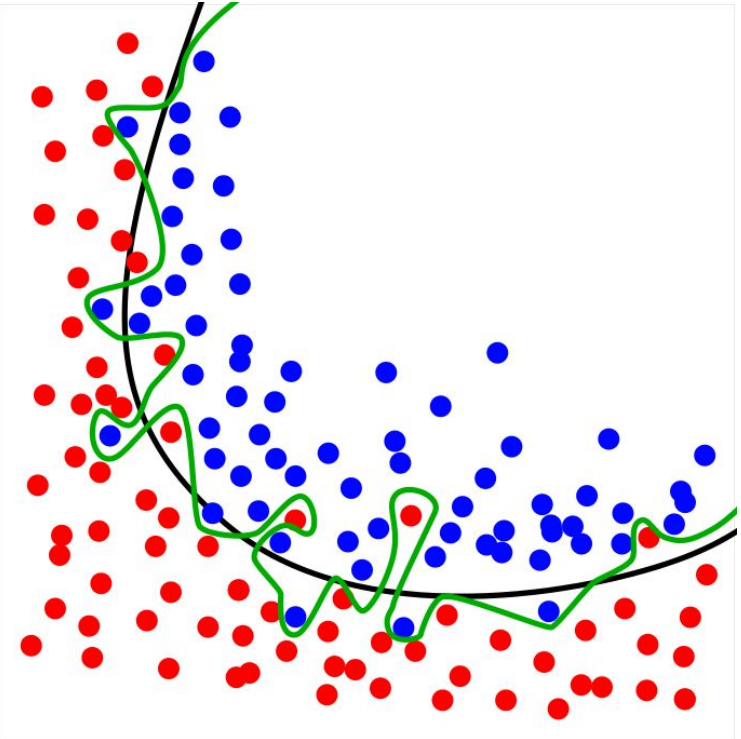
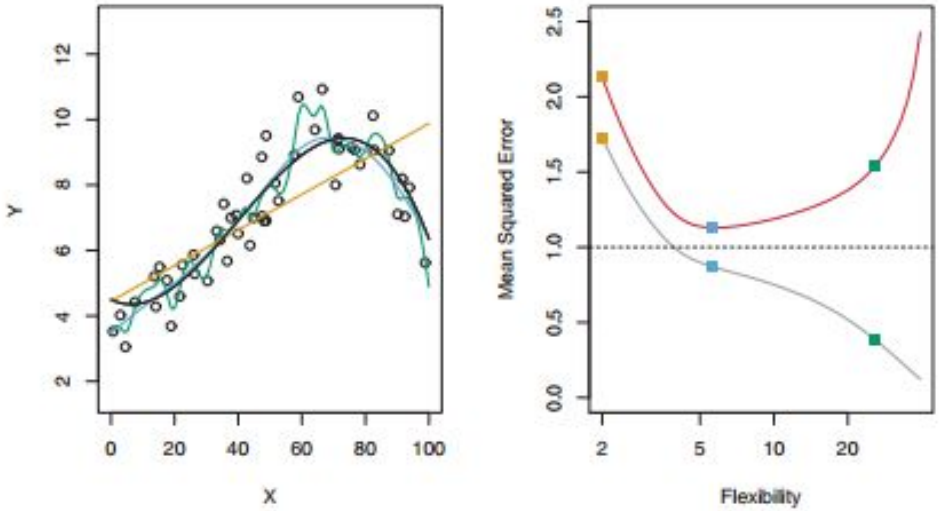


Image Source

Train vs. Test Error

- The points are the data
- The black line represents the true relationship
- The various colors refer to different estimators (linear, quadratic, & something crazy).



- The grey curve shows the training error. It decreases indefinitely.
- The red curve shows the test error. It has an elbow.
- The colored boxes correspond to the colored fits in the left plot.

Image Source

Underfitting vs. Overfitting

- Simple (e.g., linear) models are highly biased; as such, they often underfit, meaning they fail to capture regularities in the data.
- Otoh, they are not sensitive to noise (i.e., they assume so much bias that they don't change much with the data), so are comparatively low variance.

- More complicated models are less biased. Because of their flexibility, they end up modeling noise (as well as signal), and consequently overfit.
- Flexible models have high variance, b/c the models themselves can vary enormously with the data.

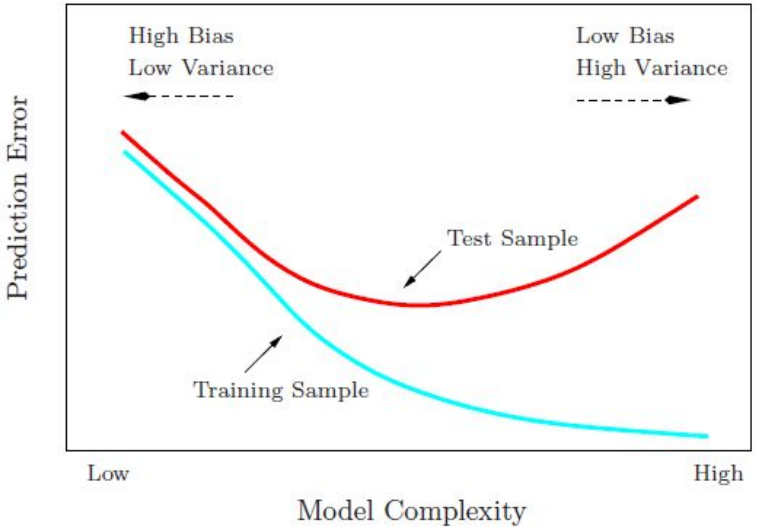


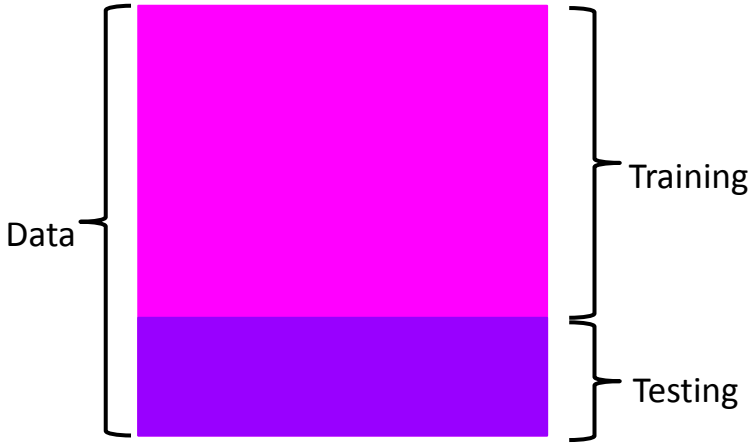
Image Source

Training vs. Test Data

- Divide data into two sets: training set and test set
- As their names suggest:
  - Train your model on the training set
  - Test your model on the test set
- Goal is to build a model that generalizes well from the training set to the test set, i.e., from in-sample data to out-of-sample data.
- To achieve this goal, the test set should be representative of the training set, and should be large enough to obtain statistically significant results.

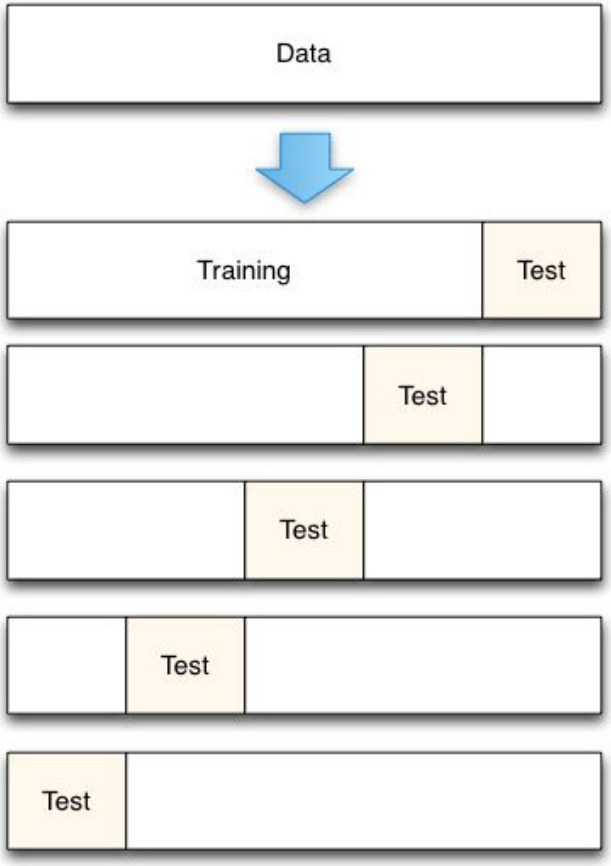
Holdout Method (to evaluate model accuracy)

- Partition our training data into a large training set and smaller testing set
- Train model on the training data
- Test model accuracy on the testing data
- Data are often shuffled first (e.g., if they were compiled by different sources)



Cross validation

- Partition data multiple times
  - If you want to partition your data 10 times, create 10 folds, and then use each fold as a test set, and the rest of the data as a training set
  - Average accuracy across all partitions to approximate model accuracy
- This is called cross validation
  - Typical to use *k* partitions for *k*-fold cross validation (usually *k* = 10)
  - Leave-one-out cross validation: cross validation, to the extreme: *k* = *n*, the sample size
- Cross validation is useful for model selection



Linear Regression, Regularized

Regularization

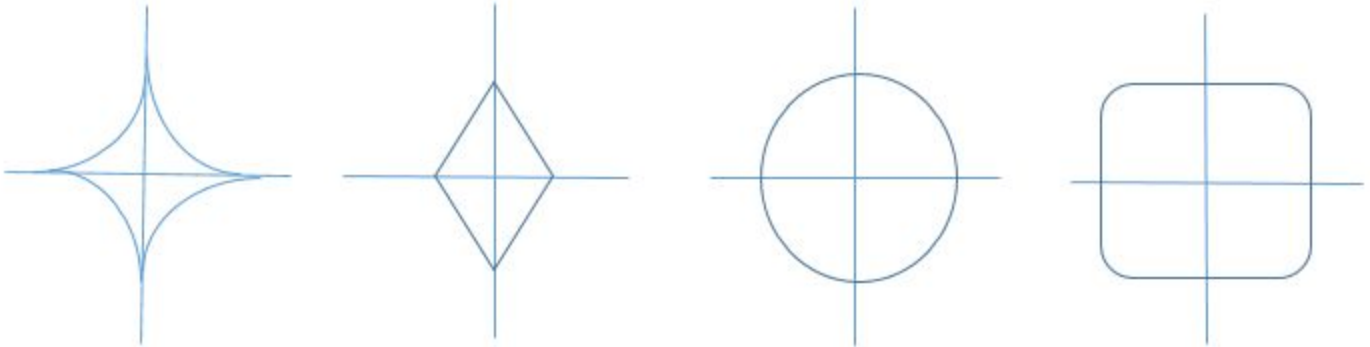
- The bias-variance decomposition suggests trading bias for variance.
- Regularization is a technique that introduces bias to reduce variance.
- Shrinkage is a form of regularization that shrinks estimates towards zero.
- This technique discourages learning a more complex, flexible model, thereby mitigating the risk of overfitting.

Regularizers

- A regularizer is a penalty term that is added to an objective function (e.g., minimize the sum of the squared residuals) to penalize large coefficients.
- Two popular choices lead to two popular variants on standard regression
- Ridge regression: Minimizes the sum of the coefficients squared
- LASSO: Minimizes the sum their absolute values
  - Least absolute shrinkage and selection operator
- Elastic Net: Minimizes a combination of the two

Regularizers Visualized in 2d

Assuming two coefficients, an increase in one is offset by a decrease in the other



$$\sum_i |\theta_i|^p = |\theta_1| + |\theta_2|.$$

Image Source

λ

- The new, regularized objective must balance the original objective against the regularization term.
- This balance is achieved via a parameter λ, the weight of the regularization term, with 1 - λ as the weight of the original objective.
- Higher λ increases bias, so decreases variance.

Two (really three) Sources of Error

- Error = Reducible Error + Irreducible Error
- MSE(θ, θ^*) is reducible error. It varies from one learning algorithm to another, presenting opportunities for improvement.
  - Reducible error decomposes into terms of bias and variance.
- Irreducible error arises because *Y* is almost never (except in toy examples) completely determined by *X*.
  - Noise in a statistical model represents missing information.
  - The variance of this noise is the model's irreducible error.
  - This error cannot be reduced, except perhaps by changing the model: e.g., adding variables/features.

Variable (i.e., Feature) Selection

- Identifying independent variables whose relationship to the dependent variable is “important”.
- Simple heuristic for eliminating variables from a model: is the coefficient (essentially) zero?
- Ridge regression does not set coefficients to zero, unless $\lambda = 0$, so it cannot be used for variable selection.
- So, while ridge regression is useful for prediction, it is less effective when the goal to explain relationships among variables.
- LASSO, however, can set some coefficients to zero, so it is more popular, and widely used when the goal is to build an interpretable model.

Curse of Dimensionality

Curse of Dimensionality

Adding new features to a model (i.e., increasing the dimensionality) in the hopes of improving performance will eventually degrade performance

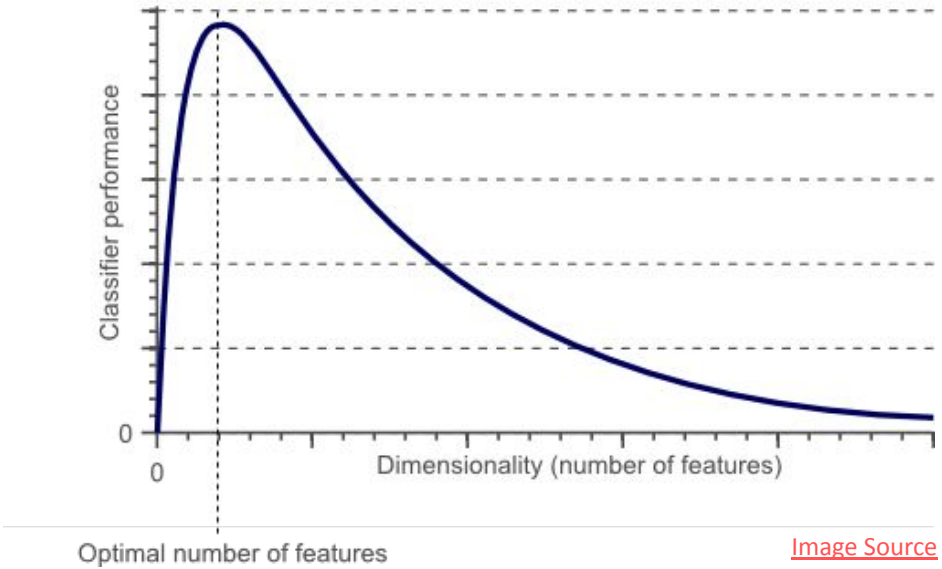


Image Source

Curse of Dimensionality

As the dimensionality of the data (i.e., the number of features) increases, Wikipedia
"the volume of the space increases so fast that the available data become sparse."

Example from Stack Exchange:

- To find on your favorite kind of cookie among four possible flavors (sweet, salty, bitter, sour), requires eating four cookies.
- If there is an additional dimension, e.g., color, and there are three possible colors, you now now have to eat 4 *x* 3 = 12 cookies to find your favorite.
- Add another dimension, e.g., shape, with five possibilities, and you now have to eat 4 *x* 3 *x* 5 = 60 cookies!

Image Source

Curse of Dimensionality

As the dimensionality of the data (i.e., the number of features) increases, Wikipedia
"the volume of the space increases so fast that the available data become sparse."

To cover 20% of the population:

- Need 20% of the data in 1 dimension: $(.2)^1 \sim .2$
- Need 45% of the data in 2 dimensions: $(.45)^2 \sim .2$
- Need 58% of the data in 3 dimensions: $(.58)^3 \sim .2$