

Lecture 1: Course logistics, introduction, bias-variance tradeoff

STATS 202: Data Mining and Analysis

Linh Tran

tranlm@stanford.edu



Department of Statistics
Stanford University

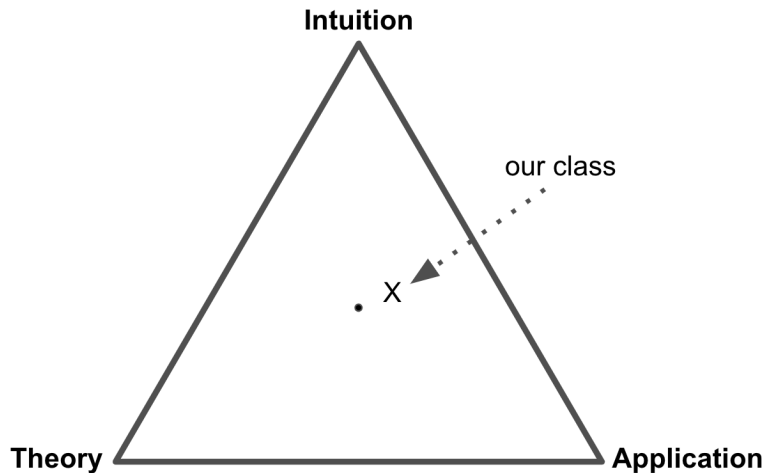
June 21, 2021



- ▶ **Class website:** *stats-202.github.io*
- ▶ **Videos:** Zoom lectures will be recorded/uploaded to *Canvas* (code: 378136).
- ▶ **Textbook:** *An Introduction to Statistical Learning*
 - ▶ **Supplemental Textbook:** *The Elements of Statistical Learning*
- ▶ **Email policy:** Please use *Piazza* (code: 2021202) for most questions. Homeworks and Exams should be submitted via *Gradescope* (code: 864BGD).
- ▶ **Office hours:** Please refer to this *Google calendar*.



- ▶ **Topics:** Intro to statistical learning and methods for analyzing large amounts of data
- ▶ **Prereqs:** STATS 60, MATH 51, CS 105
- ▶ **Grades:** 3 components
 - ▶ *4 homework assignments* (50pts each)
 - ▶ Due by 3:59pm PDT of due date
 - ▶ Accepted up to 2-days late w/ 20% penalty (2 free late days)
 - ▶ Take home midterm on Wednesday, July 21 (100pts)
 - ▶ *Final project* (200pts)
 - ▶ Submissions due on Monday, August 23, 2021
 - ▶ Write-up due on Friday, August 27, 2021





While the course textbook uses **R**, you are free to choose between R and Python. Some thoughts:

- ▶ *R (style guide)*
 - ▶ Good visualizations
 - ▶ More detailed result outputs
 - ▶ Embraced by statistician community
- ▶ *Python (style guide)*
 - ▶ Good scalability
 - ▶ More detailed debugging logs
 - ▶ Embraced by ML community

10% of your assignment grade is based upon the organization + readability of your code

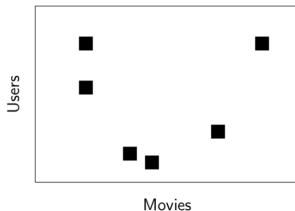


Companies are paying lots of money for statistical models.

- ▶ *Netflix*
- ▶ *Heritage Provider Network*
- ▶ *Department of Homeland Security*
- ▶ *Zillow*
- ▶ Etc...

Popularized prediction challenges by organizing an open, blind contest to improve its recommendation system.

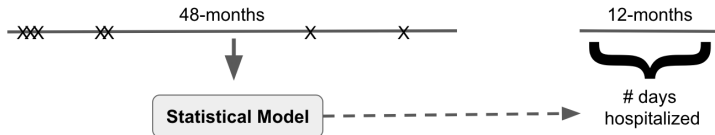
- ▶ **Prize:** \$1 million
- ▶ **Features:** User ratings (1 to 5 stars) on previously watched films
- ▶ **Outcome:** User ratings (1 to 5 stars) for unwatched films





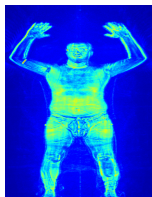
Ran for two years, with six milestone prizes during that span.

- ▶ **Prize:** \$3 million (\$500K)
- ▶ **Features:** Anonymized patient data over a 48 month period
- ▶ **Outcome:** How many days a patient will spend in a hospital in the next year



Improving the algorithms used by TSA to detect potential threats from body scans.

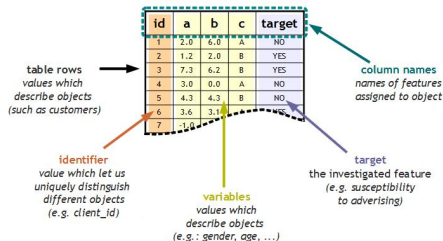
- ▶ **Prize:** \$1 million
- ▶ **Features:** Body scan images
- ▶ **Outcome:** Whether a given body zone has a threat present



Convolution
Pooling
Softmax
Other

Improve the algorithm that “*changed the world of real estate.*”

- ▶ **Prize:** \$1.2 million
- ▶ **Features:** Features related to a given house (e.g. sqft, # beds, # baths, etc.)
- ▶ **Outcome:** The sales price for a given house





Common scenario: I have a data set. What do I do?

Common approaches:

- ▶ Fit a linear model and look at p-values
- ▶ Fit a non-parametric model and get predictions
- ▶ Calculate summary statistics and form a story around the answers



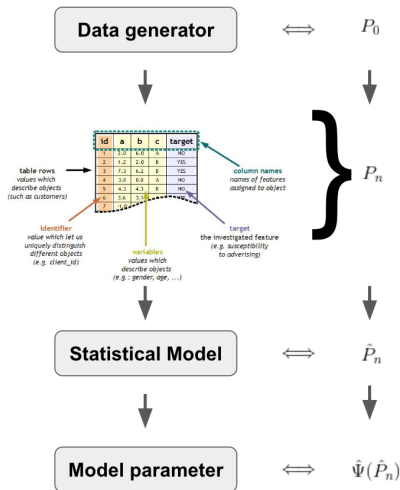
Common scenario: I have a data set. What do I do?

Common approaches:

- ▶ Fit a linear model and look at p-values
- ▶ Fit a non-parametric model and get predictions
- ▶ Calculate summary statistics and form a story around the answers

Can result in significantly different answers!

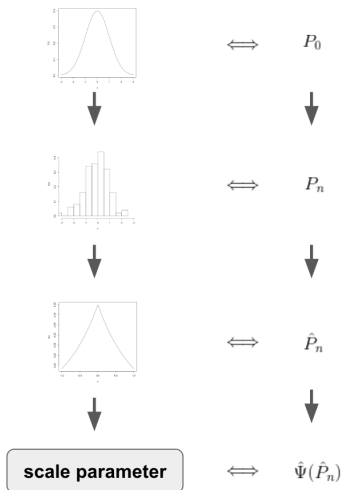
Empirical vs true distributions



Ideally, we want $\Psi(P_0)$.



Example: P_0 is Gaussian, while \hat{P}_n is Laplace





The financial crisis of 2007-2008.

- ▶ Let $O = (X_1, X_2, \dots, X_p, Y)$ be our data
 - ▶ e.g. X 's are measures of financial & real estate market. Y is CDO value.
- ▶ Generally want to estimate $\mathbb{E}_0[Y|X_1, X_2, \dots, X_p]$
- ▶ Two big issues:
 1. \hat{P}_n wasn't representative of P_0
 2. Most statistical models assume $O_i \stackrel{iid}{\sim} P_0$

Result: $\Psi(P_0)$ was no where close to $\hat{\Psi}(\hat{P}_n)$



Supervised: We have a clearly defined outcome of interest.

- ▶ Pro: More clearly defined
- ▶ Con: May take more resources to gather

Unsupervised: We don't have a clearly defined outcome of interest.

- ▶ Pro: Typically readily available
- ▶ Con: May be a more abstract problem



Typically start with a data matrix, e.g.

id	weight	height	# children	education level	gender	profession	my life story
1							
2							
3							
.							

**n.b. The data may also be unstructured (e.g. text, pixels, etc).*



id	weight	height	# children	education level	gender	profession	my life story
1							
2							
3							
.							

Two primary categories:

1. Quantitative:

- ▶ Numerical
- ▶ Ordinal

2. Qualitative:

- ▶ Categorical
- ▶ Free form



Goal: Learn the overall structure of our data. e.g.

- ▶ Clustering: learn meaningful groupings of the data
 - ▶ e.g. k-means, Expectation Maximization, etc.
- ▶ Correlation: learn meaningful relationships between variables or units
 - ▶ e.g. concordance, Pearson's, etc.
- ▶ Dimension reduction: learn compression of data for downstream tasks
 - ▶ e.g. PCA, LDA, auto-encoding, etc.

We learn these using our data.



Typically start with a data matrix **with an outcome**, e.g.

id	weight	height	# children	education level	gender	profession	my life story	outcome
1								
2								
3								
.								

Outcome can be quantitative or qualitative.



Goal: We learn a mapping from input variables to output variables.

- ▶ If quantitative, then we refer to this as Regression
 - ▶ e.g. $\mathbb{E}_0[Y|X_1, X_2, \dots, X_p]$
- ▶ If qualitative, then we refer to this as Classification
 - ▶ e.g. $\mathbb{P}_0[Y = y|X_1, X_2, \dots, X_p]$

In both cases, we're interested in learning some function,

$$f_0(X_1, X_2, \dots, X_p) \tag{1}$$

We estimate f_0 using our data.



Motivation: Why learn f_0 ?

Prediction

- ▶ Useful when we can readily get X_1, X_2, \dots, X_p , but not Y .
- ▶ Allows us to predict what Y likely is.
- ▶ **Example:** Predict stock prices next month using data from last year.

Inference

- ▶ Allows us to understand how differences in X_1, X_2, \dots, X_p might affect Y .
- ▶ **Example:** What is the influence of genetic variations on the incidence of heart disease.



How do we estimate f_0 ? Two classes of methods:

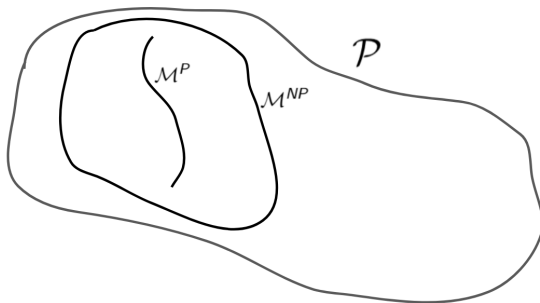
- ▶ **Parametric models:** We assume that f_0 takes a specific form. For example, a linear form:

$$f_0(X_1, X_2, \dots, X_p) = X_1\beta_1 + X_2\beta_2 + \dots + X_p\beta_p \quad (2)$$

- ▶ **Non-parametric models:** We don't make any assumptions on the form of f_0 , but we restrict how “wiggly” or “rough” the function can be. For example, using loess.



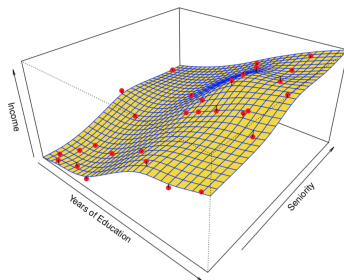
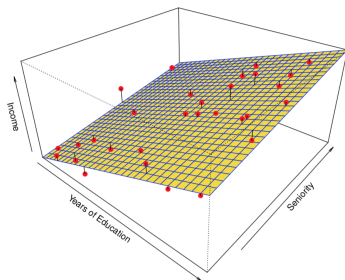
Visualization



Non-parametric models tend to be larger than parametric models.

Recall: A statistical model is simply a set of probability distributions that you allow your data to follow.

Parametric vs non-parametric fit



Non-parametric models tend to be more flexible.



Question: Why don't we just always use non-parametric models?



Question: Why don't we just always use non-parametric models?

1. **Interpretability:** parametric models are simpler to interpret



Question: Why don't we just always use non-parametric models?

1. **Interpretability:** parametric models are simpler to interpret
2. **Convenience:** less computation, more reproducibility, better behavior



Question: Why don't we just always use non-parametric models?

1. **Interpretability:** parametric models are simpler to interpret
2. **Convenience:** less computation, more reproducibility, better behavior
3. **Overfitting:** non-parametric models tend to overfit (aka. high variance)



Training data: $(x_i, y_i) : i = 1, 2, \dots, n$

Goal: Estimate f_0 with our data, resulting in \hat{f}_n

Typically: we get \hat{f}_n by minimizing a **prediction error**

- ▶ Assumes $(x_i, y_i) \stackrel{iid}{\sim} P_0$



Training data: $(x_i, y_i) : i = 1, 2, \dots, n$

Goal: Estimate f_0 with our data, resulting in \hat{f}_n

Typically: we get \hat{f}_n by minimizing a **prediction error**

- ▶ Assumes $(x_i, y_i) \stackrel{iid}{\sim} P_0$

Standard prediction error functions:

- ▶ **Classification:** Cross-entropy

$$CE(\hat{f}_n) = \mathbb{E}_0[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (3)$$



Training data: $(x_i, y_i) : i = 1, 2, \dots, n$

Goal: Estimate f_0 with our data, resulting in \hat{f}_n

Typically: we get \hat{f}_n by minimizing a **prediction error**

- ▶ Assumes $(x_i, y_i) \stackrel{iid}{\sim} P_0$

Standard prediction error functions:

- ▶ **Classification:** Cross-entropy

$$CE(\hat{f}_n) = \mathbb{E}_0[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (3)$$

- ▶ **Regression:** Mean squared error

$$MSE(\hat{f}_n) = \mathbb{E}_0[y_i - \hat{f}_n(x_i)]^2 \quad (4)$$



Cross entropy:

$$CE(\hat{f}_n) = \mathbb{E}_0[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (5)$$

n.b. We can't directly calculate this, since P_0 is unknown.



Cross entropy:

$$CE(\hat{f}_n) = \mathbb{E}_0[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (5)$$

n.b. We can't directly calculate this, since P_0 is unknown.

But:

We do have P_n , i.e. our training data $(x_i, y_i) : i = 1, 2, \dots, n$.



Cross entropy:

$$CE(\hat{f}_n) = \mathbb{E}_0[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (5)$$

n.b. We can't directly calculate this, since P_0 is unknown.

But:

We do have P_n , i.e. our training data $(x_i, y_i) : i = 1, 2, \dots, n$.

Estimating cross entropy

$$\widehat{CE}(\hat{f}_n) = \mathbb{E}_n[-\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i)] \quad (6)$$

$$= \frac{1}{n} \sum_{i=1}^n -\mathbf{y}_i \cdot \log \hat{\mathbf{f}}_n(x_i) \quad (7)$$



Similarly:

We estimate the **mean squared error** using our data.

Estimating mean squared error

$$\widehat{MSE}(\hat{f}_n) = \mathbb{E}_n[y_i - \hat{f}_n(x_i)]^2 \quad (8)$$

$$= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{f}_n(x_i))^2 \quad (9)$$



There are two common problems with prediction errors:

1. A high prediction error could mean underfitting.
 - ▶ e.g. You could have the wrong functional form

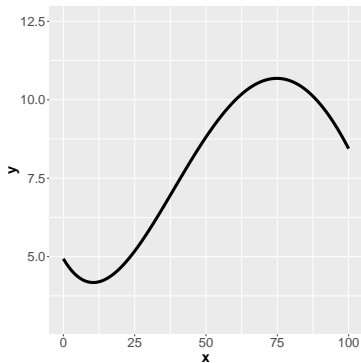


There are two common problems with prediction errors:

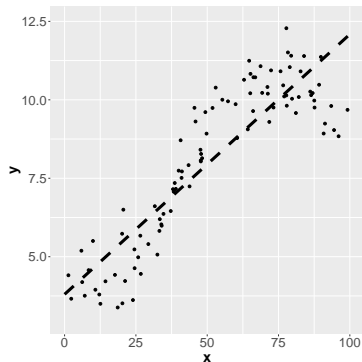
1. A high prediction error could mean underfitting.
 - ▶ e.g. You could have the wrong functional form
2. A low prediction error could mean overfitting.
 - ▶ e.g. You made your model too flexible



1. A high prediction error could mean underfitting.



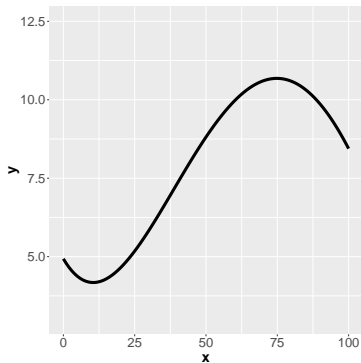
True function f_0 .



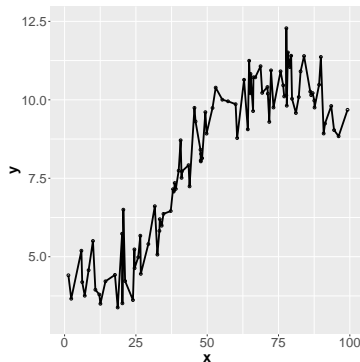
Observed data and estimated function \hat{f}_n .



2. A low prediction error could mean overfitting.



True function, f_0 .



Observed data and estimated function \hat{f}_n .



How to tell if we've under/overfit:

- Evaluate on data not used in training (i.e. from your test set).

Given our test data $(x_i', y_i') : i = 1, 2, \dots, m$, we can calculate a more accurate prediction error, e.g.:

$$\widehat{MSE}(\hat{f}_n) = \mathbb{E}_n^{test}[y_i' - \hat{f}_n(x_i')]^2 \quad (10)$$

$$= \frac{1}{m} \sum_{i=1}^m (y_i' - \hat{f}_n(x_i'))^2 \quad (11)$$



How to tell if we've under/overfit:

► So, now we have two prediction error estimates, e.g.:

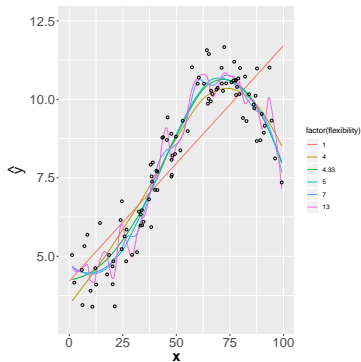
1. $\widehat{MSE}^{train}(\hat{f}_n)$ from our training data

2. $\widehat{MSE}^{test}(\hat{f}_n)$ from our test data

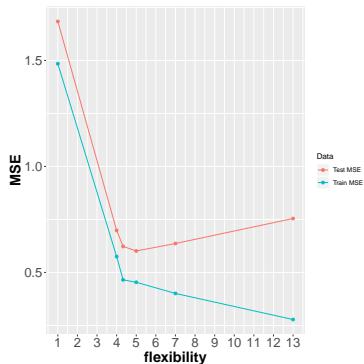
If $\widehat{MSE}^{train}(\hat{f}_n) \ll \widehat{MSE}^{test}(\hat{f}_n)$, then we've likely overfit on our training data.



How to tell if we've under/overfit:



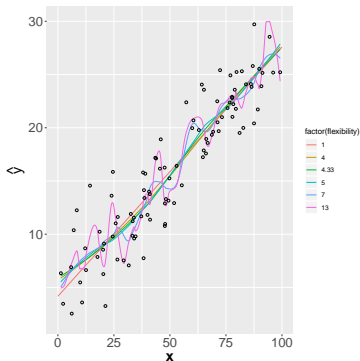
Estimates \hat{f}_n of f_0 .



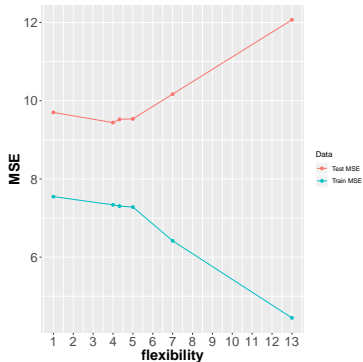
\widehat{MSE} for each \hat{f}_n .



How to tell if we've under/overfit (with an almost linear f_0):



Estimates \hat{f}_n of f_0 .

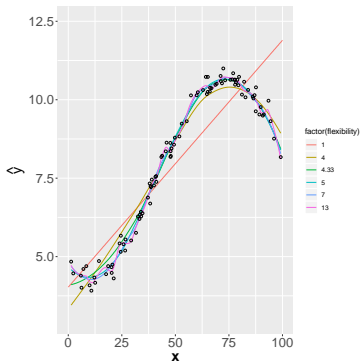


\widehat{MSE} for each \hat{f}_n .

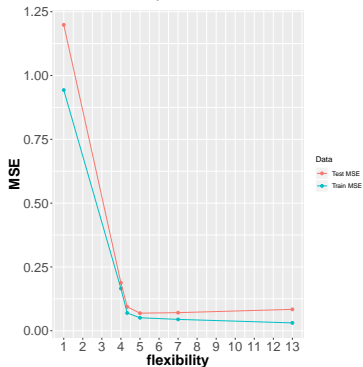
Low flexibility models work well.



How to tell if we've under/overfit (with low noise):



Estimates \hat{f}_n of f_0 .



\widehat{MSE} for each \hat{f}_n .

High flexibility models work well.



Let x_0 be a fixed point, $y_0 = f_0(x_0) + \epsilon$, and \hat{f}_n be an estimate of f_0 from $(x_i, y_i) : i = 1, 2, \dots, n$.

The MSE at x_0 can be decomposed as

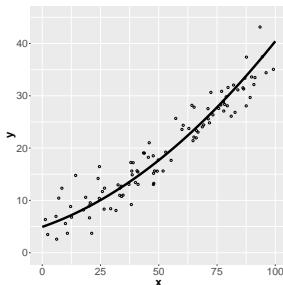
$$\begin{aligned}MSE(x_0) &= \mathbb{E}_0[y_0 - \hat{f}_n(x_0)]^2 & (12) \\&= \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0) & (13)\end{aligned}$$



$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$

$\text{Var}(\epsilon_0)$

Noise from the data distribution, i.e. irreducible error.



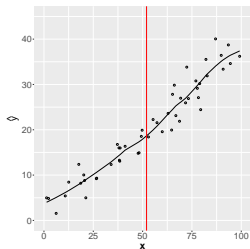
True function, f_0 and observed data.



$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$

$\text{Var}(\hat{f}_n(x_0))$

The variance of $\hat{f}_n(x_0)$ (i.e. the estimate of y).
How much the estimate \hat{f}_n at x_0 changes with new data.

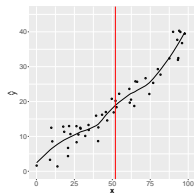
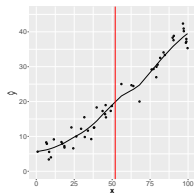
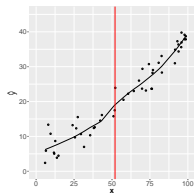
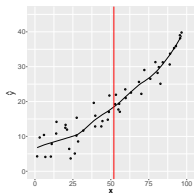
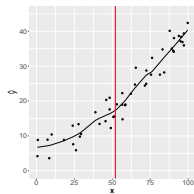
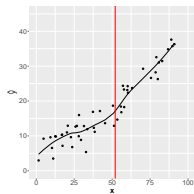
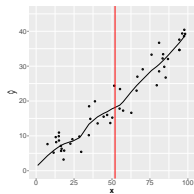
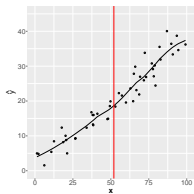


Observed data and estimate \hat{f}_n .

Bias variance decomposition



$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$



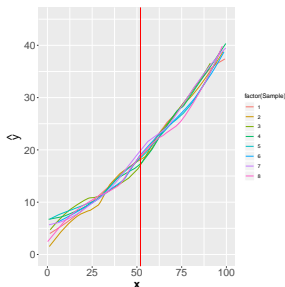
Estimates for 8 different data samples ($n = 50$).



$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$

$$\text{Var}(\hat{f}_n(x_0))$$

The variance of $\hat{f}_n(x_0)$ (i.e. the estimate of y).
How much the estimate \hat{f}_n at x_0 changes with new data.

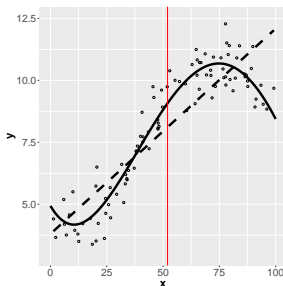




$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$

$$\text{Bias}(\hat{f}_n(x_0))^2$$

The square of the expected difference, $\mathbb{E}^2[\hat{f}_n(x_0) - f_0(x_0)]$.
How far the average prediction \hat{f}_n is from f_0 at x_0 .





$$MSE(x_0) = \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0)$$

Implications:

- ▶ The MSE is always non-negative.
- ▶ Each element on the right side is always non-negative.
- ▶ Consequently, lowering one element (beyond some point) typically increases another.

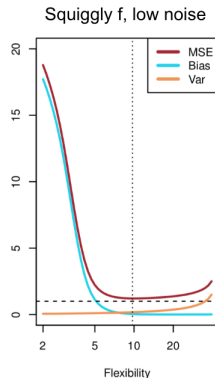
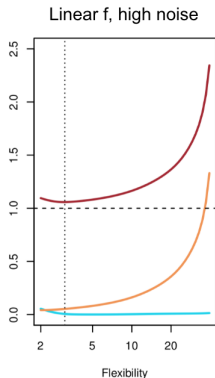
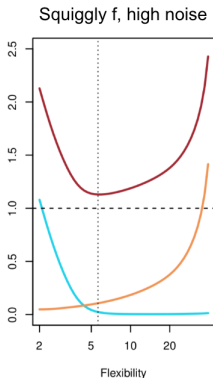
Bias variance trade-off

More flexibility \iff Higher variance \iff Lower bias



Bias variance trade-off

More flexibility \iff Higher variance \iff Lower bias





In classification, the output takes values in a discrete set (c.f. continuous values in regression).

Example

If we're trying to predict the brand of a car (based on input features), the function f_0 outputs the (conditional) probabilities of each car brand (e.g. Ford, Toyota, Mercedes, etc.), e.g.

$$\mathbb{P}_0[Y = y | X_1, X_2, \dots, X_p] : y \in \{Ford, Toyota, Mercedes, etc.\} \quad (14)$$



Regression: $f_0 = \mathbb{E}_0[Y|X_1, X_2, \dots, X_p]$

- ▶ A scalar value, i.e. $f_0 \in \mathbb{R}$
- ▶ \hat{f}_n therefore gives us estimates of y

Classification: $f_0 = \mathbb{P}_0[Y = y|X_1, X_2, \dots, X_p]$

- ▶ A vectored value, i.e.
 $f_0 = [p_1, p_2, \dots, p_K] : p_j \in [0, 1], \sum_K p_j = 1$
- ▶ n.b. In a binary setting this simplifies to a scalar, i.e.
 $f_0 = p_1 : p_1 = \mathbb{P}_0[Y = 1|X_1, X_2, \dots, X_p] \in [0, 1]$
- ▶ \hat{f}_n therefore gives us predictions of each class



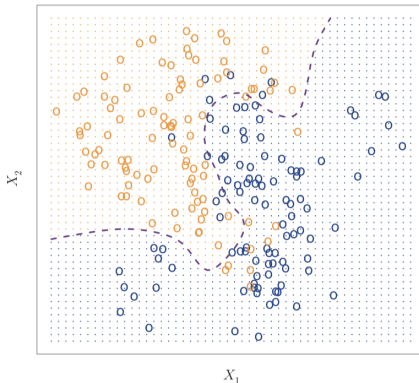
- ▶ f_0 gives us a probability of the observation belonging to each class.
- ▶ To select a class, we can just pick the element in $f_0 = [p_1, p_2, \dots, p_K]$ that's the largest
 - ▶ Called the Bayes Classifier
- ▶ As a classifier, produces the lowest error rate

Bayes error rate

$$1 - \mathbb{E}_0 \left[\max_y \mathbb{P}_0[Y = y | X_1, X_2, \dots, X_p] \right] \quad (15)$$

Analogous to the **irreducible error** described previously

Example: Classifying in 2 classes with 2 features.

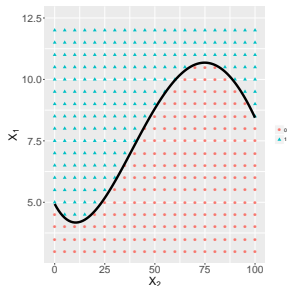


The Bayes error rate is 0.1304.

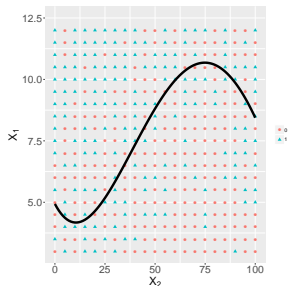


Note: $\mathcal{C}(\mathbf{x}) = \arg \max_y f_0(y)$ may seem easier to estimate

- Can still be hard, depending on the distribution f_0 , e.g.



Bayes error = 0.0



Bayes error = 0.3



[1] ISL. Chapters 1-2.

[2] ESL. Chapters 1-2.