02 - Basic Machine Learning Concepts

ml4econ, HUJI 2025

Itamar Caspi

March 23, 2025 (updated: 2025-03-22)

Replicating this Presentation

R packages used to produce this presentation:

```
library(tidyverse)    # for data manipulation, exploration, and visualization using ggplot2, dplyr, etc.
library(tidymodels)   # for machine learning modeling using a consistent framework
library(RefManageR)   # for managing and formatting bibliographic references
library(truncnorm)    # for generating random numbers from truncated normal distributions
library(dagitty)      # for creating and analyzing directed acyclic graphs (DAGs) for causal inference
library(knitr)        # for dynamic report generation and embedding graphics
```

First Things First: "Big Data"

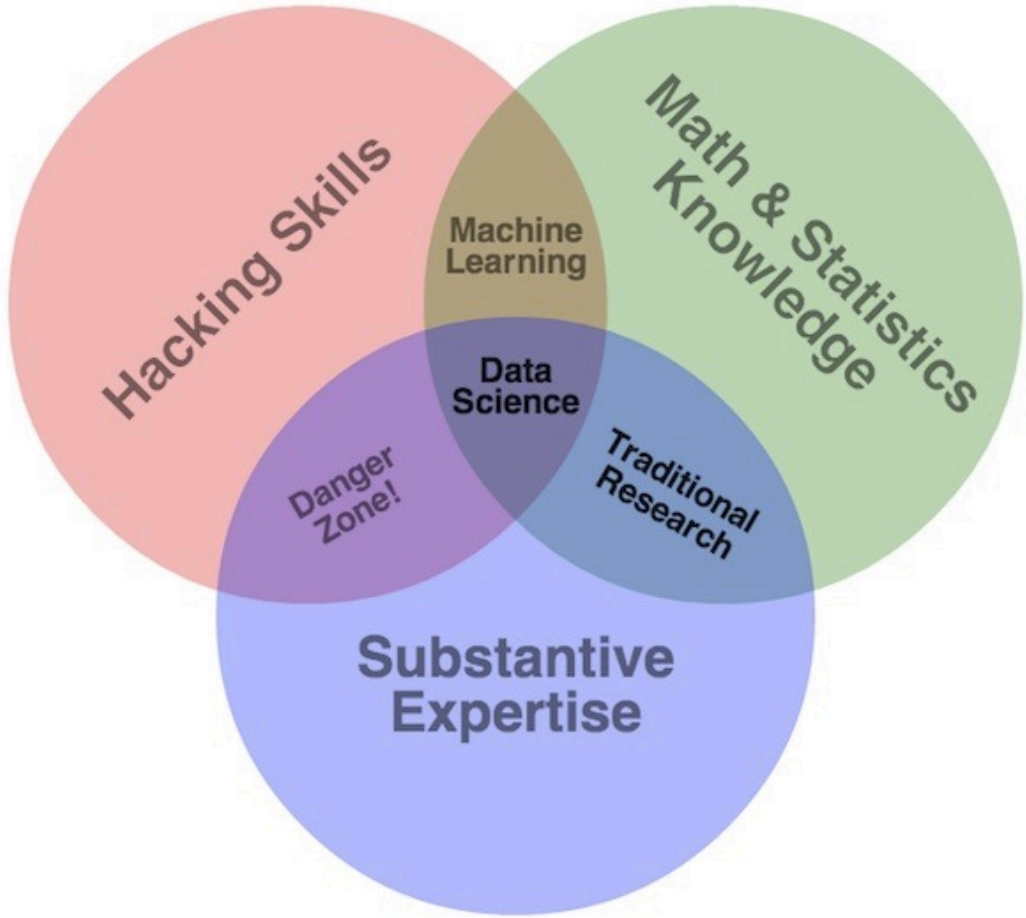
> *"A billion years ago modern homo sapiens emerged. A billion minutes ago, Christianity began. A billion seconds ago, the IBM PC was released. A billion Google searches ago ... was this morning."*
>
> Hal Varian (2013)

Today, we are at **~9 billion** Google searches per day!

The 4 Vs of big data:

- Volume - Scale of data.
- Velocity - Analysis of streaming data.
- Variety - Different forms of data.
- Veracity - Uncertainty of data.

"Data Science"



[*] Hacking ≈ coding

Outline

1. What is ML?
2. The problem of overfitting
3. Too complext? Regularize!
4. ML and Econometrics

What is ML?

So, what is ML?

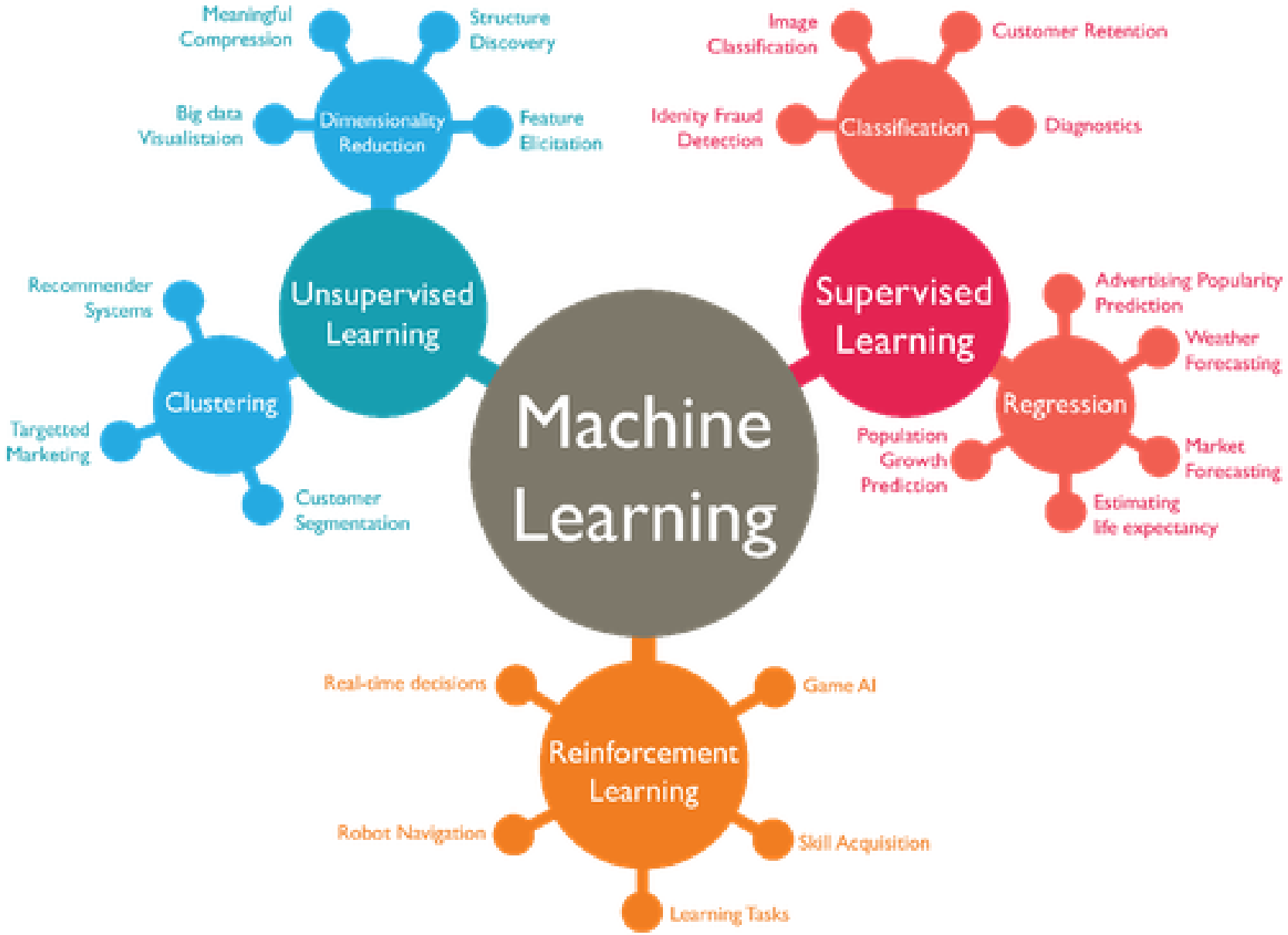
A concise definition by Athey (2018):

> "...[M]achine learning is a field that develops algorithms designed to be applied to datasets, with the main areas of focus being prediction (regression), classification, and clustering or grouping tasks."

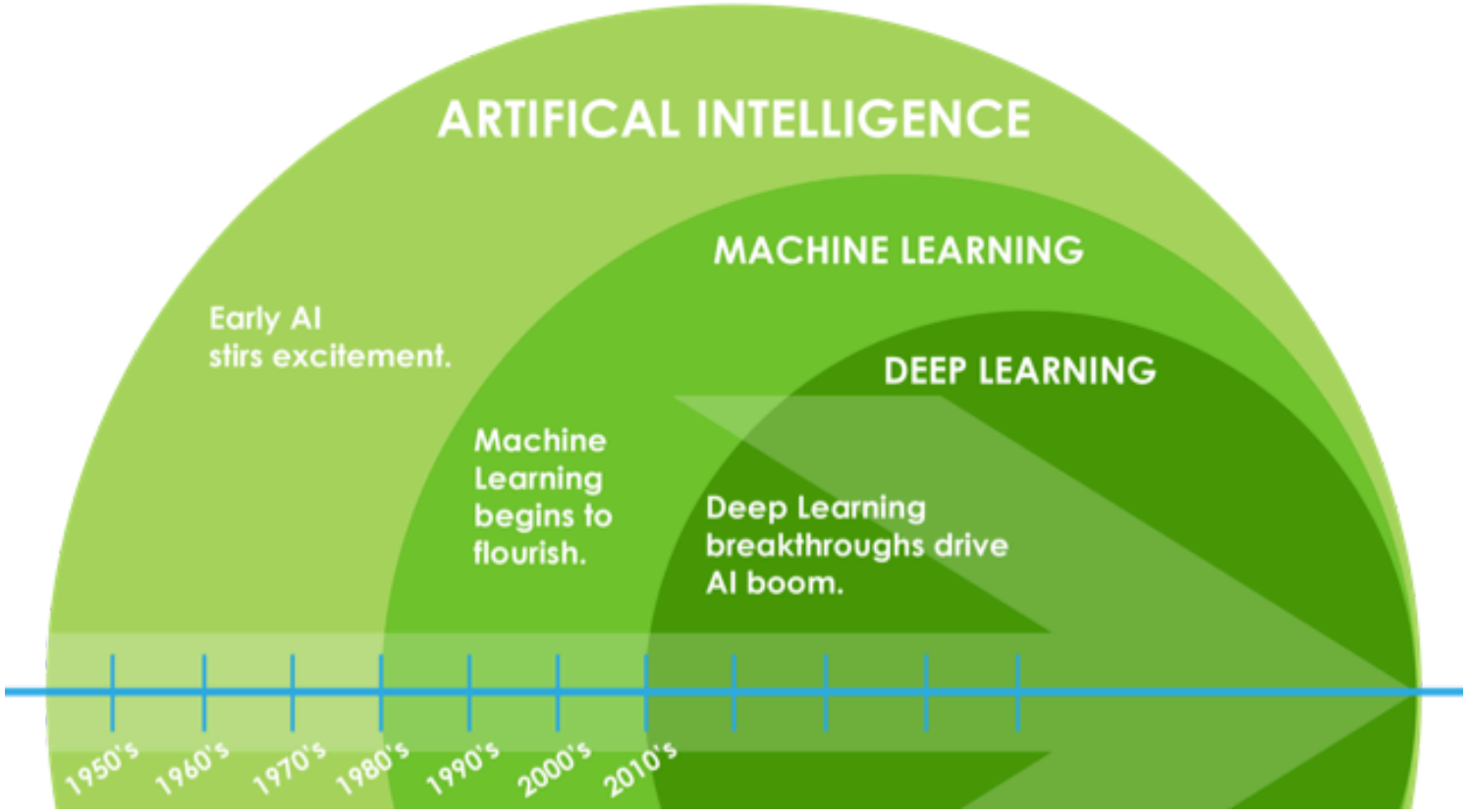
Specifically, there are three broad classifications of ML problems:

- supervised learning.
- unsupervised learning.
- reinforcement learning.

> Most of the hype you hear about in recent years relates to supervised learning, and in particular, deep learning.


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big data Visualisation
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Identity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Population Growth Prediction
Market Forecasting
Estimating life expectancy
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

An Aside: ML and Artificial Intelligence (AI)



Unsupervised Learning

In *unsupervised* learning, the goal is to divide high-dimensional data into clusters that are **similar** in their set of features (X).

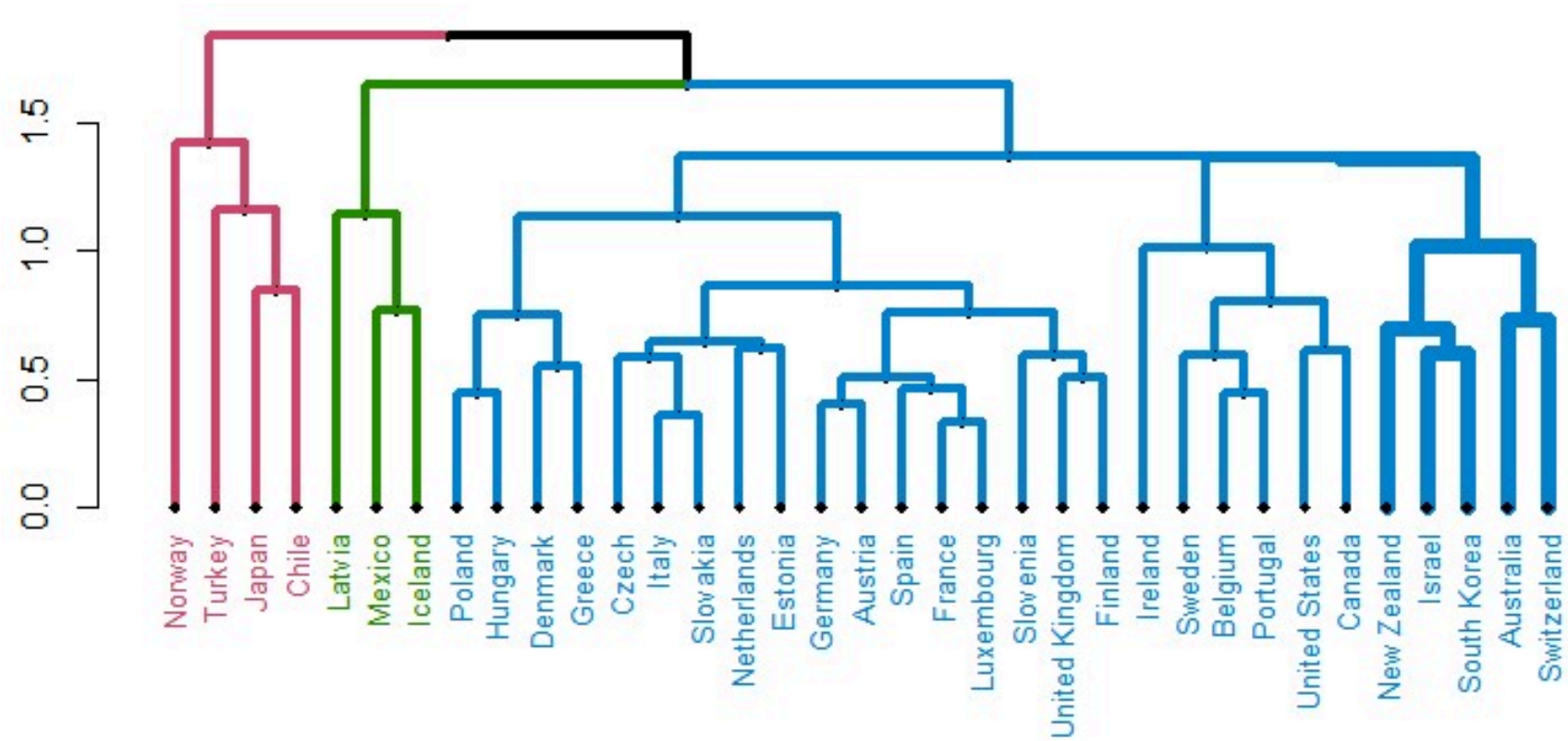
Examples of algorithms:

- principal component analysis
- k-means clustering
- Latent Dirichlet Allocation (LDA)

Applications:

- image recognition
- cluster analysis
- topic modelling

Example: Clustering OECD Inflation Rates



Source: Baudot-Trajtenberg and Caspi (2018).

Reinforcement Learning (RL)

A definition by Sutton and Barto (2018):

> *"Reinforcement learning is learning what to do—how to map situations to actions —so as to maximize a numerical reward signal. The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them."*

Prominent examples:

- Game AI (e.g., chess, AlphaGo).
- Robotics (e.g., autonomous cars).
- LLMs (e.g., ChatGPT)

Supervised Learning

Consider the following data generating process (DGP):

$$Y = f(\boldsymbol{X}) + \epsilon$$

where Y is the outcome variable, $\boldsymbol{X}$ is a $1 \times p$-dimensional vector of predictor variables (features), $f(\cdot)$ is the unknown true relationship we aim to estimate, and ϵ is the irreducible error.

- **Training set** ("in-sample"): $\{(x_i, y_i)\}_{i=1}^{n}$ - used to fit the model
- **Test set** ("out-of-sample"): $\{(x_i, y_i)\}_{i=n+1}^{m}$ - used to evaluate model performance

Key assumptions: (1) independent observations; (2) consistent data generating process across both training and test sets.

The Goal of Supervised Learning

The objective is to learn a function $\hat{f}(X)$ from a labeled training set (where both X and Y are known) that accurately predicts outcomes on unseen data (where only X is known).

Specifically, we aim to:

- Minimize the generalization error (error on unseen data)
- Create a model that captures the true underlying relationship $f(X)$, not just memorize training examples

Example: Spam Detection

ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though
spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.50 to rcv
ham	Even my brother is not like to speak with me. They treat me like aids patent.
ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune
spam	WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.
spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030

In this case:

- $Y \in \{spam, ham\}$ (a binary classification problem)
- X represents the email's features (text, sender information, timestamp, location, etc.)
- The model learns patterns from labeled emails to identify spam in new, unseen emails

The challenge lies in identifying features that consistently distinguish spam across both seen and unseen emails, while avoiding overfitting to peculiarities in the training data.

Traditional vs. Modern Approach to Supervised Learning

Machine Learning and Prediction in Economics and Finance

Traditional vs. Modern Approach to Supervised Learning

Traditional Approach: Expert-Defined Rules

- Human experts manually create rules based on domain knowledge
  - Example: For spam detection, create dictionaries of suspicious terms
  - Rules: "If email contains 'Nigerian Prince' AND unknown sender, classify as spam"
- Limitations: Labor-intensive, difficult to adapt, struggles with complex cases

Modern Approach: Data-Driven Learning

- System automatically discovers patterns from labeled examples
  - Underlying rules emerge from the data itself
  - Example: Feed algorithms thousands of labeled emails to identify patterns
- Advantages: Discovers non-obvious relationships, adapts to new patterns, scales to handle complexity beyond human comprehension

Real-World Applications of Machine Learning

Task	Outcome (Y)	Features (X)	Type
Credit scoring	Probability of default	Loan history, payment records, income	Regression
Fraud detection	Fraud / no fraud	Transaction patterns, timing, amounts	Classification
Sentiment analysis	Positive / negative / neutral	Text, review content, linguistic features	Classification
Image classification	Object categories	Pixel values, image features	Classification
Overdraft prediction	Yes / no	Account history, transaction patterns	Classification
Customer churn	Will churn / won't churn	Usage patterns, demographics, history	Classification
Disease diagnosis	Disease presence/type	Medical images, patient data, symptoms	Classification

Example: Anticipatory Package Shipping

Different applications involve different types of data structures, problem formulations, and evaluation metrics, each requiring specialized modeling approaches.

> Amazon's "Anticipatory Package Shipping" patent (December 2013): Imagine Amazon's algorithms reaching such levels of accuracy, casing it to change its business model from shopping-then-shipping to shipping-then-shopping!

Supervised Learning Algorithms: A Historical Perspective

Many core machine learning algorithms have surprisingly deep historical roots:

- Linear and logistic regression (1805, 1958)
  - The foundations of statistical modeling predating modern computing
- K-Nearest neighbors (1967)
  - An intuitive approach formalized in the era of early computing
- Decision trees (CART, ID3, 1984)
  - Emerged as computing power allowed more complex rule-based systems
- Neural networks (conceptualized 1940s, major advances 1986, 1990s)
  - Theoretical foundations existed decades before practical implementation
- Support vector machines (1990s)
  - Computational optimization applied to classification problems
- Ensemble methods (bagging 1996, Random Forests 2001, boosting 1990s)
  - Combining multiple models for improved performance

Most foundational algorithms were developed well before the recent "AI revolution," suggesting other factors must explain the current surge in ML capabilities.

So, Why Now?

- ML methods have always been data-hungry and computationally expensive. The recent boom is driven by:
  - Exponential growth in available data (internet, IoT, digitization of services)
  - Dramatic advances in computing power (GPUs, cloud infrastructure)
  - Improved software and frameworks that democratize implementation

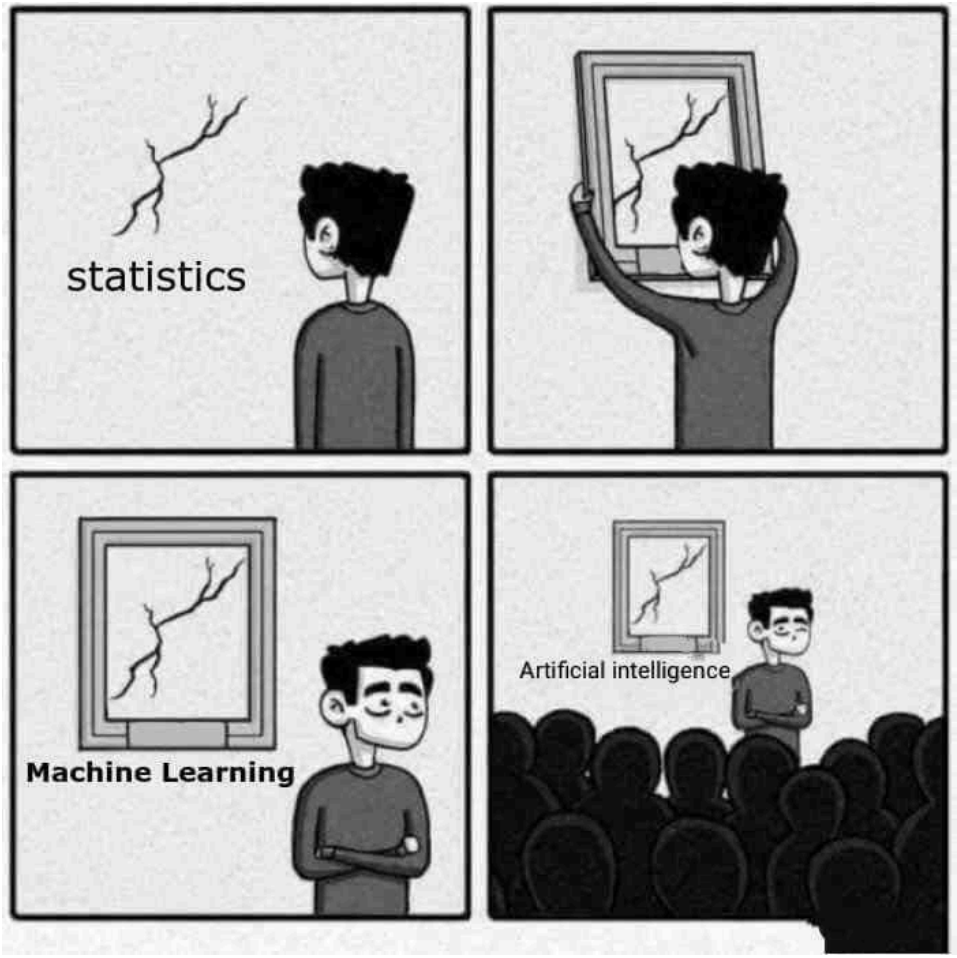
$$\textbf{big data} + \textbf{computational advancements} = \textbf{the rise of ML}$$

- As Diebold notes:

> *"[S]upervised learning [...] may involve high dimensions, non-linearities, binary variables, etc., but at the end of the day it's still just regression."*
>
> — **Francis X. Diebold**

The core mathematical principles haven't changed significantly—what's changed is our ability to implement them at unprecedented scales.

Wait, is ML Just Glorified Statistics?



The "two cultures" (Breiman, 2001):

Statistics: Assumes a data generating process and focuses on parameter estimation, inference, and understanding underlying mechanisms with strong emphasis on model assumptions and interpretability.

Machine learning: Treats the data mechanism as unknown and complex, focusing primarily on prediction accuracy and generalization performance with less concern for interpretable structures or theoretical guarantees.

Overfitting

Prediction Accuracy

To precisely define what makes a prediction "good," we need a formal way to measure performance:

Loss Functions

- Let (x^0, y^0) denote a single observation from the (unseen) **test set**
- A **loss function** $L(y^0, \hat{y}^0)$ quantifies the prediction error when $\hat{y}^0 = \hat{f}(x^0)$
- The model $\hat{f}$ is estimated using only the **training** data

Common Loss Functions

For regression:

- Squared error (SE): $L(y^0, \hat{y}^0) = (y^0 - \hat{y}^0)^2$
  - Penalizes large errors more heavily; sensitive to outliers
- Absolute error (AE): $L(y^0, \hat{y}^0) = |y^0 - \hat{y}^0|$
  - More robust to outliers; constant penalty for increasing error

For classification:

- 0-1 loss: $L(y^0, \hat{y}^0) = I(y^0 \neq \hat{y}^0)$ (misclassification rate)
- Cross-entropy loss: $L(y^0, \hat{y}^0) = -\sum_c y_c^0 \log(\hat{y}_c^0)$ (for probabilities)

The choice of loss function should reflect the specific costs of different types of prediction errors in your application.

Intuition Behind the Bias-Variance Trade-off

Imagine you are a teaching assistant grading exams. You've just graded a student's first exam and got a score of 95. Now you need to predict their score on the second exam. Two possible approaches:

1. **Use only the first score (unbiased but variable):**
    - Using 95 as your prediction is unbiased (doesn't systematically under or overestimate)
    - But single exam scores vary greatly due to luck, daily preparation, etc.
    - High variance means your prediction could be far off
2. **Shrink toward the class average (biased but less variable):**
    - If the class average is 75, you might predict (75 + 95)/2 = 85
    - This introduces bias (systematically predicting lower than 95)
    - But reduces variance since you're not fully relying on a single, noisy observation

Key insight: By accepting a small amount of bias, we can significantly reduce variance, often resulting in better overall predictions.

The Bias-Variance Decomposition

Under a **squared error loss function**, an optimal predictive model is one that minimizes the *expected* squared prediction error.

It can be shown that if the true model is $Y = f(X) + \epsilon$, then

$$\begin{aligned}
\mathbb{E}\left[\mathrm{SE}^0\right] &= \mathbb{E}\left[(y^0 - \hat{f}\,(x^0))^2\right] \\
&= \underbrace{\Big(\mathbb{E}(\hat{f}\,(x^0)) - f(x^0)\Big)^2}_{\text{bias}^2} + \underbrace{\mathbb{E}\Big[\hat{f}\,(x^0) - \mathbb{E}(\hat{f}\,(x^0))\Big]^2}_{\text{variance}} + \underbrace{\mathbb{E}\big[y^0 - f(x^0)\big]^2}_{\text{irreducible error}} \\
&= \underbrace{\mathrm{Bias}^2 + \mathbb{V}[\hat{f}\,(x^0)]}_{\text{reducible error}} + \sigma_\epsilon^2
\end{aligned}$$

where the expectation is over the training set *and* (x^0, y^0).

Understanding the Expectation Operator in Bias-Variance

The expectation $\mathbb{E}[\mathbf{SE}^0]$ integrates over two sources of randomness:

1. **Randomness in the training data**: Different training sets $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ lead to different fitted models $\hat{f}_{\mathcal{D}}(x)$

2. **Randomness in the test point**: The specific test point (x^0, y^0) we evaluate on

This gives us:

$$\mathbb{E}[\mathbf{SE}^0] = \mathbb{E}_{\mathcal{D},(x^0,y^0)}\left[(y^0 - \hat{f}_{\mathcal{D}}(x^0))^2\right]$$

Exam Grade Prediction Simulation

Let's Draw 1000 grade duplets from the following truncated normal distribution

$$g_i \sim truncN(\mu = 75, \sigma = 15, a = 0, b = 100), \quad i = 1, 2$$

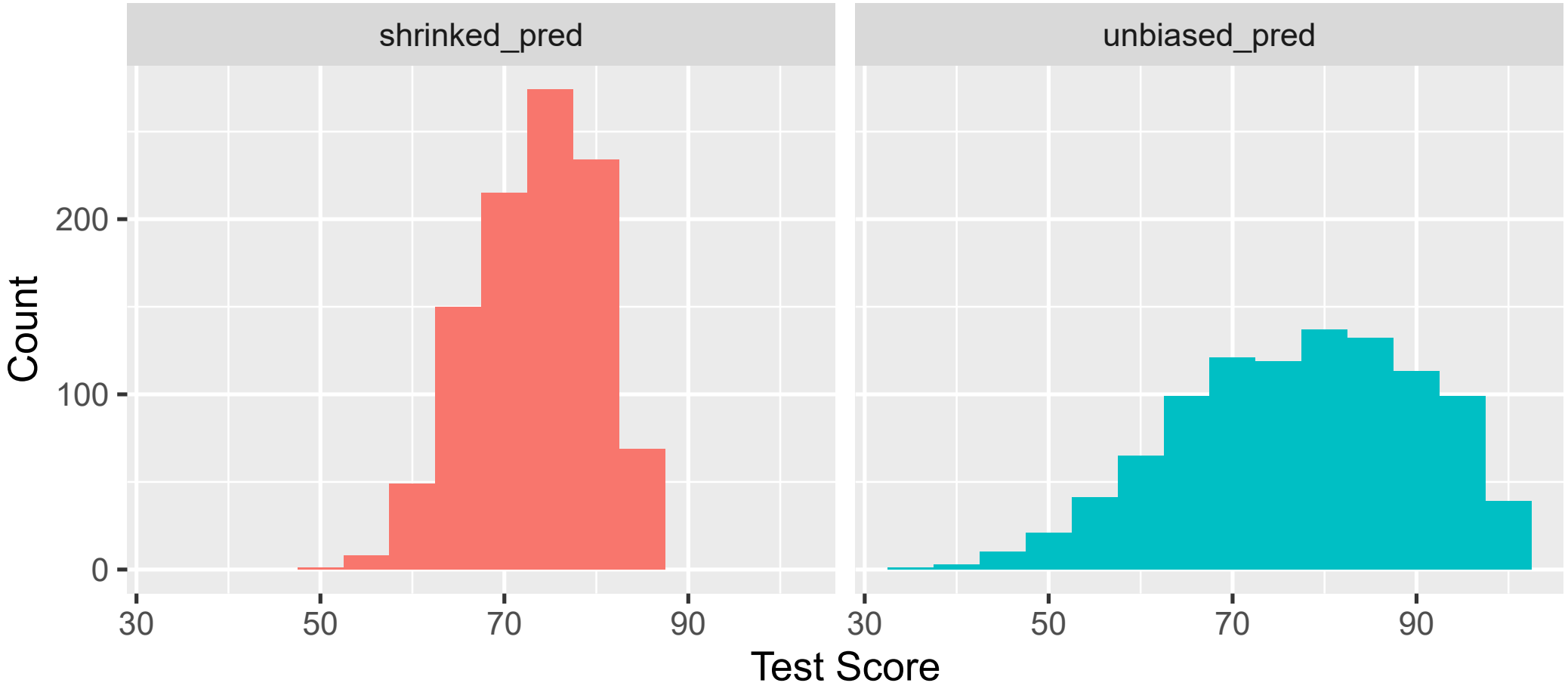
Next, calculate two types of predictions

- `unbiased_pred` is the first exam's grade.
- `shrinked_pred` is an average of the previous grade and a *prior* mean grade of 70.

Here a small sample from our simulated table:

attempt	grade1	grade2	unbiased_pred	shrinked_pred
752	76	84	76	73.0
705	83	72	83	76.5
180	84	59	84	77.0
498	99	73	99	84.5
929	77	94	77	73.5

The Distribution of Predictions



The MSE of Grade Predictions

unbiased_MSE	shrinked_MSE
312.102	199.4203

Hence, the shrunk prediction turns out to be better (in the sense of MSE) then the unbiased one!

QUESTION: Is this a general result? Why?

Illustrating Bias-Variance: The Consumption Function

- Let's see the bias-variance tradeoff in action with a practical economic example
- Consider a classic consumption function relationship:

$$consumption_i = f(income_i) + \varepsilon_i$$

- We'll explore three modeling approaches with varying complexity:

  1. **Underfitting**: Too simple (high bias, low variance)
  2. **Overfitting**: Too complex (low bias, high variance)
  3. **"Just right"**: Optimal complexity balance

- This will show how different economists might approach the same data differently

The "Experiment"

Our Experiment:

- Three economists (A, B, C) each receive a different subset of consumption data
- All data comes from the same "true" **linear** model
- Each economist must predict consumption at income = 70 (outside their observed range)
- We'll compare their approaches with different model complexities

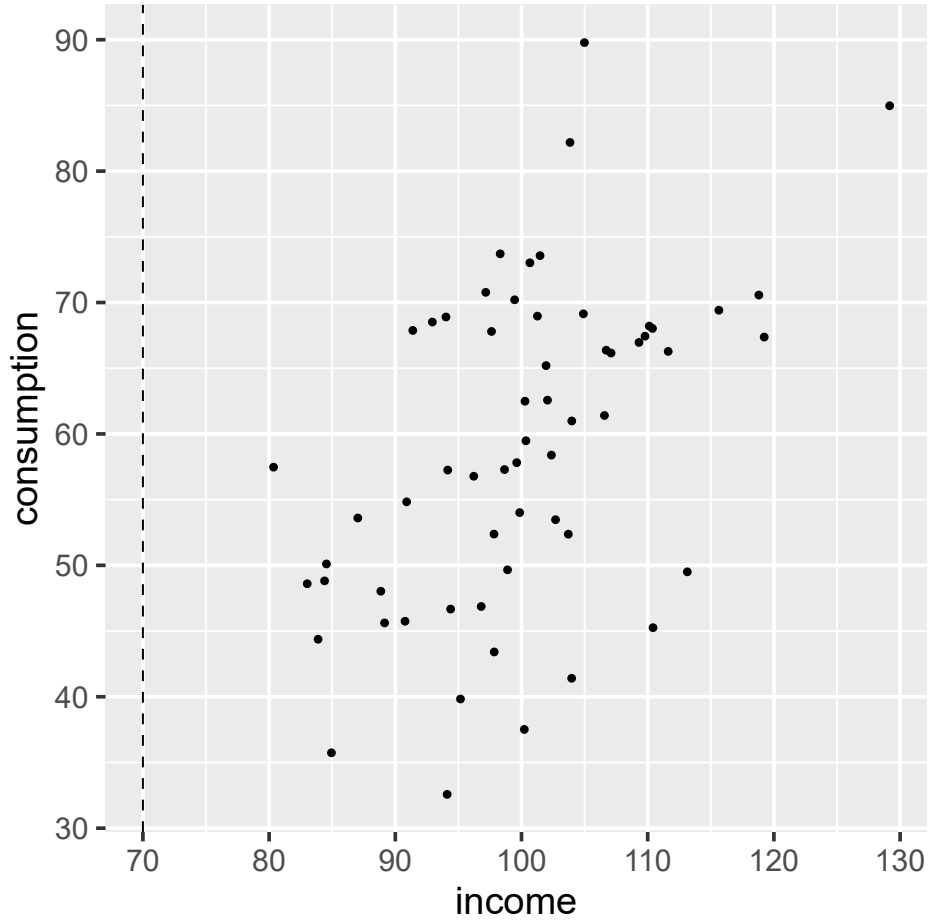
Step 1: Simulate the DGP:

```
set.seed(1505) # for replicating the simulation

df <- crossing(economist = c("A", "B", "C"),
               obs = 1:20) %>%
  mutate(economist = as.factor(economist)) %>%
  mutate(income = rnorm(n(), mean = 100, sd = 10)) %>%
  mutate(consumption = 10 + 0.5 * income + rnorm(n(), sd = 10))
```

Scatterplot of the Data

```
df %>%
  ggplot(aes(y = consumption,
             x = income)) +
  geom_point() +
  geom_vline(xintercept = 70, linetype = "dashed")
```

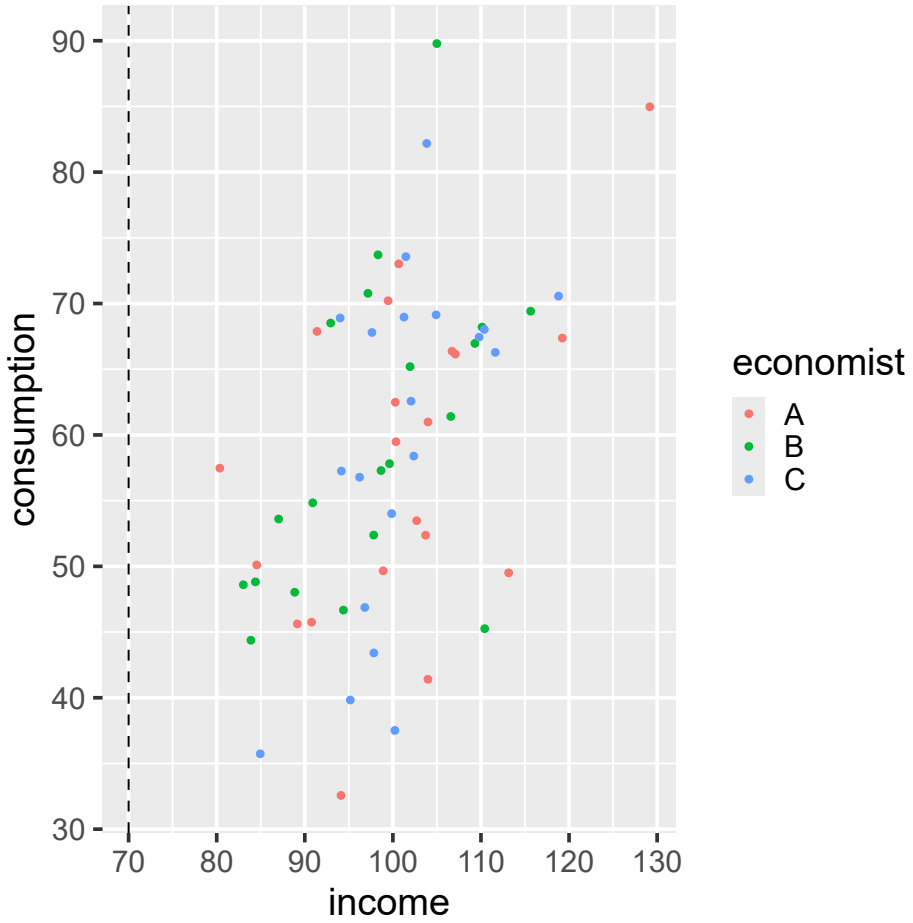


Split the Sample Between the Three Economists

```
df %>%
  ggplot(aes(y = consumption,
             x = income,
             color = economist)) +
  geom_point() +
  geom_vline(xintercept = 70, linetype = "dashed")
```

```
knitr::kable(sample_n(df,6), format = "html")
```

economist	obs	income	consumption
C	15	96.21703	56.77998
A	7	100.34957	59.47988
A	19	80.35239	57.47065
B	5	110.42183	45.25598
A	13	91.38508	67.87680
C	17	101.26113	68.96492

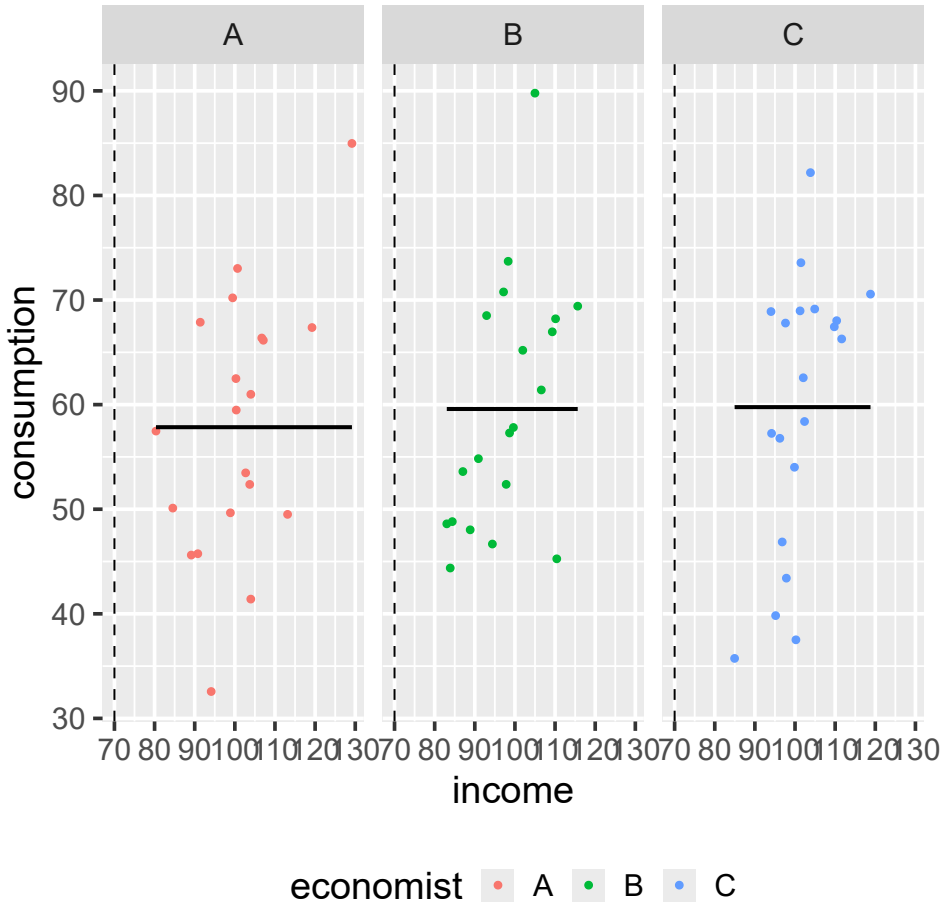


Underffiting: High Bias, Low Variance

The model: unconditional mean

$$consumption_i = \beta_0 + \varepsilon_i$$

```
df %>%
  ggplot(aes(y = consumption,
             x = income,
             color = economist)) +
  geom_point() +
  geom_smooth(method = lm,
              formula = y ~ 1,
              se = FALSE,
              color = "black") +
  facet_wrap(~ economist) +
  geom_vline(xintercept = 70, linetype = "dashed") +
  theme(legend.position = "bottom")
```

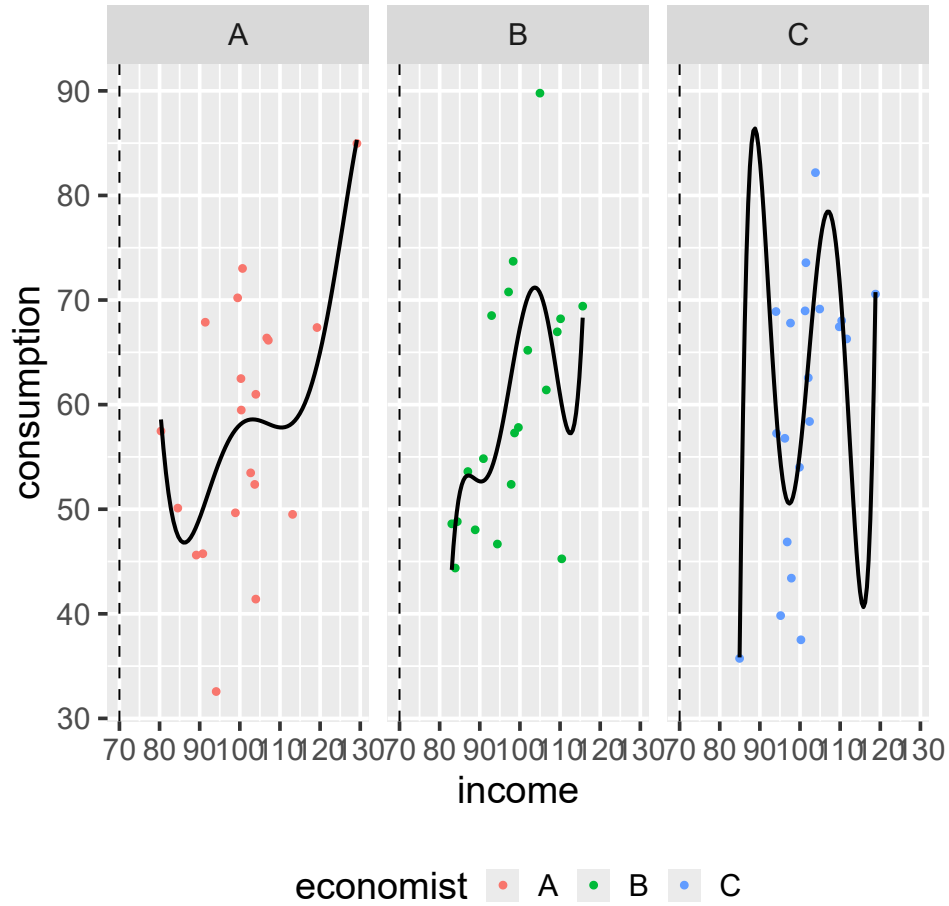


Overfitting: Low Bias, High Variance

The model: high-degree polynomial

$$consumption_i = \beta_0 + \sum_{j=1}^{p} \beta_j \times income_i^j + \varepsilon_i$$

```
df %>%
  ggplot(aes(y = consumption,
             x = income,
             color = economist)) +
  geom_point() +
  geom_smooth(method = lm,
              formula = y ~ poly(x,5),
              se = FALSE,
              color = "black") +
  facet_wrap(~ economist) +
  geom_vline(xintercept = 70, linetype = "dashed") +
  theme(legend.position = "bottom")
```

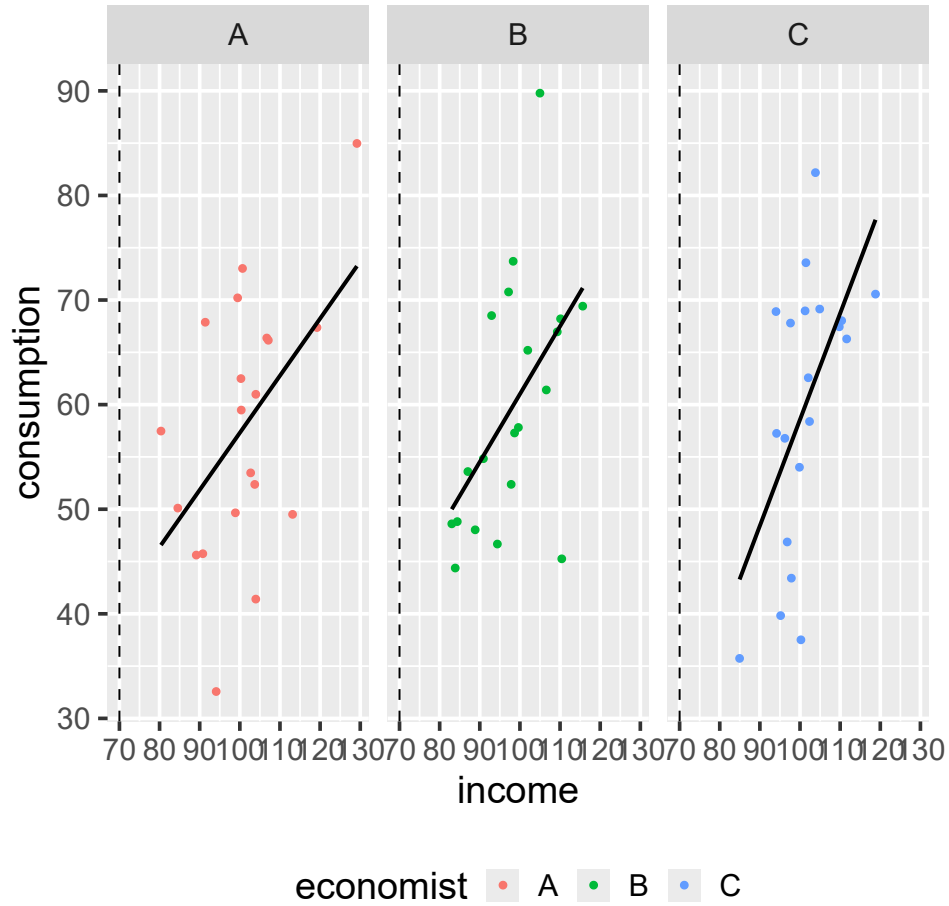


"Just Right": Bias and Variance are balanced

The model: linear

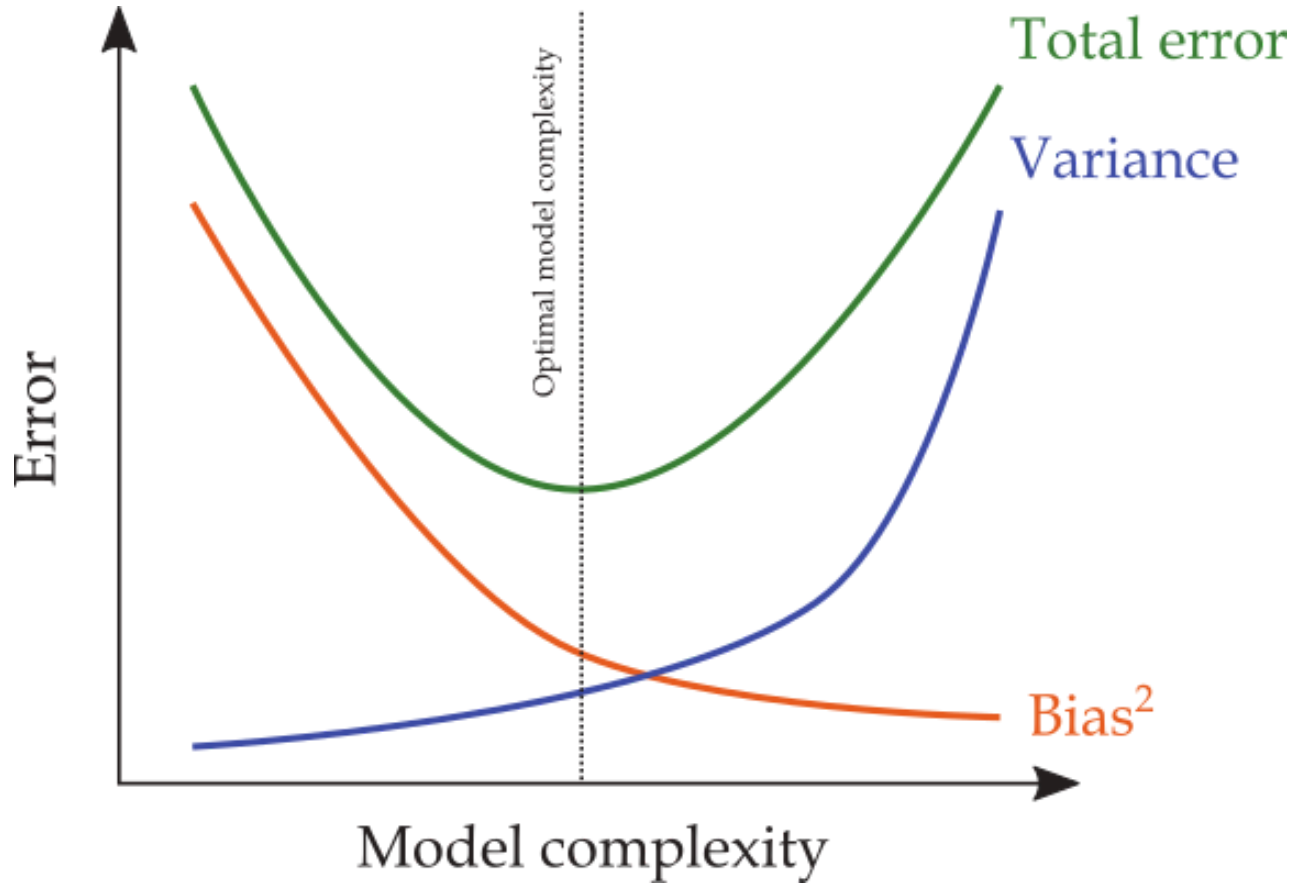
$$consumption_i = \beta_0 + \beta_1 \times income_i + \varepsilon_i$$

```
df %>%
  ggplot(aes(y = consumption,
             x = income,
             color = economist)) +
  geom_point() +
  geom_smooth(method = lm,
              formula = y ~ x,
              se = FALSE,
              color = "black") +
  facet_wrap(~ economist) +
  geom_vline(xintercept = 70, linetype = "dashed") +
  theme(legend.position = "bottom")
```



The Typical Bias-Variance Trade-off in ML

Typically, ML models strive to find levels of bias and variance that are "just right":



When is the Bias-Variance Trade-off Important?

In low-dimensional settings ($n \gg p$)

- Overfitting risk decreases as n grows
- Training MSE approximates test MSE more closely
- Conventional tools (e.g., OLS) often perform adequately
- *BUT:* Bias-variance tradeoff still matters with limited samples

INTUITION: As $n \to \infty$, model parameters converge to their true values.

In high-dimensional settings ($n \ll p$)

- Overfitting becomes almost inevitable
- Training MSE significantly underestimates test MSE
- Regularization becomes essential (LASSO, ridge, elastic net)
- Feature selection and dimensionality reduction gain importance

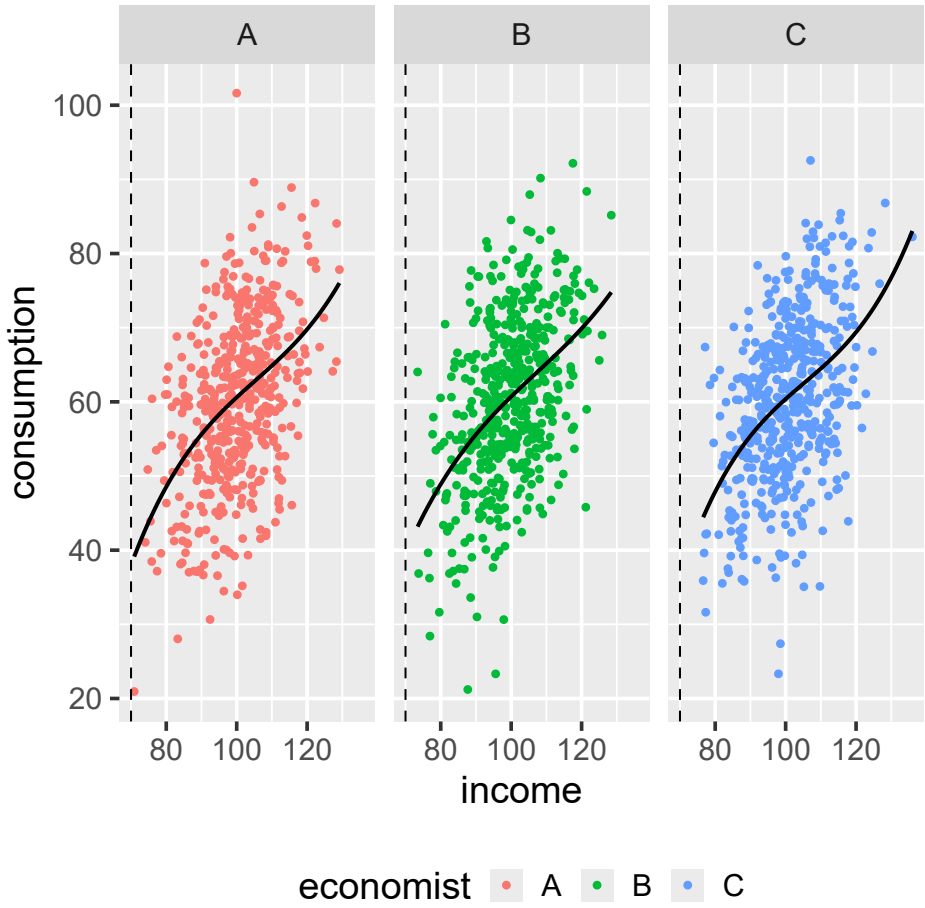
Bias-Variance Trade-off in Low-dimensional Settings

The model is a 3rd degree polynomial

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \beta_3 X_i^3 + \varepsilon_i$$

only now, the sample size for each economist increases to $N = 500$.

> **INTUITION:** as $n \to \infty$, $\hat{\beta}_2$ and $\hat{\beta}_3$ converge to their true value, zero.



Regularization

Regularization: Directly Controlling Model Complexity

- The bias-variance tradeoff shows that *increased complexity* often leads to overfitting
- **Regularization** explicitly penalizes model complexity during training
  - Adds a complexity penalty to the loss function
  - Forces the model to balance fitting training data with staying simple

> Regularization typically results in models that are mathematically "wrong" but predict more accurately.

How to Detect Overfitting Without a Test Set?

The test set MSE is *unobservable* during model development. The solution:

- **Create our own "mini test sets"** from the training data
    - Split available data into training and validation portions
    - Train on one part, evaluate on another
- **Why this works:**
    - Under the stable DGP assumption, the validation set approximates future data
    - Allows us to estimate out-of-sample error before seeing actual test data

Validation

Split the sample into three distinct sets: a training set, a validation set and a test set:

The validation process:

1. Fit a model to the training set
2. Use the model to predict outcomes from the validation set
3. Calculate validation MSE to approximate the test-MSE
4. Once model selection is complete, evaluate on the final test set

The validation error is calculated as:

$$\text{V-MSE} = \frac{1}{n_v} \sum_{i \in \mathcal{V}} (y_i - \hat{f}(x_i))^2$$

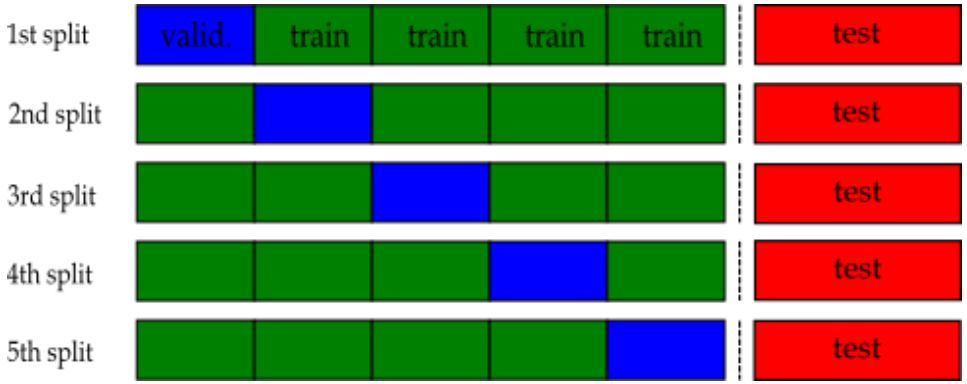
where $\mathcal{V}$ is the validation set of size n_v.

CONCERNS:

1. Sensitive to the specific training-validation split
2. Does not use all available information for training
3. Reduces effective sample size for model fitting

k-fold Cross-validation

Split the training set into k roughly equal-sized parts ($k = 5$ in this example):



Approximate the test-MSE using the mean of k split-MSEs:

$$\text{CV-MSE} = \frac{1}{k}\sum_{j=1}^{k} \text{V-MSE}_j$$

Key advantages:

1. Uses all available data for both training and validation
2. Reduces sensitivity to particular data splits
3. Provides a more robust estimate of out-of-sample performance
4. Each observation is used for both training and validation

Common choices for k:

- $k = 5$ or $k = 10$ are typical
- $k = n$ is called "leave-one-out" cross-validation

Cross-Validation and the Bias-Variance Tradeoff

- Cross-validation helps us estimate the *expected* test error by simulating multiple training-test splits
- This directly relates to our bias-variance decomposition:

$$\mathbb{E}[\mathbf{SE}^0] = \mathbf{Bias}^2 + \mathbb{V}[\hat{f}\,(x^0)] + \sigma_\epsilon^2$$

- The CV-MSE provides a practical estimate of $\mathbb{E}[\mathbf{SE}^0]$ without requiring infinite training sets
- By varying model complexity and measuring CV-MSE, we can find the optimal bias-variance tradeoff

Sample Counterparts in Practice

The CV-MSE approximates the expected test error from our bias-variance decomposition:

$$\text{CV-MSE} = \frac{1}{k}\sum_{j=1}^{k} \text{V-MSE}_j \approx \mathbb{E}_{(x^0,y^0),\mathcal{D}}\left[\left(y^0 - \hat{f}_{\mathcal{D}}\left(x^0\right)\right)^2\right]$$

This works through two levels of approximation:

1. **Training set variation**: Each fold j uses a different training set $\mathcal{D}_j$
    - This simulates drawing multiple training sets from the population
    - Helps capture variance in $\hat{f}\left(x^0\right)$ across different possible training data
2. **Test point averaging**: Within each fold, we average squared errors across validation points
    - This approximates the expectation over test points (x^0, y^0)

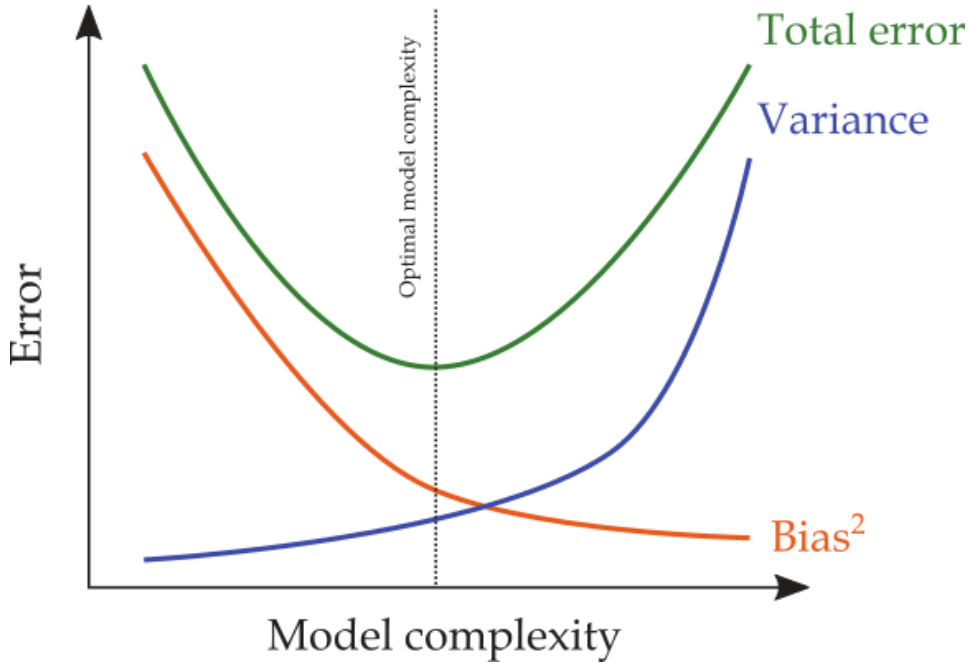
Which Model to Choose?

As model complexity increases:

- Bias typically decreases
- Variance typically increases
- CV-MSE helps us find the sweet spot

Hence, model selection amounts to choosing the complexity level that minimizes CV-MSE.

Recall that the test-MSE is unobservable, and CV-MSE is our best approximation to guide model selection.



Sounds Familiar? Same Principle Different Approach

- The model selection principle behind CV-MSE connects to criteria you already know:
  - **Adjusted R^2**: Penalizes R^2 based on degrees of freedom
  - **AIC/BIC**: Balance goodness-of-fit with complexity penalties
- All use the same core idea: $\textbf{Criterion} = \textbf{Fit measure} + \textbf{Complexity penalty}$
- **Key distinction**:
  - Traditional statistics use analytical formulas with theoretical penalties
  - ML uses cross-validation to let the data empirically determine optimal complexity
  - Both aim to avoid overfitting and improve generalization to new data

ML and Causality

ML vs. Econometrics: Key Distinctions

Apart from terminology differences, here are the fundamental contrasts:

Machine Learning	**Econometrics**
Focus on prediction $\hat{Y}$	Focus on causal effects $\hat{\beta}$
Optimize for out-of-sample accuracy	Optimize for parameter properties (unbiasedness, consistency)
Validation via prediction error	Validation via statistical significance
Requires stable environment	Enables counterfactual analysis
Often black-box approaches	Typically structural models
Question: "What will happen?"	Question: "What if we intervene?"

> ML learns correlations in data; econometrics identifies causal mechanisms.

ML and Causal Inference: The Fundamental Challenge

> *"Data are profoundly dumb about causal relationships."*
>
> — **Pearl and Mackenzie, *The Book of Why***

Why econometrics can't be replaced by ML: Intervention fundamentally violates the stable DGP assumption that ML relies on:

$$P(Y|X = x) \neq P(Y|do(X = x)),$$

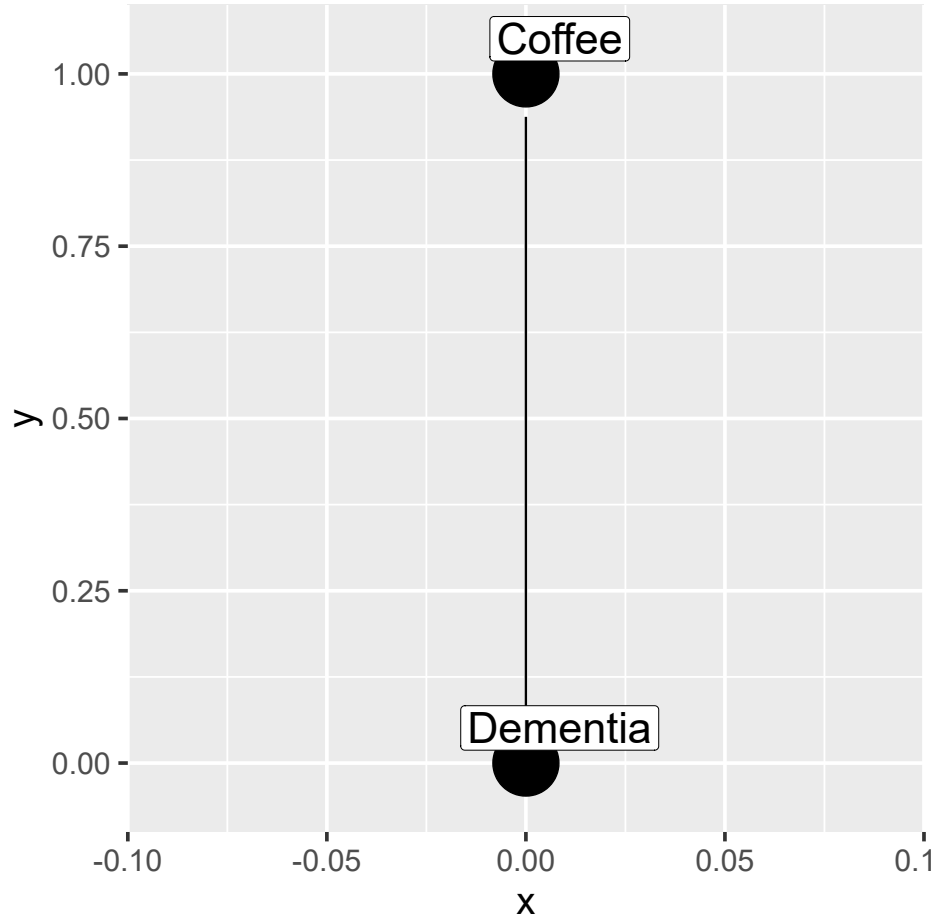
where:

- $P(Y|X = x)$ is the probability of Y given that we *observe* $X = x$ (prediction)
- $P(Y|do(X = x))$ is the probability of Y given that we *intervene* to set $X = x$ (causal effect)

> While ML excels at the first equation, econometric methods are designed to estimate the second.

From Prediction to Causation: A Simple Example

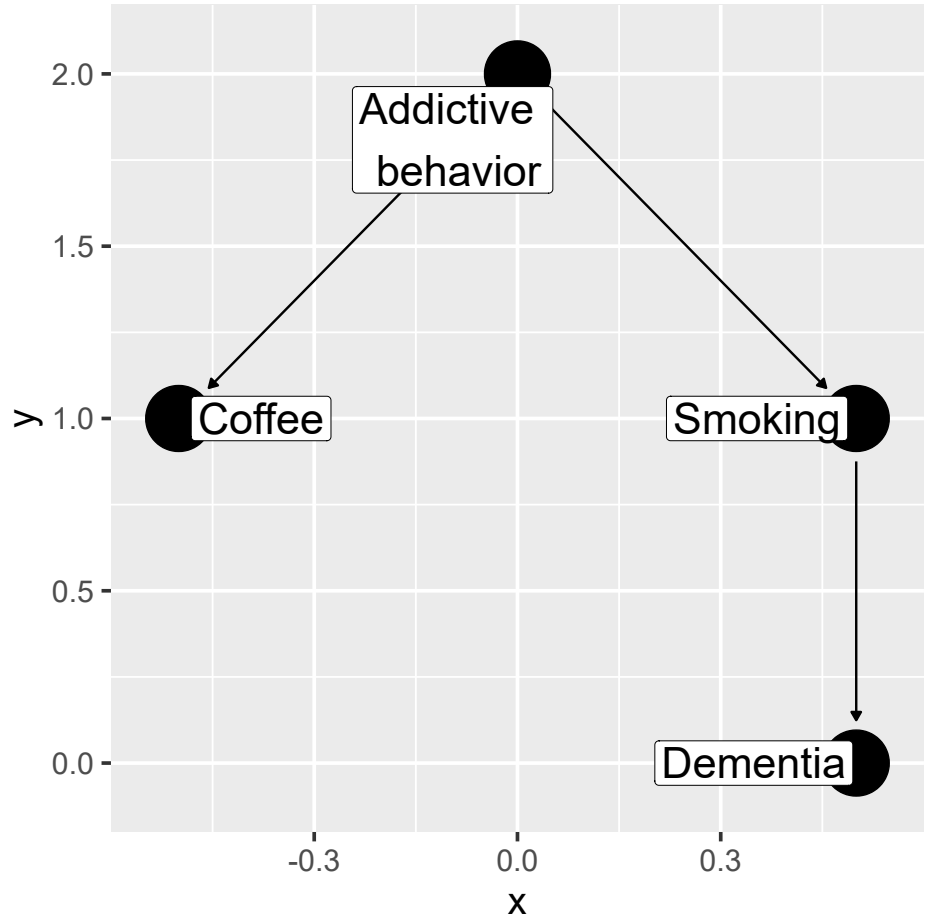
EXAMPLE: An ML algorithm finds that `coffee` consumption is a strong predictor of `dementia`. Based on this correlation, should we recommend avoiding coffee?



To Explain or to Predict?

- Despite being a good *predictor*, `coffee` has no causal effect on `dementia`
- The correlation exists because both are influenced by an underlying factor: `addictive` behavior
- `Smoking` is the actual causal factor for `dementia`

> This illustrates why ML's predictive power doesn't translate directly to policy effectiveness. The $P(Y|X = x) \neq P(Y|do(X = x))$ distinction is not just theoretical—it has practical implications.



ML in Aid of Econometrics

Consider the standard causal inference problem:

$$Y_i = \alpha + \underbrace{\tau D_i}_{\text{causal parameter}} + \underbrace{f(X_i)}_{\text{confounding factors}} + \varepsilon_i, \quad \text{for } i = 1, \ldots, n$$

where:

- Y_i is the outcome variable
- $D_i \in \{0, 1\}$ is the treatment indicator
- X_i is a potentially high-dimensional vector of control variables
- τ is our target: the average treatment effect (ATE)

ML in Aid of Econometrics

Consider the standard causal inference problem:

$$Y_i = \alpha + \underbrace{\tau D_i}_{\text{causal parameter}} + \underbrace{f(X_i)}_{\text{confounding factors}} + \varepsilon_i, \quad \text{for } i = 1, \ldots, n$$

Key insight: We can decompose this problem into:

1. **Causal identification** (econometrics): Ensuring τ has a causal interpretation
2. **Nuisance estimation** (machine learning): Modeling complex relationships in $f(X_i)$

Why this matters:

- When p is large (many controls), traditional methods struggle
- ML excels at modeling $f(X_i)$ without strong functional form assumptions
- This hybrid approach allows us to focus on causal parameter τ while letting ML handle the high-dimensional prediction task

```
slides |> end()
```

Source code

References

[1] A. Agrawal, J. Gans, and A. Goldfarb. *Prediction Machines: The Simple Economics of Artificial Intelligence*. Harvard Business Review Press, 2018.

[2] S. Athey. "The impact of machine learning on economics". In: *The Economics of Artificial Intelligence: An Agenda*. University of Chicago Press, 2018.

[3] L. Breiman. "Statistical modeling: The two cultures (with comments and a rejoinder by the author)". In: *Statistical science* 16.3 (2001), pp. 199-231.

[4] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer, 2009 .פבר. ISBN: 9780387848570.

[5] G. James, T. Hastie, D. Witten, et al. *An Introduction to Statistical Learning: With Applications in R*. Springer Texts in Statistics. Springer London, Limited, 2021. ISBN: 9781461471370.

[6] S. Mullainathan and J. Spiess. "Machine learning: an applied econometric approach". In: *Journal of Economic Perspectives* 31.2 (2017), pp. 87-106.

References

[1] S. Mullainathan and J. Spiess. "Machine learning: an applied econometric approach". In: *Journal of Economic Perspectives* 31.2 (2017), pp. 87-106.

[2] G. Shmueli. "To explain or to predict?" In: *Statistical science* 25.3 (2010), pp. 289-310.