Introduction to Machine Learning

Machine Learning course
Department of Economics and Management
University of Pavia

AY 2025-2026

Some Take Home Messages

This class is about supervised learning: building models from data that predict an outcome using a collection of input features.

- There are some powerful and exciting tools for making predictions from data.
- They are not magic! You should be skeptical. They require good data and proper internal validation.
- Human judgement and ingenuity are essential for their success.
- With big data
  - model fitting takes longer. This might test our patience for model evaluation and comparison.
  - difficult to look at the data; might be contaminated in parts.

  Careful subsampling can help with both of these.

Some Definitions

Machine Learning constructs algorithms that can learn from data.

Statistical Learning is a branch of applied statistics that emerged in response to machine learning, emphasizing statistical models and assessment of uncertainty.

Data Science is the extraction of knowledge from data, using ideas from mathematics, statistics, machine learning, computer science, engineering, ...

All of these are very similar — with different emphases.

Some Definitions

Machine Learning constructs algorithms that can learn from data.

Statistical Learning is a branch of applied statistics that emerged in response to machine learning, emphasizing statistical models and assessment of uncertainty.

Data Science is the extraction of knowledge from data, using ideas from mathematics, statistics, machine learning, computer science, engineering, ...

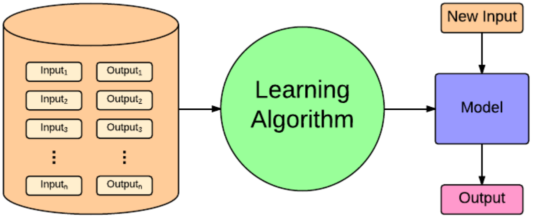
All of these are very similar — with different emphases.

For Statisticians: 15 minutes of fame

2009 “I keep saying the sexy job in the next ten years will be statisticians. And I’m not kidding!” Hal Varian, Chief Economist Google

2012 “Data Scientist: The sexiest job of the 21st century.” Harvard Business Review

The Supervising Learning Paradigm



Training Data   Fitting   Prediction

Traditional statistics: domain experts work for 10 years to learn good features; they bring the statistician a small clean dataset

Today's approach: we start with a large dataset with many features, and use a machine learning algorithm to find the good ones. A huge change.

Internal Model Validation

- IMPORTANT! Don't trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.
- Eg: divide data into two parts A and B. Run algorithm on part A and then test it on part B.
  Algorithm must not have seen any of the data in part B.
- If it works in part B, you have (some) confidence in it

Internal Model Validation

- IMPORTANT! Don't trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.
- Eg: divide data into two parts A and B. Run algorithm on part A and then test it on part B.
  Algorithm must not have seen any of the data in part B.
- If it works in part B, you have (some) confidence in it

Simple? Yes

Internal Model Validation

- IMPORTANT! Don't trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.
- Eg: divide data into two parts A and B. Run algorithm on part A and then test it on part B.
  Algorithm must not have seen any of the data in part B.
- If it works in part B, you have (some) confidence in it

  Simple? Yes
  Done properly in practice? Rarely

Internal Model Validation

- IMPORTANT! Don't trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.
- Eg: divide data into two parts A and B. Run algorithm on part A and then test it on part B.
  Algorithm must not have seen any of the data in part B.
- If it works in part B, you have (some) confidence in it

Simple? Yes
Done properly in practice? Rarely

In God we trust. All others bring data.

Big data vary in *shape*. These call for different approaches.

Wide Data

Thousands / Millions of Variables

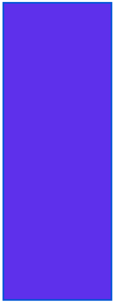
Hundreds of Samples

Screening and fdr,
Lasso, SVM, Stepwise

We have too many variables; prone to overfitting.

Need to remove variables, or regularize, or both.

Tall Data



Tens / Hundreds of Variables

Thousands / Millions of Samples

GLM, Random Forests,
Boosting, Deep Learning

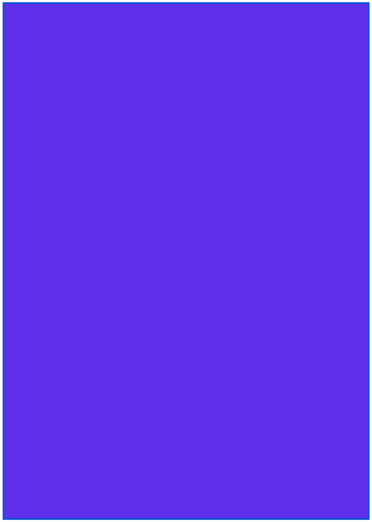
Sometimes simple models (linear) don't suffice.

We have enough samples to fit nonlinear models with many interactions, and not too many variables.

Good automatic methods for doing this.

Big data vary in *shape*. These call for different approaches.

Tall and Wide Data



Thousands / Millions of Variables

Millions to Billions of Samples

Tricks of the Trade

Exploit sparsity

Random projections / hashing

Variable screening

Subsample rows

Divide and recombine

Case/ control sampling

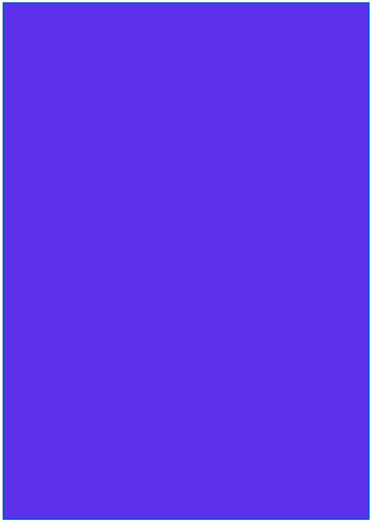
MapReduce

ADMM (divide and conquer)

.
.
.

Big data vary in *shape*. These call for different approaches.

Tall and Wide Data



Thousands / Millions of Variables

Millions to Billions of Samples

Tricks of the Trade

Exploit sparsity

Random projections / hashing

Variable screening

Subsample rows

Divide and recombine

Case/ control sampling

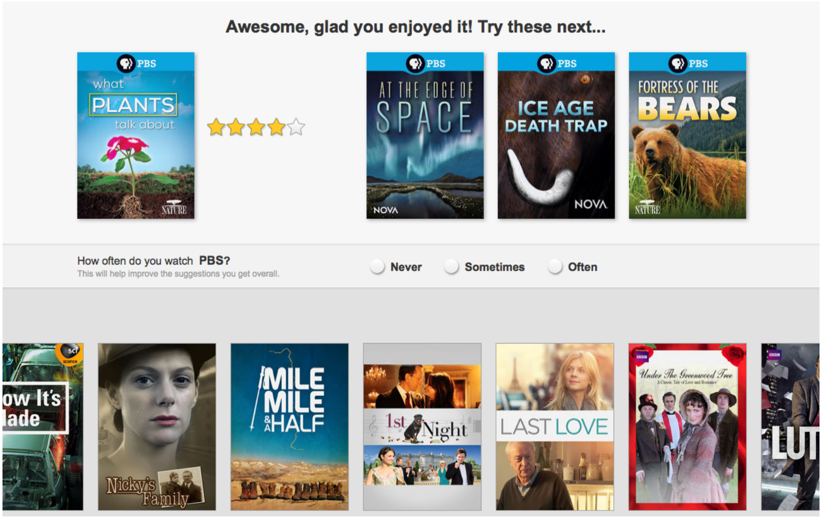
MapReduce

ADMM (divide and conquer)

.
.
.

join Google

The Netflix Recommender

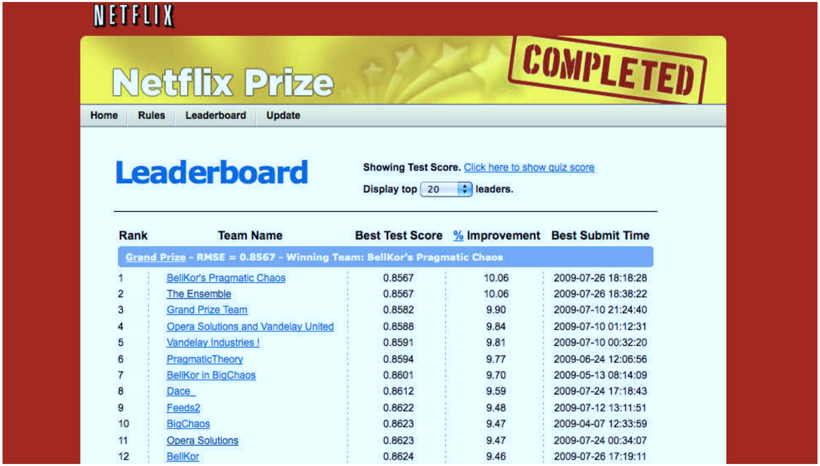


Awesome, glad you enjoyed it! Try these next...

How often do you watch **PBS?**
This will help improve the suggestions you get overall.

Never Sometimes Often

The Netflix Prize — 2006–2009



Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11

41K teams participated! Competition ran for nearly 3 years. Winner "BellKor's Pragmatic Chaos", essentially tied with "The Ensemble".

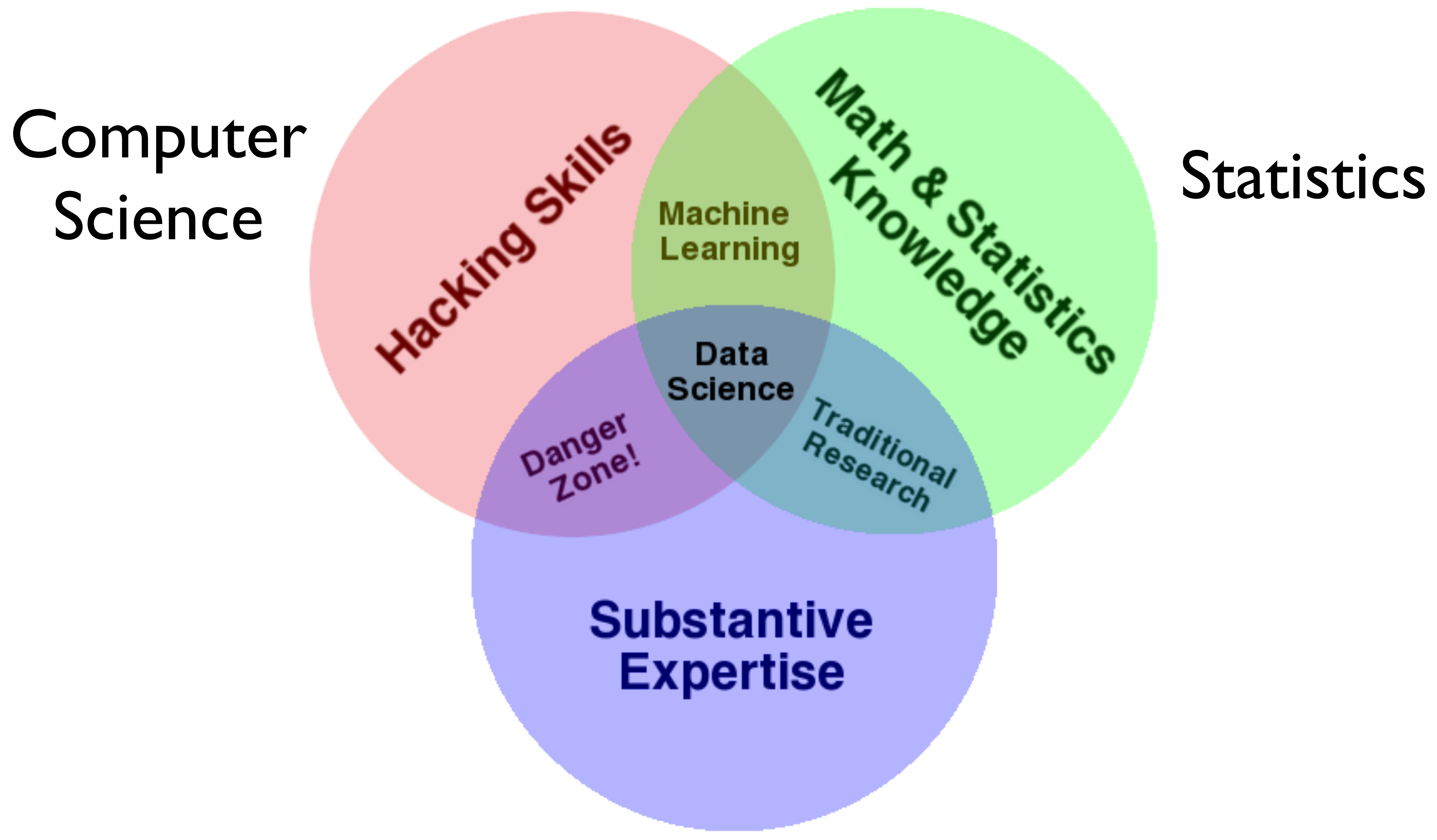
The Netflix Data Set

$$\begin{array}{lccccc}
 & \text{movie I} & \text{movie II} & \text{movie III} & \text{movie IV} & \cdots \\
\text{User A} & 1 & ? & 5 & 4 & \cdots \\
\text{User B} & ? & 2 & 3 & ? & \cdots \\
\text{User C} & 4 & 1 & 2 & ? & \cdots \\
\text{User D} & ? & 5 & 1 & 3 & \cdots \\
\text{User E} & 1 & 2 & ? & ? & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

- Training Data:
  $480K$ users, $18K$ movies, $100M$ ratings (1–5)
  (99% ratings missing)
- Goal:
  \$1M prize for 10% reduction in RMSE over Cinematch
- BellKor's Pragmatic Chaos declared winners on 9/21/2009

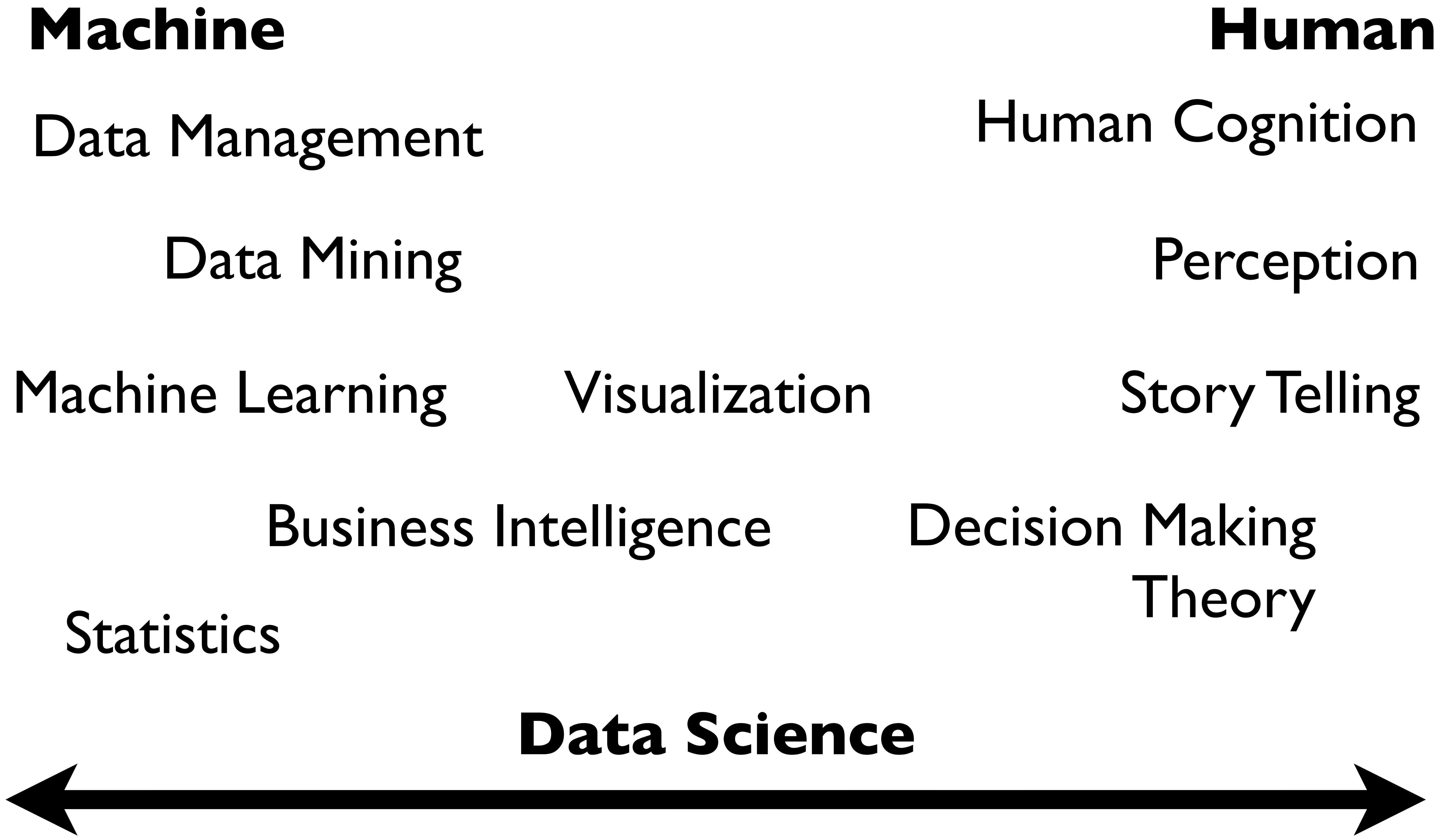
  Used ensemble of models, an important ingredient being low-rank factorization (SVD)

Data Science

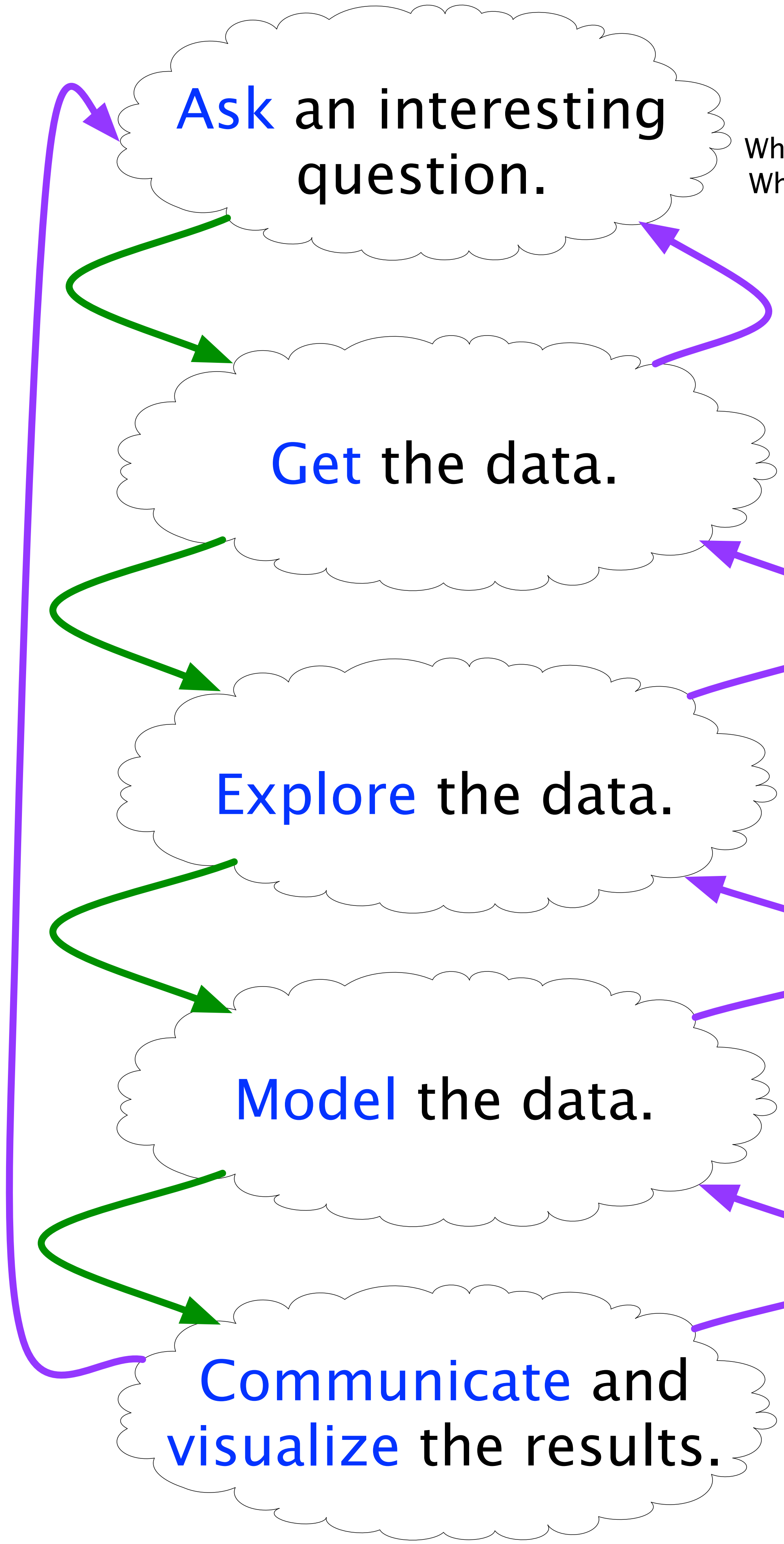


Domain Science

Drew Conway



Inspired by Daniel Keim, "Visual Analytics: Definition, Process, and Challenges"


Ask an interesting question.
Get the data.
Explore the data.
Model the data.
Communicate and visualize the results.

What is the scientific **goal**?
What would you do if you had all the **data**?
What do you want to **predict** or **estimate**?

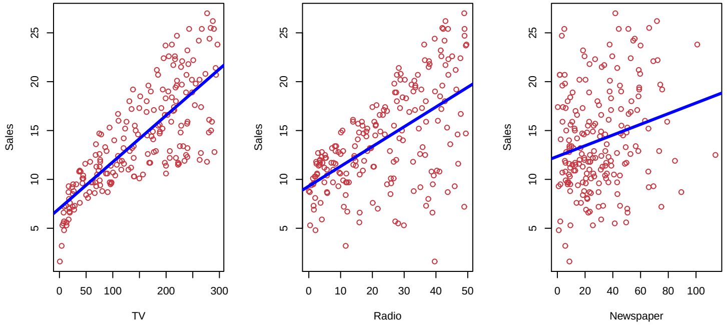
How were the data **sampled**?
Which data are **relevant**?
Are there **privacy** issues?

Plot the data.
Are there **anomalies**?
Are there **patterns**?

Build a model.
Fit the model.
Validate the model.

What did we **learn**?
Do the results make **sense**?
Can we tell a **story**?

What is Statistical Learning?



Shown are $\texttt{Sales}$ vs $\texttt{TV}$, $\texttt{Radio}$ and $\texttt{Newspaper}$, with a blue linear-regression line fit separately to each.
Can we predict $\texttt{Sales}$ using these three?
Perhaps we can do better using a model

$$\texttt{Sales} \approx f(\texttt{TV}, \texttt{Radio}, \texttt{Newspaper})$$

Notation

Here `Sales` is a *response* or *target* that we wish to predict. We generically refer to the response as Y.
`TV` is a *feature*, or *input*, or *predictor*; we name it X_1.
Likewise name `Radio` as X_2, and so on.
We can refer to the *input vector* collectively as

$$X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix}$$

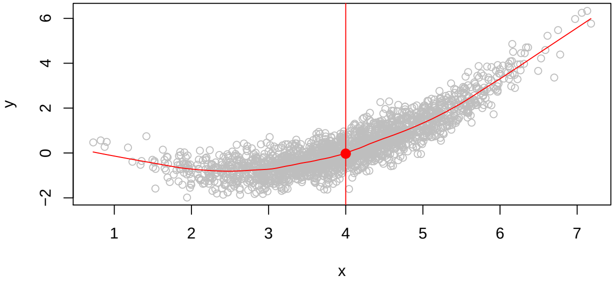
Now we write our model as

$$Y = f(X) + \epsilon$$

where ϵ captures measurement errors and other discrepancies.

What is $f(X)$ good for?

- With a good f we can make predictions of Y at new points $X = x$.
- We can understand which components of $X = (X_1, X_2, \ldots, X_p)$ are important in explaining Y, and which are irrelevant. e.g. `Seniority` and `Years of Education` have a big impact on `Income`, but `Marital Status` typically does not.
- Depending on the complexity of f, we may be able to understand how each component X_j of X affects Y.



Is there an ideal $f(X)$? In particular, what is a good value for $f(X)$ at any selected value of X, say $X = 4$? There can be many Y values at $X = 4$. A good value is

$$f(4) = E(Y|X = 4)$$

$E(Y|X = 4)$ means *expected value* (average) of Y given $X = 4$.

This ideal $f(x) = E(Y|X = x)$ is called the *regression function*.

The regression function $f(x)$

- Is also defined for vector X; e.g.
  $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$

The regression function $f(x)$

- Is also defined for vector X; e.g. $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$
- Is the *ideal* or *optimal* predictor of Y with regard to mean-squared prediction error: $f(x) = E(Y|X = x)$ is the function that minimizes $E[(Y - g(X))^2|X = x]$ over all functions g at all points $X = x$.

The regression function $f(x)$

- Is also defined for vector X; e.g. $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$
- Is the *ideal* or *optimal* predictor of Y with regard to mean-squared prediction error: $f(x) = E(Y|X = x)$ is the function that minimizes $E[(Y - g(X))^2|X = x]$ over all functions g at all points $X = x$.
- $\epsilon = Y - f(x)$ is the *irreducible* error — i.e. even if we knew $f(x)$, we would still make errors in prediction, since at each $X = x$ there is typically a distribution of possible Y values.

The regression function $f(x)$

- Is also defined for vector X; e.g. $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$
- Is the *ideal* or *optimal* predictor of Y with regard to mean-squared prediction error: $f(x) = E(Y|X = x)$ is the function that minimizes $E[(Y - g(X))^2|X = x]$ over all functions g at all points $X = x$.
- $\epsilon = Y - f(x)$ is the *irreducible* error — i.e. even if we knew $f(x)$, we would still make errors in prediction, since at each $X = x$ there is typically a distribution of possible Y values.
- For any estimate $\hat{f}(x)$ of $f(x)$, we have

$$E[(Y - \hat{f}(X))^2|X = x] = \underbrace{[f(x) - \hat{f}(x)]^2}_{\textit{Reducible}} + \underbrace{\mathrm{Var}(\epsilon)}_{\textit{Irreducible}}$$

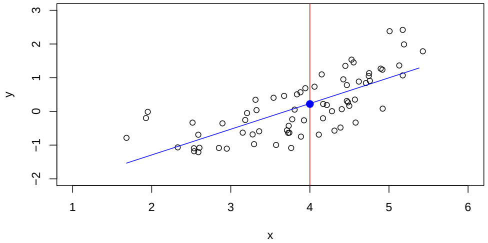
Parametric and structured models

The *linear* model is an important example of a parametric model:

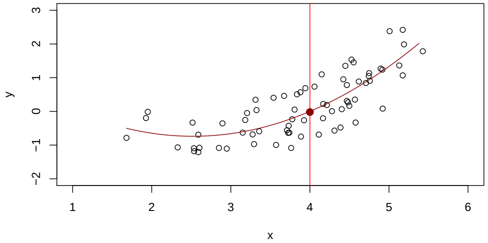
$$f_L(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots \beta_p X_p.$$

- A linear model is specified in terms of $p + 1$ parameters $\beta_0, \beta_1, \dots, \beta_p$.
- We estimate the parameters by fitting the model to training data.
- Although it is *almost never correct*, a linear model often serves as a good and interpretable approximation to the unknown true function $f(X)$.

A linear model $\hat{f}_L(X) = \hat{\beta}_0 + \hat{\beta}_1 X$ gives a reasonable fit here



A quadratic model $\hat{f}_Q(X) = \hat{\beta}_0 + \hat{\beta}_1 X + \hat{\beta}_2 X^2$ fits slightly better.



Some trade-offs

- Prediction accuracy versus interpretability.
  — Linear models are easy to interpret; thin-plate splines are not.

Some trade-offs

- Prediction accuracy versus interpretability.
  — Linear models are easy to interpret; thin-plate splines are not.
- Good fit versus over-fit or under-fit.
  — How do we know when the fit is just right?

Some trade-offs

- Prediction accuracy versus interpretability.
  — Linear models are easy to interpret; thin-plate splines are not.
- Good fit versus over-fit or under-fit.
  — How do we know when the fit is just right?
- Parsimony versus black-box.
  — We often prefer a simpler model involving fewer variables over a black-box predictor involving them all.

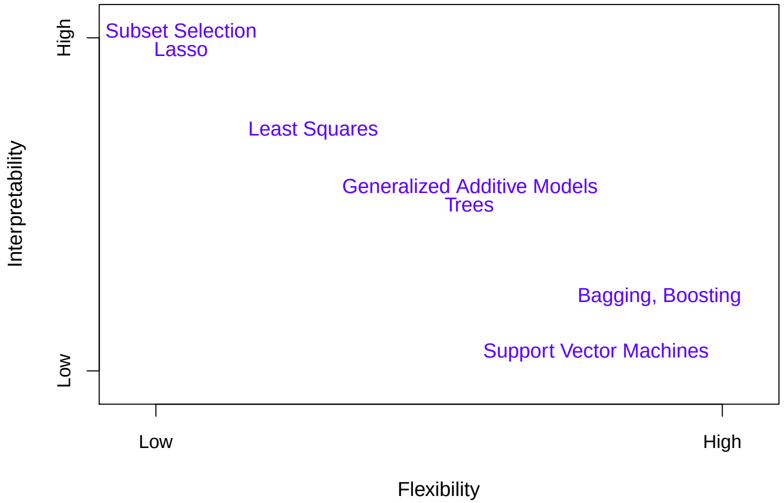


FIGURE 2.7. *A representation of the tradeoff between flexibility and inte pretability, using different statistical learning methods. In general, as the flexib*

Assessing Model Accuracy

Suppose we fit a model $\hat{f}(x)$ to some training data $\mathsf{Tr} = \{x_i, y_i\}_1^N$, and we wish to see how well it performs.

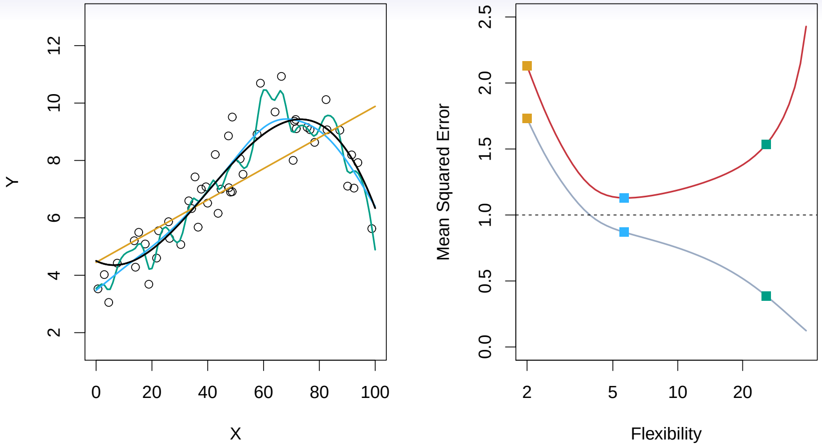
- We could compute the average squared prediction error over Tr:

$$\text{MSE}_{\mathsf{Tr}} = \text{Ave}_{i \in \mathsf{Tr}}[y_i - \hat{f}(x_i)]^2$$

This may be biased toward more overfit models.

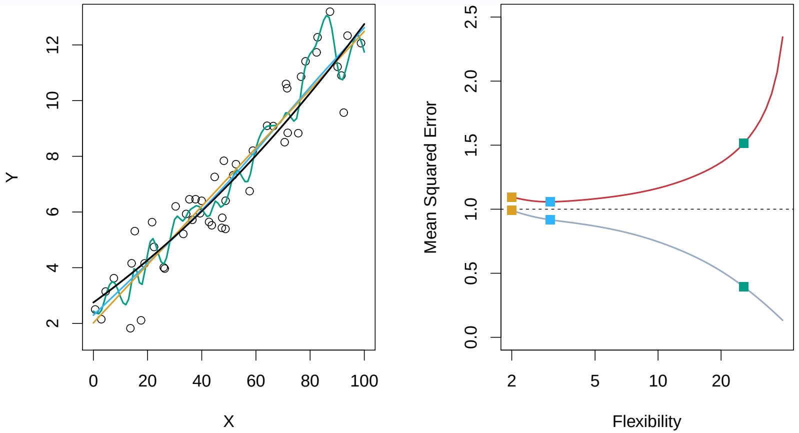
- Instead we should, if possible, compute it using fresh *test* data $\mathsf{Te} = \{x_i, y_i\}_1^M$:

$$\text{MSE}_{\mathsf{Te}} = \text{Ave}_{i \in \mathsf{Te}}[y_i - \hat{f}(x_i)]^2$$



Black curve is truth. Red curve on right is MSE_{Te}, grey curve is MSE_{Tr}. Orange, blue and green curves/squares correspond to fits of different flexibility.

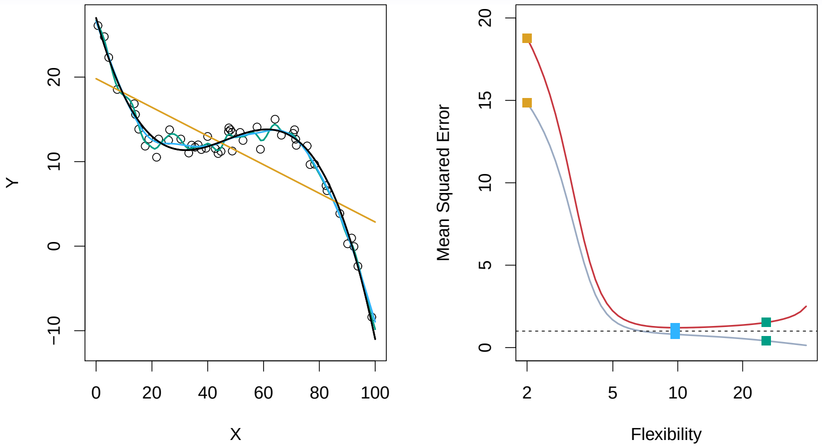
FIGURE 2.9. Left: *Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves).* Right: *Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.*



Here the truth is smoother, so the smoother fit and linear model do really well.

FIGURE 2.10. *Details are as in Figure 2.9, using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.*

the data.



Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

FIGURE 2.11. *Details are as in Figure 2.9, using a different f that is far from linear. In this setting, linear regression provides a very poor fit to the data.*

Bias-Variance Trade-off

Suppose we have fit a model $\hat{f}(x)$ to some training data Tr, and let (x_0, y_0) be a test observation drawn from the population. If the true model is $Y = f(X) + \epsilon$ (with $f(x) = E(Y|X = x)$), then

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \text{Var}(\hat{f}(x_0)) + [\text{Bias}(\hat{f}(x_0))]^2 + \text{Var}(\epsilon).$$

The expectation averages over the variability of y_0 as well as the variability in Tr. Note that $\text{Bias}(\hat{f}(x_0))] = E[\hat{f}(x_0)] - f(x_0)$.

Typically as the *flexibility* of $\hat{f}$ increases, its variance increases, and its bias decreases. So choosing the flexibility based on average test error amounts to a *bias-variance trade-off.*

Bias-variance trade-off for the three examples

36 2. Statistical Learning

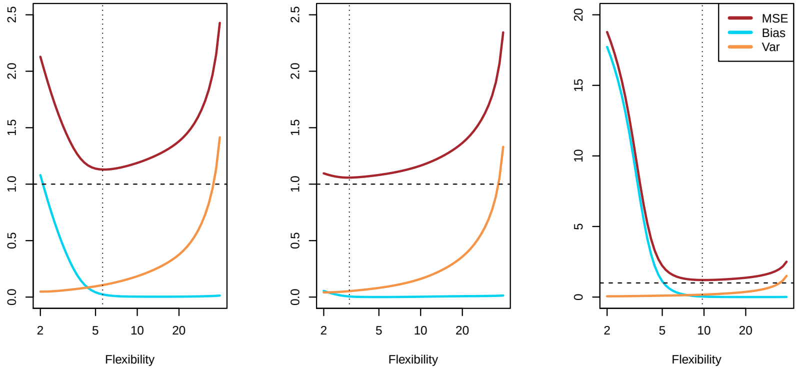
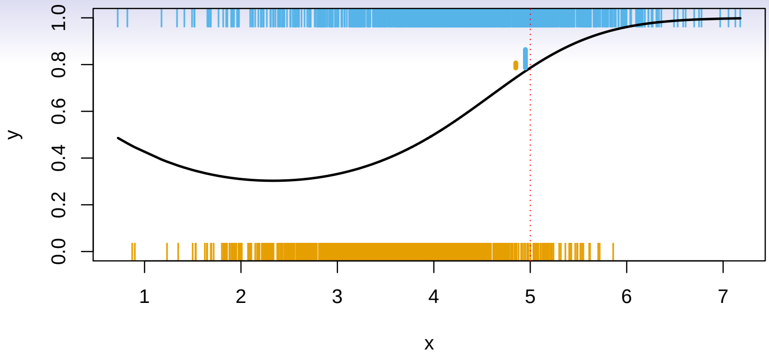


FIGURE 2.12. *Squared bias (blue curve), variance (orange curve), Var(ϵ) (dashed line), and test MSE (red curve) for the three data sets in Figures 2.9–2.11. The vertical dashed line indicates the flexibility level corresponding to the smallest test MSE.*

Classification Problems

Here the response variable Y is *qualitative* — e.g. email is one of $\mathcal{C} = (\texttt{spam}, \texttt{ham})$ (`ham`=good email), digit class is one of $\mathcal{C} = \{\texttt{0}, \texttt{1}, \ldots, \texttt{9}\}$. Our goals are to:

- Build a classifier $C(X)$ that assigns a class label from $\mathcal{C}$ to a future unlabeled observation X.
- Assess the uncertainty in each classification
- Understand the roles of the different predictors among $X = (X_1, X_2, \ldots, X_p)$.



Is there an ideal $C(X)$? Suppose the K elements in $\mathcal{C}$ are numbered $1, 2, \ldots, K$. Let

$$p_k(x) = \Pr(Y = k | X = x),\ k = 1, 2, \ldots, K.$$

These are the *conditional class probabilities* at x; e.g. see little barplot at $x = 5$. Then the *Bayes optimal* classifier at x is

$$C(x) = j \text{ if } p_j(x) = \max\{p_1(x), p_2(x), \ldots, p_K(x)\}$$

Classification: some details

- Typically we measure the performance of $\hat{C}(x)$ using the misclassification error rate:

$$\text{Err}_{\mathsf{Te}} = \text{Ave}_{i \in \mathsf{Te}} I[y_i \neq \hat{C}(x_i)]$$

- The Bayes classifier (using the true $p_k(x)$) has smallest error (in the population).

Classification: some details

- Typically we measure the performance of $\hat{C}(x)$ using the misclassification error rate:

$$\mathrm{Err}_{\mathsf{Te}} = \mathrm{Ave}_{i \in \mathsf{Te}} I[y_i \neq \hat{C}(x_i)]$$

- The Bayes classifier (using the true $p_k(x)$) has smallest error (in the population).
- Support-vector machines build structured models for $C(x)$.
- We will also build structured models for representing the $p_k(x)$. e.g. Logistic regression, generalized additive models.