

# MY474: Applied Machine Learning for Social Science

## Lecture 1: Getting Started, Brief Introduction to Machine Learning

Blake Miller

15 January 2021

# Agenda

1. Introductions
2. Syllabus
3. What is machine learning?
4. R Setup instructions
5. R, Rmarkdown Review
6. Overview of Statistical Learning



My name is Blake Miller. I'm originally from San Diego, California. You can call me Dr. Miller or Blake. I use he/him pronouns.



I'm a political scientist and a data scientist. I study how (authoritarian) governments use data to manipulate information and repress citizens.  
(Source: China Foto Press)

## About the course

## Learning from Data

- ▶ **Fact:** The amount of data and information collected and stored is constantly increasing, due to advances in data collection, computerization of many aspects of life and breakthroughs in storage technology.
- ▶ **Consequence:** Statistical problems have increased both in size and complexity.
- ▶ **The data analyst's job:** make sense of all these data! Identify patterns and trends, uncover interesting relationships among the variables and/or the observations, predict future behavior.
- ▶ Machine learning can help us process and understand these data.

# Learning from Data

- ▶ Technology helps
  - ▶ Faster computers → more flexible and thus more powerful techniques → fewer modeling assumptions
  - ▶ New capabilities from graphical processing units (GPUs)
- ▶ But not always: Faster computers do not solve all problems
  - ▶ Some problems are inherently computationally intractable
  - ▶ “Easy” black-box data analysis can lead to a lot of misuse and misunderstanding
  - ▶ Flexible models can overfit (too much of a good thing)
  - ▶ Understanding underlying assumptions and interpreting conclusions correctly remains as important as ever

## Major concepts

- ▶ **Bias/Variance Tradeoff:** How complex should our model be?  
How closely should we fit the model to our data?
- ▶ **Generalization:** How can we measure if a model will generalize well to new data?
- ▶ **Cross-validation:** How do we estimate generalization error without throwing out data?
- ▶ **Regularization:** How do we home in on important features in our data?
- ▶ **Feature engineering:** How can we best represent the data for our specific task?
- ▶ **Model selection:** How do we select a model in a principled way?

## Methods and tools

- ▶ **Estimation:** gradient descent, stochastic gradient descent, maximum likelihood, expectation maximization (EM)
- ▶ **Linear Models:** Logistic regression, linear discriminant analysis, support vector machines, perceptron
- ▶ **Regularization:** cost-complexity pruning, lasso regression, ridge regression, elasticnet
- ▶ **Non-parametric methods:** k-nearest neighbors (KNN), decision trees, random forests, gradient boosted trees
- ▶ **Dimension reduction:** principal component analysis (PCA)
- ▶ **Clustering:** k-means, hierarchical clustering

# Social science implications of the rise of ML

- ▶ **Governance**
  - ▶ How do ML tools affect the durability of democracies/autocracies?
  - ▶ How can we ensure that critical ML systems are transparent?
  - ▶ How do we hold organizations/governments accountable for misuse of ML systems?
- ▶ **Social Impact**
  - ▶ How do we avoid algorithmic bias and ensure fairness of ML systems?
  - ▶ How can we avoid negative externalities of ML systems (violence, discrimination, false news, misinformation)?

# Social science implications of the rise of ML

- ▶ **Privacy**
  - ▶ How are ML systems used by states and corporations for mass surveillance?
  - ▶ How can we maintain control over our data and privacy in the age of surveillance capitalism?
- ▶ **Interpretability**
  - ▶ How do we identify and address negative social externalities of extremely complex ML systems?
  - ▶ How can we uncover and mitigate latent biases in human-generated training data?

## Getting Meta: Social Science About ML



Black Mirror portrayed a society where your social standing and access to public goods was governed by a social-media-based rating system. (Source: BBC.co.uk)

# Getting Meta: Social Science About ML



## Experimenting with carrots and sticks

China's Social Credit System materializes in local and national pilots<sup>1</sup>

- Round 1 (August 2015)
- Round 2 (April 2016)

**Zhengzhou (Henan):**  
Refusing to comply with a court order to pay debt results in dial tone of the person's phone to be changed to a "shaming" announcement.

**Wuhan (Hubei):**  
Files for students over 18 recording misbehavior like cheating on exams, plagiarism, and unpaid tuition fees.

**Luzhou (Sichuan):**  
Social Credit System for the liquor industry. Baijiu-producing companies are monitored for compliance with regulations.

**Rongcheng (Shandong):**  
Individual scores and grades for residents. Misbehavior (littering, jaywalking) results in score deduction and punishment; exemplary behavior (caring for aged parents) in good ratings and benefits.

**Shanghai:**  
Facial recognition app retrieves data on residents from 100+ government sources and assigns ratings. App is also used for ratings of local businesses and restaurants.



### Selected measures applied nationwide or across multiple provinces

 Limits on so-called "high-end consumption" for individuals defying court orders to repay money and representatives of blacklisted companies: no high speed rail, no flights, no private schools for their children, etc.

 Renting apartments deposit-free if a background-check conducted through the Sesame Credit app is positive.

 Different classification and treatment of natural and legal persons for tax purposes, customs, etc. depending on their rating.

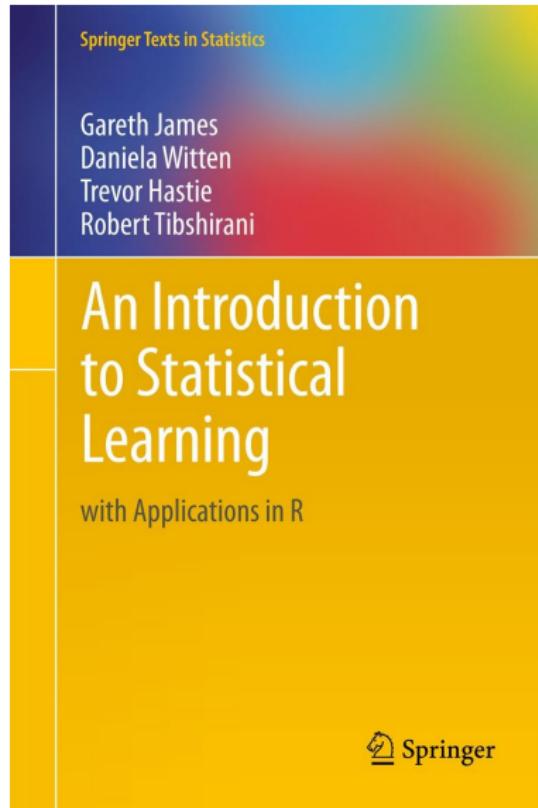
 Restricted access to public procurement, government land, social media platforms, and subsidies for blacklisted enterprises.

Source: Information compiled from [www.chinacredit.gov.cn](http://www.chinacredit.gov.cn)

© MERICS

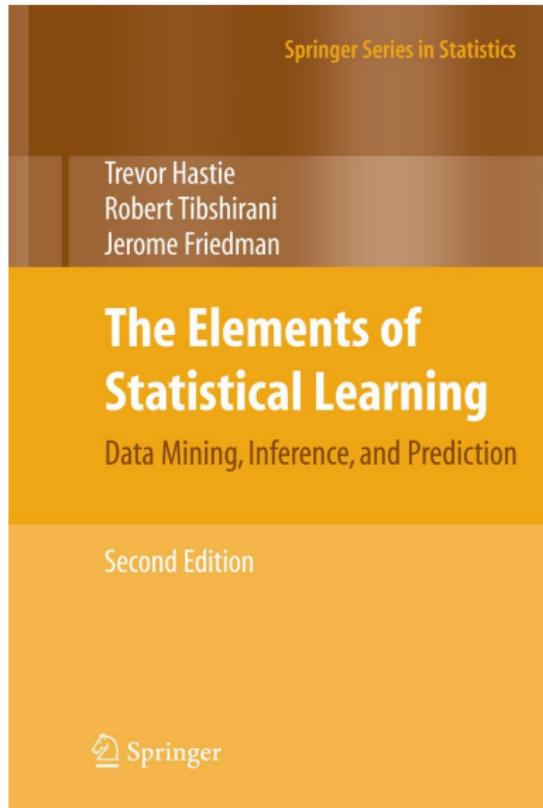
Well, it turns out the Chinese state is trying to develop something similar.  
(Source: MERICS.org)

# Books



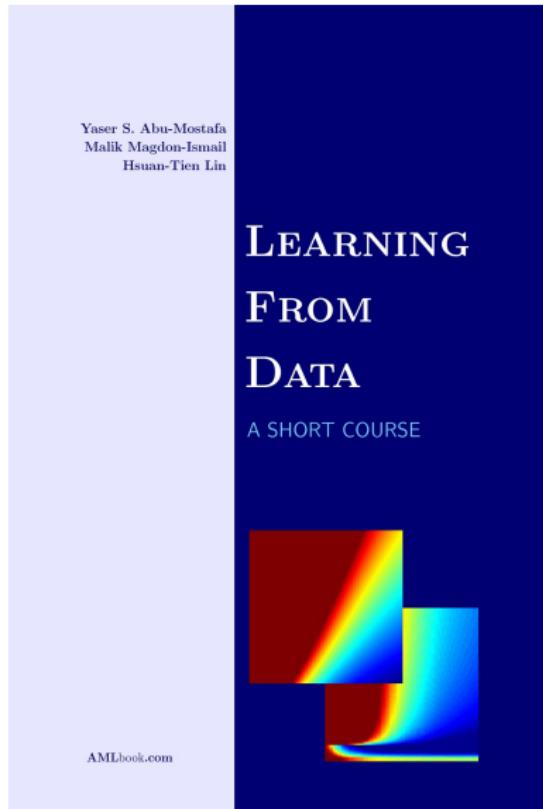
Download for free here

# Books



Download for free here

# Books



Download scan of ch. 1 here

What is machine learning?

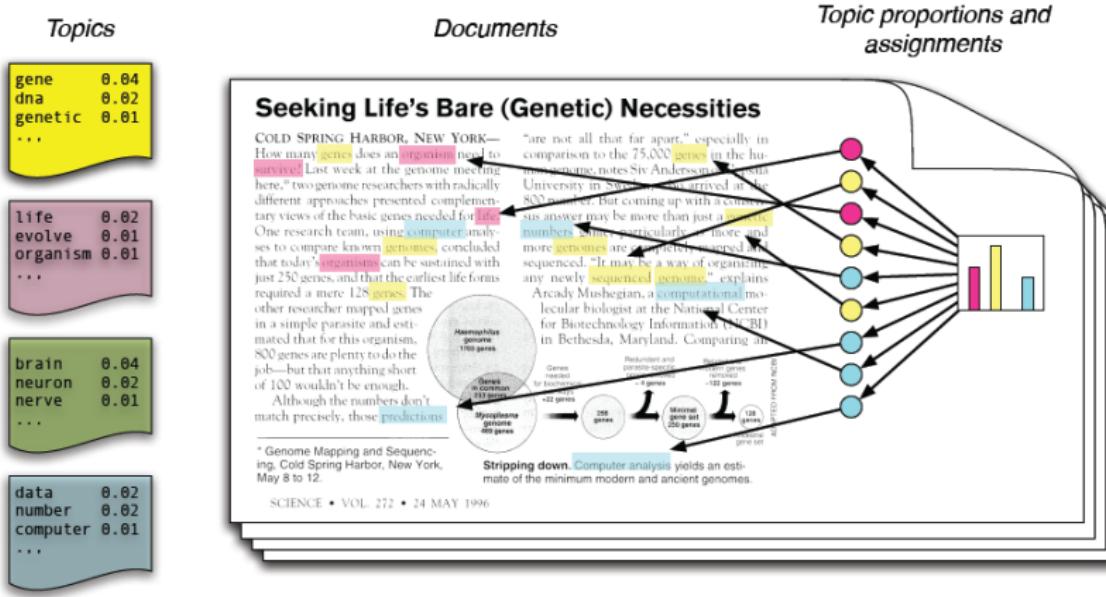
# Machine learning is many things

- ▶ A method of mining data for **patterns** or unobserved **latent variables**
- ▶ A method of **approximating unknown functions** using available data
- ▶ A predictive framework for forecasting future events
- ▶ A framework for **augmenting** and **automating** human decisions or tasks

# Unsupervised Learning

- ▶ **Objective:** Find patterns and structure among inputs  $\mathbf{x}$  in the data  $\mathcal{D}$ .
- ▶ Types of unsupervised learning:
  - ▶ **Clustering:** Given data  $\mathbf{X}$ , identify clusters of objects that balance within-cluster similarity and distinctness between clusters.
  - ▶ **Dimension Reduction:** Given data  $\mathbf{X}$ , identify manifolds or underlying factors that represent the data in fewer dimensions
- ▶ Training data does not contain any outputs  $\mathbf{y}$ . Structure is instead learned from underlying patterns and structure of the inputs  $\mathbf{x}$ .

## Unsupervised Example: Topic Modeling

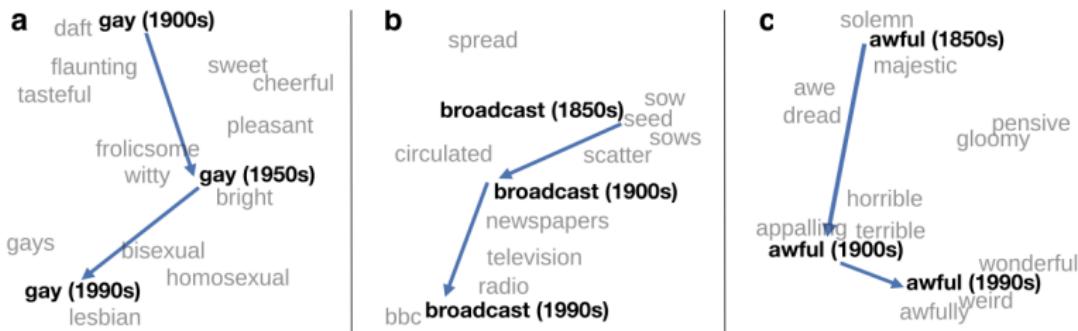


Source: David Blei

## Unsupervised Example: Topic Modeling

- ▶ Data: a corpus of documents (e.g. all New York Times articles from 2020)
- ▶ Each document is represented as a vector of counts of all words that appear in the corpus  $\mathbf{x}$  (features/inputs). This can be a vector of hundreds of thousands of words.
- ▶ There is no response/outcome  $\mathbf{y}$ , only inputs/features  $\mathbf{x}$ .
- ▶ Goal: build an algorithm to assign each document to a topic cluster (e.g. politics, business, finance, entertainment, etc.) using patterns in word use and word context across documents.

# Unsupervised Example: Word Embeddings



Source: HistWords Project

## Unsupervised Example: Word Embeddings

- ▶ Data: a corpus of documents (e.g. all New York Times articles from 2020)
- ▶ Words are represented as fixed-dimensional vectors.
- ▶ There is no response/outcome, only inputs/features (words, word sequences).
- ▶ Goal: build an algorithm to represent each word as a fixed-dimensional vector that captures semantic information
- ▶ Word embeddings quantify semantic similarities between words by learning from the contexts in which words appear; words with similar meaning will appear in similar contexts.

# Supervised Learning

- ▶ **Objective:** learn an approximation  $g(x)$  of an unknown **target function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  from data  $\mathcal{D}$ 
  - ▶  $\mathcal{X}$  represents the input space of all possible inputs  $x$  (also called predictors, regressors, covariates, features, independent variables)
  - ▶  $\mathcal{Y}$  represents the output space of all possible outputs  $y$  (also called dependent variable, response, target, outcome).
- ▶ **Regression:**  $y$  is quantitative (e.g GDP, unemployment measures, vote counts, # of COVID cases)
- ▶ **Classification:**  $y$  is a value in a finite, unordered set (e.g.  $y_i \in \{\text{ordinary, bot, troll, cyborg}\}$ )
- ▶ **Data:**  $\mathcal{D}$ , a set of input-output examples  $\{(x_1, y_1) \dots (x_N, y_N)\}$ .

## Supervised Example: Optical Character Recognition (OCR)

The image shows a large, stylized letter 'A' composed of a grid of binary code characters (0s and 1s). The letter is oriented vertically, with its top horizontal bar extending from approximately x=10 to x=90 at y=10, and its main body from y=10 to y=90. The letter is filled with a repeating pattern of binary digits, creating a pixelated or digital texture. The background is white, and the binary digits are black.

## Supervised Example: Optical Character Recognition (OCR)

- ▶ Data: images of single handwritten letters and digits
- ▶ Each image is  $20 \times 16$  pixels, with pixel intensities from 0 to 255. This vector of 320 quantitative variables is  $\mathbf{x}$  (features/inputs).
- ▶ Response/outcome: the identity of each image  $y_i \in \{A, B, \dots, Z, 0, 1, \dots, 9\}$ . This categorical variable with 36 levels is  $\mathbf{y}$ .
- ▶ Goal: build an algorithm (classifier, learner) to predict the identity  $\mathbf{y}$  from pixel values  $\mathbf{x}$  using a dataset of labelled images

# Supervised Example: Hate Speech Detection

PRODUCT AND EXPERIMENTS



**Perspective API**

Perspective API uses machine learning to identify toxic language, making it easier to host better conversations online. Perspective is used to give feedback to commenters, help moderators more easily review comments, and keep conversations open online.

[EXPLORE ↗](#)

**Tune**

A Chrome extension that helps people adjust the level of toxicity they see in comments across the internet.

[EXPLORE ↗](#)

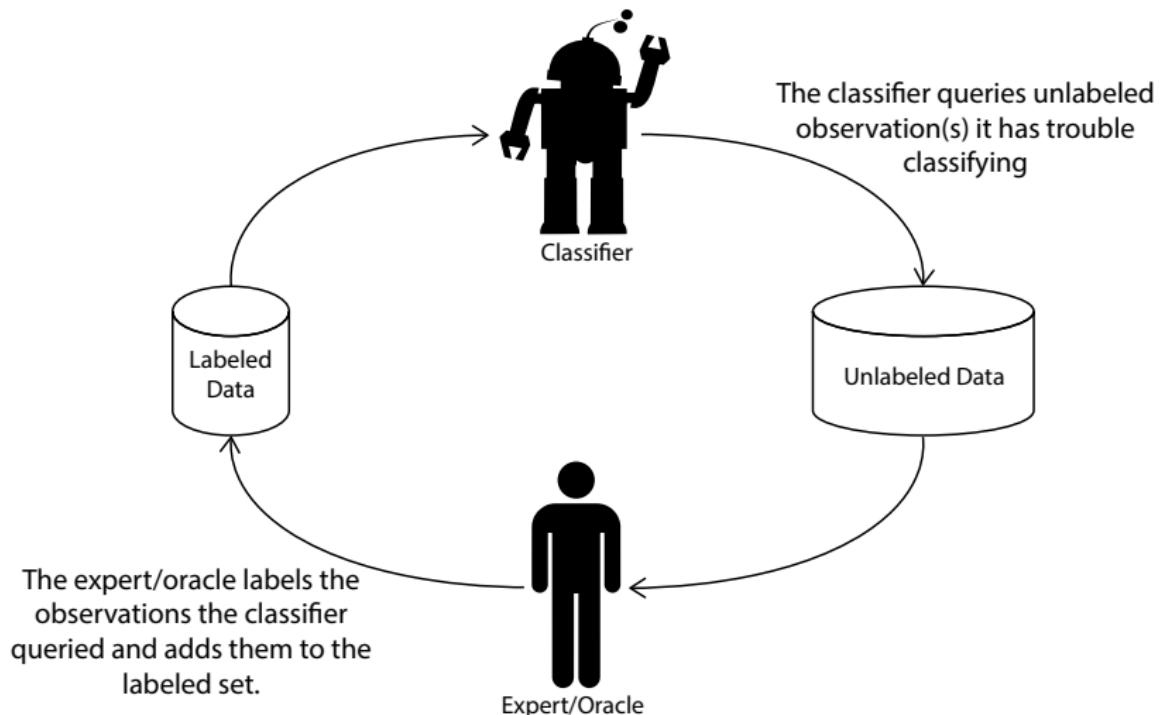
**Moderator**

An open-source tool that uses machine learning to help moderators identify and reduce toxicity in forums and comment sections.

[EXPLORE ON GITHUB ↗](#)

Source: Google/Jigsaw

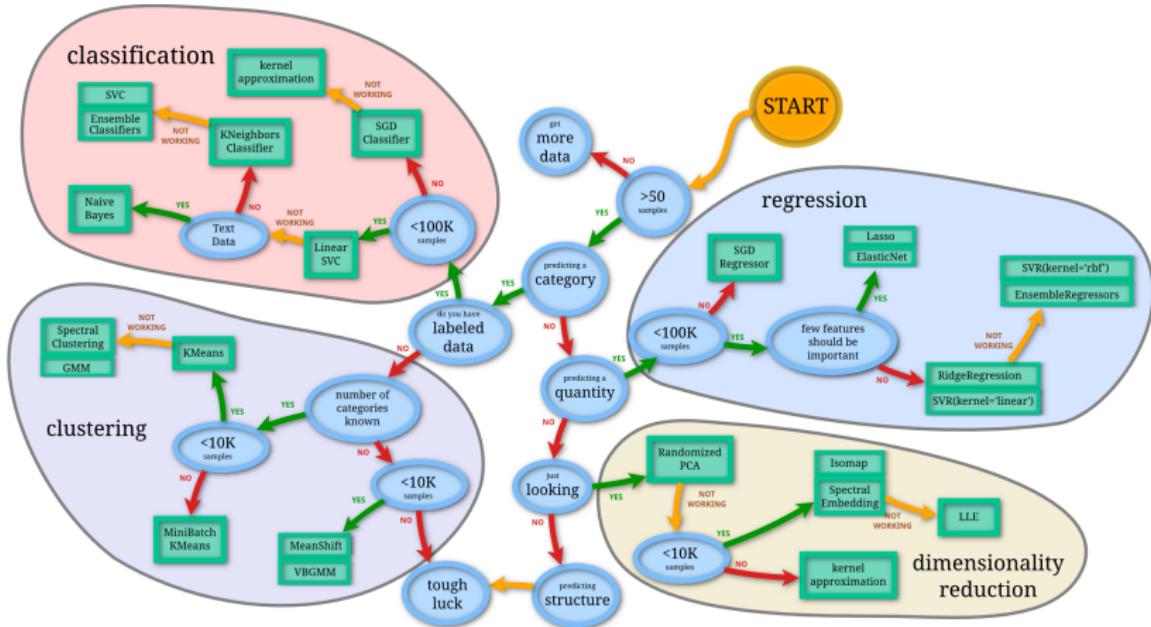
# Active Learning



# Reinforcement Learning

Source: DeepMind

## A Cheat Sheet



Source: scikit-learn

## A real world example

Which of these Wikipedia Talk Page comments are insults?

The screenshot shows a Wikipedia Talk page for the article "Coronavirus disease 2019". At the top, there is a blue navigation bar with two tabs: "Article" and "Talk". A large blue arrow points from the text below to the "Talk" tab. On the left side of the page, the Wikipedia logo and the text "WIKIPEDIA The Free Encyclopedia" are visible. The main content area has a light gray background and displays the title "Coronavirus disease 2019" in bold black text, followed by the subtitle "From Wikipedia, the free encyclopedia".

1. "I HATE your freakin guts!!!"
2. "YOU ARE AN IDIOT!!!!"
3. "Before we can accept this change, we need a citation for this second claim."
4. "I'm afraid I reverted your change. You erased all of the existing discussion there; we want to keep that."
5. "STOP YOUR DAMN NONSENSE!!!! YOU HAVE DONE NOTHING TO IMPROVE THESE ARTICLES!!! ASSHOLE!!!"

## A real world example

Which of these Wikipedia Talk Page comments are insults?



**WIKIPEDIA**  
The Free Encyclopedia

Article

Talk



### Coronavirus disease 2019

From Wikipedia, the free encyclopedia

1. **"I HATE your freakin guts!!"**
2. **"YOU ARE AN IDIOT!!!!"**
3. "Before we can accept this change, we need a citation for this second claim."
4. "I'm afraid I reverted your change. You erased all of the existing discussion there; we want to keep that."
5. **"STOP YOUR DAMN NONSENSE!!!!!! YOU HAVE DONE NOTHING TO IMPROVE THESE ARTICLES!!! ASSHOLE!!!"**

# How did you solve this problem?



- ▶ What **features** of each comment were informative?
- ▶ Could we make this process more explicit?

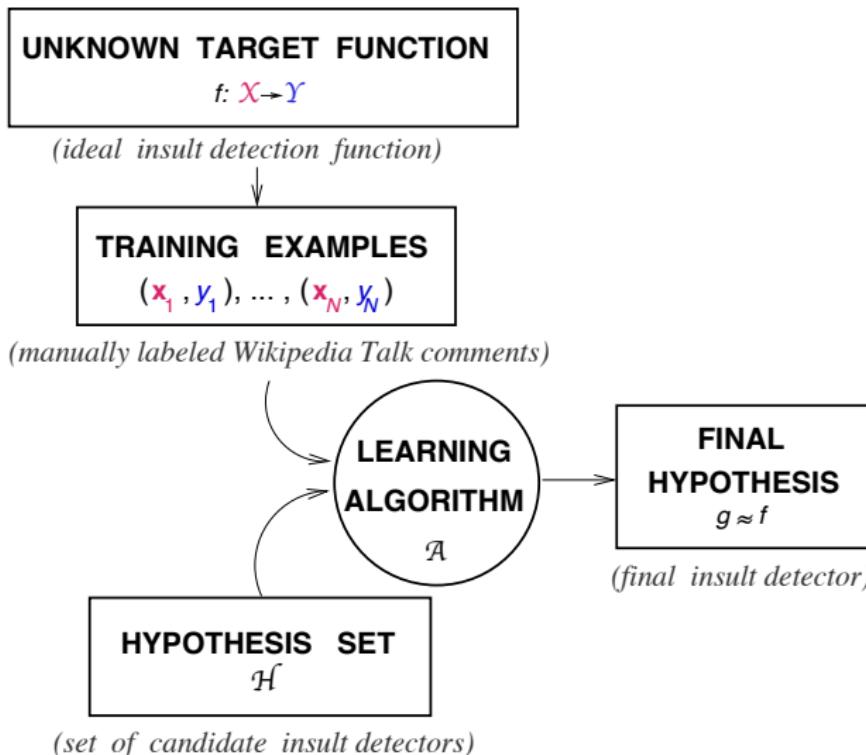
# Formalizing the Problem of Insult Classification

- ▶ Unknown **target function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  (What is an insult on Wikipedia Talk Pages)
  - ▶  $\mathcal{X}$  represents the input space of all possible inputs  $x$  (word/phrase/character/capital letter counts from Wikipedia Talk comments)
  - ▶  $\mathcal{Y} = \{\text{insult}, \neg\text{insult}\}$  represents the output space
- ▶ Data  $\mathcal{D}$ , a set of input-output examples  $\{(x_1, y_1) \dots (x_N, y_N)\}$ .

	"change"	"idiot"	"!"	# caps	$y_i$
Comment 1	0	0	2	5	insult
Comment 2	0	1	4	13	insult
Comment 3	1	0	0	1	$\neg\text{insult}$
Comment 4	1	0	0	2	$\neg\text{insult}$
Comment 5	0	0	10	60	insult

# Formalizing the Problem of Insult Classification

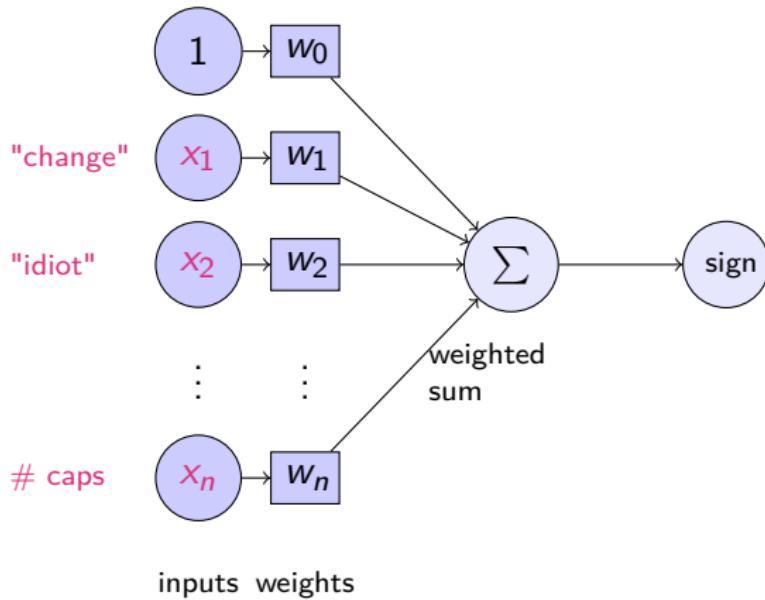
- ▶ Using the dataset  $\mathcal{D}$ , approximate  $f$  with  $g : \mathcal{X} \rightarrow \mathcal{Y}$
- ▶  $g$  is chosen from the **hypothesis set**  $\mathcal{H}$



## What is the hypothesis set $\mathcal{H}$

- ▶ All hypotheses  $h \in \mathcal{H}$  share the same functional form.
- ▶ The functional form is determined by the **learning model**
- ▶ The best hypothesis  $g \in \mathcal{H}$  is chosen by the **learning algorithm**
- ▶ In this lecture we will discuss two new learning models we could use to classify insults:
  1. **Perceptron**
  2. **K-Nearest Neighbors**

# The Perceptron Learning Model



- ▶ We operationalize  $\mathcal{Y} = \{\text{insult}, \neg\text{insult}\}$  as  $\mathcal{Y} = \{1, -1\}$
- ▶ Weights  $w_i$  correspond to each input  $x_i$  ( $w_0$  is the bias)
- ▶ The output of our learning model is  $h(\mathbf{x}) = \text{sign}(\mathbf{w}'\mathbf{x})$

# The Perceptron Learning Algorithm

---

## Algorithm 1: Perceptron Learning Algorithm

---

```
1  $P \leftarrow$  inputs with label 1;  
2  $N \leftarrow$  inputs with label -1;  
3 Initialize  $\mathbf{w}$  randomly;  
4 while any input is misclassified do  
5   Choose random  $\mathbf{x} \in P \cup N$ ;  
6   if  $\mathbf{x} \in P$  and  $\mathbf{w} \cdot \mathbf{x} < 0$  then  
7      $\mathbf{w} = \mathbf{w} + \mathbf{x}$ ;  
8   end  
9   if  $\mathbf{x} \in N$  and  $\mathbf{w} \cdot \mathbf{x} \geq 0$  then  
10     $\mathbf{w} = \mathbf{w} - \mathbf{x}$ ;  
11  end  
12 end
```

---

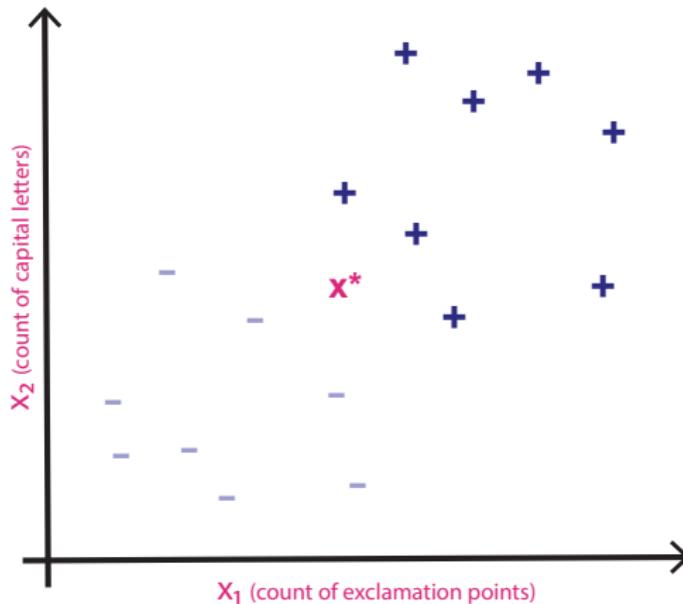
## K-Nearest Neighbors (KNN)

- ▶ The KNN learning model is the entire training dataset  $\mathcal{D}$ ; there is no learning required, just memorization
- ▶ Instead of learning weights or parameters, each  $x_i$  is loaded into memory
- ▶ KNN is a simple example of a **non-parametric model** where the model structure is determined from the dataset with no assumptions about the underlying data distribution.
- ▶ KNN does not work well in high dimensions unless data lie on or close to a low-dimensional subspace (manifold)
- ▶ The KNN learning model simply returns the majority class of the  $k$  nearest  $x_i$  in the dataset  $\mathcal{D}$ :

$$h(\mathbf{x}) = \text{sign} \left( \frac{1}{k} \sum_{x_i \in N_k(x^*)} y_i \right) \quad y_i \in \{-1, 1\}$$

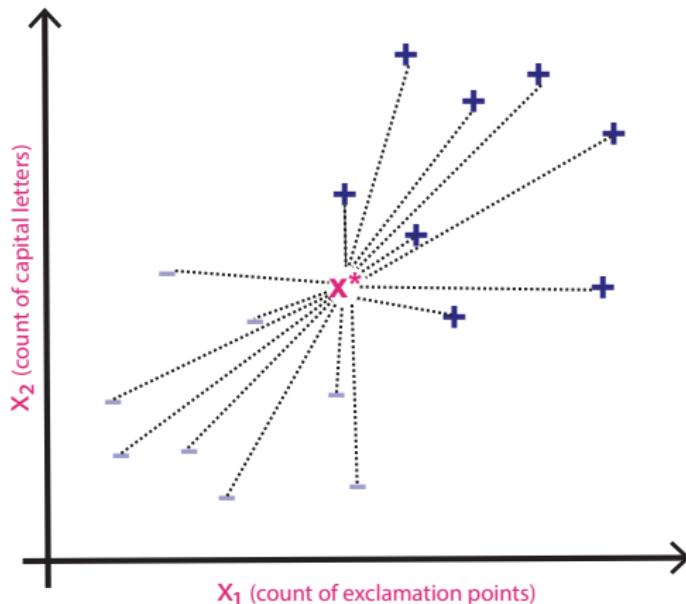
- ▶  $N_k(x^*)$  consists of the  $k$  closest points to reference point  $x^*$  in the training data ( $k$ -nearest neighborhood).

# K-Nearest Neighbors (KNN) Visualized



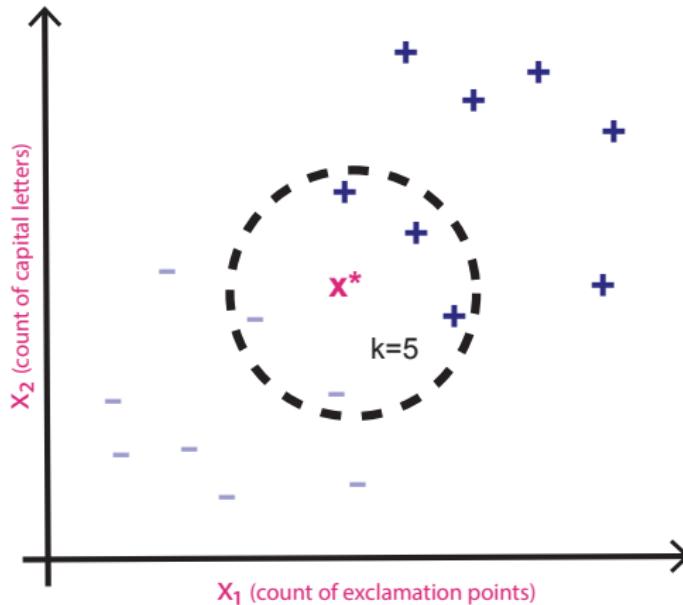
1. Calculate the distance between each  $x_i$  and reference point  $x^*$
2. Find the  $k$  closest neighbors to  $x^*$
3. Return the majority class of  $y_i \in N_k(x^*)$

# K-Nearest Neighbors (KNN) Visualized



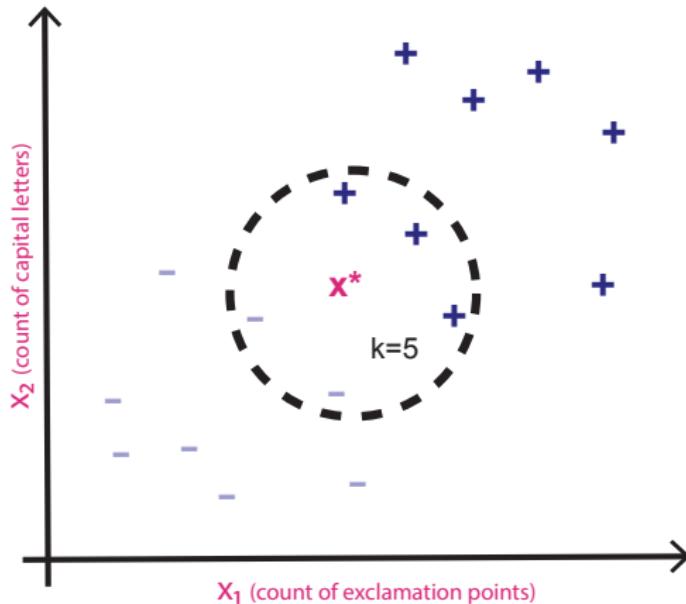
1. Calculate the distance between each  $x_i$  and reference point  $x^*$
2. Find the  $k$  closest neighbors to  $x^*$
3. Return the majority class of  $y_i \in N_k(x^*)$

# K-Nearest Neighbors (KNN) Visualized



1. Calculate the distance between each  $x_i$  and reference point  $x^*$
2. **Find the  $k$  closest neighbors to  $x^*$**
3. Return the majority class of  $y_i \in N_k(x^*)$

# K-Nearest Neighbors (KNN) Visualized



1. Calculate the distance between each  $x_i$  and reference point  $x^*$
2. Find the  $k$  closest neighbors to  $x^*$
3. **Return the majority class of  $y_i \in N_k(x^*)$** 
  - ▶ 3 votes **insult**, 2 votes **¬insult**
  - ▶ This means  $x^*$  is an **insult** according to KNN

## OLS Regression in a ML Framework

- ▶ In the case of **ordinary least squares (OLS) regression**, we assume  $f(\cdot)$  is linear on coefficients  $\beta$  and has an additive separable stochastic component  $\varepsilon$

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \quad \varepsilon \sim \mathcal{N}_n(0, \sigma^2 \mathbf{I}_n)$$

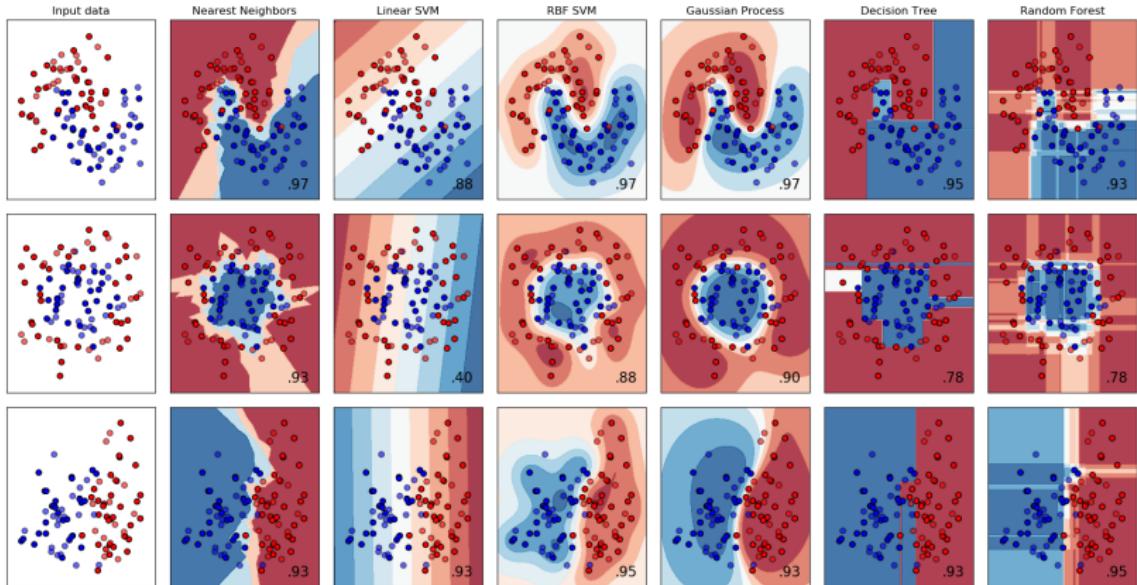
- ▶ For OLS, our learning model would have a hypothesis set containing all possible linear functions in the form

$$h(\mathbf{X}) = \mathbf{X}\hat{\beta}$$

- ▶ In this example  $g(\mathbf{x}) \in \mathcal{H}$  is usually identified using the least squares **learning algorithm** which estimates parameters  $\beta$  with  $\hat{\beta}$
- ▶  $g(\mathbf{x})$  is the best representation of the data in  $\mathcal{H}$  because it minimizes the *residual sum of squares*

Which model do we choose?

# Which model do we choose?

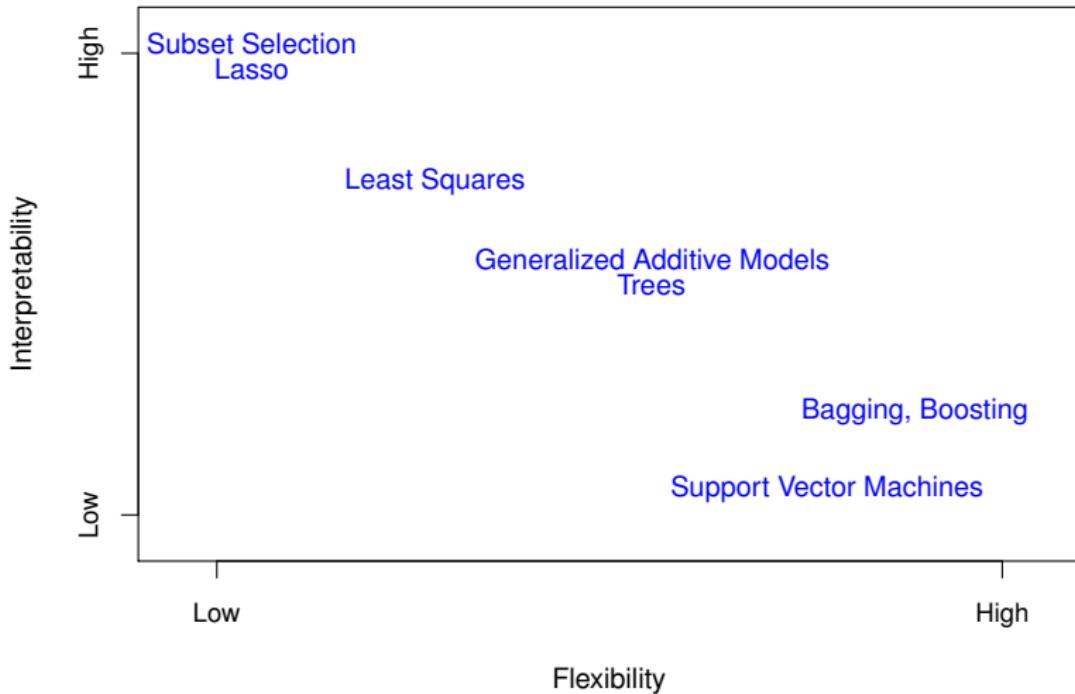


Which model works best really depends on the task and the data ("no free lunch"). (source: [sklearn.org](http://sklearn.org))

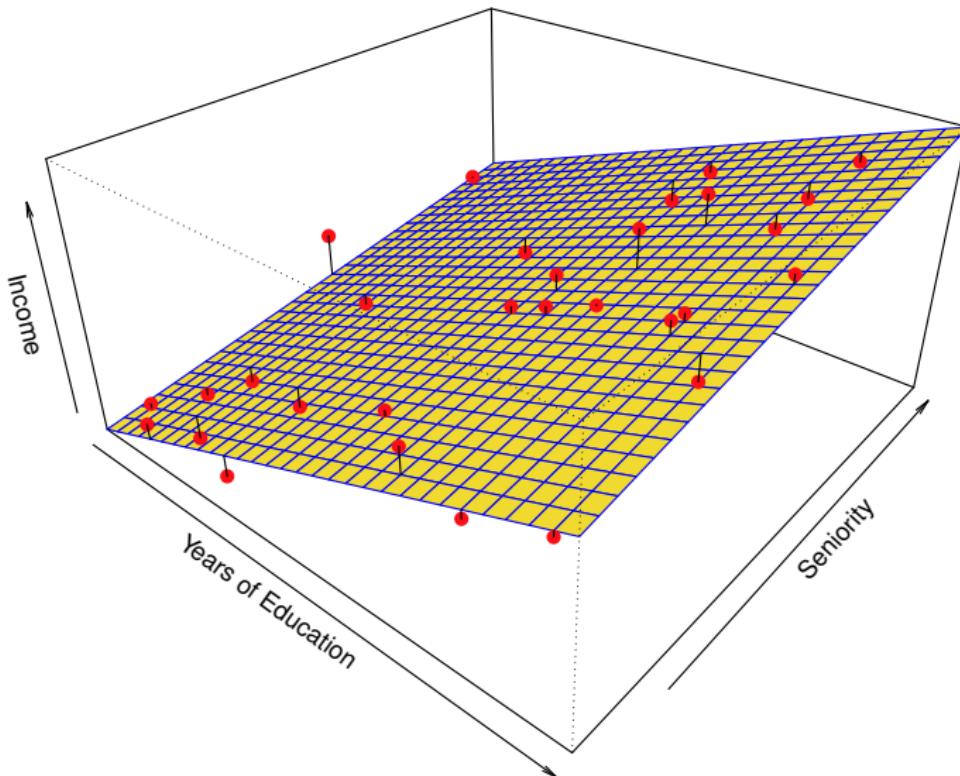
## Types of Models

- ▶ Machine learning models come in many forms, each differing in the assumptions it makes about  $f$
- ▶ **Parametric models** learn parameters that define the shape of  $f$  (e.g.  $\hat{\beta}$  in OLS regression)
- ▶ **Non-parametric models** make few assumptions about  $f$ , instead fitting to the data with a certain degree of “smoothness”
- ▶ **Interpretable** models allow us to easily dissect and understand how outputs are estimated
- ▶ **Black box** models may have millions of parameters or a complicated model structure that make it difficult to understand how outputs are estimated.

# The Trade-Off Between Prediction Accuracy and Model Interpretability



## Parametric models

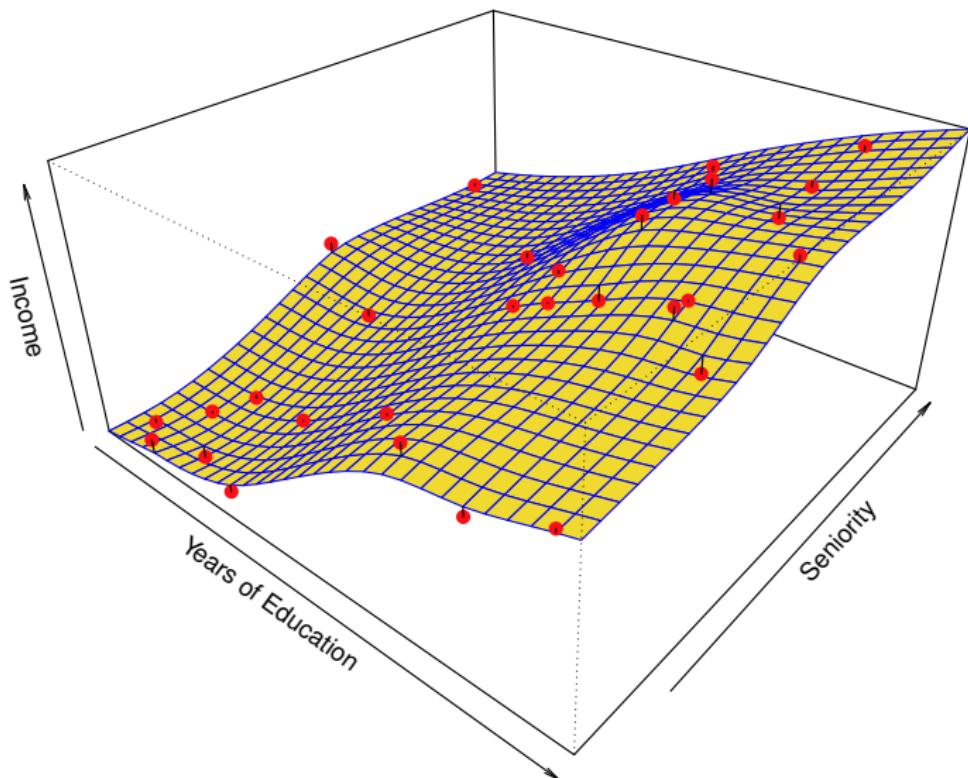


The number of parameters remains fixed, regardless of N.

## Parametric models

- ▶ Objective: Estimate parameters such as  $\hat{\beta}$  from training data
- ▶ Examples: OLS, discriminant analysis, logistic regression, neural networks
- ▶ Assumptions about the functional form (i.e. function is linear)
- ▶ Pros: More interpretable, faster, usually better for inference, require less data
- ▶ Cons: Unlikely to closely match the underlying function

## Non-parametric models



The number of “parameters” increases as N increases.

## Non-parametric models

- ▶ Objective: Get as close as possible to the data
- ▶ Examples: decision trees, KNN, splines
- ▶ Fewer assumptions about the functional form
- ▶ Pros: Fewer assumptions about the underlying functional form, sometimes more accurate
- ▶ Cons: slow, prone to **overfitting**, **curse of dimensionality**, require a lot more data

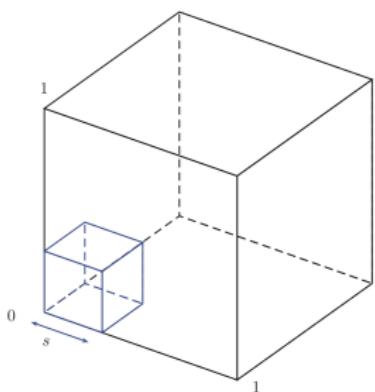
## The Curse of Dimensionality

- ▶ Non-parametric models such as KNN often fail in high dimensions
- ▶ Reason: it is difficult to find enough observations close to a target point  $x$
- ▶ If we increase  $h$  (or number of nearest neighbors  $k$ ), the estimates become severely biased

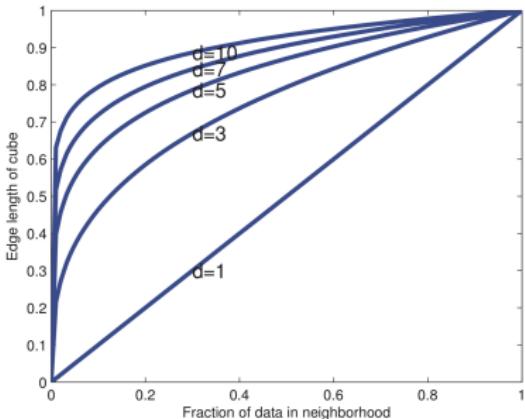
## Where does the “curse” come from?

- ▶ Suppose points  $x_1, \dots, x_n$  are uniformly distributed in a  $d$ -dimensional unit cube
- ▶ We wish to construct a hypercube neighborhood of  $x$  to capture a fraction  $\rho$  of the observations. How big should the cube be? Since the volume is  $\ell^d = \rho$ , we need edge length  $\ell^d = \rho^{1/d}$
- ▶ For example:
  - ▶ When  $d = 1$ : If  $\rho = 0.01$ ,  $\ell = 0.01$  and  $\rho = 0.1$ ,  $\ell = 0.1$ .
  - ▶ When  $d = 10$ : If  $\rho = 0.01$ ,  $\ell = 0.63$  and  $\rho = 0.1$ ,  $\ell = 0.8$ .

# The Unit Cube Illustration



(a)



(b)

**Figure 1.16** Illustration of the curse of dimensionality. (a) We embed a small cube of side  $s$  inside a larger unit cube. (b) We plot the edge length of a cube needed to cover a given volume of the unit cube as a function of the number of dimensions. Based on Figure 2.6 from (Hastie et al. 2009). Figure generated by `curseDimensionality`.

Source: *Machine Learning: A Probabilistic Perspective*

## Implications of the curse of dimensionality

- ▶ “Local” methods are no longer local when the dimension  $p$  increases.
- ▶ If  $n$  points are sufficient to estimate a function in  $\mathbb{R}^1$ ,  $n^p$  are needed to achieve similar accuracy in  $\mathbb{R}^p$ .
- ▶ Fortunately, in practice most data are not sampled uniformly from a cube (structure effectively reduces dimension), so the “curse” is not as pronounced as it is in the cube illustration.

## Next Lecture: Generalization, Inference, Prediction, and Causality

What if we gathered some more Wikipedia Talk comments? How well would the models trained using our initial dataset **generalize** to new data?

1. “You are an old cougar! You are an old cougar!”
2. “You’re full of crap. .... and you know it.”
3. “In this case there’s been little editing activity so it will probably work.”
4. “THANK YOU!!! This section looks great now! The last paragraph is missing a few citations, though.”