# 6.036 Introduction to Machine Learning

## (meets with 6.862)

# Administrivia

**HW 1-2 due tomorrow Friday 2/24 @ 9AM.**

**Recitations** (starting this Friday):

11 am - 12 pm: room **54-100**
12 pm - 1 pm: room **54-100**
1 pm - 2 pm: room **54-100**

2 pm - 3 pm: room **1-190**
3 pm - 4 pm: room **1-190**
4 pm - 5 pm: room **1-190**

**As always:**

- Check LMOD/Piazza for announcements.
- To contact staff, use Piazza (6036-staff@lists.csail.mit.edu for exceptions only)

# Last time: binary classification

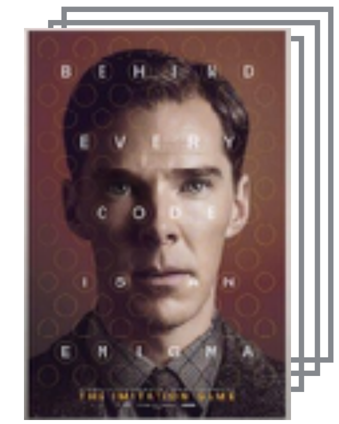
- Learn to predict **binary labels**

Training set



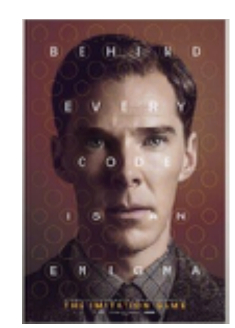


Test set



| $x^{(1)}$ | $x^{(2)}$ | $x^{(3)}$ | $x^{(4)}$ | $x^{(5)}, x^{(6)}, \ldots$ |
|---|---|---|---|---|
| -1 | -1 | +1 | +1 | ?, ?, .... |

$$h : \mathcal{X} \to \{-1, +1\}$$

$h\Big($  $\Big) = ?$

# Supervised learning +

- Multi-way classification (e.g., three-way classification)

$h\Bigg($  $\Bigg) = \text{politics}$ $\qquad h : \mathcal{X} \to \{\text{politics}, \text{sports}, \text{other}\}$

- Structured prediction

$h\Bigg($  $\Bigg) =$ A group of people shopping at an outdoor market $\qquad h : \mathcal{X} \to \{\text{English sentences}\}$

- **Regression**

$h\Bigg($  $\Bigg) = \$1{,}349{,}000$ $\qquad h : \mathcal{X} \to \mathbb{R}$

# Linear regression

- Predictor is a **linear function** of the **feature vectors**

$$f(x;\theta,\theta_0) = \theta \cdot x + \theta_0 = \sum_{i=1}^{d} \theta_i x_i + \theta_0$$

- *Feature vector x* is *d*-dimensional (and therefore, so is the *parameter* θ)

- For every choice of parameters, a different function *f*

- For now, assume that features are *given*
  In practice, choosing "good" features is extremely important

- For simplicity, we'll often assume $\theta_0$=0 (wlog).

# Example 1: salary forecast

**Task:** Predict starting salary of MIT SB graduates

Features: {GPA, #units, #courses, #terms in residence, #internships, #UROPS, major, #math courses}

Student-1: {3.9, 200, 22, 8, 2, 1, 6, 5} -> 107K

Student-2: {3.8, 212, 24, 9, 0, 2, 1, 3} -> 74K

Student-n: {4.5, 220, 21, 10, 1, 1, 2, 2} -> 82K

Other features?
Height? Gender? #FB friends? 6.036 grade?....

# Example 2: life expectancy

**Task:** Predict life expectancy of an individual

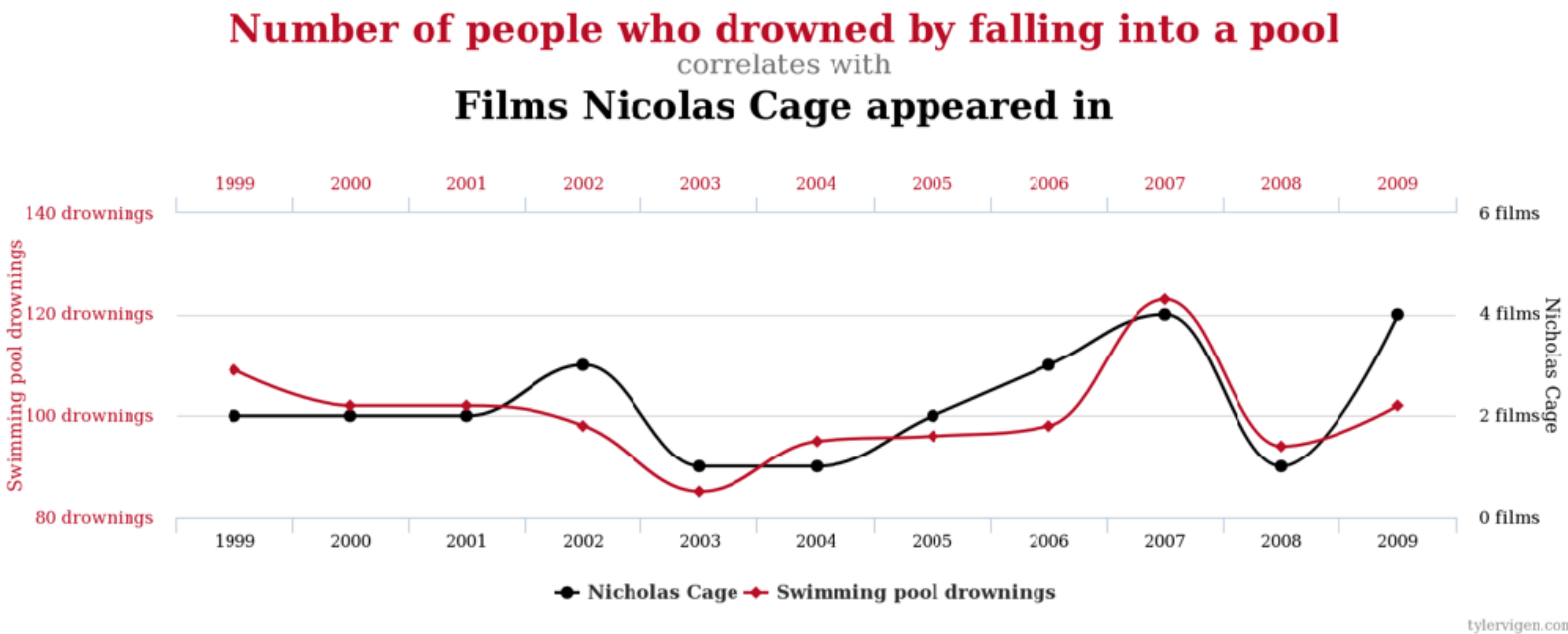
Features: {current age, weight, height, annual income, alcohol consumption, sugar consumption, access to healthcare, education, pollution, longevity parents,…}

Typically, the more features we use, the better we can predict. But…

# Correlation, not causation!

Pitfall! Regression only yields *statistical prediction*.

In particular, we *cannot* deduce a *causal relationship*.




http://www.tylervigen.com/spurious-correlations

# Questions

- How to measure **error**? How to choose $\theta$ and $\theta_0$?
  - What criteria should we use?
- Which **algorithms** to minimize training error?
  - How do they scale with dimension, problem size?
- How to ensure **generalization**?
  - What if we have too many parameters?
  - How to constrain the set of hypotheses (functions)?

# Empirical risk

- We measure error in terms of *empirical risk*

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n} \text{Loss}(y^{(t)} - \theta \cdot x^{(t)})$$

- Average prediction error on training set (measured according to a given loss function).

- Many possible loss functions.
  For now, simple *squared error*

$$\text{Loss}(z) = z^2/2$$

(motivation: small errors ok, large errors costly).

# Least squares criterion

- Putting this together: *least squares criterion*

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n} \mathrm{Loss}(y^{(t)} - \theta \cdot x^{(t)})$$
$$= \frac{1}{n}\sum_{t=1}^{n} (y^{(t)} - \theta \cdot x^{(t)})^2/2$$

- We'll choose θ to minimize $R_n(\theta)$ — "fitting". This depends *only* on the training set.

- But, keep in mind we're actually interested in *generalization error*

$$R_{n'}^{\text{test}}(\theta) = \frac{1}{n'}\sum_{t=n+1}^{n+n'} (y^{(t)} - \theta \cdot x^{(t)})^2/2$$

# Empirical risk vs. generalization

**Q:** How are *empirical risk* and *generalization error* related?

- When is *generalization error* large?
  - **Estimation error:** Bad parameter estimates, due to noisy or insufficient data (even if true relationship is linear)
  - **Structural error:** True underlying relationship is nonlinear (incorrect model class)
- Tradeoffs! Want powerful models (many parameters), but then it is hard to estimate them :(
- In statistical setting, related to *bias/variance tradeoff*
  More about this later

# Demo!

# Minimizing least-squares

A few different approaches to minimize empirical risk:

- General optimization methods (gradient descent)
- Closed form solutions (linear algebra, matrix inversion)

# (Stochastic) gradient descent

- To minimize a function *f(θ)*, can use **gradient descent**

$$\theta^{(k+1)} = \theta^{(k)} - \eta_k \nabla f(\theta^{(k)})$$

- But, there's a special feature!
  objective $R_n(\theta)$ is *sum of functions, one per data point*.

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot x^{(t)})^2/2$$

- Natural algorithm for this: **stochastic gradient**

$$\begin{aligned}
&\text{set } \theta^{(0)} = 0\\
&\quad \text{randomly select } t \in \{1,\ldots,n\}\\
&\qquad \theta^{(k+1)} = \theta^{(k)} + \eta_k (y^{(t)} - \theta \cdot x^{(t)}) x^{(t)}
\end{aligned}$$

# Closed form - Linear algebra

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot x^{(t)})^2/2$$

- Since cost is quadratic, can also solve in *closed form.*

- Computing the gradient, we have

$$\nabla R_n(\theta) = A\theta - b, \qquad \text{where} \quad A = \frac{1}{n}\sum_{t=1}^{n} x^{(t)}(x^{(t)})^T, \quad b = \frac{1}{n}\sum_{t=1}^{n} y^{(t)}x^{(t)}$$

and thus (if A is invertible):

$$\hat{\theta} = A^{-1}b$$

# Back to generalization

- Recall ML goal is to reduce **generalization (test) error**
  Instead, we minimized **empirical risk**

- Already discussed some issues (estimation error, structural error). But there's more:

- What if there is not enough training data to estimate all parameters (i.e., matrix $A$ is not invertible)?

- If there are many "good" models, how to pick the "simplest" one? (*Occam's razor* principle)

# Regularization

- Solution: add a *regularization term*

- Penalty term, to avoid large values of parameters

Regularization

$$J_{n,\lambda}(\theta) = R_n(\theta) + \frac{\lambda}{2}\|\theta\|^2$$
$$= \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot x^{(t)})^2/2 + \frac{\lambda}{2}\|\theta\|^2$$

- Many names ("ridge regression", "Tikhonov regularization")

- What is the role of *regularization parameter* λ?
  What happens for very small, or very large values of λ?

# Regularization

- Still a quadratic function of parameters

$$
\begin{aligned}
J_{n,\lambda}(\theta) &= R_n(\theta) + \frac{\lambda}{2}\|\theta\|^2 \\
&= \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot x^{(t)})^2/2 + \frac{\lambda}{2}\|\theta\|^2
\end{aligned}
$$

- Parameter λ quantifies the tradeoff between *fitting the data* and *keeping parameters small*.

- Purpose is to bias parameters towards zero (or some other value), even if weakly contradicts training data

- In the absence of strong evidence, choose simplest answer

# Demo!

# Regularization and algorithms

Algorithms can be easily modified to take penalty term into account:

- For stochastic gradient:

$$
\begin{aligned}
&\text{set } \theta^{(0)} = 0 \\
&\quad \text{randomly select } t \in \{1, \ldots, n\} \\
&\qquad \theta^{(k+1)} = (1 - \lambda\eta_k)\theta^{(k)} + \eta_k(y^{(t)} - \theta \cdot x^{(t)})x^{(t)}
\end{aligned}
$$

- For closed-form solution:

$$\hat{\theta} = \left(\frac{1}{n}X^T X + \lambda I\right)^{-1} \left(\frac{1}{n}X^T y\right)$$

# Summary - Linear regression

- Predictor is a *linear function* of feature vectors.
- Empirical risk $R_n(\theta)$ is a quadratic function of parameters $\theta$
- Tradeoff between “fitting” and “generalization”
- Minimize risk $R_n(\theta)$ using closed-form, or stochastic gradient
- For good generalization, often need regularization term.
- Regularization parameter $\lambda$ quantifies tradeoffs.